TMS570LC43x 16/32-Bit RISC Flash Microcontroller

Technical Reference Manual

Literature Number: SPNU563A
March 2018

Contents

List of Figures

List of Tables

Read This First

About This Manual

This Technical Reference Manual (TRM) details the integration, the environment, the functional description, and the programming models for each peripheral and subsystem in the device.

The TRM should not be considered a substitute for the data manual, rather a companion guide that should be used alongside the device-specific data manual to understand the details to program the device. The primary purpose of the TRM is to abstract the programming details of the device from the data manual. This allows the data manual to outline the high-level features of the device without unnecessary information about register descriptions or programming models.

Notational Conventions

This document uses the following conventions.

- Hexadecimal numbers may be shown with the suffix h or the prefix 0x. For example, the following number is 40 hexadecimal (decimal 64): 40h or 0x40.
- Registers in this document are shown in figures and described in tables.
  - Each register figure shows a rectangle divided into fields that represent the fields of the register. Each field is labeled with its bit name, its beginning and ending bit numbers above, and its read/write properties with default reset value below. A legend explains the notation used for the properties.
  - Reserved bits in a register figure can have one of multiple meanings:
    - Not implemented on the device
    - Reserved for future device expansion
    - Reserved for TI testing
    - Reserved configurations of the device that are not supported
  - Writing nondefault values to the Reserved bits could cause unexpected behavior and should be avoided.

Glossary

TI Glossary — This glossary lists and explains terms, acronyms, and definitions.

Related Documentation From Texas Instruments

For product information, visit the Texas Instruments website at http://www.ti.com.

SPNS195— *TMS570LC4357 16- and 32-Bit RISC Flash Microcontroller Data Manual.*

SPNU540— *Safety Manual for TMS570LC4x Hercules™ ARM® Safety Critical Microcontrollers User's Guide.* A safety manual for the Texas Instruments Hercules safety critical microcontroller product family. The product family utilizes a common safety architecture that is implemented in multiple application focused products.

SPNU597— *TMS570LC43x Hercules™ Development Kit (HDK) User's Guide.* Describes the board level operations of the TMS570LC43 Hercules Development Kit (HDK). The HDK is based on the Texas Instruments TMS570LC4357 Microcontroller. The TMS570LC43 HDK is a table top card that allows engineers and software developers to evaluate certain characteristics of the TMS570LC4357 microcontroller to determine if the microcontroller meets the designer’s application requirements as well as begin early application development. Evaluators can create software to execute on board or expand the system in a variety of ways.

Community Resources

The following links connect to TI community resources. Linked contents are provided "AS IS" by the respective contributors. They do not constitute TI specifications and do not necessarily reflect TI's views; see TI's Terms of Use.

TI E2E™ Online Community— *TI's Engineer-to-Engineer (E2E) Community.* Created to foster collaboration among engineers. At e2e.ti.com, you can ask questions, share knowledge, explore ideas and help solve problems with fellow engineers.

TI Embedded Processors Wiki— *Texas Instruments Embedded Processors Wiki.* Established to help developers get started with Embedded Processors from Texas Instruments and to foster innovation and growth of general knowledge about the hardware and software surrounding these devices.

Trademarks

Hercules, E2E are trademarks of Texas Instruments.
CoreSight is a trademark of ARM Limited.
ARM, Cortex are registered trademarks of ARM Limited.

Chapter 1
SPNU563A – March 2018

Introduction

1.1 Designed for Safety Applications

The TMS570LC43x device architecture has been designed from the ground up to simplify development of functionally safe systems. The basic architectural concept is known as a safe island approach. Power, clock, reset, and basic processing function are protected to a high level of diagnostic coverage in hardware. Some of the key features of the safe island region are:

- Lockstep safety concept is also extended to the Vector Interrupt Module (VIM). Dual VIMs in lockstep that detect failures at the controller's boundary on a cycle by cycle basis. VIMs internal RAM that stores the vector addresses is also ECC protected.
- ECC diagnostic for the datapath on the Level 1 cache memories as well as ECC on the Level 2 SRAM and flash memories of the R5F core. The ECC controllers are located inside the CPU for each respective memory interface. This approach has two key advantages:
  - The interconnect between CPU and the memory is also covered by the diagnostic.
  - The ECC logic itself is checked on a cycle by cycle basis.
- Hardware BIST controllers that provide an extremely high level of diagnostic coverage for the lockstep CPUs and SRAMs in the system, while executing faster and consuming less memory than equivalent software-based self-test solutions.
- Hardware BIST diagnostic also for both the N2HET timer coprocessors.
- Interconnect between the masters and the level 2 memories contain built-in hardware safety diagnostic logic that monitors the integrity of traffics in each cycle
  - Continuous monitoring of transactions going in and out of the interconnect.
  - Parity diagnostic on the address and control paths between all masters and slaves
  - BIST mode for diagnostic coverage of the interconnect.
  - ECC generation and evaluation for transactions on the datapath generated for some of the bus masters.
- Onboard voltage and reset monitoring logic
- Onboard oscillator and PLL failure detection logic including a backup RC oscillator that can be utilized upon failure

The TMS570LC43x device architecture also includes many features to simplify diagnostics of remaining logic such as:

- Continuous parity or ECC diagnostics on all peripheral memories.
- Analog and digital loopback to test for shorts on I/O.
- HW self-test and diagnostics on the ADC module to check integrity of both analog inputs and the ADC core conversion function.
- A DMA driven hardware engine for the background calculation of CRC signatures during data transfers.
- A centralized error reporting function including a status output pin to enable external monitoring of the device status.

1.2 Family Description

The TMS570LC43x family of microcontrollers are cache-based architecture based on the ARM® Cortex®-R5F Floating Point CPU that offers an efficient 1.66 DMIPS/MHz performance and has configurations that can run up to 330 MHz providing up to 498 DMIPS. The device supports the big-endian [BE32] format.

The TMS570LC43x has up to 4MB integrated Flash and up to 512KB data RAM configurations with single bit error correction and double bit error detection. The flash memory on this device is a nonvolatile, electrically erasable and programmable memory implemented with a 64-bit-wide data bus interface. The flash operates on a 3.3V supply input (same level as I/O supply) for all read, program and erase operations. The SRAM operates with a system clock frequency of up to 150 MHz. The SRAM supports read/write accesses in byte, halfword, and word modes.

The TMS570LC43x device features peripherals for real-time control-based applications, including two Next Generation High End Timer (N2HET) timing coprocessors with up to 64 total IO terminals and two 12-bit A to D converters supporting up to 41 inputs.

The N2HET is an advanced intelligent timer that provides sophisticated timing functions for real-time applications. The timer is software-controlled, using a reduced instruction set, with a specialized timer micromachine and an attached I/O port. The N2HET can be used for pulse width modulated outputs, capture or compare inputs, or general-purpose I/O. It is especially well suited for applications requiring multiple sensor information and drive actuators with complex and accurate time pulses. A High End Timer Transfer Unit (HET-TU) can perform DMA type transactions to transfer N2HET data to or from main memory. A Memory Protection Unit (MPU) is built into the HET-TU.

The device has two 12-bit-resolution MibADCs with 41 total channels and 64 words of parity protected buffer RAM each. The MibADC channels can be converted individually or can be grouped by software for sequential conversion sequences. Sixteen channels are shared between the two MibADCs. There are three separate groupings. Each sequence can be converted once when triggered or configured for continuous conversion mode.

There are three on-die temperature sensors on this device. The temperature measurements of the three temperature sensors are routed to the MibADC for conversion into digital values. CPU can read out the digital values and compare with the calibrated temperature value stored in the device's OTP.

The device has multiple communication interfaces: Five MibSPIs, two LINs, two SCIs, four DCANs, two I^2C, one Ethernet, and one FlexRay controller. The MibSPI provides a convenient method of serial interaction for high-speed communications between similar shift-register type devices. Data stored in the MibSPI's buffer RAM are protected with ECC. The LIN supports the Local Interconnect standard 2.0 and can be used as a UART in full-duplex mode using the standard Non-Return-to-Zero (NRZ) format. The DCAN supports the CAN 2.0B protocol standard and uses a serial, multimaster communication protocol that efficiently supports distributed real-time control with robust communication rates of up to 1 megabit per second (Mbps). The DCAN is ideal for applications operating in noisy and harsh environments (for example, automotive and industrial fields) that require reliable serial communication or multiplexed wiring. Messages stored at the DCAN's RAM are protected with ECC. The FlexRay uses a dual channel serial, fixed time base multimaster communication protocol with communication rates of 10 megabits per second (Mbps) per channel. Messages stored at the FlexRay's RAM are ECC protected. A FlexRay Transfer Unit (FTU) enables autonomous transfers of FlexRay data to and from main CPU memory. Transfers are protected by a dedicated, built-in Memory Protection Unit (MPU). The Ethernet module supports MII and MDIO interfaces. Transfers are protected by a standalone Enhanced Memory Protection Unit (NMPU)

The I2C module is a multi-master communication module providing an interface between the microcontroller and an I2C compatible device via the I2C serial bus. The I2C supports both 100 Kbps and 400 Kbps speeds.

The frequency-modulated phase-locked loop (FMPLL) clock module is used to multiply the external frequency reference to a higher frequency for internal use. The FMPLL provides one of the seven possible clock source inputs to the global clock module (GCM). The GCM module manages the mapping between the available clock sources and the device clock domains.

The device also has two external clock prescaler (ECP) modules that when enabled, outputs a continuous external clock on the ECLK1 and ECLK2 terminals. The ECLK frequency is a user-programmable ratio of the peripheral interface clock (VCLK) frequency. This low frequency output can be monitored externally as an indicator of the device operating frequency.

The Direct Memory Access Controller (DMA) is capable of issuing multi-threaded transactions at the same time. It can support up to 48 DMA requests that can be mapped to any one of the 32 channels. 32 control packets implemented in RAM to store channel control information are ECC protected. A first level Memory Protection Unit (MPU) is built into the DMA and a second level standalone Enhanced Memory Protection Unit (NMPU) further protect memory against erroneous transfers.

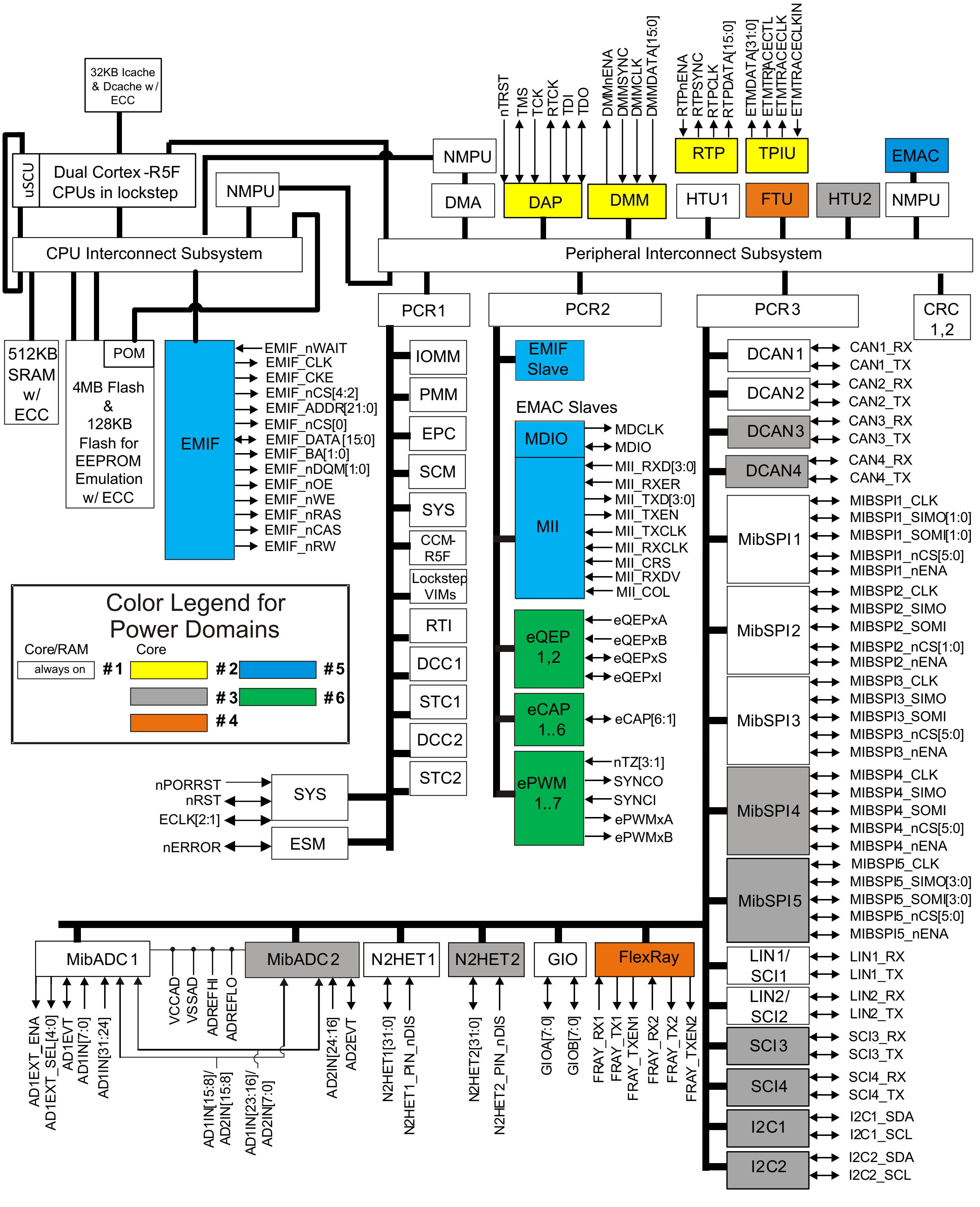
The Error Signaling Module (ESM) monitors all device errors and determines whether an interrupt or external Error pin/ball is triggered when a fault is detected. The nERROR can be monitored externally as an indicator of a fault condition in the microcontroller.

The External Memory Interface (EMIF) provides a memory extension to asynchronous and synchronous memories or other slave devices.

Several interfaces are implemented to enhance the debugging capabilities of application code. In addition to the built in ARM® Cortex®-R5F CoreSight™ debug features. Embedded Cross Trigger (ECT) supports the interaction and synchronization of multiple triggering events within the SoC. An External Trace Macrocell (ETM) provides instruction and data trace of program execution. For instrumentation purposes, a RAM Trace Port Module (RTP) is implemented to support high-speed tracing of RAM and peripheral accesses by the CPU or any other master. A Data Modification Module (DMM) gives the ability to write external data into the device memory. Both the RTP and DMM have no or only minimum impact on the program execution time of the application code. A Parameter Overlay Module (POM) is included to enhance the calibration capabilities of application code. The POM can re-route Flash accesses to internal memory or to the EMIF, thus avoiding the re-programming steps necessary for parameter updates in Flash.

With integrated safety features and a wide choice of communication and control peripherals, the TMS570LC43x is an ideal solution for high performance real time control applications with safety critical requirements.

Figure 1-1. Block Diagram



Copyright © 2016, Texas Instruments Incorporated

1.3 Endianism Considerations

1.3.1 *TMS570: Big Endian (BE32)*

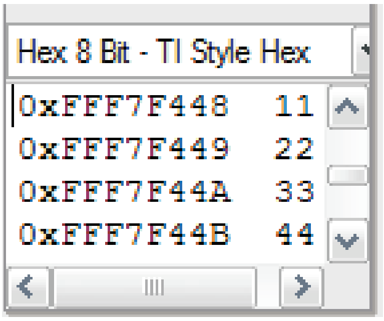
The TMS570LC43x family is based on the ARM® Cortex®-R5F core. ARM has designed this core to be used in big-endian and little-endian systems. For the TI TMS570LC43x family, the endianness has been configured to BE32. Big-endian systems store the most-significant byte of a multi-byte data field in the lowest memory address. Also, the address of the multi-byte data field is the lowest address. Following is an example of the physical addresses of individual bytes.

Figure 1-2. Example: SPIDELAY – 0xFFF7F448

31 24	23 16
C2TDELAY[7:0]	T2CDELAY[7:0]
Byte 3 - 0xFFF7F448	Byte 2 - 0xFFF7F449

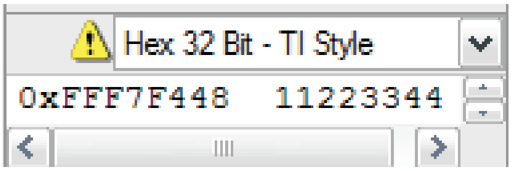
15 8	7 0
T2EDELAY[7:0]	C2EDELAY[7:0]
Byte 1 - 0xFFF7F44A	Byte 0 - 0xFFF7F44B

32-bit accesses to this register should use the lowest address, that is, 0xFFF7F448. Writing 0x11223344 to address 0xFFF7F448 shows the following when viewing the memory in 8-bit and 32-bit modes.



Hex 8 Bit - TI Style Hex

```
0xFFF7F448   11
0xFFF7F449   22
0xFFF7F44A   33
0xFFF7F44B   44
```



Hex 32 Bit - TI Style

```
0xFFF7F448   11223344
```

As such the headers provided as part of HALCoGen do take the endianness into account and provide header structures that are agnostic to endianness. This is achieved by using C directives for the compiler that make use of the compile options configured for the project by the user (__little_endian__ used in Code Composer Studio codegen tools). This directive may need to be adapted for other compilers.

```
#ifdef __little_endian__
    char  C2EDELAY     :  8U;  /**lt; 0xF448: CS to ENA        */
    char  T2EDELAY     :  8U;  /**lt; 0xF449: Transmit to ENA  */
    char  T2CDELAY     :  8U;  /**lt; 0xF44A: Transmit to CS   */
    char  C2TDELAY     :  8U;  /**lt; 0xF44B: CS to Transmit   */
#else
    char  C2TDELAY     :  8U;  /**lt; 0xF448: CS to Transmit   */
    char  T2CDELAY     :  8U;  /**lt; 0xF449: Transmit to CS   */
    char  T2EDELAY     :  8U;  /**lt; 0xF44A: Transmit to ENA  */
    char  C2EDELAY     :  8U;  /**lt; 0xF44B: CS to ENA        */
```

Chapter 2
SPNU563A–March 2018

Architecture

This chapter consists of five sections. The first section describes specific aspects of the device architecture. The second section describes the clocking structure of the microcontrollers. The third section gives an overview of the device memory organization. The fourth section details exceptions on the device, and the last section describes the system and peripheral control registers of the microcontroller.

Topic **Page**

2.1 Introduction

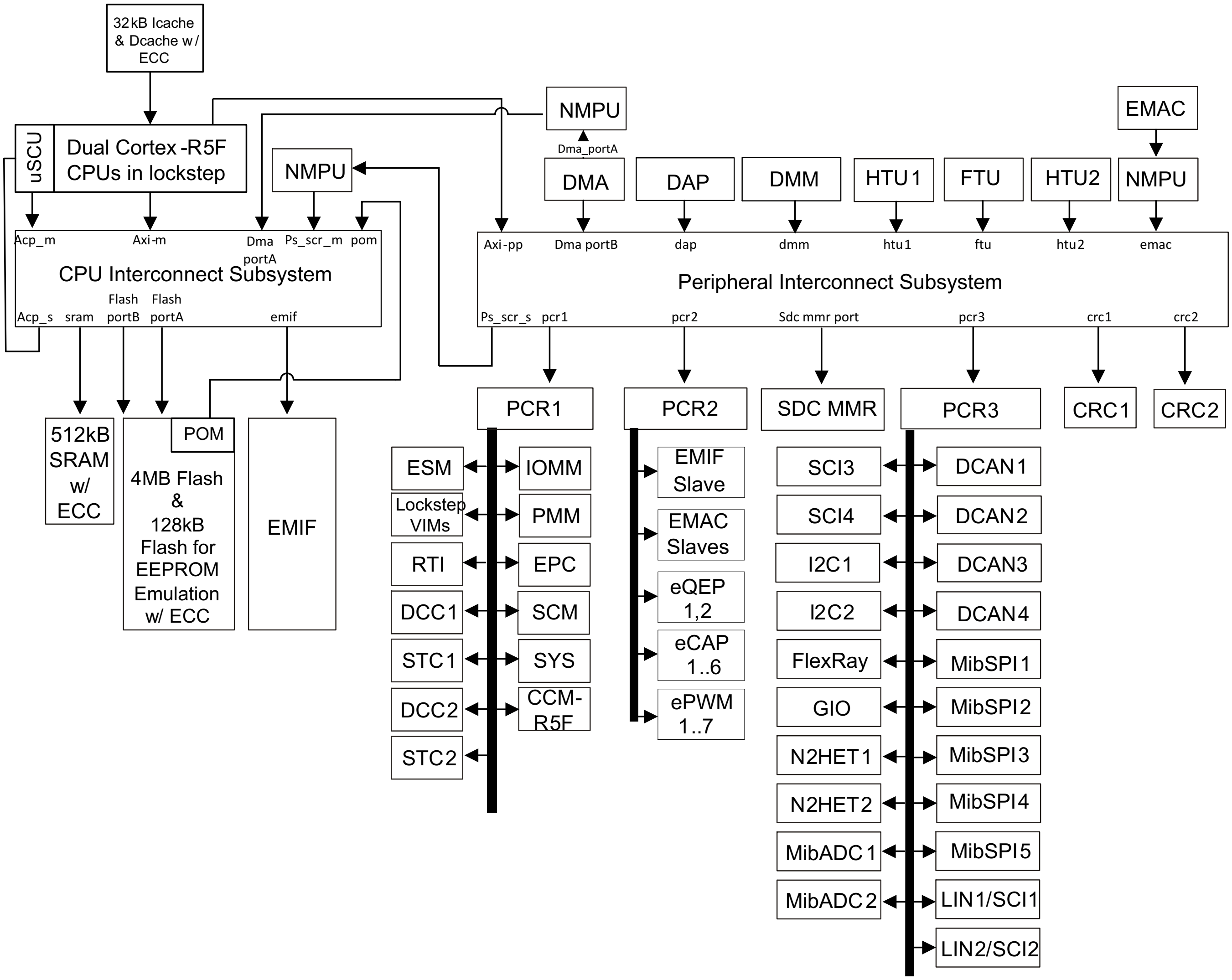
The TMS570LC43x family of microcontrollers is based on the Texas Instruments TMS570 Architecture. This chapter describes specific aspects of the architecture as applicable to the TMS570LC43x family of microcontrollers.

2.1.1 Architecture Block Diagram

The TMS570LC43x microcontrollers are based on the TMS570 Platform architecture, which defines the interconnect between the bus masters and the bus slaves. The architecture consists of two main interconnects which connect all the masters and slaves together. The separation of the two interconnects creates a concept of two safety islands. The CPU safety island consists of the CPU Interconnect Subsystem which glues the masters and slaves together. The CPU safety island contains high degree of safety diagnostics on the bus system and the memories. Memories and buses are protected by means of ECC on the data path using Single-Bit Correction Double-Bit Detection (SECDED) scheme. Parity detection scheme is used on all address and control paths between all masters and slaves. Safety diagnostic logic is built into the CPU Interconnect Subsystem where all traffics going in and out are checked against their expected behaviors during application runtime. In addition, self-test logic is built into the CPU Interconnect Subsystem which can be enabled to diagnose possible faults. The Peripheral safety island consists of the Peripheral Interconnect Subsystem which glues the rest of the masters and slaves in the device. Diagnostic on the peripheral island is by means of ECC or parity protection on the peripheral memories and MPU protection.

Figure 2-1 shows a high-level architectural block diagram for the microcontroller.

Figure 2-1. Architectural Block Diagram



2.1.2 Definitions of Terms

Table 2-1 provides a definition of terms used in the architectural block diagram.

Table 2-1. Definition of Terms

Acronym/Term	Full Form	Description
ADCx	Analog-to-Digital Converter	The ADC uses the Successive Approximation Register architecture. It features a selectable 10-bit or 12-bit resolution. The ADC module also includes a RAM to hold the conversion results. A digital logic wrapper manages accesses to the control and status registers. There are two ADC modules on this device.
CCM-R5F	CPU Compare Module for Cortex-R5F core	During lockstep mode, the outputs of the two CPUs are compared on each CPU clock cycle by this module. Any miscompare is flagged as an error of the highest severity level. The outputs of the two VIMs in lockstep are also compared on each cycle by this module.
Cortex-R5F CPU	–	The Cortex-R5F has one AXI-M master port on the CPU Interconnect Subsystem and another AXI-PP peripheral port on the peripheral Interconnect Subsystem for low latency access. Each master port is limited to accesses on the resources attached to the respective interconnect.
CPU Interconnect Subsystem	CPU Side Switched Central Resource Controller	This is one of the two main SCRs in the device. It arbitrates between the accesses from multiple bus masters to the bus slaves using a round robin priority scheme. This interconnect subsystem contains diagnostic logic to perform parity checking on address and control signals from bus masters, parity checking on response signals from slaves, ECC generation and evaluation on the datapath for transactions initiated by the non-CPU masters and also self test logic to diagnose itself.
CRCx	Cyclic Redundancy Checker	The CRC module provides two channels to perform background signature verification on any memory region using a 64-bit maximum-length linear feedback shift register (LFSR) . The CRC module is a bus slave in this device.
DAP	Debug Access Port	The DAP allows a tool such as a debugger to read from or write to any region in the device memory-map. The DAP is a bus master in this device.
DCANx	Controller Area Network controller	The DCAN supports the CAN 2.0B protocol standard and uses a serial, multi-master communication protocol that efficiently supports distributed real-time control with robust communication rates of up to 1 megabit per second (Mbps). The DCAN is ideal for applications operating in noisy and harsh environments (for example, automotive and industrial fields) that require reliable serial communication or multiplexed wiring.
DCCx	Dual Clock Comparator	This module is primarily intended for use to determine the accuracy of a clock signal during the execution of an application. An additional use of this module is to measure the frequency of a selectable clock source, using the input clock as a reference.
DMA	Direct Memory Access	The DMA module is used for transferring 8-, 16-, 32- or 64-bit data across the entire device memory-map. The DMA module is one of the bus masters on the device. That is, it can initiate a read or a write transaction. DMA has two master ports with DMA_PortA and DMA_PortB. DMA_PortA is connected to the CPU Interconnect Subsystem and DMA_PortB is connected to the Peripheral Interconnect Subsystem. DMA can transfer data from resources in CPU Interconnect Subsystem to resources in the Peripheral Interconnect Subsystem and vice versa.
DMM	Data Modification Module	The DMM allows a tool to use the special DMM I/O interface to modify any data value in any RAM on the device. The modification is done with minimal interruption to the application execution, and can be used for calibration of application algorithms. the DMM is also a bus master in this device.
eCAP	Enhanced Capture Module	The enhanced Capture (eCAP) module is essential in systems where accurate timing of external events is important.
eFuse	Electronically Programmable Fuse controller	Electrically programmable fuses (eFuses) are used to configure the device after deassertion of PORRST. The eFuse values are read and loaded into internal registers as part of the power-on-reset sequence. The eFuse values are protected with Single-Bit Error Correction Double-Bit Error Detection (SECDED) codes. These fuses are programmed during the initial factory test of the device. The eFuse controller is designed so that the state of the eFuses cannot be changed once the device is packaged.

Table 2-1. Definition of Terms (continued)

Acronym/Term	Full Form	Description
ePWM	Enhanced Pulse Width Modulator	The enhanced pulse width modulator (ePWM) peripheral is a key element in controlling many of the power electronic systems found in both commercial and industrial equipments. These systems include digital motor control, switch mode power supply control, uninterruptible power supplies (UPS), and other forms of power conversion. The ePWM peripheral performs a digital to analog (DAC) function, where the duty cycle is equivalent to a DAC analog value; it is sometimes referred to as a Power DAC.
eQEP	Enhanced Quadrature Encoder Pulse Module	The enhanced quadrature encoder pulse (eQEP) module is used for direct interface with a linear or rotary incremental encoder to get position, direction, and speed information from a rotating machine for use in a high-performance motion and position-control system.
ECC	Error Correction Code	This is a code that is used by the Single-Bit Error Correction Double-Bit Error Detection (SECDED) logic inside the two Cortex-R5F processors (CPUs) and various modules that support ECC. Depending on the memory configuration, the number of ECC bits may vary. There are 8 bits of ECC for every 64 bits of data accessed from the CPU level 2 memory such as flash and RAM. CPU's level 1 cache system consists of instruction cache and data cache and each is additionally composed of data RAM, tag RAM or dirty RAM. The number of ECC bits used to protect these RAMs vary. Modules which support ECC protection on their local RAMs can also employ different number of ECC bits depending on the RAM's configuration. For example, DMA module use 9 bits of ECC to protect its local control packet memory.
EMAC	Ethernet Media Access Controller	The EMAC has a dedicated DMA-type component that is used to transfer data to / from the EMAC descriptor memory from / to another memory in the device memory-map. This DMA-type component of the EMAC is a bus master in this device.
EMAC slaves	Ethernet Media Access Controller slave ports	There are four EMAC slaves: 1. EMAC Control Module: this provides an interface between the EMAC and MDIO modules and the bus masters. It also includes 8KB of RAM to hold EMAC packet buffer descriptors. 2. EMAC: The EMAC module interfaces to the other devices on the Ethernet Network using the Media Independent Interface (MII) or Reduced Media Independent Interface (RMII). 3. Management Data Input / Output (MDIO): The MDIO module is used to manage the physical layer (PHY) device connected to the EMAC module. 4. Communications Port Programming Interface (CPPI): This is the 8KB of RAM used to hold the EMAC packet buffer descriptors.
EMIF slaves	External Memory Interface slave ports	There are five EMIF slaves: • External SDRAM memory: EMIF chip select 0 • External asynchronous memories: EMIF chip selects 2, 3 and 4 • EMIF module control and status registers
EPC	Error Profiling Controller	This module is used to profile the occurrences of single-bit and double-bit ECC errors detected by the CPU and the CPU Interconnect Subsystem.
ESM	Error Signal Module	ESM collects and reports the various error conditions on the device. The error condition is categorized based on a severity level. Error response is then generated based on the category of the error. Possible error responses include a low priority interrupt, high priority NMI interrupt and an external pin action.
Flash Memory	Level 2 Flash Memory	There are two slave ports (Flash_PortA and Flash_PortB) to access the flash memory consisting of three flash banks. The two ports allow two masters to access among the three banks in parallel. There are two 2Mbyte banks and one EEPROM bank. The EEPROM bank is a flash bank that is dedicated for use as an emulated EEPROM. This device supports 128KB of flash for emulated EEPROM.
FlexRay	FlexRay communication controller	The FlexRay uses a dual channel serial, fixed time base multi-master communication protocol with communication rates of 10 megabits per second (Mbps) per channel.
FTU	FlexRay Transfer Unit	The FTU is a dedicated transfer unit for the FlexRay communication interface controller. The FTU has a native interface to the FlexRay message RAM and is used to transfer data to / from the FlexRay message RAM from / to another region in the device memory-map. The FTU is a bus master in this device.
GIO	General-purpose Input/Output	The GIO module allows up to 16 terminals to be used as general-purpose Input or Output. Each of these are also capable of generating an interrupt to the CPU.

Table 2-1. Definition of Terms (continued)

Acronym/Term	Full Form	Description
HTUx	High-end timer Transfer Unit	The HTU is a dedicated transfer unit for the New Enhanced High-End Timer module. The HTU has a native interface to the N2HET RAM, and is used to transfer data to / from the N2HET RAM from / to another region in the device memory-map. There is one HTU per N2HET module, so that there are 2 HTU modules on the device. The HTUx are bus masters in this device.
I2Cx	Inter-Integrated Circuit controller	The I2C module is a multi-master communication module providing an interface between the device and an I2C-compatible device via the I2C serial bus. The I2C supports both 100 Kbps and 400 Kbps speeds.
IOMM	IO Multiplexing Module	This module controls the multiplexing on the device I/Os. Multiple functions can be multiplexed onto the same device IO. Through IOMM module, user can enable a specific function onto a device pin.
LINx	Local Interconnect Network controller	The LIN module supports the Local Interconnect standard revision 2.1 and can be used as a UART in full-duplex mode using the standard Non-Return-to-Zero (NRZ) format.
Lockstep	–	This is the mode of operation of the dual ARM Cortex-R5F CPUs. The outputs of the two CPUs are compared on each CPU clock cycle. Any miscompare is flagged as an error of the highest severity level. In addition to the lockstep CPUs, the two Vector Interrupt Module (VIM) are also in lockstep.
MibSPIx	Multi-Buffered Serial Peripheral Interface	The MibSPIx modules also support the standard SPI communication protocol. The transfers are all grouped into transfer chunks called “transfer groups”. These transfer groups are made up of one ore more buffers in the MibSPIx RAM. The RAM is used to hold the control information and data to be transmitted, as well as the status information and data that is received. There are five MibSPI modules in this device.
N2HETx	New Enhanced High-End Timer	The N2HET is an advanced intelligent timer that provides sophisticated timing functions for real-time applications. The timer is software-controlled, using a reduced instruction set, with a specialized timer micromachine and an attached I/O port. The N2HET can be used for pulse width modulated outputs, capture or compare inputs, or general-purpose I/O.
NMPUx	Enhanced Memory Protection Unit	There are three standalone NMPUs on this device protecting memory transactions initiated by DMA, EMAC and other masters onto the resources on the device. In this device, all transactions initiated by non-CPU masters will go through two levels of MPU protection. The two levels can be a combination of two NMPU in series or one standalone NMPU and one build-in MPU as part of the master. One NMPU is dedicated to the DMA port connecting to the CPU Interconnect Subsystem as the second level protection while the built-in MPU inside the DMA acts as the first level protection. HTUx and FTU all have their built-in MPU acting as the first level protection. All accesses initiated by the masters on the Peripheral Interconnect Subsystem side will funnel through another NMPU sitting in between the path connecting the Peripheral Interconnect Subsystem to the CPU Interconnect Subsystem. This will act as the second level protection for HTUx, FTU and EMAC. EMAC does not have the built-in MPU and hence a standalone NMPU is instantiated between the EMAC and the interconnect.
Peripheral Interconnect Subsystem	Peripheral Side Switched Central Resource Controller	This is one of the two main SCRs in the device. It arbitrates between the accesses from multiple bus masters to the bus slaves using a round robin priority scheme.
PCRx	Peripheral Central Resource controller	The PCR manages the accesses to the peripheral registers and peripheral memories. It provides a global reset for all the peripherals. It also supports the capability to selectively enable or disable the clock for each peripheral individually. The PCR also manages the accesses to the system module registers required to configure the device’s clocks, interrupts, and so on. The system module registers also include status flags for indicating exception conditions – resets, aborts, errors, interrupts. This device has three PCR modules with each capable to access different peripherals as shown in the block diagram. The three PCRs are slaves to the Peripheral Interconnect Subsystem.
PMM	Power Management Module	This module controls the clock gating to the various logic power domains in the device. Through PMM, user can place a power domain among Active, Idle or Off modes. This device does not implement physical power domains in which power can be turned off. Trying to turn off a power domain has no effect on this device in terms of power consumption but clocks will be gated off to remove dynamic power. Idle and Off modes in this device behave the same from power consumption perspective.

Table 2-1. Definition of Terms (continued)

Acronym/Term	Full Form	Description
POM	Parameter Overlay Module	The parameter overlay module redirects accesses to a programmable region in flash memory (read-only) to a RAM memory, either on-chip or via the external memory interface (EMIF). This allows a user to evaluate the impact of changing values of constants stored in the flash memory without actually having to erase and reprogram the flash. The POM is also a bus master in this device.
PS_SCR_M	Peripheral SCR Master Port	All transactions to access the resources in the CPU Interconnect Subsystem by HTUx, FTU, EMAC, DMM and DAP will funnel through the PS_SCR_S slave port on the Peripheral Interconnect Subsystem. The PS_SCR_S slave is then connected to the PS_SCR_M master port on the CPU Interconnect Subsystem via a NMPU in between.
RTI	Real Time Interrupt module	RTI module provides timer functionality for operating systems and for benchmarking code. The module incorporates several counters, which define the timebases needed for scheduling in the operating system.
SCIx	Serial Communication Interface	The SCI module supports the standard UART in full-duplex mode using the standard Non-Return-to-Zero (NRZ) format.
SCM	SCR Control Module	This module is used to change certain configurations such as timeout counters of the CPU Interconnect Subsystem. This module is also used to initiate selftest for the CPU Interconnect Subsystem.
SDC MMR	Safety Diagnostic Checker Memory-Map Register Port for CPU Interconnect Subsystem	There are memory-mapped status registers to record both the run-time and self-test diagnostic of the CPU Interconnect Subsystem. These registers are accessed via the SDC MMR slave port in the Peripheral Interconnect Subsystem.
SRAM	Level 2 Static RAM	There is one slave port to access the on-chip SRAM.
STCx	Selftest Controller	There are two STC modules in this device. One is used to test the CPU subsystem including both CPU cores and/or the ACP component using the Deterministic Logic Bist Controller as the test engine. The other STC is used to test either or both the N2HETs in the device.
SYS	System Module	This module contains the housekeeping logic to control and log overall system functions and status such as setting up the clock sources, clock domains, generation and reception of reset sources.
µSCU	Micro Snooping Control Unit	The µSCU which is part of the Cortex-R5 processor system contains an ACP (Accelerator Coherency Port) interface which provides snoop capabilities on write-transactions coming from the non-CPU masters. Transactions are received on the ACP-S slave port, and transmitted on the memory system via the ACP-M master port. The ACP automatically invalidates the appropriate Level 1 data-cache lines at the appropriate time, allowing software maintenance free cache coherency for data in write-through cache regions, as well as non-cached.
VIM	Vectored Interrupt Manager	VIM provides hardware assistance for prioritizing and controlling the many interrupt sources present on a device. There are two VIMs in this device. When the device is configured in lockstep mode, the two VIMs are also in lockstep. The outputs of the two VIMs are compared cycle by cycle by the CCM-R5 module.

2.1.3 Bus Master / Slave Access Privileges

This device implements some restrictions on the bus slave access privileges in order to improve the overall throughput of the interconnect shown in Figure 2-1. Table 4-1 shows the implemented point to point connections between the masters and slaves connected to the Peripheral Interconnect Subsystem. Table 4-3 lists the implemented point to point connections between the masters and slaves connected to the CPU Interconnect Subsystem.

2.1.4 CPU Interconnect Subsystem SDC MMR Port

The CPU Interconnect Subsystem SDC MMR Port is a special slave to the Peripheral Interconnect Subsystem. It is memory-mapped at starting address of FA00 0000h. Various status registers pertaining to the diagnostics of the CPU Interconnect Subsystem can be access through this slave port. The CPU Interconnect Subsystem contains built-in hardware diagnostic checkers which will constantly watch transactions flowing through the interconnect. There is a checker for each master and slave attached to the CPU Interconnect Subsystem. The checker checks the expected behavior against the generated behavior by the interconnect. For example, if the CPU issues a burst read request to the flash, the checker will ensure that the expected behavior is indeed a burst read request to the proper slave module. If the interconnect generates a transaction which is not a read, or not a burst or not to the flash as the destination, then the checker will flag it in one of the registers. The detected error will also be signaled to the ESM module. Table 4-2 lists the CPU Interconnect Subsystem SDC register bit field mapping.

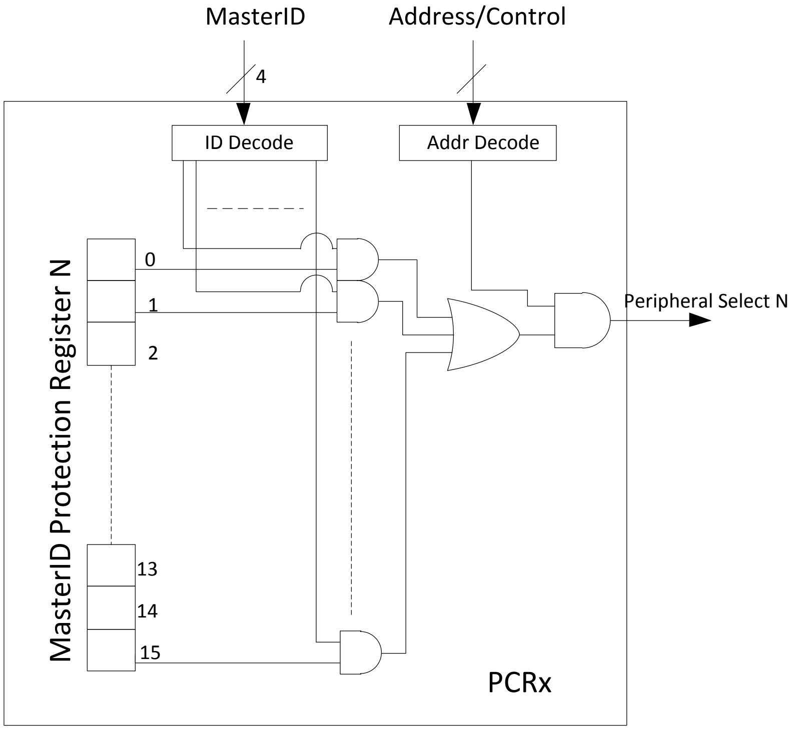
2.1.5 Interconnect Subsystem Runtime Status

Other than the runtime checker status as described in Section 2.1.4, the CPU Interconnect Subsystems and the Peripheral Interconnect Subsystem in the microcontroller also generates several status onto the system that are captured in the SCM (SCR Control Module). Table 4-4 lists the SCM register bit mapping.

2.1.6 Master ID to PCRx

The master ID associated with each master port on the Peripheral Interconnect Subsystem contains a 4-bit value. The master ID is passed along with the address and control signals to three PCR modules. PCR decodes the address and control signals to select the peripheral. In addition, it decodes this 4-bit master ID value to perform memory protection. With 4-bit of master ID, it allows the PCR to distinguish among 16 different masters to allow or dis-allow access to a given peripheral. Associated with each peripheral a 16-bit Master ID access protection register is defined. Each bit grants or denies the permission of the corresponding binary coded decimal masterID. For example, if bit 5 of the access permission register is set, it grants master ID 5 to access the peripheral. If bit 7 is clear, it denies master ID 7 to access the peripheral. Figure 2-2 illustrates the Master-ID filtering scheme. The master ID of each master that is capable of accessing the PCRx is listed in Table 4-1. Also see Section 2.5.3 for details on the registers definition.

Figure 2-2. PCR MasterID Filtering



2.2 Memory Organization

2.2.1 Memory-Map Overview

The Cortex-R5F uses a 32-bit address bus, giving it access to a memory space of 4GB. This space is divided into several regions, each addressed by different memory selects. Figure 2-3 shows the memory-map of the microcontroller.

The main flash instruction memory is addressed starting at 0x00000000 by default. This is also the reset vector location – the ARM Cortex-R5F processor core starts execution from the reset vector address of 0x00000000 whenever the core gets reset.

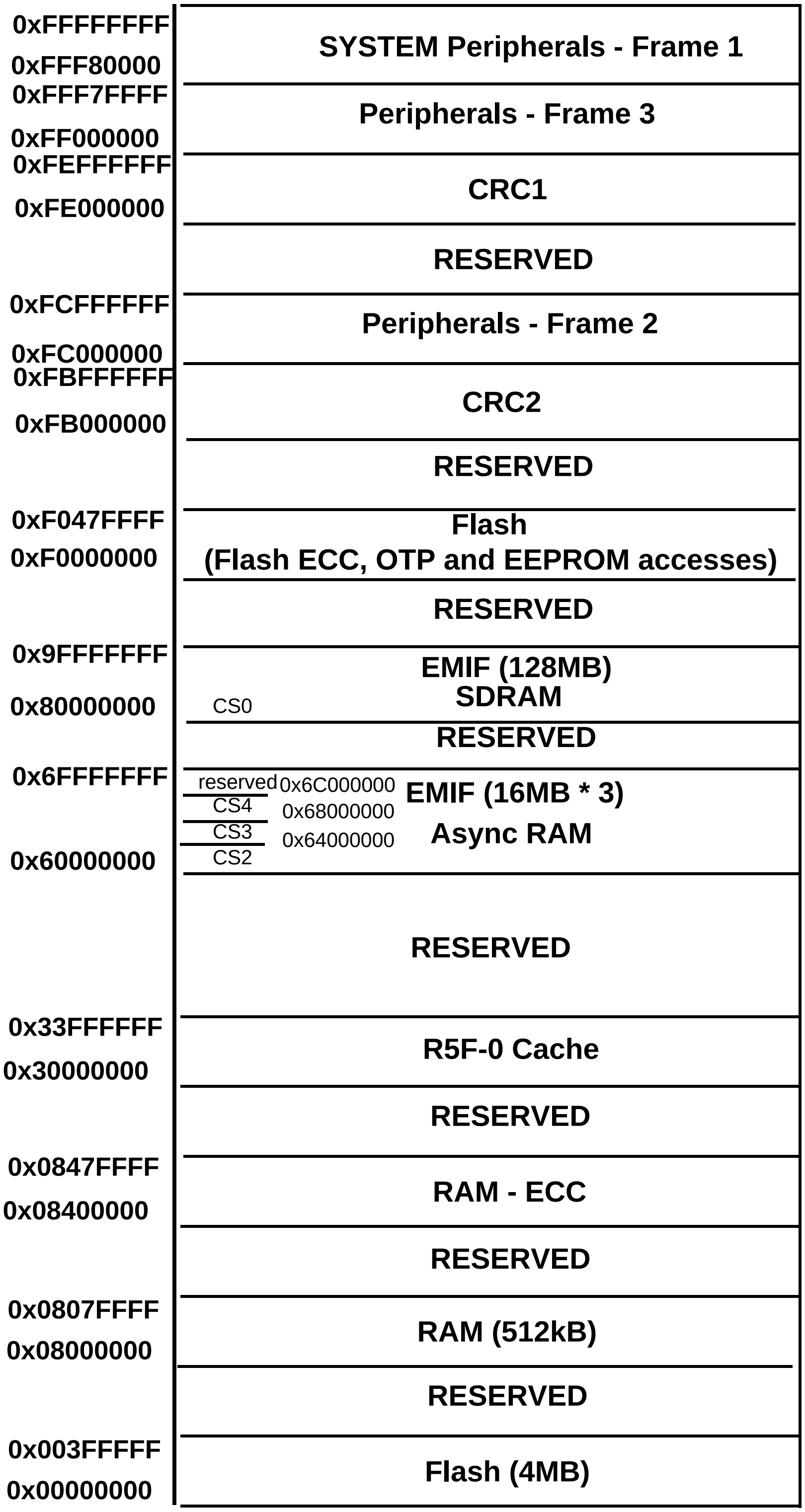
The CPU data RAM is addressed starting at 0x08000000 by default.

The device also supports the swapping of the CPU instruction memory (flash) and data memory (RAM). This can be done by configuring the MEM SWAP field of the Bus Matrix Module Control Register 1 (BMMCR1).

After swapping, the data RAM is accessed starting from 0x00000000 and the RAM ECC locations are accessed starting from 0x00400000. The flash memory is now accessed starting from 0x08000000.

NOTE: After the swap with the flash memory-mapped to 0x08000000, only 512kB of the flash memory from 0x08000000 to 0x0807FFFF will be accessible by the bus masters.

Figure 2-3. Memory-Map



2.2.2 Memory-Map Table

The control and status registers for each module are mapped within the CPU's 4GB memory space. Some modules also have associated memories, which are also mapped within this space.

Table 2-2 shows the starting and ending addresses of each module's register frame and any associated memory. The table also shows the response generated by the module or the interconnect whenever an access is made to an unimplemented location inside the register or memory frame.

Table 2-2. Module Registers / Memories Memory-Map

Target Name	Memory Select	Address Range		Frame Size	Actual Size	Response for Access to Unimplemented Locations in Frame
		Start	End			
Level 2 Memories						
Level 2 Flash Data Space		0x0000_0000	0x003F_FFFF	4MB	4MB	Abort
Level 2 SRAM		0x0800_0000	0x083F_FFFF	4MB	512kB	Abort
Level 2 SRAM ECC		0x0840_0000	0x087F_FFFF	4MB	512kB	
Accelerator Coherency Port						
Accelerator Coherency Port		0x0800_0000	0x087F_FFFF	8MB	512kB	Abort
Level 1 Cache Memories						
Cortex-R5F Data Cache Memory		0x3000_0000	0x30FF_FFFF	16MB	32kB	Abort
Cortex-R5F Instruction Cache Memory		0x3100_0000	0x31FF_FFFF	16MB	32kB	
External Memory Accesses						
EMIF Chip Select 2 (asynchronous)		0x6000_0000	0x63FF_FFFF	64MB	16MB	Access to Reserved space Generates Abort
EMIF Chip Select 3 (asynchronous)		0x6400_0000	0x67FF_FFFF	64MB	16MB	
EMIF Chip Select 4 (asynchronous)		0x6800_0000	0x6BFF_FFFF	64MB	16MB	
EMIF Chip Select 0 (synchronous)		0x8000_0000	0x87FF_FFFF	128MB	128MB	
Flash OTP, ECC, EEPROM Bank						
Customer OTP, Bank0		0xF000_0000	0xF000_1FFF	8kB	4kB	Abort
Customer OTP, Bank1		0xF000_2000	0xF000_3FFF	8kB	4kB	Abort
Customer OTP, EEPROM Bank		0xF000_E000	0xF000_FFFF	8kB	1kB	Abort
Customer OTP-ECC, Bank0		0xF004_0000	0xF004_03FF	1kB	512B	Abort
Customer OTP-ECC, Bank1		0xF004_0400	0xF004_07FF	1kB	512B	Abort
Customer OTP-ECC, EEPROM Bank		0xF004_1C00	0xF004_1FFF	1kB	128B	Abort
TI OTP, Bank0		0xF008_0000	0xF008_1FFF	8kB	4kB	Abort
TI OTP, Bank1		0xF008_2000	0xF008_3FFF	8kB	4kB	Abort
TI OTP, EEPROM Bank		0xF008_E000	0xF008_FFFF	8kB	1kB	Abort
TI OTP-ECC, Bank0		0xF00C_0000	0xF00C_03FF	1kB	512B	Abort
TI OTP-ECC, Bank1		0xF00C_0400	0xF00C_07FF	1kB	512B	Abort

Table 2-2. Module Registers / Memories Memory-Map (continued)

Target Name	Memory Select	Address Range		Frame Size	Actual Size	Response for Access to Unimplemented Locations in Frame
		Start	End			
TI OTP-ECC, EEPROM Bank		0xF00C_1C00	0xF00C_1FFF	1kB	128B	Abort
EEPROM Bank-ECC		0xF010_0000	0xF01F_FFFF	1MB	16kB	Abort
EEPROM Bank		0xF020_0000	0xF03F_FFFF	2MB	128kB	Abort
Flash Data Space ECC		0xF040_0000	0xF05F_FFFF	2MB	512kB	Abort
Interconnect SDC MMR						
Interconnect SDC MMR		0xFA00_0000	0xFAFF_FFFF	16MB	16MB	
Registers/Memories under PCR2 (Peripheral Segment 2)						
CPPI Memory Slave (Ethernet RAM)	PCS[41]	0xFC52_0000	0xFC52_1FFF	8kB	8kB	Abort
CPGMAC Slave (Ethernet Slave)	PS[30]-PS[31]	0xFCF7_8000	0xFCF7_87FF	2kB	2kB	No Error
CPGMACSS Wrapper (Ethernet Wrapper)	PS[29]	0xFCF7_8800	0xFCF7_88FF	256B	256B	No Error
Ethernet MDIO Interface	PS[29]	0xFCF7_8900	0xFCF7_89FF	256B	256B	No Error
ePWM1	PS[28]	0xFCF7_8C00	0xFCF7_8CFF	256B	256B	Abort
ePWM2		0xFCF7_8D00	0xFCF7_8DFF	256B	256B	Abort
ePWM3		0xFCF7_8E00	0xFCF7_8EFF	256B	256B	Abort
ePWM4		0xFCF7_8F00	0xFCF7_8FFF	256B	256B	Abort
ePWM5	PS[27]	0xFCF7_9000	0xFCF7_90FF	256B	256B	Abort
ePWM6		0xFCF7_9100	0xFCF7_91FF	256B	256B	Abort
ePWM7		0xFCF7_9200	0xFCF7_92FF	256B	256B	Abort
eCAP1		0xFCF7_9300	0xFCF7_93FF	256B	256B	Abort
eCAP2	PS[26]	0xFCF7_9400	0xFCF7_94FF	256B	256B	Abort
eCAP3		0xFCF7_9500	0xFCF7_95FF	256B	256B	Abort
eCAP4		0xFCF7_9600	0xFCF7_96FF	256B	256B	Abort
eCAP5		0xFCF7_9700	0xFCF7_97FF	256B	256B	Abort
eCAP6	PS[25]	0xFCF7_9800	0xFCF7_98FF	256B	256B	Abort
eQEP1		0xFCF7_9900	0xFCF7_99FF	256B	256B	Abort
eQEP2		0xFCF7_9A00	0xFCF7_9AFF	256B	256B	Abort
PCR2 registers	PPSE[4]-PPSE[5]	0xFCFF_1000	0xFCFF_17FF	2kB	2kB	
NMPU (CPGMAC)	PPSE[6]	0xFCFF_1800	0xFCFF_18FF	512B	512B	Abort
EMIF Registers	PPS[2]	0xFCFF_E800	0xFCFF_E8FF	256B	256B	Abort
Cyclic Redundancy Checker (CRC) Module Register Frame						
CRC1		0xFE00_0000	0xFEFF_FFFF	16MB	512kB	Accesses above 0xFE000200 generate abort.
CRC2		0xFB00_0000	0xFBFF_FFFF	16MB	512kB	Accesses above 0xFB000200 generate abort.
Memories under User PCR3 (Peripheral Segment 3)						
MIBSPI5 RAM	PCS[5]	0xFF0A_0000	0xFF0B_FFFF	128kB	2kB	Abort for accesses above 2KB

Table 2-2. Module Registers / Memories Memory-Map (continued)

Target Name	Memory Select	Address Range		Frame Size	Actual Size	Response for Access to Unimplemented Locations in Frame
		Start	End			
MIBSPI4 RAM	PCS[3]	0xFF06_0000	0xFF07_FFFF	128kB	2kB	Abort for accesses above 2KB
MIBSPI3 RAM	PCS[6]	0xFF0C_0000	0xFF0D_FFFF	128kB	2kB	Abort for accesses above 2KB
MIBSPI2 RAM	PCS[4]	0xFF08_0000	0xFF09_FFFF	128kB	2kB	Abort for accesses above 2KB
MIBSPI1 RAM	PCS[7]	0xFF0E_0000	0xFF0F_FFFF	128kB	4kB	Abort for accesses above 4KB
DCAN4 RAM	PCS[12]	0xFF18_0000	0xFF19_FFFF	128kB	8kB	Abort generated for accesses beyond offset 0x2000
DCAN3 RAM	PCS[13]	0xFF1A_0000	0xFF1B_FFFF	128kB	8kB	Abort generated for accesses beyond offset 0x2000
DCAN2 RAM	PCS[14]	0xFF1C_0000	0xFF1D_FFFF	128kB	8kB	Abort generated for accesses beyond offset 0x2000
DCAN1 RAM	PCS[15]	0xFF1E_0000	0xFF1F_FFFF	128kB	8kB	Abort generated for accesses beyond offset 0x2000.
MIBADC2 RAM	PCS[29]	0xFF3A_0000	0xFF3B_FFFF	128kB	8kB	Wrap around for accesses to unimplemented address offsets lower than 0x1FFF.
MIBADC2 Look-UP Table					384 bytes	Look-Up Table for ADC2 wrapper. Starts at address offset 0x2000 and ends at address offset 0x217F. Wrap around for accesses between offsets 0x0180 and 0x3FFF. Abort generation for accesses beyond offset 0x4000.

Table 2-2. Module Registers / Memories Memory-Map (continued)

Target Name	Memory Select	Address Range		Frame Size	Actual Size	Response for Access to Unimplemented Locations in Frame
		Start	End			
MIBADC1 RAM	PCS[31]	0xFF3E_0000	0xFF3F_FFFF	128kB	8kB	Wrap around for accesses to unimplemented address offsets lower than 0x1FFF.
MIBADC1 Look-UP Table					384 bytes	Look-Up Table for ADC1 wrapper. Starts at address offset 0x2000 and ends at address offset 0x217F. Wrap around for accesses between offsets 0x0180 and 0x3FFF. Abort generation for accesses beyond offset 0x4000.
NHET2 RAM	PCS[34]	0xFF44_0000	0xFF45_FFFF	128kB	16kB	Wrap around for accesses to unimplemented address offsets lower than 0x3FFF. Abort generated for accesses beyond 0x3FFF.
NHET1 RAM	PCS[35]	0xFF46_0000	0xFF47_FFFF	128kB	16kB	Wrap around for accesses to unimplemented address offsets lower than 0x3FFF. Abort generated for accesses beyond 0x3FFF.
HET TU2 RAM	PCS[38]	0xFF4C_0000	0xFF4D_FFFF	128kB	1kB	Abort
HET TU1 RAM	PCS[39]	0xFF4E_0000	0xFF4F_FFFF	128kB	1kB	Abort
FlexRay TU RAM	PCS[40]	0xFF50_0000	0xFF51_FFFF	128kB	1kB	Abort
Coresight Debug Components						
CoreSight Debug ROM	CSCS[0]	0xFFA0_0000	0xFFA0_0FFF	4kB	4kB	Reads return zeros, writes have no effect
Cortex-R5F Debug	CSCS[1]	0xFFA0_1000	0xFFA0_1FFF	4kB	4kB	Reads return zeros, writes have no effect
ETM-R5	CSCS[2]	0xFFA0_2000	0xFFA0_2FFF	4kB	4kB	Reads return zeros, writes have no effect
CoreSight TPIU	CSCS[3]	0xFFA0_3000	0xFFA0_3FFF	4kB	4kB	Reads return zeros, writes have no effect
POM	CSCS[4]	0xFFA0_4000	0xFFA0_4FFF	4kB	4kB	Reads return zeros, writes have no effect

Table 2-2. Module Registers / Memories Memory-Map (continued)

Target Name	Memory Select	Address Range		Frame Size	Actual Size	Response for Access to Unimplemented Locations in Frame
		Start	End			
CTI1	CSCS[7]	0xFFA0_7000	0xFFA0_7FFF	4kB	4kB	Reads return zeros, writes have no effect
CTI2	CSCS[8]	0xFFA0_8000	0xFFA0_8FFF	4kB	4kB	Reads return zeros, writes have no effect
CTI3	CSCS[9]	0xFFA0_9000	0xFFA0_9FFF	4kB	4kB	Reads return zeros, writes have no effect
CTI4	CSCS[10]	0xFFA0_A000	0xFFA0_AFFF	4kB	4kB	Reads return zeros, writes have no effect
CSTF	CSCS[11]	0xFFA0_B000	0xFFA0_BFFF	4kB	4kB	Reads return zeros, writes have no effect
Registers under PCR3 (Peripheral Segment 3)						
PCR3 registers	PS[31:30]	0xFFF7_8000	0xFFF7_87FF	2kB	2kB	Reads return zeros, writes have no effect
FTU	PS[23]	0xFFF7_A000	0xFFF7_A1FF	512B	512B	Reads return zeros, writes have no effect
HTU1	PS[22]	0xFFF7_A400	0xFFF7_A4FF	256B	256B	Abort
HTU2	PS[22]	0xFFF7_A500	0xFFF7_A5FF	256B	256B	Abort
NHET1	PS[17]	0xFFF7_B800	0xFFF7_B8FF	256B	256B	Reads return zeros, writes have no effect
NHET2	PS[17]	0xFFF7_B900	0xFFF7_B9FF	256B	256B	Reads return zeros, writes have no effect
GIO	PS[16]	0xFFF7_BC00	0xFFF7_BCFF	256B	256B	Reads return zeros, writes have no effect
MIBADC1	PS[15]	0xFFF7_C000	0xFFF7_C1FF	512B	512B	Reads return zeros, writes have no effect
MIBADC2	PS[15]	0xFFF7_C200	0xFFF7_C3FF	512B	512B	Reads return zeros, writes have no effect
FlexRay	PS[12]+PS[13]	0xFFF7_C800	0xFFF7_CFFF	2kB	2kB	Reads return zeros, writes have no effect
I2C1	PS[10]	0xFFF7_D400	0xFFF7_D4FF	256B	256B	Reads return zeros, writes have no effect
I2C2	PS[10]	0xFFF7_D500	0xFFF7_D5FF	256B	256B	Reads return zeros, writes have no effect
DCAN1	PS[8]	0xFFF7_DC00	0xFFF7_DDFF	512B	512B	Reads return zeros, writes have no effect
DCAN2	PS[8]	0xFFF7_DE00	0xFFF7_DFFF	512B	512B	Reads return zeros, writes have no effect

Table 2-2. Module Registers / Memories Memory-Map (continued)

Target Name	Memory Select	Address Range		Frame Size	Actual Size	Response for Access to Unimplemented Locations in Frame
		Start	End			
DCAN3	PS[7]	0xFFF7_E000	0xFFF7_E1FF	512B	512B	Reads return zeros, writes have no effect
DCAN4	PS[7]	0xFFF7_E200	0xFFF7_E3FF	512B	512B	Reads return zeros, writes have no effect
LIN1	PS[6]	0xFFF7_E400	0xFFF7_E4FF	256B	256B	Reads return zeros, writes have no effect
SCI3	PS[6]	0xFFF7_E500	0xFFF7_E5FF	256B	256B	Reads return zeros, writes have no effect
LIN2	PS[6]	0xFFF7_E600	0xFFF7_E6FF	256B	256B	Reads return zeros, writes have no effect
SCI4	PS[6]	0xFFF7_E700	0xFFF7_E7FF	256B	256B	Reads return zeros, writes have no effect
MibSPI1	PS[2]	0xFFF7_F400	0xFFF7_F5FF	512B	512B	Reads return zeros, writes have no effect
MibSPI2	PS[2]	0xFFF7_F600	0xFFF7_F7FF	512B	512B	Reads return zeros, writes have no effect
MibSPI3	PS[1]	0xFFF7_F800	0xFFF7_F9FF	512B	512B	Reads return zeros, writes have no effect
MibSPI4	PS[1]	0xFFF7_FA00	0xFFF7_FBFF	512B	512B	Reads return zeros, writes have no effect
MibSPI5	PS[0]	0xFFF7_FC00	0xFFF7_FDFF	512B	512B	Reads return zeros, writes have no effect
System Modules Control Registers and Memories under PCR1 (Peripheral Segment 1)						
DMA RAM	PPCS[0]	0xFFF8_0000	0xFFF8_0FFF	4kB	4kB	Abort
VIM RAM	PPCS[2]	0xFFF8_2000	0xFFF8_2FFF	4kB	4kB	Wrap around for accesses to unimplemented address offsets lower than 0x2FFF.
RTP RAM	PPCS[3]	0xFFF8_3000	0xFFF8_3FFF	4kB	4kB	Abort
Flash Wrapper	PPCS[7]	0xFFF8_7000	0xFFF8_7FFF	4kB	4kB	Abort
eFuse Farm Controller	PPCS[12]	0xFFF8_C000	0xFFF8_CFFF	4kB	4kB	Abort
Power Domain Control (PMM)	PPSE[0]	0xFFFF_0000	0xFFFF_01FF	512B	512B	Abort
FMTM Note: This module is only used by TI during test	PPSE[1]	0xFFFF_0400	0xFFFF_05FF	512B	512B	Reads return zeros, writes have no effect
STC2 (NHET1/2)	PPSE[2]	0xFFFF_0800	0xFFFF_08FF	256B	256B	Reads return zeros, writes have no effect
SCM	PPSE[2]	0xFFFF_0A00	0xFFFF_0AFF	256B	256B	Abort

Table 2-2. Module Registers / Memories Memory-Map (continued)

Target Name	Memory Select	Address Range		Frame Size	Actual Size	Response for Access to Unimplemented Locations in Frame
		Start	End			
EPC	PPSE[3]	0xFFFF_0C00	0xFFFF_0FFF	1kB	1kB	Abort
PCR1 registers	PPSE[4]-PPSE[5]	0xFFFF_1000	0xFFFF_17FF	2kB	2kB	Reads return zeros, writes have no effect
NMPU (PS_SCR_S)	PPSE[6]	0xFFFF_1800	0xFFFF_19FF	512B	512B	Abort
NMPU (DMA Port A)	PPSE[6]	0xFFFF_1A00	0xFFFF_1BFF	512B	512B	Abort
Pin Mux Control (IOMM)	PPSE[7]	0xFFFF_1C00	0xFFFF_1FFF	2kB	1kB	Reads return zeros, writes have no effect
System Module - Frame 2 (see platform architecture specification)	PPS[0]	0xFFFF_E100	0xFFFF_E1FF	256B	256B	Reads return zeros, writes have no effect
PBIST	PPS[1]	0xFFFF_E400	0xFFFF_E5FF	512B	512B	Reads return zeros, writes have no effect
STC1 (Cortex-R5F)	PPS[1]	0xFFFF_E600	0xFFFF_E6FF	256B	256B	Reads return zeros, writes have no effect
DCC1	PPS[3]	0xFFFF_EC00	0xFFFF_ECFF	256B	256B	Reads return zeros, writes have no effect
DMA	PPS[4]	0xFFFF_F000	0xFFFF_F3FF	1kB	1kB	Abort
DCC2	PPS[5]	0xFFFF_F400	0xFFFF_F4FF	256B	256B	Reads return zeros, writes have no effect
ESM register	PPS[5]	0xFFFF_F500	0xFFFF_F5FF	256B	256B	Reads return zeros, writes have no effect
CCM-R5	PPS[5]	0xFFFF_F600	0xFFFF_F6FF	256B	256B	Reads return zeros, writes have no effect
DMM	PPS[5]	0xFFFF_F700	0xFFFF_F7FF	256B	256B	Reads return zeros, writes have no effect
L2RAMW	PPS[6]	0xFFFF_F900	0xFFFF_F9FF	256B	256B	Abort
RTP	PPS[6]	0xFFFF_FA00	0xFFFF_FAFF	256B	256B	Reads return zeros, writes have no effect
RTI + DWWD	PPS[7]	0xFFFF_FC00	0xFFFF_FCFF	256B	256B	Reads return zeros, writes have no effect
VIM	PPS[7]	0xFFFF_FD00	0xFFFF_FEFF	512B	512B	Reads return zeros, writes have no effect
System Module - Frame 1 (see platform architecture specification)	PPS[7]	0xFFFF_FF00	0xFFFF_FFFF	256B	256B	Reads return zeros, writes have no effect

2.2.3 Flash on Microcontrollers

The TMS570LC43x microcontrollers support up to 4 MB of flash for use as program memory. The microcontrollers also support a separate 128kB flash bank for use as emulated EEPROM.

Refer to the device data manual for electrical and timing specifications related to the flash module.

2.2.3.1 Flash Bank Sectoring Configuration

The bank is divided into multiple sectors. A flash sector is the smallest region in the flash bank that must be erased. The sectoring configuration of each flash bank is shown in Table 2-3.

- The Flash banks are 288-bit wide bank with ECC support.
- The flash bank7 can be programmed while executing code from flash bank0.
- Code execution is not allowed from flash bank7.

Table 2-3. Flash Memory Banks and Sectors

Sector Number	Sector Size	Low Address	High Address
Bank 0: 2.0 MBytes			
0	16K Bytes	0x0000_0000	0x0000_3FFF
1	16K Bytes	0x0000_4000	0x0000_7FFF
2	16K Bytes	0x0000_8000	0x0000_BFFF
3	16K Bytes	0x0000_C000	0x0000_FFFF
4	16K Bytes	0x0001_0000	0x0001_3FFF
5	16K Bytes	0x0001_4000	0x0001_7FFF
6	32K Bytes	0x0001_8000	0x0001_FFFF
7	128K Bytes	0x0002_0000	0x0003_FFFF
8	128K Bytes	0x0004_0000	0x0005_FFFF
9	128K Bytes	0x0006_0000	0x0007_FFFF
10	256K Bytes	0x0008_0000	0x000B_FFFF
11	256K Bytes	0x000C_0000	0x000F_FFFF
12	256K Bytes	0x0010_0000	0x0013_FFFF
13	256K Bytes	0x0014_0000	0x0017_FFFF
14	256K Bytes	0x0018_0000	0x001B_FFFF
15	256K Bytes	0x001C_0000	0x001F_FFFF
Bank 1: 2.0 Mbytes			
0	128K Bytes	0x0020_0000	0x0021_FFFF
1	128K Bytes	0x0022_0000	0x0023_FFFF
2	128K Bytes	0x0024_0000	0x0025_FFFF
3	128K Bytes	0x0026_0000	0x0027_FFFF
4	128K Bytes	0x0028_0000	0x0029_FFFF
5	128K Bytes	0x002A_0000	0x002B_FFFF
6	128K Bytes	0x002C_0000	0x002D_FFFF
7	128K Bytes	0x002E_0000	0x002F_FFFF
8	128K Bytes	0x0030_0000	0x0031_FFFF
9	128K Bytes	0x0032_0000	0x0033_FFFF
10	128K Bytes	0x0034_0000	0x0035_FFFF
11	128K Bytes	0x0036_0000	0x0037_FFFF
12	128K Bytes	0x0038_0000	0x0039_FFFF
13	128K Bytes	0x003A_0000	0x003B_FFFF
14	128K Bytes	0x003C_0000	0x003D_FFFF
15	128K Bytes	0x003E_0000	0x003F_FFFF
Bank 7: 128 kBytes			
0	4K Bytes	0xF020_0000	0xF020_0FFF
:	:	:	:
31	4K Bytes	0xF021_F000	0xF021_FFFF

2.2.3.2 ECC Protection for Flash Accesses

The TMS570LC43x microcontrollers protect all accesses to the on-chip level 2 flash memory by dedicated Single-Bit Error Correction Double-Bit Error Detection (SECDED) logic.

The access to the program memory – flash bank 0, 1 and 7 are protected by SECDED logic implemented inside the ARM Cortex-R5F CPU.

The SECDED logic implementation uses Error Correction Codes (ECC) for correcting single-bit errors and for detecting multiple-bit errors in the values read from the flash arrays. There is an 8-bit ECC for every 64 bits of data. The ECC for the level 2 flash memory contents needs to be calculated by an external tool such as nowECC. The ECC can then be programmed into the flash array along with the actual application code.

The ECC for the flash array is stored in the flash itself, and is mapped to a region starting at 0xF0400000 for the main flash bank 0 and 1, and to a region starting at 0xF0100000 for the EEPROM emulation flash bank 7.

NOTE: The SECDED logic inside the CPU is permanently enabled for the AXI-M and AXI-S interfaces.

Code Example for Enabling ECC Protection for Main Flash Accesses:

When the CPU detects an ECC single-, or double-bit error on a read from the flash memory, it signals this on a dedicated “*Event*” bus. This event bus signaling is also not enabled by default and must be enabled by the application. The below code example can be used to enable the CPU event signaling.

```
MRC p15,#0,r1,c9,c12,#0       ;Enabling Event monitor states
ORR r1, r1, #0x00000010
MCR p15,#0,r1,c9,c12,#0       ;Set 4th bit ('X') of PMNC register
MRC p15,#0,r1,c9,c12,#0
```

The ECC error events exported onto the *Event* bus is first captured by the Error Profiling Controller (EPC) module and in turn generates error signals that are input to the central Error Signaling Module (ESM).

2.2.3.3 Error Profiling Module (EPC)

The main idea of EPC is to enable the system to tolerate a certain amount of ECC correctable errors on the same address repeatedly in the memory system with minimal runtime overhead. EPC will record different single-bit error addresses in a Content Addressable Memory (CAM). If a correctable ECC error is generated on a repeating address, the EPC will not raise an error to ESM module. This tolerance avoids the application to handle the same error when the code is in a repeating loop. There are 4correctable error interfaces implemented in EPC to capture correctable errors from 4 different sources. There are also 2 uncorrectable error interfaces implemented in EPC to capture uncorrectable errors from 2 different sources. Main features of EPC are:

- Capture the addresses of the correctable ECC faults from different sources such as CPU cores, L2 SRAM and interconnect into a 32-entry CAM (Content Addressable Memory)
- For correctable faults, the error handling depends on the following conditions:
  - if the incoming address is already in the 32-entry CAM, discard the fail. No error generated to ESM.
  - if the address is not in the CAM list, and the CAM has empty entries, add the address into the CAM list. In addition, raise the error signal to the ESM group 1 if enabled.
  - if the address is not in the CAM list, and the CAM has no empty entries, always raise the error signal to the ESM group 1.
- A 4-entry FIFO to store the correctable error events and addresses for each channel interface.
- For uncorrectable faults, capture the address and assert error signal to the ESM group 2.

Each EPC interface corresponds to a bit field in some of the EPC registers. Table 2-4 shows only those registers that associate the bits to a specific interface for this device. See EPC chapter for the full list of registers.

Figure 2-4. EPC Integration Diagram

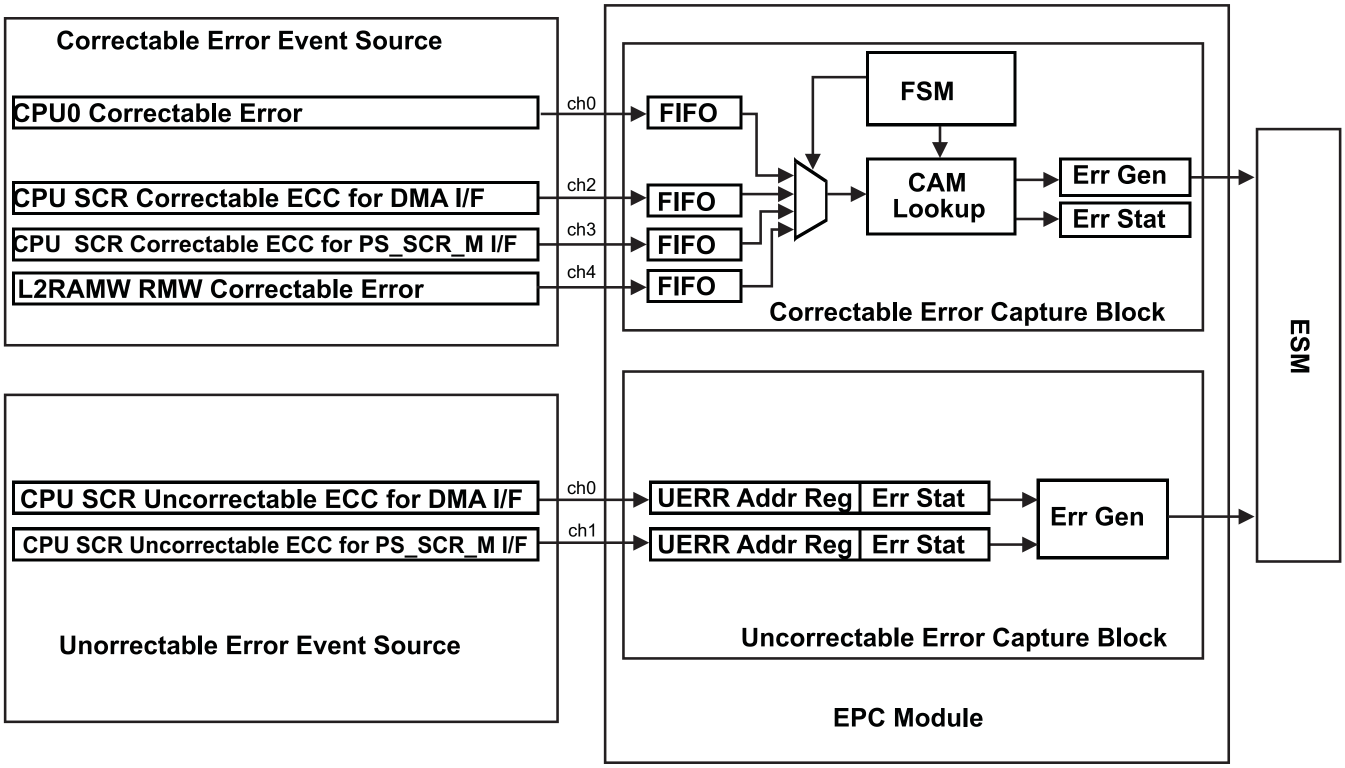

Correctable Error Event Source
CPU0 Correctable Error
CPU SCR Correctable ECC for DMA I/F
CPU SCR Correctable ECC for PS_SCR_M I/F
L2RAMW RMW Correctable Error
ch0
ch2
ch3
ch4
FIFO
FIFO
FIFO
FIFO
FSM
CAM Lookup
Err Gen
Err Stat
Correctable Error Capture Block
ESM
CPU SCR Uncorrectable ECC for DMA I/F
CPU SCR Uncorrectable ECC for PS_SCR_M I/F
Unorrectable Error Event Source
ch0
ch1
UERR Addr Reg
Err Stat
UERR Addr Reg
Err Stat
Err Gen
Uncorrectable Error Capture Block
EPC Module

Table 2-4. EPC Registers Bit Mapping

Address Offset	Register Name	Bit #	Error Source	Remark
8h	UERRSTAT	0	Uncorrectable ECC for DMA interface	• Bit associates with the Uncorrectable ECC error detected by the CPU Interconnect Subsystem for the DMA interface • See Interconnect chapter for details on the ECC generation and evaluation for DMA interface
		1	Uncorrectable ECC for PS_SCR_M interface	• Bit associates with the Uncorrectable ECC error detected by the CPU Interconnect Subsystem for the PS_SCR_M interface • See Interconnect chapter for details on the ECC generation and evaluation for DMA interface
10h	FIFOFULLSTAT	0	CPU Correctable ECC error	• Bit associates with the FIFO full status for the interface that is used to capture the CPU correctable error event • Correctable error event exported by CPU's event bus.
		1	Reserved	
		2	Correctable ECC for DMA interface	• Bit associates with the FIFO full status for the interface that is used to capture the DMA correctable error event • Correctable error event detected by the CPU Interconnect Subsystem for the DMA PortA interface.
		3	Correctable ECC for PS_SCR_M interface	• Bit associates with the FIFO full status for the interface that is used to capture the PS_SCR_M correctable error event • Correctable error event detected by the CPU Interconnect Subsystem for the PS_SCR_M interface.
		4	Correctable ECC error from L2 SRAM	• Bit associates with the FIFO full status for the interface that is used to capture the L2 SRAM correctable error event • Correctable error event detected by the L2 SRAM wrapper during the read phase of a Read-Modify-Write operation due to a less than 64-bit write from the bus master.
14h	OVRFLWSTAT	0	CPU Correctable ECC error	• Bit associates with the FIFO overflow status for the interface that is used to capture the CPU correctable error event
		1	Reserved	
		2	Correctable ECC for DMA interface	• Bit associates with the FIFO overflow status for the interface that is used to capture the DMA correctable error event
		3	Correctable ECC for PS_SCR_M interface	• Bit associates with the FIFO overflow status for the interface that is used to capture the PS_SCR_M correctable error event
		4	Correctable ECC error from L2 SRAM	• Bit associates with the FIFO overflow status for the interface that is used to capture the L2 SRAM correctable error event
20h	UERRADDR0	31:0	Uncorrectable ECC for DMA interface	• Uncorrectable error address register for the DMA interface
24h	UERRADDR1	31:0	Uncorrectable ECC for PS_SCR_M interface	• Uncorrectable error address register for the PS_SCR_M interface

2.2.4 On-Chip SRAM

Several SRAM modules are implemented on the device to support the functionality of the modules included.

Reads from both the level 1 and level 2 SRAM are protected by ECC calculated inside the CPU. Reads from all other memories are protected by either the parity with configurable odd or even parity scheme or ECC that is evaluated in parallel with the actual read.

The TMS570LC43x microcontrollers are targeted towards safety-critical applications, and it is critical for any failures in the on-chip SRAM modules to be identified before these modules are used for safety-critical functions. These microcontrollers support a Programmable Built-In Self-Test (PBIST) mechanism that is used to test each on-chip SRAM module for faults. The PBIST is usually run on device start-up as it is a destructive test and all contents of the tested SRAM module are overwritten during the test.

The microcontrollers also support a hardware-based auto-initialization of on-chip SRAM modules. This process also takes into account the read protection scheme implemented for each SRAM module – ECC or parity.

TI recommends that the PBIST routines be executed on the SRAM modules prior to the auto-initialization. The following sections describe these two processes.

2.2.4.1 PBIST RAM Grouping and Algorithm Mapping For On-Chip SRAM Modules

Table 2-5 shows the groupings of the various on-chip memories for PBIST. It also lists the memory types and their assigned RAM Group Select (RGS) and Return Data Select (RDS). Refer to the PBIST chapter for more details on the usage of the RGS and RDS information.

Table 2-5. PBIST Memory Grouping

Module	RAM Group #	RGS	RDS	Memory Type
PBIST_ROM	1	1	1	ROM
STC1_1_ROM_R5	2	14	1	ROM
STC1_2_ROM_R5	3	14	2	ROM
STC2_ROM_N2HET	4	15	1	ROM
AWM1	5	2	1	Two-port
DCAN1	6	3	1 to 6	Two-port
DCAN2	7	4	1 to 6	Two-port
DMA	8	5	1 to 6	Two-port
HTU1	9	6	1 to 6	Two-port
MIBSPI1	10	8	1 to 4	Two-port
MIBSPI2	11	9	1 to 4	Two-port
MIBSPI3	12	10	1 to 4	Two-port
N2HET1	13	11	1 to 12	Two-port
VIM	14	12	1, 2	Two-port
Reserved	15	13	1, 2	Two-port
RTP	16	16	1 to 12	Two-port
ATB	17	17	1 to 16	Two-port
AWM2	18	18	1	Two-port
DCAN3	19	19	1 to 6	Two-port
DCAN4	20	20	1 to 6	Two-port
HTU2	21	21	1 to 6	Two-port
MIBSPI4	22	22	1 to 4	Two-port
MIBSPI5	23	23	1 to 4	Two-port
N2HET2	24	24	1 to 12	Two-port
FTU	25	25	1	Two-port
FRAY_INBUF_OUTBUF	26	26	1 to 8	Two-port

Table 2-5. PBIST Memory Grouping (continued)

Module	RAM Group #	RGS	RDS	Memory Type
CPGMAC_STATE_RXADDR	27	27	1 to 3	Two-port
CPGMAC_STAT_FIFO	28	27	4 to 6	Two-port
L2RAMW	29	7	1	Single-port
			6	Single-port
L2RAMW	30	32	1	Single-port
			6	Single-port
			11	Single-port
			16	Single-port
			21	Single-port
			26	Single-port
R5_ICACHE	31	40	1	Single-port
			6	Single-port
			11	Single-port
			16	Single-port
R5_DCACHE	32	41	1	Single-port
			6	Single-port
			11	Single-port
			16	Single-port
			21	Single-port
			26	Single-port
Reserved	33	43	1	Single-port
			6	Single-port
			11	Single-port
			16	Single-port
Reserved	34	44	1	Single-port
			6	Single-port
			11	Single-port
			16	Single-port
			21	Single-port
			26	Single-port
FRAY_TRBUF_MSGRAM	35	26	9 to 11	Single-port
CPGMAC_CPPI	36	27	7	Single-port
R5_DCACHE_Dirty	37	42	2	Single-port
Reserved	38	45	2	Single-port

Table 2-6 maps the different algorithms supported in application mode for the RAM groups. The table also lists the background pattern options available for each algorithm.

Table 2-6. PBIST Algorithm Mapping

Sr. No.	ALGO Register Value	Algorithm	Memories Under Test	Available Background Patterns	Valid RAM Groups	Valid RINFOL/RINFOU Register Value
1	0x00000001	triple_read_slow_read	ROM		1,2,3,4	0x0000000F/ 0x00000000
2	0x00000002	triple_read_fast_read	ROM		1,2,3,4	0x0000000F/ 0x00000000
3	0x00000004	march13n	Two-port	0x00000000, 0x96699669, 0x0F0F0F0F, 0xAA55AA55, 0xC3C3C3C3	5,6,7,8,9,10,11,12,13 , 14,16,17,18,19,20,21 , 22,23,24,25,26,27,28	0x0FFFBFF0/ 0x00000000
4	0x00000008	march13n	Single-port	0x00000000, 0x96699669, 0x0F0F0F0F, 0xAA55AA55, 0xC3C3C3C3	29,30,31,32,35,36,37	0xF0000000/ 0x0000001C

NOTE: Recommended Memory Test Algorithm

March13 is the most recommended algorithm for the memory self-test.

For GCLK1 = 300 MHz, HCLK = 150 MHz, VCLK = 75 MHz, PBIST ROM_CLK = 75 MHz, the March13 algorithm takes about 29.08 ms to run on all on-chip SRAMs.

NOTE: PBIST ROM_CLK can be prescaled from GCLK1 via ROM_DIV bits of the MSTGCR register. The valid ratio is either /1, /2 or /4 or /8. See Section 2.5.1.20 for detail. Maximum PBIST ROM_CLK frequency supported is 82.5MHz.

2.2.4.2 Auto-Initialization of On-Chip SRAM Modules

The device system provides the capability to perform a hardware initialization on most memories on the system bus and on the peripheral bus. The memory used for the FlexRay message objects is ECC protected but is not directly CPU addressable, hence there is no memory auto-initialization support for this memory.

The intent of having the hardware initialization is to program the memory arrays with error detection capability to a known state based on their error detection scheme – odd/even parity or ECC. For example, the contents of the CPU level 2 SRAM after power-on reset is unknown. A hardware auto-initialization can be started so that there is no ECC error.

NOTE: Effect of ECC or Parity on Memory Auto-Initialization

The ECC or parity should be enabled on the RAMs before hardware auto-initialization starts if parity or ECC is being used.

Auto-Initialization Sequence:

1. Enable the global hardware memory initialization key by programming 0xA into MINITGCR[3:0], the Memory Initialization Key field (MINITGENA) of the Memory Hardware Initialization Global Control Register (MINITGCR) register.
2. Select the module on which the memory hardware initialization has to be performed by programming the appropriate value into the MSINENA(31–0) bits in the MSINENA register. See Table 2-7.
3. If the global auto-initialization scheme is enabled, the corresponding module will initialize its memories based on its memory error checking scheme (even parity or odd parity or ECC).
4. When the memory initialization is complete, the module will signal “memory initialization done”, which sets the corresponding bit in the system module MIDONE field of the MINISTAT register to indicate the completion of its memory initialization.
5. When the memory hardware initialization completes for all modules, (indicated by each module’s MIDONE bit being set), the memory hardware initialization done bit (MINIDONE) is set in the MSTCGSTAT register.

Figure 2-5. Hardware Memory Initialization Protocol

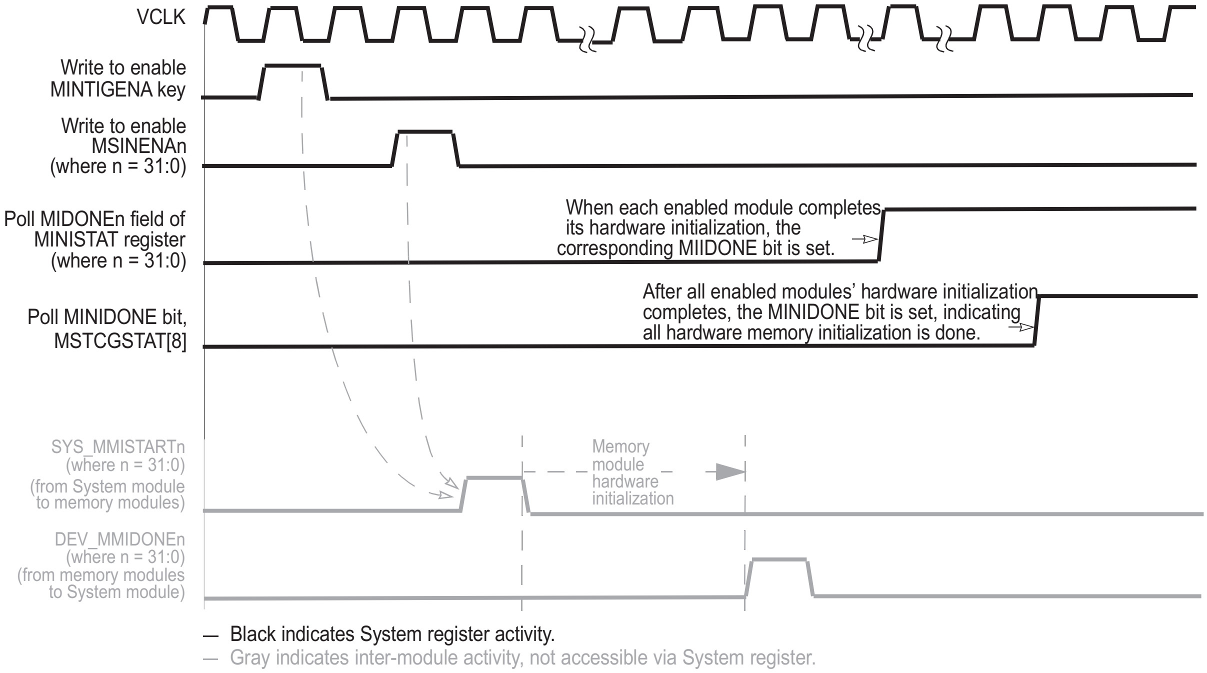

VCLK
Write to enable MINTIGENA key
Write to enable MSINENAn (where n = 31:0)
Poll MIDONEn field of MINISTAT register (where n = 31:0)
When each enabled module completes its hardware initialization, the corresponding MIIDONE bit is set.
Poll MINIDONE bit, MSTCGSTAT[8]
After all enabled modules' hardware initialization completes, the MINIDONE bit is set, indicating all hardware memory initialization is done.
SYS_MMISTARTn (where n = 31:0) (from System module to memory modules)
Memory module hardware initialization
DEV_MMIDONEn (where n = 31:0) (from memory modules to System module)
— Black indicates System register activity.
— Gray indicates inter-module activity, not accessible via System register.

Table 2-7. Memory Initialization Select Mapping[1][2]

Connecting Module	Memory Protection Scheme	Address Range		SYS.MSINENA Register Bit #	L2RAMW.MEMINT_ENA Register Bit #[3]
		Base Address	Ending Address		
L2 SRAM	ECC	0x08000000	0x0800FFFF	0	0
L2 SRAM	ECC	0x08010000	0x0801FFFF	0	1
L2 SRAM	ECC	0x08020000	0x0802FFFF	0	2
L2 SRAM	ECC	0x08030000	0x0803FFFF	0	3
L2 SRAM	ECC	0x08040000	0x0804FFFF	0	4
L2 SRAM	ECC	0x08050000	0x0805FFFF	0	5
L2 SRAM	ECC	0x08060000	0x0806FFFF	0	6
L2 SRAM	ECC	0x08070000	0x0807FFFF	0	7
MIBSPI5 RAM[4]	ECC	0xFF0A0000	0xFF0BFFFF	12	n/a
MIBSPI4 RAM[4]	ECC	0xFF060000	0xFF07FFFF	19	n/a
MIBSPI3 RAM[4]	ECC	0xFF0C0000	0xFF0DFFFF	11	n/a
MIBSPI2 RAM[4]	ECC	0xFF080000	0xFF09FFFF	18	n/a
MIBSPI1 RAM[4]	ECC	0xFF0E0000	0xFF0FFFFF	7	n/a
DCAN4 RAM	ECC	0xFF180000	0xFF19FFFF	20	n/a
DCAN3 RAM	ECC	0xFF1A0000	0xFF1BFFFF	10	n/a
DCAN2 RAM	ECC	0xFF1C0000	0xFF1DFFFF	6	n/a
DCAN1 RAM	ECC	0xFF1E0000	0xFF1FFFFF	5	n/a
MIBADC2 RAM	Parity	0xFF3A0000	0xFF3BFFFF	14	n/a
MIBADC1 RAM	Parity	0xFF3E0000	0xFF3FFFFF	8	n/a
NHET2 RAM	Parity	0xFF440000	0xFF45FFFF	15	n/a
NHET1 RAM	Parity	0xFF460000	0xFF47FFFF	3	n/a
HET TU2 RAM	Parity	0xFF4C0000	0xFF4DFFFF	16	n/a
HET TU1 RAM	Parity	0xFF4E0000	0xFF4FFFFF	4	n/a
DMA RAM	ECC	0xFFF80000	0xFFF80FFF	1	n/a
VIM RAM	ECC	0xFFF82000	0xFFF82FFF	2	n/a
FlexRay TU RAM	ECC	0xFF500000	0xFF51FFFF	13	n/a

[1] If parity protection is enabled for the peripheral SRAM modules, then the parity bits will also be initialized along with the SRAM modules.

[2] If ECC protection is enabled for the CPU data RAM or peripheral SRAM modules, then the auto-initialization process also initializes the corresponding ECC space.

[3] The level 2 SRAM range from 128kB to 512kB is divided into 6 memory regions. Each region has an associated control bit to enable auto-initialization.

[4] The MibSPIx modules perform an initialization of the transmit and receive RAMs as soon as the multi-buffered mode is enabled. This is independent of whether the application has already initialized these RAMs using the auto-initialization method or not. The MibSPIx modules need to be released from reset by writing 1 to their SPIGCR0 registers before starting auto-initialization on their respective RAMs.

2.3 Exceptions

An “Exception” is an event that makes the processor temporarily halt the normal flow of program execution, for example, to service an interrupt from a peripheral. Before attempting to handle an exception, the processor preserves the critical parts of the current processor state so that the original program can resume when the handler routine has finished.

The following sections describe three exceptions – Reset, Abort and the System Software Interrupts.

For complete details on all exceptions, refer to the ARM® Cortex®-R5F Technical Reference Manual.

2.3.1 Resets

The TMS570LC43x microcontroller can be reset by either of the conditions described in Table 2-8. Each reset condition is indicated in the System Exception Status Register (SYSESR).

The device nRST terminal is an I/O. It can be driven low by an external circuit to force a warm reset on the microcontroller. This terminal will be driven low as an output for a minimum of 32 peripheral clock (VCLK) cycles for any device system reset condition. As a result the EXTRST bit in the SYSESR register, SYSESR[3], gets set for all reset conditions listed in Table 2-8. The nRST is driven low as an output for a longer duration during device power-up or whenever the power-on reset (nPORRST) is driven low externally. Refer the device data manual for the electrical and timing specifications for the nRST.

Table 2-8. Causes of Resets

Condition	Description
Driving nPORRST pin low externally	Cold reset, or power-on reset. This reset signal is typically driven by an external voltage supervisor. This reset is flagged by the PORST bit in the SYSESR register, SYSESR[15].
Voltage Monitor reset	The microcontroller has an embedded voltage monitor that generates a power-on reset when the core voltage gets out of a valid range, or when the I/O voltage falls below a threshold. This reset is also flagged by the PORST bit in the SYSESR register, SYSESR[15]. **Note:** The voltage monitor is not an alternative for an external voltage supervisor.
Driving nRST pin low externally	Warm reset. This reset input is typically used in a system with multiple ICs and which requires that the microcontroller also gets reset whenever the other IC detects a fault condition. This reset is flagged by the EXTRST bit in the SYSESR, register SYSESR[3].
Oscillator failure	This reset is generated by the system module when the clock monitor detects an oscillator fail condition. Whether or not a reset is generated is also dictated by a register in the system module. This reset is flagged by the OSCRST bit in the SYSESR register, SYSESR[14].
Software reset	This reset is generated by the application software writing a 1 to bit 15 of System Exception Control Register (SYSECR) or a 0 to bit 14 of SYSECR. It is typically used by a bootloader type of code that uses a software reset to allow the code execution to branch to the application code once it is programmed into the program memory. This reset is flagged by the SWRST bit in the SYSESR register, SYSESR[4].
CPU reset	This reset is generated by the CPU self-test controller (LBIST) or by changing the memory protection (MMU/MPU) configuration in the CPURSTCR register or after the CPU Interconnect Subsystem self test. This reset is flagged by the CPURST bit in the SYSESR register, SYSESR[5].
Debug reset	The ICEPICK logic implemented on the microcontroller allows a system reset to be generated via the debug logic. This reset is flagged by the DBGRST bit in the SYSESR register, SYSESR[13].
Watchdog reset	This reset is generated by the digital windowed watchdog (DWWD) module on the microcontroller. The DWWD can generate a reset whenever the watchdog service window is violated. This reset is flagged by the WDRST bit in the SYSESR register, SYSESR[13].

2.3.2 Aborts

When the ARM Cortex-R5F processor's memory system cannot complete a memory access successfully, an abort is generated. An error occurring on an instruction fetch generates a ***prefetch abort***. Errors occurring on data accesses generate ***data aborts***. Aborts are also categorized as being either ***precise*** or ***imprecise***.

2.3.2.1 Prefetch Aborts

When a Prefetch Abort (PABT) occurs, the processor marks the prefetched instruction as invalid, but does not take the exception until the instruction is to be executed. If the instruction is not executed, for example because a branch occurs while it is in the pipeline, the abort does not take place.

All prefetch aborts are precise aborts.

2.3.2.2 Data Aborts

An error occurring on a data memory access can generate a data abort. If the instruction generating the memory access is not executed, for example, because it fails its condition codes, or is interrupted, the data abort does not take place.

A Data Abort (DABT) can be either precise or imprecise, depending on the type of fault that caused it.

2.3.2.3 Precise Aborts

A precise abort, also known as a synchronous abort, is one for which the exception is guaranteed to be taken on the instruction that generated the aborting memory access. The abort handler can use the value in the Link Register (r14_abt) to determine which instruction generated the abort, and the value in the Saved Program Status Register (SPSR_abt) to determine the state of the processor when the abort occurred.

2.3.2.4 Imprecise Aborts

An imprecise abort, also known as an asynchronous abort, is one for which the exception is taken on a later instruction to the instruction that generated the aborting memory access. The abort handler cannot determine which instruction generated the abort, or the state of the processor when the abort occurred. Therefore, imprecise aborts are normally fatal.

Imprecise aborts can be generated by store instructions to normal-type or device-type memory. When the store instruction is committed, the data is normally written into a buffer that holds the data until the memory system has sufficient bandwidth to perform the write access. This gives read accesses higher priority. The write data can be held in the buffer for a long period, during which many other instructions can complete. If an error occurs when the write is finally performed, this generates an imprecise abort.

The TMS570LC43x microcontroller architecture applies techniques at the system level to mitigate the impact of imprecise aborts. System level adoption of write status sidebands to the data path allow bus masters to comprehend imprecise aborts, turning them into precise aborts. In cases where this approach is not feasible, buffering bridges or other sources of imprecision may build a FIFO of current transactions such that an imprecise abort may be registered at the point of imprecision for later analysis.

Masking Of Imprecise Aborts:

The nature of imprecise aborts means that they can occur while the processor is handling a different abort. If an imprecise abort generates a new exception in such a situation, the banked link register (R14_abt) and the Saved Processor Status Register (SPSR_abt) values are overwritten. If this occurs before the data is pushed to the stack in memory, the state information about the first abort is lost. To prevent this from happening, the Current Processor Status Register (CPSR) contains a mask bit to indicate that an imprecise abort cannot be accepted, the A-bit. When the A-bit is set, any imprecise abort that occurs is held pending by the processor until the A-bit is cleared, when the exception is actually taken. The A-bit is automatically set when abort, IRQ or FIQ exceptions are taken, and on reset. The application must only clear the A-bit in an abort handler after the state information has either been stacked to memory, or is no longer required.

NOTE: Default Behavior for Imprecise Aborts

The A-bit in the CPSR is set by default. This means that no imprecise abort exception will occur. The application must enable imprecise abort exception generation by clearing the A-bit of the CPSR.

2.3.2.5 Conditions That Generate Aborts

An Abort is generated under the following conditions on the TMS570LC43x microcontrollers.

- Access to an illegal address (a non-implemented address)
- Access to a protected address (protection violation)
- Parity / ECC / Time-out Error on a valid access

Illegal Addresses:

The illegal addresses and the responses to an access to these addresses are defined in Table 2-2.

Addresses Protected By MPU:

For more details on the MPU configuration, refer to the ARM® Cortex®-R5F Technical Reference Manual.

A memory access violation is logged as a permission fault in the CPU's fault status register and the virtual address of the access is logged into the CPU's fault address register.

Protection of Peripheral Register and Memory Frames:

Accesses to the peripheral register and memory frames can be protected either by configuring the MPU or by configuring the Peripheral Central Resource (PCR) controller registers.

The PCR module PPROTSETx registers contain one bit per peripheral select quadrant. These bits define the access permissions to the peripheral register frames. If the CPU attempts to write to a peripheral register for which it does not have the correct permissions, a protection violation is detected and an Abort occurs.

Some modules also enforce register updates to only be allowed when the CPU is in a privileged mode of operation. If the CPU writes to these registers in user mode, the writes are ignored.

The PCR module PMPROTSETx registers contain one bit per peripheral memory frame. These bits define the access permissions to the peripheral memory frames. If the CPU attempts to write to a peripheral memory for which it does not have the correct permissions, a protection violation is detected and an Abort occurs.

NOTE: No Access Protection for Reads

The PCR PPROTSETx and PMPROTSETx registers protect the peripheral registers and memories against illegal writes by the CPU. The CPU can read from the peripheral registers and memories in both user and privileged modes.

2.3.3 System Software Interrupts

The system module provides the capability of generating up to four software interrupts. A software interrupt is generated by writing the correct key value to either of the four System Software Interrupt Registers (SSIRx). The SSI registers also allow the application to provide a label for that software interrupt. This label is an 8-bit value that can then be used by the interrupt service routine to perform the required task based on the value provided. The source of the system software interrupt is reflected in the system software interrupt vector (SSIVEC) register. The pending interrupt flag is captured in SSIF register.

NOTE: The SSIRx, SSIVEC and SSIF registers are banked registers. This allows the system module to support up to two CPUs for system software interrupt generation. Each CPU will have its own banked SSI registers. Both CPUs will see the SSI registers at the same address. The system module decodes the unique master ID corresponding to the CPU's access to the banked registers.

2.4 Clocks

This section describes the clocking structure of the TMS570LC43x microcontrollers.

2.4.1 Clock Sources

The devices support up to 7 clock sources. These are shown in Table 2-9. The electrical specifications as well as timing requirements for each of the clock sources are specified in the device data manual.

Table 2-9. Clock Sources

Clock Source #	Clock Source Name	Description
0	OSCIN	Main oscillator. This is the primary clock for the microcontroller and is the only clock that is input to the phase-locked loops. The oscillator frequency must be between 5 and 20 MHz.
1	PLL1	This is the output of the main PLL. The PLL is capable of modulating its output frequency in a controlled manner to reduce the radiated emissions.
2	Reserved	This clock source is not available and must not be enabled or used as source for any clock domain.
3	EXTCLKIN1	External clock input 1. A square wave input can be applied to this device input and used as a clock source inside the device.
4	LF LPO (Low-Frequency LPO) (CLK80K)	This is the low-frequency output of the internal reference oscillator. This is typically an 80 KHz signal (CLK80K) that is used by the real-time interrupt module for generating periodic interrupts to wake up from a low power mode.
5	HF LPO (High-Frequency LPO) (CLK10M)	This is the high-frequency output of the internal reference oscillator. This is typically a 10 MHz signal (CLK10M) that is used by the clock monitor module as a reference clock to monitor the main oscillator frequency.
6	PLL2	This is the output of the second PLL. There is no option of modulating this PLL’s output signal. This separate non-modulating PLL allows the generation of an asynchronous clock source that is independent of the CPU clock frequency.
7	EXTCLKIN2	External clock input 2. A square wave input can be applied to this device input and used as a clock source inside the device.

2.4.1.1 Enabling / Disabling Clock Sources

Each clock source can be independently enabled or disabled using the set of Clock Source Disable registers – CSDIS, CSDISSET and CSDISCLR.

Each bit in these registers corresponds to the clock source number indicated in Table 2-9. For example, setting bit 1 in the CSDIS or CSDISSET registers disables the PLL#1.

NOTE: Disabling the Main Oscillator or HF LPO

By default, the clock monitoring circuit is enabled and checks for the main oscillator frequency to be within a certain range using the HF LPO as a reference. If the main oscillator and/or the HF LPO are disabled with the clock monitoring still enabled, the clock monitor will indicate an oscillator fault. The clock monitoring must be disabled before disabling the main oscillator or the HF LPO clock source(s).

The clock source is only disabled once there is no active clock domain that is using that clock source. Also check the “Oscillator and PLL” user guide for more information on enabling / disabling the oscillator and PLL.

On the TMS570LC43x microcontrollers, the clock sources 0, 4, and 5 are enabled by default.

2.4.1.2 Checking for Valid Clock Sources

The application can check whether a clock source is valid or not by checking the corresponding bit to be set in the Clock Source Valid Status (CSVSTAT) register. For example, the application can check if bit 1 in CSVSTAT is set before using the output of PLL#1 as the source for any clock domain.

2.4.2 Clock Domains

The clocking on this device is divided into multiple clock domains for flexibility in control as well as clock source selection. There are 10 clock domains on this device. Each of these are described in Table 2-10.

Each of the control registers listed in Table 2-10 are defined in Section 2.5. The AC timing characteristics for each clock domain are specified in the device data manual.

Table 2-10. Clock Domains

Clock Domain	Clock Disable Bit	Default Source	Source Selection Register	Special Considerations
GCLK1	CDDIS.0	OSCIN	GHVSRC[3:0]	• This the main clock from which HCLK is divided down • In phase with HCLK • Is disabled separately from HCLK via the CDDISx registers bit 0 • Can be divided by 1 up to 8 when running CPU self-test (LBIST) using the CLKDIV field of the STCCLKDIV register at address 0xFFFFE108
HCLK	CDDIS.1	OSCIN	GHVSRC[3:0]	• Divided from GCLK1 via HCLKCNTL register • Allowable clock ratio from 1:1 to 4:1 • Is disabled via the CDDISx registers bit 1
VCLK	CDDIS.2	OSCIN	GHVSRC[3:0]	• Divided down from HCLK via CLKCNTL register • Can be HCLK/1, HCLK/2,... or HCLK/16 • Is disabled separately from HCLK via the CDDISx registers bit 2 • HCLK:VCLK2:VCLK must be integer ratios of each other
VCLK2	CDDIS.3	OSCIN	GHVSRC[3:0]	• Divided down from HCLK • Can be HCLK/1, HCLK/2,... or HCLK/16 • Frequency must be an integer multiple of VCLK frequency • Is disabled separately from HCLK via the CDDISx registers bit 3
VCLK3	CDDIS.8	OSCIN	GHVSRC[3:0]	• Divided down from HCLK • Can be HCLK/1, HCLK/2,... or HCLK/16 • HCLK:VCLK3 must be integer ratios of each other • Is disabled separately from HCLK via the CDDISx registers bit 8
VCLKA1	CDDIS.4	VCLK	VCLKASRC[3:0]	• Defaults to VCLK as the source • Is disabled via the CDDISx registers bit 4
VCLKA2	CDDIS.5	VCLK	VCLKASRC[3:0]	• Defaults to VCLK as the source • Is disabled via the CDDISx registers bit 5
VCLKA4	CDDIS.11	VCLK	VCLKACON1[19:16]	• Defaults to VCLK as the source • Is disabled via the CDDISx registers bit 11
VCLKA4_DIVR	VCLKACON1.20	VCLK	VCLKACON1[19:16]	• Divided down from VCLKA4 using the VCLKA4R field of the VCLKACON1 register • Frequency can be VCLKA4/1, VCLKA4/2, ..., or VCLKA4/8 • Default frequency is VCLKA4/2 • Is disabled separately via the VCLKACON1 register's VCLKA4_DIV_CDDIS bit, if the VCLKA4 is not already disabled

Table 2-10. Clock Domains (continued)

Clock Domain	Clock Disable Bit	Default Source	Source Selection Register	Special Considerations
RTICLK1	CDDIS.6	VCLK	RCLKSRC[3:0]	• Defaults to VCLK as the source • If a clock source other than VCLK is selected for RTICLK1, then the RTICLK1 frequency must be less than or equal to VCLK/3 • Application can ensure this by programming the RTI1DIV field of the RCLKSRC register, if necessary • Is disabled via the CDDISx registers bit 6

2.4.2.1 Enabling / Disabling Clock Domains

Each clock domain can be independently enabled or disabled using the set of Clock Domain Disable registers – CDDIS, CDDISSET, and CDDISCLR.

Each bit in these registers corresponds to the clock domain number indicated in Table 2-10. For example, setting bit 1 in the CDDIS or CDDISSET registers disables the HCLK clock domain. The clock domain will be turned off only when every module that uses the HCLK domain gives the “permission” for HCLK to be turned off.

All clock domains are enabled by default, or upon a system reset, or whenever a wake up condition is detected.

2.4.2.2 Mapping Clock Sources to Clock Domains

Each clock domain needs to be mapped to a valid clock source. There are control registers that allow an application to choose the clock sources for each clock domain.

- ***Selecting clock source for GCLK1, HCLK, and VCLKx domains***

The CPU clock (GCLK1), the system module clock (HCLK), and the peripheral bus clocks (VCLKx) all use the same clock source. This clock source is selected via the GHVSRC register. The default source for the GCLK1, HCLK, and VCLKx is the main oscillator. That is, after power up, the GCLK1 and HCLK are running at the OSCIN frequency, while the VCLKx frequency is the OSCIN frequency divided by 2.

- ***Selecting clock source for VCLKA1 and VCLKA2 domains***

The clock source for VCLKA1 and VCLKA2 domains is selected via the VCLKASRC register. The default source for the VCLKA1 and VCLKA2 domains is the VCLK.

- ***Selecting clock source for VCLKA4 domain***

The clock source for VCLKA4 domain is selected via the VCLKACON1 register. The default source for the VCLKA4 domain is the VCLK.

- ***Selecting clock source for RTICLK1 domain***

The clock source for RTICLK1 domain is selected via the RCLKSRC register. The default source for the RTICLK1 domain is the VCLK.

NOTE: Selecting a clock source for RTICLK1 that is not VCLK

When the application chooses a clock source for RTICLK1 domain that is not VCLK, then the application must ensure that the resulting RTICLK1 frequency must be less than or equal to VCLK frequency divided by 3. The application can configure the RTI1DIV field of the RCLKSRC register for dividing the selected clock source frequency by 1, 2, 4 or 8 to meet this requirement.

2.4.3 Low Power Modes

All clock domains are active in the normal operating mode. This is the default mode of operation. As described in Section 2.4.1.1 and Section 2.4.2.1, the application can choose to disable any particular clock source and domain that it does not plan to use. Also, the peripheral central resource controller (PCR) has control registers to enable / disable the peripheral clock (VCLK) for each peripheral select. This offers the application a large number of choices for enabling / disabling clock sources, or clock domains, or clocks to specific peripherals.

This section describes three particular low-power modes and their typical characteristics. They are not the only low-power modes configurable by the application, as just described.

Table 2-11. Typical Low-Power Modes

Mode Name	Active Clock Source(s)	Active Clock Domain(s)	Wake Up Options	Suggested Wake Up Clock Source(s)	Wake Up Time(wake up detected -to- CPU code execution start)
Doze	Main oscillator	RTICLK1	RTI interrupt, GIO interrupt, CAN message, SCI message	Main oscillator	Flash pump sleep -> active transition time + Flash bank sleep -> standby transition time + Flash bank standby -> active transition time
Snooze	LF LPO	RTICLK1	RTI interrupt, GIO interrupt, CAN message, SCI message	HF LPO	HF LPO warm start-up time + Flash pump sleep -> active transition time + Flash bank sleep -> standby transition time + Flash bank standby -> active transition time
Sleep	None	None	GIO interrupt, CAN message, SCI message	HF LPO	HF LPO warm start-up time + Flash pump sleep -> active transition time + Flash bank sleep -> standby transition time + Flash bank standby -> active transition time

2.4.3.1 Typical Software Sequence to Enter a Low-Power Mode

1. Disable all non-CPU bus masters so they do not carry out any further bus transactions.
2. Program the flash banks and flash pump fall-back modes to be “sleep”.

   The flash pump transitions from active to sleep mode only after all the flash banks have switched from active to sleep mode.
3. Disable the clock sources that are not required to be kept active.

   A clock source does not get disabled until all clock domains using that clock source are disabled first, or are configured to use an alternate clock source.
4. Disable the clock domains that are not required to be kept active.

   A clock domain does not get disabled until all modules using that clock domain “give their permission” for that clock domain to be turned off.
5. Idle the Cortex-R5F core.

   The ARM Cortex-R5F CPU has internal power management logic, and requires a dedicated instruction to be used in order to enter a low power mode. This is the Wait For Interrupt (WFI) instruction.

   When a WFI instruction is executed, the Cortex-R5F core flushes its pipeline, flushes all write buffers, and completes all pending bus transactions. At this time the core indicates to the system that the clock to the core can be stopped. This indication is used by the Global Clock Module (GCM) to turn off the CPU clock domain (GCLK1) if the CDDIS register bit 0 is set.

2.4.3.2 Special Considerations for Entry to Low Power Modes

Some bus master modules – DMA, High-End Timer Transfer Units (HTUx), FlexRay Transfer Unit (FTU), and Parameter Overlay Module (POM), can have ongoing transactions when the application wants to enter a low power mode to turn off the clocks to those modules. This is not recommended as it could leave the device in an unpredictable state. Refer to the individual module user guides for more information about the sequence to be followed to safely enter a low-power mode.

2.4.3.3 Selecting Clock Source Upon Wake Up

The domains for CPU clock (GCLK1), the system clock (HCLK) and the peripheral clock (VCLKx) use the same clock source selected via the GHVSRC field of the GHVSRC register. The GHVSRC register also allows the application to choose the clock source after wake up via the GHVWAKE field.

When a wake up condition is detected, if the selected wake up clock source is not already active, the global clock module (GCM) will enable this selected clock source, wait for it to become valid, and then use it for the GCLK1, HCLK, and VCLKx domains. The other clock domains VCLKAx and RTICLK1 retain the configuration for their clock source selection registers – VCLKASRC, VCLKACON1 and RCLKSRC.

2.4.4 Clock Test Mode

The TMS570LC43x microcontrollers support a test mode which allows a user to bring out several different clock sources and clock domains on to the ECLK1 terminal in addition to outputting the external clock. This is very useful information for debug purposes. Each clock source also has a corresponding clock source valid status flag in the Clock Source Valid Status (CSVSTAT) register. The clock source valid status flags can also be brought out on to the N2HET1[12] terminal in this clock test mode.

The clock test mode is controlled by the CLKTEST register in the system module register frame (see Section 2.5.1.31).

The clock test mode is enabled by writing 0x5 to the CLK_TEST_EN field.

The signal to be brought out on to the ECLK1 terminal is defined by the SEL_ECP_PIN field, and the signal to be brought out on to the N2HET1[12] terminal is defined by the SEL_GIO_PIN field. The choices for these selections are defined in Table 2-12.

Table 2-12. Clock Test Mode Options

SEL_ECP_PIN	Signal on ECLK	SEL_GIO_PIN	Signal on N2HET1[12]
00000	Oscillator clock	0000	Oscillator Valid Status
00001	PLL1 clock output	0001	PLL1 Valid Status
00010	Reserved	0010	Reserved
00011	EXTCLKIN1	0011	Reserved
00100	Low-frequency LPO (Low-Power Oscillator) clock [CLK80K]	0100	Reserved
00101	High-frequency LPO (Low-Power Oscillator) clock [CLK10M]	0101	HF LPO Clock Output Valid Status [CLK10M]
00110	PLL2 clock output	0110	PLL2 Valid Status
00111	EXTCLKIN2	0111	Reserved
01000	GCLK1	1000	LF LPO Clock Output Valid Status [CLK80K]
01001	RTI1 Base	1001	Oscillator Valid Status
01010	Reserved	1010	Oscillator Valid Status
01011	VCLKA1	1011	Oscillator Valid Status
01100	VCLKA2	1100	Oscillator Valid Status
01101	Reserved	1101	Reserved
01110	VCLKA4_DIVR	1110	VCLKA4
01111	Flash HD Pump Oscillator	1111	Oscillator Valid Status
10000	Reserved		
10001	HCLK		
10010	VCLK		
10011	VCLK2		
10100	VCLK3		
10101-10110	Reserved		
10111	EMAC clock output		
11000-11111	Reserved		

2.4.5 Embedded Trace Macrocell (ETM-R5)

The TMS570LC43x microcontrollers contain an ETM-R5 module with a 32-bit internal data port. The ETM-R5 module is connected to a Trace Port Interface Unit (TPIU) with a 32-bit data bus; the TPIU provides a 35-bit (32-bit data and 3-bit control) external interface for trace. The ETM-R5 is CoreSight compliant and follows the ETM v3 specification. For more details on the ETM-R5 specification, refer to the Embedded Trace Macrocell Architecture Specification.

The ETM clock source is selected as either VCLK or the external ETMTRACECLKIN pin. The selection is done by the EXTCTLOUT control bits of the TPIU EXTCTL_Out_Port register. The address of this register is TPIU base address + 0x404.

Before you begin accessing TPIU registers, the TPIU should be unlocked via the CoreSight key and 1h or 2h should be written to this register.

Figure 2-6. EXTCTL_Out_Port Register [offset = 404h]

<table>
<tr><td>31</td><td>16</td></tr>
<tr><td colspan="2">Reserved</td></tr>
<tr><td colspan="2">R-0</td></tr>
</table>

<table>
<tr><td>15</td><td>2</td><td>1</td><td>0</td></tr>
<tr><td colspan="2">Reserved</td><td colspan="2">EXTCTLOUT</td></tr>
<tr><td colspan="2">R-0</td><td colspan="2">R/W-0</td></tr>
</table>

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 2-13. EXTCTL_Out_Port Register Field Descriptions

Bit	Field	Value	Description
31-2	Reserved	0	Reads return 0. Writes have no effect.
1-0	EXTCTLOUT		EXTCTL output control.
		0	Tied-zero
		1h	VCLK
		2h	ETMTRACECLKIN
		3h	Tied-zero

2.4.6 Safety Considerations for Clocks

The TMS570LC43x microcontrollers are targeted for use in several safety-critical applications. The following sections describe the internal or external monitoring mechanisms that detect and signal clock source failures.

2.4.6.1 Oscillator Monitor

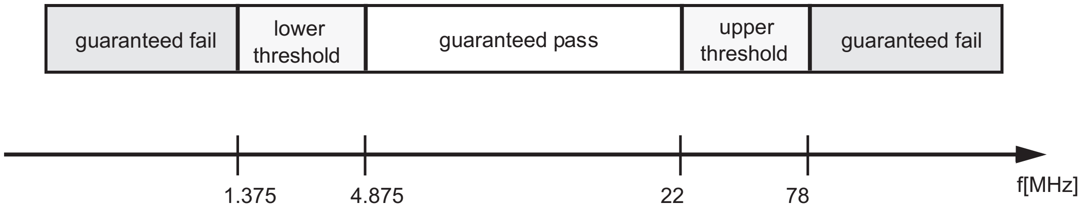
The oscillator clock frequency is monitored by a dedicated circuitry called CLKDET using the HF LPO as the reference clock. The CLKDET flags an oscillator fail condition whenever the OSCIN frequency falls outside of a range which is defined by the HF LPO frequency.

The valid OSCIN range is defined as a minimum of $f_{(HF\ LPO)}$ / 4 to a maximum of $f_{(HF\ LPO)}$ × 4.

The untrimmed HF LPO frequency on this device can range from 5.5 MHz to 19.5 MHz. This results in a valid OSCIN frequency range depicted in Figure 2-7.

The application can select the device response to an oscillator fail indication. Refer to Chapter 14 for more details on the oscillator monitoring and the system response choices.

Figure 2-7. LPO and Clock Detection, Untrimmed HF LPO



2.4.6.2 PLL Slip Detector

Both the PLL macros implemented on the microcontrollers have an embedded slip detection circuit. A PLL slip is detected by the slip detector under the following conditions:

1. Reference cycle slip, RFSLIP — the output clock is running *too fast* relative to the reference clock
2. Feedback cycle slip, FBSLIP — the output clock is running *too slow* relative to the reference clock

The device also includes optional filters that can be enabled before a slip indication from the PLL is actually logged in the system module Global Status Register (GLBSTAT). Also, once a PLL slip condition is logged in the system module global status register, the application can choose the device's response to the slip indication. Refer to Chapter 14 for more details on PLL slip and the system response choices.

2.4.6.3 External Clock Monitor

The microcontrollers support two terminals called ECLK1 and ECLK2 – External Clock, which are used to output a slow frequency which is divided down from the device system clock frequency. An external circuit can monitor the ECLK1 and/or ECLK2 frequency in order to check that the device is operating at the correct frequency.

The frequency of the signal output on the ECLKx pin can be divided down by 1 to 65536 from the peripheral clock (VCLK) frequency using the External Clock Prescaler Control Register (ECPCNTL) for ECLK1 and ECPCNTL1 for ECLK2. The actual clock output on ECLK1 is enabled by setting the ECP CLK FUN bit of the SYSPC1 control register. By default, the ECLK1 terminal is in GIO mode. ECLK2 functionality can be enabled by writing 5h to the ECP_KEY field of the ECPCNTL1 register.

NOTE: ECLK2 is multiplexed with EMIF_CLK and ECLK2 is not a primary function after reset. User will need to select ECLK2 to be brought out to the terminal using IOMM module.

2.4.6.4 Dual-Clock Comparators

The microcontrollers include two instances of the dual-clock comparator (DCC) module. This module includes two down counters which independently count from two separate seed values at the rate of two independent clock frequencies. One of the clock inputs is a reference clock input, selectable between the main oscillator or the HF LPO in functional mode. The second clock input is selectable from among a set of defined signals as described in Section 2.4.6.4.1 and Section 2.4.6.4.2. This mechanism can be used to use a known-good clock to measure the frequency of another clock.

2.4.6.4.1 DCC1

As can be seen, the main oscillator (OSCIN) can be used for counter 0 as a “known-good” reference clock. The clock for counter 1 can be selected from among 8 options. Refer to the DCC module chapter for more details on the DCC usage.

Table 2-14. DCC1 Counter 0 Clock Inputs

Clock Source [3–0]	Clock / Signal Name
All other values	oscillator (OSCIN)
5h	HF LPO
Ah	test clock (TCK)

Table 2-15. DCC1 Counter 1 Clock / Signal Inputs

Key [3–0]	Clock Source [3–0]	Clock / Signal Name
Ah	0h	PLL1 free-running clock output
	1h	PLL2 free-running clock output
	2h	LF LPO
	3h	HF LPO
	4h	Flash pump oscillator
	5h	EXTCLKIN1
	6h	EXTCLKIN2
	7	Reserved
	8h-Fh	VCLK
All other values	any value	N2HET1[31]

2.4.6.4.2 DCC2

As can be seen, the main oscillator (OSCIN) can be used for counter 0 as a “known-good” reference clock. The clock for counter 1 can be selected from among 2 options. Refer to the DCC module chapter for more details on the DCC usage.

Table 2-16. DCC2 Counter 0 Clock Inputs

Clock Source [3–0]	Clock / Signal Name
others	oscillator (OSCIN)
0xA	test clock (TCK)

Table 2-17. DCC2 Counter 1 Clock / Signal Inputs

Key [3–0]	Clock Source [3–0]	Clock / Signal Name
Ah	0h	Reserved
	1h	PLL2 post_ODCLK/8
	2h	PLL2 post_ODCLK/16
	3h-7h	Reserved
	8h-Fh	VCLK
All other values	any value	N2HET2[0]

2.5 System and Peripheral Control Registers

The following sections describe the system and peripheral control registers of the TMS570LC43x microcontroller.

2.5.1 Primary System Control Registers (SYS)

This section describes the SYSTEM registers. These registers are divided into two separate frames. The start address of the primary system module frame is FFFF FF00h. The start address of the secondary system module frame is FFFF E100h. The registers support 8-, 16-, and 32-bit writes. The offset is relative to the system module frame start address.

Table 2-18 contains a list of the primary system control registers.

Table 2-18. Primary System Control Registers

Offset	Acronym	Register Description	Section
00h	SYSPC1	SYS Pin Control Register 1	Section 2.5.1.1
04h	SYSPC2	SYS Pin Control Register 2	Section 2.5.1.2
08h	SYSPC3	SYS Pin Control Register 3	Section 2.5.1.3
0Ch	SYSPC4	SYS Pin Control Register 4	Section 2.5.1.4
10h	SYSPC5	SYS Pin Control Register 5	Section 2.5.1.5
14h	SYSPC6	SYS Pin Control Register 6	Section 2.5.1.6
18h	SYSPC7	SYS Pin Control Register 7	Section 2.5.1.7
1Ch	SYSPC8	SYS Pin Control Register 8	Section 2.5.1.8
20h	SYSPC9	SYS Pin Control Register 9	Section 2.5.1.9
30h	CSDIS	Clock Source Disable Register	Section 2.5.1.10
34h	CSDISSET	Clock Source Disable Set Register	Section 2.5.1.11
38h	CSDISCLR	Clock Source Disable Clear Register	Section 2.5.1.12
3Ch	CDDIS	Clock Domain Disable Register	Section 2.5.1.13
40h	CDDISSET	Clock Domain Disable Set Register	Section 2.5.1.14
44h	CDDISCLR	Clock Domain Disable Clear Register	Section 2.5.1.15
48h	GHVSRC	GCLK1, HCLK, VCLK, and VCLK2 Source Register	Section 2.5.1.16
4Ch	VCLKASRC	Peripheral Asynchronous Clock Source Register	Section 2.5.1.17
50h	RCLKSRC	RTI Clock Source Register	Section 2.5.1.18
54h	CSVSTAT	Clock Source Valid Status Register	Section 2.5.1.19
58h	MSTGCR	Memory Self-Test Global Control Register	Section 2.5.1.20
5Ch	MINITGCR	Memory Hardware Initialization Global Control Register	Section 2.5.1.21
60h	MSINENA	Memory Self-Test/Initialization Enable Register	Section 2.5.1.22
68h	MSTCGSTAT	MSTC Global Status Register	Section 2.5.1.23
6Ch	MINISTAT	Memory Hardware Initialization Status Register	Section 2.5.1.24
70h	PLLCTL1	PLL Control Register 1	Section 2.5.1.25
74h	PLLCTL2	PLL Control Register 2	Section 2.5.1.26
78h	SYSPC10	SYS Pin Control Register 10	Section 2.5.1.27
7Ch	DIEIDL	Die Identification Register, Lower Word	Section 2.5.1.28
80h	DIEIDH	Die Identification Register, Upper Word	Section 2.5.1.29
88h	LPOMONCTL	LPO/CLock Monitor Control Register	Section 2.5.1.31
8Ch	CLKTEST	Clock Test Register	Section 2.5.1.31
90h	DFTCTRLREG	DFT Control Register	Section 2.5.1.32
94h	DFTCTRLREG2	DFT Control Register 2	Section 2.5.1.33
A0h	GPREG1	General Purpose Register	Section 2.5.1.34
B0h	SSIR1	System Software Interrupt Request 1 Register	Section 2.5.1.35
B4h	SSIR2	System Software Interrupt Request 2 Register	Section 2.5.1.36
B8h	SSIR3	System Software Interrupt Request 3 Register	Section 2.5.1.37

Table 2-18. Primary System Control Registers (continued)

Offset	Acronym	Register Description	Section
BCh	SSIR4	System Software Interrupt Request 4 Register	Section 2.5.1.38
C0h	RAMGCR	RAM Control Register	Section 2.5.1.39
C4h	BMMCR1	Bus Matrix Module Control Register 1	Section 2.5.1.40
CCh	CPURSTCR	CPU Reset Control Register	Section 2.5.1.41
D0h	CLKCNTL	Clock Control Register	Section 2.5.1.42
D4h	ECPCNTL	ECP Control Register	Section 2.5.1.43
DCh	DEVCR1	DEV Parity Control Register 1	Section 2.5.1.44
E0h	SYSECR	System Exception Control Register	Section 2.5.1.45
E4h	SYSESR	System Exception Status Register	Section 2.5.1.46
E8h	SYSTASR	System Test Abort Status Register	Section 2.5.1.47
ECh	GLBSTAT	Global Status Register	Section 2.5.1.48
F0h	DEVID	Device Identification Register	Section 2.5.1.49
F4h	SSIVEC	Software Interrupt Vector Register	Section 2.5.1.50
F8h	SSIF	System Software Interrupt Flag Register	Section 2.5.1.51

2.5.1.1 SYS Pin Control Register 1 (SYSPC1)

The SYSPC1 register, shown in Figure 2-8 and described in Table 2-19, controls the function of the ECLK pin.

Figure 2-8. SYS Pin Control Register 1 (SYSPC1) (offset = 00h)

31	16
Reserved	
R-0	

15	1	0
Reserved		ECPCLKFUN
R-0		R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 2-19. SYS Pin Control Register 1 (SYSPC1) Field Descriptions

Bit	Field	Value	Description
31-1	Reserved	0	Reads return 0. Writes have no effect.
0	ECPCLKFUN		ECLK function. This bit changes the function of the ECLK pin.
		0	ECLK is in GIO mode.
		1	ECLK is in functional mode as a clock output.
			Note: Proper ECLK duty cycle is not guaranteed until 1 ECLK cycle has elapsed after switching into functional mode.

2.5.1.2 SYS Pin Control Register 2 (SYSPC2)

The SYSPC2 register, shown in Figure 2-9 and described in Table 2-20, controls whether the pin is an input or an output when in GIO mode.

Figure 2-9. SYS Pin Control Register 2 (SYSPC2) (offset = 04h)

31	16
Reserved	
R-0	

15	1	0
Reserved		ECPCLKDIR
R-0		R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 2-20. SYS Pin Control Register 2 (SYSPC2) Field Descriptions

Bit	Field	Value	Description
31-1	Reserved	0	Reads return 0. Writes have no effect.
0	ECPCLKDIR		ECLK data direction. This bit controls the direction of the ECLK pin when it is configured to be in GIO mode only.
		0	The ECLK pin is an input.
			Note: If the pin direction is set as an input, the output buffer is tristated.
		1	The ECLK pin is an output.
			Note: The ECLK pin is placed into GIO mode by clearing the ECPCLKFUN bit to 0 in the SYSPC1 register.

2.5.1.3 SYS Pin Control Register 3 (SYSPC3)

The SYSPC3 register, shown in Figure 2-10 and described in Table 2-21, displays the logic state of the ECLK pin when it is in GIO mode.

Figure 2-10. SYS Pin Control Register 3 (SYSPC3) (offset = 08h)

31	16
Reserved	
R-0	

15	1	0
Reserved		ECPCLKDIN
R-0		R-U

LEGEND: R = Read only; U = value is undefined; -*n* = value after reset

Table 2-21. SYS Pin Control Register 3 (SYSPC3) Field Descriptions

Bit	Field	Value	Description
31-1	Reserved	0	Reads return 0. Writes have no effect.
0	ECPCLKDIN		ECLK data in. This bit displays the logic state of the ECLK pin when it is configured to be in GIO mode.
		0	The ECLK pin is at logic low (0).
		1	The ECLK pin is at logic high (1).

2.5.1.4 SYS Pin Control Register 4 (SYSPC4)

The SYSPC4 register, shown in Figure 2-11 and described in Table 2-22, controls the logic level output function of the ECLK pin when it is configured as an output in GIO mode.

Figure 2-11. SYS Pin Control Register 4 (SYSPC4) (offset = 0Ch)

31	16
Reserved	
R-0	

15	1	0
Reserved		ECPCLKDOUT
R-0		R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 2-22. SYS Pin Control Register 4 (SYSPC4) Field Descriptions

Bit	Field	Value	Description
31-1	Reserved	0	Reads return 0. Writes have no effect.
0	ECPCLKDOUT		ECLK data out write. This bit is only active when the ECLK pin is configured to be in GIO mode. Writes to this bit will only take effect when the ECLK pin is configured as an output in GIO mode. The current logic state of the ECLK pin will be displayed by this bit in both input and output GIO mode.
		0	The ECLK pin is driven to logic low (0).
		1	The ECLK pin is driven to logic high (1).
			Note: The ECLK pin is placed into GIO mode by clearing the ECPCLKFUN bit to 0 in the SYSPC1 register. The ECLK pin is placed in output mode by setting the ECPCLKDIR bit to 1 in the SYSPC2 register.

2.5.1.5 SYS Pin Control Register 5 (SYSPC5)

The SYSPC5 register, shown in Figure 2-12 and described in Table 2-23, controls the set function of the ECLK pin when it is configured as an output in GIO mode.

Figure 2-12. SYS Pin Control Register 5 (SYSPC5) (offset = 10h)

31	16
Reserved	
R-0	

15	1	0
Reserved		ECPCLKSET
R-0		R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 2-23. SYS Pin Control Register 5 (SYSPC5) Field Descriptions

Bit	Field	Value	Description
31-1	Reserved	0	Reads return 0. Writes have no effect.
0	ECPCLKSET		ECLK data out set. This bit drives the output of the ECLK pin high when set in GIO output mode.
		0	*Write:* Writing a 0 has no effect.
		1	*Write:* The ECLK pin is driven to logic high (1).
			Note: The current logic state of the ECPCLKDOUT bit will also be displayed by this bit when the pin is configured in GIO output mode.
			Note: The ECLK pin is placed into GIO mode by clearing the ECPCLKFUN bit to 0 in the SYSPC1 register. The ECLK pin is placed in output mode by setting the ECPCLKDIR bit to 1 in the SYSPC2 register.

2.5.1.6 SYS Pin Control Register 6 (SYSPC6)

The SYSPC6 register, shown in Figure 2-13 and described in Table 2-24, controls the clear function of the ECLK pin when it is configured as an output in GIO mode..

Figure 2-13. SYS Pin Control Register 6 (SYSPC6) (offset = 14h)

31	16
Reserved	
R-0	

15	1	0
Reserved		ECPCLKCLR
R-0		R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 2-24. SYS Pin Control Register 6 (SYSPC6) Field Descriptions

Bit	Field	Value	Description
31-1	Reserved	0	Reads return 0. Writes have no effect.
0	ECPCLKCLR		ECLK data out clear. This bit drives the output of the ECLK pin low when set in GIO output mode.
		0	*Write:* The ECLK pin value is unchanged.
		1	*Write:* The ECLK pin is driven to logic low (0).
			Note: The current logic state of the ECPCLKDOUT bit will also be displayed by this bit when the pin is configured in GIO output mode.
			Note: The ECLK pin is placed into GIO mode by clearing the ECPCLKFUN bit to 0 in the SYSPC1 register. The ECLK pin is placed in output mode by setting the ECPCLKDIR bit to 1 in the SYSPC2 register.

2.5.1.7 SYS Pin Control Register 7 (SYSPC7)

The SYSPC7 register, shown in Figure 2-14 and described in Table 2-25, controls the open drain function of the ECLK pin.

Figure 2-14. SYS Pin Control Register 7 (SYSPC7) (offset = 18h)

31	16
Reserved	
R-0	

15	1	0
Reserved		ECPCLKODE
R-0		R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 2-25. SYS Pin Control Register 7 (SYSPC7) Field Descriptions

Bit	Field	Value	Description
31-1	Reserved	0	Reads return 0. Writes have no effect.
0	ECPCLKODE		ECLK open drain enable. This bit is only active when ECLK is configured to be in GIO mode.
		0	The ECLK pin is configured in push/pull (normal GIO) mode.
		1	The ECLK pin is configured in open drain mode. The ECPCLKDOUT bit in the SYSPC4 register controls the state of the ECLK output buffer: ECPCLKDOUT = 0: The ECLK output buffer is driven low. ECPCLKDOUT = 1: The ECLK output buffer is tristated. **Note: The ECLK pin is placed into GIO mode by clearing the ECPCLKFUN bit to 0 in the SYSPC1 register.**

2.5.1.8 SYS Pin Control Register 8 (SYSPC8)

The SYSPC8 register, shown in Figure 2-15 and described in Table 2-26, controls the pull enable function of the ECLK pin when it is configured as an input in GIO mode.

Figure 2-15. SYS Pin Control Register 8 (SYSPC8) (offset = 1Ch)

31	16
Reserved	
R-0	

15	1	0
Reserved		ECPCLKPUE
R-0		R/W-0

LEGEND: R/W = Read/Write; R = Read only; D = Device Specific; -*n* = value after reset

Table 2-26. SYS Pin Control Register 8 (SYSPC8) Field Descriptions

Bit	Field	Value	Description
31-1	Reserved	0	Reads return 0. Writes have no effect.
0	ECPCLKPUE		ECLK pull enable. Writes to this bit will only take effect when the ECLK pin is configured as an input in GIO mode.
		0	ECLK pull enable is active.
		1	ECLK pull enable is inactive. **Note: The pull direction (up/down) is selected by the ECPCLKPS bit in the SYSPC9 register.** **Note: The ECLK pin is placed into GIO mode by clearing the ECPCLKFUN bit to 0 in the SYSPC1 register. The ECLK pin is placed in input mode by clearing the ECPCLKDIR bit to 0 in the SYSPC2 register.**

2.5.1.9 SYS Pin Control Register 9 (SYSPC9)

The SYSPC9 register, shown in Figure 2-16 and described in Table 2-27, controls the pull up/pull down configuration of the ECLK pin when it is configured as an input in GIO mode.

Figure 2-16. SYS Pin Control Register 9 (SYSPC9) (offset = 20h)

31	16
Reserved	
R-0	

15	1	0
Reserved		ECPCLKPS
R-0		R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 2-27. SYS Pin Control Register 9 (SYSPC9) Field Descriptions

Bit	Field	Value	Description
31-1	Reserved	0	Reads return 0. Writes have no effect.
0	ECPCLKPS		ECLK pull up/pull down select. This bit is only active when ECLK is configured as an input in GIO mode and the pull up/pull down logic is enabled.
		0	ECLK pull down is selected, when pull up/pull down logic is enabled.
		1	ECLK pull up is selected, when pull up/pull down logic is enabled.
			Note: The ECLK pin pull up/pull down logic is enabled by clearing the ECPCLKPUE bit to 0 in the SYSPC8 register.
			Note: The ECLK pin is placed into GIO mode by clearing the ECPCLKFUN bit to 0 in the SYSPC1 register. The ECLK pin is placed in input mode by clearing the ECPCLKDIR bit to 0 in the SYSPC2 register.

2.5.1.10 Clock Source Disable Register (CSDIS)

The CSDIS register, shown in Figure 2-17 and described in Table 2-28, controls and displays the state of the device clock sources.

Figure 2-17. Clock Source Disable Register (CSDIS) (offset = 30h)

31-8
Reserved
R-0

7	6	5	4	3	2	1	0
CLKSR7OFF	CLKSR6OFF	CLKSR5OFF	CLKSR4OFF	CLKSR3OFF	Reserved	CLKSR1OFF	CLKSR0OFF
R/WP-1	R/WP-1	R/WP-0	R/WP-0	R/WP-1	R-1	R/WP-1	R/WP-0

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privileged mode only; -*n* = value after reset

Table 2-28. Clock Source Disable Register (CSDIS) Field Descriptions

Bit	Field	Value	Description
31-8	Reserved	0	Reads return 0. Writes have no effect.
7-3	CLKSR[7-3]OFF		Clock source[7-3] off.
		0	Clock source[7-3] is enabled.
		1	Clock source[7-3] is disabled.
			Note: On wakeup, only clock sources 0, 4, and 5 are enabled.
2	Reserved	1	Reads return 1. Writes have no effect.
1-0	CLKSR[1-0]OFF		Clock source[1-0] off.
		0	Clock source[1-0] is enabled.
		1	Clock source[1-0] is disabled.
			Note: On wakeup, only clock sources 0, 4, and 5 are enabled.

Table 2-29. Clock Sources Table

Clock Source #	Clock Source Name
Clock Source 0	Oscillator
Clock Source1	PLL1
Clock Source 2	Not Implemented
Clock Source 3	EXTCLKIN
Clock Source 4	Low Frequency LPO (Low Power Oscillator) clock
Clock Source 5	High frequency LPO (Low Power Oscillator) clock
Clock Source 6	PLL2
Clock Source 7	EXTCLKIN2

NOTE: Non-implemented clock sources should not be enabled or used.

2.5.1.11 Clock Source Disable Set Register (CSDISSET)

The CSDISSET register, shown in Figure 2-18 and described in Table 2-30, sets clock sources to the disabled state.

Figure 2-18. Clock Source Disable Set Register (CSDISSET) (offset = 34h)

31	8
Reserved	
R-0	

7	6	5	4	3	2	1	0
SETCLKSR7 OFF	SETCLKSR6 OFF	SETCLKSR5 OFF	SETCLKSR4 OFF	SETCLKSR3 OFF	Reserved	SETCLKSR1 OFF	SETCLKSR0 OFF
R/WP-1	R/WP-1	R/WP-0	R/WP-0	R/WP-1	R-1	R/WP-1	R/WP-0

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privileged mode only; -*n* = value after reset

Table 2-30. Clock Source Disable Set Register (CSDISSET) Field Descriptions

Bit	Field	Value	Description
31-8	Reserved	0	Reads return 0. Writes have no effect.
7-3	SETCLKSR[7-3]OFF		Set clock source[7-3] to the disabled state.
		0	*Read:* Clock source[7-3] is enabled. *Write:* Clock source[7-3] is unchanged.
		1	*Read:* Clock source[7-3] is disabled. *Write:* Clock source[7-3] is set to the disabled state. **Note: After a new clock source disable bit is set via the CSDISSET register, the new status of the bit will be reflected in the CSDIS register (offset 30h), the CSDISSET register (offset 34h), and the CSDISCLR register (offset 38h).**
2	Reserved	1	Reads return 1. Writes have no effect.
1-0	SETCLKSR[1-0]OFF		Set clock source[1-0] to the disabled state.
		0	*Read:* Clock source[1-0] is enabled. *Write:* Clock source[1-0] is unchanged.
		1	*Read:* Clock source[1-0] is disabled. *Write:* Clock source[1-0] is set to the disabled state. **Note: After a new clock source disable bit is set via the CSDISSET register, the new status of the bit will be reflected in the CSDIS register (offset 30h), the CSDISSET register (offset 34h), and the CSDISCLR register (offset 38h).**

NOTE: A list of the available clock sources is shown in the Table 2-29.

2.5.1.12 Clock Source Disable Clear Register (CSDISCLR)

The CSDISCLR register, shown in Figure 2-19 and described in Table 2-31, clears clock sources to the enabled state.

Figure 2-19. Clock Source Disable Clear Register (CSDISCLR) (offset = 38h)

31 — 8
Reserved
R-0

7	6	5	4	3	2	1	0
CLRCLKSR7 OFF	CLRCLKSR6 OFF	CLRCLKSR5 OFF	CLRCLKSR4 OFF	CLRCLKSR3 OFF	Reserved	CLRCLKSR1 OFF	CLRCLKSR0 OFF
R/WP-1	R/WP-1	R/WP-0	R/WP-0	R/WP-1	R-1	R/WP-1	R/WP-0

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privileged mode only; -*n* = value after reset

Table 2-31. Clock Source Disable Clear Register (CSDISCLR) Field Descriptions

Bit	Field	Value	Description
31-8	Reserved	0	Reads return 0. Writes have no effect.
7-3	CLRCLKSR[7-3]OFF		Enables clock source[7-3].
		0	*Read:* Clock source[7-3] is enabled. *Write:* Clock source[7-3] is unchanged.
		1	*Read:* Clock source[7-3] is enabled. *Write:* Clock source[7-3] is set to the enabled state. **Note: After a new clock source disable bit is set via the CSDISSET register, the new status of the bit will be reflected in the CSDIS register (offset 30h), the CSDISSET register (offset 34h), and the CSDISCLR register (offset 38h).**
2	Reserved	1	Reads return 1. Writes have no effect.
1-0	CLRCLKSR[1-0]OFF		Enables clock source[1-0].
		0	*Read:* Clock source[1-0] is enabled. *Write:* Clock source[1-0] is unchanged.
		1	*Read:* Clock source[1-0] is enabled. *Write:* Clock source[1-0] is set to the enabled state. **Note: After a new clock source disable bit is set via the CSDISSET register, the new status of the bit will be reflected in the CSDIS register (offset 30h), the CSDISSET register (offset 34h) and the CSDISCLR register (offset 38h).**

NOTE: A list of the available clock sources is shown in the Table 2-29.

2.5.1.13 Clock Domain Disable Register (CDDIS)

The CDDIS register, shown in Figure 2-20 and described in Table 2-32, controls the state of the clock domains.

NOTE: All the clock domains are enabled on wakeup.

The application should assure that when HCLK and VCLK_sys are turned off through the HCLKOFF bit, the GCLK1 domain is also turned off.

The register bits in CDDIS are designated as high-integrity bits and have been implemented with error-correcting logic such that each bit, although read and written as a single bit, is actually a multi-bit key with error correction capability. As such, single-bit flips within the “key” can be corrected allowing protection of the system as a whole. An error detected is signaled to the ESM module.

Figure 2-20. Clock Domain Disable Register (CDDIS) (offset = 3Ch)

31-16
Reserved
R-0

15-12	11	10	9	8
Reserved	VCLKA4OFF	Reserved	Reserved	VCLK3OFF
R-0	R/WP-0	R/WP-0	R/WP-0	R/WP-0

7	6	5	4	3	2	1	0
Reserved	RTICLK1OFF	VCLKA2OFF	VCLKA1OFF	VCLK2OFF	VCLKPOFF	HCLKOFF	GCLK1OFF
R/WP-0	R/WP-0	R/WP-0	R/WP-0	R/WP-0	R/WP-0	R/WP-0	R/WP-0

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privileged mode only; -*n* = value after reset

Table 2-32. Clock Domain Disable Register (CDDIS) Field Descriptions

Bit	Field	Value	Description
31-12	Reserved	0-1	Reads return 0 or 1 and privilege mode writes allowed.
11	VCLKA4OFF		VCLKA4 domain off.
		0	The VCLKA4 domain is enabled.
		1	The VCLKA4 domain is disabled.
10-9	Reserved	0-1	Reads return 0 or 1 and privilege mode writes allowed.
8	VCLK3OFF		VCLK3 domain off.
		0	The VCLK3 domain is enabled.
		1	The VCLK3 domain is disabled.
7	Reserved	0-1	Reads return 0 or 1 and privilege mode writes allowed.
6	RTICLK1OFF		RTICLK1 domain off.
		0	The RTICLK1 domain is enabled.
		1	The RTICLK1 domain is disabled.
5-4	VCLKA[2-1]OFF		VCLKA[2-1] domain off.
		0	The VCLKA[2-1] domain is enabled.
		1	The VCLKA[2-1] domain is disabled.
3	VCLK2OFF		VCLK2 domain off.
		0	The VCLK2 domain is enabled.
		1	The VCLK2 domain is disabled.
2	VCLKPOFF		VCLK_periph domain off.
		0	The VCLK_periph domain is enabled.
		1	The VCLK_periph domain is disabled.

Table 2-32. Clock Domain Disable Register (CDDIS) Field Descriptions (continued)

Bit	Field	Value	Description
1	HCLKOFF		HCLK and VCLK_sys domains off.
		0	The HCLK and VCLK_sys domains are enabled.
		1	The HCLK and VCLK_sys domains are disabled.
0	GCLK1OFF		GCLK1 domain off.
		0	The GCLK1 domain is enabled.
		1	The GCLK1 domain is disabled.

2.5.1.14 Clock Domain Disable Set Register (CDDISSET)

This CDDISSET register, shown in Figure 2-21 and described in Table 2-33, sets clock domains to the disabled state.

Figure 2-21. Clock Domain Disable Set Register (CDDISSET) (offset = 40h)

31	16
Reserved	
R-0	

15	12	11	10	9	8
Reserved		SETVCLKA4 OFF	Reserved	Reserved	SETVCLK3 OFF
R-0		R/WP-0	R/WP-0	R/WP-0	R/WP-0

7	6	5	4	3	2	1	0
Reserved	SETRTICLK1 OFF	SETVCLKA2 OFF	SETVCLKA1 OFF	SETVCLK2 OFF	SETVCLKP OFF	SETHCLK OFF	SETGCLK1 OFF
R/WP-0	R/WP-0	R/WP-0	R/WP-0	R/WP-0	R/WP-0	R/WP-0	R/WP-0

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privileged mode only; -*n* = value after reset

Table 2-33. Clock Domain Disable Set Register (CDDISSET) Field Descriptions

Bit	Field	Value	Description
31-12	Reserved	0	Reads return 0. Writes have no effect.
11	SETVCLKA4OFF		Set VCLKA4 domain.
		0	*Read:* The VCLKA4 domain is enabled. *Write:* The VCLKA4 domain is unchanged.
		1	*Read:* The VCLKA4 domain is disabled. *Write:* The VCLKA4 domain is set to the enabled state.
10-9	Reserved	0	Reads return zero or one and privilege mode writes allowed.
8	SETVCLK3OFF		Set VCLK3 domain.
		0	*Read:* The VCLK3 domain is enabled. *Write:* The VCLK3 domain is unchanged.
		1	*Read:* The VCLK3 domain is disabled. *Write:* The VCLK3 domain is set to the enabled state.
7	Reserved	0-1	Reads return 0 or 1 and privilege mode writes allowed.
6	SETRTICLK1OFF		Set RTICLK1 domain.
		0	*Read:* The RTICLK1 domain is enabled. *Write:* The RTICLK1 domain is unchanged.
		1	*Read:* The RTICLK1 domain is disabled. *Write:* The RTICLK1 domain is set to the enabled state.
5-4	SETVCLKA[2-1]OFF		Set VCLKA[2-1] domain.
		0	*Read:* The VCLKA[2-1] domain is enabled. *Write:* The VCLKA[2-1] domain is unchanged.
		1	*Read:* The VCLKA[2-1] domain is disabled. *Write:* The VCLKA[2-1] domain is set to the enabled state.
3	SETVCLK2OFF		Set VCLK2 domain.
		0	*Read:* The VCLK2 domain is enabled. *Write:* The VCLK2 domain is unchanged.
		1	*Read:* The VCLK2 domain is disabled. *Write:* The VCLK2 domain is set to the enabled state.

Table 2-33. Clock Domain Disable Set Register (CDDISSET) Field Descriptions (continued)

Bit	Field	Value	Description
2	SETVCLKPOFF		Set VCLK_periph domain.
		0	*Read:* The VCLK_periph domain is enabled. *Write:* The VCLK_periph domain is unchanged.
		1	*Read:* The VCLK_periph domain is disabled. *Write:* The VCLK_periph domain is set to the enabled state.
1	SETHCLKOFF		Set HCLK and VCLK_sys domains.
		0	*Read:* The HCLK and VCLK_sys domain is enabled. *Write:* The HCLK and VCLK_sys domain is unchanged.
		1	*Read:* The HCLK and VCLK_sys domain is disabled. *Write:* The HCLK and VCLK_sys domain is set to the enabled state.
0	SETGCLK1OFF		Set GCLK1 domain.
		0	*Read:* The GCLK1 domain is enabled. *Write:* The GCLK1 domain is unchanged.
		1	*Read:* The GCLK1 domain is disabled. *Write:* The GCLK1 domain is set to the enabled state.

2.5.1.15 Clock Domain Disable Clear Register (CDDISCLR)

The CDDISCLR register, shown in Figure 2-22 and described in Table 2-34, clears clock domains to the enabled state.

Figure 2-22. Clock Domain Disable Clear Register (CDDISCLR) (offset = 44h)

31-16
Reserved
R-0

15-12	11	10	9	8
Reserved	CLRVCLKA4 OFF	Reserved	Reserved	CLRVCLK3 OFF
R-0	R/WP-0	R/WP-0	R/WP-0	R/WP-0

7	6	5	4	3	2	1	0
Reserved	CLRRTICLK1 OFF	CLRVCLKA2 OFF	CLRVCLKA1 OFF	CLRVCLK2 OFF	CLRVCLKP OFF	CLRHCLK OFF	CLRGCLK1 OFF
R/WP-0	R/WP-0	R/WP-0	R/WP-0	R/WP-0	R/WP-0	R/WP-0	R/WP-0

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privileged mode only; -*n* = value after reset

Table 2-34. Clock Domain Disable Clear Register (CDDISCLR) Field Descriptions

Bit	Field	Value	Description
31-12	Reserved	0	Reads return 0. Writes have no effect.
11	CLRVCLKA4OFF		Clear VCLKA4 domain.
		0	*Read:* The VCLKA4 domain is enabled. *Write:* The VCLKA4 domain is unchanged.
		1	*Read:* The VCLKA4 domain is disabled. *Write:* The VCLKA4 domain is cleared to the enabled state.
10-9	Reserved	0	Reads return zero or one and privilege mode writes allowed.
8	CLRVCLK3OFF		Clear VCLK3 domain.
		0	*Read:* The VCLK3 domain is enabled. *Write:* The VCLK3 domain is unchanged.
		1	*Read:* The VCLK3 domain is disabled. *Write:* The VCLK3 domain is cleared to the enabled state.
7	Reserved	0-1	Reads return 0 or 1 and privilege mode writes allowed.
6	CLRRTICLK1OFF		Clear RTICLK1 domain.
		0	*Read:* The RTICLK1 domain is enabled. *Write:* The RTICLK1 domain is unchanged.
		1	*Read:* The RTICLK1 domain is disabled. *Write:* The RTICLK1 domain is cleared to the enabled state.
5-4	CLRVCLKA[2-1]OFF		Clear VCLKA[2-1] domain.
		0	*Read:* The VCLKA[2-1] domain is enabled. *Write:* The VCLKA[2-1] domain is unchanged.
		1	*Read:* The VCLKA[2-1] domain is disabled. *Write:* The VCLKA[2-1] domain is cleared to the enabled state.
3	CLRVCLK2OFF		Clear VCLK2 domain.
		0	*Read:* The VCLK2 domain is enabled. *Write:* The VCLK2 domain is unchanged.
		1	*Read:* The VCLK2 domain is disabled. *Write:* The VCLK2 domain is cleared to the enabled state.

Table 2-34. Clock Domain Disable Clear Register (CDDISCLR) Field Descriptions (continued)

Bit	Field	Value	Description
2	CLRVCLKPOFF		Clear VCLK_periph domain.
		0	*Read:* The VCLK_periph domain is enabled. *Write:* The VCLK_periph domain is unchanged.
		1	*Read:* The VCLK_periph domain is disabled. *Write:* The VCLK_periph domain is cleared to the enabled state.
1	CLRHCLKOFF		Clear HCLK and VCLK_sys domains.
		0	*Read:* The HCLK and VCLK_sys domain is enabled. *Write:* The HCLK and VCLK_sys domain is unchanged.
		1	*Read:* The HCLK and VCLK_sys domain is disabled. *Write:* The HCLK and VCLK_sys domain is cleared to the enabled state.
0	CLRGCLK1OFF		Clear GCLK1 domain.
		0	*Read:* The GCLK1 domain is enabled. *Write:* The GCLK1 domain is unchanged.
		1	*Read:* The GCLK1 domain is disabled. *Write:* The GCLK1 domain is cleared to the enabled state.

2.5.1.16 GCLK1, HCLK, VCLK, and VCLK2 Source Register (GHVSRC)

The GHVSRC register, shown in Figure 2-23 and described in Table 2-35, controls the clock source configuration for the GCLK1, HCLK, VCLK and VCLK2 clock domains.

Figure 2-23. GCLK1, HCLK, VCLK, and VCLK2 Source Register (GHVSRC) (offset = 48h)

31 28	27 24	23 20	19 16
Reserved	GHVWAKE	Reserved	HVLPM
R-0	R/WP-0	R-0	R/WP-0

15 4	3 0
Reserved	GHVSRC
R-0	R/WP-0

LEGEND: R = Read only; R/W = Read/Write; WP = Write in privileged mode only; -*n* = value after reset

Table 2-35. GCLK1, HCLK, VCLK, and VCLK2 Source Register (GHVSRC) Field Descriptions

Bit	Field	Value	Description
31-28	Reserved	0	Reads return 0. Writes have no effect.
27-24	GHVWAKE		GCLK1, HCLK, VCLK source on wakeup.
		0	Clock source0 is the source for GCLK1, HCLK, VCLK on wakeup.
		1h	Clock source1 is the source for GCLK1, HCLK, VCLK on wakeup.
		2h	Clock source2 is the source for GCLK1, HCLK, VCLK on wakeup.
		3h	Clock source3 is the source for GCLK1, HCLK, VCLK on wakeup.
		4h	Clock source4 is the source for GCLK1, HCLK, VCLK on wakeup.
		5h	Clock source5 is the source for GCLK1, HCLK, VCLK on wakeup.
		6h	Clock source6 is the source for GCLK1, HCLK, VCLK on wakeup.
		7h	Clock source7 is the source for GCLK1, HCLK, VCLK on wakeup.
		8h-Fh	Reserved
23-20	Reserved	0	Reads return 0. Writes have no effect.
19-16	HVLPM		HCLK, VCLK, VCLK2 source on wakeup when GCLK1 is turned off.
		0	Clock source0 is the source for HCLK, VCLK, VCLK2 on wakeup.
		1h	Clock source1 is the source for HCLK, VCLK, VCLK2 on wakeup.
		2h	Clock source2 is the source for HCLK, VCLK, VCLK2 on wakeup.
		3h	Clock source3 is the source for HCLK, VCLK, VCLK2 on wakeup.
		4h	Clock source4 is the source for HCLK, VCLK, VCLK2 on wakeup.
		5h	Clock source5 is the source for HCLK, VCLK, VCLK2 on wakeup.
		6h	Clock source6 is the source for HCLK, VCLK, VCLK2 on wakeup.
		7h	Clock source7 is the source for HCLK, VCLK, VCLK2 on wakeup.
		8h-Fh	Reserved
15-4	Reserved	0	Reads return 0. Writes have no effect.
3-0	GHVSRC		GCLK1, HCLK, VCLK, VCLK2 current source. **Note: The GHVSRC[3-0] bits are updated with the HVLPM[3-0] setting when GCLK1 is turned off, and are updated with the GHVWAKE[3-0] setting on system wakeup.**
		0	Clock source0 is the source for GCLK1, HCLK, VCLK, VCLK2.
		1h	Clock source1 is the source for GCLK1, HCLK, VCLK, VCLK2.
		2h	Clock source2 is the source for GCLK1, HCLK, VCLK, VCLK2.
		3h	Clock source3 is the source for GCLK1, HCLK, VCLK, VCLK2.
		4h	Clock source4 is the source for GCLK1, HCLK, VCLK, VCLK2.
		5h	Clock source5 is the source for GCLK1, HCLK, VCLK, VCLK2.
		6h	Clock source6 is the source for GCLK1, HCLK, VCLK, VCLK2.
		7h	Clock source7 is the source for GCLK1, HCLK, VCLK, VCLK2.
		8h-Fh	Reserved

NOTE: Non-implemented clock sources should not be enabled or used. A list of the available clock sources is shown in the Table 2-29.

2.5.1.17 Peripheral Asynchronous Clock Source Register (VCLKASRC)

The VCLKASRC register, shown in Figure 2-24 and described in Table 2-36, sets the clock source for the asynchronous peripheral clock domains to be configured to run from a specific clock source.

Figure 2-24. Peripheral Asynchronous Clock Source Register (VCLKASRC) (offset = 4Ch)

31 28	27 24	23 20	19 16
Reserved	Reserved	Reserved	Reserved
R-0	R/WP-1h	R-0	R/WP-1h

15 12	11 8	7 4	3 0
Reserved	VCLKA2S	Reserved	VCLKA1S
R-0	R/WP-9h	R-0	R/WP-9h

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privileged mode only; -*n* = value after reset

Table 2-36. Peripheral Asynchronous Clock Source Register (VCLKASRC) Field Descriptions

Bit	Field	Value	Description
31-28	Reserved	0	Reads return 0. Writes have no effect.
27-24	Reserved	0-1	Reads return 0 or 1 and privilege mode writes allowed.
23-20	Reserved	0	Reads return 0. Writes have no effect.
19-16	Reserved	0-1	Reads return 0 or 1 and privilege mode writes allowed.
15-12	Reserved	0	Reads return 0. Writes have no effect.
11-8	VCLKA2S		Peripheral asynchronous clock2 source.
		0	Clock source0 is the source for peripheral asynchronous clock2.
		1h	Clock source1 is the source for peripheral asynchronous clock2.
		2h	Clock source2 is the source for peripheral asynchronous clock2.
		3h	Clock source3 is the source for peripheral asynchronous clock2.
		4h	Clock source4 is the source for peripheral asynchronous clock2.
		5h	Clock source5 is the source for peripheral asynchronous clock2.
		6h	Clock source6 is the source for peripheral asynchronous clock2.
		7h	Clock source7 is the source for peripheral asynchronous clock2.
		8h-Fh	VCLK is the source for peripheral asynchronous clock2.
7-4	Reserved	0	Reads return 0. Writes have no effect.
3-0	VCLKA1S		Peripheral asynchronous clock1 source.
		0	Clock source0 is the source for peripheral asynchronous clock1.
		1h	Clock source1 is the source for peripheral asynchronous clock1.
		2h	Clock source2 is the source for peripheral asynchronous clock1.
		3h	Clock source3 is the source for peripheral asynchronous clock1.
		4h	Clock source4 is the source for peripheral asynchronous clock1.
		5h	Clock source5 is the source for peripheral asynchronous clock1.
		6h	Clock source6 is the source for peripheral asynchronous clock1.
		7h	Clock source7 is the source for peripheral asynchronous clock1.
		8h-Fh	VCLK is the source for peripheral asynchronous clock1.

NOTE: Non-implemented clock sources should not be enabled or used. A list of the available clock sources is shown in Table 2-29.

2.5.1.18 RTI Clock Source Register (RCLKSRC)

The RCLKSRC register, shown in Figure 2-25 and described in Table 2-37, controls the RTI (Real Time Interrupt) clock source selection.

NOTE: Important constraint when the RTI clock source is not VCLK

If the RTIx clock source is chosen to be anything other than the default VCLK, then the RTI clock needs to be at least three times slower than the VCLK. This can be achieved by configuring the RTIxCLK divider in this register. This divider is internally bypassed when the RTIx clock source is VCLK.

Figure 2-25. RTI Clock Source Register (RCLKSRC) (offset = 50h)

<table>
<tr><td>31</td><td>16</td></tr>
<tr><td colspan="2">Reserved</td></tr>
<tr><td colspan="2">R-0</td></tr>
</table>

<table>
<tr><td>15</td><td>10</td><td>9</td><td>8</td><td>7</td><td>4</td><td>3</td><td>0</td></tr>
<tr><td colspan="2">Reserved</td><td colspan="2">RTI1DIV</td><td colspan="2">Reserved</td><td colspan="2">RTI1SRC</td></tr>
<tr><td colspan="2">R-0</td><td colspan="2">R/WP-1h</td><td colspan="2">R-0</td><td colspan="2">R/WP-9h</td></tr>
</table>

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privileged mode only; -*n* = value after reset

Table 2-37. RTI Clock Source Register (RCLKSRC) Field Descriptions

Bit	Field	Value	Description
31-10	Reserved	0	Reads return 0. Writes have no effect.
9-8	RTI1DIV		RTI clock1 Divider.
		0	RTICLK1 divider value is 1.
		1h	RTICLK1 divider value is 2.
		2h	RTICLK1 divider value is 4.
		3h	RTICLK1 divider value is 8.
7-4	Reserved	0	Reads return 0. Writes have no effect.
3-0	RTI1SRC		RTI clock1 source.
		0	Clock source0 is the source for RTICLK1.
		1h	Clock source1 is the source for RTICLK1.
		2h	Clock source2 is the source for RTICLK1.
		3h	Clock source3 is the source for RTICLK1.
		4h	Clock source4 is the source for RTICLK1.
		5h	Clock source5 is the source for RTICLK1.
		6h	Clock source6 is the source for RTICLK1.
		7h	Clock source7 is the source for RTICLK1.
		8h-Fh	VCLK is the source for RTICLK1.

NOTE: A list of the available clock sources is shown in the Table 2-29.

2.5.1.19 Clock Source Valid Status Register (CSVSTAT)

The CSVSTAT register, shown in Figure 2-26 and described in Table 2-38, indicates the status of usable clock sources.

Figure 2-26. Clock Source Valid Status Register (CSVSTAT) (offset = 54h)

31	8
Reserved	
R-0	

7	6	5	4	3	2	1	0
CLKSR7V	CLKSR6V	CLKSR5V	CLKSR4V	CLKSR3V	Reserved	CLKSR1V	CLKSR0V
R-1	R-0	R-0	R-1	R-1	R-0	R-0	R-0

LEGEND: R = Read only; -*n* = value after reset

Table 2-38. Clock Source Valid Register (CSVSTAT) Field Descriptions

Bit	Field	Value	Description
31-8	Reserved.	0	Reads return 0. Writes have no effect.
7-3	CLKSR[7-3]V		Clock source[7-0] valid.
		0	Clock source[7-0] is not valid.
		1	Clock source[7-0] is valid.
			Note: If the valid bit of the source of a clock domain is not set (that is, the clock source is not fully stable), the respective clock domain is disabled by the Global Clock Module (GCM).
2	Reserved.	0	Reads return 0. Writes have no effect.
1-0	CLKSR[1-0]V		Clock source[1–0] valid.
		0	Clock source[1–0] is not valid.
		1	Clock source[1–0] is valid.
			Note: If the valid bit of the source of a clock domain is not set (that is, the clock source is not fully stable), the respective clock domain is disabled.

NOTE: A list of the available clock sources is shown in the Table 2-29.

2.5.1.20 Memory Self-Test Global Control Register (MSTGCR)

The MSTGCR register, shown in Figure 2-27 and described in Table 2-39, controls several aspects of the PBIST (Programmable Built-In Self Test) memory controller.

Figure 2-27. Memory Self-Test Global Control Register (MSTGCR) (offset = 58h)

31	24	23	16
Reserved		Reserved	
R-0		R/WP-0	

15	10	9	8	7	4	3	0
Reserved		ROM_DIV		Reserved		MSTGENA	
R-0		R/WP-0		R-0		R/WP-5h	

LEGEND: R = Read only; R/W = Read/Write; WP = Write in privileged mode only; -*n* = value after reset

Table 2-39. Memory Self-Test Global Control Register (MSTGCR) Field Descriptions

Bit	Field	Value	Description
31-24	Reserved	0	Reads return 0. Writes have no effect.
23-16	Reserved	0-1	Reads return 0 or 1 and depends on what is written in privileged mode. The functionality of these bits are unavailable in this device.
15-10	Reserved	0	Reads return 0. Writes have no effect.
9-8	ROM_DIV		Prescaler divider bits for ROM clock source.
		0	ROM clock source is GCLK1 divided by 1. PBIST will reset for 16 VBUS cycles.
		1h	ROM clock source is GCLK1 divided by 2. PBIST will reset for 32 VBUS cycles.
		2h	ROM clock source is GCLK1 divided by 4. PBIST will reset for 64 VBUS cycles.
		3h	ROM clock source is GCLK1 divided by 8. PBIST will reset for 96 VBUS cycles.
7-4	Reserved	0	Reads return 0. Writes have no effect.
3-0	MSTGENA		Memory self-test controller global enable key **Note: Enabling the MSTGENA key will generate a reset to the state machine of the selected PBIST controller.**
		Ah	Memory self-test controller is enabled.
		Others	Memory self-test controller is disabled. **Note: It is recommended that a value of Ah be used to disable the memory self-test controller. This value will give maximum protection from a bit flip inducing event that would inadvertently enable the controller.**

2.5.1.21 Memory Hardware Initialization Global Control Register (MINITGCR)

The MINITGCR register, shown in Figure 2-28 and described in Table 2-40, enables automatic hardware memory initialization.

Figure 2-28. Memory Hardware Initialization Global Control Register (MINITGCR) (offset = 5Ch)

31	16
Reserved	
R-0	

15	4	3	0
Reserved		MINITGENA	
R-0		R/WP-5h	

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privileged mode only; -*n* = value after reset

Table 2-40. Memory Hardware Initialization Global Control Register (MINITGCR) Field Descriptions

Bit	Field	Value	Description
31-4	Reserved	0	Reads return 0. Writes have no effect.
3-0	MINITGENA		Memory hardware initialization global enable key.
		Ah	Global memory hardware initialization is enabled.
		Others	Global memory hardware initialization is disabled.
			Note: It is recommended that a value of 5h be used to disable memory hardware initialization. This value will give maximum protection from an event that would inadvertently enable the controller.

2.5.1.22 MBIST Controller/ Memory Initialization Enable Register (MSINENA)

The MSINENA register, shown in Figure 2-29 and described in Table 2-41, enables PBIST controllers for memory self test and the memory modules initialized during automatic hardware memory initialization.

Figure 2-29. MBIST Controller/Memory Initialization Enable Register (MSINENA) (offset = 60h)

31	16
MSIENA	
R/WP-0	

15	0
MSIENA	
R/WP-0	

LEGEND: R/W = Read/Write; WP = Write in privileged mode only; -*n* = value after reset

Table 2-41. MBIST Controller/Memory Initialization Enable Register (MSINENA) Field Descriptions

Bit	Field	Value	Description
31-0	MSIENA		PBIST controller and memory initialization enable register. In memory self-test mode, all the corresponding bits of the memories to be tested should be set before enabling the global memory self-test controller key (MSTGENA) in the MSTGCR register (offset 58h). The reason for this is that MSTGENA, in addition to being the global enable for all individual PBIST controllers, is the source for the reset generation to all the PBIST controller state machines. Disabling the MSTGENA or MINITGENA key (by writing from an Ah to any other value) will reset all the MSIENA[31-0] bits to their default values.

		0	*In memory self-test mode (MSTGENA = Ah):*
			PBIST controller [31-0] is disabled.
			In memory Initialization mode (MINITGENA = Ah):
			Memory module [31-0] auto hardware initialization is disabled.

		1	*In memory self-test mode (MSTGENA = Ah):* PBIST controller [31-0] is enabled. *In memory Initialization mode (MINITGENA = Ah):* Memory module [31-0] auto hardware initialization is enabled. **Note: Software should ensure that both the memory self-test global enable key (MSTGENA) and the memory hardware initialization global key (MINITGENA) are not enabled at the same time.**

2.5.1.23 MSTC Global Status Register (MSTCGSTAT)

The MSTCGSTAT register, shown in Figure 2-30 and described in Table 2-42, shows the status of the memory hardware initialization and the memory self-test.

Figure 2-30. MSTC Global Status Register (MSTCGSTAT) (offset = 68h)

31	16
Reserved	
R-0	

15	9	8	7	1	0
Reserved		MINIDONE	Reserved		MSTDONE
R-0		R/WPC-0	R-0		R/WP-0

LEGEND: R/W = Read/Write; R = Read only; C = Clear; WP = Write in privileged mode only; -*n* = value after reset

Table 2-42. MSTC Global Status Register (MSTCGSTAT) Field Descriptions

Bit	Field	Value	Description
31-9	Reserved	0	Reads return 0. Writes have no effect.
8	MINIDONE		Memory hardware initialization complete status. **Note: Disabling the MINITGENA key (By writing from a Ah to any other value) will clear the MINIDONE status bit to 0.** **Note: Individual memory initialization status is shown in the MINISTAT register.**
		0	*Read:* Memory hardware initialization is not complete for all memory. *Write:* A write of 0 has no effect.
		1	*Read:* Hardware initialization of all memory is completed. *Write:* The bit is cleared to 0.
7-1	Reserved	0	Reads return 0. Writes have no effect.
0	MSTDONE		Memory self-test run complete status. **Note: Disabling the MSTGENA key (by writing from a Ah to any other value) will clear the MSTDONE status bit to 0.**
		0	*Read:* Memory self-test is not completed. *Write:* A write of 0 has no effect.
		1	*Read:* Memory self-test is completed. *Write:* The bit is cleared to 0.

2.5.1.24 Memory Hardware Initialization Status Register (MINISTAT)

The MINISTAT register, shown in Figure 2-31 and described in Table 2-43, indicates the status of hardware memory initialization.

Figure 2-31. Memory Hardware Initialization Status Register (MINISTAT) (offset = 6Ch)

31	16
MIDONE	
R/WP-0	

15	0
MIDONE	
R/WP-0	

LEGEND: R/W = Read/Write; WP = Write in privileged mode only; -*n* = value after reset

Table 2-43. Memory Hardware Initialization Status Register (MINISTAT) Field Descriptions

Bit	Field	Value	Description
31-0	MIDONE		Memory hardware initialization status bit.
		0	*Read:* Memory module[31-0] hardware initialization is not completed. *Write:* A write of 0 has no effect.
		1	*Read:* Memory module[31-0] hardware initialization is completed. *Write:* The bit is cleared to 0. **Note: Disabling the MINITGENA key (by writing from a Ah to any other value) will reset all the individual status bits to 0.**

2.5.1.25 PLL Control Register 1 (PLLCTL1)

The PLLCTL1 register, shown in Figure 2-32 and described in Table 2-44, controls the output frequency of PLL1 (Clock Source 1 - FMzPLL). It also controls the behavior of the device if a PLL slip or oscillator failure is detected.

Figure 2-32. PLL Control Register 1 (PLLCTL1) (offset = 70h)

31	30-29	28-24
ROS	BPOS	PLLDIV
R/WP-0	R/WP-1h	R/WP-Fh

23	22	21-16
ROF	Reserved	REFCLKDIV
R/WP-0	R-0	R/WP-3h

15-0
PLLMUL
R/WP-4100h

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privileged mode only; -*n* = value after reset

Table 2-44. PLL Control Register 1 (PLLCTL1) Field Descriptions

Bit	Field	Value	Description
31	ROS		Reset on PLL Slip.
		0	Do not reset system when PLL slip is detected.
		1	Reset when PLL slip is detected.
			Note: BPOS (Bits 30-29) must also be enabled for ROS to be enabled.
30-29	BPOS		Bypass of PLL Slip.
		2h	Bypass on PLL Slip is disabled. If a PLL Slip is detected no action is taken.
		Others	Bypass on PLL Slip is enabled. If a PLL Slip is detected the device will automatically bypass the PLL and use the oscillator to provide the device clock.
			Note: If ROS (Bit 31) is set to 1, the device will be reset if a PLL Slip and the PLL will be bypassed after the reset occurs.
28-24	PLLDIV		PLL Output Clock Divider
			R = PLLDIV + 1 $f_{PLL\ CLK} = f_{post\_ODCLK} / R$
		0	$f_{PLL\ CLK} = f_{post\text{-}ODCLK} / 1$
		1h	$f_{PLL\ CLK} = f_{post\text{-}ODCLK} / 2$
		:	:
		1Fh	$f_{PLL\ CLK} = f_{post\text{-}ODCLK} / 32$
23	ROF		Reset on Oscillator Fail.
		0	Do not reset system when oscillator is out of range.
		1	The ROF bit enables the OSC_FAIL condition to generate a system reset. If the ROF bit in the PLLCTL1 register is set when the oscillator fails, then a system reset occurs.
22	Reserved	0	Value has no effect on PLL operation.
21-16	REFCLKDIV		Reference Clock Divider
			NR = REFCLKDIV + 1 $f_{INT\ CLK} = f_{OSCIN} / NR$
		0	$f_{INT\ CLK} = f_{OSCIN} / 1$
		1h	$f_{INT\ CLK} = f_{OSCIN} / 2$
		:	:
		3Fh	$f_{INT\ CLK} = f_{OSCIN} / 64$
15-0	PLLMUL		PLL Multiplication Factor
			NF = (PLLMUL / 256) + 1, valid multiplication factors are from 1 to 256. $f_{VCO\ CLK} = f_{INT\ CLK} \times NF$
		0h	$f_{VCO\ CLK} = f_{INT\ CLK} \times 1$
		100h	$f_{VCO\ CLK} = f_{INT\ CLK} \times 2$
		:	:
		5B00h	$f_{VCO\ CLK} = f_{INT\ CLK} \times 92$
		5C00h	$f_{VCO\ CLK} = f_{INT\ CLK} \times 93$
		:	:
		FF00h	$f_{VCO\ CLK} = f_{INT\ CLK} \times 256$

2.5.1.26 PLL Control Register 2 (PLLCTL2)

The PLLCTL2 register, shown in Figure 2-33 and described in Table 2-45, controls the modulation characteristics and the output divider of the PLL.

Figure 2-33. PLL Control Register 2 (PLLCTL2) (offset = 74h)

31	30 22	21	20 16
FMENA	SPREADINGRATE	Rsvd	MULMOD
R/WP-0	R/WP-1FFh	R-0	R/WP-0

15 12	11 9	8 0
MULMOD	ODPLL	SPR_AMOUNT
R/WP-7h	R/WP-0	R/WP-0

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privileged mode only; -*n* = value after reset

Table 2-45. PLL Control Register 2 (PLLCTL2) Field Descriptions

Bit	Field	Value	Description
31	FMENA		Frequency Modulation Enable.
		0	Disable frequency modulation.
		1	Enable frequency modulation.
30-22	SPREADINGRATE		NS = SPREADINGRATE + 1 $f_{mod} = f_s = f_{INT\ CLK}/(2 \times NS)$
		0	$f_{mod} = f_s = f_{INT\ CLK} / (2 \times 1)$
		1h	$f_{mod} = f_s = f_{INT\ CLK} / (2 \times 2)$
		:	:
		1FFh	$f_{mod} = f_s = f_{INT\ CLK} / (2 \times 512)$
21	Reserved	0	Value has no effect on PLL operation.
20-12	MULMOD		Multiplier Correction when Frequency Modulation is enabled.
			When FMENA = 0, MUL_when_MOD = 0; when FMENA = 1, MUL_when_MOD = (MULMOD / 256)
		0	No adder to NF.
		8h	MUL_when_MOD = 8/256
		9h	MUL_when_MOD = 9/256
		:	:
		1FFh	MUL_when_MOD = 511/256
11-9	ODPLL		Internal PLL Output Divider
			OD = ODPLL + 1 $f_{post\text{-}ODCLK} = f_{VCO\ CLK}/OD$
			Note: PLL output clock is gated off, if ODPLL is changed while the PLL is active.
		0	$f_{post\text{-}ODCLK} = f_{VCO\ CLK} / 1$
		1h	$f_{post\text{-}ODCLK} = f_{VCO\ CLK} / 2$
		:	:
		7h	$f_{post\text{-}ODCLK} = f_{VCO\ CLK} / 8$
8-0	SPR_AMOUNT		Spreading Amount
			NV = (SPR_AMOUNT + 1)/2048
			NV ranges from 1/2048 to 512/2048
			Note that the PLL output clock is disabled for 1 modulation period, if the SPR_AMOUNT field is changed while the frequency modulation is enabled. If frequency modulation is disabled and SPR_AMOUNT is changed, there is no effect on the PLL output clock.
		0	NV = 1/2048
		1h	NV = 2/2048
		:	:
		1FFh	NV = 512/2048

2.5.1.27 SYS Pin Control Register 10 (SYSPC10)

The SYSPC10 register, shown in Figure 2-34 and described in Table 2-46, controls the function of the ECPCLK slew mode.

Figure 2-34. SYS Pin Control Register 10 (SYSPC10) (offset = 78h)

31	16
Reserved	
R-0	

15	1	0
Reserved		ECPCLK_SLEW
R-0		R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 2-46. SYS Pin Control Register 10 (SYSPC10) Field Descriptions

Bit	Field	Value	Description
31-1	Reserved	0	Reads return 0. Writes have no effect.
0	ECPCLK_SLEW		ECPCLK slew control. This bit controls between the fast or slow slew mode.
		0	Fast mode is enabled; the normal output buffer is used for this pin.
		1	Slow mode is enabled; slew rate control is used for this pin.

2.5.1.28 Die Identification Register Lower Word (DIEIDL)

The DIEIDL register, shown in Figure 2-35 and described in Table 2-47, contains information about the die wafer number, and X, Y wafer coordinates.

Figure 2-35. Die Identification Register, Lower Word (DIEIDL) [offset = 7Ch]

31 24	23 16
WAFER #	Y WAFER COORDINATE
R-D	R-D

15 12	11 0
Y WAFER COORDINATE	X WAFER COORDINATE
R-D	R-D

LEGEND: R = Read only; D = value is device specific; -*n* = value after reset

Table 2-47. Die Identification Register, Lower Word (DIEIDL) Field Descriptions

Bit	Field	Description
31-24	WAFER #	These read-only bits contain the wafer number of the device.
23-12	Y WAFER COORDINATE	These read-only bits contain the Y wafer coordinate of the device.
11-0	X WAFER COORDINATE	These read-only bits contain the X wafer coordinate of the device.

NOTE: Die Identification Information

The die identification information will vary from unit to unit. This information is programmed by TI as part of the initial device test procedure.

2.5.1.29 Die Identification Register Upper Word (DIEIDH)

The DIEIDH register, shown in Figure 2-36 and described in Table 2-48, contains information about the die lot number.

Figure 2-36. Die Identification Register, Upper Word (DIEIDH) [offset = 80h]

31 24	23 16
Reserved	LOT #
R-0	R-D

15 0
LOT #
R-D

LEGEND: R = Read only; D = value is device specific; -*n* = value after reset

Table 2-48. Die Identification Register, Upper Word (DIEIDH) Field Descriptions

Bit	Field	Description
31-24	Reserved	Reserved for TI use. Writes have no effect.
23-0	LOT #	This read-only register contains the device lot number.

NOTE: Die Identification Information

The die identification information will vary from unit to unit. This information is programmed by TI as part of the initial device test procedure.

2.5.1.30 LPO/Clock Monitor Control Register (LPOMONCTL)

The LPOMONCTL register, shown in Figure 2-37 and described in Table 2-49, controls the Low Frequency (Clock Source 4) and High Frequency (Clock Source 5) Low Power Oscillator's trim values.

Figure 2-37. LPO/Clock Monitor Control Register (LPOMONCTL) (offset = 088h)

31 25	24	23 17	16
Reserved	BIAS ENABLE	Reserved	OSCFRQCONFIGCNT
R-0	R/WP-1	R-0	R/WP-0

15 13	12 8	7 5	4 0
Reserved	HFTRIM	Reserved	LFTRIM
R-0	R/WP-10h	R-0	R/WP-10h

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privileged mode only; -*n* = value after reset

Table 2-49. LPO/Clock Monitor Control Register (LPOMONCTL) Field Descriptions

Bit	Field	Value	Description
31-25	Reserved	0	Reads return 0. Writes have no effect.
24	BIAS ENABLE		Bias enable.
		0	The bias circuit inside the low-power oscillator (LPO) is disabled.
		1	The bias circuit inside the low-power oscillator (LPO) is enabled.
23-17	Reserved	0	Reads return 0. Writes have no effect.
16	OSCFRQCONFIGCNT		Configures the counter based on OSC frequency.
		0	*Read:* OSC freq is ≤ 20MHz. *Write:* A write of 0 has no effect.
		1	*Read:* OSC freq is > 20MHz and ≤ 80MHz. *Write:* A write of 1 has no effect.
15-13	Reserved	0	Reads return 0. Writes have no effect.

Table 2-49. LPO/Clock Monitor Control Register (LPOMONCTL) Field Descriptions (continued)

Bit	Field	Value	Description
12-8	HFTRIM		High-frequency oscillator trim value. This four-bit value is used to center the HF oscillator's frequency. **Caution: This value should only be changed when the HF oscillator is not the source for a clock domain, otherwise a system failure could result.** The following values are the ratio: f / fo in the F021 process.
		0	29.52
		1h	34.24%
		2h	38.85%
		3h	43.45%
		4h	47.99%
		5h	52.55%
		6h	57.02%
		7h	61.46%
		8h	65.92%
		9h	70.17
		Ah	74.55%
		Bh	78.92%
		Ch	83.17%
		Dh	87.43%
		Eh	91.75%
		Fh	95.89%
		10h	100.00% Default at Reset.
		11h	104.09
		12h	108.17
		13h	112.32
		14h	116.41
		15h	120.67
		16h	124.42
		17h	128.38
		18h	132.24
		19h	136.15
		1Ah	140.15
		1Bh	143.94
		1Ch	148.02
		1Dh	151.80x
		1Eh	155.50x
		1Fh	159.35%
7-5	Reserved	0	Reads return 0. Writes have no effect.

Table 2-49. LPO/Clock Monitor Control Register (LPOMONCTL) Field Descriptions (continued)

Bit	Field	Value	Description
4-0	LFTRIM		Low-frequency oscillator trim value. This four-bit value is used to center the LF oscillator's frequency. **Caution: This value should only be changed when the LF oscillator is not the source for a clock domain, otherwise a system failure could result.** The following values are the ratio: f / fo in the F021 process.
		0	20.67
		1h	25.76
		2h	30.84
		3h	35.90
		4h	40.93
		5h	45.95
		6h	50.97
		7h	55.91
		8h	60.86
		9h	65.78
		Ah	70.75
		Bh	75.63
		Ch	80.61
		Dh	85.39
		Eh	90.23
		Fh	95.11
		10h	100.00% Default at Reset
		11h	104.84
		12h	109.51
		13h	114.31
		14h	119.01
		15h	123.75
		16h	128.62
		17h	133.31
		18h	138.03
		19h	142.75
		1Ah	147.32
		1Bh	152.02
		1Ch	156.63
		1Dh	161.38
		1Eh	165.90
		1Fh	170.42

2.5.1.31 Clock Test Register (CLKTEST)

The CLKTEST register, shown in Figure 2-38 and described in Table 2-50, controls the clock signal that is supplied to the ECLK pin for test and debug purposes.

NOTE: Clock Test Register Usage

This register should only be used for test and debug purposes.

Figure 2-38. Clock Test Register (CLKTEST) (offset = 8Ch)

31 27	26	25	24
Reserved	TEST	RANGEDET CTRL	RANGEDET ENASSEL
R-0	R/WP-0	R/WP-0	R/WP-0

23 20	19 16
Reserved	CLK_TEST_EN
R-0	R/WP-Ah

15 12	11 8	7 5	4 0
Reserved	SEL_GIO_PIN	Reserved	SEL_ECP_PIN
R-0	R/WP-0	R-0	R/WP-0

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privileged mode only; -*n* = value after reset

Table 2-50. Clock Test Register (CLKTEST) Field Descriptions

Bit	Field	Value	Description
31-27	Reserved	0	Reads return 0. Writes have no effect.
26	TEST	0	This bit is used for test purposes. It must be written to 0.
25	RANGEDETCTRL		Range detection control. This bit's functionality is dependant on the state of the RANGEDETENSSEL bit (Bit 24) of the CLKTEST register.
		0	The clock monitor range detection circuitry (RANGEDETECTENABLE) is disabled.
		1	The clock monitor range detection circuitry (RANGEDETECTENABLE) is enabled.
24	RANGEDETENASSEL		Selects range detection enable. This bit resets asynchronously on power on reset.
		0	The range detect enable is generated by the hardware in the clock monitor wrapper.
		1	The range detect enable is controlled by the RANGEDETCTRL bit (Bit 25) of the CLKTEST register.
23-20	Reserved	0	Reads return 0. Writes have no effect.
19-16	CLK_TEST_EN		Clock test enable. This bit enables the clock going to the ECLK pin. This bit field enables or disables clock going to device pins. Two pins in a device can get clock sources by enabling CLK_TEST_EN bits. One pin is the ECP and second pin is a device specific GIO pin. These bits need to asynchronously reset. **Note: The ECLK pin must also be placed into Functional mode by setting the ECPCLKFUN bit to 1 in the SYSPC1 register.**
		5h	Clock going to ECLK pin is enabled.
		Others	Clock going to ECLK pin is disabled.
15-12	Reserved	0	Reads return 0. Writes have no effect.

Table 2-50. Clock Test Register (CLKTEST) Field Descriptions (continued)

Bit	Field	Value	Description
11-8	SEL_GIO_PIN		GIOB[0] pin clock source valid, clock source select
		0	Oscillator valid status
		1h	PLL1 valid status
		2h-4h	Reserved
		5h	High-frequency LPO (Low-Power Oscillator) clock output valid status [CLK10M]
		6h	PLL2 valid status
		7h	Reserved
		8h	Low-frequency LPO (Low-Power Oscillator) clock output valid status [CLK80K]
		9h-Ch	Oscillator valid status
		Dh	Reserved
		Eh	VCLKA4
		Fh	Oscillator valid status
7-5	Reserved	0	Reads return 0. Writes have no effect.
4-0	SEL_ECP_PIN		ECLK pin clock source select **Note: Only valid clock sources can be selected for the ECLK pin. Valid clock sources are displayed by the CSVSTAT register.**
		0	Oscillator clock
		1h	PLL1 clock output
		2h	Reserved
		3h	EXTCLKIN1
		4h	Low-frequency LPO (Low-Power Oscillator) clock [CLK80K]
		5h	High-frequency LPO (Low-Power Oscillator) clock [CLK10M]
		6h	PLL2 clock output
		7h	EXTCLKIN2
		8h	GCLK1
		9h	RTI1 Base
		Ah	Reserved
		Bh	VCLKA1
		Ch	VCLKA2
		Dh	Reserved
		Eh	VCLKA4_DIVR
		Fh	Flash HD Pump Oscillator
		10h	Reserved
		11h	HCLK
		12h	VCLK
		13h	VCLK2
		14h	VCLK3
		15h-16h	Reserved
		17h	EMAC clock output
		18h-1Fh	Reserved

NOTE: Non-implemented clock sources should not be enabled or used.

2.5.1.32 DFT Control Register (DFTCTRLREG)

This register is shown in Figure 2-39 and described in Table 2-51.

Figure 2-39. DFT Control Register (DFTCTRLREG) (offset = 90h)

31 16
Reserved
R-0

15 14	13 12	11 10	9 8	7 4	3 0
Reserved	DFTWRITE	Reserved	DFTREAD	Reserved	TEST_MODE_KEY
R-0	R/WP-1h	R-0	R/WP-1h	R-0	R/WP-5h

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privileged mode only; -*n* = value after reset

Table 2-51. DFT Control Register (DFTCTRLREG) Field Descriptions

Bit	Field	Value	Description
31-14	Reserved	0	Reads return 0. Writes have no effect.
13-12	DFTWRITE		DFT logic access. For F021: DFTWRITE[0] = 0 and DFTREAD[0] = 0 configured in stress mode. DFTWRITE[1] = 0 and DFTREAD[1] = 0 configured in stress mode. DFTWRITE[0] = 0 and DFTREAD[0] = 0 configured in fast mode. DFTWRITE[1] = 1 and DFTREAD[1] = 1 configured in fast mode. DFTWRITE[0] = 1 and DFTREAD[0] = 1 configured in slow mode. DFTWRITE[1] = 0 and DFTREAD[1] = 0 configured in slow mode. DFTWRITE[0] = 1 and DFTREAD[0] = 1 configured in screen mode. DFTWRITE[1] = 1 and DFTREAD[1] = 1 configured in screen mode.
11-10	Reserved	0	Reads return 0. Writes have no effect.
9-8	DFTREAD		DFT logic access. For F021: DFTWRITE[0] = 0 and DFTREAD[0] = 0 configured in stress mode. DFTWRITE[1] = 0 and DFTREAD[1] = 0 configured in stress mode. DFTWRITE[0] = 0 and DFTREAD[0] = 0 configured in fast mode. DFTWRITE[1] = 1 and DFTREAD[1] = 1 configured in fast mode. DFTWRITE[0] = 1 and DFTREAD[0] = 1 configured in slow mode. DFTWRITE[1] = 0 and DFTREAD[1] = 0 configured in slow mode. DFTWRITE[0] = 1 and DFTREAD[0] = 1 configured in screen mode. DFTWRITE[1] = 1 and DFTREAD[1] = 1 configured in screen mode.
7-4	Reserved	0	Reads return 0. Writes have no effect.
3-0	TEST_MODE_KEY		Test mode key. This register is for internal TI use only.
		0 - Fh (except Ah)	Register key disable. All bits in this register will maintain their default value and cannot be written.
		Ah	Register key enable. ALL the bits can be written to only when the key is enabled. On reset, these bits will be set to 5h.

2.5.1.33 DFT Control Register 2 (DFTCTRLREG2)

This register is shown in Figure 2-40 and described in Table 2-52. For information on filtering the RFSLIP see Section 2.5.2.7.

Figure 2-40. DFT Control Register 2 (DFTCTRLREG2) (offset = 94h)

31 — 16
IMPDF(27:12)
R/WP-0

15 — 4	3 — 0
IMPDF(11:0)	TEST_MODE_KEY
R/WP-0	R/WP-5h

LEGEND: R/W = Read/Write; WP = Write in privileged mode only; -*n* = value after reset

Table 2-52. DFT Control Register 2 (DFTCTRLREG2) Field Descriptions

Bit	Field	Value	Description
31-4	IMPDF[27:0]		DFT Implementation defined bits.
		0	IMPDF[27:0] is disabled.
		1	IMPDF[27:0] is enabled.
3-0	TEST_MODE_KEY		Test mode key. This register is for internal TI use only.
		0-Fh (except Ah)	Register key disable. All bits in this register will maintain their default value and cannot be written.
		Ah	Register key enable. ALL the bits can be written to only when the key is enabled.

2.5.1.34 General Purpose Register (GPREG1)

This register is shown in Figure 2-41 and described in Table 2-53. For information on filtering the RFSLIP, see Section 2.5.2.7.

Figure 2-41. General Purpose Register (GPREG1) (offset = A0h)

31 … 26	25 … 20	19 … 16
Reserved	PLL1_FBSLIP_FILTER_COUNT	PLL1_FBSLIP_FILTER_KEY
R-0	R/WP-0	R/WP-0

15 … 0
Reserved
R/WP-0

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privileged mode only; -*n* = value after reset

Table 2-53. General Purpose Register (GPREG1) Field Descriptions

Bit	Field	Value	Description
31-26	Reserved	0	Reads return 0. Writes have no effect.
25-20	PLL1_FBSLIP_FILTER_COUNT		FBSLIP down counter programmed value. Configures the system response when a FBSLIP is indicated by the PLL macro. When PLL1_FBSLIP_FILTER_KEY is not Ah, the down counter counts from the programmed value on every LPO high-frequency clock once PLL macro indicates FBSLIP. When the count reaches 0, if the synchronized FBSLIP signal is still high, an FBSLIP condition is indicated to the system module and is captured in the global status register. When the FBSLIP signal from the PLL macro is de-asserted before the count reaches 0, the counter is reloaded with the programmed value. On reset, counter value is 0. Counter must be programmed to a non-zero value and enabled for the filtering to be enabled.
		0	Filtering is disabled.
		1h	Filtering is enabled. Every slip is recognized.
		2h	Filtering is enabled. The slip must be at least 2 HF LPO cycles wide in order to be recognized as a slip.
		:	:
		3Fh	Filtering is enabled. The slip must be at least 63 HF LPO cycles wide in order to be recognized as a slip.
19-16	PLL1_FBSLIP_FILTER_KEY		Enable the FBSLIP filtering.
		5h	On reset, the FBSLIP filter is disabled and the FBSLIP passes through.
		Fh	This is an unsupported value. You should avoid writing this value to this bit field.
		All other values	FBSLIP filtering is enabled. Recommended to program Ah in this bit field. Enabling of the FBSLIP occurs when the KEY is programmed and a non-zero value is present in the COUNT field.
15-0	Reserved	0-1	Reads return 0 or 1 and write in privilege mode. The functionality of this bit is unavailable in this device.

2.5.1.35 System Software Interrupt Request 1 Register (SSIR1)

The SSIR1 register, shown in Figure 2-42 and described in Table 2-54, is used for software interrupt generation.

Figure 2-42. System Software Interrupt Request 1 Register (SSIR1) (offset = B0h)

31	16
Reserved	
R-0	

15	8	7	0
SSKEY1		SSDATA1	
R/W-0		R/W-0	

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 2-54. System Software Interrupt Request 1 Register (SSIR1) Field Descriptions

Bit	Field	Value	Description
31-16	Reserved	0	Reads return 0. Writes have no effect.
15-8	SSKEY1	0-FFh	System software interrupt request key. A 075h written to these bits initiates IRQ/FIQ interrupts. Data in this field is always read as 0. The SSKEY1 field can be written into only if the write data matches the key (75h). The SSDATA1 field can only be written into if the write data into this field, the SSKEY1 field, matches the key (75h).
7-0	SSDATA1	0-FFh	System software interrupt data. These bits contain user read/write register bits. They may be used by the application software as different entry points for the interrupt routine. The SSDATA1 field cannot be written into unless the write data into the SSKEY1 field matches the key (75h); therefore, byte writes cannot be performed on the SSDATA1 field.

NOTE: This register is mirrored at offset FCh for compatibility reasons.

2.5.1.36 System Software Interrupt Request 2 Register (SSIR2)

The SSIR2 register, shown in Figure 2-43 and described in Table 2-55, is used for software interrupt generation.

Figure 2-43. System Software Interrupt Request 2 Register (SSIR2) (offset = B4h)

31	16
Reserved	
R-0	

15	8	7	0
SSKEY2		SSDATA2	
R/W-0		R/W-0	

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 2-55. System Software Interrupt Request 2 Register (SSIR2) Field Descriptions

Bit	Field	Value	Description
31-16	Reserved	0	Reads return 0. Writes have no effect.
15-8	SSKEY2	0-FFh	System software interrupt2 request key. A 84h written to these bits initiates IRQ/FIQ interrupts. Data in this field is always read as 0. The SSKEY2 field can be written into only if the write data matches the key (84h). The SSDATA2 field can only be written into if the write data into this field, the SSKEY2 field, matches the key (84h).
7-0	SSDATA2	0-FFh	System software interrupt data. These bits contain user read/write register bits. They may be used by the application software as different entry points for the interrupt routine. The SSDATA2 field cannot be written into unless the write data into the SSKEY2 field matches the key (84h); therefore, byte writes cannot be performed on the SSDATA2 field.

2.5.1.37 System Software Interrupt Request 3 Register (SSIR3)

The SSIR3 register, shown in Figure 2-44 and described in Table 2-56, is used for software interrupt generation.

Figure 2-44. System Software Interrupt Request 3 Register (SSIR3) (offset = B8h)

<table>
<tr><td>31</td><td></td><td></td><td>16</td></tr>
<tr><td colspan="4">Reserved</td></tr>
<tr><td colspan="4">R-0</td></tr>
<tr><td>15</td><td>8</td><td>7</td><td>0</td></tr>
<tr><td colspan="2">SSKEY3</td><td colspan="2">SSDATA3</td></tr>
<tr><td colspan="2">R/W-0</td><td colspan="2">R/W-0</td></tr>
</table>

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 2-56. System Software Interrupt Request 3 Register (SSIR3) Field Descriptions

Bit	Field	Value	Description
31-16	Reserved	0	Reads return 0. Writes have no effect.
15-8	SSKEY3	0-FFh	System software interrupt request key. A 93h written to these bits initiates IRQ/FIQ interrupts. Data in this field is always read as 0. The SSKEY3 field can be written into only if the write data matches the key (93h). The SSDATA3 field can only be written into if the write data into this field, the SSKEY3 field, matches the key (93h).
7-0	SSDATA3	0-FFh	System software interrupt data. These bits contain user read/write register bits. They may be used by the application software as different entry points for the interrupt routine. The SSDATA3 field cannot be written into unless the write data into the SSKEY3 field matches the key (93h); therefore, byte writes cannot be performed on the SSDATA3 field.

2.5.1.38 System Software Interrupt Request 4 Register (SSIR4)

The SSIR4 register, shown in Figure 2-45 and described in Table 2-57, is used for software interrupt generation.

Figure 2-45. System Software Interrupt Request 4 Register (SSIR4) (offset = BCh)

31	16
Reserved	
R-0	

15	8	7	0
SSKEY4		SSDATA4	
R/W-0		R/W-0	

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 2-57. System Software Interrupt Request 4 Register (SSIR4) Field Descriptions

Bit	Field	Value	Description
31-16	Reserved	0	Reads return 0. Writes have no effect.
15-8	SSKEY4	0-FFh	System software interrupt2 request key. A A2h written to these bits initiates IRQ/FIQ interrupts. Data in this field is always read as 0. The SSKEY4 field can be written into only if the write data matches the key (A2h). The SSDATA4 field can only be written into if the write data into this field, the SSKEY4 field, matches the key (A2h).
7-0	SSDATA4	0-FFh	System software interrupt data. These bits contain user read/write register bits. They may be used by the application software as different entry points for the interrupt routine. The SSDATA4 field cannot be written into unless the write data into the SSKEY4 field matches the key (A2h); therefore, byte writes cannot be performed on the SSDATA4 field.

2.5.1.39 RAM Control Register (RAMGCR)

NOTE: The RAM_DFT_EN bits are for TI internal use only.

The contents of the RAM_DFT_EN field should not be changed.

Figure 2-46. RAM Control Register (RAMGCR) (offset = C0h)

31–20	19–16
Reserved	RAM_DFT_EN
R-0	R/WP-5h

15	14	13	12	11	10	9	8
Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
R-0	R/WP-0	R-0	R/WP-0	R-0	R/WP-0	R-0	R/WP-0

7	6	5	4	3	2	1	0
Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
R-0	R/WP-0	R-0	R/WP-0	R-0	R/WP-0	R-0	R/WP-0

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privileged mode only; -*n* = value after reset

Table 2-58. RAM Control Register (RAMGCR) Field Descriptions

Bit	Field	Value	Description
31-20	Reserved	0	Reads return 0. Writes have no effect.
19-16	RAM_DFT_EN		Functional mode RAM DFT (Design For Test) port enable key. **Note: For TI internal use only.**
		Ah	RAM DFT port is enabled.
		Others	RAM DFT port is disabled. **Note: It is recommended that a value of 5h be used to disable the RAM DFT port. This value will give maximum protection from a bit-flip inducing event that would inadvertently enable the controller.**
15	Reserved	0	Reads return 0. Writes have no effect.
14	Reserved	0-1	Reads return 0 or 1 depends on what is written in privileged mode. The functionality of this bit is unavailable in this device.
13	Reserved	0	Reads return 0. Writes have no effect.
12	Reserved	0-1	Reads return 0 or 1 depends on what is written in privileged mode. The functionality of this bit is unavailable in this device.
11	Reserved	0	Reads return 0. Writes have no effect.
10	Reserved	0-1	Reads return 0 or 1 depends on what is written in privileged mode. The functionality of this bit is unavailable in this device.
9	Reserved	0	Reads return 0. Writes have no effect.
8	Reserved	0-1	Reads return 0 or 1 depends on what is written in privileged mode. The functionality of this bit is unavailable in this device.
7	Reserved	0	Reads return 0. Writes have no effect.
6	Reserved	0-1	Reads return 0 or 1 depends on what is written in privileged mode. The functionality of this bit is unavailable in this device.
5	Reserved	0	Reads return 0. Writes have no effect.
4	Reserved	0-1	Reads return 0 or 1 depends on what is written in privileged mode. The functionality of this bit is unavailable in this device.
3	Reserved	0	Reads return 0. Writes have no effect.
2	Reserved	0-1	Reads return 0 or 1 depends on what is written in privileged mode. The functionality of this bit is unavailable in this device.
1	Reserved	0	Reads return 0. Writes have no effect.
0	Reserved	0-1	Reads return 0 or 1 depends on what is written in privileged mode. The functionality of this bit is unavailable in this device.

2.5.1.40 Bus Matrix Module Control Register 1 (BMMCR1)

The BMMCR1 register, shown in Figure 2-47 and described in Table 2-59, allows RAM and Program (Flash) memory addresses to be swapped.

Figure 2-47. Bus Matrix Module Control Register 1 (BMMCR) (offset = C4h)

31-16
Reserved
R-0

15-4	3-0
Reserved	MEMSW
R-0	R/WP-Ah

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privileged mode only; -*n* = value after reset

Table 2-59. Bus Matrix Module Control Register 1 (BMMCR) Field Descriptions

Bit	Field	Value	Description
31-4	Reserved	0	Reads return 0. Writes have no effect.
3-0	MEMSW		Memory swap key. **Note: A CPU reset must be issued after the memory swap key has been changed for the memory swap to occur. A CPU reset can be initiated by changing the state of the CPU RESET bit in the CPURSTCR register.**
		Ah	Default memory-map: Program memory (Flash) starts at address 0. eSRAM starts at address 800 0000h.
		5h	Swapped memory-map: eSRAM starts at address 0. Program memory (Flash) starts at address 800 0000h.
		Others	The device memory-map is unchanged.

2.5.1.41 CPU Reset Control Register (CPURSTCR)

The CPURSTCR register shown in Figure 2-48 and described in Table 2-60 allows a reset to the Cortex-R5F CPU to be generated.

NOTE: The register bits in CPURSTCR are designated as high-integrity bits and have been implemented with error-correcting logic such that each bit, although read and written as a single bit, is actually a multi-bit key with error correction capability. As such, single-bit flips within the "key" can be corrected allowing protection of the system as a whole. An error detected is signaled to the ESM module.

Figure 2-48. CPU Reset Control Register (CPURSTCR) (offset = CCh)

31 17	16
Reserved	Reserved
R-0	R/WP-0

15 1	0
Reserved	CPU RESET
R-0	R/WP-0

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privileged mode only; -*n* = value after reset

Table 2-60. CPU Reset Control Register (CPURSTGCR) Field Descriptions

Bit	Field	Value	Description
31-1	Reserved	0	Reads return 0. Writes have no effect.
0	CPU RESET		CPU RESET. Only the CPU is reset whenever this bit is toggled. There is no system reset.

2.5.1.42 Clock Control Register (CLKCNTL)

The CLKCNTL register, shown in Figure 2-49 and described in Table 2-61, controls peripheral reset and the peripheral clock divide ratios.

NOTE: VCLK and VCLK2 clock ratio restrictions.

The VCLK2 frequency must always be greater than or equal to the VCLK frequency. The VCLK2 frequency must be an integer multiple of the VCLK frequency.

In addition, the VCLK and VCLK2 clock ratios must not be changed simultaneously. When increasing the frequency (decreasing the divider), first change the VCLK2R field and then change the VCLKR field. When reducing the frequency (increasing the divider), first change the VCLKR field and then change the VCLK2R field.

You should do a read-back between the two writes. This assures that there are enough clock cycles between the two writes.

Figure 2-49. Clock Control Register (CLKCNTL) (offset = D0h)

31 28	27 24	23 20	19 16
Reserved	VCLK2R	Reserved	VCLKR
R-0	R/WP-1h	R-0	R/WP-1h

15 9	8	7 0
Reserved	PENA	Reserved
R-0	R/WP-0	R-0

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privileged mode only; -*n* = value after reset

Table 2-61. Clock Control Register (CLKCNTL) Field Descriptions

Bit	Field	Value	Description
31-28	Reserved	0	Reads return 0. Writes have no effect.
27-24	VCLK2R		VBUS clock2 ratio.
			Note: The VCLK2 frequency must always be greater than or equal to the VCLK frequency. The VCLK2 frequency must be an integer multiple of the VCLK frequency. In addition, the VCLK and VCLK2 clock ratios must not be changed simultaneously.
		0	The VCLK2 speed is HCLK divided by 1.
		:	:
		Fh	The VCLK2 speed is HCLK divided by 16.
23-20	Reserved	0	Reads return 0. Writes have no effect.
19-16	VCLKR		VBUS clock ratio.
			Note: The VCLK2 frequency must always be greater than or equal to the VCLK frequency. The VCLK2 frequency must be an integer multiple of the VCLK frequency. In addition, the VCLK and VCLK2 clock ratios must not be changed simultaneously.
		0	The VCLK speed is HCLK divided by 1.
		:	:
		Fh	The VCLK speed is HCLK divided by 16.
15-9	Reserved	0	Reads return 0. Writes have no effect.
8	PENA		Peripheral enable bit. The application must set this bit before accessing any peripheral.
		0	The global peripheral/peripheral memory frames are in reset.
		1	All peripheral/peripheral memory frames are out of reset.
7-0	Reserved	0	Reads return 0. Writes have no effect.

2.5.1.43 ECP Control Register (ECPCNTL)

The ECP register, shown in Figure 2-50 and described in Table 2-62, configures the ECLK pin in functional mode.

NOTE: ECLK Functional mode configuration.

The ECLK pin must be placed into Functional mode by setting the ECPCLKFUN bit to 1 in the SYSPC1 register before a clock source will be visible on the ECLK pin.

Figure 2-50. ECP Control Register (ECPCNTL) (offset = D4h)

31 — 25	24	23	22 — 18	17 — 16
Reserved	ECPSSEL	ECPCOS	Reserved	Reserved
R-0	R/W-0	R/W-0	R-0	R/W-0

15 — 0
ECPDIV
R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 2-62. ECP Control Register (ECPCNTL) Field Descriptions

Bit	Field	Value	Description
31-25	Reserved	0	Reads return 0. Writes have no effect.
24	ECPSSEL		This bit allows the selection between VCLK and OSCIN as the clock source for ECLK. **Note: Other ECLK clock sources are available for debug purposes by configuring the CLKTEST register.**
		0	VCLK is selected as the ECP clock source.
		1	OSCIN is selected as the ECP clock source.
23	ECPCOS		ECP continue on suspend. **Note: Suspend mode is entered while performing certain JTAG debugging operations.**
		0	ECLK output is disabled in suspend mode. ECLK output will be shut off and will not be seen on the I/O pin of the device.
		1	ECLK output is not disabled in suspend mode. ECLK output will not be shut off and will be seen on the I/O pin of the device.
22-18	Reserved	0	Reads return 0. Writes have no effect.
17-16	Reserved	0	Reads return 0 or 1 depends on what is written. The functionality of this bit is unavailable in this device.
15-0	ECPDIV	0-FFFFh	ECP divider value. The value of ECPDIV bits determine the external clock (ECP clock) frequency as a ratio of VBUS clock or OSCIN as shown in the formula: $ECLK = \frac{VCLK or OSCIN}{(ECPDIV + 1)}$

2.5.1.44 DEV Parity Control Register 1 (DEVCR1)

This register is shown in Figure 2-51 and described in Table 2-63.

Figure 2-51. DEV Parity Control Register 1 (DEVCR1) (offset = DCh)

31–16
Reserved
R-0

15–4	3–0
Reserved	DEVPARSEL
R-0	R/WP-Ah

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privileged mode only; -*n* = value after reset

Table 2-63. DEV Parity Control Register 1 (DEVCR1) Field Descriptions

Bit	Field	Value	Description
31-4	Reserved	0	Reads return 0. Writes have no effect.
3-0	DEVPARSEL		Device parity select bit key. **Note: After an odd (DEVPARSEL = Ah) or even (DEVPARSEL = 5h) scheme is programmed into the DEVPARSEL register, any one bit change can be detected and will retain its programmed scheme. More than one bit changes in DEVPARSEL will cause a default to odd parity scheme.**
		5h	The device parity is even.
		Ah	The device parity is odd.

2.5.1.45 System Exception Control Register (SYSECR)

The SYSECR register, shown in Figure 2-52 and described in Table 2-64, is used to generate a software reset.

NOTE: The register bits in SYSECR are designated as high-integrity bits and have been implemented with error-correcting logic such that each bit, although read and written as a single bit, is actually a multi-bit key with error correction capability. As such, single-bit flips within the “key” can be corrected allowing protection of the system as a whole. An error detected is signaled to the ESM module.

Figure 2-52. System Exception Control Register (SYSECR) (offset = E0h)

31–16
Reserved
R-0

15	14	13–0
RESET1	RESET0	Reserved
R/WP-0	R/WP-1	R-0

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privileged mode only; -*n* = value after reset

Table 2-64. System Exception Control Register (SYSECR) Field Descriptions

Bit	Field	Value	Description
31-16	Reserved	0	Reads return 0. Writes have no effect.
15-14	RESET[1-0]		Software reset bits. Setting RESET1 or clearing RESET0 causes a system software reset.
		1h	No reset will occur.
		0, 2h-3h	A global system reset will occur.
13-0	Reserved	0	Reads return 0. Writes have no effect.

2.5.1.46 System Exception Status Register (SYSESR)

The SYSESR register, shown in Figure 2-53 and described in Table 2-65, shows the source for different resets encountered. Previous reset source status bits are not automatically cleared if new resets occur. After reading this register, the software should clear any flags that are set so that the source of future resets can be determined. Any bit in this register can be cleared by writing a 1 to the bit.

Figure 2-53. System Exception Status Register (SYSESR) (offset = E4h)

31	16
Reserved	
R-0	

15	14	13	12	11	10	8
PORST	OSCRST	WDRST	Reserved	DBGRST	Reserved	
R/WC-X	R/WC-X*	R/WC-X*	R-0	R/WC-X*	R-0	

7	6	5	4	3	2	0
ICSTRST	Reserved	CPURST	SWRST	EXTRST	Reserved	
R/WC-X*	R/WC-X*	R/WC-X*	R/WC-X*	R/WC-X*	R-0	

LEGEND: R/W = Read/Write; R = Read only; C = Clear; X = value is unchanged after reset; X* = 0 after PORST but unchanged after other resets; -*n* = value after reset

Table 2-65. System Exception Status Register (SYSESR) Field Descriptions

Bit	Field	Value	Description
31-16	Reserved	0	Reads return 0. Writes have no effect.
15	PORST		Power-on reset. This bit is set when a power-on reset occurs, either internally asserted by the VMON or externally asserted by the nPORRST pin.
		0	No power-on reset has occurred since this bit was last cleared.
		1	A reset was caused by a power-on reset. (This bit should be cleared after being read so that subsequent resets can be properly identified as not being power-on resets.)
14	OSCRST		Reset caused by an oscillator failure or PLL cycle slip. This bit is set when a reset is caused by an oscillator failure or PLL slip. Write 1 will clear this bit. Write 0 has no effect. **Note: The action taken when an oscillator failure or PLL slip is detected must configured in the PLLCTL1 register.**
		0	No reset has occurred due to an oscillator failure or a PLL cycle slip.
		1	A reset was caused by an oscillator failure or a PLL cycle slip.
13	WDRST		Watchdog reset flag. This bit is set when the last reset was caused by the digital watchdog (DWD). Write 1 will clear this bit. Write 0 has no effect.
		0	No reset has occurred because of the DWD.
		1	A reset was caused by the DWD.
12	Reserved	0	Reads return 0. Writes have no effect.
11	DBGRST		Debug reset flag. This bit is set when the last reset was caused by the debugger reset request. Write 1 will clear this bit. Write 0 has no effect.
		0	No reset has occurred because of the debugger.
		1	A reset was caused by the debugger.
10-8	Reserved	0	Reads return 0. Writes have no effect.
7	ICSTRST		Interconnect reset flag. This bit is set when the last CPU reset was caused by the entering and exiting of interconnect self-test check. While the interconnect is under self-test check, the CPU is also held in reset until the interconnect self-test is complete.
		0	No CPUx reset has occurred because of an interconnect self-test check.
		1	A reset has occurred to the CPUx because of the interconnect self-test check.
6	Reserved	0	Reads return 0. Writes have no effect.

Table 2-65. System Exception Status Register (SYSESR) Field Descriptions (continued)

Bit	Field	Value	Description
5	CPURST		CPU reset flag. This bit is set when the CPU is reset. Write 1 will clear this bit. Write 0 has no effect. **Note: A CPU reset can be initiated by the CPU self-test controller (LBIST) or by toggling the CPU RESET bit field in CPURSTCR register.**
		0	No CPU reset has occurred.
		1	A CPU reset occurred.
4	SWRST		Software reset flag. This bit is set when a software system reset has occurred. Write 1 will clear this bit. Write 0 has no effect. **Note: A software system reset can be initiated by writing to the RESET bits in the SYSECR register.**
		0	No software reset has occurred.
		1	A software reset occurred.
3	EXTRST		External reset flag. This bit is set when a reset is caused by the external reset pin nRST or by any reset that also asserts the nRST pin (PORST, OSCRST, WDRST, WD2RST, and SWRST).
		0	The external reset pin has not asserted a reset.
		1	A reset has been caused by the external reset pin.
2-0	Reserved	0	Reads return 0. Writes have no effect.

2.5.1.47 System Test Abort Status Register (SYSTASR)

This register is shown in Figure 2-54 and described in Table 2-66.

Figure 2-54. System Test Abort Status Register (SYSTASR) (offset = E8h)

31	16
Reserved	
R-0	

15	5	4	0
Reserved		EFUSE_Abort	
R-0		R/WPC-X/0	

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privileged mode only; C = Clear; -X = value is unchanged after reset; -*n* = value after reset

Table 2-66. System Test Abort Status Register (SYSTASR) Field Descriptions

Bit	Field	Value	Description
31-5	Reserved	0	Reads return 0. Writes have no effect.
4-0	EFUSE_Abort		Test Abort status flag. These bits are set when test abort occurred:
		0	*Read:* The last operation (if any) completed successfully. This is also the value that the error/status register is set to after reset.
		1h	*Read:* Controller times out because there is no last row sent from the FuseROM.
		2h	*Read:* The autoload machine was started, either through the SYS_INITZ signal from the system or the JTAG data register. In either case, the autoload machine did not find enough FuseROM data to fill the scan chain.
		3h	*Read:* The autoload machine was started, either through the SYS_INITZ signal from the system or the JTAG data register. In either case, the autoload machine starts the scan chain with a signature it expects to see after the scan chain is full. The autoload machine was able to fill the scan chain, but the wrong signature was returned.
		4h	*Read:* The autoload machine was started, either through the SYS_INITZ signal from the system or the JTAG data register. In either case, the autoload machine was not able or not allowed to complete its operation.
		Others	*Read:* Reserved.
		1Fh	*Write:* These bits are cleared to 0.

2.5.1.48 Global Status Register (GLBSTAT)

The GLBSTAT register, shown in Figure 2-55 and described in Table 2-67, indicates the FMzPLL (PLL1) slip status and the oscillator fail status.

NOTE: PLL and OSC fail behavior

The device behavior after a PLL slip or an oscillator failure is configured in the PLLCTL1 register.

Figure 2-55. Global Status Register (GLBSTAT) (offset = ECh)

31	16
Reserved	
R-0	

15	10	9	8	7	1	0
Reserved		FBSLIP	RFSLIP	Reserved		OSCFAIL
R-0		R/W1C-n	R/W1C-n	R-0		R/W1C-n

LEGEND: R/W = Read/Write; R = Read only; W1C = Write 1 to Clear; -n = value after reset

Table 2-67. Global Status Register (GLBSTAT) Field Descriptions

Bit	Field	Value	Description
31-10	Reserved	0	Reads return 0. Writes have no effect.
9	FBSLIP	0	PLL over cycle slip detection. (cleared by nPORRST, maintains its previous value for all other resets). *Read:* No PLL over cycle slip has been detected. *Write:* The bit is unchanged.
		1	*Read:* A PLL over cycle slip has been detected. *Write:* The bit is cleared to 0.
8	RFSLIP	0	PLL under cycle slip detection. (cleared by nPORRST, maintains its previous value for all other resets). *Read:* No PLL under cycle slip has been detected. *Write:* The bit is unchanged.
		1	*Read:* A PLL under cycle slip has been detected. *Write:* The bit is cleared to 0.
7-1	Reserved	0	Reads return 0. Writes have no effect.
0	OSCFAIL	0	Oscillator fail flag bit. (cleared by nPORRST, maintains its previous value for all other resets). *Read:* No oscillator failure has been detected. *Write:* The bit is unchanged.
		1	*Read:* An oscillator failure has been detected. *Write:* The bit is cleared to 0.

2.5.1.49 Device Identification Register (DEVID)

The DEVID is a read-only register. It contains device-specific information that is hard-coded during device manufacture. For the initial silicon version, the device identification code value is 8044 AD05h. This register is shown in Figure 2-56 and described in Table 2-68.

Figure 2-56. Device Identification Register (DEVID) (offset = F0h)

31	30	17	16
CP15	UNIQUE ID		TECH
R-K	R-K		R-K

15	13	12	11	10	9	8
TECH		I/O VOLTAGE	PERIPHERAL PARITY	FLASH ECC		RAM ECC
R-K		R-K	R-K	R-K		R-K

7	3	2	0
VERSION		PLATFORM ID	
R-K		R-K	

LEGEND: R = Read only; K =Constant value; -*n* = value after reset

Table 2-68. Device Identification Register (DEVID) Field Descriptions

Bit	Field	Value	Description
31	CP15		CP15 CPU. This bit indicates whether the CPU has a coprocessor 15 (CP15).
		0	The CPU has no CP15 present.
		1	The CPU has a CP15 present. The CPU ID can be read using the CP15 C0,C0,0 register.
30-17	UNIQUE ID	0-3FFFh	Device ID. The device ID is unique by device configuration.
16-13	TECH		These bits define the process technology by which the device was manufactured.
		0	Device manufactured in the C05 process technology.
		1h	Device manufactured in the F05 process technology.
		2h	Device manufactured in the C035 process technology.
		3h	Device manufactured in the F035 process technology.
		4h	Device manufactured in the C021 process technology.
		5h	Device manufactured in the F021 process technology.
		6h-7h	Reserved
12	I/O VOLTAGE		Input/output voltage. This bit defines the I/O voltage of the device.
		0	The I/O voltage is 3.3 V.
		1	The I/O voltage is 5 V.
11	PERIPHERAL PARITY		Peripheral parity. This bit indicates whether or not peripheral memory parity is present.
		0	The peripheral memories have no parity.
		1	The peripheral memories have parity.
10-9	FLASH ECC		These bits indicate which parity is present for the program memory.
		0	No memory protection is present.
		1h	The program memory (Flash) has single-bit parity.
		2h	The program memory (Flash) has ECC.
		3h	This combination is reserved.
8	RAM ECC		RAM ECC. This bit indicates whether or not RAM memory ECC is present.
		0	The RAM memories do not have ECC.
		1	The RAM memories have ECC.
7-3	VERSION	0-1Fh	Version. These bits provide the revision of the device.
2-0	PLATFORM ID	5h	The device is part of the TMS570Px family. The TMS570Px ID is always 5h.

2.5.1.50 Software Interrupt Vector Register (SSIVEC)

The SSIVEC register, shown in Figure 2-57 and described in Table 2-69, contains information about software interrupts.

Figure 2-57. Software Interrupt Vector Register (SSIVEC) (offset = F4h)

31	16
Reserved	
R-0	

15	8	7	0
SSIDATA		SSIVECT	
R-0		R-0	

LEGEND: R = Read only; -*n* = value after reset

Table 2-69. Software Interrupt Vector Register (SSIVEC) Field Descriptions

Bit	Field	Value	Description
31-16	Reserved	0	Reads return 0. Writes have no effect.
15-8	SSIDATA	0-FFh	System software interrupt data key. These bits contain the data key value of the source for the system software interrupt, which is indicated by the vector in the SSIVEC[7-0] field.
7-0	SSIVECT		These bits contain the source for the system software interrupt. **Note: A read from the SSIVECT bits clears the corresponding SSI_FLAG[4-1] bit in the SSIF register, corresponding to the source vector of the system software interrupt.** **Note: The SSIR[4-1] interrupt has the following priority order:** SSIR1 has the highest priority. SSIR4 has the lowest priority.
		0	No software interrupt is pending.
		1h	A software interrupt has been generated by writing the correct key value to The SSIR1 register.
		2h	A software interrupt has been generated by writing the correct key value to The SSIR2 register.
		3h	A software interrupt has been generated by writing the correct key value to The SSIR3 register.
		4h	A software interrupt has been generated by writing the correct key value to The SSIR4 register.
		5h-FFh	Reserved

2.5.1.51 System Software Interrupt Flag Register (SSIF)

The SSIF register, shown in Figure 2-58 and described in Table 2-70, contains software interrupt flag status information.

Figure 2-58. System Software Interrupt Flag Register (SSIF) (offset = F8h)

31	16
Reserved	
R-0	

15	8
Reserved	
R-0	

7	4	3	2	1	0
Reserved		SSI_FLAG4	SSI_FLAG3	SSI_FLAG2	SSI_FLAG1
R-0		R/WC-0	R/WC-0	R/WC-0	R/WC-0

LEGEND: R/W = Read/Write; R = Read only; C = Clear; -*n* = value after reset

Table 2-70. System Software Interrupt Flag Register (SSIF) Field Descriptions

Bit	Field	Value	Description
31-4	Reserved	0	Reads return 0. Writes have no effect.
3-0	SSI_FLAG[4-1]		System software interrupt flag[4-1]. This flag is set when the correct SSKEY is written to the SSIR register[4-1]. **Note: A read from the SSIVEC register clears the corresponding SSI_FLAG[4-1] bit in the SSIF, corresponding to the source vector of the system software interrupt.**
		0	*Read:* No IRQ/FIQ interrupt was generated since the bit was last cleared. *Write:* The bit is unchanged.
		1	*Read:* An IRQ/FIQ interrupt was generated. *Write:* The bit is cleared to 0.

2.5.2 Secondary System Control Registers (SYS2)

This section describes the secondary frame of system registers. The start address of the secondary system module frame is FFFF E100h. The registers support 8-, 16-, and 32-bit writes. The offset is relative to the system module frame start address.

Table 2-71 contains a list of the secondary system control registers.

NOTE: All additional registers in the secondary system frame are reserved.

Table 2-71. Secondary System Control Registers

Offset	Acronym	Register Description	Section
00h	PLLCTL3	PLL Control Register 3	Section 2.5.2.1
08h	STCLKDIV	CPU Logic BIST Clock Divider	Section 2.5.2.2
24h	ECPCNTL	ECP Control Register. The ECPCNTL register has the mirrored location at this address.	Section 2.5.1.43
28h	ECPCNTL1	ECP Control Register 1.	Section 2.5.2.3
3Ch	CLK2CNTRL	Clock 2 Control Register	Section 2.5.2.4
40h	VCLKACON1	Peripheral Asynchronous Clock Configuration 1 Register	Section 2.5.2.5
54h	HCLKCNTL	HCLK Control Register	Section 2.5.2.6
70h	CLKSLIP	Clock Slip Control Register	Section 2.5.2.7
78h	IP1ECCERREN	IP ECC Error Enable Register	Section 2.5.2.8
ECh	EFC_CTLREG	EFUSE Controller Control Register	Section 2.5.2.9
F0h	DIEIDL_REG0	Die Identification Register Lower Word	Section 2.5.2.10
F4h	DIEIDH_REG1	Die Identification Register Upper Word	Section 2.5.2.11
F8h	DIEIDL_REG2	Die Identification Register Lower Word	Section 2.5.2.12
FCh	DIEIDH_REG3	Die Identification Register Upper Word	Section 2.5.2.13

2.5.2.1 PLL Control Register 3 (PLLCTL3)

The PLLCTL3 register is shown in Figure 2-59 and described in Table 2-72; controls the settings of PLL2 (Clock Source 6 - FPLL).

Figure 2-59. PLL Control Register 3 (PLLCTL3) (offset = 00h)

31-29	28-24	23-22	21-16
ODPLL2	PLLDIV2	Reserved	REFCLKDIV2
R/WP-0	R/WP-4h	R-0	R/WP-0

15-0
PLLMUL2
R/WP-1300h

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privileged mode only; -*n* = value after reset

Table 2-72. PLL Control Register 3 (PLLCTL3) Field Descriptions

Bit	Field	Value	Description
31-29	ODPLL2		Internal PLL Output Divider OD2 = ODPLL2 + 1, ranges from 1 to 8. $f_{post\_ODCLK2} = f_{output\_CLK2\ /\ OD2}$ **Note: PLL output clock is gated off if ODPLL2 is changed while the PLL#2 is active.**
		0	$f_{post\_ODCLK2} = f_{output\_CLK2\ /\ 1}$
		1h	$f_{post\_ODCLK2} = f_{output\_CLK2\ /\ 2}$
		:	:
		7h	$f_{post\_ODCLK2} = f_{output\_CLK2\ /\ 8}$
28-24	PLLDIV2		PLL2 Output Clock Divider R2 = PLLDIV2 + 1, ranges from 1 to 32. $f_{PLL2\ CLK} = f_{post\_ODCLK2}$ / R2
		0	$f_{PLL2\ CLK} = f_{post\_ODCLK2}$ / 1
		1h	$f_{PLL2\ CLK} = f_{post\_ODCLK2}$ / 2
		:	:
		1Fh	$f_{PLL2\ CLK} = f_{post\_ODCLK2}$ / 32
23-22	Reserved	0	Value has no effect on PLL operation.
21-16	REFCLKDIV2		Reference Clock Divider NR2 = REFCLKDIV2 + 1, ranges from 1 to 64. $f_{INTCLK2} = f_{OSCIN}$ / NR2 **Note: This value should not be changed while the PLL2 is active.**
		0	$f_{INTCLK2} = f_{OSCIN}$ / 1
		1h	$f_{INTCLK2} = f_{OSCIN}$ / 2
		:	:
		3Fh	$f_{INTCLK2} = f_{OSCIN}$ / 64
15-0	PLLMUL2		PLL2 Multiplication Factor NF2 = (PLLMUL2 / 256) + 1, valid multiplication factors are from 1 to 256. $f_{VCOCLK2} = f_{INTCLK2}$ x NF2 User and privileged mode (read): Privileged mode (write):
		100h	$f_{VCOCLK2} = f_{INTCLK2}$ x 1
		:	:
		5B00h	$f_{VCOCLK2} = f_{INTCLK2}$ x 92
		5C00h	$f_{VCOCLK2} = f_{INTCLK2}$ x 93
		:	:
		FF00h	$f_{VCOCLK2} = f_{INTCLK2}$ x 256

2.5.2.2 CPU Logic Bist Clock Divider (STCLKDIV)

This register is shown in Figure 2-60 and described in Table 2-73.

Figure 2-60. CPU Logic BIST Clock Prescaler (STCLKDIV) (offset = 08h)

31 27	26 24	23 16
Reserved	CLKDIV	Reserved
R-0	R/WP-0	R-0

15 0
Reserved
R-0

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privileged mode only; -*n* = value after reset

Table 2-73. CPU Logic BIST Clock Prescaler (STCLKDIV) Field Descriptions

Bit	Field	Value	Description
31-27	Reserved	0	Reads return 0. Writes have no effect.
26-24	CLKDIV	0	Clock divider/prescaler for CPU clock during logic BIST
23-0	Reserved	0	Reads return 0. Writes have no effect.

2.5.2.3 ECP Control Register 1 (ECPCNTL1)

The ECP register, shown in Figure 2-61 and described in Table 2-74, configures the ECLK2 pin in functional mode.

NOTE: ECLK2 Functional mode configuration.

The ECLK2 pin must be placed into Functional mode by setting the ECPCLKFUN bit to 1 in the SYSPC1 register before a clock source will be visible on the ECLK pin.

Figure 2-61. ECP Control Register 1 (ECPCNTL1) (offset = 28h)

31 28	27 25	24	23	22 16
ECP_KEY	Reserved	ECPSSEL	ECPCOS	Reserved
R/WP-5h	R-0	R/W-0	R/W-0	R-0

15 0
ECPDIV
R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 2-74. ECP Control Register 1 (ECPCNTL1) Field Descriptions

Bit	Field	Value	Description
31-28	ECP_KEY		Enable ECP clock logic for ECLK2.
		Ah	Clock functionality of ECP clock is enabled.
		Others	Clock functionality of ECP clock is disabled.
27-25	Reserved	0	Reads return 0. Writes have no effect.
24	ECPSSEL		This bit allows the selection between VCLK and OSCIN as the clock source for ECLK2.
		0	VCLK is selected as the ECP clock source.
		1	OSCIN is selected as the ECP clock source.
23	ECPCOS		ECP continue on suspend. **Note: Suspend mode is entered while performing certain JTAG debugging operations.**
		0	ECLK output is disabled in suspend mode. ECLK output will be shut off and will not be seen on the I/O pin of the device.
		1	ECLK output is not disabled in suspend mode. ECLK output will not be shut off and will be seen on the I/O pin of the device.
22-16	Reserved	0	Reads return 0. Writes have no effect.
15-0	ECPDIV	0-FFFFh	ECP divider value. The value of ECPDIV bits determine the external clock (ECP clock) frequency as a ratio of VBUS clock or OSCIN as shown in the formula: $ECLK = \frac{VCLKorOSCIN}{(ECPDIV + 1)}$

2.5.2.4 Clock 2 Control Register (CLK2CNTRL)

This register is shown in Figure 2-62 and described in Table 2-75.

Figure 2-62. Clock 2 Control Register (CLK2CNTRL) (offset = 3Ch)

31 16
Reserved
R-0

15 12	11 8	7 4	3 0
Reserved	Reserved	Reserved	VCLK3R
R-0	R/WP-1h	R-0	R/WP-1h

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privileged mode only; -*n* = value after reset

Table 2-75. Clock 2 Control Register (CLK2CNTRL) Field Descriptions

Bit	Field	Value	Description
31-12	Reserved	0	Reads return 0. Writes have no effect.
11-8	Reserved		Reads return value and writes allowed in privilege mode.
7-4	Reserved	0	Reads return 0. Writes have no effect.
3-0	VCLK3R		VBUS clock3 ratio.
		0	The ratio is HCLK divide by 1.
		:	:
		Fh	The ratio is HCLK divided by 16.

2.5.2.5 Peripheral Asynchronous Clock Configuration 1 Register (VCLKACON1)

This register is shown in Figure 2-63 and described in Table 2-76.

Figure 2-63. Peripheral Asynchronous Clock Configuration 1 Register (VCLKACON1) [offset = 40h]

31 27	26 24
Reserved	VCLKA4R
R-0	R/WP-1h

23 21	20	19 16
Reserved	VCLKA4_DIV_ CDDIS	VCLKA4S
R-0	R/WP-0	R/WP-9h

15 11	10 8	7 5	4 0
Reserved	Reserved	Reserved	Reserved
R-0	R/WP-1h	R-0	R/WP-9h

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privileged mode only; -*n* = value after reset

Table 2-76. Peripheral Asynchronous Clock Configuration 1 Register (VCLKACON1) Field Descriptions

Bit	Field	Value	Description
31-27	Reserved	0	Reads return 0. Writes have no effect.
26-24	VCLKA4R		Clock divider for the VCLKA4 source. Output will be present on VCLKA4_DIVR.
			VCLKA4 domain will be enabled by writing to the CDDIS register and VCLKA4_DIV_CDDIS bit. It can inferred that VCLKA4_DIV clock is disabled when VCLKA4 clock is disabled.
		0	The ratio is VCLKA4 divided by 1.
		:	:
		7h	The ratio is VCLKA4 divided by 8.
23-21	Reserved	0	Reads return 0. Writes have no effect.
20	VCLKA4_DIV_CDDIS		Disable the VCLKA4 divider output.
			VCLKA4 domain will be enabled by writing to the CDDIS register.
		0	Enable the prescaled VCLKA4 clock on VCLKA4_DIVR.
		1	Disable the prescaled VCLKA4 clock on VCLKA4_DIVR.
19-16	VCLKA4S		Peripheral asynchronous clock4 source.
		0	Clock source0 is the source for peripheral asynchronous clock4.
		1h	Clock source1 is the source for peripheral asynchronous clock4.
		2h	Clock source2 is the source for peripheral asynchronous clock4.
		3h	Clock source3 is the source for peripheral asynchronous clock4.
		4h	Clock source4 is the source for peripheral asynchronous clock4.
		5h	Clock source5 is the source for peripheral asynchronous clock4.
		6h	Clock source6 is the source for peripheral asynchronous clock4.
		7h	Clock source7 is the source for peripheral asynchronous clock4.
		8h-Fh	VCLK or a divided VCLK is the source for peripheral asynchronous clock4. See the device-specific data manual for details.
15-0	Reserved	109h	Reserved

NOTE: Non-implemented clock sources should not be enabled or used. A list of the available clock sources is shown in the Table 2-29.

2.5.2.6 HCLK Control Register (HCLKCNTL)

This register is shown in Figure 2-64 and described in Table 2-77.

Figure 2-64. HCLK Control Register (HCLKCNTL) (offset = 54h)

31	16
Reserved	
R-0	

15	2	1	0
Reserved		HCLKR	
R-0		R/WP-0	

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privileged mode only; -*n* = value after reset

Table 2-77. HCLK Control Register (HCLKCNTL) Field Descriptions

Bit	Field	Value	Description
31-2	Reserved	0	Reads return 0. Writes have no effect.
1-0	HCLKR		HCLK divider value. The value of HCLKR bits determine the HCLK frequency as a ratio of GCLK1.
		0	HCLK is equal to GCLK1 divide by 1.
		1h	HCLK is equal to GCLK1 divide by 2.
		2h	HCLK is equal to GCLK1 divide by 3.
		3h	HCLK is equal to GCLK1 divide by 4.

2.5.2.7 Clock Slip Control Register (CLKSLIP)

This register is shown in Figure 2-65 and described in Table 2-78. For information on filtering the FBSLIP, see Section 2.5.1.34.

Figure 2-65. Clock Slip Control Register (CLKSLIP) (offset = 70h)

31 16
Reserved
R-0

15 14	13 8	7 4	3 0
Reserved	PLL1_RFSLIP_FILTER_COUNT	Reserved	PLL1_RFSLIP_FILTER_KEY
R-0	R/WP-0	R-0	R/WP-5h

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privileged mode only; -*n* = value after reset

Table 2-78. Clock Slip Control Register (CLKSLIP) Field Descriptions

Bit	Field	Value	Description
31-14	Reserved	0	Reads return 0. Writes have no effect.
13-8	PLL1_RFSLIP_FILTER_COUNT		PLL RFSLIP down counter programmed value. Count is on 10M clock.
			On reset, counter value is 0. Counter must be programmed to a non-zero value and enabled for the filtering to be enabled.
		0	Filtering is disabled.
		1h	Filtering is enabled. Every slip is recognized.
		2h	Filtering is enabled. The slip must be at least 2 HF LPO cycles wide in order to be recognized as a slip.
		:	:
		3Fh	Filtering is enabled. The RFSLIP must be at least 63 HF LPO cycles wide in order to be recognized as a slip.
7-4	Reserved	0	Reads return 0. Writes have no effect.
3-0	PLL1_RFSLIP_FILTER_KEY		Enable the PLL RFSLIP filtering.
		5h	On reset, the PLL RFSLIP filter is disabled and the PLL RFSLIP passes through.
		Fh	This is an unsupported value. You should avoid writing this value to this bit field.
		Others	PLL RFSLIP filtering is enabled. Recommended to program Ah in this bit field. Enabling of the PLL RFSLIP occurs when the KEY is programmed and a non-zero value is present in the COUNT field.

2.5.2.8 IP ECC Error Enable Register (IP1ECCERREN)

This register is shown in Figure 2-66 and described in Table 2-79.

Figure 2-66. IP ECC Error Enable Register (IP1ECCERREN) (offset = 78h)

31 28	27 24	23 20	19 16
Reserved	Reserved	Reserved	Reserved
R-0	R/WP-5h		R/WP-5h

15 12	11 8	7 4	3 0
Reserved	IP2_ECC_KEY	Reserved	IP1_ECC_KEY
R-0	R/WP-5h	R-0	R/WP-5h

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privileged mode only; -*n* = value after reset

Table 2-79. Clock Slip Register (CLKSLIP) Field Descriptions

Bit	Field	Value	Description
31-28	Reserved	0	Reads return 0. Writes have no effect.
27-24	Reserved	0-Fh	Reads return 0 or 1 and depends on what is written in privileged mode. The functionality of this bit is unavailable in this device.
23-20	Reserved	0	Reads return 0. Writes have no effect.
19-16	Reserved	0-Fh	Reads return 0 or 1 and depends on what is written in privileged mode. The functionality of this bit is unavailable in this device.
15-12	Reserved	0	Reads return 0. Writes have no effect.
11-8	IP2_ECC_KEY		ECC Error Enable Key for PS_SCR_M master. There is an ECC Evaluation block inside the CPU Interconnect Subsystem responsible for ECC correction and detection on the data path for transactions initiated by the PS_SCR_M master. If an ECC error (either single-bit or double-bit error) is detected, then the corresponding error signal is asserted if ECC enable key written to IP2_ECC_KEY is Ah.
		Others	Disable ECC error generation for ECC errors detected on PS_SCR_M master by the CPU Interconnect Subsystem.
		Ah	Enable ECC error generation for ECC errors detected on PS_SCR_M master by the CPU Interconnect Subsystem.
7-4	Reserved	0	Reads return 0. Writes have no effect.
3-0	IP1_ECC_KEY		ECC Error Enable Key for DMA Port A master. There is an ECC Evaluation block inside the CPU Interconnect Subsystem responsible for ECC correction and detection on the data path for transactions initiated by the DMA Port A master. If an ECC error (either single-bit or double-bit error) is detected, then the corresponding error signal is asserted if ECC enable key written to IP1_ECC_KEY is Ah.
		Others	Disable ECC error generation for ECC errors detected on DMA Port A master by the CPU Interconnect Subsystem.
		Ah	Enable ECC error generation for ECC errors detected on DMA Port A master by the CPU Interconnect Subsystem.

2.5.2.9 EFUSE Controller Control Register (EFC_CTLREG)

This register is shown in Figure 2-67 and described in Table 2-80.

Figure 2-67. EFUSE Controller Control Register (EFC_CTLREG) (offset = ECh)

31–16
Reserved
R-0

15–4	3–0
Reserved	EFC_INSTR_WEN
R-0	R/WP-5h

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privileged mode only; -*n* = value after reset

Table 2-80. EFUSE Controller Control Register (EFC_CTLREG) Field Descriptions

Bit	Field	Value	Description
31-4	Reserved	0	Reads return 0. Writes have no effect.
3-0	EFC_INSTR_WEN		Enable user write of 4 EFUSE controller instructions. SYS module generates the enable signal that will be tied to OCP_FROM_WRITE_DISABLE on efuse controller port.
		Ah	Writing of instructions (Program, ProgramCRA, RunAutoload, and LoadFuseScanchain) to EFC is allowed.
		Others	Writing of instructions (Program, ProgramCRA, RunAutoload, and LoadFuseScanchain) in EFC registers is blocked.

2.5.2.10 Die Identification Register Lower Word (DIEIDL_REG0)

The DIEIDL_REG0 register is a duplicate of the DIEIDL register, see Section 2.5.1.28. The DIEIDL_REG0 register, shown in Figure 2-68 and described in Table 2-81, contains information about the die wafer number, and X, Y wafer coordinates.

Figure 2-68. Die Identification Register, Lower Word (DIEIDL_REG0) [offset = F0h]

31–24	23–16
WAFER #	Y WAFER COORDINATE
R-D	R-D

15–12	11–0
Y WAFER COORDINATE	X WAFER COORDINATE
R-D	R-D

LEGEND: R = Read only; D = value is device specific; -*n* = value after reset

Table 2-81. Die Identification Register, Lower Word (DIEIDL_REG0) Field Descriptions

Bit	Field	Description
31-24	WAFER #	These read-only bits contain the wafer number of the device.
23-12	Y WAFER COORDINATE	These read-only bits contain the Y wafer coordinate of the device.
11-0	X WAFER COORDINATE	These read-only bits contain the X wafer coordinate of the device.

NOTE: Die Identification Information

The die identification information will vary from unit to unit. This information is programmed by TI as part of the initial device test procedure.

2.5.2.11 Die Identification Register Upper Word (DIEIDH_REG1)

The DIEIDH_REG1 register is a duplicate of the DIEIDH register, see Section 2.5.1.29. The DIEIDH_REG1 register, shown in Figure 2-69 and described in Table 2-82, contains information about the die lot number.

Figure 2-69. Die Identification Register, Upper Word (DIEIDH_REG1) [offset = F4h]

31 24	23 16
Reserved	LOT #
R-0	R-D

15 0
LOT #
R-D

LEGEND: R = Read only; D = value is device specific; -*n* = value after reset

Table 2-82. Die Identification Register, Upper Word (DIEIDH_REG1) Field Descriptions

Bit	Field	Description
31-24	Reserved	Reserved for TI use. Writes have no effect.
23-0	LOT #	This read-only register contains the device lot number.

NOTE: Die Identification Information

The die identification information will vary from unit to unit. This information is programmed by TI as part of the initial device test procedure.

2.5.2.12 Die Identification Register Lower Word (DIEIDL_REG2)

This register is shown in Figure 2-70 and described in Table 2-83.

Figure 2-70. Die Identification Register, Lower Word (DIEIDL_REG2) [offset = F8h]

31 0
DIEIDL2
R-X

LEGEND: R = Read only; X = value is unchanged after reset; -*n* = value after reset

Table 2-83. Die Identification Register, Lower Word (DIEIDL_REG2) Field Descriptions

Bit	Field	Value	Description
31-0	DIEIDL2(95-64)	0-FFFF FFFFh	This read-only register contains the lower word (95:64) of the die ID information. The contents of this register is reserved.

2.5.2.13 Die Identification Register Upper Word (DIEIDH_REG3)

This register is shown in Figure 2-71 and described in Table 2-84.

Figure 2-71. Die Identification Register, Upper Word (DIEIDH_REG3) [offset = FCh]

<table>
<tr><td>31</td><td>0</td></tr>
<tr><td colspan="2">DIEIDH2</td></tr>
<tr><td colspan="2">R-X</td></tr>
</table>

LEGEND: R = Read only; X = value is unchanged after reset ; -*n* = value after reset

Table 2-84. Die Identification Register, Upper Word (DIEIDH_REG3) Field Descriptions

Bit	Field	Value	Description
31-0	DIEIDH2(127-96)	0-FFFF FFFFh	This read-only register contains the upper word (127:97) of the die ID information. The contents of this register is reserved.

2.5.3 Peripheral Central Resource (PCR) Control Registers

This section describes the Peripheral Central Resource (PCR) control registers. The are three PCRx in this microcontroller. The start address of the PCR1 register frame is FFFF 1000h. The start address of the PCR2 register frame is FCFF 1000h. The start address of the PCR3 register frame is FFF7 8000h. Table 2-85 lists the registers in the PCR, which are used to configure the following main functionalities:

- Protection control to the peripherals in PCS (Peripheral Memory) and PS (Peripheral) regions.
- Powerdown control to the peripherals in PCS (Peripheral Memory) and PS (Peripheral) regions.
- Powerdown control to the CoreSight debug peripherals in debug frame region from FFA0 0000h to FFAF FFFFh.
- Master-ID Filtering control to the peripherals in PS (Peripheral), PPS (Privileged Peripheral) , PPSE (Privileged Peripheral Extended) regions.
- Master-ID Filtering control to the peripheral memories in PCS (Peripheral Memory), and PPCS (Privileged Peripheral Memory) regions.

The following sections provide detailed information about these registers. Not all chip selects exist on this device.

Table 2-85. Peripheral Central Resource Control Registers

Offset	Acronym	Register Description	Section
00h	PMPROTSET0	Peripheral Memory Protection Set Register 0	Section 2.5.3.1
04h	PMPROTSET1	Peripheral Memory Protection Set Register 1	Section 2.5.3.2
10h	PMPROTCLR0	Peripheral Memory Protection Clear Register 0	Section 2.5.3.3
14h	PMPROTCLR1	Peripheral Memory Protection Clear Register 1	Section 2.5.3.4
20h	PPROTSET0	Peripheral Protection Set Register 0	Section 2.5.3.5
24h	PPROTSET1	Peripheral Protection Set Register 1	Section 2.5.3.6
28h	PPROTSET2	Peripheral Protection Set Register 2	Section 2.5.3.7
2Ch	PPROTSET3	Peripheral Protection Set Register 3	Section 2.5.3.8
40h	PPROTCLR0	Peripheral Protection Clear Register 0	Section 2.5.3.9
44h	PPROTCLR1	Peripheral Protection Clear Register 1	Section 2.5.3.10
48h	PPROTCLR2	Peripheral Protection Clear Register 2	Section 2.5.3.11
4Ch	PPROTCLR3	Peripheral Protection Clear Register 3	Section 2.5.3.12
60h	PCSPWRDWNSET0	Peripheral Memory Power-Down Set Register 0	Section 2.5.3.13
64h	PCSPWRDWNSET1	Peripheral Memory Power-Down Set Register 1	Section 2.5.3.14
70h	PCSPWRDWNCLR0	Peripheral Memory Power-Down Clear Register 0	Section 2.5.3.15
74h	PCSPWRDWNCLR1	Peripheral Memory Power-Down Clear Register 1	Section 2.5.3.16
80h	PSPWRDWNSET0	Peripheral Power-Down Set Register 0	Section 2.5.3.17
84h	PSPWRDWNSET1	Peripheral Power-Down Set Register 1	Section 2.5.3.18
88h	PSPWRDWNSET2	Peripheral Power-Down Set Register 2	Section 2.5.3.19
8Ch	PSPWRDWNSET3	Peripheral Power-Down Set Register 3	Section 2.5.3.20
A0h	PSPWRDWNCLR0	Peripheral Power-Down Clear Register 0	Section 2.5.3.21
A4h	PSPWRDWNCLR1	Peripheral Power-Down Clear Register 1	Section 2.5.3.22
A8h	PSPWRDWNCLR2	Peripheral Power-Down Clear Register 2	Section 2.5.3.23
ACh	PSPWRDWNCLR3	Peripheral Power-Down Clear Register 3	Section 2.5.3.24
C0h	PDPWRDWNSET	Debug Frame Power-Down Set Register	Section 2.5.3.25
C4h	PDPWRDWNCLR	Debug Frame Power-Down Clear Register	Section 2.5.3.26
200h	MSTIDWRENA	MasterID Protection Write Enable Register	Section 2.5.3.27
204h	MSTIDENA	MasterID Protection Enable Register	Section 2.5.3.28
208h	MSTIDDIAGCTRL	MasterID Diagnostic Control Register	Section 2.5.3.29
300h	PS0MSTID_L	Peripheral Frame 0 Master-ID Protection Register_L	Section 2.5.3.30
304h	PS0MSTID_H	Peripheral Frame 0 Master-ID Protection Register_H	Section 2.5.3.31
308h	PS1MSTID_L	Peripheral Frame 1 Master-ID Protection Register_L	Section 2.5.3.32

Table 2-85. Peripheral Central Resource Control Registers (continued)

Offset	Acronym	Register Description	Section
30Ch	PS1MSTID_H	Peripheral Frame 1 Master-ID Protection Register_H	Section 2.5.3.32
310h	PS2MSTID_L	Peripheral Frame 2 Master-ID Protection Register_L	Section 2.5.3.32
314h	PS2MSTID_H	Peripheral Frame 2 Master-ID Protection Register_H	Section 2.5.3.32
318h	PS3MSTID_L	Peripheral Frame 3 Master-ID Protection Register_L	Section 2.5.3.32
31Ch	PS3MSTID_H	Peripheral Frame 3 Master-ID Protection Register_H	Section 2.5.3.32
320h	PS4MSTID_L	Peripheral Frame 4 Master-ID Protection Register_L	Section 2.5.3.32
324h	PS4MSTID_H	Peripheral Frame 4 Master-ID Protection Register_H	Section 2.5.3.32
328h	PS5MSTID_L	Peripheral Frame 5 Master-ID Protection Register_L	Section 2.5.3.32
32Ch	PS5MSTID_H	Peripheral Frame 5 Master-ID Protection Register_H	Section 2.5.3.32
330h	PS6MSTID_L	Peripheral Frame 6 Master-ID Protection Register_L	Section 2.5.3.32
334h	PS6MSTID_H	Peripheral Frame 6 Master-ID Protection Register_H	Section 2.5.3.32
338h	PS7MSTID_L	Peripheral Frame 7 Master-ID Protection Register_L	Section 2.5.3.32
33Ch	PS7MSTID_H	Peripheral Frame 7 Master-ID Protection Register_H	Section 2.5.3.32
340h	PS8MSTID_L	Peripheral Frame 8 Master-ID Protection Register_L	Section 2.5.3.32
344h	PS8MSTID_H	Peripheral Frame 8 Master-ID Protection Register_H	Section 2.5.3.32
348h	PS9MSTID_L	Peripheral Frame 9 Master-ID Protection Register_L	Section 2.5.3.32
34Ch	PS9MSTID_H	Peripheral Frame 9 Master-ID Protection Register_H	Section 2.5.3.32
350h	PS10MSTID_L	Peripheral Frame 10 Master-ID Protection Register_L	Section 2.5.3.32
354h	PS10MSTID_H	Peripheral Frame 10 Master-ID Protection Register_H	Section 2.5.3.32
358h	PS11MSTID_L	Peripheral Frame 11 Master-ID Protection Register_L	Section 2.5.3.32
35Ch	PS11MSTID_H	Peripheral Frame 11 Master-ID Protection Register_H	Section 2.5.3.32
360h	PS12MSTID_L	Peripheral Frame 12 Master-ID Protection Register_L	Section 2.5.3.32
364h	PS12MSTID_H	Peripheral Frame 12 Master-ID Protection Register_H	Section 2.5.3.32
368h	PS13MSTID_L	Peripheral Frame 13 Master-ID Protection Register_L	Section 2.5.3.32
36Ch	PS13MSTID_H	Peripheral Frame 13 Master-ID Protection Register_H	Section 2.5.3.32
370h	PS14MSTID_L	Peripheral Frame 14 Master-ID Protection Register_L	Section 2.5.3.32
374h	PS14MSTID_H	Peripheral Frame 14 Master-ID Protection Register_H	Section 2.5.3.32
378h	PS15MSTID_L	Peripheral Frame 15 Master-ID Protection Register_L	Section 2.5.3.32
37Ch	PS15MSTID_H	Peripheral Frame 15 Master-ID Protection Register_H	Section 2.5.3.32
380h	PS16MSTID_L	Peripheral Frame 16 Master-ID Protection Register_L	Section 2.5.3.32
384h	PS16MSTID_H	Peripheral Frame 16 Master-ID Protection Register_H	Section 2.5.3.32
388h	PS17MSTID_L	Peripheral Frame 17 Master-ID Protection Register_L	Section 2.5.3.32
38Ch	PS17MSTID_H	Peripheral Frame 17 Master-ID Protection Register_H	Section 2.5.3.32
390h	PS18MSTID_L	Peripheral Frame 18 Master-ID Protection Register_L	Section 2.5.3.32
394h	PS18MSTID_H	Peripheral Frame 18 Master-ID Protection Register_H	Section 2.5.3.32
398h	PS19MSTID_L	Peripheral Frame 19 Master-ID Protection Register_L	Section 2.5.3.32
39Ch	PS19MSTID_H	Peripheral Frame 19 Master-ID Protection Register_H	Section 2.5.3.32
3A0h	PS20MSTID_L	Peripheral Frame 20 Master-ID Protection Register_L	Section 2.5.3.32
3A4h	PS20MSTID_H	Peripheral Frame 20 Master-ID Protection Register_H	Section 2.5.3.32
3A8h	PS21MSTID_L	Peripheral Frame 21 Master-ID Protection Register_L	Section 2.5.3.32
3ACh	PS21MSTID_H	Peripheral Frame 21 Master-ID Protection Register_H	Section 2.5.3.32
3B0h	PS22MSTID_L	Peripheral Frame 22 Master-ID Protection Register_L	Section 2.5.3.32
3B4h	PS22MSTID_H	Peripheral Frame 22 Master-ID Protection Register_H	Section 2.5.3.32
3B8h	PS23MSTID_L	Peripheral Frame 23 Master-ID Protection Register_L	Section 2.5.3.32
3BCh	PS23MSTID_H	Peripheral Frame 23 Master-ID Protection Register_H	Section 2.5.3.32
3C0h	PS24MSTID_L	Peripheral Frame 24 Master-ID Protection Register_L	Section 2.5.3.32
3C4h	PS24MSTID_H	Peripheral Frame 24 Master-ID Protection Register_H	Section 2.5.3.32

Table 2-85. Peripheral Central Resource Control Registers (continued)

Offset	Acronym	Register Description	Section
3C8h	PS25MSTID_L	Peripheral Frame 25 Master-ID Protection Register_L	Section 2.5.3.32
3CCh	PS25MSTID_H	Peripheral Frame 25 Master-ID Protection Register_H	Section 2.5.3.32
3D0h	PS26MSTID_L	Peripheral Frame 26 Master-ID Protection Register_L	Section 2.5.3.32
3D4h	PS26MSTID_H	Peripheral Frame 26 Master-ID Protection Register_H	Section 2.5.3.32
3D8h	PS27MSTID_L	Peripheral Frame 27 Master-ID Protection Register_L	Section 2.5.3.32
3DCh	PS27MSTID_H	Peripheral Frame 27 Master-ID Protection Register_H	Section 2.5.3.32
3E0h	PS28MSTID_L	Peripheral Frame 28 Master-ID Protection Register_L	Section 2.5.3.32
3E4h	PS28MSTID_H	Peripheral Frame 28 Master-ID Protection Register_H	Section 2.5.3.32
3E8h	PS29MSTID_L	Peripheral Frame 29 Master-ID Protection Register_L	Section 2.5.3.32
3ECh	PS29MSTID_H	Peripheral Frame 29 Master-ID Protection Register_H	Section 2.5.3.32
3E0h	PS30MSTID_L	Peripheral Frame 30 Master-ID Protection Register_L	Section 2.5.3.32
3F4h	PS30MSTID_H	Peripheral Frame 30 Master-ID Protection Register_H	Section 2.5.3.32
3F8h	PS31MSTID_L	Peripheral Frame 31 Master-ID Protection Register_L	Section 2.5.3.32
3FCh	PS31MSTID_H	Peripheral Frame 31 Master-ID Protection Register_H	Section 2.5.3.32
400h	PPS0MSTID_L	Privileged Peripheral Frame 0 Master-ID Protection Register_L	Section 2.5.3.33
404h	PPS0MSTID_H	Privileged Peripheral Frame 0 Master-ID Protection Register_H	Section 2.5.3.34
408h	PPS1MSTID_L	Privileged Peripheral Frame 1 Master-ID Protection Register_L	Section 2.5.3.35
40Ch	PPS1MSTID_H	Privileged Peripheral Frame 1 Master-ID Protection Register_H	Section 2.5.3.35
410h	PPS2MSTID_L	Privileged Peripheral Frame 2 Master-ID Protection Register_L	Section 2.5.3.35
414h	PPS2MSTID_H	Privileged Peripheral Frame 2 Master-ID Protection Register_H	Section 2.5.3.35
418h	PPS3MSTID_L	Privileged Peripheral Frame 3 Master-ID Protection Register_L	Section 2.5.3.35
41Ch	PPS3MSTID_H	Privileged Peripheral Frame 3 Master-ID Protection Register_H	Section 2.5.3.35
420h	PPS4MSTID_L	Privileged Peripheral Frame 4 Master-ID Protection Register_L	Section 2.5.3.35
424h	PPS4MSTID_H	Privileged Peripheral Frame 4 Master-ID Protection Register_H	Section 2.5.3.35
428h	PPS5MSTID_L	Privileged Peripheral Frame 5 Master-ID Protection Register_L	Section 2.5.3.35
42Ch	PPS5MSTID_H	Privileged Peripheral Frame 5 Master-ID Protection Register_H	Section 2.5.3.35
430h	PPS6MSTID_L	Privileged Peripheral Frame 6 Master-ID Protection Register_L	Section 2.5.3.35
434h	PPS6MSTID_H	Privileged Peripheral Frame 6 Master-ID Protection Register_H	Section 2.5.3.35
438h	PPS7MSTID_L	Privileged Peripheral Frame 7 Master-ID Protection Register_L	Section 2.5.3.35
43Ch	PPS7MSTID_H	Privileged Peripheral Frame 7 Master-ID Protection Register_H	Section 2.5.3.35
440h	PPSE0MSTID_L	Privilege Peripheral Extended Frame 0 Master-ID Protection Register_L	Section 2.5.3.36
444h	PPSE0MSTID_H	Privilege Peripheral Extended Frame 0 Master-ID Protection Register_H	Section 2.5.3.37
448h	PPSE1MSTID_L	Privilege Peripheral Extended Frame 1 Master-ID Protection Register_L	Section 2.5.3.38
44Ch	PPSE1MSTID_H	Privilege Peripheral Extended Frame 1 Master-ID Protection Register_H	Section 2.5.3.38
450h	PPSE2MSTID_L	Privilege Peripheral Extended Frame 2 Master-ID Protection Register_L	Section 2.5.3.38
454h	PPSE2MSTID_H	Privilege Peripheral Extended Frame 2 Master-ID Protection Register_H	Section 2.5.3.38
458h	PPSE3MSTID_L	Privilege Peripheral Extended Frame 3 Master-ID Protection Register_L	Section 2.5.3.38
45Ch	PPSE3MSTID_H	Privilege Peripheral Extended Frame 3 Master-ID Protection Register_H	Section 2.5.3.38
460h	PPSE4MSTID_L	Privilege Peripheral Extended Frame 4 Master-ID Protection Register_L	Section 2.5.3.38
464h	PPSE4MSTID_H	Privilege Peripheral Extended Frame 4 Master-ID Protection Register_H	Section 2.5.3.38

Table 2-85. Peripheral Central Resource Control Registers (continued)

Offset	Acronym	Register Description	Section
468h	PPSE5MSTID_L	Privilege Peripheral Extended Frame 5 Master-ID Protection Register_L	Section 2.5.3.38
46Ch	PPSE5MSTID_H	Privilege Peripheral Extended Frame 5 Master-ID Protection Register_H	Section 2.5.3.38
470h	PPSE6MSTID_L	Privilege Peripheral Extended Frame 6 Master-ID Protection Register_L	Section 2.5.3.38
474h	PPSE6MSTID_H	Privilege Peripheral Extended Frame 6 Master-ID Protection Register_H	Section 2.5.3.38
478h	PPSE7MSTID_L	Privilege Peripheral Extended Frame 7 Master-ID Protection Register_L	Section 2.5.3.38
47Ch	PPSE7MSTID_H	Privilege Peripheral Extended Frame 7 Master-ID Protection Register_H	Section 2.5.3.38
480h	PPSE8MSTID_L	Privilege Peripheral Extended Frame 8 Master-ID Protection Register_L	Section 2.5.3.38
484h	PPSE8MSTID_H	Privilege Peripheral Extended Frame 8 Master-ID Protection Register_H	Section 2.5.3.38
488h	PPSE9MSTID_L	Privilege Peripheral Extended Frame 9 Master-ID Protection Register_L	Section 2.5.3.38
48Ch	PPSE9MSTID_H	Privilege Peripheral Extended Frame 9 Master-ID Protection Register_H	Section 2.5.3.38
490h	PPSE10MSTID_L	Privilege Peripheral Extended Frame 10 Master-ID Protection Register_L	Section 2.5.3.38
494h	PPSE10MSTID_H	Privilege Peripheral Extended Frame 10 Master-ID Protection Register_H	Section 2.5.3.38
498h	PPSE11MSTID_L	Privilege Peripheral Extended Frame 11 Master-ID Protection Register_L	Section 2.5.3.38
49Ch	PPSE11MSTID_H	Privilege Peripheral Extended Frame 11 Master-ID Protection Register_H	Section 2.5.3.38
4A0h	PPSE12MSTID_L	Privilege Peripheral Extended Frame 12 Master-ID Protection Register_L	Section 2.5.3.38
4A4h	PPSE12MSTID_H	Privilege Peripheral Extended Frame 12 Master-ID Protection Register_H	Section 2.5.3.38
4A8h	PPSE13MSTID_L	Privilege Peripheral Extended Frame 13 Master-ID Protection Register_L	Section 2.5.3.38
4ACh	PPSE13MSTID_H	Privilege Peripheral Extended Frame 13 Master-ID Protection Register_H	Section 2.5.3.38
4B0h	PPSE14MSTID_L	Privilege Peripheral Extended Frame 14 Master-ID Protection Register_L	Section 2.5.3.38
4B4h	PPSE14MSTID_H	Privilege Peripheral Extended Frame 14 Master-ID Protection Register_H	Section 2.5.3.38
4B8h	PPSE15MSTID_L	Privilege Peripheral Extended Frame 15 Master-ID Protection Register_L	Section 2.5.3.38
4BCh	PPSE15MSTID_H	Privilege Peripheral Extended Frame 15 Master-ID Protection Register_H	Section 2.5.3.38
4C0h	PPSE16MSTID_L	Privilege Peripheral Extended Frame 16 Master-ID Protection Register_L	Section 2.5.3.38
4C4h	PPSE16MSTID_H	Privilege Peripheral Extended Frame 16 Master-ID Protection Register_H	Section 2.5.3.38
4C8h	PPSE17MSTID_L	Privilege Peripheral Extended Frame 17 Master-ID Protection Register_L	Section 2.5.3.38
4CCh	PPSE17MSTID_H	Privilege Peripheral Extended Frame 17 Master-ID Protection Register_H	Section 2.5.3.38
4D0h	PPSE18MSTID_L	Privilege Peripheral Extended Frame 18 Master-ID Protection Register_L	Section 2.5.3.38
4D4h	PPSE18MSTID_H	Privilege Peripheral Extended Frame 18 Master-ID Protection Register_H	Section 2.5.3.38

Table 2-85. Peripheral Central Resource Control Registers (continued)

Offset	Acronym	Register Description	Section
4D8h	PPSE19MSTID_L	Privilege Peripheral Extended Frame 19 Master-ID Protection Register_L	Section 2.5.3.38
4DCh	PPSE19MSTID_H	Privilege Peripheral Extended Frame 19 Master-ID Protection Register_H	Section 2.5.3.38
4E0h	PPSE20MSTID_L	Privilege Peripheral Extended Frame 20 Master-ID Protection Register_L	Section 2.5.3.38
4E4h	PPSE20MSTID_H	Privilege Peripheral Extended Frame 20 Master-ID Protection Register_H	Section 2.5.3.38
4E8h	PPSE21MSTID_L	Privilege Peripheral Extended Frame 21 Master-ID Protection Register_L	Section 2.5.3.38
4ECh	PPSE21MSTID_H	Privilege Peripheral Extended Frame 21 Master-ID Protection Register_H	Section 2.5.3.38
4E0h	PPSE22MSTID_L	Privilege Peripheral Extended Frame 22 Master-ID Protection Register_L	Section 2.5.3.38
4F4h	PPSE22MSTID_H	Privilege Peripheral Extended Frame 22 Master-ID Protection Register_H	Section 2.5.3.38
4F8h	PPSE23MSTID_L	Privilege Peripheral Extended Frame 23 Master-ID Protection Register_L	Section 2.5.3.38
4FCh	PPSE23MSTID_H	Privilege Peripheral Extended Frame 23 Master-ID Protection Register_H	Section 2.5.3.38
500h	PPSE24MSTID_L	Privilege Peripheral Extended Frame 24 Master-ID Protection Register_L	Section 2.5.3.38
504h	PPSE24MSTID_H	Privilege Peripheral Extended Frame 24 Master-ID Protection Register_H	Section 2.5.3.38
508h	PPSE25MSTID_L	Privilege Peripheral Extended Frame 25 Master-ID Protection Register_L	Section 2.5.3.38
50Ch	PPSE25MSTID_H	Privilege Peripheral Extended Frame 25 Master-ID Protection Register_H	Section 2.5.3.38
510h	PPSE26MSTID_L	Privilege Peripheral Extended Frame 26 Master-ID Protection Register_L	Section 2.5.3.38
514h	PPSE26MSTID_H	Privilege Peripheral Extended Frame 26 Master-ID Protection Register_H	Section 2.5.3.38
518h	PPSE27MSTID_L	Privilege Peripheral Extended Frame 27 Master-ID Protection Register_L	Section 2.5.3.38
51Ch	PPSE27MSTID_H	Privilege Peripheral Extended Frame 27 Master-ID Protection Register_H	Section 2.5.3.38
520h	PPSE28MSTID_L	Privilege Peripheral Extended Frame 28 Master-ID Protection Register_L	Section 2.5.3.38
524h	PPSE28MSTID_H	Privilege Peripheral Extended Frame 28 Master-ID Protection Register_H	Section 2.5.3.38
528h	PPSE29MSTID_L	Privilege Peripheral Extended Frame 29 Master-ID Protection Register_L	Section 2.5.3.38
52Ch	PPSE29MSTID_H	Privilege Peripheral Extended Frame 29 Master-ID Protection Register_H	Section 2.5.3.38
530h	PPSE30MSTID_L	Privilege Peripheral Extended Frame 30 Master-ID Protection Register_L	Section 2.5.3.38
534h	PPSE30MSTID_H	Privilege Peripheral Extended Frame 30 Master-ID Protection Register_H	Section 2.5.3.38
538h	PPSE31MSTID_L	Privilege Peripheral Extended Frame 31 Master-ID Protection Register_L	Section 2.5.3.38
53Ch	PPSE31MSTID_H	Privilege Peripheral Extended Frame 31 Master-ID Protection Register_H	Section 2.5.3.38
540h	PCS0MSTID	Peripheral Memory Frame Master-ID Protection Register0	Section 2.5.3.39
544h	PCS1MSTID	Peripheral Memory Frame Master-ID Protection Register1	Section 2.5.3.39
548h	PCS2MSTID	Peripheral Memory Frame Master-ID Protection Register2	Section 2.5.3.39
54Ch	PCS3MSTID	Peripheral Memory Frame Master-ID Protection Register3	Section 2.5.3.39

Table 2-85. Peripheral Central Resource Control Registers (continued)

Offset	Acronym	Register Description	Section
550h	PCS4MSTID	Peripheral Memory Frame Master-ID Protection Register4	Section 2.5.3.39
554h	PCS5MSTID	Peripheral Memory Frame Master-ID Protection Register5	Section 2.5.3.39
558h	PCS6MSTID	Peripheral Memory Frame Master-ID Protection Register6	Section 2.5.3.39
55Ch	PCS7MSTID	Peripheral Memory Frame Master-ID Protection Register7	Section 2.5.3.39
560h	PCS8MSTID	Peripheral Memory Frame Master-ID Protection Register8	Section 2.5.3.39
564h	PCS9MSTID	Peripheral Memory Frame Master-ID Protection Register9	Section 2.5.3.39
568h	PCS10MSTID	Peripheral Memory Frame Master-ID Protection Register10	Section 2.5.3.39
56Ch	PCS11MSTID	Peripheral Memory Frame Master-ID Protection Register11	Section 2.5.3.39
570h	PCS12MSTID	Peripheral Memory Frame Master-ID Protection Register12	Section 2.5.3.39
574h	PCS13MSTID	Peripheral Memory Frame Master-ID Protection Register13	Section 2.5.3.39
578h	PCS14MSTID	Peripheral Memory Frame Master-ID Protection Register14	Section 2.5.3.39
57Ch	PCS15MSTID	Peripheral Memory Frame Master-ID Protection Register15	Section 2.5.3.39
580h	PCS16MSTID	Peripheral Memory Frame Master-ID Protection Register16	Section 2.5.3.39
584h	PCS17MSTID	Peripheral Memory Frame Master-ID Protection Register17	Section 2.5.3.39
588h	PCS18MSTID	Peripheral Memory Frame Master-ID Protection Register18	Section 2.5.3.39
58Ch	PCS19MSTID	Peripheral Memory Frame Master-ID Protection Register19	Section 2.5.3.39
590h	PCS20MSTID	Peripheral Memory Frame Master-ID Protection Register20	Section 2.5.3.39
594h	PCS21MSTID	Peripheral Memory Frame Master-ID Protection Register21	Section 2.5.3.39
598h	PCS22MSTID	Peripheral Memory Frame Master-ID Protection Register22	Section 2.5.3.39
59Ch	PCS23MSTID	Peripheral Memory Frame Master-ID Protection Register23	Section 2.5.3.39
5A0h	PCS24MSTID	Peripheral Memory Frame Master-ID Protection Register24	Section 2.5.3.39
5A4h	PCS25MSTID	Peripheral Memory Frame Master-ID Protection Register25	Section 2.5.3.39
5A8h	PCS26MSTID	Peripheral Memory Frame Master-ID Protection Register26	Section 2.5.3.39
5ACh	PCS27MSTID	Peripheral Memory Frame Master-ID Protection Register27	Section 2.5.3.39
5B0h	PCS28MSTID	Peripheral Memory Frame Master-ID Protection Register28	Section 2.5.3.39
5B4h	PCS29MSTID	Peripheral Memory Frame Master-ID Protection Register29	Section 2.5.3.39
5B8h	PCS30MSTID	Peripheral Memory Frame Master-ID Protection Register30	Section 2.5.3.39
5BCh	PCS31MSTID	Peripheral Memory Frame Master-ID Protection Register31	Section 2.5.3.39
5C0h	PPCS0MSTID	Privileged Peripheral Memory Frame Master-ID Protection Register0	Section 2.5.3.40
5C4h	PPCS1MSTID	Privileged Peripheral Memory Frame Master-ID Protection Register1	Section 2.5.3.40
5C8h	PPCS2MSTID	Privileged Peripheral Memory Frame Master-ID Protection Register2	Section 2.5.3.40
5CCh	PPCS3MSTID	Privileged Peripheral Memory Frame Master-ID Protection Register3	Section 2.5.3.40
5D0h	PPCS4MSTID	Privileged Peripheral Memory Frame Master-ID Protection Register4	Section 2.5.3.40
5D4h	PPCS5MSTID	Privileged Peripheral Memory Frame Master-ID Protection Register5	Section 2.5.3.40
5D8h	PPCS6MSTID	Privileged Peripheral Memory Frame Master-ID Protection Register6	Section 2.5.3.40
5DCh	PPCS7MSTID	Privileged Peripheral Memory Frame Master-ID Protection Register7	Section 2.5.3.40

2.5.3.1 Peripheral Memory Protection Set Register 0 (PMPROTSET0)

This register is shown in Figure 2-72 and described in Table 2-86.

NOTE: Only those bits that have a slave at the corresponding bit position are implemented. Writes to non-implemented bits have no effect and reads are 0.

Figure 2-72. Peripheral Memory Protection Set Register 0 (PMPROTSET0) (offset = 00h)

31	0
PCS[31-0]PROTSET	
R/WP-0	

LEGEND: R/W = Read/Write; WP = Write in privileged mode only; -*n* = value after reset

Table 2-86. Peripheral Memory Protection Set Register 0 (PMPROTSET0) Field Descriptions

Bit	Field	Value	Description
31-0	PCS[31-0]PROTSET		Peripheral memory frame protection set.
		0	*Read:* The peripheral memory frame*n* can be written to and read from in both user and privileged modes. *Write:* The bit is unchanged.
		1	*Read:* The peripheral memory frame*n* can be written to only in privileged mode, but it can be read in both user and privileged modes. *Write:* The corresponding bit in PMPROTSET0 and PMPROTCLR0 registers is set to 1.

2.5.3.2 Peripheral Memory Protection Set Register 1 (PMPROTSET1)

This register is shown in Figure 2-73 and described in Table 2-87.

NOTE: Only those bits that have a slave at the corresponding bit position are implemented. Writes to unimplemented bits have no effect and reads are 0.

Figure 2-73. Peripheral Memory Protection Set Register 1 (PMPROTSET1) (offset = 04h)

31	0
PCS[63-32]PROTSET	
R/WP-0	

LEGEND: R/W = Read/Write; WP = Write in privileged mode only; -*n* = value after reset

Table 2-87. Peripheral Memory Protection Set Register 1 (PMPROTSET1) Field Descriptions

Bit	Field	Value	Description
31-0	PCS[63-32]PROTSET		Peripheral memory frame protection set.
		0	*Read:* The peripheral memory frame*n* can be written to and read from in both user and privileged modes. *Write:* The bit is unchanged.
		1	*Read:* The peripheral memory frame*n* can be written to only in privileged mode, but it can be read in both user and privileged modes. *Write:* The corresponding bit in PMPROTSET1 and PMPROTCLR1 registers is set to 1.

2.5.3.3 Peripheral Memory Protection Clear Register 0 (PMPROTCLR0)

This register is shown in Figure 2-74 and described in Table 2-88.

NOTE: Only those bits that have a slave at the corresponding bit position are implemented. Writes to unimplemented bits have no effect and reads are 0.

Figure 2-74. Peripheral Memory Protection Clear Register 0 (PMPROTCLR0) (offset = 10h)

31 … 0
PCS[31-0]PROTCLR
R/WP-0

LEGEND: R/W = Read/Write; WP = Write in privileged mode only; -*n* = value after reset

Table 2-88. Peripheral Memory Protection Clear Register 0 (PMPROTCLR0) Field Descriptions

Bit	Field	Value	Description
31-0	PCS[31-0]PROTCLR		Peripheral memory frame protection clear.
		0	*Read:* The peripheral memory frame*n* can be written to and read from in both user and privileged modes. *Write:* The bit is unchanged.
		1	*Read:* The peripheral memory frame*n* can be written to only in privileged mode, but it can be read in both user and privileged modes. *Write:* The corresponding bit in PMPROTSET0 and PMPROTCLR0 registers is cleared to 0.

2.5.3.4 Peripheral Memory Protection Clear Register 1 (PMPROTCLR1)

This register is shown in Figure 2-75 and described in Table 2-89.

NOTE: Only those bits that have a slave at the corresponding bit position are implemented. Writes to unimplemented bits have no effect and reads are 0.

Figure 2-75. Peripheral Memory Protection Clear Register 1 (PMPROTCLR1) (offset = 14h)

31 … 0
PCS[63-32]PROTCLR
R/WP-0

LEGEND: R/W = Read/Write; WP = Write in privileged mode only; -*n* = value after reset

Table 2-89. Peripheral Memory Protection Clear Register 1 (PMPROTCLR1) Field Descriptions

Bit	Field	Value	Description
31-0	PCS[63-32]PROTCLR		Peripheral memory frame protection clear.
		0	*Read:* The peripheral memory frame*n* can be written to and read from in both user and privileged modes. *Write:* The bit is unchanged.
		1	*Read:* The peripheral memory frame*n* can be written to only in privileged mode, but it can be read in both user and privileged modes. *Write:* The corresponding bit in PMPROTSET1 and PMPROTCLR1 registers is cleared to 0.

2.5.3.5 Peripheral Protection Set Register 0 (PPROTSET0)

There is one bit for each quadrant for PS0 to PS7.

The following are the ways that quadrants are used within a PS frame:

a. The slave uses all the four quadrants.

Only the bit corresponding to the quadrant 0 of PSn is implemented. It protects the whole 1K-byte frame. The remaining three bits are not implemented.

b. The slave uses two quadrants.

Each quadrant has to be in one of these groups: (Quad 0 and Quad 1) or (Quad 2 and Quad 3).

For the group Quad0/Quad1, the bit quadrant 0 protects both quadrants 0 and 1. The bit quadrant 1 is not implemented.

For the group Quad2/Quad3, the bit quadrant 2 protects both quadrants 2 and 3. The bit quadrant 3 is not implemented

c. The slave uses only one quadrant.

In this case, the bit, as specified in Table 2-90, protects the slave.

The above arrangement is true for all the peripheral selects (PS0 to PS31), presented in Section 2.5.3.6 - Section 2.5.3.12. This register holds bits for PS0 to PS7 and is shown in Figure 2-76 and described in Table 2-90.

NOTE: Writes to unimplemented bits have no effect and reads are 0.

Figure 2-76. Peripheral Protection Set Register 0 (PPROTSET0) (offset = 20h)

31 ... 0
PS[7-0]QUAD[3-0]PROTSET
R/WP-0

LEGEND: R/W = Read/Write; WP = Write in privileged mode only; -*n* = value after reset

Table 2-90. Peripheral Protection Set Register 0 (PPROTSET0) Field Descriptions

Bit	Field	Value	Description
31-0	PS[7-0]QUAD[3-0] PROTSET		Peripheral select quadrant protection set.
		0	*Read:* The peripheral select quadrant an be written to and read from in both user and privileged modes. *Write:* The bit is unchanged.
		1	*Read:* The peripheral select quadrant can be written to only in privileged mode, but it can be read in both user and privileged modes. *Write:* The corresponding bit in PPROTSET0 and PPROTCLR0 registers is set to 1.

2.5.3.6 Peripheral Protection Set Register 1 (PPROTSET1)

There is one bit for each quadrant for PS8 to PS15. The protection scheme is described in Section 2.5.3.5. This register is shown in Figure 2-77 and described in Table 2-91.

NOTE: Only those bits that have a slave at the corresponding bit position are implemented. Writes to unimplemented bits have no effect and reads are 0.

Figure 2-77. Peripheral Protection Set Register 1 (PPROTSET1) (offset = 24h)

31	0
PS[15-8]QUAD[3-0]PROTSET	
R/WP-0	

LEGEND: R/W = Read/Write; WP = Write in privileged mode only; -*n* = value after reset

Table 2-91. Peripheral Protection Set Register 1 (PPROTSET1) Field Descriptions

Bit	Field	Value	Description
31-0	PS[15-8]QUAD[3-0] PROTSET		Peripheral select quadrant protection set.
		0	*Read:* The peripheral select quadrant can be written to and read from in both user and privileged modes. *Write:* The bit is unchanged.
		1	*Read:* The peripheral select quadrant can be written to only in privileged mode, but it can be read in both user and privileged modes. *Write:* The corresponding bit in PPROTSET1 and PPROTCLR1 registers is set to 1.

2.5.3.7 Peripheral Protection Set Register 2 (PPROTSET2)

There is one bit for each quadrant for PS16 to PS23. The protection scheme is described in Section 2.5.3.5. This register is shown in Figure 2-78 and described in Table 2-92.

NOTE: Only those bits that have a slave at the corresponding bit position are implemented. Writes to unimplemented bits have no effect and reads are 0.

Figure 2-78. Peripheral Protection Set Register 2 (PPROTSET2) (offset = 28h)

31	0
PS[23-16]QUAD[3-0]PROTSET	
R/WP-0	

LEGEND: R/W = Read/Write; WP = Write in privileged mode only; -*n* = value after reset

Table 2-92. Peripheral Protection Set Register 2 (PPROTSET2) Field Descriptions

Bit	Field	Value	Description
31-0	PS[23-16]QUAD[3-0] PROTSET		Peripheral select quadrant protection set.
		0	*Read:* The peripheral select quadrant can be written to and read from in both user and privileged modes. *Write:* The bit is unchanged.
		1	*Read:* The peripheral select quadrant can be written to only in privileged mode, but it can be read in both user and privileged modes. *Write:* The corresponding bit in PPROTSET2 and PPROTCLR2 registers is set to 1.

2.5.3.8 Peripheral Protection Set Register 3 (PPROTSET3)

There is one bit for each quadrant for PS24 to PS31. The protection scheme is described in Section 2.5.3.5. This register is shown in Figure 2-79 and described in Table 2-93.

NOTE: Only those bits that have a slave at the corresponding bit position are implemented. Writes to unimplemented bits have no effect and reads are 0.

Figure 2-79. Peripheral Protection Set Register 3 (PPROTSET3) (offset = 2Ch)

31	0
PS[31-24]QUAD[3-0]PROTSET	
R/WP-0	

LEGEND: R/W = Read/Write; WP = Write in privileged mode only; -*n* = value after reset

Table 2-93. Peripheral Protection Set Register 3 (PPROTSET3) Field Descriptions

Bit	Field	Value	Description
31-0	PS[31-24]QUAD[3-0] PROTSET		Peripheral select quadrant protection set.
		0	*Read:* The peripheral select quadrant can be written to and read from in both user and privileged modes. *Write:* The bit is unchanged.
		1	*Read:* The peripheral select quadrant can be written to only in privileged mode, but it can be read in both user and privileged modes. *Write:* The corresponding bit in PPROTSET3 and PPROTCLR3 registers is set to 1.

2.5.3.9 Peripheral Protection Clear Register 0 (PPROTCLR0)

There is one bit for each quadrant for PS0 to PS7. The protection scheme is described in Section 2.5.3.5. This register is shown in Figure 2-80 and described in Table 2-94.

NOTE: Only those bits that have a slave at the corresponding bit position are implemented. Writes to unimplemented bits have no effect and reads are 0.

Figure 2-80. Peripheral Protection Clear Register 0 (PPROTCLR0) (offset = 40h)

31	0
PS[7-0]QUAD[3-0]PROTCLR	
R/WP-0	

LEGEND: R/W = Read/Write; WP = Write in privileged mode only; -*n* = value after reset

Table 2-94. Peripheral Protection Clear Register 0 (PPROTCLR0) Field Descriptions

Bit	Field	Value	Description
31-0	PS[7-0]QUAD[3-0] PROTCLR		Peripheral select quadrant protection clear.
		0	*Read:* The peripheral select quadrant can be written to and read from in both user and privileged modes. *Write:* The bit is unchanged.
		1	*Read:* The peripheral select quadrant can be written to only in privileged mode, but it can be read in both user and privileged modes. *Write:* The corresponding bit in PPROTSET0 and PPROTCLR0 registers is cleared to 0.

2.5.3.10 Peripheral Protection Clear Register 1 (PPROTCLR1)

There is one bit for each quadrant for PS8 to PS15. The protection scheme is described in Section 2.5.3.5. This register is shown in Figure 2-81 and described in Table 2-95.

NOTE: Only those bits that have a slave at the corresponding bit position are implemented. Writes to unimplemented bits have no effect and reads are 0.

Figure 2-81. Peripheral Protection Clear Register 1 (PPROTCLR1) (offset = 44h)

31	0
PS[15-8]QUAD[3-0]PROTCLR	
R/WP-0	

LEGEND: R/W = Read/Write; WP = Write in privileged mode only; -*n* = value after reset

Table 2-95. Peripheral Protection Clear Register 1 (PPROTCLR1) Field Descriptions

Bit	Field	Value	Description
31-0	PS[15-8]QUAD[3-0] PROTCLR		Peripheral select quadrant protection clear.
		0	*Read:* The peripheral select quadrant can be written to and read from in both user and privileged modes. *Write:* The bit is unchanged.
		1	*Read:* The peripheral select quadrant can be written to only in privileged mode, but it can be read in both user and privileged modes. *Write:* The corresponding bit in PPROTSET1 and PPROTCLR1 registers is cleared to 0.

2.5.3.11 Peripheral Protection Clear Register 2 (PPROTCLR2)

There is one bit for each quadrant for PS16 to PS23. The protection scheme is described in Section 2.5.3.5. This register is shown in Figure 2-82 and described in Table 2-96.

NOTE: Only those bits that have a slave at the corresponding bit position are implemented. Writes to unimplemented bits have no effect and reads are 0.

Figure 2-82. Peripheral Protection Clear Register 2 (PPROTCLR2) (offset = 48h)

31	0
PS[23-16]QUAD[3-0]PROTCLR	
R/WP-0	

LEGEND: R/W = Read/Write; WP = Write in privileged mode only; -*n* = value after reset

Table 2-96. Peripheral Protection Clear Register 2 (PPROTCLR2) Field Descriptions

Bit	Field	Value	Description
31-0	PS[23-16]QUAD[3-0] PROTCLR		Peripheral select quadrant protection clear.
		0	*Read:* The peripheral select quadrant can be written to and read from in both user and privileged modes. *Write:* The bit is unchanged.
		1	*Read:* The peripheral select quadrant can be written to only in privileged mode, but it can be read in both user and privileged modes. *Write:* The corresponding bit in PPROTSET2 and PPROTCLR2 registers is cleared to 0.

2.5.3.12 Peripheral Protection Clear Register 3 (PPROTCLR3)

There is one bit for each quadrant for PS24 to PS31. The protection scheme is described in Section 2.5.3.5. This register is shown in Figure 2-83 and described in Table 2-97.

NOTE: Only those bits that have a slave at the corresponding bit position are implemented. Writes to unimplemented bits have no effect and reads are 0.

Figure 2-83. Peripheral Protection Clear Register 3 (PPROTCLR3) (offset = 4Ch)

31	0
PS[31-24]QUAD[3-0]PROTCLR	
R/WP-0	

LEGEND: R/W = Read/Write; WP = Write in privileged mode only; -*n* = value after reset

Table 2-97. Peripheral Protection Clear Register 3 (PPROTCLR3) Field Descriptions

Bit	Field	Value	Description
31-0	PS[31-24]QUAD[3-0]PROTCLR		Peripheral select quadrant protection clear.
		0	*Read:* The peripheral select quadrant can be written to and read from in both user and privileged modes. *Write:* The bit is unchanged.
		1	*Read:* The peripheral select quadrant can be written to only in privileged mode, but it can be read in both user and privileged modes. *Write:* The corresponding bit in PPROTSET3 and PPROTCLR3 registers is cleared to 0.

2.5.3.13 Peripheral Memory Power-Down Set Register 0 (PCSPWRDWNSET0)

Each bit corresponds to a bit at the same index in the PMPROT register in that they both relate to the same peripheral. This register is shown in Figure 2-84 and described in Table 2-98.

NOTE: Only those bits that have a slave at the corresponding bit position are implemented. Writes to unimplemented bits have no effect and reads are 0.

Figure 2-84. Peripheral Memory Power-Down Set Register 0 (PCSPWRDWNSET0) (offset = 60h)

31 … 0
PCS[31-0]PWRDNSET
R/WP-1

LEGEND: R/W = Read/Write; WP = Write in privileged mode only; -*n* = value after reset

Table 2-98. Peripheral Memory Power-Down Set Register 0 (PCSPWRDWNSET0) Field Descriptions

Bit	Field	Value	Description
31-0	PCS[31-0]PWRDNSET		Peripheral memory clock power-down set.
		0	*Read:* The peripheral memory clock[31-0] is active. *Write:* The bit is unchanged.
		1	*Read:* The peripheral memory clock[31-0] is inactive. *Write:* The corresponding bit in the PCSPWRDWNSET0 and PCSPWRDWNCLR0 registers is set to 1.

2.5.3.14 Peripheral Memory Power-Down Set Register 1 (PCSPWRDWNSET1)

This register is shown in Figure 2-85 and described in Table 2-99.

NOTE: Only those bits that have a slave at the corresponding bit position are implemented. Writes to unimplemented bits have no effect and reads are 0.

Figure 2-85. Peripheral Memory Power-Down Set Register 1 (PCSPWRDWNSET1) (offset = 64h)

31 … 0
PCS[63-32]PWRDNSET
R/WP-1

LEGEND: R/W = Read/Write; WP = Write in privileged mode only; -*n* = value after reset

Table 2-99. Peripheral Memory Power-Down Set Register 1 (PCSPWRDWNSET1) Field Descriptions

Bit	Field	Value	Description
31-0	PCS[63-32]PWRDNSET		Peripheral memory clock power-down set.
		0	*Read:* The peripheral memory clock[63-32] is active. *Write:* The bit is unchanged.
		1	*Read:* The peripheral memory clock[63-32] is inactive. *Write:* The corresponding bit in the PCSPWRDWNSET1 and PCSPWRDWNCLR1 registers is set to 1.

2.5.3.15 Peripheral Memory Power-Down Clear Register 0 (PCSPWRDWNCLR0)

This register is shown in Figure 2-86 and described in Table 2-100.

NOTE: Only those bits that have a slave at the corresponding bit position are implemented. Writes to unimplemented bits have no effect and reads are 0.

Figure 2-86. Peripheral Memory Power-Down Clear Register 0 (PCSPWRDWNCLR0) (offset = 70h)

31	0
PCS[31-0]PWRDNCLR	
R/WP-1	

LEGEND: R/W = Read/Write; WP = Write in privileged mode only; -*n* = value after reset

Table 2-100. Peripheral Memory Power-Down Clear Register 0 (PCSPWRDWNCLR0) Field Descriptions

Bit	Field	Value	Description
31-0	PCS[31-0]PWRDNCLR		Peripheral memory clock power-down clear.
		0	*Read:* The peripheral memory clock[31-0] is active.
			Write: The bit is unchanged.
		1	*Read:* The peripheral memory clock[31-0] is inactive.
			Write: The corresponding bit in the PCSPWRDWNSET0 and PCSPWRDWNCLR0 registers is cleared to 0.

2.5.3.16 Peripheral Memory Power-Down Clear Register 1 (PCSPWRDWNCLR1)

This register is shown in Figure 2-87 and described in Table 2-101.

NOTE: Only those bits that have a slave at the corresponding bit position are implemented. Writes to unimplemented bits have no effect and reads are 0.

Figure 2-87. Peripheral Memory Power-Down Clear Register 1 (PCSPWRDWNCLR1) (offset = 74h)

31	0
PCS[63-32]PWRDNCLR	
R/WP-1	

LEGEND: R/W = Read/Write; WP = Write in privileged mode only; -*n* = value after reset

Table 2-101. Peripheral Memory Power-Down Set Register 1 (PCSPWRDWNCLR1) Field Descriptions

Bit	Field	Value	Description
31-0	PCS[63-32]PWRDNCLR		Peripheral memory clock power-down clear.
		0	*Read:* The peripheral memory clock[63-32] is active.
			Write: The bit is unchanged.
		1	*Read:* The peripheral memory clock[63-32] is inactive.
			Write: The corresponding bit in the PCSPWRDWNSET1 and PCSPWRDWNCLR1 registers is cleared to 0.

2.5.3.17 Peripheral Power-Down Set Register 0 (PSPWRDWNSET0)

There is one bit for each quadrant for PS0 to PS7. Each bit of this register corresponds to the bit at the same index in the corresponding PPROT register in that they relate to the same peripheral. These bits are used to power down/power up the clock to the corresponding peripheral.

For every bit implemented in the PPROT register, there is one bit in the PSnPWRDWN register, except when two peripherals (both in PS area) share buses. In that case, only one Power-Down bit is implemented, at the position corresponding to that peripheral whose quadrant comes first (the lower numbered).

The ways in which quadrants can be used within a frame are identical to what is described under PPROTSET0, Section 2.5.3.5.

This arrangement is the same for bits of PS8 to PS31, presented in Section 2.5.3.18 - Section 2.5.3.24. This register holds bits for PS0 to PS7. This register is shown in Figure 2-88 and described in Table 2-102.

NOTE: Only those bits that have a slave at the corresponding bit position are implemented. Writes to unimplemented bits have no effect and reads are 0.

Figure 2-88. Peripheral Power-Down Set Register 0 (PSPWRDWNSET0) (offset = 80h)

31	0
PS[7-0]QUAD[3-0]PWRDWNSET	
R/WP-1	

LEGEND: R/W = Read/Write; WP = Write in privileged mode only; -*n* = value after reset

Table 2-102. Peripheral Power-Down Set Register 0 (PSPWRDWNSET0) Field Descriptions

Bit	Field	Value	Description
31-0	PS[7-0]QUAD[3-0] PWRDWNSET		Peripheral select quadrant clock power-down set.
		0	*Read:* The clock to the peripheral select quadrant is active. *Write:* The bit is unchanged.
		1	*Read:* The clock to the peripheral select quadrant is inactive. *Write:* The corresponding bit in PSPWRDWNSET0 and PSPWRDWNCLR0 registers is set to 1.

2.5.3.18 Peripheral Power-Down Set Register 1 (PSPWRDWNSET1)

There is one bit for each quadrant for PS8 to PS15. The protection scheme is described in Section 2.5.3.17. This register is shown in Figure 2-89 and described in Table 2-103.

NOTE: Only those bits that have a slave at the corresponding bit position are implemented. Writes to unimplemented bits have no effect and reads are 0.

Figure 2-89. Peripheral Power-Down Set Register 1 (PSPWRDWNSET1) (offset = 84h)

31	0
PS[15-8]QUAD[3-0]PWRDWNSET	
R/WP-1	

LEGEND: R/W = Read/Write; WP = Write in privileged mode only; -*n* = value after reset

Table 2-103. Peripheral Power-Down Set Register 1 (PSPWRDWNSET1) Field Descriptions

Bit	Field	Value	Description
31-0	PS[15-8]QUAD[3-0] PWRDWNSET		Peripheral select quadrant clock power-down set.
		0	*Read:* The clock to the peripheral select quadrant is active. *Write:* The bit is unchanged.
		1	*Read:* The clock to the peripheral select quadrant is inactive. *Write:* The corresponding bit in PSPWRDWNSET1 and PSPWRDWNCLR1 registers is set to 1.

2.5.3.19 Peripheral Power-Down Set Register 2 (PSPWRDWNSET2)

There is one bit for each quadrant for PS16 to PS23. The protection scheme is described in Section 2.5.3.17. This register is shown in Figure 2-90 and described in Table 2-104.

NOTE: Only those bits that have a slave at the corresponding bit position are implemented. Writes to unimplemented bits have no effect and reads are 0.

Figure 2-90. Peripheral Power-Down Set Register 2 (PSPWRDWNSET2) (offset = 88h)

31	0
PS[23-16]QUAD[3-0]PWRDWNSET	
R/WP-1	

LEGEND: R/W = Read/Write; WP = Write in privileged mode only; -*n* = value after reset

Table 2-104. Peripheral Power-Down Set Register 2 (PSPWRDWNSET2) Field Descriptions

Bit	Field	Value	Description
31-0	PS[23-16]QUAD[3-0] PWRDWNSET		Peripheral select quadrant clock power-down set.
		0	*Read:* The clock to the peripheral select quadrant is active. *Write:* The bit is unchanged.
		1	*Read:* The clock to the peripheral select quadrant is inactive. *Write:* The corresponding bit in PSPWRDWNSET2 and PSPWRDWNCLR2 registers is set to 1.

2.5.3.20 Peripheral Power-Down Set Register 3 (PSPWRDWNSET3)

There is one bit for each quadrant for PS24 to PS31. The protection scheme is described in Section 2.5.3.17. This register is shown in Figure 2-91 and described in Table 2-105.

NOTE: Only those bits that have a slave at the corresponding bit position are implemented. Writes to unimplemented bits have no effect and reads are 0.

Figure 2-91. Peripheral Power-Down Set Register 3 (PSPWRDWNSET3) (offset = 8Ch)

31	0
PS[31-24]QUAD[3-0]PWRDWNSET	
R/WP-1	

LEGEND: R/W = Read/Write; WP = Write in privileged mode only; -*n* = value after reset

Table 2-105. Peripheral Power-Down Set Register 3 (PSPWRDWNSET3) Field Descriptions

Bit	Field	Value	Description
31-0	PS[31-24]QUAD[3-0] PWRDWNSET		Peripheral select quadrant clock power-down set.
		0	*Read:* The clock to the peripheral select quadrant is active. *Write:* The bit is unchanged.
		1	*Read:* The clock to the peripheral select quadrant is inactive. *Write:* The corresponding bit in PSPWRDWNSET3 and PSPWRDWNCLR3 registers is set to 1.

2.5.3.21 Peripheral Power-Down Clear Register 0 (PSPWRDWNCLR0)

There is one bit for each quadrant for PS0 to PS7. The protection scheme is described in Section 2.5.3.17. This register is shown in Figure 2-92 and described in Table 2-106.

NOTE: Only those bits that have a slave at the corresponding bit position are implemented. Writes to unimplemented bits have no effect and reads are 0.

Figure 2-92. Peripheral Power-Down Clear Register 0 (PSPWRDWNCLR0) (offset = A0h)

31	0
PS[7-0]QUAD[3-0]PWRDWNCLR	
R/WP-1	

LEGEND: R/W = Read/Write; WP = Write in privileged mode only; -*n* = value after reset

Table 2-106. Peripheral Power-Down Clear Register 0 (PSPWRDWNCLR0) Field Descriptions

Bit	Field	Value	Description
31-0	PS[7-0]QUAD[3-0] PWRDWNCLR		Peripheral select quadrant clock power-down clear.
		0	*Read:* The clock to the peripheral select quadrant is active. *Write:* The bit is unchanged.
		1	*Read:* The clock to the peripheral select quadrant is inactive. *Write:* The corresponding bit in PSPWRDWNSET0 and PSPWRDWNCLR0 registers is cleared to 0.

2.5.3.22 Peripheral Power-Down Clear Register 1 (PSPWRDWNCLR1)

There is one bit for each quadrant for PS8 to PS15. The protection scheme is described in Section 2.5.3.17. This register is shown in Figure 2-93 and described in Table 2-107.

NOTE: Only those bits that have a slave at the corresponding bit position are implemented. Writes to unimplemented bits have no effect and reads are 0.

Figure 2-93. Peripheral Power-Down Clear Register 1 (PSPWRDWNCLR1) (offset = A4h)

31	0
PS[15-8]QUAD[3-0]PWRDWNCLR	
R/WP-1	

LEGEND: R/W = Read/Write; WP = Write in privileged mode only; -*n* = value after reset

Table 2-107. Peripheral Power-Down Clear Register 1 (PSPWRDWNCLR1) Field Descriptions

Bit	Field	Value	Description
31-0	PS[15-8]QUAD[3-0] PWRDWNCLR		Peripheral select quadrant clock power-down clear.
		0	*Read:* The clock to the peripheral select quadrant is active. *Write:* The bit is unchanged.
		1	*Read:* The clock to the peripheral select quadrant is inactive. *Write:* The corresponding bit in PSPWRDWNSET1 and PSPWRDWNCLR1 registers is cleared to 0.

2.5.3.23 Peripheral Power-Down Clear Register 2 (PSPWRDWNCLR2)

There is one bit for each quadrant for PS16 to PS23. The protection scheme is described in Section 2.5.3.17. This register is shown in Figure 2-94 and described in Table 2-108.

NOTE: Only those bits that have a slave at the corresponding bit position are implemented. Writes to unimplemented bits have no effect and reads are 0.

Figure 2-94. Peripheral Power-Down Clear Register 2 (PSPWRDWNCLR2) (offset = A8h)

31	0
PS[23-16]QUAD[3-0]PWRDWNCLR	
R/WP-1	

LEGEND: R/W = Read/Write; WP = Write in privileged mode only; -*n* = value after reset

Table 2-108. Peripheral Power-Down Clear Register 2 (PSPWRDWNCLR2) Field Descriptions

Bit	Field	Value	Description
31-0	PS[23-16]QUAD[3-0] PWRDWNCLR		Peripheral select quadrant clock power-down clear.
		0	*Read:* The clock to the peripheral select quadrant is active. *Write:* The bit is unchanged.
		1	*Read:* The clock to the peripheral select quadrant is inactive. *Write:* The corresponding bit in PSPWRDWNSET2 and PSPWRDWNCLR2 registers is cleared to 0.

2.5.3.24 Peripheral Power-Down Clear Register 3 (PSPWRDWNCLR3)

There is one bit for each quadrant for PS24 to PS31. The protection scheme is described in Section 2.5.3.17. This register is shown in Figure 2-95 and described in Table 2-109.

NOTE: Only those bits that have a slave at the corresponding bit position are implemented. Writes to unimplemented bits have no effect and reads are 0.

Figure 2-95. Peripheral Power-Down Clear Register 3 (PSPWRDWNCLR) (offset = ACh)

31 … 0
PS[31-24]QUAD[3-0]PWRDWNCLR
R/WP-1

LEGEND: R/W = Read/Write; WP = Write in privileged mode only; -*n* = value after reset

Table 2-109. Peripheral Power-Down Clear Register 3 (PSPWRDWNCLR3) Field Descriptions

Bit	Field	Value	Description
31-0	PS[31-24]QUAD[3-0] PWRDWNCLR		Peripheral select quadrant clock power-down clear.
		0	*Read:* The clock to the peripheral select quadrant is active. *Write:* The bit is unchanged.
		1	*Read:* The clock to the peripheral select quadrant is inactive. *Write:* The corresponding bit in PSPWRDWNSET3 and PSPWRDWNCLR3 registers is cleared to 0.

2.5.3.25 Debug Frame Powerdown Set Register (PDPWRDWNSET)

Figure 2-96. Debug Frame Powerdown Set Register (PDPWRDWNSET) (offset = C0h)

31 … 1	0
Reserved	PDWRDWNSET
R-0	R/WP-1

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privileged mode only; -*n* = value after reset

Table 2-110. Debug Frame Powerdown Set Register (PDPWRDWNSET) Field Descriptions

Bit	Field	Value	Description
31-1	Reserved	0	Reads return 0. Writes have no effect.
0	PDWRDWNSET		Debug Frame Powerdown Set Register.
		0	*Read:* The clock to the debug frame is active. *Write:* The bit is unchanged.
		1	*Read:* The clock to the debug frame is inactive. *Write:* Set the bit to 1.

2.5.3.26 Debug Frame Powerdown Clear Register (PDPWRDWNCLR)

Figure 2-97. Debug Frame Powerdown Clear Register (PDPWRDWNCLR) (offset = C4h)

31 1	0
Reserved	PDWRDWNCLR
R-0	R/WP-1

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privileged mode only; -*n* = value after reset

Table 2-111. Debug Frame Powerdown Clear Register (PDPWRDWNCLR) Field Descriptions

Bit	Field	Value	Description
31-1	Reserved	0	Reads return 0. Writes have no effect.
0	PDWRDWNCLR		Debug Frame Powerdown Set Register.
		0	*Read:* The clock to the debug frame is active. *Write:* The bit is unchanged.
		1	*Read:* The clock to the debug frame is inactive. *Write:* Clear the bit to 0.

2.5.3.27 MasterID Protection Write Enable Register (MSTIDWRENA)

Figure 2-98. MasterID Protection Write Enable Register (MSTIDWRENA) (offset = 200h)

31 16
Reserved
R-0

15 4	3 0
Reserved	MSTIDREG_WRENA
R-0	R/WP-5h

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privileged mode only; -*n* = value after reset

Table 2-112. MasterID Protection Write Enable Register (MSTIDWRENA) Field Descriptions

Bit	Field	Value	Description
31-4	Reserved	0	Reads return 0. Writes have no effect.
3-0	MSTIDREG_WRENA		MasterID Register Write Enable. This is a 4-bit key for enabling writes to all Master-ID registers from address offset 0x300-0x5DC. This key must be programmed with 1010 to unlock writes to all Master-ID registers.
		Ah	*Read:* All master-ID registers are unlocked and available for writes. *Write:* Writes to master-ID registers are unlocked.
		Others	*Read:* Writes to all master-ID registers are locked. *Write:* Write to master-ID registers are locked.

2.5.3.28 MasterID Enable Register (MSTIDENA)

Figure 2-99. MasterID Enable Register (MSTIDENA) (offset = 204h)

31	16
Reserved	
R-0	

15	4	3	0
Reserved		MSTID_CHK_ENA	
R-0		R/WP-5h	

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privileged mode only; -*n* = value after reset

Table 2-113. MasterID Enable Register (MSTIDENA) Field Descriptions

Bit	Field	Value	Description
31-4	Reserved	0	Reads return 0. Writes have no effect.
3-0	MSTID_CHK_ENA		MasterID Check Enable. This is a 4-bit key for enabling Master-ID check. This key must be programmed with 1010 to enable Master-ID Check functionality.
		Ah	*Read:* The master-ID check is enabled. *Write:* Enable master-ID check.
		Others	*Read:* The master-ID check is disabled. *Write:* Disable master-ID check.

2.5.3.29 MasterID Diagnostic Control Register (MSTIDDIAGCTRL)

Figure 2-100. MasterID Diagnostic Control Register (MSTIDDIAGCTRL) (offset = 208h)

<table>
<tr><td>31</td><td>16</td></tr>
<tr><td colspan="2">Reserved</td></tr>
<tr><td colspan="2">R-0</td></tr>
</table>

<table>
<tr><td>15</td><td>12</td><td>11</td><td>8</td><td>7</td><td>4</td><td>3</td><td>0</td></tr>
<tr><td colspan="2">Reserved</td><td colspan="2">DIAG_CMP_VALUE</td><td colspan="2">Reserved</td><td colspan="2">DIAG_MODE_ENA</td></tr>
<tr><td colspan="2">R-0</td><td colspan="2">R/WP-0</td><td colspan="2">R-0</td><td colspan="2">R/WP-5h</td></tr>
</table>

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privileged mode only; -*n* = value after reset

Table 2-114. MasterID Diagnostic Control Register (MSTIDDIAGCTRL) Field Descriptions

Bit	Field	Value	Description
31-12	Reserved	0	Reads return 0. Writes have no effect.
11-8	DIAG_CMP_VALUE		Diagnostic Compare Value. The value stored in this register is compared against the programmed master-ID register bits for all accesses. In diagnostic mode, the master-ID register selection depends on the DIAG_CMP_VALUE instead of the input 4-bit master-ID generated by the interconnect. Any mismatch will be signaled to the bus master as a bus error. After the diagnostic mode is enabled in DIAG_MODE_ENA register and a diagnostic compare value is programmed into the DIAG_CMP_VALUE register, the application must issue a dummy diagnostic write access to any one of the peripherals to cause a diagnostic check. For example, if all master-ID protection registers listed from address 0x300-0x5DC are programmed to block master-ID 5 from write access to the peripherals, then the application can program the DIAG_CMP_VALUE to 5. The application can use the CPU (whose master-ID is 0) to issue a dummy write access to any peripheral to cause master-ID violation during diagnostic mode instead of using the bus master whose master-ID is 5 to perform this write access.
		Ah	*Read:* The master-ID check is enabled. *Write:* Enable master-ID check.
		Others	*Read:* The master-ID check is disabled. *Write:* Disable master-ID check.
7-4	Reserved	0	Reads return 0. Writes have no effect.
3-0	DIAG_MODE_ENA		Diagnostic Mode Enable. This is a 4-bit key for enabling Diagnostic Mode. This key must be programmed with 1010 to enable Diagnostic Mode.
		Ah	*Read:* The diagnostic mode is enabled. *Write:* Enable diagnostic mode.
		Others	*Read:* The diagnostic mode is disabled. *Write:* Disable diagnostic mode.

2.5.3.30 Peripheral Frame 0 MasterID Protection Register_L (PS0MSTID_L)

There is one bit for each quadrant for PS0 to PS31.

NOTE: If a module occupies two quadrants, then only the lower quadrant register is used to enable or disable the masterID. The upper quadrant register remains zeros.

The following are the ways that quadrants are used within a PS frame:

a. The slave uses all the four quadrants.

   Only the bit corresponding to the quadrant 0 of PSn is implemented. It protects the whole 1K-byte frame. The remaining three bits are not implemented.

b. The slave uses two quadrants.

   Each quadrant has to be in one of these groups: (Quad 0 and Quad 1) or (Quad 2 and Quad 3).

   For the group Quad0/Quad1, the bit quadrant 0 protects both quadrants 0 and 1. The bit quadrant 1 is not implemented.

   For the group Quad2/Quad3, the bit quadrant 2 protects both quadrants 2 and 3. The bit quadrant 3 is not implemented

c. The slave uses only one quadrant.

   In this case, the bit, as specified in Table 2-115, protects the slave.

The above arrangement is true for all the peripheral selects (PS0 to PS31), presented in Section 2.5.3.31 - Section 2.5.3.32. This register holds bits for PS0 and is shown in Figure 2-101 and described in Table 2-115.

Figure 2-101. Peripheral Frame 0 MasterID Protection Register_L (PS0MSTID_L) (offset = 300h)

31	16
PS0_QUAD1_MSTID	
R/WP-FFFFh	

15	0
PS0_QUAD0_MSTID	
R/WP-FFFFh	

LEGEND: R/W = Read/Write; WP = Write in privileged mode only; -*n* = value after reset

Table 2-115. Peripheral Frame 0 MasterID Protection Register_L (PS0MSTID_L) Field Descriptions

Bit	Field	Value	Description
31-16	PS0_QUAD1_MSTID		MasterID filtering for Quadrant 1 of PS[0]. There are 16 bits for each quadrant in PS frame. Each bit corresponds to a master-ID value. For example, bit 0 corresponds to master-ID 0 and bit 15 corresponds to master-ID 15. These bits set the permission for maximum of 16 masters to address the peripheral mapped in each of the quadrant. The following examples shows the usage of these register bits. (a) If bits 15:0 are 1010_1010_1010_1010, then the peripheral that is mapped to Quadrant 0 of PS[0] can be addressed by Masters with Master-ID equals to 1,3,5,7,9,11,13,15. (b) if bits 15:0 are 0000_0000_0000_0001, then the peripheral that is mapped to Quadrant 0 of PS[0] can only addressed by the master with the Master-ID equal to 0.
		0	*Read:* The corresponding master-ID is not permitted to access the peripheral mapped to this quadrant. *Write:* Disable the permission of the corresponding master to access the peripheral mapped to this quadrant.
		1	*Read:* The corresponding master-ID is permitted to access the peripheral mapped to this quadrant. *Write:* Enable the permission of the corresponding master to access the peripheral mapped to this quadrant.

Table 2-115. Peripheral Frame 0 MasterID Protection Register_L (PS0MSTID_L) Field Descriptions (continued)

Bit	Field	Value	Description
15-0	PS0_QUAD0_MSTID		MasterID filtering for Quadrant 0 of PS[0].
		0	*Read:* The corresponding master-ID is not permitted to access the peripheral mapped to this quadrant. *Write:* Disable the permission of the corresponding master to access the peripheral mapped to this quadrant.
		1	*Read:* The corresponding master-ID is permitted to access the peripheral mapped to this quadrant. *Write:* Enable the permission of the corresponding master to access the peripheral mapped to this quadrant.

2.5.3.31 Peripheral Frame 0 MasterID Protection Register_H (PS0MSTID_H)

There is one bit for each quadrant for PS0 to PS31. The protection scheme is described in Section 2.5.3.30. This register is shown in Figure 2-102 and described in Table 2-116.

Figure 2-102. Peripheral Frame 0 MasterID Protection Register_H (PS0MSTID_H) (offset = 304h)

31	16
PS0_QUAD3_MSTID	
R/WP-FFFFh	

15	0
PS0_QUAD2_MSTID	
R/WP-FFFFh	

LEGEND: R/W = Read/Write; WP = Write in privileged mode only; -*n* = value after reset

Table 2-116. Peripheral Frame 0 MasterID Protection Register_H (PS0MSTID_H) Field Descriptions

Bit	Field	Value	Description
31-16	PS0_QUAD3_MSTID		MasterID filtering for Quadrant 3 of PS[0].
		0	*Read:* The corresponding master-ID is not permitted to access the peripheral. *Write:* Disable the permission of the corresponding master to access the peripheral.
		1	*Read:* The corresponding master-ID is permitted to access the peripheral. *Write:* Enable the permission of the corresponding master to access the peripheral.
15-0	PS0_QUAD2_MSTID		MasterID filtering for Quadrant 2 of PS[0].
		0	*Read:* The corresponding master-ID is not permitted to access the peripheral. *Write:* Disable the permission of the corresponding master to access the peripheral.
		1	*Read:* The corresponding master-ID is permitted to access the peripheral. *Write:* Enable the permission of the corresponding master to access the peripheral.

2.5.3.32 Peripheral Frame n MasterID Protection Register_L/H (PS[1-31]MSTID_L/H)

There is one bit for each quadrant for PS0 to PS31. The protection scheme is described in Section 2.5.3.30. This register is shown in Figure 2-103 and described in Table 2-117.

Figure 2-103. Peripheral Frame n MasterID Protection Register_L/H (PSnMSTID_L/H) (offset = 308h-3FCh)

31	16
PSn_QUAD3_MSTID or PSn_QUAD1_MSTID	
R/WP-FFFFh	

15	0
PSn_QUAD2_MSTID or PSn_QUAD0_MSTID	
R/WP-FFFFh	

LEGEND: R/W = Read/Write; WP = Write in privileged mode only; -*n* = value after reset

Table 2-117. Peripheral Frame n MasterID Protection Register_L/H (PSnMSTID_L/H) Field Descriptions

Bit	Field	Value	Description
31-16	PSn_QUAD3_MSTID or PSn_QUAD1_MSTID		n: 1 to 31. L: quadrant0 and quadrant1. H: quadrant2 and quadrant3. MasterID filtering for Quadrant 3 of PS[n] or Quadrant 1 of PS[n].
		0	*Read:* The corresponding master-ID is not permitted to access the peripheral. *Write:* Disable the permission of the corresponding master to access the peripheral.
		1	*Read:* The corresponding master-ID is permitted to access the peripheral. *Write:* Enable the permission of the corresponding master to access the peripheral.
15-0	PSn_QUAD2_MSTID or PSn_QUAD0_MSTID		MasterID filtering for Quadrant 2 of PS[n] or Quadrant 0 of PS[n].
		0	*Read:* The corresponding master-ID is not permitted to access the peripheral. *Write:* Disable the permission of the corresponding master to access the peripheral.
		1	*Read:* The corresponding master-ID is permitted to access the peripheral. *Write:* Enable the permission of the corresponding master to access the peripheral.

2.5.3.33 Privileged Peripheral Frame 0 MasterID Protection Register_L (PPS0MSTID_L)

Figure 2-104. Privileged Peripheral Frame 0 MasterID Protection Register_L (PPS0MSTID_L) (offset = 400h)

31	16
PPS0_QUAD1_MSTID	
R/WP-FFFFh	

15	0
PPS0_QUAD0_MSTID	
R/WP-FFFFh	

LEGEND: R/W = Read/Write; WP = Write in privileged mode only; -*n* = value after reset

Table 2-118. Privileged Peripheral Frame 0 MasterID Protection Register_L (PPS0MSTID_L) Field Descriptions

Bit	Field	Value	Description
31-16	PPS0_QUAD1_MSTID		MasterID filtering for Quadrant 1 of PPS[0]. There are 16 bits for each quadrant in PPS frame. Each bit corresponds to a master-ID value. For example, bit 0 corresponds to master-ID 0 and bit 15 corresponds to master-ID 15. These bits set the permission for maximum of 16 masters to address the peripheral mapped in each of the quadrant. The following examples shows the usage of these register bits. (a) If bits 15:0 are 1010_1010_1010_1010, then the peripheral that is mapped to Quadrant 0 of PPS[0] can be addressed by Masters with Master-ID equals to 1,3,5,7,9,11,13,15. (b) if bits 15:0 are 0000_0000_0000_0001, then the peripheral that is mapped to Quadrant 0 of PPS[0] can only addressed by the master with the Master-ID equal to 0.
		0	*Read:* The corresponding master-ID is not permitted to access the peripheral mapped to this quadrant. *Write:* Disable the permission of the corresponding master to access the peripheral mapped to this quadrant.
		1	*Read:* The corresponding master-ID is permitted to access the peripheral mapped to this quadrant. *Write:* Enable the permission of the corresponding master to access the peripheral mapped to this quadrant.
15-0	PPS0_QUAD0_MSTID		MasterID filtering for Quadrant 0 of PPS[0].
		0	*Read:* The corresponding master-ID is not permitted to access the peripheral mapped to this quadrant. *Write:* Disable the permission of the corresponding master to access the peripheral mapped to this quadrant.
		1	*Read:* The corresponding master-ID is permitted to access the peripheral mapped to this quadrant. *Write:* Enable the permission of the corresponding master to access the peripheral mapped to this quadrant.

2.5.3.34 Privileged Peripheral Frame 0 MasterID Protection Register_H (PPS0MSTID_H)

Figure 2-105. Privileged Peripheral Frame 0 MasterID Protection Register_H (PPS0MSTID_H) (offset = 404h)

31	16
PPS0_QUAD3_MSTID	
R/WP-FFFFh	

15	0
PPS0_QUAD2_MSTID	
R/WP-FFFFh	

LEGEND: R/W = Read/Write; WP = Write in privileged mode only; -*n* = value after reset

Table 2-119. Privileged Peripheral Frame 0 MasterID Protection Register_H (PPS0MSTID_H) Field Description

Bit	Field	Value	Description
31-16	PPS0_QUAD3_MSTID		MasterID filtering for Quadrant 3 of PPS[0].
		0	*Read:* The corresponding master-ID is not permitted to access the peripheral. *Write:* Disable the permission of the corresponding master to access the peripheral.
		1	*Read:* The corresponding master-ID is permitted to access the peripheral. *Write:* Enable the permission of the corresponding master to access the peripheral.
15-0	PPS0_QUAD2_MSTID		MasterID filtering for Quadrant 2 of PPS[0].
		0	*Read:* The corresponding master-ID is not permitted to access the peripheral. *Write:* Disable the permission of the corresponding master to access the peripheral.
		1	*Read:* The corresponding master-ID is permitted to access the peripheral. *Write:* Enable the permission of the corresponding master to access the peripheral.

2.5.3.35 Privileged Peripheral Frame n MasterID Protection Register_L/H (PPS[1-7]MSTID_L/H)

Figure 2-106. Privileged Peripheral Frame n MasterID Protection Register_L/H (PPSnMSTID_L/H) (offset = 408h-43Ch)

31	16
PPSn_QUAD3_MSTID or PPSn_QUAD1_MSTID	
R/WP-FFFFh	

15	0
PPSn_QUAD2_MSTID or PPSn_QUAD0_MSTID	
R/WP-FFFFh	

LEGEND: R/W = Read/Write; WP = Write in privileged mode only; -*n* = value after reset

Table 2-120. Privileged Peripheral Frame n MasterID Protection Register_L/H (PPSnMSTID_L/H) Field Descriptions

Bit	Field	Value	Description
31-16	PPSn_QUAD3_MSTID or PPSn_QUAD1_MSTID		n: 1 to 7. L: quadrant0 and quadrant1. H: quadrant2 and quadrant3. MasterID filtering for Quadrant 3 of PPS[n] or Quadrant 1 of PPS[n].
		0	*Read:* The corresponding master-ID is not permitted to access the peripheral. *Write:* Disable the permission of the corresponding master to access the peripheral.
		1	*Read:* The corresponding master-ID is permitted to access the peripheral. *Write:* Enable the permission of the corresponding master to access the peripheral.
15-0	PPSn_QUAD2_MSTID or PPSn_QUAD0_MSTID		MasterID filtering for Quadrant 2 of PPS[n] or Quadrant 0 of PPS[n].
		0	*Read:* The corresponding master-ID is not permitted to access the peripheral. *Write:* Disable the permission of the corresponding master to access the peripheral.
		1	*Read:* The corresponding master-ID is permitted to access the peripheral. *Write:* Enable the permission of the corresponding master to access the peripheral.

2.5.3.36 Privileged Peripheral Extended Frame 0 MasterID Protection Register_L (PPSE0MSTID_L)

Figure 2-107. Privileged Peripheral Extended Frame 0 MasterID Protection Register_L (PPSE0MSTID_L) (offset = 440h)

31	16
PPSE0_QUAD1_MSTID	
R/WP-FFFFh	

15	0
PPSE0_QUAD0_MSTID	
R/WP-FFFFh	

LEGEND: R/W = Read/Write; WP = Write in privileged mode only; -*n* = value after reset

Table 2-121. Privileged Peripheral Extended Frame 0 MasterID Protection Register_L (PPSE0MSTID_L) Field Descriptions

Bit	Field	Value	Description
31-16	PPSE0_QUAD1_MSTID		MasterID filtering for Quadrant 1 of PPSE[0]. There are 16 bits for each quadrant in PPSE frame. Each bit corresponds to a master-ID value. For example, bit 0 corresponds to master-ID 0 and bit 15 corresponds to master-ID 15. These bits set the permission for maximum of 16 masters to address the peripheral mapped in each of the quadrant. The following examples shows the usage of these register bits. (a) If bits 15:0 are 1010_1010_1010_1010, then the peripheral that is mapped to Quadrant 0 of PPSE[0] can be addressed by Masters with Master-ID equals to 1,3,5,7,9,11,13,15. (b) if bits 15:0 are 0000_0000_0000_0001, then the peripheral that is mapped to Quadrant 0 of PPSE[0] can only addressed by the master with the Master-ID equal to 0.
		0	*Read:* The corresponding master-ID is not permitted to access the peripheral mapped to this quadrant. *Write:* Disable the permission of the corresponding master to access the peripheral mapped to this quadrant.
		1	*Read:* The corresponding master-ID is permitted to access the peripheral mapped to this quadrant. *Write:* Enable the permission of the corresponding master to access the peripheral mapped to this quadrant.
15-0	PPSE0_QUAD0_MSTID		MasterID filtering for Quadrant 0 of PPSE[0].
		0	*Read:* The corresponding master-ID is not permitted to access the peripheral mapped to this quadrant. *Write:* Disable the permission of the corresponding master to access the peripheral mapped to this quadrant.
		1	*Read:* The corresponding master-ID is permitted to access the peripheral mapped to this quadrant. *Write:* Enable the permission of the corresponding master to access the peripheral mapped to this quadrant.

2.5.3.37 Privileged Peripheral Extended Frame 0 MasterID Protection Register_H (PPSE0MSTID_H)

Figure 2-108. Privileged Peripheral Extended Frame 0 MasterID Protection Register_H (PPSE0MSTID_H) (offset = 444h)

31	16
PPSE0_QUAD3_MSTID	
R/WP-FFFFh	

15	0
PPSE0_QUAD2_MSTID	
R/WP-FFFFh	

LEGEND: R/W = Read/Write; WP = Write in privileged mode only; -*n* = value after reset

Table 2-122. Privileged Peripheral Extended Frame 0 MasterID Protection Register_H (PPSE0MSTID_H) Field Descriptions

Bit	Field	Value	Description
31-16	PPSE0_QUAD3_MSTID		MasterID filtering for Quadrant 3 of PPSE[0].
		0	*Read:* The corresponding master-ID is not permitted to access the peripheral. *Write:* Disable the permission of the corresponding master to access the peripheral.
		1	*Read:* The corresponding master-ID is permitted to access the peripheral. *Write:* Enable the permission of the corresponding master to access the peripheral.
15-0	PPSE0_QUAD2_MSTID		MasterID filtering for Quadrant 2 of PPSE[0].
		0	*Read:* The corresponding master-ID is not permitted to access the peripheral. *Write:* Disable the permission of the corresponding master to access the peripheral.
		1	*Read:* The corresponding master-ID is permitted to access the peripheral. *Write:* Enable the permission of the corresponding master to access the peripheral.

2.5.3.38 Privileged Peripheral Extended Frame n MasterID Protection Register_L/H (PPSE[1-31]MSTID_L/H)

Figure 2-109. Privileged Peripheral Extended Frame n MasterID Protection Register_L/H (PPSEnMSTID_L/H) (offset = 448h-53Ch)

31	16
PPSEn_QUAD3_MSTID or PPSEn_QUAD1_MSTID	
R/WP-FFFFh	

15	0
PPSEn_QUAD2_MSTID or PPSEn_QUAD0_MSTID	
R/WP-FFFFh	

LEGEND: R/W = Read/Write; WP = Write in privileged mode only; -*n* = value after reset

Table 2-123. Privileged Peripheral Extended Frame n MasterID Protection Register_L/H (PPSEnMSTID_L/H) Field Descriptions

Bit	Field	Value	Description
31-16	PPSEn_QUAD3_MSTID or PPSEn_QUAD1_MSTID	0	n: 1 to 31. L: quadrant0 and quadrant1. H: quadrant2 and quadrant3. MasterID filtering for Quadrant 3 of PPSE[n] or Quadrant 1 of PPSE[n]. *Read:* The corresponding master-ID is not permitted to access the peripheral. *Write:* Disable the permission of the corresponding master to access the peripheral.
		1	*Read:* The corresponding master-ID is permitted to access the peripheral. *Write:* Enable the permission of the corresponding master to access the peripheral.
15-0	PPSEn_QUAD2_MSTID or PPSEn_QUAD0_MSTID	0	MasterID filtering for Quadrant 2 of PPSE[n] or Quadrant 0 of PPSE[n]. *Read:* The corresponding master-ID is not permitted to access the peripheral. *Write:* Disable the permission of the corresponding master to access the peripheral.
		1	*Read:* The corresponding master-ID is permitted to access the peripheral. *Write:* Enable the permission of the corresponding master to access the peripheral.

2.5.3.39 Peripheral Memory Frame MasterID Protection Register (PCS[0-31]MSTID)

Figure 2-110. Peripheral Memory Frame MasterID Protection Register (PCSnMSTID) (offset = 540h-5BCh)

31	16
PCS(2n+1)_MSTID	
R/WP-FFFFh	

15	0
PCS(2n)_MSTID	
R/WP-FFFFh	

LEGEND: R/W = Read/Write; WP = Write in privileged mode only; -*n* = value after reset

Table 2-124. Peripheral Memory Frame MasterID Protection Register (PCSnMSTID) Field Descriptions

Bit	Field	Value	Description
31-16	PCS(2n+1)_MSTID		MasterID filtering for PCS[2n+1], where n = 0 to 31.
		0	*Read:* The corresponding master-ID is not permitted to access the peripheral. *Write:* Disable the permission of the corresponding master to access the peripheral.
		1	*Read:* The corresponding master-ID is permitted to access the peripheral. *Write:* Enable the permission of the corresponding master to access the peripheral.
15-0	PCS(2n)_MSTID		MasterID filtering for PCS[2n], where n = 0 to 31.
		0	*Read:* The corresponding master-ID is not permitted to access the peripheral. *Write:* Disable the permission of the corresponding master to access the peripheral.
		1	*Read:* The corresponding master-ID is permitted to access the peripheral. *Write:* Enable the permission of the corresponding master to access the peripheral.

2.5.3.40 Privileged Peripheral Memory Frame MasterID Protection Register (PPCS[0-7]MSTID)

Figure 2-111. Privileged Peripheral Memory Frame MasterID Protection Register (PPCSnMSTID) (offset = 5C0h-5DCh)

31	16
PPCS(2n+1)_MSTID	
R/WP-FFFFh	

15	0
PPCS(2n)_MSTID	
R/WP-FFFFh	

LEGEND: R/W = Read/Write; WP = Write in privileged mode only; -*n* = value after reset

Table 2-125. Privileged Peripheral Memory Frame MasterID Protection Register (PPCSnMSTID) Field Descriptions

Bit	Field	Value	Description
31-16	PPCS(2n+1)_MSTID		MasterID filtering for PPCS[2n+1], where n = 0 to 7.
		0	*Read:* The corresponding master-ID is not permitted to access the peripheral. *Write:* Disable the permission of the corresponding master to access the peripheral.
		1	*Read:* The corresponding master-ID is permitted to access the peripheral. *Write:* Enable the permission of the corresponding master to access the peripheral.
15-0	PPCS(2n)_MSTID		MasterID filtering for PPCS[2n], where n = 0 to 7.
		0	*Read:* The corresponding master-ID is not permitted to access the peripheral. *Write:* Disable the permission of the corresponding master to access the peripheral.
		1	*Read:* The corresponding master-ID is permitted to access the peripheral. *Write:* Enable the permission of the corresponding master to access the peripheral.

Chapter 3

SPNU563A–March 2018

SCR Control Module (SCM)

This chapter describes the SCR control module (SCM). SCR is the CPU Interconnect Subsystem.

Topic **Page**

3.1 Overview

The SCR (Switch Central Resource) Control Module (SCM) provides a means to control and monitor the main interconnect.

Interconnect hardware checker performs four major functional checks on interconnect:

- Arbitration
- Timeout
- Protocol conversion
- Parity on control / address signals

Any of these errors will force the interconnect to trigger an error event to ESM group 1 (see ESM group1 mapping). It is recommended that you should set the interconnect error to toggle ESM pin action. The reason is that if an error occurs and CPU can not access to Flash or RAM to run diagnostic or retry, external monitoring ASIC can be notified by ESM error pin.

3.1.1 Features

The following main features are supported:

- Compares the real time running counter of transaction command request to transaction command accept from each initiator agent (IA - is the bus master that initiates transactions to the interconnect. Refer to the Interconnect chapter for more definition and connection) with the REQ2ACCEPT threshold value. if the real time counter is equal or larger than the threshold, the SCM will trigger an error event to ESM. A corresponding status bit of the corresponding IA will also be set.
- Compares the real time running counter of transaction command request accepted to transaction command response accepted from each IA with the REQ2RESP threshold value. If the real time counter is equal or larger than the threshold, the SCM will trigger an error event to ESM. A corresponding status bit of the corresponding IA will also be set.
- Provides a control key to clear the time out counters overrun inside hardware checker of the interconnect. This control bit will clear all registers and make the timeout module available to restart properly.
- Provides a control key to start a self-test sequence of the interconnect hardware checker.
- Provides a control key to clear global error flag inside interconnect hardware checker.
- Captures the active status of each IA and target agent (TA - is the bus slave receives transaction request from interconnect and responses to it. Refer to the Interconnect chapter of the TRM for more definition and connection) of the interconnect. The active status bit indicates that there is still pending transactions inside interconnect.
- Provides the ability to override parity polarity of the interconnect hardware checker so that the parity detection logic can be self-tested.

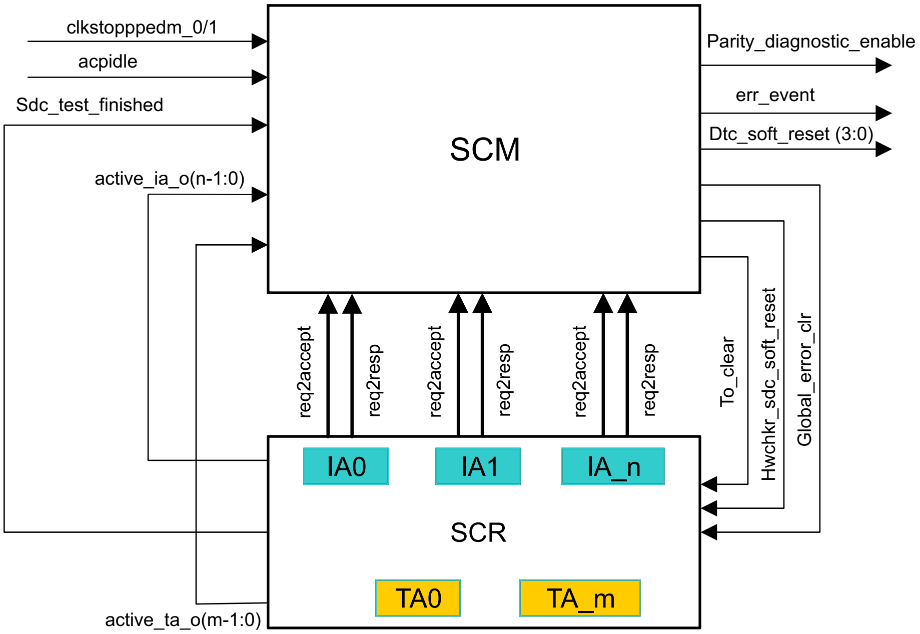
3.1.2 System Block Diagram

Figure 3-1 shows the system level block diagram of SCM and interconnect (SCR).

SCM compares the transaction command request to transaction command accept (req2accept) counters and transaction command request to transaction command response (req2resp) counters of each initiator agent (IA) to the corresponding threshold values (programmable). If the req2accept or req2resp counters are larger than or equal to the threshold, SCM will generate error event to ESM module.

SCM can clear the req2accept and req2resp counters inside interconnect SCR. It can also initiate self-test sequence to make sure the hardware checker diagnostic logic is functioning properly.

Figure 3-1. System Level Block Diagram



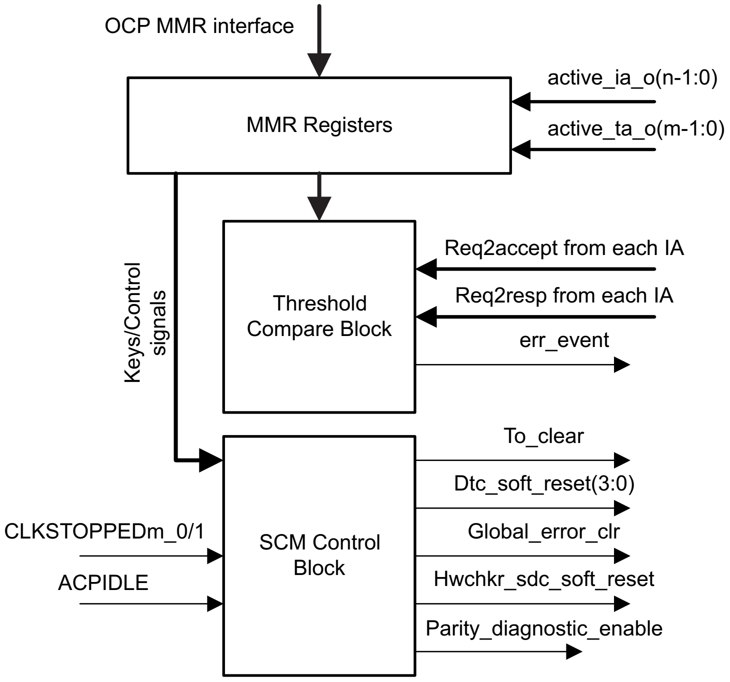
n is the maximum number of IA. m is the maximum number of TA.

3.2 Module Operation

3.2.1 Block Diagram

Figure 3-2 shows the block diagram of SCM.

Figure 3-2. SCM Block Diagram



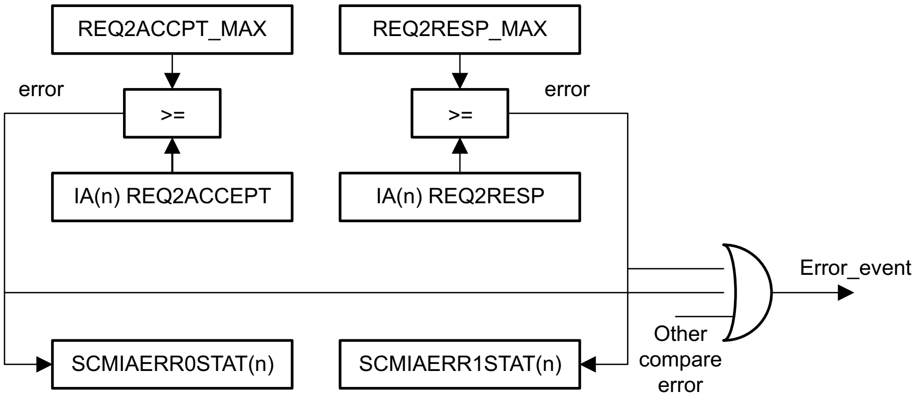
n is the maximum number of IA. m is the maximum number of TA.

3.2.2 Timeout Threshold Compare Block

The threshold compare block (Figure 3-3) takes the real-time counters (command request to command accepted and command request to command response) from each IA of the interconnect hardware checker module and compare against the corresponding threshold value in SCM every cycle. If any IA comparison fails, the SCM module will update the corresponding status bit in SCMIAERR0STAT and SCMIAERR1STAT registers. SCMIAERR0STAT logs the time out error for command request to command accepted. SCMIAERR1STAT logs the time out error for command request to command response. Any status bit set in these two status registers will trigger an error event to ESM (Error Signaling Module) and will not trigger again until cleared by CPU.

Any of these status bits can be cleared by a privilege write to the individual bit. The write clear from CPU to these status bits will always take higher priority than setting of the status bits from the interconnect.

Figure 3-3. Timeout Threshold Compare



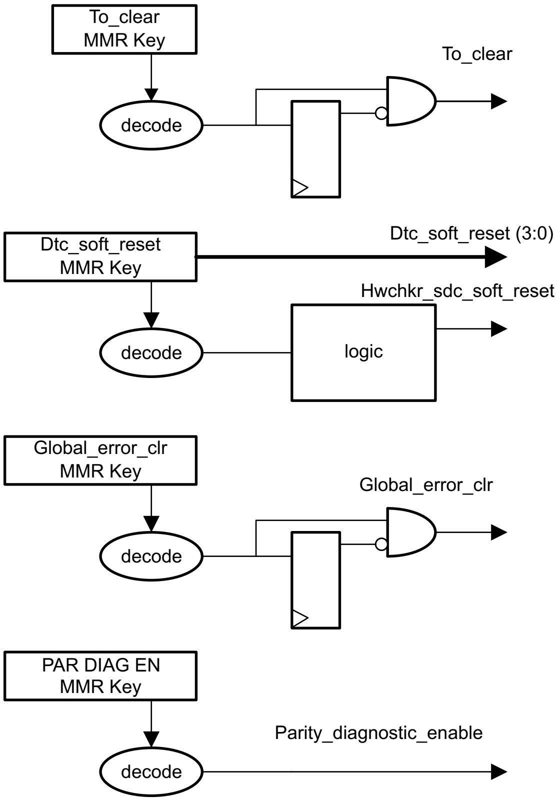
3.2.2.1 Interconnect Timeout Clearing Control Key

When the threshold compare block triggers a time out error, the ESM will be notified and can interrupt the main CPU. The interconnect hardware checker real-time counter needs to be reset to 0 in order to restart properly. Section 3.3 has recommendations on how you should react in this case. You can clear all the real time counter values inside interconnect hardware checker. This is necessary to restart the real time counter.

3.2.3 SCM Control Block

Figure 3-4 shows a block diagram of the SCM.

Figure 3-4. SCM Control Block



3.2.3.1 Control Key to invert Parity Polarity for Interconnect Hardware Checker Parity Detection Diagnostic

The interconnect receives parity bits associated with input control and address signals and does the parity checking. The interconnect also generates parity bits for corresponding control and address signals. To test the parity checking logic, the SCM can invert the parity polarity bit to the interconnect to purposely creates a fail or pass parity checking condition.

Section 3.3 has recommendations on how you should test the parity detection logic inside interconnect hardware checker.

3.2.3.2 Global Error Clearing Control Key

Interconnect subsystem triggers global error in case of any of the following errors happen:

- Parity checking error on any bus master.
- Arbitration error.
- Protocol conversion error.
- Self-test fail in self-test diagnostic mode.

A global error from interconnect subsystem can result in non-recoverable condition for the device. It is recommended that user issues a global error clear by writing 0xA to the GLOBAL_ERR_CLR of the SCMCNTRL register in conjunction to system reset.

3.2.3.3 Interconnect Hardware Checker Self-test

Interconnect hardware checker performs four major diagnostic checks on interconnect:

- Arbitration
- Timeout
- Protocol conversion
- Parity on control / address signals

Thus, it is necessary to be able to do self-test of interconnect hardware checker logic whenever you decide at appropriate time in the application control loop. The self-test logic will create normal and erroneous transaction from each master to each slave according to the bus connection matrix to verify that the hardware checker properly functioning. See Section 3.3.2 for detail on how to start self-test.

3.3 How to Use SCM

3.3.1 How to Check the Parity Compare Logic

Interconnect has associated parity bits for control and address bit of the communication bus. Parity check is done for all control and address input. Parity generation is done for all control and address output.

For fail safety reason, parity checking logic needs to be tested at your choice of time in their control loop.

To enable the parity detection test, you should switch to privilege mode and write 0xA to SCMCNTRL[27:24] control register. This will invert the parity polarity and testing for only one cycle. The SCM module will reset the control key back to 0x5 once it triggers an inversion parity polarity to interconnect hardware checker. Since parity polarity is inverted only inside interconnect, the interconnect will flag parity error for input control and address signals. The interconnect also output an inverted polarity for output control and address signals. Thus, master and slave IP connected to interconnect could potentially generate parity error as well. This way, the corresponding parity detection logic in master and slave IP can be tested at the same time. You should clear all parity error status bits residing in master IP, slave IP, or interconnect status registers.

Note that the hardware only does parity inversion check in one cycle so that it does not block out CPU access to Flash and RAM on subsequence cycle.

3.3.2 How to Initiate Self-test Sequence

It is necessary to be able to do self-test of the interconnect hardware checker logic to detect residual faults when you decide at appropriate time in the application control loop. The self-test logic will create normal and erroneous transaction from each master to each slave according to the bus connection matrix to verify that the hardware checker and the interconnect functioning properly.

To initiate the self-test sequence, you should switch to privilege mode.

1. Software needs to ensure that MASK_SOFT_RESET control bit of the interconnect self-test control register (Interconnect SDC MMR offset at 0xFA00_0000[0]) is 0.
2. Software needs to ensure that GCLK1 is still running.
3. Software needs to ensure that all bus master connecting to interconnect should stop sending new transaction to interconnect. The hardware will make sure that all outstanding transaction will complete.
4. Software writes to SCM control register bit field DTC_SOFT_RESET a key value: 0xA to initiate self-test.
5. CPU0 and CPU1 must execute WFI instruction.
   a. At this point, the hardware will ensure that there is no outstanding transaction inside interconnect and will trigger self-test.
   b. While hardware checker self-test is ongoing, the CPU will be held in reset and released until self-test completes
6. Once self-test completes, CPU will boot up from 0x0 again and you need to read interconnect diagnostic register to inspect for any error detected during self-test. Refer to device technical reference manual for base address.

3.3.3 How to Configure Timeout Check

The threshold compare block takes the real time counters (command request to command accepted and command request to command response) from each IA of the interconnect hardware checker module and compare against the corresponding threshold value in SCM every cycle. If any IA comparison fails, the SCM module will update the corresponding status bit in SCMIAERR0STAT and SCMIAERR1STAT registers. SCMIAERR0STAT logs the time out error for command request to command accepted. SCMIAERR1STAT logs the time out error for command request to command response. Any status bit set in these two status registers will trigger an error event to ESM (Error Signaling Module) and will not trigger again until cleared by CPU.

You should configure the SCMTHRESHOLD control register to setup the command transaction request to command transaction acceptance threshold as well as command transaction request to command transaction response threshold. It is recommended that you use the default reset value of decimal 1024 (400h) for the SCMTHRESHOLD control registers. However, you can change this values depending on application depending on the number of IA and TA required by the interconnect.

When threshold compare block triggers a time out error, the error will be sent to the ESM module resulting in an interrupt exception to the CPU.

When interrupted, it is recommended that you read the SCMIAERR0STAT and SCMIAERR1STAT to find out which master or slave having the time out issue and clear the real time counter inside interconnect. Then, issue a retry on the transaction

1. If the retry is successful, you can resume operation.
2. If the retry fails because time out still occurs, you should trigger a self-test to check for any issue of interconnect. If self-test fails or time out error still occurs after passing self-test, you should try to shut down the system in a safe way. In the case that interconnect has issue and blocking access to Flash or RAM, ESM pin action can not be reset thus external monitoring ASIC will be notified.

To clear real time counters inside SCR, you should switch to privilege mode and write Ah to the SCMCNTRL[3:0] control register. The SCM module will reset the control key back to 5h once it triggers a clear command to interconnect hardware checker. The interconnect hardware checker real time counter needs to be reset to 0 in order to restart properly.

3.4 SCM Registers

The SCM registers are listed in Table 3-1 . Each register begins on a word boundary. The registers support 8-, 16-, and 32-bit accesses. The address offset is specified from the base address of FFFF 0A00h.

Registers are accessed through a dedicated MMR interface. Support only read, write, and write non-posted. Read and write are always returning response status. A write to reserved bits has no effect.

If there is soft error or any other event that results in an unsupported command such as readlink-write conditional or broadcast bus transactions, the MMR interface will return with response bus error for unsupported command. Software should check for valid address and whether the target is in low power mode or not prior to issue a retry access.

Table 3-1. SCM Registers

Offset	Acronym	Register Description	Section
00h	SCMREVID	SCM REVID Register	Section 3.4.1
04h	SCMCNTRL	SCM Control Register	Section 3.4.2
08h	SCMTHRESHOLD	SCM Compare Threshold Counter Register	Section 3.4.3
10h	SCMIAERR0STAT	SCM Initiator Error0 Status Register	Section 3.4.4
14h	SCMIAERR1STAT	SCM Initiator Error1 Status Register	Section 3.4.5
18h	SCMIASTAT	SCM Initiator Active Status Register	Section 3.4.6
20h	SCMTASTAT	SCM Target Active Status Register	Section 3.4.7

3.4.1 SCM REVID Register (SCMREVID)

Figure 3-5. SCM REVID Register (SCMREVID) [offset = 00h]

31 30	29 28	27 16
SCHEME	Reserved	FUNC
R-1	R-0	R-A0Bh

15 11	10 8	7 6	5 0
RTL	MAJOR	CUSTOM	MINOR
R-0	R-0	R-0	R-2h

LEGEND: R = Read only; -*n* = value after synchronous reset on system reset

Table 3-2. SCM REVID Register (SCMREVID) Field Descriptions

Bit	Field	Value	Description
31-30	SCHEME	1	Identification scheme.
29-28	Reserved	0	Reserved. Reads return 0.
27-16	FUNC	A0Bh	Indicates functionally equivalent module family.
15-11	RTL	0	RTL version number.
10-8	MAJOR	0	Major revision number.
7-6	CUSTOM	0	Indicates device-specific implementation.
5-0	MINOR	2h	Minor revision number.

3.4.2 SCM Control Register (SCMCNTRL)

Figure 3-6. SCM Control Register (SCMCNTRL) [offset = 04h]

31 28	27 24	23 20	19 16
Reserved	PAR DIAG EN	Reserved	GLOBAL_ERROR_CLR
R-0	R/WP-5h	R-0	R/WP-5h

15 12	11 8	7 4	3 0
Reserved	DTC_SOFT_RESET	Reserved	TO_CLEAR
R-0	R/WP-5h	R-0	R/WP-5h

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privileged mode only; -*n* = value after synchronous reset on system reset

Table 3-3. SCM Control Register (SCMCNTRL) Field Descriptions

Bit	Field	Value	Description
31-28	Reserved	0	Reserved. Reads return 0.
27-24	PAR DIAG EN		Sticky key write values of Ah. Writing Ah sends out an active-high “pulse” to the SCR and resets the key back to 5h.
			Read:
		5h	Sticky key.
		All other values	Reserved
			Write in Privilege:
		Ah	Parity diagnostic enable.
		All other values	Reserved
23-20	Reserved	0	Reserved. Reads return 0.
19-16	GLOBAL_ERROR_CLR		Clear global error in interconnect. Writing Ah sends out a clear pulse to the SCR and resets the key back to 5h.
			Read:
		5h	Sticky key.
		All other values	Reserved
			Write in Privilege:
		Ah	Enable global error clear.
		All other values	Reserved
15-12	Reserved	0	Reserved. Reads return 0.
11-8	DTC_SOFT_RESET		Diagnostic self-test error enable. Writing Ah forces the SCM to initiate self-test sequence. The hardware will reset the key back to 0x5 whenever self-test is completed.
			Read:
		5h	Sticky key.
		All other values	Reserved
			Write in Privilege:
		Ah	Enable sequence to start interconnect self-test.
		All other values	Reserved
7-4	Reserved	0	Reserved. Reads return 0.
3-0	TO_CLEAR		Clear real time counters inside SCR. Writing Ah sends out a clear pulse to the SCR and resets the key back to 5h.
			Read:
		5h	Sticky key.
		All other values	Reserved
			Write in Privilege:
		Ah	Enable global error clear.
		All other values	Reserved

3.4.3 SCM Compare Threshold Counter Register (SCMTHRESHOLD)

Figure 3-7. SCM Compare Threshold Counter Register (SCMTHRESHOLD) [offset = 08h]

31	16
REQ2RESPONSE_MAX	
R/WP-400h	

15	0
REQ2ACCEPT_MAX	
R/WP-400h	

LEGEND: R/W = Read/Write; WP = Write in privileged mode only; -*n* = value after synchronous reset on system reset

Table 3-4. SCM Compare Threshold Counter Register (SCMTHRESHOLD) Field Descriptions

Bit	Field	Value	Description
31-16	REQ2RESPONSE_MAX	0-FFFFh	Request to Response Threshold values. You need to configure the maximum threshold values for request to response timeout. Reset values equals to the values of REQ2RESP_RST generic parameter. Read: Values of counter. Write in Privilege: Values of counter.
15-0	REQ2ACCEPT_MAX	0-FFFFh	Request to Accept Threshold values. You need to configure the maximum threshold values for request to accept timeout. Reset values equals to the values of REQ2ACCEPT_RST generic parameter. Read: Values of counter. Write in Privilege: Values of counter.

3.4.4 SCM Initiator Error0 Status Register (SCMIAERR0STAT)

Figure 3-8. SCM Initiator Error0 Status Register (SCMIAERR0STAT) [offset = 10h]

31	8
Reserved	
R-0	

7	6	5	4	3	2	1	0
R2A7	R2A6	R2A5	R2A4	R2A3	R2A2	R2A1	R2A0
R/W1CP-0	R/W1CP-0	R/W1CP-0	R/W1CP-0	R/W1CP-0	R/W1CP-0	R/W1CP-0	R/W1CP-0

LEGEND: R/W = Read/Write; W1CP = Write 1 to clear in privilege mode only; -*n* = value after synchronous reset by power-on reset

Table 3-5. SCM Initiator Error0 Status Register (SCMIAERR0STAT) Field Descriptions

Bit	Field	Value	Description
31-8	Reserved	0	Reserved. Read returns 0.
7-0	R2A*n*		Request to Acceptance Timeout Error happens on IA*n*. Each bit *n* corresponds to request to accept time out error occurred for each IA. Refer to Interconnect chapter of the TRM for specific mapping of each R2A*n* to a particular IP.
			Read:
		0	No request to accept time out error happens on IA*n*.
		1	Request to accept time out error happens on IA*n*.
			Write in Privilege:
		0	No effect.
		1	Clear this flag bit.

3.4.5 SCM Initiator Error1 Status Register (SCMIAERR1STAT)

Figure 3-9. SCM Initiator Error1 Status Register (SCMIAERR1STAT) [offset = 14h]

31	8
Reserved	
R-0	

7	6	5	4	3	2	1	0
R2R7	R2R6	R2R5	R2R4	R2R3	R2R2	R2R1	R2R0
R/W1CP-0	R/W1CP-0	R/W1CP-0	R/W1CP-0	R/W1CP-0	R/W1CP-0	R/W1CP-0	R/W1CP-0

LEGEND: R/W = Read/Write; W1CP = Write 1 to clear in privilege mode only; -*n* = value after synchronous reset by power-on reset

Table 3-6. SCM Initiator Error1 Status Register (SCMIAERR1STAT) Field Descriptions

Bit	Field	Value	Description
31-8	Reserved	0	Reserved. Read returns 0.
7-0	R2R*n*		Request to Response Timeout Error happens on IA*n*. Each bit *n* corresponds to request to response time out error occurred for each IA.. Refer to Interconnect chapter of the TRM
			Read:
		0	No request to response time out error happens on IA*n*.
		1	Request to response time out error happens on IA*n*.
			Write in Privilege:
		0	No effect.
		1	Clear this flag bit.

3.4.6 SCM Initiator Active Status Register (SCMIASTAT)

Figure 3-10. SCM Initiator Active Status Register (SCMIASTAT) [offset = 18h]

31	16
Reserved	
R-0	

15	14	13	12	11	10	9	8
Reserved		IAST13	IAST12	IAST11	IAST10	IAST9	IAST8
R-0		R-0	R-0	R-0	R-0	R-0	R-0

7	6	5	4	3	2	1	0
IAST7	IAST6	IAST5	IAST4	IAST3	IAST2	IAST1	IAST0
R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0

LEGEND: R = Read only; -*n* = value after synchronous reset by system reset

Table 3-7. SCM Initiator Active Status Register (SCMIASTAT) Field Descriptions

Bit	Field	Value	Description
31-14	Reserved	0	Reserved. Read returns 0.
13-0	IAST*n*		IA (Initiator Agent) Status. Each bit *n* indicates that there is a pending transaction on the corresponding IA*n*. Refer to Interconnect chapter of the TRM for mapping of master port to the SCMIASTAT register bit.
		0	No pending transaction in IA*n*.
		1	Pending transaction in IA*n*.

3.4.7 SCM Target Active Status Register (SCMTASTAT)

Figure 3-11. SCM Target Active Status Register (SCMTASTAT) [offset = 20h]

31	16
Reserved	
R-0	

15	14	13	12	11	10	9	8
Reserved		TAST13	TAST12	TAST11	TAST10	TAST9	TAST8
R-0		R-0	R-0	R-0	R-0	R-0	R-0

7	6	5	4	3	2	1	0
TAST7	TAST6	TAST5	TAST4	TAST3	TAST2	TAST1	TAST0
R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0

LEGEND: R = Read only; -*n* = value after synchronous reset by system reset

Table 3-8. SCM Target Active Status Register (SCMTASTAT) Field Descriptions

Bit	Field	Value	Description
31-14	Reserved	0	Reserved. Read returns 0.
13-0	TAST*n*		TA (Target Agent) Status. Each bit *n* indicates that there is a pending transaction on the corresponding TA*n*.Refer to Interconnect chapter of the TRM for mapping of slave port to the SCMTASTAT register bit.
		0	No pending transaction in TA*n*.
		1	Pending transaction in TA*n*.

Interconnect

This chapter describes the two interconnects in the microcontroller.

Topic **Page**

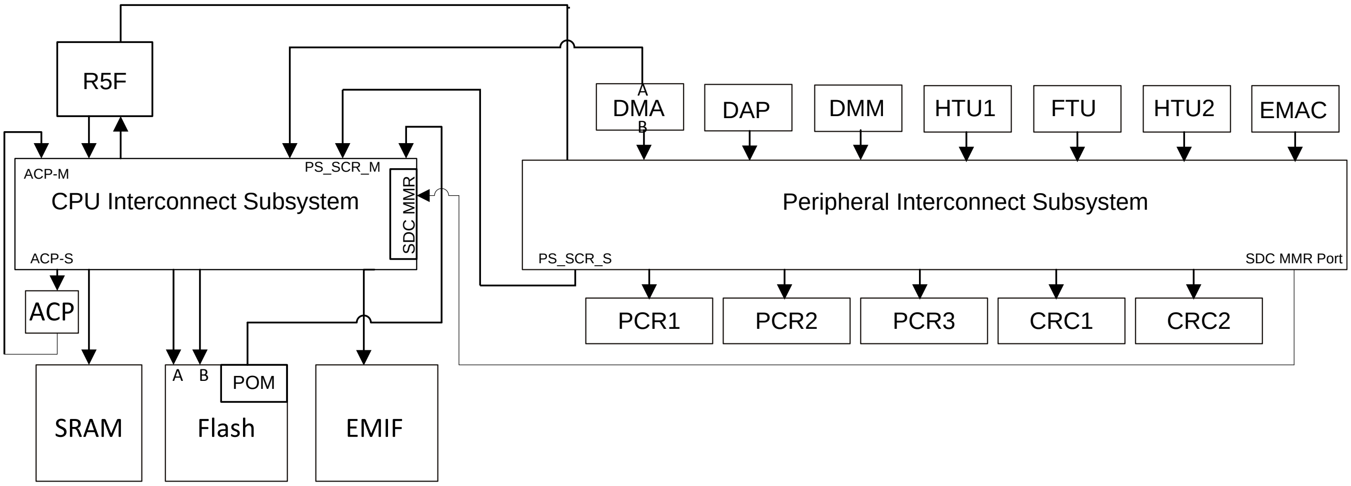
4.1 Overview

The interconnect is a bus matrix which interconnects the CPU cores, System DMA, other bus masters and device specific slaves within the microcontroller. There are two interconnects in the microcontroller: the CPU Interconnect Subsystem and the Peripheral Interconnect Subsystem. The interconnects direct the access requests by the masters by providing decoding, arbitration, and routing of the requests to the various slaves.

4.1.1 Block Diagram

Figure 4-1 is a block diagram of the Interconnects implemented in this family of microcontrollers.

Figure 4-1. Interconnect Block Diagram



4.2 Peripheral Interconnect Subsystem

There are masters and slaves connected to the Peripheral Interconnect Subsystem. The Peripheral Interconnect Subsystem is not a full cross-bar. Not all masters can access to all slaves. Table 4-1 lists the implemented point-to-point connections between the masters and slaves.

Table 4-1. Bus Master / Slave Connectivity for Peripheral Interconnect Subsystem

Masters	Master ID to PCRx	Access Mode	Slaves on Peripheral Interconnect Subsystem						
			CRC1	CRC2	PCR1	PCR2	PCR3	PS_SCR_S	SDC MMR Port
CPU Read/Write	0	User/Privilege	Yes	Yes	Yes	Yes	Yes	No	Yes
DMA Port B	2	User	Yes	Yes	Yes	Yes	Yes	No	No
HTU1	3	Privilege	No	No	No	No	No	Yes	No
HTU2	4	Privilege	No	No	No	No	No	Yes	No
FTU	5	User	No	No	No	No	No	Yes	No
DMM	7	User	Yes	Yes	Yes	Yes	Yes	Yes	No
DAP	9	Privilege	Yes	Yes	Yes	Yes	Yes	Yes	No
EMAC	10	User	No	No	No	Yes	Yes	Yes	No

4.2.1 Accessing PCRx and CRCx Slave

System peripherals can be accessed via the PCR1 slave port. User peripherals can be accessed via either the PCR2 or PCR3 slave ports. Refer to the datasheet for information on what peripherals are available through each PCR. Peripheral Central Resource (PCR) is responsible to further decode the slave address to select the desired peripheral.

There are two CRC modules implemented in the device. Both are direct slaves to the Peripheral Interconnect Subsystem.

4.2.2 Accessing SDC MMR Port Slave

Safety Diagnostic Controller (SDC) MMR Port is a slave to the Peripheral Interconnect Subsystem to access the safety diagnostic related control and status registers of the CPU Interconnect Subsystem. Table 4-2 lists the CPU Interconnect Subsystem SDC register bit field mapping.

Table 4-2. CPU Interconnect Subsystem SDC Register Bit Field Mapping

Register Name	Bit 0	Bit 1	Bit 2	Bit 3	Bit 4	Bit 5	Bit 6	Remark
ERR_GENERIC_PARITY	PS_SCR_M	POM	DMA_PORTA	CPU AXI-M	Reserved	ACP-M	Reserved	Each bit indicates the transaction processing block inside the interconnect corresponding to the master that is detected by the interconnect checker to have a fault. Error related to parity mismatch in the incoming address.
ERR_UNEXPECTED_TRANS	PS_SCR_M	POM	DMA_PORTA	CPU AXI-M	Reserved	ACP-M	Reserved	Error related to unexpected transaction sent by the master.
ERR_TRANS_ID	PS_SCR_M	POM	DMA_PORTA	CPU AXI-M	Reserved	ACP-M	Reserved	Error related to mismatch on the transaction ID.
ERR_TRANS_SIGNATURE	PS_SCR_M	POM	DMA_PORTA	CPU AXI-M	Reserved	ACP-M	Reserved	Error related to mismatch on the transaction signature.
ERR_TRANS_TYPE	PS_SCR_M	POM	DMA_PORTA	CPU AXI-M	Reserved	ACP-M	Reserved	Error related to mismatch on the transaction type.
ERR_USER_PARITY	PS_SCR_M	POM	DMA_PORTA	CPU AXI-M	Reserved	ACP-M	Reserved	Error related to mismatch on the parity.
SERR_UNEXPECTED_MID	L2 SRAM Wrapper	L2 Flash Wrapper Port A	L2 Flash Wrapper Port B	EMIF	Reserved	CPU AXi-S	ACP-S	Each bit indicates the transaction processing block inside the interconnect corresponding to the slave that is detected by the interconnect checker to have a fault. Error related to mismatch on the master ID.
SERR_ADDR_DECODE	L2 SRAM Wrapper	L2 Flash Wrapper Port A	L2 Flash Wrapper Port B	EMIF	Reserved	CPU AXi-S	ACP-S	Error related to mismatch on the most significant address bits.
SERR_USER_PARITY	L2 SRAM Wrapper	L2 Flash Wrapper Port A	L2 Flash Wrapper Port B	EMIF	Reserved	CPU AXi-S	ACP-S	Error related to mismatch on the parity of the most significant address bits.

4.2.3 Accessing Other Slaves via PS_SCR_S

In order for some of the masters connected to the Peripheral Interconnect Subsystem to access the slaves such as L2 Flash and L2 SRAM in the CPU Interconnect Subsystem, their requests are first funneled into the PS_SCR_S slave where it then becomes a master on the CPU Interconnect Subsystem as PS_SCR_M. The request appearing on the PS_SCR_M is then decoded and routed to the intended slave by the CPU Interconnect Subsystem.

4.3 CPU Interconnect Subsystem

The masters and slaves are connected to the CPU Interconnect Subsystem. The CPU Interconnect Subsystem is not a full cross-bar. Not all masters can access to all slaves. Table 4-3 lists the implemented point to point connections between the masters and slaves. What is also unique to the CPU Interconnect Subsystem is that the interconnect and all the masters and slaves that connect to it constitute one safety island where all transactions to and from the masters and slaves are protected on the data path by ECC. Address and control signals on all transactions are protected by parity. In addition, the CPU Interconnect Subsystem contains a built-in hardware Safety Diagnostic Checker on each master and slave interface where it constantly monitors the integrity of traffics between the masters and slaves. The CPU Interconnect Subsystem also has a self-test capability that when enabled will inject test stimulus onto each master and slave interface and diagnose the interconnect itself.

Table 4-3. Bus Master / Slave Connectivity for CPU Interconnect Subsystem

Masters	Access Mode	Slaves on CPU Interconnect Subsystem					
		L2 Flash Port A	L2 Flash Port B	L2 SRAM	CPU AXI-S	EMIF	ACP-S
CPU Read	User/Privilege	Yes	Yes	Yes	Yes	Yes	No
CPU Write	User/Privilege	No	No	Yes	Yes	Yes	No
DMA Port A	User	No	Yes	Yes	No	Yes	Yes
POM	User	No	No	Yes	No	Yes	Yes
PS_SCR_M	See [1]	No	Yes	Yes	No	Yes	Yes
ACP_M	See [1]	No	No	Yes	No	No	No

[1] The access mode for PS_SCR_M depends on which master on the peripheral side (HTU1, HTU2, FTU, DMM, DAP, and EMAC), see Table 4-1, is accessing the memories on the CPU side. The ACP_M access mode reflects the PS_SCR_M access mode.

4.3.1 Slave Accessing

4.3.1.1 Accessing L2 Flash Slave

There are two flash slave ports which allow possible parallel requests by the masters to different flash banks at the same time. There are two flash banks of 2Mbytes each implemented in the device. It is possible for CPU0 to access one flash bank via Flash PortA while DMA accesses to the other flash bank via Flash PortB.

4.3.1.2 Accessing L2 SRAM Slave

In order for the DMA PortA, POM and PS_SCR_M to access the L2 SRAM, their requests are first funneled into ACP-S slave port. Accelerated Coherency Port (ACP) is a hardware which provides memory coherency checking between each CPU in the Cortex-R5 group and an external master. Accesses made by the DMA PortA, POM and PS_SCR_M are first checked by the ACP coherency hardware to see if the write data is already in the CPU's data cache. When a write from the DMA PortA, POM and PS_SCR_M appears on the ACP slave, the ACP records some information about it and forward the write transactions to the L2 SRAM on the ACP-M master port. When the memory system sends the write response on the ACP-M master port, the ACP records the response and recalls if the transaction was coherent. If the transaction is not coherent, the ACP forwards the response to the bus master on the ACP-S slave port. If the transaction is coherent, the ACP first sends coherency maintenance operations to the CPU's data cache controller for the addresses spanned by the write transaction, and wait until the cache controller has acknowledged that all necessary coherency maintenance operations have been carried out to forward the write response to the ACP-S slave port. CPUs have direct access to the L2 SRAM.

4.3.1.3 Accessing EMIF Slave

All bus masters on the CPU Interconnect Subsystem have a point to point connection to the EMIF slave without going through ACP for coherency check. Coherency maintenance on the EMIF between the CPU and other masters will need to be handled by software.

4.3.1.4 Accessing Cache Memories

Both the instruction and data caches of theCPU are memory mapped in the device and can be accessed via the AXI-S slave port. Only the CPU core has point to point connection to the AXI-S slave port.

4.3.2 ECC Generation and Evaluation

CPU core contains the built-in ECC generation and evaluation logic for its AXI interface. Therefore, CPU will generate the ECC checksum along with its write data. The write data and the corresponding ECC checksum are transported by the interconnect to the selected slave such as L2 SRAM. When CPU core performs a read from a slave, the slave returns the data and the corresponding ECC checksum. Upon receiving the data and the ECC checksum, the CPU will evaluate the integrity of the data by performing the ECC check. ECC errors detected on the CPU's AXI interface are exported by the CPU to its event bus output. The error signals if enabled and the corresponding error addresses are first routed to the Error Profiling Controller (EPC) module. EPC is used to record different single bit error addresses in a Content Addressable Memory (CAM). The main purpose of the EPC module is to enable the system to tolerate a certain amount of ECC correctable errors on the same address repeated in the memory system with minimal runtime overhead. If an ECC error is generated on a repeating address, the EPC will not raise an error to ESM module. This tolerance avoids the application to handle the same error when the code is in a repeating loop. See EPC chapter for more information.

DMA PortA and PS_SCR_M masters do not have built-in ECC generation and evaluation logic. Therefore, the CPU Interconnect Subsystem contain a standalone ECC generation and evaluation logic for each DMA PortA and PS_SCR_M master. Write transactions initiated by the DMA PortA and PS_SCR_M masters are first treated by the ECC block to generate the ECC checksum before transporting to the final destination. For read transactions, the data and ECC checksum returned by the slaves will pass through the ECC block for data integrity evaluation.

ECC errors detected are also routed to the Error Profiling Controller (EPC) module. In order for the standalone ECC block to assert the error signals to the EPC, the error enable key must be first set in the IP1ECCERREN register of the SYS2 module.

NOTE: To enable error signal assertion to the ESM for ECC errors detected for DMA, the application must write 0xA to the IP1_ECC_KEY bits. To enable error signal assertion to the ESM for ECC errors detected for PS_SCR_M, the application must write 0xA to the IP2_ECC_KEY bits.

4.3.3 Safety Diagnostic Checker

For each master and slave interface in the CPU Interconnect Subsystem, there is a runtime Safety Diagnostic Checker. The hardware checker continuously watches transactions flowing through the interconnect and ensuring they are non corrupted at all time. If a mismatch is detected between an ongoing transaction and the expected transaction flow then an error is asserted to the ESM Group 1. Types of errors are recorded in the SDC MMR registers. See Section 4.4 for all the registers. Once an error is detected and the error type is logged, the application will clear the runtime diagnostic errors by writing an 0xA key to the GLOBAL_ERROR_CLR bits of the SCMCNTRL register in the SCR Control Module (SCM). See the SCM Chapter for more information.

4.3.4 Interconnect Self-test

CPU Interconnect Subsystem can be put into self-test. When in self-test, the self-test logic will apply test stimulus to each master and slave interface. If an error is detected, the type of error for the corresponding interface is logged. An error is asserted to ESM Group 3 if the self-test does not complete successfully.

NOTE: Application must only launch CPU Interconnect Subsystem self-test when there are no bus transactions from any masters including the CPU cores. While in self-test, the interconnect can not service any requests. Bus master requests can be lost or corrupted. It is recommend that the self-test is only exercised as part of the device initialization before any master is setup by the CPU.

To launch the self-test, the applicable must follow the below sequence:

1. Write 0xA key to the DTC_ERROR_RESET bits of the SCMCNTRL register in the SCM module.
2. CPU executes WFI instruction to put itself in idle state. The start of self-test is gated by the idle state of the CPU.
3. When both step 1 and 2 are met, the self-test will start. While self-test is on-going, the CPU cores is forced into reset. Note that reset is only held to the CPU cores while the rest of the system is not.
4. When self-test is complete, the DTC_ERROR_RESET bits is automatically reverted back to 0x5 as the reset value.
5. After the self-test is complete, a reset is applied to the CPU Interconnect Subsystem for 16 HCLK cycles. During this time, the CPU is also held in reset.
6. After the interconnect and the CPU comes out of the reset, normal code execution can then start. CPU can check the self-test status by reading the NT_OK bit and the PT_OK bit of the SDC_STATUS register. These two bits indicate if the negative test and positive self-test sequence have passed. In addition, if the self-test has failed, the error is asserted to the ESM module.

4.3.5 Interconnect Timeout

The CPU Interconnect Subsystem contains timeout counters to count the amount of time it is taking for a master request to be accepted by the slave and also to count the amount of time it takes from an accepted request to the slave response. There are two separate counters per master interface. When either the request-to-accept counter or the accept-to-response counter expires by the slave, a timeout error is asserted to the ESM. The counter threshold value beyond which the timeout error will be generated is programmable in the SCM module. When a timeout happens to an interface, the request-to-accept timeout error is captured in the SCM's SCMIAERR0STAT register and the accept-to-response timeout error is captured in the SCMIAERR1STAT. See Table 4-4 for the mapping between each interface to each bit field. Application needs to write 0xA key to the TO_CLEAR bits of the SCMCNTRL register to reset the timeout logic inside the CPU Interconnect Subsystem as part of the error handling in the ISR.

4.3.6 Interconnect Runtime Status

Both the CPU Interconnect Subsystem and the Peripheral Interconnect Subsystem will output its status on each master and slave interface to the SCM indicating if the interface is currently active. The status are captured in the SCMIASTAT register for the master interfaces and SCMTASTAT for the slave interfaces. See Table 4-4 for the mapping between each interface to each bit field.

Table 4-4. SCM Register Bit Mapping

Register	Bit 0	Bit 1	Bit 2	Bit 3	Bit 4	Bit 5	Bit 6	Bit 7	Remark
SCMIAERR0 STAT	PS_SCR _M	POM	DMA Port A	Reserved	Reserved	CPU AXI-M Read	CPU AXI-M Write	ACP-M	Each bit indicates the transaction processing block inside the interconnect corresponding to the master that is detected by the interconnect checker to have a fault. A timeout error when the time the request is issued by the master until the time the request is accepted by the slave has expired
SCMIAERR1 STAT	PS_SCR _M	POM	DMA Port A	Reserved	Reserved	CPU AXI-M Read	CPU AXI-M Write	ACP-M	A timeout error when the time the request is accepted by the slave until the time the request is responded by the slave has expired
SCMIASTAT	PS_SCR _M	POM	DMA Port A	Reserved	Reserved	CPU AXI-M Read	CPU AXI-M Write	ACP-M	Each bit indicates that there is still pending transactions for the corresponding master to be processed by the interconnect
	Bit 8	**Bit 9**	**Bit 10**	**Bit 11**	**Bit 12**	**Bit 13**			
	DMA PortB	HTUx/ FTU	DAP/ DMM	Ethernet	CPU PP-AXI	Reserved			
SCMTASTAT	L2 RAM	L2 Flash Port B	L2 Flash Port A	EMIF	Reserved	CPU AXI-S	ACP-S	PS_SCR _S	Each bit indicates that there is still pending transactions for the corresponding slave to be processed by the interconnect
	Bit 8	**Bit 9**	**Bit 10**	**Bit 11**	**Bit 12**	**Bit 13**			
	PCR1	PCR2	PCR3	CRC1	CRC2	SDC MMR			

4.4 SDC MMR Registers

Table 4-5 lists the Safety Diagnostic Checker registers. The registers support only 32-bit reads. The offset is relative to the base address. The base address for the registers is FA00 0000h.

Table 4-5. SDC MMR Registers

Offset	Acronym	Register Description	Section
0h	SDC_STATUS	SDC Status Register	Section 4.4.1
4h	SDC_CONTROL	SDC Control Register	Section 4.4.2
8h	ERR_GENERIC_PARITY	Error Generic Parity Register	Section 4.4.3
Ch	ERR_UNEXPECTED_TRANS	Error Unexpected Transaction Register	Section 4.4.4
10h	ERR_TRANS_ID	Error Transaction ID Register	Section 4.4.5
14h	ERR_TRANS_SIGNATURE	Error Transaction Signature Register	Section 4.4.6
18h	ERR_TRANS_TYPE	Error Transaction Type Register	Section 4.4.7
1Ch	ERR_USER_PARITY	Error User Parity Register	Section 4.4.8
20h	SERR_UNEXPECTED_MID	Slave Error Unexpected Master ID register	Section 4.4.9
24h	SERR_ADDR_DECODE	Slave Error Address Decode Register	Section 4.4.10
28h	SERR_USER_PARITY	Slave Error User Parity Register	Section 4.4.11

4.4.1 SDC Status Register (SDC_STATUS)

Figure 4-2. SDC Status Register (SDC_STATUS) (offset = 00h)

31	16
Reserved	
R-0	

15	5	4	3	2	1	0
		GLOBAL_ERROR	NT_OK	NT_RUN	PT_OK	PT_RUN
R-0		R-0	R-0	R-0	R-0	R-0

LEGEND: R = Read only; -*n* = value after reset

Table 4-6. SDC Status Register (SDC_STATUS) Field Descriptions

Bit	Field	Value	Description
31-5	Reserved	0	Reads return 0 and writes have no effect.
4	GLOBAL_ERROR		This bit indicates that one safety diagnostic checker has asserted an error input that is captured in error log registers located at address offset from 0x08 to 0x28.
		0	No error is detected by any checker.
		1	Error is detected by one checker. To find out the type of error from which checker, read the error log registers located at address offset from 0x08 to 0x28.
3	NT_OK		Negative test OK status for self-test.
		0	Negative test has failed.
		1	Negative test has passed.
2	NT_RUN		Negative test on-going status.
		0	Negative test has ended.
		1	Negative test is on-going.
1	PT_OK		Positive test OK status for self-test.
		0	Positive test has failed.
		1	Positive test has passed.
0	PT_RUN		Positive test on-going status.
		0	Positive test has ended.
		1	Positive test is on-going.

4.4.2 SDC Control Register (SDC_CONTROL)

Figure 4-3. SDC Control Register (SDC_STATUS) (offset = 04h)

31	16
Reserved	
R-0	

15	1	0
Reserved		MASK_SOFT_RESET
R-0		R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 4-7. SDC Control Register (SDC_CONTROL) Field Descriptions

Bit	Field	Value	Description
31-1	Reserved	0	Reads return 0 and writes have no effect.
0	MASK_SOFT_RESET		This bit enables the self-test sequence to be launched by the SCM (SCR Control Module) module. You should always keep this bit cleared.
		0	Enable SCM to launch self-test on the interconnect.
		1	Disable SCM to launch self-test on the interconnect.

4.4.3 Error Generic Parity Register (ERR_GENERIC_PARITY)

Figure 4-4. Error Generic Parity Register (ERR_GENERIC_PARITY) (offset = 08h)

31	16
Reserved	
R-0	

15	6	5	0
Reserved		ERR_GENERIC_PARITY	
R-0		R-0	

LEGEND: R = Read only; -*n* = value after reset

Table 4-8. Error Generic Parity Register (ERR_GENERIC_PARITY) Field Descriptions

Bit	Field	Value	Description
31-6	Reserved	0	Reads return 0 and writes have no effect.
5-0	ERR_GENERIC_PARITY		Error related to parity mismatch in the higher order bits of the incoming address getting transmitted across interconnect incorrectly. When set, each bit indicates the transaction processing block inside the interconnect corresponding to the master is detected by the interconnect checker to have a fault. bit 0: PS_SCR_M master bit 1: POM master bit 2: DMA PortA master bit 3: Reserved bit 4: Cortex-R5F CPU master. bit 5: ACP-M master

4.4.4 Error Unexpected Transaction Register (ERR_UNEXPECTED_TRANS)

Figure 4-5. Error Unexpected Transaction Register (ERR_UNEXPECTED_TRANS) (offset = 0Ch)

31	16
Reserved	
R-0	

15	6	5	0
Reserved		ERR_UNEXPECTED_TRANS	
R-0		R-0	

LEGEND: R = Read only; -*n* = value after reset

Table 4-9. Error Unexpected Transaction Register (ERR_UNEXPECTED_TRANS) Field Descriptions

Bit	Field	Value	Description
31-6	Reserved	0	Reads return 0 and writes have no effect.
5-0	ERR_UNEXPECTED_TRANS		Error related to unexpected transaction sent by the master. When set, each bit indicates the transaction processing block inside the interconnect corresponding to the master is detected by the interconnect checker to have a fault. bit 0: PS_SCR_M master bit 1: POM master bit 2: DMA PortA master bit 3: Reserved bit 4: Cortex-R5F CPU master. bit 5: ACP-M master

4.4.5 Error Transaction ID Register (ERR_TRANS_ID)

Figure 4-6. Error Transaction ID Register (ERR_TRANS_ID) (offset = 10h)

31	16
Reserved	
R-0	

15	6	5	0
Reserved		ERR_TRANS_ID	
R-0		R-0	

LEGEND: R = Read only; -*n* = value after reset

Table 4-10. Error Transaction ID Register (ERR_TRANS_ID) Field Descriptions

Bit	Field	Value	Description
31-6	Reserved	0	Reads return 0 and writes have no effect.
5-0	ERR_TRANS_ID		Error related to mismatch on the transaction ID. When set, each bit indicates the transaction processing block inside the interconnect corresponding to the master is detected by the interconnect checker to have a fault. bit 0: PS_SCR_M master bit 1: POM master bit 2: DMA PortA master bit 3: Reserved bit 4: Cortex-R5F CPU master. bit 5: ACP-M master

4.4.6 Error Transaction Signature Register (ERR_TRANS_SIGNATURE)

Figure 4-7. Error Transaction Signature Register (ERR_TRANS_SIGNATURE) (offset = 14h)

31	16
Reserved	
R-0	

15	6	5	0
Reserved		ERR_TRANS_SIGNATURE	
R-0		R-0	

LEGEND: R = Read only; -*n* = value after reset

Table 4-11. Error Transaction Signature Register (ERR_TRANS_SIGNATURE) Field Descriptions

Bit	Field	Value	Description
31-6	Reserved	0	Reads return 0 and writes have no effect.
5-0	ERR_TRANS_ SIGNATURE		Error related to mismatch on the transaction signature. The transaction signature is a value computed using the lower bits of the address, number of bytes and the byte enables of the transaction. The signature calculated by the master is sent to the decoded slave where the signature is computed again and compared to the original signature. When set, each bit indicates the transaction processing block inside the interconnect corresponding to the master is detected by the interconnect checker to have a fault. bit 0: PS_SCR_M master bit 1: POM master bit 2: DMA PortA master bit 3: Reserved bit 4: Cortex-R5F CPU master. bit 5: ACP-M master

4.4.7 Error Transaction Type Register (ERR_TRANS_TYPE)

Figure 4-8. Error Transaction Type Register (ERR_TRANS_TYPE) (offset = 18h)

31	16
Reserved	
R-0	

15	6	5	0
Reserved		ERR_TRANS_TYPE	
R-0		R-0	

LEGEND: R = Read only; -*n* = value after reset

Table 4-12. Error Transaction Type Register (ERR_TRANS_TYPE) Field Descriptions

Bit	Field	Value	Description
31-6	Reserved	0	Reads return 0 and writes have no effect.
5-0	ERR_TRANS_TYPE		Error related to mismatch on the transaction type. When set, each bit indicates the transaction processing block inside the interconnect corresponding to the master is detected by the interconnect checker to have a fault. bit 0: PS_SCR_M master bit 1: POM master bit 2: DMA PortA master bit 3: Reserved bit 4: Cortex-R5F CPU master. bit 5: ACP-M master

4.4.8 Error User Parity Register (ERR_USER_PARITY)

Figure 4-9. Error User Parity Register (ERR_USER_PARITY) (offset = 1Ch)

31-16
Reserved
R-0

15-6	5-0
Reserved	ERR_USER_PARITY
R-0	R-0

LEGEND: R = Read only; -*n* = value after reset

Table 4-13. Error User Parity Register (ERR_USER_PARITY) Field Descriptions

Bit	Field	Value	Description
31-6	Reserved	0	Reads return 0 and writes have no effect.
5-0	ERR_USER_PARITY		Error related to mismatch on the parity. When set, each bit indicates the transaction processing block inside the interconnect corresponding to the master is detected by the interconnect checker to have a fault. bit 0: PS_SCR_M master bit 1: POM master bit 2: DMA PortA master bit 3: Reserved bit 4: Cortex-R5F CPU master. bit 5: ACP-M master

4.4.9 Slave Error Unexpected Master ID Register (SERR_UNEXPECTED_MID)

Figure 4-10. Slave Error Unexpected Master ID Register (SERR_UNEXPECTED_MID) (offset = 20h)

31-16
Reserved
R-0

15-7	6-0
Reserved	SERR_UNEXPECTED_MID
R-0	R-0

LEGEND: R = Read only; -*n* = value after reset

Table 4-14. Slave Error Unexpected Master ID Register (SERR_UNEXPECTED_MID) Field Descriptions

Bit	Field	Value	Description
31-7	Reserved	0	Reads return 0 and writes have no effect.
6-0	SERR_UNEXPECTED_MID		Error related to mismatch on the master ID. When set, each bit indicates the transaction processing block inside the interconnect corresponding to the slave that is detected by the interconnect checker to have a fault. bit 0: L2 SRAM slave bit 1: L2 Flash PortB slave bit 2: L2 Flash PortA slave bit 3: EMIF slave bit 4: Reserved bit 5: Cortex-R5F CPU AXI slave bit 6: ACP-S slave

4.4.10 Slave Error Address Decode Register (SERR_ADDR_DECODE)

Figure 4-11. Slave Error Address Decode Register (SERR_ADDR_DECODE) (offset = 24h)

31	16
Reserved	
R-0	

15	7	6	0
Reserved		SERR_ADDR_DECODE	
R-0		R-0	

LEGEND: R = Read only; -*n* = value after reset

Table 4-15. Slave Error Address Decode Register (SERR_ADDR_DECODED) Field Descriptions

Bit	Field	Value	Description
31-7	Reserved	0	Reads return 0 and writes have no effect.
6-0	SERR_ADDR_ DECODE		Error related to mismatch on the most-significant address bits. When set, each bit indicates the transaction processing block inside the interconnect corresponding to the slave that is detected by the interconnect checker to have a fault. bit 0: L2 SRAM slave bit 1: L2 Flash PortB slave bit 2: L2 Flash PortA slave bit 3: EMIF slave bit 4: Reserved bit 5: Cortex-R5F CPU AXI slave bit 6: ACP-S slave

4.4.11 Slave Error User Parity Register (SERR_USER_PARITY)

Figure 4-12. Slave Error User Parity Register (SERR_USER_PARITY) (offset = 28h)

31	16
Reserved	
R-0	

15	7	6	0
Reserved		SERR_USER_PARITY	
R-0		R-0	

LEGEND: R = Read only; -*n* = value after reset

Table 4-16. Slave Error User Parity Register (SERR_USER_PARITYID) Field Descriptions

Bit	Field	Value	Description
31-7	Reserved	0	Reads return 0 and writes have no effect.
6-0	SERR_USER_PARITY		Error related to mismatch on the parity on the response signals for the slave. When set, each bit indicates the transaction processing block inside the interconnect corresponding to the slave that is detected by the interconnect checker to have a fault. bit 0: L2 SRAM slave bit 1: L2 Flash PortB slave bit 2: L2 Flash PortA slave bit 3: EMIF slave bit 4: Reserved bit 5: Cortex-R5F CPU AXI slave bit 6: ACP-S slave

Power Management Module (PMM)

This chapter describes the power management module (PMM).

Topic **Page**

5.1 Overview

The microcontroller is part of the family of microcontrollers from Texas Instruments for safety-critical applications. Several functions are implemented on this microcontroller targeted towards varied applications. The core logic is divided into several domains that can be independently turned on or off based on the application's requirements. Turning off a domain has the effect to only turn off the clocks into the domain. Dynamic current is virtually reduced to zero. Leakage will remain the same as in this device no physical power switches are implemented to isolate a domain from its core supply.

This chapter describes the Power Management Module (PMM). The PMM provides memory-mapped registers that control the states of the supported power domains. The PMM includes interfaces to the Power Mode Controller (PMC) and the Power State Controller (PSCON). The PMC and PSCON control the power up/down sequence of each power domain.

5.1.1 Features

The main features of the PMM implemented on the microcontroller are:

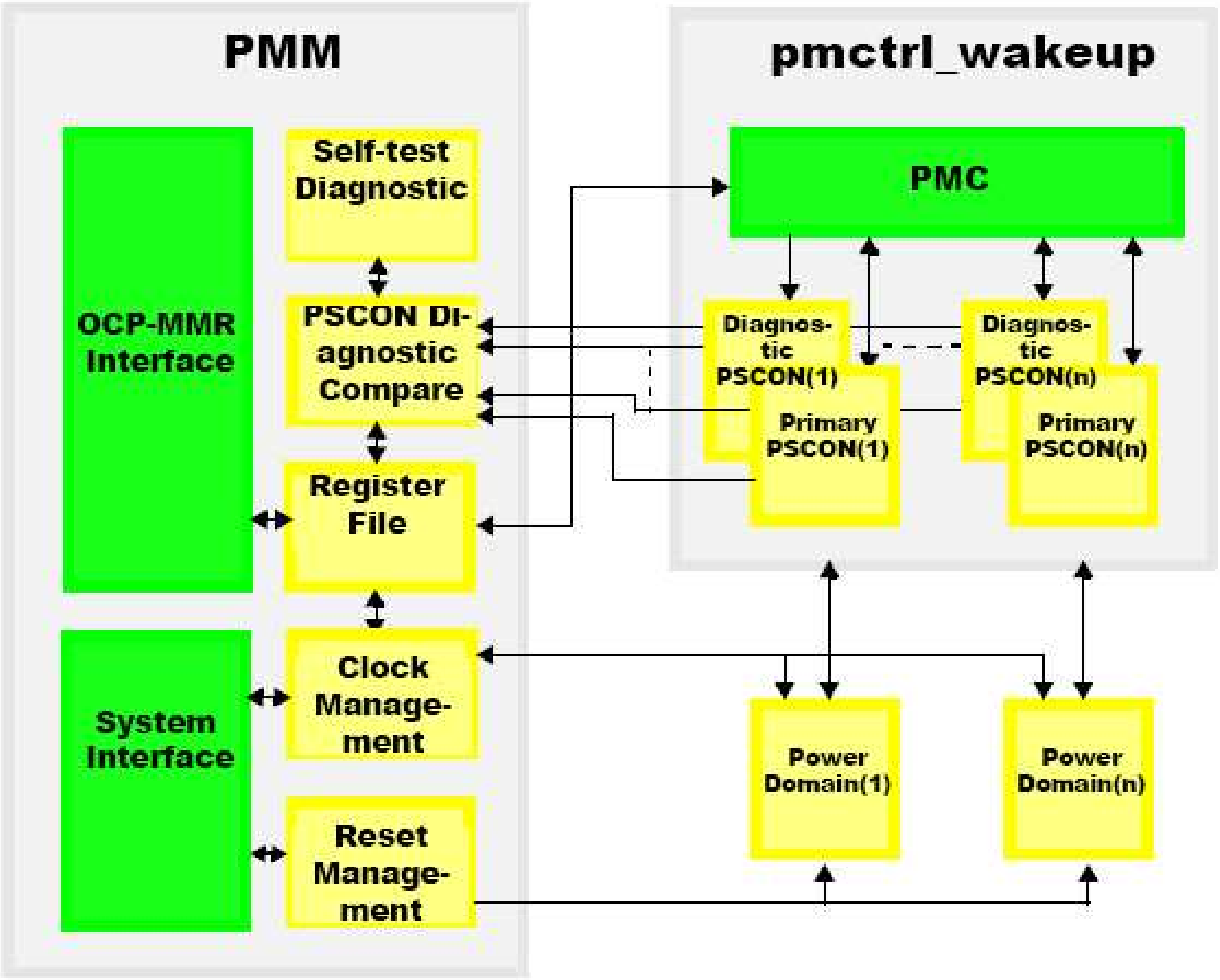
- Supports 6 logic power domains: PD1, PD2, PD3, PD4, PD5 and PD6
- Allows configurable default states for each power domain
- Allows each power domain to be permanently disabled
- Manages the clocks for each power domain
- Manages the resets to each power domain
- Includes failsafe compare logic to continuously monitor the states of each power domain
- Supports diagnostic and self-test logic to validate failsafe compare logic

5.1.2 Block Diagram

PMM consists of several key components:

- Register interface – the PMM control registers are mapped to the device memory space and start at address 0xFFFF0000.
- System Interface – the PMM receives the clocks, resets, errors and all other control signals through this interface.
- PSCON Diagnostic Compare – this block compares the outputs of each primary PSCON and the respective diagnostic PSCON implemented for failsafe safety.
- Self-Test Diagnostic – this block contains the logic to place the PSCON diagnostic compare block in a self-test mode in order to test the failsafe feature.
- Clock management – the PMM provides independent clock gating and handshaking controls for each power domain and also generates the clock domains for each power domain.
- Reset Management – the PMM provides independent reset signals for each power domain.
- Power Mode Controller (PMC) – The PMC is a finite state machine that controls the power sequence from one power mode to another. A power mode is the states of all power domains at a given time.
- Power State Controller (PSCON) – The PSCON is a finite state machine that controls the power sequence of a power domain from one state to another. Each power domain is controlled by one dedicated PSCON.
- Power Domain – In this device, a power domain is a group of logic and/or memories which shares the common control inputs.

Figure 5-1. PMM Block Diagram



5.2 Power Domains

Figure 5-2 shows the core domains implemented on the microcontroller.

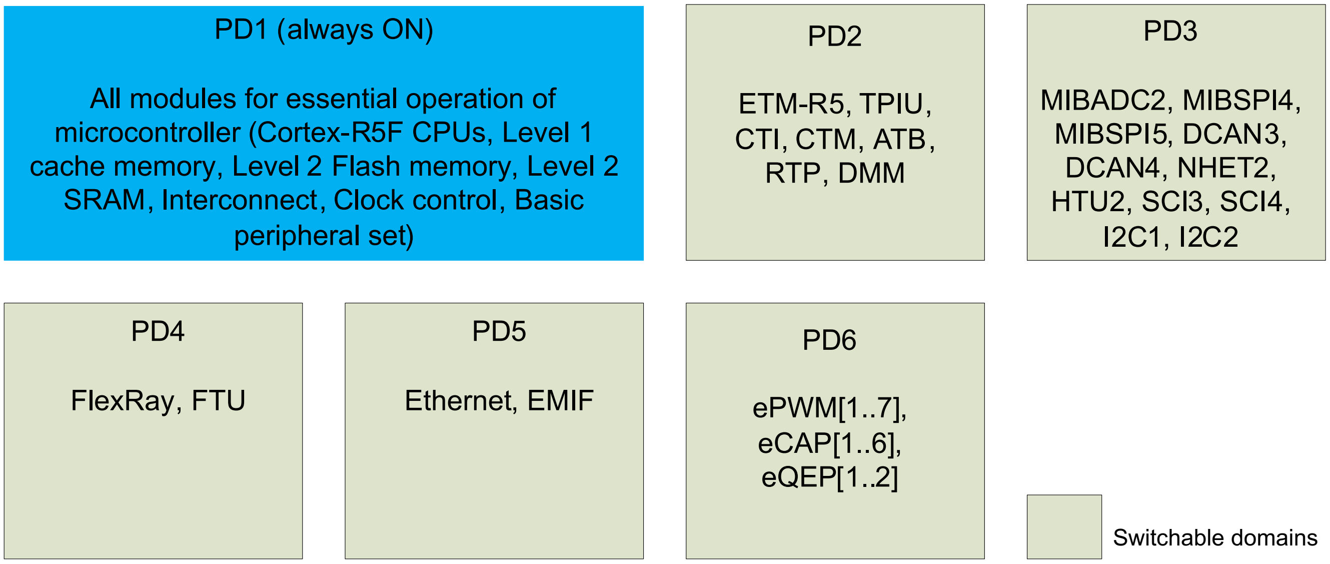
This device has 6 separate core power domains:

- PD1 is an always-ON domain and is not controlled by PMM. It contains the CPU as well as other principal modules and the interconnect required for operation of the microcontroller. This domain also includes the level 1 cache memory and the level 2 flash memory and SRAM. The PD1 can operate on its own even when all the other core power domains are turned off by the PMM. Note that all I/Os are in this always-ON domain as well.

Core power domains PD2 through PD6 are controlled by the PMM.

- PD2 contains logic related to debug, instrumentation and trace such as the Embedded Trace Macrocell (ETM-R5), RAM Trace Port (RTP), and Data Modification Module (DMM) components.
- PD3 contains some additional peripheral modules as an enhanced configuration over and above the peripheral set available in PD1. These include a second High-End Timer (N2HET2) with its dedicated transfer unit (HTU2), a second Analog-to-Digital Converter (ADC2), two Serial Communication Interfaces (SCI3 and SCI4), two Inter-Integrated Circuit controllers (I2C1 and I2C2), two Controller Area Network controller (DCAN3 and DCAN4), and two Multi-buffer Serial Peripheral Interface module (MibSPI4 and MibSPI5).
- PD4 contains the FlexRay controller and its dedicated transfer unit (FTU).
- PD5 contains the Ethernet controller (EMAC), the External Memory Interface (EMIF), as well as some components of the interconnect fabric required by these modules.
- PD6 contains seven Enhanced Pulse Width Modulation modules (ePWM), two Quadrature Encoder Pulse modules (nQEP), and six Enhanced Capture modules (eCAP).

Figure 5-2. Core Power Domains


PD1 (always ON)
All modules for essential operation of microcontroller (Cortex-R5F CPUs, Level 1 cache memory, Level 2 Flash memory, Level 2 SRAM, Interconnect, Clock control, Basic peripheral set)
PD2
ETM-R5, TPIU, CTI, CTM, ATB, RTP, DMM
PD3
MIBADC2, MIBSPI4, MIBSPI5, DCAN3, DCAN4, NHET2, HTU2, SCI3, SCI4, I2C1, I2C2
PD4
FlexRay, FTU
PD5
Ethernet, EMIF
PD6
ePWM[1..7], eCAP[1..6], eQEP[1..2]
Switchable domains

5.3 PMM Operation

It is important to understand some fundamental concepts beforehand.

5.3.1 Power Domain State

Each core power domain can be in one of three states: Active, Idle, or Off.

In the ***Active*** state, a power domain is fully powered with normal supply voltage.

In the ***Idle*** state, all clocks to a power domain are turned off (driven low). The supply voltage is still maintained at the normal level.

In this device, the ***Off*** state is equivalent to the ***Idle*** state in terms of power saving. Users can still from a programmer's model perspective put a power domain into the Off state as if the power domain can be physically turned off.

NOTE: This device does not implement power switches to physically isolate the power domain from its power supply. Putting a power domain into the Off state has no effect to remove leakage power. Power domains in this device are group of modules surrounded by the isolation cells. Isolation cells are placed at the outputs of the power domains. When a power domain is put into Off state, the isolation cells are enabled and force inactive states on the output signals. PMM and the PSCONs do not know the physical implementation of the power domains. The logic to control the transition from one power state to another will behave the same as if the power domains can be physically turned off.

5.3.2 Default Power Domain State

The default state of each power domain, except for PD1, is controlled by TI during production testing via programmation of individual bits within the reset configuration word in the TI-OTP sector of flash bank 0. This allows each power domain to default to either the active state or the off state.

5.3.3 Disabling a Power Domain Permanently

TI can also permanently disable any power domain, except for PD1. This is also controlled by programmation of individual bits within the reset configuration word in the TI-OTP sector of flash bank 0.

5.3.4 Changing Power Domain State

A domain can only change state when commanded by the application. Each domain has an associated 4-bit key to define the intended power state. When the correct key is programmed, the PMM initiates the sequence to transition that domain to the commanded state.

5.3.4.1 Turning a Power Domain Off

It is necessary to turn off all clocks going to a power domain before that domain can be powered down. PMM contains the hardware interlocks to handle this. Each power domain has an associated memory-mapped register which allows the application to turn off clocks to that power domain.

Steps to power down a domain with logic – PD2, PD3, PD4, PD5, PD6:

1. Write to the PDCLK_DISx register to disable all clocks to the power domain.
2. Write 0xA to the LOGICPDPWRCTRL0 register to power down the domain.
3. Poll for LOGICPDPWRSTATx to become “00”. The power domain is now powered down.

5.3.4.2 Turning a Power Domain On

A power domain can be turned on by writing the correct key to the LOGICPDON register. PMM will automatically restart the clocks to the power domain once the ***Active*** power state is restored if the “automatic clock enable upon wake up” option is selected. If this option is not selected, the application can turn on clocks to the power domain by clearing the PDCLK_DIS register manually. The application must poll the DOMAINISON register to ensure that the power has been fully restored before enabling the clocks.

5.3.5 Reset Management

PMM handles the reset sequence for each power domain. When a power domain is turned on from an off state, the PMM will reset the power domain to ensure that all logic begins in its default reset state.

PMM generates nPORRST (power-on reset), nRST (system reset), nPRST (peripheral reset), and nTRST (test / debug logic reset) for each domain.

5.3.6 Diagnostic Power State Controller (PSCON)

Each power domain state is controlled by a primary PSCON. There is a second PSCON as well for each power domain. This is the diagnostic PSCON. All power management inputs to a power domain are controlled only by the primary PSCON. All power management outputs from the power domain are fed back to both the primary and the diagnostic PSCON.

The PMM commands both the PSCON identically so that they are always in a lock-step operating mode. A dedicated compare unit checks the outputs of the two PSCON modules on every cycle.

5.3.7 PSCON Compare Block

The diagnostic compare block can operate in one of four modes.

5.3.7.1 Lock-Step Mode

This is the default mode of operation of the PSCON compare block. The PSCON diagnostic compare block compares the outputs from the two PSCONs on every cycle. Any mismatch in the PSCON outputs is indicated as a PSCON compare error. This error signal is mapped to the Error Signaling Module’s (ESM) Group1 channel 38. The application can define the response to this error.

5.3.7.2 Self-Test Mode

A self-test mechanism is provided to check the PSCON compare logic for faults. The compare error signal output is disabled in self-test mode. The PSCON diagnostic compare block generates two types of patterns during self-test mode: compare match test followed by compare mismatch test. During the self-test, each test pattern is applied on both PSCON signal ports of the PSCON diagnostic compare block and then is clocked for one cycle. The duration of the self-test is 24 cycles. Any detected fault is indicated as a self-test error, mapped to ESM group1 channel 39. If no fault is detected, the self-test complete flag is set.

The application can poll for this flag to be set and then switch the mode of the PSCON compare block back to lock-step mode by writing to the mode key register.

NOTE: PSCON operation when compare block is in self-test mode

When the PSCON compare block is in its self-test mode, both PSCONs continue to function normally. However, there is no comparison done on the PSCON outputs.

Compare match test:

An identical vector is applied to both input ports at the same time, thereby expecting a compare match. If the compare unit produces a mismatch then the self-test error flag is set and the self-test error signal is generated. The compare match test is terminated if a compare mismatch is detected. The compare match test takes 4 cycles to complete when the test passes.

Compare mismatch test:

A vector with all 1's is applied to the PSCON diagnostic compare block’s primary input port and the same input is also applied to the secondary input port but with one bit flipped starting from bit position 0. The unequal vectors should cause the PSCON diagnostic compare block to generate a compare mismatch at bit position 0. In case a mismatch is not detected, a self-test error is indicated. This compare mismatch test algorithm is repeated until every single bit position is verified on both PSCON signal ports.

5.3.7.3 Error-Forcing Mode

This mode is designed specifically to ensure that the error signal output from the PSCON compare block is not stuck inactive. In this mode, a test pattern is applied to the PSCON related inputs of the compare logic to force an error. The application can clear the flag for ESM group1 channel 38 once the error is flagged. If the ESM group1 channel 38 flag does not get set, this indicates that the PSCON compare error signal is stuck inactive and cannot be relied upon to detect a PSCON mismatch.

5.3.7.4 Self-Test Error-Forcing Mode

In this mode, an error is forced so that the self-test error output from the PSCON compare block is activated. The application can clear the flag for ESM group1 channel 39 once the error is flagged. If the ESM group1 channel 39 flag does not get set, this indicates that the PSCON compare block self-test error signal is stuck inactive and there is no self-test mechanism available for the PSCON compare block.

5.3.7.5 PMM Operation During CPU Halt Debug Mode

No compare errors are generated when the CPU is halted in debug mode, regardless of the mode of the diagnostic compare block. No status flags are updated in this mode. Normal operation of the compare block is resumed once the CPU exits the debug mode.

5.4 PMM Registers

Table 5-1 lists the control registers in the PMM module. The registers support 8-, 16-, and 32-bit accesses. The address offset is specified from the base address of FFFF 0000h. Any access to an unimplemented location within the PMM register frame will generate a bus error that results in an Abort exception.

Table 5-1. PMM Registers

Offset	Acronym	Register Description	Section
00h	LOGICPDPWRCTRL0	Logic Power Domain Control Register 0	Section 5.4.1
04h	LOGICPDPWRCTRL1	Logic Power Domain Control Register 1	Section 5.4.2
20h	PDCLKDIS	Power Domain Clock Disable Register	Section 5.4.3
24h	PDCLKDISSET	Power Domain Clock Disable Set Register	Section 5.4.4
28h	PDCLKDISCLR	Power Domain Clock Disable Clear Register	Section 5.4.5
40h	LOGICPDPWRSTAT0	Logic Power Domain PD2 Power Status Register	Section 5.4.6
44h	LOGICPDPWRSTAT1	Logic Power Domain PD3 Power Status Register	Section 5.4.7
48h	LOGICPDPWRSTAT2	Logic Power Domain PD4 Power Status Register	Section 5.4.8
4Ch	LOGICPDPWRSTAT3	Logic Power Domain PD5 Power Status Register	Section 5.4.9
50h	LOGICPDPWRSTAT4	Logic Power Domain PD6 Power Status Register	Section 5.4.10
A0h	GLOBALCTRL1	Global Control Register 1	Section 5.4.11
A8h	GLOBALSTAT	Global Status Register	Section 5.4.12
ACh	PRCKEYREG	PSCON Diagnostic Compare Key Register	Section 5.4.13
B0h	LPDDCSTAT1	LogicPD PSCON Diagnostic Compare Status Register 1	Section 5.4.14
B4h	LPDDCSTAT2	LogicPD PSCON Diagnostic Compare Status Register 2	Section 5.4.15
C0h	ISODIAGSTAT	Isolation Diagnostic Status Register	Section 5.4.16

5.4.1 Logic Power Domain Control Register (LOGICPDPWRCTRL0)

The default values of the control fields are determined by the device reset configuration word stored in the TI-OTP region of flash bank 0.

Figure 5-3. Logic Power Domain Control Register (LOGICPDPWRCTRL0) (offset = 00h)

31 28	27 24	23 20	19 16
Reserved	LOGICPDON0	Reserved	LOGICPDON1
R-0	R/WP-n	R-0	R/WP-n

15 12	11 8	7 4	3 0
Reserved	LOGICPDON2	Reserved	LOGICPDON3
R-0	R/WP-n	R-0	R/WP-n

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privileged mode only; -*n* = value after reset

Table 5-2. Logic Power Domain Control Register (LOGICPDPWRCTRL0) Field Descriptions

Bit	Field	Value	Description
31-28	Reserved	0	Reads return 0. Writes have no effect.
27-24	LOGICPDON0		Read in User and Privileged Mode. Write in Privileged Mode only.
		Ah	Read: Power domain PD2 is in OFF state. Write: Power domain PD2 is commanded to switch to OFF state.
		9h	Reserved
		Any other value	Read: Power domain PD2 is in Active state. Write: Power domain PD2 is commanded to switch to Active state.
23-20	Reserved	0	Reads return 0. Writes have no effect.
19-16	LOGICPDON1		Read in User and Privileged Mode. Write in Privileged Mode only.
		Ah	Read: Power domain PD3 is in OFF state. Write: Power domain PD3 is commanded to switch to OFF state.
		9h	Reserved
		Any other value	Read: Power domain PD3 is in Active state. Write: Power domain PD3 is commanded to switch to Active state.
15-12	Reserved	0	Reads return 0. Writes have no effect.
11-8	LOGICPDON2		Read in User and Privileged Mode. Write in Privileged Mode only.
		Ah	Read: Power domain PD4 is in OFF state. Write: Power domain PD4 is commanded to switch to OFF state.
		9h	Reserved
		Any other value	Read: Power domain PD4 is in Active state. Write: Power domain PD4 is commanded to switch to Active state.
7-4	Reserved	0	Reads return 0. Writes have no effect.
3-0	LOGICPDON3		Read in User and Privileged Mode. Write in Privileged Mode only.
		Ah	Read: Power domain PD5 is in OFF state. Write: Power domain PD5 is commanded to switch to OFF state.
		9h	Reserved
		Any other value	Read: Power domain PD5 is in Active state. Write: Power domain PD5 is commanded to switch to Active state.

5.4.2 Logic Power Domain Control Register (LOGICPDPWRCTRL1)

The default values of the control fields are determined by the device reset configuration word stored in the TI-OTP region of flash bank 0.

Figure 5-4. Logic Power Domain Control Register (LOGICPDPWRCTRL1) (offset = 04h)

31 28	27 24	23 16
Reserved	LOGICPDON4	Reserved
R-0	R/WP-n	R-0

15 0
Reserved
R-0

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privileged mode only; -*n* = value after reset

Table 5-3. Logic Power Domain Control Register (LOGICPDPWRCTRL1) Field Descriptions

Bit	Field	Value	Description
31-28	Reserved	0	Reads return 0. Writes have no effect.
27-24	LOGICPDON4	Ah	Read in User and Privileged Mode. Write in Privileged Mode only. Read: Power domain PD6 is in OFF state. Write: Power domain PD6 is commanded to switch to OFF state.
		9h	Reserved
		Any other value	Read: Power domain PD6 is in Active state. Write: Power domain PD6 is commanded to switch to Active state.
23-0	Reserved	0	Reads return 0. Writes have no effect.

5.4.3 Power Domain Clock Disable Register (PDCLKDISREG)

The default values of the control fields are determined by the device reset configuration word stored in the TI-OTP region of flash bank 0.

Figure 5-5. Power Domain Clock Disable Register (PDCLKDISREG) (offset = 20h)

31	8
Reserved	
R-0	

7	5	4	3	2	1	0
Reserved		PDCLK_DIS[4]	PDCLK_DIS[3]	PDCLK_DIS[2]	PDCLK_DIS[1]	PDCLK_DIS[0]
R-0		R/WP-n	R/WP-n	R/WP-n	R/WP-n	R/WP-n

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privileged mode only; -*n* = value after reset

Table 5-4. Power Domain Clock Disable Register (PDCLKDISREG) Field Descriptions

Bit	Field	Value	Description
31-5	Reserved	0	Reads return 0. Writes have no effect.
4	PDCLK_DIS[4]		Read in User and Privileged Mode returns the current value of PDCLK_DIS[4]. Write in Privileged Mode only.
		0	Enable clocks to logic power domain PD6.
		1	Disable clocks to logic power domain PD6.
3	PDCLK_DIS[3]		Read in User and Privileged Mode returns the current value of PDCLK_DIS[3]. Write in Privileged Mode only.
		0	Enable clocks to logic power domain PD5.
		1	Disable clocks to logic power domain PD5.
2	PDCLK_DIS[2]		Read in User and Privileged Mode returns the current value of PDCLK_DIS[2]. Write in Privileged Mode only
		0	Enable clocks to logic power domain PD4.
		1	Disable clocks to logic power domain PD4.
1	PDCLK_DIS[1]		Read in User and Privileged Mode returns the current value of PDCLK_DIS[1]. Write in Privileged Mode only.
		0	Enable clocks to logic power domain PD3.
		1	Disable clocks to logic power domain PD3.
0	PDCLK_DIS[0]		Read in User and Privileged Mode returns the current value of PDCLK_DIS[0]. Write in Privileged Mode only.
		0	Enable clocks to logic power domain PD2.
		1	Disable clocks to logic power domain PD2.

5.4.4 Power Domain Clock Disable Set Register (PDCLKDISSETREG)

The default values of the control fields are determined by the device reset configuration word stored in the TI-OTP region of flash bank 0.

Figure 5-6. Power Domain Clock Disable Set Register (PDCLKDISSETREG) (offset = 24h)

31	8
Reserved	
R-0	

7	5	4	3	2	1	0
Reserved		PDCLK_DISSET[4]	PDCLK_DISSET[3]	PDCLK_DISSET[2]	PDCLK_DISSET[1]	PDCLK_DISSET[0]
R-0		R/WP-n	R/WP-n	R/WP-n	R/WP-n	R/WP-n

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privileged mode only; -*n* = value after reset

Table 5-5. Power Domain Clock Disable Set Register (PDCLKDISSETREG) Field Descriptions

Bit	Field	Value	Description
31-5	Reserved	0	Reads return 0. Writes have no effect.
4	PDCLK_DISSET[4]		Read in User and Privileged Mode returns the current value of PDCLK_DISSET[4]. Write in Privileged Mode only.
		0	No effect to state of clocks to power domain PD6.
		1	Disable clocks to logic power domain PD6.
3	PDCLK_DISSET[3]		Read in User and Privileged Mode returns the current value of PDCLK_DISSET[3]. Write in Privileged Mode only.
		0	No effect to state of clocks to power domain PD5.
		1	Disable clocks to logic power domain PD5.
2	PDCLK_DISSET[2]		Read in User and Privileged Mode returns the current value of PDCLK_DISSET[2]. Write in Privileged Mode only.
		0	No effect to state of clocks to power domain PD4.
		1	Disable clocks to logic power domain PD4.
1	PDCLK_DISSET[1]		Read in User and Privileged Mode returns the current value of PDCLK_DISSET[1]. Write in Privileged Mode only.
		0	No effect to state of clocks to power domain PD3.
		1	Disable clocks to logic power domain PD3.
0	PDCLK_DISSET[0]		Read in User and Privileged Mode returns the current value of PDCLK_DISSET[0]. Write in Privileged Mode only.
		0	No effect to state of clocks to power domain PD2.
		1	Disable clocks to logic power domain PD2.

5.4.5 Power Domain Clock Disable Clear Register (PDCLKDISCLRREG)

The default values of the control fields are determined by the device reset configuration word stored in the TI-OTP region of flash bank 0.

Figure 5-7. Power Domain Clock Disable Clear Register (PDCLKDISCLRREG) (offset = 28h)

31–8
Reserved
R-0

7–5	4	3	2	1	0
Reserved	PDCLK_DISCLR[4]	PDCLK_DISCLR[3]	PDCLK_DISCLR[2]	PDCLK_DISCLR[1]	PDCLK_DISCLR[0]
R-0	R/WP-n	R/WP-n	R/WP-n	R/WP-n	R/WP-n

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privileged mode only; -*n* = value after reset

Table 5-6. Power Domain Clock Disable Clear Register (PDCLKDISCLRREG) Field Descriptions

Bit	Field	Value	Description
31-5	Reserved	0	Reads return 0. Writes have no effect.
4	PDCLK_DISCLR[4]		Read in User and Privileged Mode returns the current value of PDCLK_DIS[4]. Write in Privileged Mode only.
		0	No effect to state of clocks to power domain PD6.
		1	Enable clocks to logic power domain PD6.
3	PDCLK_DISCLR[3]		Read in User and Privileged Mode returns the current value of PDCLK_DIS[3]. Write in Privileged Mode only.
		0	No effect to state of clocks to power domain PD5.
		1	Enable clocks to logic power domain PD5.
2	PDCLK_DISCLR[2]		Read in User and Privileged Mode returns the current value of PDCLK_DIS[2]. Write in Privileged Mode only.
		0	No effect to state of clocks to power domain PD4.
		1	Enable clocks to logic power domain PD4.
1	PDCLK_DISCLR[1]		Read in User and Privileged Mode returns the current value of PDCLK_DIS[1]. Write in Privileged Mode only.
		0	No effect to state of clocks to power domain PD3.
		1	Enable clocks to logic power domain PD3.
0	PDCLK_DISCLR[0]		Read in User and Privileged Mode returns the current value of PDCLK_DIS[0]. Write in Privileged Mode only.
		0	No effect to state of clocks to power domain PD2.
		1	Enable clocks to logic power domain PD2.

5.4.6 Logic Power Domain PD2 Power Status Register (LOGICPDPWRSTAT0)

This is a read-only register. All writes are ignored.

The default values of the control fields are determined by the device reset configuration word stored in the TI-OTP region of flash bank 0.

Figure 5-8. Logic Power Domain PD2 Power Status Register (LOGICPDPWRSTAT0) (offset = 40h)

31 - 25	24	23 - 17	16
Reserved	LOGIC IN TRANS0	Reserved	MEM IN TRANS0
R-0	R-n	R-0	R-n

15 - 9	8	7 - 2	1 - 0
Reserved	DOMAIN ON0	Reserved	LOGICPDPWR STAT0
R-0	R-n	R-0	R-n

LEGEND: R = Read only; -*n* = value after reset

Table 5-7. Logic Power Domain PD2 Power Status Register (LOGICPDPWRSTAT0) Field Descriptions

Bit	Field	Value	Description
31-25	Reserved	0	Reads return 0. Writes have no effect.
24	LOGIC IN TRANS0		Logic in transition status for power domain PD2.
			Read in User and Privileged Mode.
		0	Logic in power domain PD2 is in the steady Active or OFF state.
		1	Logic in power domain PD2 is in the process of power-down/up.
13-17	Reserved	0	Reads return 0. Writes have no effect.
16	MEM IN TRANS0		Memory in transition status for power domain PD2.
			Read in User and Privileged Mode.
		0	Memory in power domain PD2 is in the steady Active or OFF state.
		1	Memory in power domain PD2 is in the process of power-down/up.
15-9	Reserved	0	Reads return 0. Writes have no effect.
8	DOMAIN ON0		Current state of power domain PD2. The default value of this field is controlled by the device reset configuration word in the TI-OTP region of flash bank 0.
			Read in User and Privileged Mode.
		0	Power domain PD2 is in the OFF state.
		1	Power domain PD2 is in the Active state.
7-2	Reserved	0	Reads return 0. Writes have no effect.
1-0	LOGICPDPWR STAT0		Logic power domain PD2 power state. The default value of this field is controlled by the device reset configuration word in the TI-OTP region of flash bank 0.
			Read in User and Privileged Mode.
		0	Logic power domain PD2 is switched OFF.
		1h	Logic power domain PD2 is in Idle state.
		2h	Reserved
		3h	Logic power domain PD2 is in Active state.

5.4.7 Logic Power Domain PD3 Power Status Register (LOGICPDPWRSTAT1)

This is a read-only register. All writes are ignored.

The default values of the control fields are determined by the device reset configuration word stored in the TI-OTP region of flash bank 0.

Figure 5-9. Logic Power Domain PD3 Power Status Register (LOGICPDPWRSTAT1) (offset = 44h)

31	25	24	23	17	16
Reserved		LOGIC IN TRANS1	Reserved		MEM IN TRANS1
R-0		R-n	R-0		R-n

15	9	8	7	2	1	0
Reserved		DOMAIN ON1	Reserved		LOGICPDPWR STAT1	
R-0		R-n	R-0		R-n	

LEGEND: R = Read only; -*n* = value after reset

Table 5-8. Logic Power Domain PD3 Power Status Register (LOGICPDPWRSTAT1) Field Descriptions

Bit	Field	Value	Description
31-25	Reserved	0	Reads return 0. Writes have no effect.
24	LOGIC IN TRANS1		Logic in transition status for power domain PD3.
			Read in User and Privileged Mode.
		0	Logic in power domain PD3 is in the steady Active or OFF state.
		1	Logic in power domain PD3 is in the process of power-down/up.
13-17	Reserved	0	Reads return 0. Writes have no effect.
16	MEM IN TRANS1		Memory in transition status for power domain PD3.
			Read in User and Privileged Mode.
		0	Memory in power domain PD3 is in the steady Active or OFF state.
		1	Memory in power domain PD3 is in the process of power-down/up.
15-9	Reserved	0	Reads return 0. Writes have no effect.
8	DOMAIN ON1		Current state of power domain PD3. The default value of this field is controlled by the device reset configuration word in the TI-OTP region of flash bank 0.
			Read in User and Privileged Mode.
		0	Power domain PD3 is in the OFF state.
		1	Power domain PD3 is in the Active state.
7-2	Reserved	0	Reads return 0. Writes have no effect.
1-0	LOGICPDPWR STAT1		Logic power domain PD3 power state. The default value of this field is controlled by the device reset configuration word in the TI-OTP region of flash bank 0.
			Read in User and Privileged Mode.
		0	Logic power domain PD3 is switched OFF.
		1h	Logic power domain PD3 is in Idle state.
		2h	Reserved
		3h	Logic power domain PD3 is in Active state.

5.4.8 Logic Power Domain PD4 Power Status Register (LOGICPDPWRSTAT2)

This is a read-only register. All writes are ignored.

The default values of the control fields are determined by the device reset configuration word stored in the TI-OTP region of flash bank 0.

Figure 5-10. Logic Power Domain PD4 Power Status Register (LOGICPDPWRSTAT2) (offset = 48h)

31-25	24	23-17	16
Reserved	LOGIC IN TRANS2	Reserved	MEM IN TRANS2
R-0	R-n	R-0	R-n

15-9	8	7-2	1-0
Reserved	DOMAIN ON2	Reserved	LOGICPDPWR STAT2
R-0	R-n	R-0	R-n

LEGEND: R = Read only; -*n* = value after reset

Table 5-9. Logic Power Domain PD4 Power Status Register (LOGICPDPWRSTAT2) Field Descriptions

Bit	Field	Value	Description
31-25	Reserved	0	Reads return 0. Writes have no effect.
24	LOGIC IN TRANS2		Logic in transition status for power domain PD4.
			Read in User and Privileged Mode.
		0	Logic in power domain PD4 is in the steady Active or OFF state.
		1	Logic in power domain PD4 is in the process of power-down/up.
13-17	Reserved	0	Reads return 0. Writes have no effect.
16	MEM IN TRANS2		Memory in transition status for power domain PD4.
			Read in User and Privileged Mode.
		0	Memory in power domain PD4 is in the steady Active or OFF state.
		1	Memory in power domain PD4 is in the process of power-down/up.
15-9	Reserved	0	Reads return 0. Writes have no effect.
8	DOMAIN ON2		Current state of power domain PD4. The default value of this field is controlled by the device reset configuration word in the TI-OTP region of flash bank 0.
			Read in User and Privileged Mode.
		0	Power domain PD4 is in the OFF state.
		1	Power domain PD4 is in the Active state.
7-2	Reserved	0	Reads return 0. Writes have no effect.
1-0	LOGICPDPWR STAT2		Logic power domain PD4 power state. The default value of this field is controlled by the device reset configuration word in the TI-OTP region of flash bank 0.
			Read in User and Privileged Mode.
		0	Logic power domain PD4 is switched OFF.
		1h	Logic power domain PD4 is in Idle state.
		2h	Reserved
		3h	Logic power domain PD4 is in Active state.

5.4.9 *Logic Power Domain PD5 Power Status Register (LOGICPDPWRSTAT3)*

This is a read-only register. All writes are ignored.

The default values of the control fields are determined by the device reset configuration word stored in the TI-OTP region of flash bank 0.

Figure 5-11. Logic Power Domain PD5 Power Status Register (LOGICPDPWRSTAT3) (offset = 4Ch)

31 25	24	23 17	16
Reserved	LOGIC IN TRANS3	Reserved	MEM IN TRANS3
R-0	R-n	R-0	R-n

15 9	8	7 2	1 0
Reserved	DOMAIN ON3	Reserved	LOGICPDPWR STAT3
R-0	R-n	R-0	R-n

LEGEND: R = Read only; -*n* = value after reset

Table 5-10. Logic Power Domain PD5 Power Status Register (LOGICPDPWRSTAT3) Field Descriptions

Bit	Field	Value	Description
31-25	Reserved	0	Reads return 0. Writes have no effect.
24	LOGIC IN TRANS3		Logic in transition status for power domain PD5.
			Read in User and Privileged Mode.
		0	Logic in power domain PD5 is in the steady Active or OFF state.
		1	Logic in power domain PD5 is in the process of power-down/up.
13-17	Reserved	0	Reads return 0. Writes have no effect.
16	MEM IN TRANS3		Memory in transition status for power domain PD5.
			Read in User and Privileged Mode.
		0	Memory in power domain PD5 is in the steady Active or OFF state.
		1	Memory in power domain PD5 is in the process of power-down/up.
15-9	Reserved	0	Reads return 0. Writes have no effect.
8	DOMAIN ON3		Current state of power domain PD5. The default value of this field is controlled by the device reset configuration word in the TI-OTP region of flash bank 0.
			Read in User and Privileged Mode.
		0	Power domain PD5 is in the OFF state.
		1	Power domain PD5 is in the Active state.
7-2	Reserved	0	Reads return 0. Writes have no effect.
1-0	LOGICPDPWR STAT3		Logic power domain PD5 power state. The default value of this field is controlled by the device reset configuration word in the TI-OTP region of flash bank 0.
			Read in User and Privileged Mode.
		0	Logic power domain PD5 is switched OFF.
		1h	Logic power domain PD5 is in Idle state.
		2h	Reserved
		3h	Logic power domain PD5 is in Active state.

5.4.10 Logic Power Domain PD6 Power Status Register (LOGICPDPWRSTAT4)

This is a read-only register. All writes are ignored.

The default values of the control fields are determined by the device reset configuration word stored in the TI-OTP region of flash bank 0.

Figure 5-12. Logic Power Domain PD6 Power Status Register (LOGICPDPWRSTAT4) (offset = 50h)

31 25	24	23 17	16
Reserved	LOGIC IN TRANS4	Reserved	MEM IN TRANS4
R-0	R-n	R-0	R-n

15 9	8	7 2	1 0
Reserved	DOMAIN ON4	Reserved	LOGICPDPWR STAT4
R-0	R-n	R-0	R-n

LEGEND: R = Read only; -*n* = value after reset

Table 5-11. Logic Power Domain PD6 Power Status Register (LOGICPDPWRSTAT4) Field Descriptions

Bit	Field	Value	Description
31-25	Reserved	0	Reads return 0. Writes have no effect.
24	LOGIC IN TRANS4		Logic in transition status for power domain PD6. Read in User and Privileged Mode.
		0	Logic in power domain PD6 is in the steady Active or OFF state.
		1	Logic in power domain PD6 is in the process of power-down/up.
13-17	Reserved	0	Reads return 0. Writes have no effect.
16	MEM IN TRANS4		Memory in transition status for power domain PD6. Read in User and Privileged Mode.
		0	Memory in power domain PD6 is in the steady Active or OFF state.
		1	Memory in power domain PD6 is in the process of power-down/up.
15-9	Reserved	0	Reads return 0. Writes have no effect.
8	DOMAIN ON4		Current state of power domain PD6. The default value of this field is controlled by the device reset configuration word in the TI-OTP region of flash bank 0. Read in User and Privileged Mode.
		0	Power domain PD6 is in the OFF state.
		1	Power domain PD6 is in the Active state.
7-2	Reserved	0	Reads return 0. Writes have no effect.
1-0	LOGICPDPWR STAT4		Logic power domain PD6 power state. The default value of this field is controlled by the device reset configuration word in the TI-OTP region of flash bank 0. Read in User and Privileged Mode.
		0	Logic power domain PD6 is switched OFF.
		1h	Logic power domain PD6 is in Idle state.
		2h	Reserved
		3h	Logic power domain PD6 is in Active state.

5.4.11 Global Control Register 1 (GLOBALCTRL1)

Figure 5-13. Global Control Register 1 (GLOBALCTRL1) (offset = A0h)

31	16
Reserved	
R-0	

15	9	8
Reserved		PMCTRL PWRDN
R-0		R/WP-0

7	1	0
Reserved		AUTO CLK WAKE ENA
R-0		R/WP-0

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privileged mode only; -*n* = value after reset

Table 5-12. Global Control Register 1 (GLOBALCTRL1) Field Descriptions

Bit	Field	Value	Description
31-9	Reserved	0	Reads return 0. Writes have no effect.
8	PMCTRL PWRDN		PMC/PSCON Power Down Read in User and Privileged Mode returns current value of PMCTRL PWRDN. Write in Privileged mode only.
		0	Enable the clock to pmctrl_wakeup block.
		1	Disable the clock to pmctrl_wakeup block, which contains PMC and all PSCONs.
7-1	Reserved	0	Reads return 0. Writes have no effect.
0	AUTO CLK WAKE ENA		Automatic Clock Enable on Wake Up Read in User and Privileged Mode returns current value of AUTO CLK WAKE ENA. Write in Privileged mode only.
		0	Disable automatic clock wake up. The application must enable clocks by clearing the correct bit in the PDCLK_DIS register.
		1	Enable automatic clock wake up when a power domain transitions to Active state.

5.4.12 Global Status Register (GLOBALSTAT)

Figure 5-14. Global Status Register (GLOBALSTAT) (offset = A8h)

31	16
Reserved	
R-0	

15	1	0
Reserved		PMCTRL IDLE
R-0		R-1

LEGEND: R = Read only; -*n* = value after reset

Table 5-13. Global Status Register (GLOBALSTAT) Field Descriptions

Bit	Field	Value	Description
31-1	Reserved	0	Reads return 0. Writes have no effect.
0	PMCTRL IDLE		State of PMC and all PSCONs. The PMM captures the status of PMC and PSCONs as they do not have a register interface to the host CPU.
		0	PMC and PSCONs for all power domains are in the process of generating power state transition control sequence for logic and/or SRAM.
		1	PMC and PSCONs for all power domains have completed generating power state transition control sequence triggered by PMC input control signals.

5.4.13 PSCON Diagnostic Compare Key Register (PRCKEYREG)

Figure 5-15. PSCON Diagnostic Compare Key Register (PRCKEYREG) (offset = ACh)

31	16
Reserved	
R-0	

15	4	3	0
Reserved		MKEY	
R-0		R/WP-0	

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privileged mode only; -*n* = value after reset

Table 5-14. PSCON Diagnostic Compare Key Register (PRCKEYREG) Field Descriptions

Bit	Field	Value	Description
31-4	Reserved	0	Reads return 0. Writes have no effect.
3-0	MKEY		Diagnostic PSCON Mode Key. The mode key is applied to all individual PSCON compare units.
			Read in User and Privileged mode returns the current value of MKEY. Write in Privileged mode only.
		0	Lock Step mode
		6h	Self-test mode
		9h	Error Forcing mode
		Fh	Self-test Error Forcing Mode
		All others	Lock Step mode

5.4.14 LogicPD PSCON Diagnostic Compare Status Register 1 (LPDDCSTAT1)

Figure 5-16. LogicPD PSCON Diagnostic Compare Status Register 1 (LPDDCSTAT1) (offset = B0h)

31	24
Reserved	
R-0	

23	21	20	19	18	17	16
Reserved		LCMPE[4]	LCMPE[3]	LCMPE[2]	LCMPE[1]	LCMPE[0]
R-0		R/W1CP-0	R/W1CP-0	R/W1CP-0	R/W1CP-0	R/W1CP-0

15	8
Reserved	
R-0	

7	5	4	3	2	1	0
Reserved		LSTC[4]	LSTC[3]	LSTC[2]	LSTC[1]	LSTC[0]
R-0		R-0	R-0	R-0	R-0	R-0

LEGEND: R/W = Read/Write; R = Read only; W1C = Write 1 to Clear; WP = Write in privileged mode only; -*n* = value after reset

Table 5-15. LogicPD PSCON Diagnostic Compare Status Register 1 (LPDDCSTAT1) Field Descriptions

Bit	Field	Value	Description
31-21	Reserved	0	Reads return 0. Writes have no effect.
20-16	LCMPE[4-0]		Logic Power Domain Compare Error Each of these bits corresponds to a logic power domain: Bit 4 for PD6, Bit 3 for PD5, Bit 2 for PD4, Bit 1 for PD3, Bit 0 for PD2. Read in User and Privileged Mode. Write in Privileged mode only.
		0	Read: PSCON signals are identical. Write: Writing 0 has no effect.
		1	Read: PSCON signal compare mismatch identified. Write: Clears the corresponding LCMPE bit, if set.
15-5	Reserved	0	Reads return 0. Writes have no effect.
4-0	LSTC[4-0]		Logic Power Domain Self-test Complete Each of these bits corresponds to a logic power domain: Bit 4 for PD6, Bit 3 for PD5, Bit 2 for PD4, Bit 1 for PD3, Bit 0 for PD2. Read in User and Privileged Mode. Writes have no effect.
		0	Self-test is ongoing if self-test mode is entered.
		1	Self-test is complete.

5.4.15 LogicPD PSCON Diagnostic Compare Status Register 2 (LPDDCSTAT2)

Figure 5-17. LogicPD PSCON Diagnostic Compare Status Register 2 (LPDDCSTAT2) (offset = B4h)

31-24
Reserved
R-0

23-21	20	19	18	17	16
Reserved	LSTET[4]	LSTET[3]	LSTET[2]	LSTET[1]	LSTET[0]
R-0	R-0	R-0	R-0	R-0	R-0

15-8
Reserved
R-0

7-5	4	3	2	1	0
Reserved	LSTE[4]	LSTE[3]	LSTE[2]	LSTE[1]	LSTE[0]
R-0	R-0	R-0	R-0	R-0	R-0

LEGEND: R = Read only; -*n* = value after reset

Table 5-16. LogicPD PSCON Diagnostic Compare Status Register 2 (LPDDCSTAT2) Field Descriptions

Bit	Field	Value	Description
31-21	Reserved	0	Reads return 0. Writes have no effect.
20-16	LSTET[4-0]		Logic Power Domain Self-test Error Type Each of these bits corresponds to a logic power domain: Bit 4 for PD6, Bit 3 for PD5, Bit 2 for PD4, Bit 1 for PD3, Bit 0 for PD2. Read in User and Privileged Mode. Writes have no effect.
		0	Self-test failed during compare match test.
		1	Self-test failed during compare mismatch test.
15-5	Reserved	0	Reads return 0. Writes have no effect.
4-0	LSTE[4-0]		Logic Power Domain Self-test Error Each of these bits corresponds to a logic power domain: Bit 4 for PD6, Bit 3 for PD5, Bit 2 for PD4, Bit 1 for PD3, Bit 0 for PD2. Read in User and Privileged Mode. Writes have no effect.
		0	Self-test passed.
		1	Self-test failed.

5.4.16 Isolation Diagnostic Status Register (ISODIAGSTAT)

Figure 5-18. Isolation Diagnostic Status Register (ISODIAGSTAT) (offset = C0h)

31	8
Reserved	
R-0	

	4	3	2	1	0
Reserved	ISO DIAG[4]	ISO DIAG[3]	ISO DIAG[2]	ISO DIAG[1]	ISO DIAG[0]
R-0	R-0	R-0	R-0	R-0	R-0

LEGEND: R = Read only; -*n* = value after reset

Table 5-17. Isolation Diagnostic Status Register (ISODIAGSTAT) Field Descriptions

Bit	Field	Value	Description
31-5	Reserved	0	Reads return 0. Writes have no effect.
4-0	ISO DIAG[4-0]		Isolation Diagnostic Each of these bits corresponds to a logic power domain: Bit 4 for PD6, Bit 3 for PD5, Bit 2 for PD4, Bit 1 for PD3, Bit 0 for PD2. Read in User and Privileged Mode. Writes have no effect.
		0	Isolation is enabled for corresponding power domain
		1	Isolation is disabled for corresponding power domain

TEXAS INSTRUMENTS

Chapter 6

SPNU563A–March 2018

I/O Multiplexing and Control Module (IOMM)

This chapter describes the I/O Multiplexing and Control Module (IOMM).

Topic **Page**

6.1 Overview

This chapter describes the overall features of the module that control the I/O multiplexing on the device. The mapping of control registers to multiplexing options is specified in Section 6.7.13.

6.2 Main Features of I/O Multiplexing Module (IOMM)

The IOMM contains memory-mapped registers (MMR) that control device-specific multiplexed functions. The safety and diagnostic features of the IOMM are:

- Kicker mechanism to protect the MMRs from accidental writes
- Error indication for access violations

6.3 Control of Multiplexed Outputs

The signal multiplexing controlled by each memory-mapped control register (PINMMRn) is described in Table 6-1. Each byte in the PINMMRs control the functionality output on a single terminal. Consider the following example for the PINMMR9 control register.

Figure 6-1. PINMMR9 Control Register [Address Offset = 134h]

31 … 27	26	25	24
Reserved	GIOB[4]	Reserved	EMIF_nCS[2]
R/WP-0	R/WP-0	R/WP-0	R/WP-1

23 … 19	18	17	16
Reserved	N2HET2[7]	RTP_DATA[15]	EMIF nCS[0]
R/WP-0	R/WP-0	R/WP-0	R/WP-1

15 … 11	10	9	8
Reserved	GIOB[3]	Reserved	EMIF_nCAS
R/WP-0	R/WP-0	R/WP-0	R/WP-1

7 … 3	2	1	0
Reserved	ECLK2	EMIF_CLK	Reserved
R/WP-0	R/WP-0	R/WP-0	R/WP-1

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privileged mode only; -*n* = value after reset

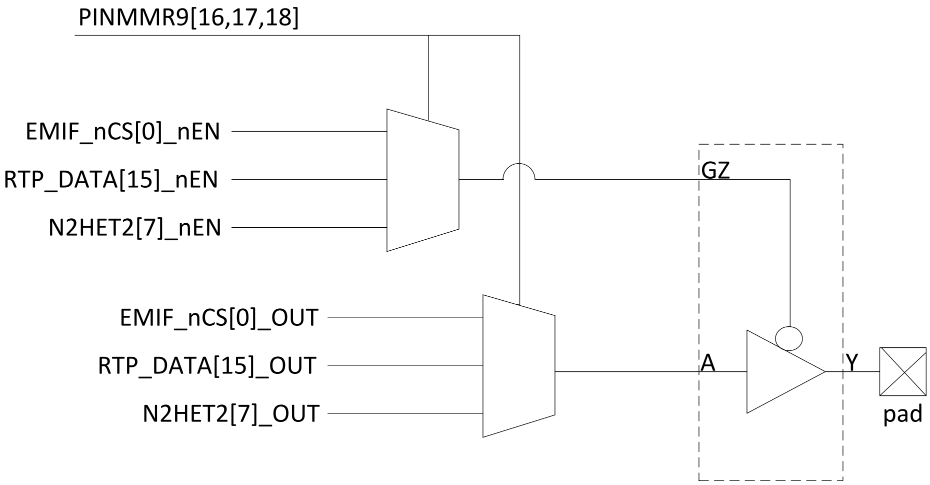
- Consider the multiplexing controlled by PINMMR9[23–16]. These bits control the multiplexing between the EMIF_nCS[0], RTP_DATA[15] and N2HET2[7] on the ball N17 of the 337BGA package for this device. The default function on the N17 ball is EMIF_nCS[0]. This is dictated by bit 16 of the PINMMR9 register being set.
- If the application wants to use N17 as an N2HET2[7] signal, then bit 16 of PINMMR9 must be cleared and bit 18 must be set. Likewise, if RTP_DATA[15] is to be brought out, then bit 16 of PINMMR9 must be cleared and bit 17 must be set.
- Each feature of the output function is determined by the function selected to be output on a terminal.

  For example, the ball N17 on the 337BGA package is driven by an output buffer with an 8mA drive strength. This output buffer has the following signals: A (signal to be output) and GZ (output enable). Each of these signals is an output of a multiplexor that allows the selected function to control all available features of the output buffer. Some output buffers may have additional options as output strength, slew rate, and so on. This options are also controlled by the multiplexor output.
- The PINMMR control registers are used to implement a one-hot encoding scheme for selecting the multiplexed function.
  - For example, for the N17 ball on the 337BGA package for this device only one out of bit 16, 17 or bit 18 must be set.
  - If the application clears bits 16, 17 and 18, then the default function, EMIF_nCS[0], will be selected for output on N17.
  - If the application sets 16, 17 and 18, then the default function will be selected for output on N17.

- – If the application sets one or more reserved bit(s) within the byte 23–16, then the default function will be selected for output on N17.

Figure 6-2 shows the multiplexing between the output functions for the N17 ball. This terminal uses an 8mA output buffer.

Figure 6-2. Output Multiplexing Example



In Table 6-1, the column "Selection Bit" contains a value of type x[y] that corresponds to the control register PINMMRx, bit y. It indicates the multiplexing control register and the bit that must be set in order to select the corresponding functionality to be brought out to the terminal. If an un-implemented alternate function is selected where a physical pin is attached, the default function is used. When a PINMMRx register is completely reserved, none of its 8-bit fields are attached to any physical pin.

6.4 Control of Multiplexed Inputs

In this microcontroller, some signals are connected to more than one terminal, so that the inputs for these signals can come from either of these terminals. A multiplexor is implemented to let the application choose the terminal that will be used for providing the input signal from among the available options. The input path selection is done based on two bits in the PINMMR control register as shown in Table 6-2.

- The input to a module comes from the **Default Terminal** when the associated bit in the **Terminal 1 Input Multiplex Control** column is set and the bit in the **Terminal 2 Input Multiplex Control** column is clear. By default, the bit in the **Terminal 1 Input Multiplex Control** column is set after reset.
- The input to a module comes from the **Alternate Terminal** when the associated bit in the **Terminal 2 Input Multiplex Control** column is set and the bit in the **Terminal 1 Input Multiplex Control** column is clear.

NOTE: If multiple bits or no bit are selected in the Input Multiplex Control, the Default Function will then be selected.

Some signals, like eCAPx and eQEPx, are by default mapped to an unavailable ball on the 337ZWT package. The alternate terminals have to be used, in this case, in order to use these signals.

Table 6-1. Multiplexing for Outputs on 337ZWT Package

Address Offset	337ZWT BALL	Default Function	Selection Bit	Alternate Function 1	Selection Bit	Alternate Function 2	Selection Bit	Alternate Function 3	Selection Bit	Alternate Function 4	Selection Bit	Alternate Function 5	Selection Bit
110h	N19	AD1EVT	0[0]			MII_RX_ER	0[2]	RMII_RX_ER	0[3]			nTZ1_1	0[5]
	D4	EMIF_ADDR[00]	0[8]			N2HET2[01]	0[10]						
	D5	EMIF_ADDR[01]	0[16]			N2HET2[03]	0[18]						
	C4	EMIF_ADDR[06]	0[24]	RTP_DATA[13]	0[25]	N2HET2[11]	0[26]						
114h	C5	EMIF_ADDR[07]	1[0]	RTP_DATA[12]	1[1]	N2HET2[13]	1[2]						
	C6	EMIF_ADDR[08]	1[8]	RTP_DATA[11]	1[9]	N2HET2[15]	1[10]						
	C7	EMIF_ADDR[09]	1[16]	RTP_DATA[10]	1[17]								
	C8	EMIF_ADDR[10]	1[24]	RTP_DATA[09]	1[25]								
118h	C9	EMIF_ADDR[11]	2[0]	RTP_DATA[08]	2[1]								
	C10	EMIF_ADDR[12]	2[8]	RTP_DATA[06]	2[9]								
	C11	EMIF_ADDR[13]	2[16]	RTP_DATA[05]	2[17]								
	C12	EMIF_ADDR[14]	2[24]	RTP_DATA[04]	2[25]								
11Ch	C13	EMIF_ADDR[15]	3[0]	RTP_DATA[03]	3[1]								
	D14	EMIF_ADDR[16]	3[8]	RTP_DATA[02]	3[9]								
	C14	EMIF_ADDR[17]	3[16]	RTP_DATA[01]	3[17]								
	D15	EMIF_ADDR[18]	3[24]	RTP_DATA[00]	3[25]								
120h	C15	EMIF_ADDR[19]	4[0]	RTP_nENA	4[1]								
	C16	EMIF_ADDR[20]	4[8]	RTP_nSYNC	4[9]								
	C17	EMIF_ADDR[21]	4[16]	RTP_CLK	4[17]								
124h-12Ch	Reserved												
130h	PINMMR8[23:0] are reserved												
	D16	EMIF_BA[1]	8[24]		8[25]	N2HET2[05]	8[26]						
134h	K3	RESERVED	9[0]	EMIF_CLK	9[1]	ECLK2	9[2]						
	R4	EMIF_nCAS	9[8]			GIOB[3]	9[10]						
	N17	EMIF_nCS[0]	9[16]	RTP_DATA[15]	9[17]	N2HET2[07]	9[18]						
	L17	EMIF_nCS[2]	9[24]			GIOB[4]	9[26]						
138h	K17	EMIF_nCS[3]	10[0]	RTP_DATA[14]	10[1]	N2HET2[09]	10[2]						
	M17	EMIF_nCS[4]	10[8]	RTP_DATA[07]	10[9]	GIOB[5]	10[10]						
	R3	EMIF_nRAS	10[16]			GIOB[6]	10[18]						
	P3	EMIF_nWAIT	10[24]			GIOB[7]	10[26]						
13Ch	D17	EMIF_nWE	11[0]	EMIF_RNW	11[1]								
	E9	ETMDATA[08]	11[8]	EMIF_ADDR[05]	11[9]								
	E8	ETMDATA[09]	11[16]	EMIF_ADDR[04]	11[17]								
	E7	ETMDATA[10]	11[24]	EMIF_ADDR[03]	11[25]								
140h	E6	ETMDATA[11]	12[0]	EMIF_ADDR[02]	12[1]								
	E13	ETMDATA[12]	12[8]	EMIF_BA[0]	12[9]								
	E12	ETMDATA[13]	12[16]	EMIF_nOE	12[17]								
	E11	ETMDATA[14]	12[24]	EMIF_nDQM[1]	12[25]								

Table 6-1. Multiplexing for Outputs on 337ZWT Package (continued)

Address Offset	337ZWT BALL	Default Function	Selection Bit	Alternate Function 1	Selection Bit	Alternate Function 2	Selection Bit	Alternate Function 3	Selection Bit	Alternate Function 4	Selection Bit	Alternate Function 5	Selection Bit
144h	E10	ETMDATA[15]	13[0]	EMIF_nDQM[0]	13[1]								
	K15	ETMDATA[16]	13[8]	EMIF_DATA[00]	13[9]								
	L15	ETMDATA[17]	13[16]	EMIF_DATA[01]	13[17]								
	M15	ETMDATA[18]	13[24]	EMIF_DATA[02]	13[25]								
148h	N15	ETMDATA[19]	14[0]	EMIF_DATA[03]	14[1]								
	E5	ETMDATA[20]	14[8]	EMIF_DATA[04]	14[9]								
	F5	ETMDATA[21]	14[16]	EMIF_DATA[05]	14[17]								
	G5	ETMDATA[22]	14[24]	EMIF_DATA[06]	14[25]								
14Ch	K5	ETMDATA[23]	15[0]	EMIF_DATA[07]	15[1]								
	L5	ETMDATA[24]	15[8]	EMIF_DATA[08]	15[9]	N2HET2[24]	15[10]	MIBSPI5NCS[4]	15[11]				
	M5	ETMDATA[25]	15[16]	EMIF_DATA[09]	15[17]	N2HET2[25]	15[18]	MIBSPI5NCS[5]	15[19]				
	N5	ETMDATA[26]	15[24]	EMIF_DATA[10]	15[25]	N2HET2[26]	15[26]						
150h	P5	ETMDATA[27]	16[0]	EMIF_DATA[11]	16[1]	N2HET2[27]	16[2]						
	R5	ETMDATA[28]	16[8]	EMIF_DATA[12]	16[9]	N2HET2[28]	16[10]	GIOA[0]	16[11]				
	R6	ETMDATA[29]	16[16]	EMIF_DATA[13]	16[17]	N2HET2[29]	16[18]	GIOA[1]	16[19]				
	R7	ETMDATA[30]	16[24]	EMIF_DATA[14]	16[25]	N2HET2[30]	16[26]	GIOA[3]	16[27]				
154h	R8	ETMDATA[31]	17[0]	EMIF_DATA[15]	17[1]	N2HET2[31]	17[2]	GIOA[4]	17[3]				
	R9	ETMTRACECLKIN	17[8]	EXTCLKIN2	17[9]			GIOA[5]	17[11]				
	R10	ETMTRACECLKOUT	17[16]					GIOA[6]	17[19]				
	R11	ETMTRACECTL	17[24]					GIOA[7]	17[27]				
158h	B15	FRAYTX1	18[0]					GIOA[2]	18[3]				
	B8	FRAYTX2	18[8]					GIOB[0]	18[11]				
	B16	FRAYTXEN1	18[16]					GIOB[1]	18[19]				
	B9	FRAYTXEN2	18[24]					GIOB[2]	18[27]				
15Ch	C1	GIOA[2]	19[0]			N2HET2[00]	19[2]					eQEP2I	19[5]
	E1	GIOA[3]	19[8]			N2HET2[02]	19[10]						
	B5	GIOA[5]	19[16]					EXTCLKIN	19[19]			ePWM1A	19[21]
	H3	GIOA[6]	19[24]			N2HET2[04]	19[26]					ePWM1B	19[29]
160h	M1	GIOA[7]	20[0]			N2HET2[06]	20[2]					ePWM2A	20[5]
	F2	GIOB[2]	20[8]					DCAN4TX	20[11]				
	W10	GIOB[3]	20[16]					DCAN4RX	20[19]				
	J2	GIOB[6]	20[24]	nERROR1	20[25]								
164h	F1	GIOB[7]	21[0]	nERROR2	21[1]							nTZ1_2	21[5]
	R2	MIBSPI1NCS[0]	21[8]	MIBSPI1SOMI[1]	21[9]	MII_TXD[2]	21[10]					ECAP6	21[13]
	F3	MIBSPI1NCS[1]	21[16]			MII_COL	21[18]	N2HET1[17]	21[19]			eQEP1S	21[21]
	G3	MIBSPI1NCS[2]	21[24]			MDIO	21[26]	N2HET1[19]	21[27]				

Table 6-1. Multiplexing for Outputs on 337ZWT Package (continued)

Address Offset	337ZWT BALL	Default Function	Selection Bit	Alternate Function 1	Selection Bit	Alternate Function 2	Selection Bit	Alternate Function 3	Selection Bit	Alternate Function 4	Selection Bit	Alternate Function 5	Selection Bit
168h	J3	MIBSPI1NCS[3]	22[0]					N2HET1[21]	22[3]			nTZ1_3	22[5]
	G19	MIBSPI1NENA	22[8]			MII_RXD[2]	22[10]	N2HET1[23]	22[11]			ECAP4	22[13]
	V9	MIBSPI3CLK	22[16]	EXT_SEL[01]	22[17]							eQEP1A	22[21]
	V10	MIBSPI3NCS[0]	22[24]	AD2EVT	22[25]							eQEP1I	22[29]
16Ch	V5	MIBSPI3NCS[1]	23[0]			MDCLK	23[2]	N2HET1[25]	23[3]				
	B2	MIBSPI3NCS[2]	23[8]	I2C1_SDA	23[9]			N2HET1[27]	23[11]			nTZ1_2	23[13]
	C3	MIBSPI3NCS[3]	23[16]	I2C1_SCL	23[17]			N2HET1[29]	23[19]			nTZ1_1	23[21]
	W9	MIBSPI3NENA	23[24]	MIBSPI3NCS[5]	23[25]			N2HET1[31]	23[27]			eQEP1B	23[29]
170h	W8	MIBSPI3SIMO	24[0]	EXT_SEL[00]	24[1]							ECAP3	24[5]
	V8	MIBSPI3SOMI	24[8]	EXT_ENA	24[9]							ECAP2	24[13]
	H19	MIBSPI5CLK	24[16]	DMM_DATA[04]	24[17]	MII_TXEN	24[18]	RMII_TXEN	24[19]				
	E19	MIBSPI5NCS[0]	24[24]	DMM_DATA[05]	24[25]							ePWM4A	24[29]
174h	B6	MIBSPI5NCS[1]	25[0]	DMM_DATA[06]	25[1]								
	W6	MIBSPI5NCS[2]	25[8]	DMM_DATA[02]	25[9]								
	T12	MIBSPI5NCS[3]	25[16]	DMM_DATA[03]	25[17]								
	H18	MIBSPI5NENA	25[24]	DMM_DATA[07]	25[25]	MII_RXD[3]	25[26]					ECAP5	25[29]
178h	J19	MIBSPI5SIMO[0]	26[0]	DMM_DATA[08]	26[1]	MII_TXD[1]	26[2]	RMII_TXD[1]	26[3]				
	E16	MIBSPI5SIMO[1]	26[8]	DMM_DATA[09]	26[9]					EXT_SEL[00]	26[12]		
	H17	MIBSPI5SIMO[2]	26[16]	DMM_DATA[10]	26[17]					EXT_SEL[01]	26[20]		
	G17	MIBSPI5SIMO[3]	26[24]	DMM_DATA[11]	26[25]	I2C2_SDA	26[26]			EXT_SEL[02]	26[28]		
17Ch	J18	MIBSPI5SOMI[0]	27[0]	DMM_DATA[12]	27[1]	MII_TXD[0]	27[2]	RMII_TXD[0]	27[3]				
	E17	MIBSPI5SOMI[1]	27[8]	DMM_DATA[13]	27[9]					EXT_SEL[03]	27[12]		
	H16	MIBSPI5SOMI[2]	27[16]	DMM_DATA[14]	27[17]					EXT_SEL[04]	27[20]		
	G16	MIBSPI5SOMI[3]	27[24]	DMM_DATA[15]	27[25]	I2C2_SCL	27[26]			EXT_ENA	27[28]		
180h	K18	N2HET1[00]	28[0]	MIBSPI4CLK	28[1]							ePWM2B	28[5]
	V2	N2HET1[01]	28[8]	MIBSPI4NENA	28[9]			N2HET2[08]	28[11]			eQEP2A	28[13]
	W5	N2HET1[02]	28[16]	MIBSPI4SIMO	28[17]							ePWM3A	28[21]
	U1	N2HET1[03]	28[24]	MIBSPI4NCS[0]	28[25]			N2HET2[10]	28[27]			eQEP2B	28[29]
184h	B12	N2HET1[04]	29[0]	MIBSPI4NCS[1]	29[1]							ePWM4B	29[5]
	V6	N2HET1[05]	29[8]	MIBSPI4SOMI	29[9]			N2HET2[12]	29[11]			ePWM3B	29[13]
	W3	N2HET1[06]	29[16]	SCI3RX	29[17]							ePWM5A	29[21]
	T1	N2HET1[07]	29[24]	MIBSPI4NCS[2]	29[25]			N2HET2[14]	29[27]			ePWM7B	29[29]
188h	E18	N2HET1[08]	30[0]	MIBSPI1SIMO[1]	30[1]	MII_TXD[3]	30[2]						
	V7	N2HET1[09]	30[8]	MIBSPI4NCS[3]	30[9]			N2HET2[16]	30[11]			ePWM7A	30[13]
	D19	N2HET1[10]	30[16]	MIBSPI4NCS[4]	30[17]	MII_TX_CLK	30[18]					nTZ1_3	30[21]
	E3	N2HET1[11]	30[24]	MIBSPI3NCS[4]	30[25]			N2HET2[18]	30[27]			EPWM1SYNCO	30[29]

Table 6-1. Multiplexing for Outputs on 337ZWT Package (continued)

Address Offset	337ZWT BALL	Default Function	Selection Bit	Alternate Function 1	Selection Bit	Alternate Function 2	Selection Bit	Alternate Function 3	Selection Bit	Alternate Function 4	Selection Bit	Alternate Function 5	Selection Bit
18Ch	B4	N2HET1[12]	31[0]	MIBSPI4NCS[5]	31[1]	MII_CRS	31[2]	RMII_CRS_DV	31[3]				
	N2	N2HET1[13]	31[8]	SCI3TX	31[9]			N2HET2[20]	31[11]			ePWM5B	31[13]
	N1	N2HET1[15]	31[16]	MIBSPI1NCS[4]	31[17]			N2HET2[22]	31[19]			ECAP1	31[21]
	A4	N2HET1[16]	31[24]					EPWM1SYNCI	31[27]			EPWM1SYNCO	31[29]
190h	A13	N2HET1[17]	32[0]	EMIF_nOE	32[1]	SCI4RX	32[2]						
	J1	N2HET1[18]	32[8]	EMIF_RNW	32[9]							ePWM6A	32[13]
	B13	N2HET1[19]	32[16]	EMIF_nDQM[0]	32[17]	SCI4TX	32[18]						
	P2	N2HET1[20]	32[24]	EMIF_nDQM[1]	32[25]							ePWM6B	32[29]
194h	H4	N2HET1[21]	33[0]	EMIF_nDQM[2]	33[1]								
	B3	N2HET1[22]	33[8]	EMIF_nDQM[3]	33[9]								
	J4	N2HET1[23]	33[16]	EMIF_BA[0]	33[17]								
	P1	N2HET1[24]	33[24]	MIBSPI1NCS[5]	33[25]	MII_RXD[0]	33[26]	RMII_RXD[0]	33[27]				
198h	A14	N2HET1[26]	34[0]			MII_RXD[1]	34[2]	RMII_RXD[1]	34[3]				
	K19	N2HET1[28]	34[8]			MII_RXCLK	34[10]	RMII_REFCLK	34[11]				
	B11	N2HET1[30]	34[16]			MII_RX_DV	34[18]					eQEP2S	34[21]
	D8	N2HET2[01]	34[24]	N2HET1_NDIS[1]	34[25]								
19Ch	D7	N2HET2[02]	35[0]	N2HET2_NDIS[1]	35[1]								
	D3	N2HET2[12]	35[8]							MIBSPI2NENA	35[12]	MIBSPI2NCS[1]	35[13]
	D2	N2HET2[13]	35[16]							MIBSPI2SOMI	35[20]		
	D1	N2HET2[14]	35[24]							MIBSPI2SIMO	35[28]		
1A0h	P4	N2HET2[19]	36[0]	LIN2RX	36[1]								
	T5	N2HET2[20]	36[8]	LIN2TX	36[9]								
	T4	MII_RXCLK	36[16]										
	U7	MII_TX_CLK	36[24]										
1A4h	E2	N2HET2[03]	37[0]							MIBSPI2CLK	37[4]		
	N3	N2HET2[07]	37[8]							MIBSPI2NCS[0]	37[12]		

[1] Selecting N2HET1_NDIS or N2HET2_NDIS forces the pin to a high-impedance state and changes the pull type to pull up.

Table 6-2. Input Multiplexing and Control on 337ZWT Package

Address Offset	Signal Name	Default Terminal	Terminal 1 Input Multiplex Control	Alternate Terminal	Terminal 2 Input Multiplex Control
250h	AD2EVT	T10	PINMMR80[0]	V10	PINMMR80[1]
	ECAP1	N/A on 337ZWT	PINMMR80[8]	N1	PINMMR80[9]
	ECAP2	N/A on 337ZWT	PINMMR80[16]	V8	PINMMR80[17]
	ECAP3	N/A on 337ZWT	PINMMR80[24]	W8	PINMMR80[25]
254h	ECAP4	N/A on 337ZWT	PINMMR81[0]	G19	PINMMR81[1]
	ECAP5	N/A on 337ZWT	PINMMR81[8]	H18	PINMMR81[9]
	ECAP6	N/A on 337ZWT	PINMMR81[16]	R2	PINMMR81[17]
	eQEP1A	N/A on 337ZWT	PINMMR81[24]	V9	PINMMR81[25]
258h	eQEP1B	N/A on 337ZWT	PINMMR82[0]	W9	PINMMR82[1]
	eQEP1I	N/A on 337ZWT	PINMMR82[8]	V10	PINMMR82[9]
	eQEP1S	N/A on 337ZWT	PINMMR82[16]	F3	PINMMR82[17]
	eQEP2A	N/A on 337ZWT	PINMMR82[24]	V2	PINMMR82[25]
25Ch	eQEP2B	N/A on 337ZWT	PINMMR83[0]	U1	PINMMR83[1]
	eQEP2I	N/A on 337ZWT	PINMMR83[8]	C1	PINMMR83[9]
	eQEP2S	N/A on 337ZWT	PINMMR83[16]	B11	PINMMR83[17]
	GIOA[0]	A5	PINMMR83[24]	R5	PINMMR83[25]
260h	GIOA[1]	C2	PINMMR84[0]	R6	PINMMR84[1]
	GIOA[2]	C1	PINMMR84[8]	B15	PINMMR84[9]
	GIOA[3]	E1	PINMMR84[16]	R7	PINMMR84[17]
	GIOA[4]	A6	PINMMR84[24]	R8	PINMMR84[25]
264h	GIOA[5]	B5	PINMMR85[0]	R9	PINMMR85[1]
	GIOA[6]	H3	PINMMR85[8]	R10	PINMMR85[9]
	GIOA[7]	M1	PINMMR85[16]	R11	PINMMR85[17]
	GIOB[0]	M2	PINMMR85[24]	B8	PINMMR85[25]
268h	GIOB[1]	K2	PINMMR86[0]	B16	PINMMR86[1]
	GIOB[2]	F2	PINMMR86[8]	B9	PINMMR86[9]
	GIOB[3]	W10	PINMMR86[16]	R4	PINMMR86[17]
	GIOB[4]	G1	PINMMR86[24]	L17	PINMMR86[25]
26Ch	GIOB[5]	G2	PINMMR87[0]	M17	PINMMR87[1]
	GIOB[6]	J2	PINMMR87[8]	R3	PINMMR87[9]
	GIOB[7]	F1	PINMMR87[16]	P3	PINMMR87[17]
	MDIO	F4	PINMMR87[24]	G3	PINMMR87[25]
270h	MIBSPI1NCS[4]	U10	PINMMR88[0]	N1	PINMMR88[1]
	MIBSPI1NCS[5]	U9	PINMMR88[8]	P1	PINMMR88[9]
	MIBSPI2NCS[1]	N/A on 337ZWT	PINMMR88[16]	D3	PINMMR88[17]
274h	MII_COL	W4	PINMMR89[16]	F3	PINMMR89[17]
	MII_CRS	V4	PINMMR89[24]	B4	PINMMR89[25]
278h	MII_RX_DV	U6	PINMMR90[0]	B11	PINMMR90[1]
	MII_RX_ER	U5	PINMMR90[8]	N19	PINMMR90[9]
	MII_RXCLK	T4	PINMMR90[16]	K19	PINMMR90[17]
	MII_RXD[0]	U4	PINMMR90[24]	P1	PINMMR90[25]
27Ch	MII_RXD[1]	T3	PINMMR91[0]	A14	PINMMR91[1]
	MII_RXD[2]	U3	PINMMR91[8]	G19	PINMMR91[9]
	MII_RXD[3]	V3	PINMMR91[16]	H18	PINMMR91[17]
	MII_TX_CLK	U7	PINMMR91[24]	D19	PINMMR91[25]

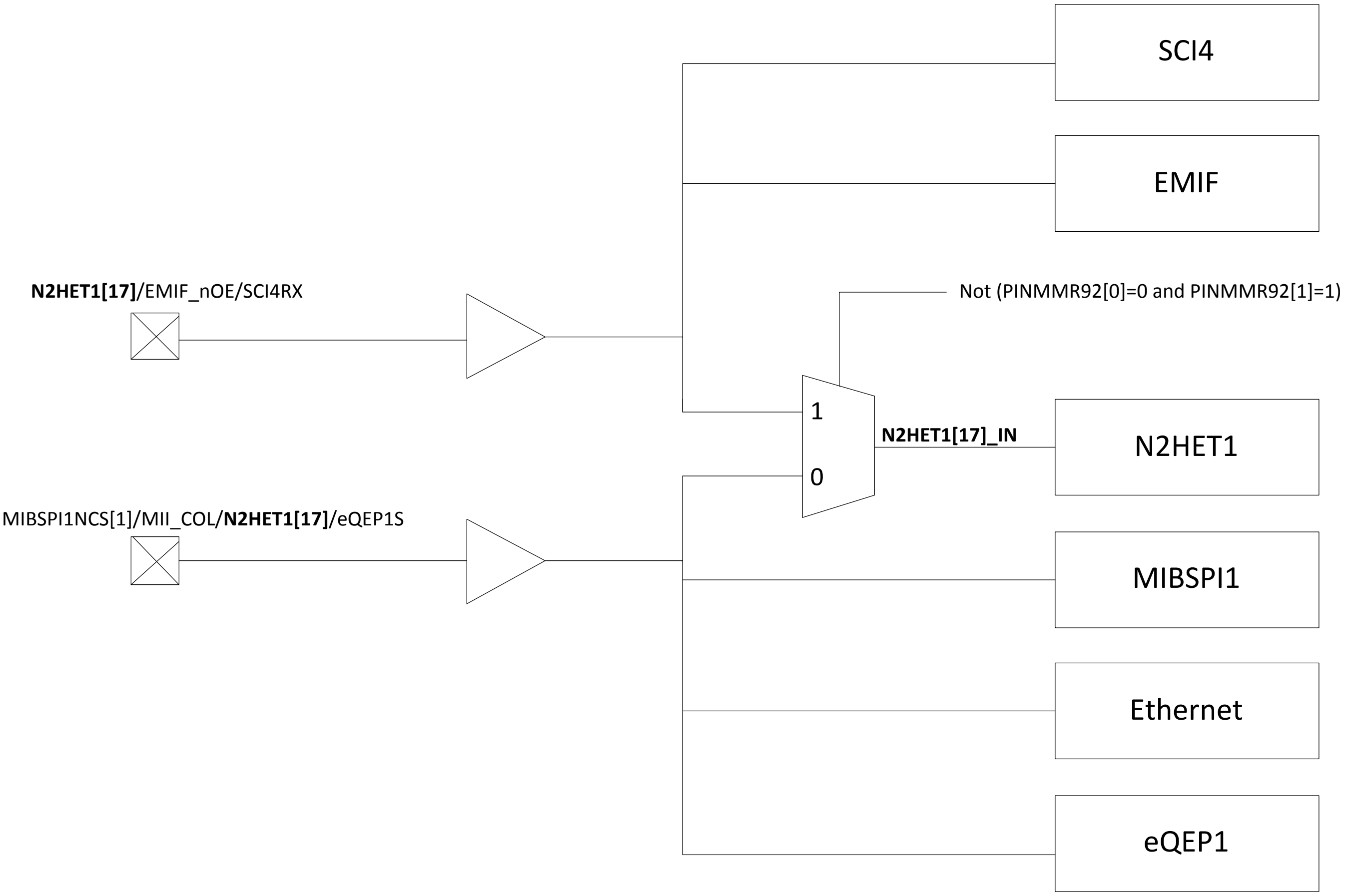
Table 6-2. Input Multiplexing and Control on 337ZWT Package (continued)

Address Offset	Signal Name	Default Terminal	Terminal 1 Input Multiplex Control	Alternate Terminal	Terminal 2 Input Multiplex Control
280h	N2HET1[17]	A13	PINMMR92[0]	F3	PINMMR92[1]
	N2HET1[19]	B13	PINMMR92[8]	G3	PINMMR92[9]
	N2HET1[21]	H4	PINMMR92[16]	J3	PINMMR92[17]
	N2HET1[23]	J4	PINMMR92[24]	G19	PINMMR92[25]
284h	N2HET1[25]	M3	PINMMR93[0]	V5	PINMMR93[1]
	N2HET1[27]	A9	PINMMR93[8]	B2	PINMMR93[9]
	N2HET1[29]	A3	PINMMR93[16]	C3	PINMMR93[17]
	N2HET1[31]	J17	PINMMR93[24]	W9	PINMMR93[25]
288h	N2HET2[00]	D6	PINMMR94[0]	C1	PINMMR94[1]
	N2HET2[01]	D8	PINMMR94[8]	D4	PINMMR94[9]
	N2HET2[02]	D7	PINMMR94[16]	E1	PINMMR94[17]
	N2HET2[03]	E2	PINMMR94[24]	D5	PINMMR94[25]
28Ch	N2HET2[04]	D13	PINMMR95[0]	H3	PINMMR95[1]
	N2HET2[05]	D12	PINMMR95[8]	D16	PINMMR95[9]
	N2HET2[06]	D11	PINMMR95[16]	M1	PINMMR95[17]
	N2HET2[07]	N3	PINMMR95[24]	N17	PINMMR95[25]
290h	N2HET2[08]	K16	PINMMR96[0]	V2	PINMMR96[1]
	N2HET2[09]	L16	PINMMR96[8]	K17	PINMMR96[9]
	N2HET2[10]	M16	PINMMR96[16]	U1	PINMMR96[17]
	N2HET2[11]	N16	PINMMR96[24]	C4	PINMMR96[25]
294h	N2HET2[12]	D3	PINMMR97[0]	V6	PINMMR97[1]
	N2HET2[13]	D2	PINMMR97[8]	C5	PINMMR97[9]
	N2HET2[14]	D1	PINMMR97[16]	T1	PINMMR97[17]
	N2HET2[15]	K4	PINMMR97[24]	C6	PINMMR97[25]
298h	N2HET2[16]	L4	PINMMR98[0]	V7	PINMMR98[1]
	N2HET2[18]	N4	PINMMR98[8]	E3	PINMMR98[9]
	N2HET2[20]	T5	PINMMR98[16]	N2	PINMMR98[17]
	N2HET2[22]	T7	PINMMR98[24]	N1	PINMMR98[25]
29Ch	nTZ1_1	N19	PINMMR99[0]	C3	PINMMR99[1]
	nTZ1_2	F1	PINMMR99[8]	B2	PINMMR99[9]
	nTZ1_3	J3	PINMMR99[16]	D19	PINMMR99[17]

NOTE: Inputs are broadcast to all multiplexed functions

The input signals are broadcast to all modules hooked up to a terminal. The application must ensure that modules that are not being used in the application do not react to a change on their input functions. For example, a GIO signal toggle can trigger an interrupt request, when the application actually is using the function multiplexed with this GIO signal.

Figure 6-3. Input Multiplexing Example


SCI4
EMIF
N2HET1[17]/EMIF_nOE/SCI4RX
Not (PINMMR92[0]=0 and PINMMR92[1]=1)
1
N2HET1[17]_IN
N2HET1
0
MIBSPI1NCS[1]/MII_COL/N2HET1[17]/eQEP1S
MIBSPI1
Ethernet
eQEP1

6.5 Control of Special Multiplexed Options

Several of the PINMMR registers are used to control specific functions on this microcontroller.

Table 6-3. Special Multiplexed Controls

Control Function	PINMMR Register Address Offset	PINMMR Register Bits Used			Reference
GIOB[2] Select	390h	PINMMR160[16]	PINMMR160[17]		See Section 6.5.6
MII/RMII Select		PINMMR160[24]			See Section 6.5.3
ADC Alternate Trigger Table Select	394h	PINMMR161[0]	PINMMR161[1]		See Section 6.5.4
ADC1 Alternate Trigger Source for Trigger Input 2		PINMMR161[8]	PINMMR161[9]		
ADC1 Alternate Trigger Source for Trigger Input 4		PINMMR161[16]	PINMMR161[17]		
ADC1 Alternate Trigger Source for Trigger Input 6		PINMMR161[24]	PINMMR161[25]		
ADC1 Alternate Trigger Source for Trigger Input 7	398h	PINMMR162[0]	PINMMR162[1]		
ADC1 Alternate Trigger Source for Trigger Input 8		PINMMR162[8]	PINMMR162[9]		
ADC2 Alternate Trigger Source for Trigger Input 2		PINMMR162[16]	PINMMR162[17]		
ADC2 Alternate Trigger Source for Trigger Input 4		PINMMR162[24]	PINMMR162[25]		
ADC2 Alternate Trigger Source for Trigger Input 6	39Ch	PINMMR163[0]	PINMMR163[1]		
ADC2 Alternate Trigger Source for Trigger Input 7		PINMMR163[8]	PINMMR163[9]		
ADC2 Alternate Trigger Source for Trigger Input 8		PINMMR163[16]	PINMMR163[17]		
Selecting Start of Conversion (SOC1A) of ePWM1	3A0h	PINMMR164[0]			See Section 6.5.5
Selecting Start of Conversion (SOC2A) of ePWM2		PINMMR164[8]			
Selecting Start of Conversion (SOC3A) of ePWM3		PINMMR164[16]			
Selecting Start of Conversion (SOC4A) of ePWM4		PINMMR164[24]			
Selecting Start of Conversion (SOC5A) of ePWM5	3A4h	PINMMR165[0]			
Selecting Start of Conversion (SOC6A) of ePWM6		PINMMR165[8]			
Selecting Start of Conversion (SOC7A) of ePWM7		PINMMR165[16]			
ePWM1 SYNCI Select		PINMMR165[24]	PINMMR165[25]		See Section 6.5.8
ePWMx TBCLKSYNC Enable	3A8h	PINMMR166[1]			See Section 6.5.7
ePWM1 Trip Zone 4 Select	3ACh	PINMMR167[0]	PINMMR167[1]	PINMMR167[2]	See Section 6.5.9
ePWM2 Trip Zone 4 Select		PINMMR167[8]	PINMMR167[9]	PINMMR167[10]	
ePWM3 Trip Zone 4 Select		PINMMR167[16]	PINMMR167[17]	PINMMR167[18]	
ePWM4 Trip Zone 4 Select		PINMMR167[24]	PINMMR167[25]	PINMMR167[26]	
ePWM5 Trip Zone 4 Select	3B0h	PINMMR168[0]	PINMMR168[1]	PINMMR168[2]	
ePWM6 Trip Zone 4 Select		PINMMR168[8]	PINMMR168[9]	PINMMR168[10]	
ePWM7 Trip Zone 4 Select		PINMMR168[16]	PINMMR168[17]	PINMMR168[18]	

Table 6-3. Special Multiplexed Controls (continued)

Control Function	PINMMR Register Address Offset	PINMMR Register Bits Used			Reference
eCAP1 Input Filtering Select	3B4h	PINMMR169[0]	PINMMR169[1]		See Section 6.5.10
eCAP2 Input Filtering Select		PINMMR169[8]	PINMMR169[9]		
eCAP3 Input Filtering Select		PINMMR169[16]	PINMMR169[17]		
eCAP4 Input Filtering Select		PINMMR169[24]	PINMMR169[25]		
eCAP5 Input Filtering Select	3B8h	PINMMR170[0]	PINMMR170[1]		
eCAP6 Input Filtering Select		PINMMR170[8]	PINMMR170[9]		
eQEP1A Input Filtering Select		PINMMR170[16]	PINMMR170[17]		See Section 6.5.11
eQEP1B Input Filtering Select		PINMMR170[24]	PINMMR170[25]		
eQEP1I Input Filtering Select	3BCh	PINMMR171[0]	PINMMR171[1]		
eQEP1S Input Filtering Select		PINMMR171[8]	PINMMR171[9]		
eQEP2A Input Filtering Select		PINMMR171[16]	PINMMR171[17]		
eQEP2B Input Filtering Select		PINMMR171[24]	PINMMR171[25]		
eQEP2I Input Filtering Select	3C0h	PINMMR172[0]	PINMMR172[1]		
eQEP2S Input Filtering Select		PINMMR172[8]	PINMMR172[9]		
ePWMx Trip Zone1 (TZ1n) Input Filtering Select		PINMMR172[16]	PINMMR172[17]	PINMMR172[18]	See Section 6.5.9
ePWMx Trip Zone2 (TZ2n) Input Filtering Select		PINMMR172[24]	PINMMR172[25]	PINMMR172[26]	
ePWMx Trip Zone3 (TZ3n) Input Filtering Select	3C4h	PINMMR173[0]	PINMMR173[1]	PINMMR173[2]	
ePWM SYNCI Input Filtering Select		PINMMR173[8]	PINMMR173[9]	PINMMR173[10]	
Temperature Sensor 1 Select		PINMMR173[16]	PINMMR173[17]		See Section 6.5.13
Temperature Sensor 2 Select		PINMMR173[24]	PINMMR173[25]		
Temperature Sensor 3 Select	3C8h	PINMMR174[0]	PINMMR174[1]		
EMIF Output Enable		PINMMR174[8]	PINMMR174[9]		See Section 6.5.2
ESM1 nERROR Select		PINMMR174[16]	PINMMR174[17]		See Section 6.5.11.1
Temperature Sensor Power Down Enable		PINMMR174[24]			See Section 6.5.13
GIOA[0] DMA Request Select	3CCh	PINMMR175[0]			See Section 6.5.12
GIOA[1] DMA Request Select		PINMMR175[8]			
GIOA[2] DMA Request Select		PINMMR175[16]			
GIOA[3] DMA Request Select		PINMMR175[24]			
GIOA[4] DMA Request Select	3D0h	PINMMR176[0]			
GIOA[5] DMA Request Select		PINMMR176[8]			
GIOA[6] DMA Request Select		PINMMR176[16]			
GIOA[7] DMA Request Select		PINMMR176[24]			
GIOB[0] DMA Request Select	3D4h	PINMMR177[0]			
GIOB[1] DMA Request Select		PINMMR177[8]			
GIOB[2] DMA Request Select		PINMMR177[16]			
GIOB3] DMA Request Select		PINMMR177[24]			
GIOB[4] DMA Request Select	3D8h	PINMMR178[0]			
GIOB[5] DMA Request Select		PINMMR178[8]			
GIOB[6] DMA Request Select		PINMMR178[16]			
GIOB[7] DMA Request Select		PINMMR178[24]			
NHET2 Pin Disable Select	3DCh	PINMMR179[0]	PINMMR179[1]		See Section 6.5.6
NHET1 Pin Disable Select		PINMMR179[8]	PINMMR179[9]		

6.5.1 Control of SDRAM Clock (EMIF_CLK)

As shown in Table 6-1, PINMMR9[0] is set by default. This blocks the EMIF SDRAM clock signal (EMIF_CLK) from being output from the microcontroller. If the EMIF is used to connect to an external SDRAM module, then the application must enable the SDRAM clock output by clearing the PINMMR9[0] bit and set the PINMMR9[1].

6.5.2 Control for other EMIF Outputs

There are some EMIF signals (EMIF_ADDR[00], EMIF_ADDR[01], EMIF_ADDR[06], EMIF_ADDR[07], EMIF_ADDR[08], EMIF_BA[1], EMIF_nCS[0], EMIF_nCS[3]), that are multiplexed with N2HET2 signals. For applications that require the use of these N2HET2 signals, it is inconvenient if the EMIF starts driving these address and control signals as output after reset is released and before the application can configure the I/O Multiplexing Module registers. Therefore, these EMIF signals are blocked from being output by default when PINMMR174[8]=1 and PINMMR174[9]=0. In this condition, these EMIF/N2HET2 terminals are configured as inputs and pulled down. An application that requires the EMIF functionality must set PINMMR174[8]=0 and PINMMR174[9]=1. This causes the EMIF address and control signals to then be output on the EMIF/N2HET2 terminals when the EMIF functionality is selected via the IOMM output multiplexing control registers.

6.5.3 Control of Ethernet Controller Mode

PINMMR160[24] is set by default. This bit is used to enable the RMII (Reduced Media Independent Interface of the Ethernet controller). If the application desires to use the MII (Media Independent Interface of the Ethernet controller), then the PINMMR160[24] must be cleared.

6.5.4 Control of ADC Trigger Events

The microcontrollers contain two Analog-to-Digital Converter modules: ADC1 and ADC2. The ADC conversions can be started using a rising or falling or both edges as the trigger event. Both the ADC modules support up to eight event trigger inputs. There are two sets of these 8 inputs for each ADC. The option for each of these 8 inputs are controlled by registers in the I/O multiplexing module as shown in Table 6-4 and Table 6-5.

Table 6-4. ADC1 Trigger Event Selection

Group Source Select, G1SRC, G2SRC or EVSRC	Event #	PINMMR161[0]	PINMMR161[1]	Control Option A	Control Option B	Trigger Source
000	1	x	x	NA	NA	AD1EVT
001	2	1	0	PINMMR161[8] = x	PINMMR161[9] = x	N2HET1[8]
		0	1	PINMMR161[8] = 1	PINMMR161[9] = 0	N2HET2[5]
		0	1	PINMMR161[8] = 0	PINMMR161[9] = 1	ePWM_B
010	3	1	0	NA	NA	N2HET1[10]
		0	1	NA	NA	N2HET1[27]
011	4	1	0	PINMMR161[16] = x	PINMMR161[17] = x	RTI1 Comp0
		0	1	PINMMR161[16] = 1	PINMMR161[17] = 0	RTI1 Comp0
		0	1	PINMMR161[16] = 0	PINMMR161[17] = 1	ePWM_A1
100	5	1	0	NA	NA	N2HET1[12]
		0	1	NA	NA	N2HET1[17]
101	6	1	0	PINMMR161[24] = x	PINMMR161[25] = x	N2HET1[14]
		0	1	PINMMR161[24] = 1	PINMMR161[25] = 0	N2HET1[19]
		0	1	PINMMR161[24] = 0	PINMMR161[25] = 1	N2HET2[1]
110	7	1	0	PINMMR162[0] = x	PINMMR162[1] = x	GIOB[0]
		0	1	PINMMR162[0] = 1	PINMMR162[1] = 0	N2HET1[11]
		0	1	PINMMR162[0] = 0	PINMMR162[1] = 1	ePWM_A2
111	8	1	0	PINMMR162[8] = x	PINMMR162[9] = x	GIOB[1]
		0	1	PINMMR162[8] = 1	PINMMR162[9] = 0	N2HET2[13]
		0	1	PINMMR162[8] = 0	PINMMR162[9] = 1	ePWM_AB

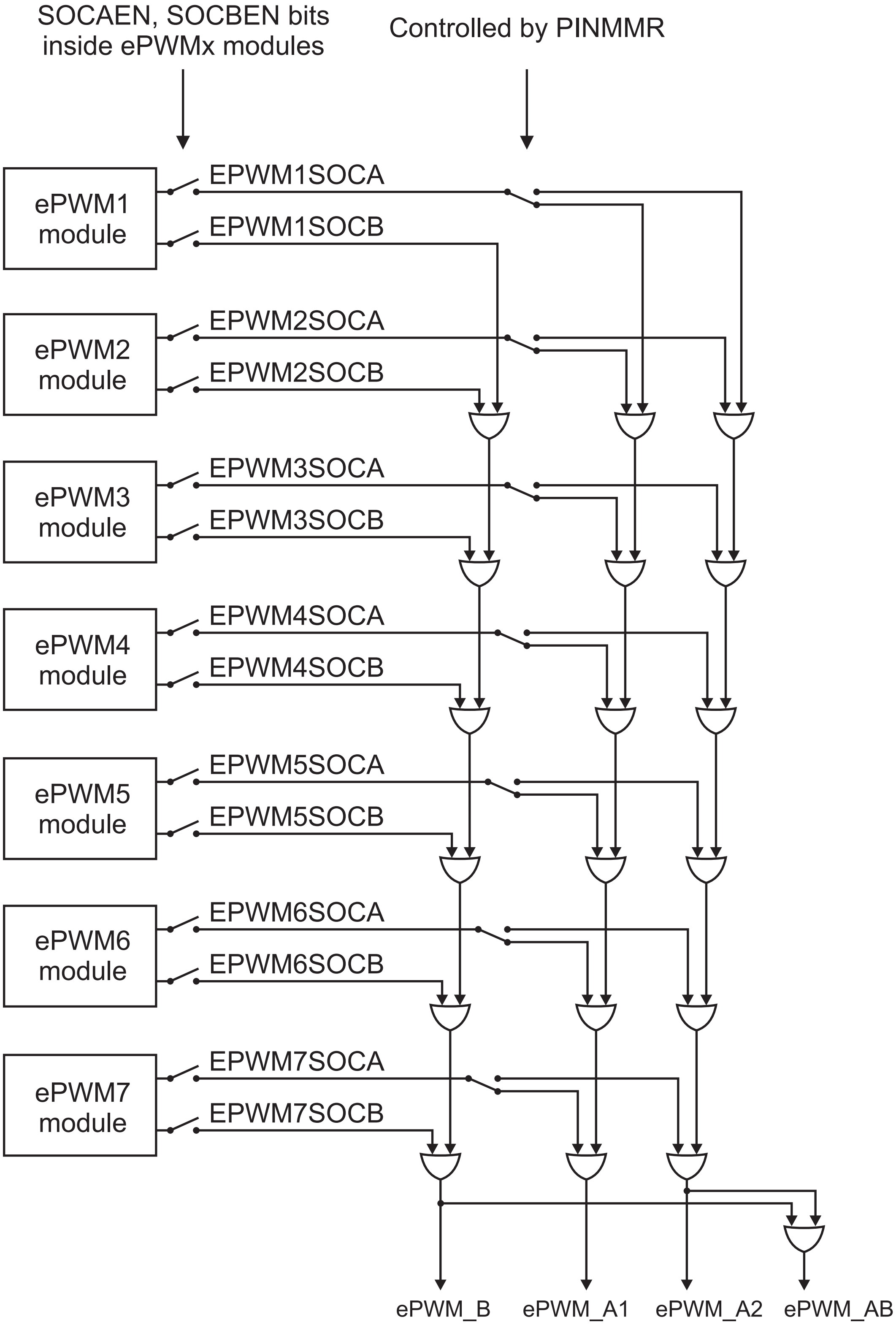
Table 6-5. ADC2 Trigger Event Selection

Group Source Select, G1SRC, G2SRC or EVSRC	Event #	PINMMR161[0]	PINMMR161[1]	Control Option A	Control Option B	Trigger Source
000	1	x	x	NA	NA	AD2EVT
001	2	1	0	PINMMR162[16] = x	PINMMR162[17] = x	N2HET1[8]
		0	1	PINMMR162[16] = 1	PINMMR162[17] = 0	N2HET2[5]
		0	1	PINMMR162[16] = 0	PINMMR162[17] = 1	ePWM_B
010	3	1	0	NA	NA	N2HET1[10]
		0	1	NA	NA	N2HET1[27]
011	4	1	0	PINMMR162[24] = x	PINMMR162[25] = x	RTI1 Comp0
		0	1	PINMMR162[24] = 1	PINMMR162[25] = 0	RTI1 Comp0
		0	1	PINMMR162[24] = 0	PINMMR162[25] = 1	ePWM_A1
100	5	1	0	NA	NA	N2HET1[12]
		0	1	NA	NA	N2HET1[17]
101	6	1	0	PINMMR163[0] = x	PINMMR163[1] = x	N2HET1[14]
		0	1	PINMMR163[0] = 1	PINMMR163[1] = 0	N2HET1[19]
		0	1	PINMMR163[0] = 0	PINMMR163[1] = 1	N2HET2[1]
110	7	1	0	PINMMR163[8] = x	PINMMR163[9] = x	GIOB[0]
		0	1	PINMMR163[8] = 1	PINMMR163[9] = 0	N2HET1[11]
		0	1	PINMMR163[8] = 0	PINMMR163[9] = 1	ePWM_A2
111	8	1	0	PINMMR163[16] = x	PINMMR163[17] = x	GIOB[1]
		0	1	PINMMR163[16] = 1	PINMMR163[17] = 0	N2HET2[13]
		0	1	PINMMR163[16] = 0	PINMMR163[17] = 1	ePWM_AB

6.5.5 Control for ADC Event Trigger Signal Generation from ePWMx Modules

This microcontroller implements 7 ePWM modules, see Figure 6-4. Each of these modules generate two outputs, SOCA (Start Of Conversion) and SOCB, for use in triggering the on-chip ADC modules. Registers from the I/O multiplexing module are used to control the logic for generation of the ePWM_A1, ePWM_A2, ePWM_AB, and ePWM_B signals from these ePWMx_SOCA and ePWMx_SOCB signals.

Figure 6-4. ADC Trigger Event Signal Generation from ePWMx



The logic equations for the 4 outputs from the combinational logic shown in Figure 6-4 are:

- B = SOC1B or SOC2B or SOC3B or SOC4B or SOC5B or SOC6B or SOC7B
- A1 = [SOC1A and not(SOC1A_SEL)] or
  [SOC2A and not(SOC2A_SEL)] or
  [SOC3A and not(SOC3A_SEL)] or
  [SOC4A and not(SOC4A_SEL)] or
  [SOC5A and not(SOC5A_SEL)] or
  [SOC6A and not(SOC6A_SEL)] or
  [SOC7A and not(SOC7A_SEL)] or
- A2 = [SOC1A and SOC1A_SEL] or
  [SOC2A and SOC2A_SEL] or
  [SOC3A and SOC3A_SEL] or
  [SOC4A and SOC4A_SEL] or
  [SOC5A and SOC5A_SEL] or
  [SOC6A and SOC6A_SEL] or
  [SOC7A and SOC7A_SEL] or
- AB = B or A2

The SOCxA_SEL signals used in the above logic equations are generated using registers in the I/O multiplexing module.

- PINMMR164[0] defines the value of SOC1A_SEL. This bit is set by default and can be cleared by the application.
- PINMMR164[8] defines the value of SOC2A_SEL. This bit is set by default and can be cleared by the application.
- PINMMR164[16] defines the value of SOC3A_SEL. This bit is set by default and can be cleared by the application.
- PINMMR164[24] defines the value of SOC4A_SEL. This bit is set by default and can be cleared by the application.
- PINMMR165[0] defines the value of SOC5A_SEL. This bit is set by default and can be cleared by the application.
- PINMMR165[8] defines the value of SOC6A_SEL. This bit is set by default and can be cleared by the application.
- PINMMR165[16] defines the value of SOC7A_SEL. This bit is set by default and can be cleared by the application.

6.5.6 Control for Generating Interrupt Upon External Fault Indication to N2HETx

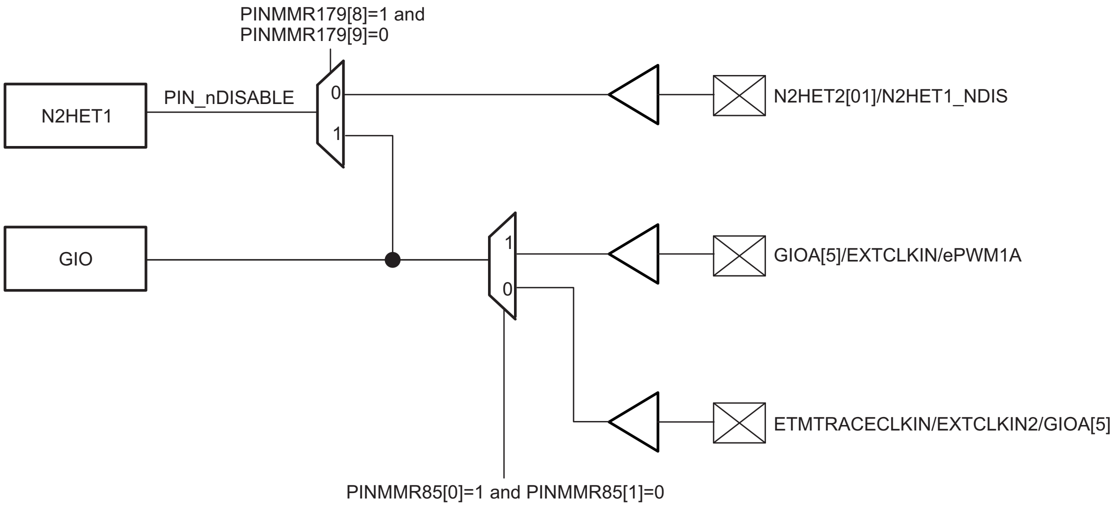
The N2HET module on this microcontroller allows the application to selectively disable any PWM output from the N2HET module whenever a fault condition is indicated to the N2HET. This fault condition is input to the N2HET module via the PIN_nDISABLE input signal. It is important for the CPU to be notified with an interrupt whenever this fault condition is indicated to the N2HET module.

The PIN_nDISABLE signal for the N2HET1 module can come from two different paths at either the GIOA[5] terminals or the N2HET2[01] terminal. By default with PINMMR179[8]=1 and PINMMR179[9]=0 the GIOA[5] / EXTCLKIN / ePWM1A terminal is selected as the input for signaling the fault condition. Setting PINMMR179[8]=0 and PINMMR179[9]=1 will select the terminal N2HET2[01] / N2HET1_NDIS for signaling the fault condition.

Note that there are two terminals from which to choose the GIOA[5] signal since GIOA[5] is available in two different terminals. By default with PINMMR85[0]=1 and PINMMR85[1]=0 the terminal shared by GIOA[5] / EXTCLKIN / ePWM1A is selected. Setting PINMMR85[0]=0 and PINMMR85[1]=1 will select the terminal shared by ETMTRACECLKIN / EXTCLKIN2 / GIOA[5]. When GIOA[5] is chosen to signal the fault condition it can also be an interrupt to the CPU if the application enables the interrupt generation whenever the GIOA[5] terminal is driven low. Figure 6-5 illustrates the multiplexing scheme.

NOTE: The default settings will choose GIOA[5] / EXTCLKIN / ePWM1A terminal for signaling the fault condition to the N2HET1 and this will be compatible to other TMS570LSxx family of microcontrollers which have this available feature.

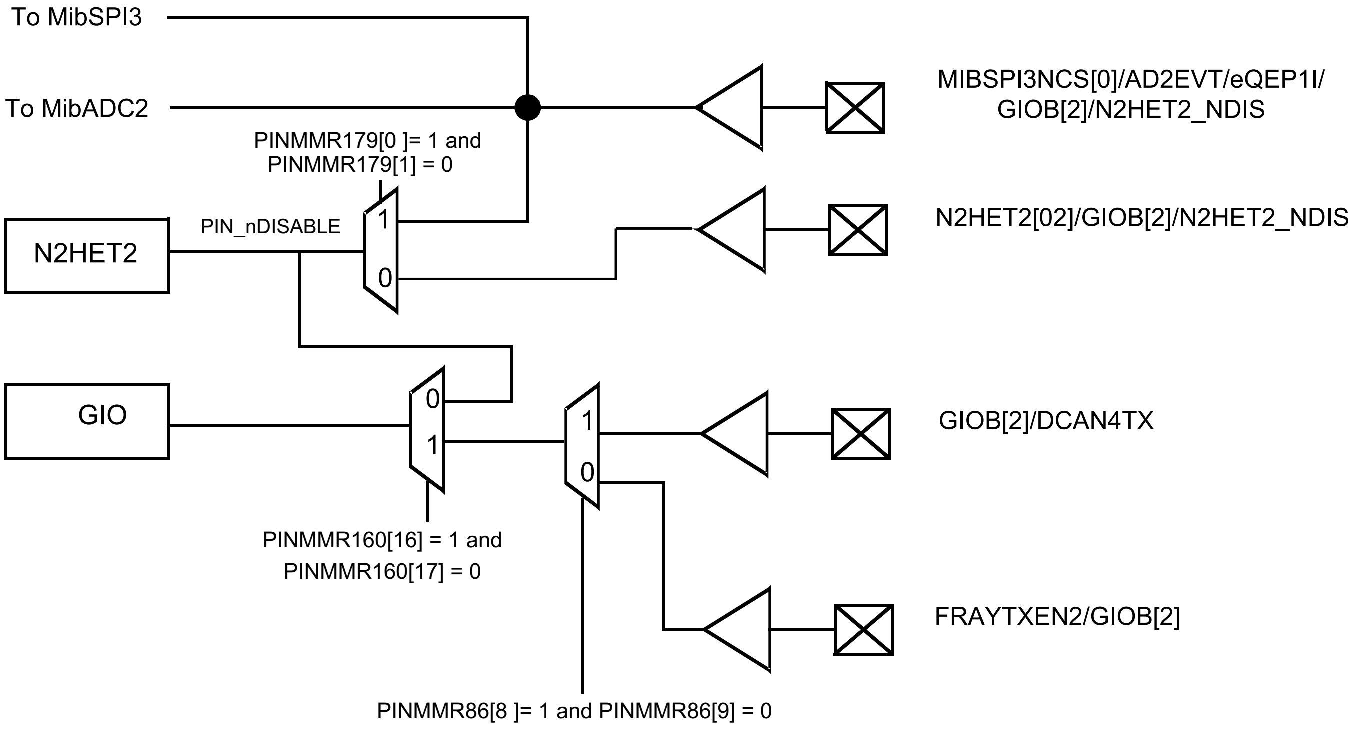
Figure 6-5. GIOA[5] and N2HET1_NDIS Input Multiplexing Scheme



The PIN_nDISABLE signal for the N2HET2 module can also come from two different paths at either the MIBSPI3NCS[0] / AD2EVT / eQEP1I / GIOB[2] / N2HET2_NDIS terminal or the N2HET2[02] / GIOB[2] / N2HET2_NDIS terminal. By default with PINMMR179[0]=1 and PINMMR179[1]=0, the MIBSPI3NCS[0] / AD2EVT / eQEP1I /GIOB[2] / N2HET2_NDIS terminal is selected as the input for signaling the fault condition. Setting PINMMR179[0]=0 and PINMMR179[1]=1 will select the N2HET2[02] / GIOB[2] / N2HET2_NDIS terminal for signaling the fault condition. By default with PINMMR160[16]=1 and PINMMR160[17]=0, these signals do not offer the capability of generating an interrupt to the GIO module when they are driven low. Therefore, the input from this terminal can optionally be connected to the GIOB[2] input. This connection is enabled by setting PINMMR160[0]=0 and PINMMR160[1]=1.

Note that the GIO module has four sources from which to choose the GIOB[2] signal. By default with PINMMR86[8]=1 and PINMMR86[9]=0, the GIOB[2] / DCAN4TX terminal is selected that is defined in Table 6-2, see register at address FFFF_1E68h. By setting PINMMR86[8]=0 and PINMMR86[9]=1, the terminal shared by FRAYTXEN2 and GIOB[2] is selected. When either one of these two terminals is selected, it is not possible to use GIO module to cause an interrupt to the CPU when a fault condition is detected. To cause an interrupt to the GIO using GIOB[2] or N2HET2_NDIS signal, the PINMMR160[16] must be clear and PINMMR160[17] must be set while either MibSPI3NCS[0] / AD2EVT /GIOB[2] terminal or the N2HET2[02] / N2HET2_NDIS terminal is connected to the external monitor circuit. Figure 6-6 illustrates the multiplexing scheme.

Figure 6-6. GIOB[2] and N2HET2_NDIS Input Multiplexing Scheme



NOTE: The default settings will choose MIBSPI3NCS[0] / AD2EVT / GIOB[2] terminal for signaling the fault condition to the N2HET2 and this will be compatible to other TMS570LSxx family of microcontrollers which have this available feature.

6.5.7 Control for Synchronizing Time Bases for All ePWMx Modules

The ePWMx modules implement a mechanism that allows their time bases to be synchronized. This is done by using a signal called TBCLKSYNC, which is a common input to all the ePWMx modules. This TBCLKSYNC is generated by a register bit in the I/O multiplexing module. PINMMR166[1] is the TBCLKSYNC signal. This bit is cleared (0) by default.

When TBCLKSYNC = 0, the time-base clock of all ePWMx modules is stopped. This is the default condition.

When TBCLKSYNC = 1, the time-base clocks of all ePWMx modules are started aligned to the rising edge of the TBCLKSYNC signal.

The correct procedure for enabling and synchronizing the time-base clocks of all the ePWMx modules is:

1. Enable the clocks to the desired individual ePWMx modules if they have been disabled
2. Set TBCLKSYNC = 0. This will stop the time-base clocks of any enabled ePWMx module.
3. Configure the time-base clock prescaler values and desired ePWM modes.
4. Set TBCLKSYNC = 1.

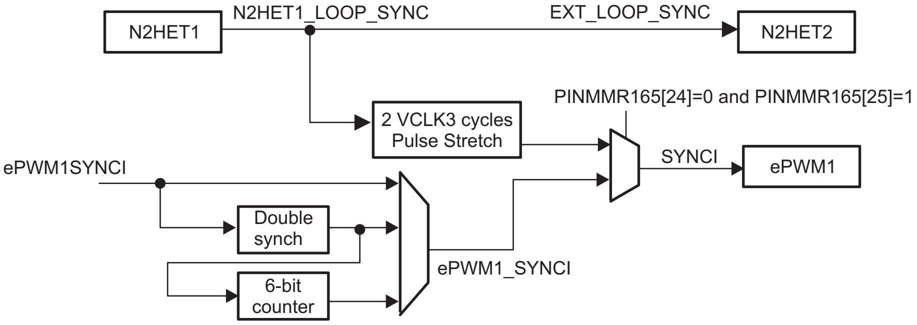
6.5.8 Control for Synchronizing all ePWMx Modules to N2HET1 Module Time-Base

Some applications require a synchronized time base for all PWM signals generated by the microcontroller. The N2HET1 module uses a time base that is created by configuring the high-resolution and loop-resolution prescalers in the N2HET1 module control registers. The N2HET1 module outputs the loop-resolution clock signal (N2HET1_LOOP_SYNC) so that other timer modules on the microcontroller can use it to synchronize their time bases to the N2HET1 loop-resolution clock.

There is a dedicated connection between the N2HET1 and N2HET2 modules, which allows the N2HET2 to use the N2HET1_LOOP_SYNC signal to synchronize its own time base to that of N2HET1.

The seven ePWMx modules can also optionally use the N2HET1_LOOP_SYNC for their time-base synchronization using a specially designed scheme.

Figure 6-7. Synchronizing ePWMx Modules to N2HET1 Time-Base



PINMMR165[24] and PINMMR165[25] are used to select between the ePWM1_SYNCI and the stretched N2HET1_LOOP_SYNC signals.

- If PINMMR165[24] = 1 and PINMMR165[25] = 0, the SYNCI input to the ePWM1 comes from the ePWM1_SYNCI which is an output from a multiplexor. This is the default connection. There are also three possible selections from which to choose the ePWM1_SYNC1.
  - When PINMMR173[8] = 1, the EPWM1SYNCI terminal is connected directly to the ePWM1 module's SYNCI port. This is the default connection.
  - When PINMMR173[8] = 0, PINMMR173[9] = 1, and PINMMR173[10] = 0, the EPWM1SYNCI terminal is double-synchronized using VCLK3 and then connected to the ePWM1 module's SYNCI port.
  - When PINMMR173[8] = 0, PINMMR173[9] = 0, and PINMMR173[10] = 1, the EPWM1SYNCI terminal is double-synchronized using VCLK3, qualified through a 6-cycle counter using VCLK3 and then connected to the ePWM1 module's SYNCI port.
- If PINMMR165[24]=0 and PINMMR165[25]=1, the SYNCI input to the ePWM1 comes from the pulse-stretched N2HET1_LOO_SYNC signal.

6.5.9 Control for Input Connections to ePWMx Modules

The ePWMx modules take the following signals as input:

- ePWM1_SYNCI: external time-base input to the ePWMx
- nTZ1 through nTZ6: trip-zone inputs to the ePWMx

Of the six trip-zone inputs, three are input from device terminals while the other three are connected to internal error events. Registers from the I/O multiplexing module are used to control various aspects of these input connections to the ePWMx modules.

Table 6-6. Controls for ePWMx Inputs

Input Signal	Control for Asynchronous Input (default connection)	Control for Double-VCLK3-Synchronized Input	Control for Double-VCLK3-Synchronized and 6-VCLK3-Filtered Input

nTZ1	PINMMR172[16] = 1	PINMMR172[16] = 0
		PINMMR172[17] = 1
			PINMMR172[17:16] = "00"
			PINMMR172[18] = 1

nTZ2	PINMMR172[24] = 1	PINMMR172[24] = 0
		PINMMR172[25] = 1
			PINMMR172[25:24] = "00"
			PINMMR172[26] = 1

nTZ3	PINMMR173[0] = 1	PINMMR173[0] = 0
		PINMMR173[1] = 1
			PINMMR173[1:0] = "00"
			PINMMR173[2] = 1

ePWM1_SYNCI	PINMMR173[8] = 1	PINMMR173[8] = 0
		PINMMR173[9] = 1
			PINMMR173[9:8] = "00"
			PINMMR173[10] = 1

Of the three internal error events for the trip-zone inputs, nTZ4 is connected to the eQEPx error output signal. There are two eQEP modules on this microcontroller, and registers in the I/O multiplexing module are used to allow a flexible scheme for the connection between the eQEPx error signal and the nTZ4 inputs of the ePWMx modules.

Table 6-7. Controls for eQEPx_ERROR Connection to ePWMx nTZ4 Inputs

ePWMx Module	Control for nTZ4 = not(eQEP1ERR or eQEP2ERR) (default connection)	Control for nTZ4 = not(eQEP1ERR)	Control for nTZ4 = not(eQEP2ERR)

ePWM1	PINMMR167[0] = 1	PINMMR167[0] = 0
		PINMMR167[1] = 1
			PINMMR167[1:0] = "00"
			PINMMR167[2] = 1

ePWM2	PINMMR167[8] = 1	PINMMR167[8] = 0
		PINMMR167[9] = 1
			PINMMR167[9:8] = "00"
			PINMMR167[10] = 1

ePWM3	PINMMR167[16] = 1	PINMMR167[16] = 0
		PINMMR167[17] = 1
			PINMMR167[17:16] = "00"
			PINMMR167[18] = 1

ePWM4	PINMMR167[24] = 1	PINMMR167[24] = 0
		PINMMR167[25] = 1
			PINMMR167[25:24] = "00"
			PINMMR167[26] = 1

ePWM5	PINMMR168[0] = 1	PINMMR168[0] = 0
		PINMMR168[1] = 1
			PINMMR168[1:0] = "00"
			PINMMR168[2] = 1

ePWM6	PINMMR168[8] = 1	PINMMR168[8] = 0
		PINMMR168[9] = 1
			PINMMR168[9:8] = "00"
			PINMMR168[10] = 1

ePWM7	PINMMR168[16] = 1	PINMMR168[16] = 0
		PINMMR168[17] = 1
			PINMMR168[17:16] = "00"
			PINMMR168[18] = 1

6.5.10 Control for Input Connections to eCAPx Modules

Each eCAPx module has a single input from the device terminals. This input can be connected to the eCAPx module in one of two ways:

1. Double-synchronized using VCLK3
2. Double-synchronized using VCLK3 and then filtered through a 6-VCLK3-cycle counter

Registers in the I/O multiplexing module are used to control these input connections for each eCAPx module.

Table 6-8. Controls for eCAPx Inputs

eCAPx Input	Control for Double-VCLK3-Synchronized Input (default connection)	Control for Double-VCLK3-Synchronized and 6-VCLK3-Filtered Input

eCAP1	PINMMR169[0] = 0	PINMMR169[0] = 0
		PINMMR169[1] = 1

eCAP2	PINMMR169[8] = 0	PINMMR169[8] = 0
		PINMMR169[9] = 1

eCAP3	PINMMR169[16] = 0	PINMMR169[16] = 0
		PINMMR169[17] = 1

eCAP4	PINMMR169[24] = 0	PINMMR169[24] = 0
		PINMMR169[25] = 1

eCAP5	PINMMR170[0] = 0	PINMMR170[0] = 0
		PINMMR170[1] = 1

eCAP6	PINMMR170[8] = 0	PINMMR170[8] = 0
		PINMMR170[9] = 1

6.5.11 Control for Input Connections to eQEPx Modules

Each eQEPx module has four inputs from the device terminals. These inputs can be connected to the eQEPx module in one of two ways:

1. Double-synchronized using VCLK3
2. Double-synchronized using VCLK3 and then filtered through a 6-VCLK3-cycle counter

Registers in the I/O multiplexing module are used to control these input connections for each eQEPx module.

Table 6-9. Controls for eQEPx Inputs

eQEPx Input	Control for Double-VCLK3-Synchronized Input (default connection)	Control for Double-VCLK3-Synchronized and 6-VCLK3-Filtered Input

eQEP1A	PINMMR170[16] = 1	PINMMR170[16] = 0
		PINMMR170[17] = 1

eQEP1B	PINMMR170[24] = 1	PINMMR170[24] = 0
		PINMMR170[25] = 1

eQEP1I	PINMMR171[0] = 1	PINMMR171[0] = 0
		PINMMR171[1] = 1

eQEP1S	PINMMR171[8] = 1	PINMMR171[8] = 0
		PINMMR171[9] = 1

eQEP2A	PINMMR171[16] = 1	PINMMR171[16] = 0
		PINMMR171[17] = 1

eQEP2B	PINMMR171[24] = 1	PINMMR171[24] = 0
		PINMMR171[25] = 1

eQEP2I	PINMMR172[0] = 1	PINMMR172[0] = 0
		PINMMR172[1] = 1

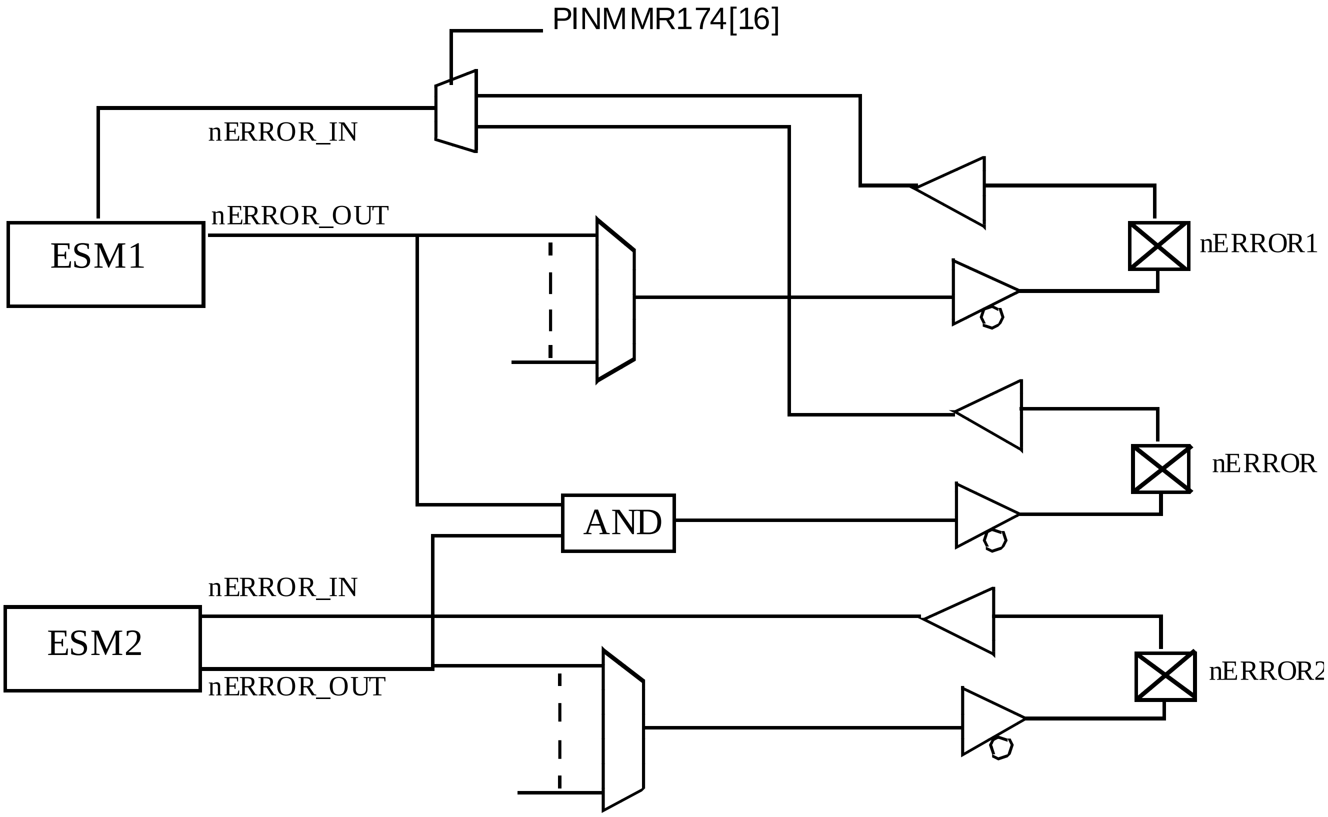
eQEP2S	PINMMR172[8] = 1	PINMMR172[8] = 0
		PINMMR172[9] = 1

6.5.11.1 nERROR and nERROR1 Input Multiplexing

There are two ESM modules in this microcontroller. When the microcontroller is in lockstep mode, only ESM1 is active.

By default, the ESM1 Error Pin Status Register (ESMEPSR) samples the nERROR pin input. The default is achieved with PINMMR174[16] = 1. By setting PINMMR174[16] = 0, the ESM1 Error Pin Status register will sample from the nERROR1 pin instead as illustrated in Figure 6-8.

Figure 6-8. nERROR and nERROR1 Input Multiplexing



6.5.12 Selecting GIO Port for External DMA Request

GIOA and GIOB ports can be used to generate external DMA request. See the datasheet for the DMA request mapping. The polarity of the GIO pin to trigger a DMA request can be selected inside the DMA module. In order to use GIO pin as an external DMA request input, the corresponding pin must be first selected by the application. By default, it is unselected. See Figure 6-9 for illustration. In this illustration, the DMQREQ[32] input to the DMA module can come from several different sources including GIOA[0]. By default with PINMMR175[0] = 1, all other sources except GIOA[0] can be selected to generate a DMA request. Care must be taken by the application that only one source is active while other sources are inactive from their respective modules. When PINMMR175[0] = 0, only GIOA[0] is selected to generate the DMA request.

Figure 6-9. Using GIO Pin for External DMA Request

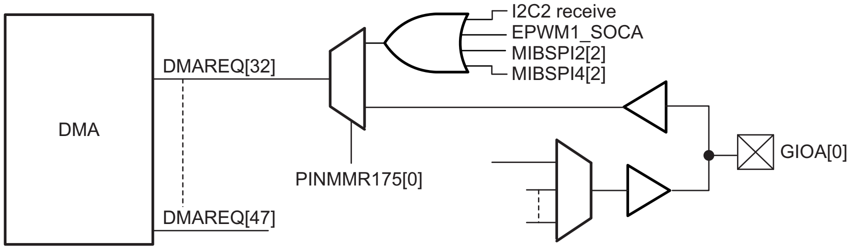


Table 6-10. GIO DMA Request Select Bit Mapping

GIO Pin	GIO DMA Request Select Bit	Bit Value to Select GIO
GIOA[0]	PINMMR175[0]	0
GIOA[1]	PINMMR175[8]	0
GIOA[2]	PINMMR175[16]	0
GIOA[3]	PINMMR175[24]	0
GIOA[4]	PINMMR176[0]	0
GIOA[5]	PINMMR176[8]	0
GIOA[6]	PINMMR176[16]	0
GIOA[7]	PINMMR176[24]	0
GIOB[0]	PINMMR177[0]	0
GIOB[1]	PINMMR177[8]	0
GIOB[2]	PINMMR177[16]	0
GIOB[3]	PINMMR177[24]	0
GIOB[4]	PINMMR178[0]	0
GIOB[5]	PINMMR178[8]	0
GIOB[6]	PINMMR178[16]	0
GIOB[7]	PINMMR178[24]	0

6.5.13 Temperature Sensor Selection

There are three instances of temperature sensors in this microcontroller. The measured temperatures are analog signals. These analog signals are connected to the on-chip ADCs for conversion. Before the temperature sensors can be used, they must be enabled. By default, they are disabled with PINMMR174[24] = 1. To enable the temperature sensors, PINMMR174[24] must be cleared to 0.

- Temperature sensor 1's output is multiplexed with AD1IN[31].
- Temperature sensor 2's output is multiplexed with AD2IN[31].
- Temperature sensor 3's output is multiplexed with AD2IN[30].

NOTE: To use AD1IN[31], PINMMR174[24] must be cleared to 0.

Table 6-11. Temperature Sensor Selection

Decode[1]	Select
AD1CHNSEL(31) = 1 PINMMR173(16) = 0 PinMMR173(17) = 1	Temp Sensor 1
AD1CHNSEL(31) = 1 PINMMR173(16) =1 PinMMR173(17) = 0	AD1IN[31]
AD2CHNSEL(31) = 1 PINMMR173(24) =0 PINMMR173(25) = 1	Temp Sensor 2
AD2CHNSEL(31) = 1 PINMMR173(24) =1 PINMMR173(25) = 0	AD2IN[31]
AD2CHNSEL(30) = 1 PINMMR174(0) =0 PINMMR174(1) = 1	Temp Sensor 3
AD2CHNSEL(30) = 1 PINMMR174(0) =1 PINMMR174(1) = 0	AD2IN[30]

[1] AD1CHNSEL is configured inside the MibADC1 Wrapper. AD2CHNSEL is configured inside the MibADC2 Wrapper.

6.6 Safety Features

The IOMM supports certain safety functions that are designed to prevent unintentional changes to the I/O multiplexing configuration. These are described in the following sections.

6.6.1 Locking Mechanism for Memory-Mapped Registers

The IOMM contains a mechanism to prevent any spurious writes from changing any of the PINMMR values. The PINMMRs are locked by default and after any system reset. None of the IOMM registers can be written under this condition. The application can read any of the IOMM registers regardless of the state of the locking mechanism.

- ***Enabling Write Access to the PINMMRs***

To enable write access to the PINMMRs, the CPU must write 0x83e70b13 to the kick0 register followed by a write of 0x95a4f1e0 to the kick1 register.

- ***Disabling Write Access to the PINMMRs***

It is recommended to disable write access to the PINMMRs once the I/O multiplexing configuration is completed. This can be done by:

- writing any other data value to either of the kick registers, or
- restarting the unlock sequence by writing 0x83e70b13 to the kick0 register

NOTE: No Error On Write to Locked PINMMRs

There is no error response on any write accesses to the PINMMRs when write access is disabled. None of the PINMMRs change state due to this write.

6.6.2 Error Conditions

The IOMM generates one error signal that is mapped to the Error Signaling Module’s Group 1, channel 37. This error signal is generated under either of the following two conditions:

- Address Error – occurs when there is a read or a write access to an un-implemented memory location within the IOMM register frame.
- Protection Error – occurs when the CPU writes to an IOMM register while not in a privileged mode of operation.

6.7 IOMM Registers

Table 6-12 lists the control registers in the IOMM. The address offset is specified from the base address of FFFF 1C00h.

Table 6-12. IOMM Registers

Offset	Acronym	Register Description	Section
00h	REVISION_REG	Revision Register	Section 6.7.1
20h	BOOT_REG	Boot Mode Register	Section 6.7.2
38h	KICK_REG0	Kicker Register 0	Section 6.7.3
3Ch	KICK_REG1	Kicker Register 1	Section 6.7.4
E0h	ERR_RAW_STATUS_REG	Error Raw Status / Set Register	Section 6.7.5
E4h	ERR_ENABLED_STATUS_REG	Error Enabled Status / Clear Register	Section 6.7.6
E8h	ERR_ENABLE_REG	Error Signaling Enable Register	Section 6.7.7
ECh	ERR_ENABLE_CLR_REG	Error Signaling Enable Clear Register	Section 6.7.8
F4h	FAULT_ADDRESS_REG	Fault Address Register	Section 6.7.9
F8h	FAULT_STATUS_REG	Fault Status Register	Section 6.7.10
FCh	FAULT_CLEAR_REG	Fault Clear Register	Section 6.7.11
110h-1A4h	PINMMRnn	Output Pin Multiplexing Control Registers	Section 6.7.12
250h-29Ch	PINMMRnn	Input Pin Multiplexing Control Registers	Section 6.7.13
390h-3DCh	PINMMRnn	Special Functionality Control Registers	Section 6.7.14

6.7.1 REVISION_REG: Revision Register

Figure 6-10. REVISION_REG: Revision Register (Offset = 00h)

31 30	29 28	27 16
REV SCHEME	Reserved	REV MODULE
R-1	R-0	R-E84h

15 11	10 8	7 6	5 0
REV RTL	REV MAJOR	REV CUSTOM	REV MINOR
R-0	R-1	R-0	R-2h

LEGEND: R = Read only; C = Clear; -*n* = value after reset

Table 6-13. Revision Register Field Descriptions

Bit	Field	Value	Description
31-30	REV SCHEME	01	Revision Scheme
29-28	Reserved	0	Reads return 0, writes have no effect.
27-16	REV MODULE	E84h	Module Id
15-11	REV RTL	0	RTL Revision
10-8	REV MAJOR	001	Major Revision
7-6	REV CUSTOM	0	Custom Revision
5-0	REV MINOR	2h	Minor Revision

6.7.2 BOOT_REG: Boot Mode Register

Figure 6-11. BOOT_REG: Boot Mode Register (Offset = 20h)

31	16
Reserved	
R-0	

15	1	0
Reserved		ENDIAN
R-0		R-D

LEGEND: R = Read only; D = Value read is determined by external configuration; -*n* = value after reset

Table 6-14. Boot Mode Register Field Descriptions

Bit	Field	Value	Description
31-1	Reserved	0	Reads return 0, writes have no effect.
0	ENDIAN		Device endianness.
		0	Device is configured in little-endian mode.
		1	Device is configured in big-endian mode.

6.7.3 KICK_REG0: Kicker Register 0

This register forms the first part of the unlock sequence for being able to update the I/O multiplexing control registers (PINMMRnn).

Figure 6-12. KICK_REG0: Kicker Register 0 (Offset = 38h)

31	16
KICK0	
R/W-0	

15	0
KICK0	
R/W-0	

LEGEND: R/W = Read/Write; -*n* = value after reset

Table 6-15. Kicker Register 0 Field Descriptions

Bit	Field	Description
31-0	KICK0	Kicker 0 Register. The value 83E7 0B13h must be written to KICK0 as part of the process to unlock the CPU write access to the PINMMRnn registers.

6.7.4 KICK_REG1: Kicker Register 1

This register forms the second part of the unlock sequence for being able to update the I/O multiplexing control registers (PINMMRnn).

Figure 6-13. KICK_REG1: Kicker Register 1 (Offset = 3Ch)

31	16
KICK1	
R/W-0	

15	0
KICK1	
R/W-0	

LEGEND: R/W = Read/Write; -*n* = value after reset

Table 6-16. Kicker Register 1 Field Descriptions

Bit	Field	Description
31-0	KICK1	Kicker 1 Register. The value 95A4 F1E0h must be written to the KICK1 as part of the process to unlock the CPU write access to the PINMMRnn registers.

6.7.5 ERR_RAW_STATUS_REG: Error Raw Status / Set Register

This register shows the status of the error conditions (before enabling) and allows setting the error status.

Figure 6-14. ERR_RAW_STATUS_REG: Error Raw Status / Set Register (Offset = E0h)

31 ... 8
Reserved
R-0

7 ... 2	1	0
Reserved	ADDR_ERR	PROT_ERR
R-0	R/WP-0	R/WP-0

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privileged mode only; -*n* = value after reset

Table 6-17. Error Raw Status / Set Register Field Descriptions

Bit	Field	Value	Description
31-2	Reserved	0	Reads return 0, writes have no effect.
1	ADDR_ERR		Addressing Error Status. An Addressing Error occurs when an unimplemented location inside the IOMM register frame is accessed.
		0	Read: Addressing Error has not occurred. Write: Writing 0 has no effect.
		1	Read: Addressing Error has been detected. Write: Addressing Error status is set.
0	PROT_ERR		Protection Error Status. A Protection Error occurs when any control register inside the IOMM is written in the CPU's user mode of operation.
		0	Read: Protection Error has not occurred. Write: Writing 0 has no effect.
		1	Read: Protection Error has been detected. Write: Protection Error status is set.

6.7.6 ERR_ENABLED_STATUS_REG: Error Enabled Status / Clear Register

This register shows the status of the error conditions and allows clearing of the error status.

Figure 6-15. ERR_ENABLED_STATUS_REG: Error Enabled Status / Clear Register (Offset = E4h)

31	8
Reserved	
R-0	

7	2	1	0
Reserved		ENABLED_ADDR_ERR	ENABLED_PROT_ERR
R-0		R/WP-0	R/WP-0

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privileged mode only; -*n* = value after reset

Table 6-18. Error Signaling Enabled Status / Clear Register Field Descriptions

Bit	Field	Value	Description
31-2	Reserved	0	Reads return 0, writes have no effect.
1	ENABLED_ADDR_ERR		Addressing Error Signaling Enable and Status Clear
		0	Read: Addressing Error Signaling is disabled. Write: Writing 0 has no effect.
		1	Read: Addressing Error Signaling is enabled. Write: Addressing Error status is cleared.
0	ENABLED_PROT_ERR		Protection Error Signaling Enable and Status Clear
		0	Read: Protection Error Signaling is disabled. Write: Writing 0 has no effect.
		1	Read: Protection Error Signaling is enabled. Write: Protection Error status is cleared.

6.7.7 ERR_ENABLE_REG: Error Signaling Enable Register

This register shows the interrupt enable status and allows enabling of the interrupts.

Figure 6-16. ERR_ENABLE_REG: Error Signaling Enable Register (Offset = E8h)

31	8
Reserved	
R-0	

7	2	1	0
Reserved		ADDR_ERR_EN	PROT_ERR_EN
R-0		R/WP-0	R/WP-0

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privileged mode only; -*n* = value after reset

Table 6-19. Error Enable Register Field Descriptions

Bit	Field	Value	Description
31-2	Reserved	0	Reads return 0, writes have no effect.
1	ADDR_ERR_EN		Addressing Error Signaling Enable
		0	Read: Addressing Error Signaling is disabled. Write: Writing 0 has no effect.
		1	Read: Addressing Error Signaling is enabled. Write: Addressing Error Signaling is enabled.
0	PROT_ERR_EN		Protection Error Signaling Enable
		0	Read: Protection Error Signaling is disabled. Write: Writing 0 has no effect.
		1	Read: Protection Error Signaling is enabled. Write: Protection Error Signaling is enabled.

6.7.8 ERR_ENABLE_CLR_REG: Error Signaling Enable Clear Register

This register shows the error signaling enable status and allows disabling of the error signaling.

Figure 6-17. ERR_ENABLE_CLR_REG: Error Signaling Enable Clear Register (Offset = ECh)

31 – 8
Reserved
R-0

7 – 2	1	0
Reserved	ADDR_ERR_EN_CLR	PROT_ERR_EN_CLR
R-0	R/WP-0	R/WP-0

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privileged mode only; -*n* = value after reset

Table 6-20. Interrupt Enable Clear Register Field Descriptions

Bit	Field	Value	Description
31-2	Reserved	0	Reads return 0, writes have no effect.
1	ADDR_ERR_EN_CLR		Addressing Error Signaling Enable Clear
		0	Read: Addressing Error signaling is disabled. Write: Writing 0 has no effect.
		1	Read: Addressing Error signaling is enabled. Write: Addressing Error signaling is disabled.
0	PROT_ERR_EN_CLR		Protection Error Signaling Enable Clear
		0	Read: Protection Error signaling is disabled. Write: Writing 0 has no effect.
		1	Read: Protection Error signaling is enabled. Write: Protection Error signaling is disabled.

6.7.9 FAULT_ADDRESS_REG: Fault Address Register

This register holds the address offset of the first fault transfer.

Figure 6-18. FAULT_ADDRESS_REG: Fault Address Register (Offset = F4h)

31 – 16
Reserved
R-0

15 – 9	8 – 0
Reserved	FAULT_ADDR
R-0	R/WP-0

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privileged mode only; -*n* = value after reset

Table 6-21. Fault Address Register Field Descriptions

Bit	Field	Value	Description
31-9	Reserved	0	Reads return 0, writes have no effect.
8-0	FAULT_ADDR		Fault Address. The fault address offset in case of an address error or a protection error condition.

6.7.10 FAULT_STATUS_REG: Fault Status Register

This register holds the status and attributes of the first fault transfer.

Figure 6-19. FAULT_STATUS_REG: Fault Status Register (Offset = F8h)

31 28	27 24	23 16
Reserved	FAULT_ID	FAULT_MSTID
R-0	R-0	R-0

15 13	12 9	8	7	6	5 0
Reserved	FAULT_PRIVID	Rsvd	FAULT_NS	Rsvd	FAULT_TYPE
R-0	R-0	R-0	R-0	R-0	R-0

LEGEND: R = Read only; -*n* = value after reset

Table 6-22. Fault Status Register Field Descriptions

Bit	Field	Value	Description
31-28	Reserved	0	Reads return 0, writes have no effect.
27-24	FAULT_ID		Faulting Transaction ID
23-16	FAULT_MSTID		ID of Master that initiated the faulting transaction
15-13	Reserved	0	Reads return 0, writes have no effect.
12-9	FAULT_PRIVID		Faulting Privilege ID
8	Reserved	0	Reads return 0, writes have no effect.
7	FAULT_NS		Fault: Non-secure access detected
6	Reserved	0	Reads return 0, writes have no effect.
5-0	FAULT_TYPE		Type of fault detected.
		0	No fault
		1h	User execute fault
		2h	User write fault
		4h	User read fault
		8h	Supervisor execute fault
		10h	Supervisor write fault
		20h	Supervisor read fault

6.7.11 FAULT_CLEAR_REG: Fault Clear Register

This register allows the application to clear the current fault so that another can be captured when 1 is written to this register.

Figure 6-20. FAULT_CLEAR_REG: Fault Clear Register (Offset = FCh)

31–16
Reserved
R-0

15–1	0
Reserved	FAULT_CLEAR
R-0	R/WP-0

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privileged mode only; -*n* = value after reset

Table 6-23. FAULT_CLEAR_REG: Fault Clear Register Field Descriptions

Bit	Field	Value	Description
31-1	Reserved	0	Reads return 0, writes have no effect.
0	FAULT_CLEAR		Fault Clear
		0	Read: Current value of the FAULT_CLEAR bit is 0. Write: Writing 0 has no effect.
		1	Read: Current value of the FAULT_CLEAR bit is 1. Write: Writing a 1 clears the current fault.

6.7.12 PINMMRnn: Output Pin Multiplexing Control Registers

These registers control the output multiplexing of the functionality available on each pad on the microcontroller. There are 38 such registers – PINMMR0 through PINMMR37. Each 8-bit field of a PINMMR register controls the functionality of a single ball/pin. The mapping between the PINMMRx control registers and the functionality selected on a given terminal is defined in Table 6-1.

Figure 6-21. PINMMRnn: Pin Multiplexing Control Registers (Offset = 110h-1A4h)

31–24	23–16
PINMMRx[31-24]	PINMMRx[23-16]
R/WP-1	R/WP-1

15–8	7–0
PINMMRx[15-8]	PINMMRx[7-0]
R/WP-1	R/WP-1

LEGEND: R/W = Read/Write; WP = Write in privileged mode only; -*n* = value after reset

Table 6-24. Pin Multiplexing Control Registers Field Descriptions

Bit	Field	Value	Description
31-24	PINMMRx[31-24]	1h	Each of these byte-fields control the functionality on a given ball/pin. Please refer to Table 6-1 for a list of multiplexed signals.
23-16	PINMMRx[23-16]	1h	
15-8	PINMMRx[15-8]	1h	
7-0	PINMMRx[7-0]	1h	

6.7.13 PINMMRnn: Input Pin Multiplexing Control Registers

These registers control the input multiplexing of the functionality available on each pad on the microcontroller. There are 20 such registers – PINMMR80 through PINMMR99. Each 8-bit field of a PINMMR register controls the functionality of a single ball/pin. The mapping between the PINMMRx control registers and the functionality selected on a given terminal is defined in Table 6-2.

Figure 6-22. PINMMRnn: Pin Multiplexing Control Registers (Offset = 250h-29Ch)

31 24	23 16
PINMMRx[31-24]	PINMMRx[23-16]
R/WP-1	R/WP-1

15 8	7 0
PINMMRx[15-8]	PINMMRx[7-0]
R/WP-1	R/WP-1

LEGEND: R/W = Read/Write; WP = Write in privileged mode only; -*n* = value after reset

Table 6-25. Pin Multiplexing Control Registers Field Descriptions

Bit	Field	Value	Description
31-24	PINMMRx[31-24]	1h	Each of these byte-fields control the functionality on a given ball/pin. Please refer to Table 6-2 for a list of multiplexed signals.
23-16	PINMMRx[23-16]	1h	
15-8	PINMMRx[15-8]	1h	
7-0	PINMMRx[7-0]	1h	

6.7.14 PINMMRnn: Special Functionality Multiplexing Control Registers

These registers control the special functionalities on the microcontroller. There are 20 such registers – PINMMR160 through PINMMR179. Each 8-bit field of a PINMMR register controls one special functionality. The mapping between the PINMMRx control registers and the functionality selected on a given terminal is defined in Table 6-3.

Figure 6-23. PINMMRnn: Pin Multiplexing Control Registers (Offset = 390h-3DCh)

31 24	23 16
PINMMRx[31-24]	PINMMRx[23-16]
R/WP-1	R/WP-1

15 8	7 0
PINMMRx[15-8]	PINMMRx[7-0]
R/WP-1	R/WP-1

LEGEND: R/W = Read/Write; WP = Write in privileged mode only; -*n* = value after reset

Table 6-26. Pin Multiplexing Control Registers Field Descriptions

Bit	Field	Value	Description
31-24	PINMMRx[31-24]	1h	Each of these byte-fields control the functionality on a given ball/pin. Please refer to Table 6-3 for a list of multiplexed signals.
23-16	PINMMRx[23-16]	1h	
15-8	PINMMRx[15-8]	1h	
7-0	PINMMRx[7-0]	1h	

F021 Level 2 Flash Module Controller (L2FMC)

The Flash electrically-erasable programmable read-only memory module is a type of nonvolatile memory that has fast read access times and is able to be reprogrammed in the field or in the application. It also allows remapping of the Flash to RAM spaces in order to save on repeated program/erase cycles. This chapter describes the Level 2 F021 Flash module controller (L2FMC).

Topic **Page**

7.1 Overview

The F021 Flash is used to provide non-volatile memory for instruction execution or data storage. The Flash can be electrically programmed and erased many times to ease code development.

Refer to the following documents for support on how to initialize and use the on-chip Flash and its API:

- *F021 (Texas Instruments 65nm Flash) Flash API Reference Guide* (SPNU501)

7.1.1 Features

- Symmetric dual port (Port A and Port B) for higher performance and concurrent access to different banks from one or more bus masters.
- Read, program and erase with a single 3.3 V supply voltage
- Supports error detection and correction
  - Single Error Correction and Double Error Detection (SECDED)
  - Error Correction Code (ECC) is evaluated in the CPU.
  - Address bits included in ECC calculation
- Provides different read modes to optimize performance and verify the integrity of Flash contents
- Provides software controllable power mode control logic
- Integrated program/erase state machine
  - Simplifies software algorithms
  - Supports simultaneous read access on up to two banks while performing a write or erase operation on any one of the remaining banks
  - Suspend command allows read access to a sector being programmed/erased
  - Fast erase and program times (for details, see the device-specific data sheet
- Allows remapping of Flash to RAM spaces through "Parameter Overlay Module" (POM)

For the actual size of the Flash memory for the device, see the device-specific data sheet.

7.1.2 Definition of Terms

Terms used in this document have the following meaning:

- bw - Normal data space bank data width of a Flash bank. The bw is 256 bits (288 bits including the error correction bits).
- bwe - EEPROM emulation bank is 64-bit wide (72 bits including the error correction bits).
- Bus Master - Any of CPU, DMA or other modules which can request data access.
- Charge pump: Voltage generators and associated control (logic, oscillator, and bandgap, for example).
- CSM: Program/erase command state machine
- Flash bank: A group of Flash sectors that share input/output buffers, data paths, sense amplifiers, and control logic.
- FEE - Flash EEPROM Emulation. Features on the L2FMC to support using a Flash type memory in place of an EEPROM Flash memory. EEPROM is erasable by the word while this Flash memory is only erasable by the sector. The FEE bank is accessible through the same bus as the main bank (in a special address range) and always resides in bank 7.
- Flash module: Flash banks, charge pump, and Flash wrapper.
- Flash wrapper: Power and mode control logic, data path, wait logic, and write/erase state machines.
- L2FMC - Level 2 Flash Module Controller.
- Command - A sequence of coded instructions to Flash module to execute a certain task.
- FSM (Flash State Machine) - State machine that parses and decodes FSM commands. It executes embedded algorithms and generates control signals to both Flash bank and charge pump during the actual program/erase operation.

- OTP (one-time programmable): A program-only-once Flash sector (cannot be erased)
- Sector: A contiguous region of Flash memory that must be erased simultaneously.
- Wide_Word - the width of the data output from the Flash bank. This is 288-bits wide for main Flash banks and 72-bits wide for the FEE bank.
- Prefetch Mode - Provides higher performance by fetching the subsequent cache line ahead of the actual request.
- Read Margin 1 mode: More stringent read mode designed for early detection of marginally erased bits.
- Read Margin 0 mode: More stringent read mode designed for early detection of marginally programmed bits.
- Implicit read - At startup the L2FMC performs multiple automatic reads from OTP to read device settings.
- Bus Error - L2FMC will generate a bus error to the bus master on certain accesses for example, writes to Flash on Port A/Port B or access to addresses beyond the available Flash space.
- POM - Parameter Overlay Module provides a method to remap the Flash when there is a need to have different values in the Flash contents without actually erasing and reprogramming the Flash.

7.1.3 F021 Flash Tools

Texas Instruments provides the following tools for F021 Flash:

- nowECC Generation Tool - to generate the Flash ECC from the Flash data.
- UNIFLASH Programming Tool - to erase/program/verify the device Flash content through JTAG.
- Code Composer Studio - the development environment with integrated Flash programming capabilities.
- F021 Flash API Library - a set of software peripheral functions to program/erase the Flash module. Refer to *F021 Flash API Reference Guide* (SPNU501) for more information.

7.2 Default Flash Configuration

At power up, the Flash module state exhibits the following properties:

- Wait states are set to 1 data wait state. An implicit address wait states are set to 1 and cannot be changed.
- Prefetch mode is enabled
- The Flash content is protected from modification
- Power modes are set to *Active* (no power savings)
- The boot code must initialize the wait states and the desired prefetch mode by initializing the FRDCNTL register to achieve the optimum system performance. This needs to be done before switching to the final device operating frequency.

7.3 EEPROM Emulation Support

Several features of the L2FMC support EEPROM emulation. They are listed here.

- In order to allow zeroing out used portions of Flash when the table has to be moved to a new block, L2FMC allows replacing the all-zero ECC with the correct ECC value of an all-zero 64b data. This is enabled by setting the EE_FEDACCTRL1.EZCV bit at address offset 8h.
- Similarly, in order to be able to read the Flash after successfully erasing it, L2FMC will compute the correct ECC for all-ones 64b data. This is enabled by setting the EE_FEDACCTRL1.EOCV bit at address offset 8h.
- Normally, for ECC to correctly work all the 64b of data must be programmed into the Flash. However, it is not uncommon to program partial words in the EEPROM Emulation bank. In order to allow this, L2FMC provides the FEDACSDIS and FEDACSDIS2 registers that identifies up to 4 chosen sectors where partial words may be programmed. In such a case, L2FMC computes ECC on the fly for these sectors thus avoiding any errors.

7.4 SECDED

The Flash memory can be protected by Single Error Correction Double Error Detection (SECDED). This protection is enabled by the SECDED circuit inside of the bus master.

7.4.1 SECDED Initialization

Flash error detection and correction is enabled at reset.

The ECC values for all of the Flash memory space (Flash banks 0 through 6) must be programmed into the Flash before the program/data can be read. This can be done by generating the correct values of the ECC with an external tool such as nowECC or may be generated by the programming tool. The Cortex R5F CPU may generate speculative fetches to any location within the Flash memory space. A speculative fetch to a location with invalid ECC, which is subsequently not used, will not create an abort, but will set the ESM flags for a correctable or uncorrectable error. An uncorrectable error will unconditionally cause the nERROR pin to toggle low. Therefore care must be taken to generate the correct ECC for the entire Flash space including the holes between sections and any unused or blank Flash areas.

The Cortex R5F CPU does not generate speculative fetches into the address space of bank 7, the EEPROM Emulation Flash. Therefore, it is only necessary to initialize the ECC values of the locations that will be intentionally read by the CPU or other bus masters.

7.4.2 ECC Encoding

Twenty-nine address lines are also included in the ECC calculation. A failure of a single address line inside of the bank will result in an uncorrectable error at the bus master. The ECC encoding is shown in Table 7-1.

Table 7-1. ECC Encoding for BE32 Devices

		92	91	90	89	88	87	86	85	84	83	82	81	80	79	78	77	76	75	74	73	72	71	70	69	68	67	66	65	64
		Participating Address Bits																												
ADDR_MSW_LSW	**ECC**	**31**	**30**	**29**	**28**	**27**	**26**	**25**	**24**	**23**	**22**	**21**	**20**	**19**	**18**	**17**	**16**	**15**	**14**	**13**	**12**	**11**	**10**	**09**	**08**	**07**	**06**	**05**	**04**	**03**
1FC0007F_00FFFF00_FF0000FF	7	x	x	x	x	x	x	x																x	x	x	x	x	x	x
3FFF80_FF0000FF_FF0000FF	6								x	x	x	x	x	x	x	x	x	x	x	x	x	x	x							
1FC07F80_00FF00FF_00FF00FF	5	x	x	x	x	x	x	x								x	x	x	x	x	x	x	x							
FC19F83_FCC0FCC0_FCC0FCC0	4		x	x	x	x	x	x						x	x			x	x	x	x	x	x						x	x
13C6A78D_E338E338_E338E338	3	x			x	x	x	x				x	x		x		x			x	x	x	x				x	x		x
14DAA9B5_99A699A6_99A699A6	2	x		x			x	x		x	x		x		x		x		x			x	x		x	x		x		x
1D68BAD1_57155715_57155715	1	x	x	x		x		x	x		x				x		x	x	x		x		x	x		x				x
A7554EA_D1B4D1B4_2E4B2E4B	0		x		x			x	x	x		x		x		x		x		x			x	x	x		x		x	

Participating Data Bits																																				
63	**62**	**61**	**60**	**59**	**58**	**57**	**56**	**55**	**54**	**53**	**52**	**51**	**50**	**49**	**48**	**47**	**46**	**45**	**44**	**43**	**42**	**41**	**40**	**39**	**38**	**37**	**36**	**35**	**34**	**33**	**32**	**31**	**30**	**29**	**28**	**27**
								x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x									x	x	x	x	x
x	x	x	x	x	x	x	x																	x	x	x	x	x	x	x	x	x	x	x	x	x
								x	x	x	x	x	x	x	x									x	x	x	x	x	x	x	x					
x	x	x	x	x	x			x	x							x	x	x	x	x	x			x	x							x	x	x	x	x
x	x	x				x	x			x	x	x				x	x	x				x	x			x	x	x				x	x	x		
x			x	x			x	x		x			x	x		x			x	x			x	x		x			x	x		x			x	x
	x		x		x	x	x				x		x		x		x		x		x	x	x				x		x		x		x		x	
x	x		x				x	x		x	x		x			x	x		x				x	x		x	x		x					x		x

Participating Data Bits																											**Parity**[1]	**Check Bits**[2]
26	**25**	**24**	**23**	**22**	**21**	**20**	**19**	**18**	**17**	**16**	**15**	**14**	**13**	**12**	**11**	**10**	**09**	**08**	**07**	**06**	**05**	**04**	**03**	**02**	**01**	**00**		
x	x	x																	x	x	x	x	x	x	x	x	Even	ECC[7]
x	x	x																	x	x	x	x	x	x	x	x	Even	ECC[6]
			x	x	x	x	x	x	x	x									x	x	x	x	x	x	x	x	Even	ECC[5]
x			x	x							x	x	x	x	x	x			x	x							Even	ECC[4]
	x	x			x	x	x				x	x	x				x	x			x	x	x				Odd	ECC[3]
		x	x		x			x	x		x			x	x			x	x		x			x	x		Odd	ECC[2]
x	x	x				x		x		x		x		x		x	x	x				x		x		x	Even	ECC[1]
x	x			x			x		x	x			x		x	x	x			x			x		x	x	Even	ECC[0]

[1] For Odd parity, XOR a 1 to the row's XOR result. For even Parity, use the row's XOR result directly.

[2] Each ECC[x] bit represents the XOR of all the address and data bits marked with x in the same row.

7.4.3 Syndrome Table: Decode to Bit in Error

The syndrome is an 8-bit value that decodes to the bit in error. The bit in error can be a bit among the 64 data bits or a bit among the 8 ECC check bits. A syndrome value of 0000 0000 indicates there is no error. Any other syndrome combinations not shown in the table are uncorrectable multi-bit error. Errors of three of more bits may escape detection. The syndrome decoding is shown in Table 7-2.

Table 7-2. Syndrome Table

Data Bit Error Position																																				
63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32	31	30	29	28	27
0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	1	1	1	1	1
1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	1	1	1	1	1
1	1	0	0	0	0	0	0	1	1	1	1	1	1	0	0	1	1	0	0	0	0	0	0	1	1	1	1	1	1	0	0	1	1	0	0	0
0	0	1	1	1	0	0	0	1	1	1	0	0	0	1	1	0	0	1	1	1	0	0	0	1	1	1	0	0	0	1	1	0	0	1	1	1
1	0	1	0	0	1	1	0	1	0	0	1	1	0	0	1	1	0	1	0	0	1	1	0	1	0	0	1	1	0	0	1	1	0	1	0	0
0	0	0	1	0	1	0	1	0	1	0	1	0	1	1	1	0	0	0	1	0	1	0	1	0	1	0	1	0	1	1	1	0	0	0	1	0
1	0	1	1	0	1	0	0	1	1	0	1	0	0	0	1	1	0	1	1	0	1	0	0	1	1	0	1	0	0	0	1	0	1	0	0	1

Data Bit Error Position																											ECC Error Bit								
26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00	07	06	05	04	03	02	01	00	
1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	Bit[7]
1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	0	1	0	0	0	0	0	0	Bit[6]
1	1	1	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	Bit[5]
0	0	0	1	1	1	1	1	1	0	0	1	1	0	0	0	0	0	0	1	1	1	1	1	1	0	0	0	0	0	1	0	0	0	0	Bit[4]
0	0	0	1	1	1	0	0	0	1	1	0	0	1	1	1	0	0	0	1	1	1	0	0	0	1	1	0	0	0	0	1	0	0	0	Bit[3]
1	1	0	1	0	0	1	1	0	0	1	1	0	1	0	0	1	1	0	1	0	0	1	1	0	0	1	0	0	0	0	0	1	0	0	Bit[2]
1	0	1	0	1	0	1	0	1	1	1	0	0	0	1	0	1	0	1	0	1	0	1	0	1	1	1	0	0	0	0	0	0	1	0	Bit1]
0	1	1	0	0	1	0	1	1	1	0	0	1	0	0	1	0	1	1	0	0	1	0	1	1	1	0	0	0	0	0	0	0	0	1	Bit[0]

7.4.4 Syndrome Table: An Alternate Method

Table 7-3. Alternate Syndrome Table

Syndrome LSB: 3:0	Syndrome MSB 7:4															
	0x	**1x**	**2x**	**3x**	**4x**	**5x**	**6x**	**7x**	**8x**	**9x**	**Ax**	**Bx**	**Cx**	**Dx**	**Ex**	**Fx**
x0	good	E04	E05	D	E06	D	D	D62	E07	D	D	D46	D	M	M	D
x1	E00	D	D	D14	D	M	M	D	D	M	M	D	M	D	D	D30
x2	E01	D	D	M	D	D34	D56	D	D	D50	D40	D	M	D	D	M
x3	D	D18	D08	D	M	D	D	M	M	D	D	M	D	D02	D24	D
x4	E02	D	D	D15	D	D35	D57	D	D	D51	D41	D	M	D	D	D31
x5	D	D19	D09	D	M	D	D	D63	M	D	D	D47	D	D03	D25	D
x6	D	D20	D10	D	M	D	D	M	M	D	D	M	D	D04	D26	D
x7	M	D	D	M	D	D36	D58	D	D	D52	D42	D	M	D	D	M
x8	E03	D	D	M	D	D37	D59	D	D	D53	D43	D	M	D	D	M
x9	D	D21	D11	D	M	D	D	M	M	D	D	M	D	D05	D27	D
xA	D	D22	D12	D	D33	D	D	M	D49	D	D	M	D	D06	D28	D
xB	D17	D	D	M	D	D38	D60	D	D	D54	D44	D	D01	D	D	M
xC	D	D23	D13	D	M	D	D	M	M	D	D	M	D	D07	D29	D
xD	M	D	D	M	D	D39	D61	D	D	D55	D45	D	M	D	D	M
xE	D16	D	D	M	D	M	M	D	D	M	M	D	D00	D	D	M
xF	D	M	M	D	D32	D	D	M	D48	D	D	M	D	M	M	D

- E0x - Single-bit ECC error, correctable
- Dxx - Single-bit data error, correctable
- D - Double-bit error, uncorrectable
- M - Multi-bit errors, uncorrectable

7.5 Memory Map

The Flash module contains the program memory, which is mapped starting at location 0, and one Customer OTP sector and one TI OTP sector per bank. The Customer OTP sectors may be programmed by the customer, but cannot be erased. They are typically blank in new parts. The TI OTP sectors are used to contain manufacturing information. They may be read by the customer but can not be programmed or erased. The TI OTP sectors contain settings used by the Flash API to setup the Flash state machine for erase and program operations.

All of these OTP regions are memory-mapped to facilitate ease of access by the CPU. They are memory-mapped to an offset starting at F000 0000h in the CPUs memory map.

The RWAIT value is used to define the number of wait states for the program memory Flash. The EWAIT value is used to define the number of wait states for the data Flash in bank 7. Bank 7 starting at offset F020 0000h is dedicated for data storage such as EEPROM Emulation.

7.5.1 Location of Flash ECC Bits

The ECC bits are packed in their memory space as shown in Figure 7-1 and Figure 7-2.

NOTE: Unlike previous versions of this module, all the ECC bytes corresponding to the address and size of access are returned. For example, if a Load Multiple (LDM) was used to fetch 32 bytes of ECC, all of the actual bytes corresponding to the range of the access are returned. There is no replication of the bytes returned.

Figure 7-1. ECC Organization for Bank 0-1 (288-Bits Wide)

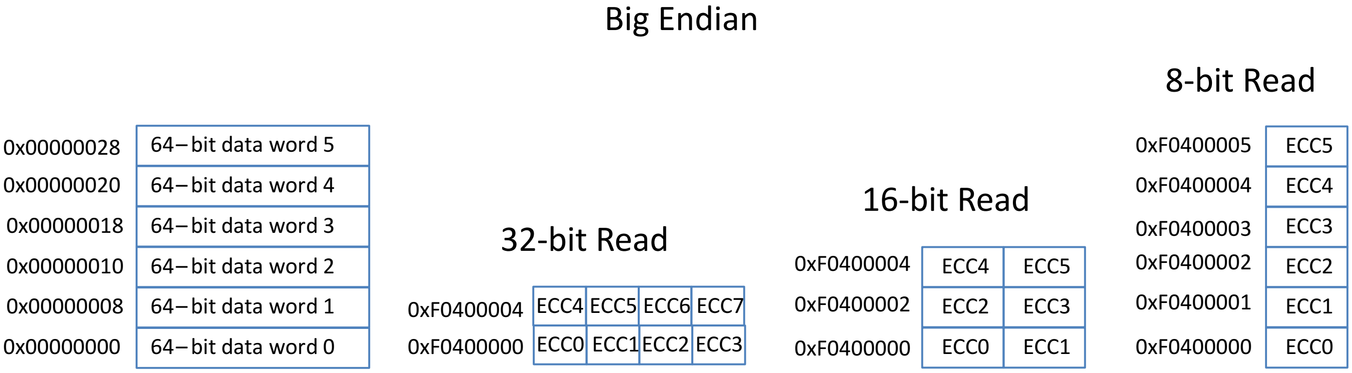
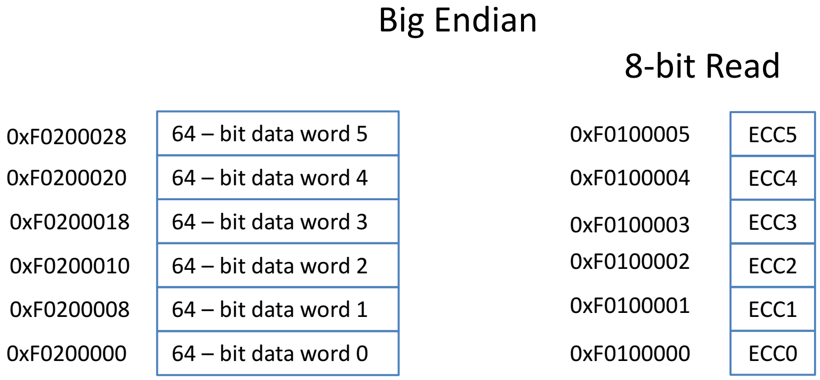


Figure 7-2. ECC Organization for Bank 7 (72-Bits Wide)



7.5.2 OTP Memory

7.5.2.1 Flash Bank and Sector Sizes

Flash Bank/Sectoring information can be determined from the device-specific datasheet or can be computed by reading locations in the TI OTP and L2FMC registers.

The number of banks, which banks are available, and the number of sectors for bank 0 can be read from TI OTP location F008 0158h as shown in Figure 7-3 and described in Table 7-4.

Figure 7-3. TI OTP Bank 0 Sector Information

31	24	23	16
Reserved		BX_NUM_Sectors	
R		R	

15							8	7	0
B7	B6	B5	B4	B3	B2	B1	B0	NUM_Banks	
R-1	R-0	R-0	R-0	R-0	R-0	R-1	R-1	R	

LEGEND: R = Read only

Table 7-4. TI OTP Bank 0 Sector Information Field Descriptions

Bit	Field	Value	Description
31-24	Reserved	0	Reserved. All bits will be read as 0.
23-16	BX_NUM_Sectors	1-32	Number of sectors in this bank.
15	B7	1	1 = Bank 7 is present
14	B6	0	0 = Bank 6 is not present
13	B5	0	0 = Bank 5 is not present
12	B4	0	0 = Bank 4 is not present
11	B3	0	0 = Bank 3 is not present
10	B2	0	0 = Bank 2 is not present
9	B1	1	1 = Bank 1 is present
8	B0	1	1 = Bank 0 is present
7-0	NUM_Banks	3	Number of banks on this part.

The bank sector information is repeated once for each bank in the device. The number of sectors is unique for each bank. The number of banks and which banks are implemented is repeated in each location. Use the TI OTP information for bank 0 to determine which banks are in the device, and then read the number of sectors for each bank using the TI OTP locations shown in Table 7-5.

Table 7-5. TI OTP Sector Information Address

Bank	TI OTP Address
0	F008 0158h
1	F008 2158h
2	F008 4158h
3	F008 6158h
4	F008 8158h
5	F008 A158h
6	F008 C158h
7	F008 E158h

7.5.2.2 Package and Memory Size

Package and memory size information can be determined from the device-specific datasheet, or can be computed by reading locations in the TI OTP Bank 0 registers.

The package and memory size can be read from TI OTP location F008 015Ch as shown in Figure 7-4 and described in Table 7-6.

Figure 7-4. TI OTP Bank 0 Package and Memory Size Information

31 28	27 16
Reserved	PACKAGE
R	R

15 0
MEMORY_SIZE
R

LEGEND: R = Read only

Table 7-6. TI OTP Bank 0 Package and Memory Size Information Field Descriptions

Bit	Field	Description
31-28	Reserved	Reserved
27-16	PACKAGE	Count of pins in the package.
15-0	MEMORY_SIZE	Flash memory size in Kbytes.

7.5.2.3 LPO Trim and Max HCLK

The HF LPO trim solution, LF LPO trim solution and maximum GCLK1 frequency can be read from TI OTP location F008 01B4h as shown in Figure 7-5 and described in Table 7-7.

Figure 7-5. TI OTP Bank 0 LPO Trim and Max HCLK Information

31 24	23 16
HFLPO_TRIM	LFLPO_TRIM
R	R

15 0
MAX_GCLK
R

LEGEND: R = Read only

Table 7-7. TI OTP Bank 0 LPO Trim and Max HCLK Information Field Descriptions

Bit	Field	Description
31-24	HFLPO_TRIM	HF LPO Trim Solution
23-16	LFLPO_TRIM	LF LPO Trim Solution
15-0	MAX_GCLK	Maximum GCLK1 Speed

7.5.2.4 Part Number Symbolization

Device part number symbolization information can be determined from the device-specific data manual or can be computed by reading locations in the TI OTP bank 0 registers.

For example, the device part number symbolization "TMS570LC4357AZWTQQ1" can be read from TI OTP bank 0 location F008 01E0h through F008 01FFh as shown in Figure 7-6. The part number is stored as a null terminated ASCII string.

Figure 7-6. TI OTP Bank 0 Symbolization Information (F008 01E0h-F008 01FFh)

0x00	0x01	0x02	0x03	0x04	0x05	0x06	0x07	0x08	0x09	0x0A	0x0B	0x0C	0x0D	0x0E	0x0F
0x54	0x4D	0x53	0x35	0x37	0x30	0x4C	0x43	0x34	0x33	0x35	0x37	0x41	0x5A	0x57	0x54
R															

0x10	0x11	0x12	0x13	0x14	0x15	0x16	0x17	0x18	0x19	0x1A	0x1B	0x1C	0x1D	0x1E	0x1F
0x51	0x51	0x31	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00
R															

LEGEND: R = Read only

7.5.2.5 Temperature Sensor

There are three temperature sensors that can be used to read the internal junction temperature on this device. The temperature sensors are connected to the ADC converter. See Section 6.5.13 for information on how to select the temperature sensors. During device testing, the value read from the ADC along with the junction temperature of the silicon was recorded in the OTP at three different temperatures. These values can be read from TI OTP starting at location F008 0310h as shown in Figure 7-7 and described in Table 7-8. The values recorded were measured with ADREFHI equal to 3.3V.

Figure 7-7. TI OTP Bank 0 Temperature Sensor 1 Calibration Information (F008 0310h-F008 031Fh)

0x00	0x02	0x04	0x06	0x08	0x0A	0x0C	0x0E
S1TEMP1VAL	S1TEMP1	S1TEMP2VAL	S1TEMP2	S1TEMP3VAL	S1TEMP3	0xFFFF	0xFFFF
R	R	R	R	R	R	R	R

LEGEND: R = Read only

Figure 7-8. TI OTP Bank 0 Temperature Sensor 2 Calibration Information (F008 0320h-F008 032Fh)

0x00	0x02	0x04	0x06	0x08	0x0A	0x0C	0x0E
S2TEMP1VAL	S2TEMP1	S2TEMP2VAL	S2TEMP2	S2TEMP3VAL	S2TEMP3	0xFFFF	0xFFFF
R	R	R	R	R	R	R	R

LEGEND: R = Read only

Figure 7-9. TI OTP Bank 0 Temperature Sensor 3 Calibration Information (F008 0330h-F008 033Fh)

0x00	0x02	0x04	0x06	0x08	0x0A	0x0C	0x0E
S3TEMP1VAL	S3TEMP1	S3TEMP2VAL	S3TEMP2	S3TEMP3VAL	S3TEMP3	0xFFFF	0xFFFF
R	R	R	R	R	R	R	R

LEGEND: R = Read only

Table 7-8. TI OTP Bank 0 Temperature Sensor Calibration Information Field Descriptions

Address	Width	Field	Description
F008 03x0h	16 bits	SxTEMP1VAL	The value read from the ADC for this sensor at the first calibration temperature.
F008 03x2h	16 bits	SxTEMP1	The temperature in degrees Kelvin.
F008 03x4h	16 bits	SxTEMP2VAL	The value read from the ADC for this sensor at the second calibration temperature.
F008 03x6h	16 bits	SxTEMP2	The temperature in degrees Kelvin.
F008 03x8h	16 bits	SxTEMP3VAL	The value read from the ADC for this sensor at the third calibration temperature.
F008 03xAh	16 bits	SxTEMP3	The temperature in degrees Kelvin.
F008 03xCh	16 bits	0xFFFF	Reserved
F008 03xEh	16 bits	0xFFFF	Reserved

7.5.2.6 Deliberate ECC Errors for FMC ECC Checking

Deliberate single-bit and double-bit errors have been placed in the OTP for checking the L2FMC ECC functionality. Any portion of the 64 bits in TI OTP bank 0 location F008 03F0h through F008 03F7h as shown in Figure 7-10 will generate a single-bit error. Any portion of the 64 bits in TI OTP bank 0 location F008 03F8h through F008 03FFh as shown in Figure 7-10 will generate a double-bit error.

Figure 7-10. TI OTP Bank 0 Deliberate ECC Error Information

0x00	0x04	0x08	0x0C
0x12345678	0x9ABCDEF1	0x12345678	0x9ABCDEF3
R	R	R	R

LEGEND: R = Read only, * ECC is calculated for the value 0x123456789ABCDEF0

7.6 Power On, Power Off Considerations

7.6.1 Error Checking at Power On

As the device is coming out of the device reset sequence, the Flash wrapper reads a configuration word from the TI OTP section of bank 0. These are known as Implicit Reads. This is also readable from a bus master at address F008 0140h. During these reads ECC is enabled. Single-bit errors are corrected and uncorrectable errors will generate an error event. Accordingly, the IMPLICIT_COR_ERR or the IMPLICIT_UNC_ERR bits in the FEDAC_GBLSTATUS register (offset = 1Ch) will get set. Refer to the data manual to find the ESM group and channel number on which it is triggered.

7.6.2 Flash Integrity at Power Off

If power is lost during a programming or erase operation, a power-on reset must be asserted before the core supply voltage drops below specification. The $\overline{\text{PORRST}}$ pin has a glitch filter that means that the $\overline{\text{PORRST}}$ pin must be asserted low $t_{f(nPORRST)}$ (2 µs) before the core supply drops below Vcc_{MIN} (1.14V). If this requirement is met, then the bits being programmed when $\overline{\text{PORRST}}$ goes low are indeterminate; however, the other bits in the Flash are not disturbed. Likewise, if this requirement is met, and $\overline{\text{PORRST}}$ is asserted while erasing, the sector or sectors being erased will have indeterminate bits; however, the other sectors in the same bank and the other banks will not be disturbed.

7.7 Emulation and SIL3 Diagnostic Modes

7.7.1 System Emulation

During emulation when the SUSPEND signal is high, address tag and command parity error events are not generated.

7.7.2 Diagnostic Mode

The Flash wrapper can be put in diagnostic mode to verify various logic. There are multiple diagnostic modes supported by the wrapper. A specific diagnostic mode is selected via the DIAGMODE control bits in the diagnostic control register (FDIAGCTRL), as listed in Table 7-9.

The diagnostic mode is only enabled by a 4-bit key stored in the DIAG_EN_KEY bits in FDIAGCTRL register. Only DIAG_EN_KEY = 5h enables any diagnostic mode and all diagnostic modes use the DIAG_TRIG bit in FDIAGCTRL register to initiate the action.

For all modes it is best to follow this sequence:

1. Write 5h to the DIAG_EN_KEY bits and set the desired DIAGMODE control bits.
2. Set any data registers needed for this mode.
3. Write a 1 to the DIAG_TRIG bit to initiate the action and allow an error to happen.
4. Write a Ah to the DIAG_EN_KEY bits to disable the diagnostic modes.

Table 7-9. DIAGMODE Encoding

Mode	DIAGMODE Bits			Description
0	0	0	0	Diagnostic mode is disabled. Same as DIAG_EN_KEY not equal to 5h.
5	1	0	1	Address Tag Register test mode
7	1	1	1	ECC Data Correction Diagnostic test mode
Others	Other Combinations			Reserved

7.7.2.1 Address Tag Register Test Mode: DIAGMODE = 5

There are six sets of address tag registers, two for Port A and four for Port B. Each set consists of a primary and a duplicate address tag registers. Normally, these registers store the recently issued CPU addresses during prefetch mode. To detect errors in these registers, the primary and duplicate address tag registers are continuously compared to each other if the buffer is valid. If they are different, then an address tag register error event is generated.

These registers are memory-mapped. All primary address tag registers are memory-mapped to one address and, likewise, all duplicate tag registers are mapped to another single address. During diagnostic mode, each individual set can be selected by the DIAG_BUF_SEL (Diagnostic Buffer Select) bit in the FDIAGCTRL register. User-supplied values can be written into the selected set during a diagnostic mode. This diagnostic mode uses the FRAW_ADDR register to supply the alternate address. When the DIAG_TRIG bit is set, the FRAW_ADDR register value is compared with the primary and the duplicate address tag registers. If the results of the comparison are different, then the ADD_TAG_ERR (Address Tag Error) flag in the FEDAC_PxSTATUS register will be set. Also, refer to the device data manual for the specific error channel that will be asserted in this situation.

The sequence to do this test would be:

1. Branch to a non-Flash region for executing this sequence. Ensure no requests from any bus master are arriving at the port (A or B) that is being diagnosed.
2. Set DIAGMODE to 5h and DIAG_EN_KEY to 5h in the FDIAGCTRL register.
3. Select the appropriate buffer to be diagnosed using the DIAG_BUF_SEL bits in the FDIAGCTRL register using the table in Section 7.10.23.
4. Set the FRAW_ADDR register to a certain arbitrary value 'A'. The lowest 5 bits should be cleared to 0.
5. Set the FPRIM_ADD_TAG register and the FDUP_ADD_TAG register in such a way that one of them equals 'A' and the other one does not. The lowest 5 bits in both these writes should be cleared to 0.
6. Set the DIAG_TRIG bit in the FDIAGCTRL register.
7. Now check the appropriate ADD_TAG_ERR flag in the FEDAC_PxSTATUS register based on the port being diagnosed. Ensure that it is 1, implying successful operation of the compare logic.
8. Write 1 to the ADD_TAG_ERR bit to clear it.
9. Repeat for the different buffers.
10. At the end of the test, clear DIAGMODE bit to 0 and set DIAG_EN_KEY bits to Ah in the FDIAGCTRL register to completely disable the test.

All address tags and buffer valid bits will be cleared to 0 when leaving diag_mode 5.

NOTE: You should pre-load the registers with the test values with DIAG_TRIG = 0. After all test values are written, the DIAG_TRIG should then be set high to validate the diagnostic result.

7.7.2.2 ECC Data Correction Diagnostic Test Mode: DIAGMODE = 7

Testing the error correction and ECC logic in the CPU involves corrupting the ECC value returned to the CPU. By inverting one or more bits of the ECC, the CPU will detect errors in a selected data or ECC bit, or in any possible value returned by the ECC.

To set an error for a particular bit use the syndrome, see Section 7.4.3. For example, if you want to corrupt data bit 62 then put the value 70h into the test register.

The method uses the FEMU_ECC and the FEMU_DxSW registers to alter the ECC and data, respectively, for one Flash access port read. The values in the FEMU_ECC and FEMU_DxSW registers will be XORed with the current ECC and data, respectively, to give a bad ECC or data value back to the bus master. This will only occur for one read when the DIAGMODE is 7h, the DIAG_EN_KEY is 5h, and the DIAG_TRIG is written with value of 1 in the FDIAGCTRL register.

The sequence to do this test is:

1. Branch to a non-Flash region to execute this sequence.
2. Set DIAGMODE to 7h and DIAG_EN_KEY to 5h in the FDIAGCTRL register.
3. Set desired values to XOR in the FEMU_ECC and FEMU_DxSW registers.
4. Set DIAG_TRIG to 1 in the FDIAGCTRL register.
5. Select the appropriate port in which the flip is desired using the DIAG_BUF_SEL bits in the FDIAGCTRL register. Only legal values are 0 for port A and 4h for port B.
6. Do a port A or B read to the desired address. The L2FMC will XOR the data and ECC with FEMU_DxSW and FEMU_ECC, respectively, for this read before delivering it to the CPU. No further reads are affected by this diagnostic.
7. The error routine of the bus master (for example, CPU) shall cause the address and erroneous bit to be known. This should match with the bit flipped in step 3.
8. Repeat as necessary to test out various bits of data and ECC.
9. Clear DIAGMODE to 0 and set DIAG_EN_KEY to Ah in the FDIAGCTRL register to completely disable this test.

NOTE: Make sure the address to be used for diagnostic is not already cached; otherwise, the read will read from the cache memory instead of the Flash.

7.7.3 Diagnostic Mode Summary

Table 7-10 gives a summary of the input registers needed for each mode, the possible registers that can change, and the possible error bits in FEDACSTATUS that may set.

Table 7-10. Diagnostic Mode Summary

DIAG MODE	Name	Inputs	Possible Outputs	Possible Error Bits Set	Notes
5	Address Tag Register test mode	FPRIM_ADD_TAG FDUP_ADD_TAG FRAW_ADDR		ADD_TAG_ERR in FEDAC_PxSTATUS register	This will cause ESM error. Please refer to the data manual to find group and channel number.
7	ECC Data Correction Diagnostic test mode	FEMU_ECC FEMU_DxSW	Bus master will indicate data ECC single-bit or multi-bit error.	None	This will cause ESM error. Please refer to the data manual to find group and channel number.

7.7.4 SECDED Software Diagnostic

The SECDED block is used to perform error detection and correction on the implicit reads made after reset by L2FMC. To simplify the diagnostic for this logic, a software mechanism is used. To check that the SECDED module correctly performed its operation, the following steps must be used:

1. CPU reads the 64-bit memory location of the implicit read. For example, implicit read location is at 0xF008_0140.
2. Next, CPU reads the memory mapped registers RCR_VALUEx registers accessible at address offsets D0h and D4h.
3. The two 64-bit values read in steps 1 and 2 are compared for being equal.
   a. If the two values are equal, then the location in memory after correction by the CPU SECDED is the same value as location in memory after correction by L2FMC SECDED. Assuming the CPU SECDED can be independently verified, the L2FMC SECDED must be functioning correctly.
   b. If the two values are not equal, then L2FMC SECDED is not functioning correctly.

7.7.5 Read Margin

When the bits are programmed or erased, they are checked against a program_verify or erase_verify reference level that is far away from the normal read reference point. Over time, bit levels may drift toward the normal read point and if it is too much then a bit will read the wrong value. To counteract this, the bits can be read using different read_margin reference points to give an early detection of the problem. The bits can then be either re-programmed (most common) or the sector can be erased and reprogrammed.

7.8 Parameter Overlay Module (POM)

In many applications it is important to be able to change certain parameters in the program without having to re-flash the device and immediately test these changes either in a hardware-in-the-loop simulation or in a real environment. The Parameter Overlay Module (POM) helps to achieve this goal. The POM provides a mechanism to redirect accesses to non-volatile memory into a volatile memory that can be internal to the device or external. The data requested by the master will be fetched from the overlay memory instead of the main non-volatile memory. The overlay memory can be accessed by other masters in the system to provide an easy update path of the stored data. Other masters can be, for example, the main CPU, DMA, DMM, or DAP AHB-AP.

7.8.1 Example Procedure to Configure the POM

Suppose the intent is to remap 128KB of Flash at address 10_0000h to 8000_0000h. Note that both program region and overlay region have to be aligned to the size of the region. Sequence to perform this configuration would be as follows:

1. Ensure that there are no active accesses to this space while the following configuration is ongoing.
2. Write to the POMGLBCTRL.OTADDR (offset 0h) a value of 200h. These are the upper 10bits of the overlay region base address.
3. Write to the POMPROGSTART0.STARTADDRESS (offset 200h) a value of 0x10_0000h.
4. Write to the POMOVLSTART0.STARTADDRESS (offset 204h) a value of 00_0000h. These are the bits 21-17 of the overlay region address. Since the region size is 128KB the lower bits do not matter.
5. Write to the POMREGSIZE0.SIZE (offset 208h) a value of Ch.
6. Finally write to the POMGLBCTRL.ON_OFF to Ah.
7. End of sequence.

7.9 Summary of L2FMC Errors

Table 7-11. Errors in L2FMC

Scenario	Does this error cause a Bus Error?	Does this error go to ESM?	Is a flag in L2FMC set?	Name of Flag
Access parity error/Internal parity errors	Yes	Yes	Yes	FEDAC_PxSTATUS. ADD_PAR_ERR
Port A/B Idle State parity error	No	Yes	Yes	FEDAC_PxSTATUS. MCMD_PAR_ERR
Address tag error	Yes	Yes	Yes	FEDAC_PxSTATUS. ADD_TAG_ERR
Access to Flash space beyond available size	Yes	No	No	-
Access to Flash while pump/bank are not active	Yes	No	No	-
Flash Access time-out	Yes	Yes	Yes	FEDAC_PxSTATUS. ACC_TOUT
Invalid access to L2FMC (for example, writes)	Yes	No	No	-
Single-bit Error during Implicit Reads	No	Yes	Yes	FEDAC_GBLSTATUS. IMPLICIT_COR_ERR
Uncorrectable Error during Implicit Reads	No	Yes	Yes	FEDAC_GBLSTATUS. IMPLICIT_UNC_ERR
Access to bank while program/erase operations are ongoing on the same bank	Yes	No	No	-
Access to register address offsets between 2C8h and 3FFh or 4B8h and 7FFh	Yes	No	No	-
Redirected access to POM received a bus error	Yes	No	No	-
Response of redirected access to POM has access parity error	Yes	No	Yes	POMFLG.PERR_Px
POM Idle State parity error	No	Yes	Yes	FEDAC_PxSTATUS. MCMD_PAR_ERR
Soft Errors in high integrity bits carrying Implicit read data	No	Yes	Yes	FEDAC_GBLSTATUS. RCR_ERR

7.10 Flash Control Registers

This section details the Flash module registers, summarized in Table 7-12. The Flash module control registers can only be read and/or written by the CPU while in privileged mode. Each register begins on a word boundary. All registers are 32-bit, 16-bit and 8-bit accessible. The start address of the Flash module is FFF8 7000h.

Table 7-12. Flash Control Registers

Offset	Acronym	Register Description	Section
00h	FRDCNTL	Flash Read Control Register	Section 7.10.1
04h	FSPRD	Read Margin Control Register	Section 7.10.2
08h	EE_FEDACCTRL1	EEPROM Error Correction Control Register 1	Section 7.10.3
14h	FEDAC_PASTATUS	Flash Port A Error and Status Register	Section 7.10.4
18h	FEDAC_PBSTATUS	Flash Port B Error and Status Register	Section 7.10.5
1Ch	FEDAC_GBLSTATUS	Flash Global Error and Status Register	Section 7.10.6
24h	FEDACSDIS	Flash Error Detection and Correction Sector Disable Register	Section 7.10.7
28h	FPRIM_ADD_TAG	Flash Primary Address Tag Register	Section 7.10.8
2Ch	FDUP_ADD_TAG	Flash Duplicate Address Tag Register	Section 7.10.9
30h	FBPROT	Flash Bank Protection Register	Section 7.10.10
34h	FBSE	Flash Bank Sector Enable Register	Section 7.10.11
38h	FBBUSY	Flash Bank Busy Register	Section 7.10.12
3Ch	FBAC	Flash Bank Access Control Register	Section 7.10.13
40h	FBPWRMODE	Flash Bank Power Mode Register	Section 7.10.14
44h	FBPRDY	Bank/Pump Ready Register	Section 7.10.15
48h	FPAC1	Flash Pump Access Control Register 1	Section 7.10.16
50h	FMAC	Flash Module Access Control Register	Section 7.10.17
54h	FMSTAT	Flash Module Status Register	Section 7.10.18
58h	FEMU_DMSW	EEPROM Emulation Data MSW Register	Section 7.10.19
5Ch	FEMU_DLSW	EEPROM Emulation Data LSW Register	Section 7.10.20
60h	FEMU_ECC	EEPROM Emulation Address Register	Section 7.10.21
64h	FLOCK	Flash Lock Register	Section 7.10.22
6Ch	FDIAGCTRL	Diagnostic Control Register	Section 7.10.23
74h	FRAW_ADDR	Raw Address	Section 7.10.24
7Ch	FPAR_OVR	Parity Override Register	Section 7.10.25
B4h	RCR_VALID	Reset Configuration Valid Register	Section 7.10.26
B8h	ACC_THRESHOLD	Crossbar Access Time Threshold Register	Section 7.10.27
C0h	FEDACSDIS2	Flash Error Detection and Correction Sector Disable Register 2	Section 7.10.28
D0h	RCR_VALUE0	Lower Word of the Reset Configuration Read Register	Section 7.10.29
D4h	RCR_VALUE1	Upper Word of the Reset Configuration Read Register	Section 7.10.30
288h	FSM_WR_ENA	FSM Register Write Enable Register	Section 7.10.31
2B8h	EEPROM_CONFIG	EEPROM Emulation Configuration Register	Section 7.10.32
2C0h	FSM_SECTOR1	FSM Sector Register 1	Section 7.10.33
2C4h	FSM_SECTOR2	FSM Sector Register 2	Section 7.10.34
400h	FCFG_BANK	Flash Bank Configuration Register	Section 7.10.35

7.10.1 Flash Read Control Register (FRDCNTL)

FRDCNTL supports prefetch mode. This register controls Flash timings for the main Flash banks. For the equivalent register that controls Flash timings for the EEPROM Emulation Flash bank (bank 7), see Section 7.10.32.

Figure 7-11. Flash Read Control Register (FRDCNTL) (offset = 00h)

31 16
Reserved
R-0

15 12	11 8	7 2	1	0
Reserved	RWAIT	Reserved	PFUENB	PFUENA
R-0	R/WP-1	R-0	R/WP-1	R/WP-1

LEGEND: R/W = Read/Write; R = Read only; WP = Write in Privilege Mode; -*n* = value after reset

Table 7-13. Flash Read Control Register (FRDCNTL) Field Descriptions

Bit	Field	Value	Description
31-12	Reserved	0	Reads return 0. Writes have no effect.
11-8	RWAIT	0-Fh	Random/data Read Wait State The random read wait state bits indicate how many wait states are added to a Flash read access. Address wait state is fixed to 1 HCLK cycle. **Note:** The required wait states for each HCLK frequency can be found in the device-specific data sheet.
7-2	Reserved	0	Reads return 0. Writes have no effect.
1	PFUENB		Prefetch Enable for Port B
		0	Prefetch Mode is disabled.
		1	Prefetch Mode is enabled. (Recommended)
0	PFUENA		Prefetch Enable for Port A
		0	Prefetch Mode is disabled.
		1	Prefetch Mode is enabled. (Recommended)

7.10.2 Read Margin Control Register (FSPRD)

This register controls the read margin mode.

NOTE: If both RM0 and RM1 bits are set then Read Margin 0 is enabled.

Figure 7-12. Read Margin Control Register (FSPRD) (offset = 04h)

31–16
Reserved
R-0

15–8	7–2	1	0
RMBSEL[7:0]	Reserved	RM1	RM0
R/WP-0	R-0	R/WP-0	R/WP-0

LEGEND: R/W = Read/Write; R = Read only; WP = Write in Privilege Mode; -*n* = value after reset

Table 7-14. Read Margin Control Register (FSPRD) Field Descriptions

Bit	Field	Value	Description
31-16	Reserved	0	Reads return 0. Writes have no effect.
15-8	RMBSEL[*n*]		Read Margin Bank Select. Each bit corresponds to a Flash bank. RMBSEL is only decoded if either the RM1 or RM0 bit is set. When either RM1 or RM0 is set, the RMBSEL bit corresponding to a bank forces the selected bank(s) to be read in the selected margin mode. The unselected bank(s) are still read in normal mode. There must be 2 accesses to the bank before the read margin takes effect.
7-2	Reserved	0	Reads return 0. Writes have no effect.
1	RM1		Read Margin 1
		0	Read Margin 1 Mode is disabled.
		1	Read Margin 1 Mode is enabled.
0	RM0		Read Margin 0
		0	Read Margin 0 Mode is disabled.
		1	Read Margin 0 Mode is enabled.

7.10.3 EEPROM Error Correction Control Register (EE_FEDACCTRL1)

When a EEPROM bank is erased or zeroed out, the contents will be all 1's or all 0's, respectively. In such a case, the ECC will be incorrect. EE_FEDACCTRL1 lets the L2FMC ignore an all 1's and all 0's condition, on reads from the EEPROM bank.

Figure 7-13. EEPROM Error Correction Control Register (EE_FEDACCTRL1) (offset = 08h)

31 16
Reserved
R-0

15 6	5	4	3 0
Reserved	EOCV	EZCV	Reserved
R-0	R/WP-0	R/WP-0	R-0

LEGEND: R/W = Read/Write; R = Read only; WP = Write in Privilege Mode; -*n* = value after reset

Table 7-15. EEPROM Error Correction Control Register (EE_FEDACCTRL1) Field Descriptions

Bit	Field	Value	Description
31-6	Reserved	0	Reads return 0. Writes have no effect.
5	EOCV		One condition valid
		0	DO NOT allow the condition of all data bits and ECC bits to be 1.
		1	Allow the condition of all data bits and ECC bits to be 1.
4	EZCV		Zero condition valid
		0	DO NOT allow the condition of all data bits and ECC bits to be 0.
		1	Allow the condition of all data bits and ECC bits to be 0.
3-0	Reserved	0	Reads return 0. Writes have no effect.

7.10.4 Flash Port A Error and Status Register (FEDAC_PASTATUS)

This register applies to accesses made to the main or EEPROM Flash banks through Port A.

All these error status bits can be cleared by writing a 1 to the bit; writing a 0 has no effect.

Figure 7-14. Flash Port A Error and Status Register (FEDAC_PASTATUS) (offset = 14h)

31	24
Reserved	
R-0	

23	16
Reserved	
R-0	

15	14	13	12	11	10	9	8
ACCTOUT	MCMD_PAR_ ERR	Reserved		ADD_TAG_ ERR	ADD_PAR_ ERR	Reserved	
RCP-0	RCP-u	R-0		RCP-u	RCP-u	R-0	

7	0
Reserved	
R-0	

LEGEND: R = Read only; RCP = Read and Clear in Privilege Mode; -u = unchanged value on internal reset, cleared on power up; -*n* = value after reset

Table 7-16. Flash Port A Error and Status Register (FEDAC_PASTATUS) Field Descriptions

Bit	Field	Value	Description
31-16	Reserved	0	Reads return 0. Writes have no effect.
15	ACCTOUT		Severe internal switch timeout/parity error on Port A access.
		0	L2FMC internal switch has NOT encountered a severe error (access timeout or parity).
		1	L2FMC internal switch has encountered a severe error (access timeout or parity).
			This error is routed to the ESM. Refer to the device data manual to find the group and channel on which it is routed.
14	MCMD_PAR_ERR		Parity Error in idle state. This bit is set when a parity error occurs during idle state of Port A.
		0	No idle state parity error is detected.
		1	Parity error is detected in idle state.
			This error is routed to the ESM. Refer to the device data manual to find the group and channel on which it is routed.
13-12	Reserved	0	Reads return 0. Writes have no effect.
11	ADD_TAG_ERR		Port A Address Tag Register Error Flag. This bit is set if the primary address tag has a hit but the duplicate address tag does not match the primary address tag. This bit is functional only when Port A prefetch mode is enabled (PFUENA = 1).
		0	Address Tag Register Error not detected on Port A.
		1	Address Tag Register Error detected on Port A.
			This error is routed to the ESM. Refer to the device data manual to find the group and channel on which it is routed.
10	ADD_PAR_ERR		Address Parity Error Flag.
		0	No parity error was detected on the incoming access to the L2FMC Port A.
		1	A parity error was detected on the incoming access to the L2FMC Port A. The address of the erroneous access is not stored in L2FMC.
			This error is routed to the ESM. Refer to the device data manual to find the specific group and channel on which it is routed.
9-0	Reserved	0	Reads return 0. Writes have no effect.

7.10.5 Flash Port B Error and Status Register (FEDAC_PBSTATUS)

This register applies to accesses made to the main or EEPROM Flash banks through Port B.

All these error status bits can be cleared by writing a 1 to the bit; writing a 0 has no effect.

Figure 7-15. Flash Port B Error and Status Register (FEDAC_PBSTATUS) (offset = 18h)

31	24
Reserved	
R-0	

23	16
Reserved	
R-0	

15	14	13	12	11	10	9	8
ACCTOUT	MCMD_PAR_ERR	Reserved		ADD_TAG_ERR	ADD_PAR_ERR	Reserved	
RCP-0	RCP-u	R-0		RCP-u	RCP-u	R-0	

7	0
Reserved	
R-0	

LEGEND: R = Read only; RCP = Read and Clear in Privilege Mode; -u = unchanged value on internal reset, cleared on power up; -*n* = value after reset

Table 7-17. Flash Port B Error and Status Register (FEDAC_PBSTATUS) Field Descriptions

Bit	Field	Value	Description
31-16	Reserved	0	Reads return 0. Writes have no effect.
15	ACCTOUT		Severe error - internal switch timeout.
		0	L2FMC internal switch has NOT encountered a severe error (access timeout).
		1	L2FMC internal switch has encountered a severe error (access timeout).
			This error is routed to the ESM. Refer to the device data manual to find the group and channel on which it is routed.
14	MCMD_PAR_ERR		Parity Error in idle state. This bit is set when a parity error occurs during idle state of Port B.
		0	No idle state parity error is detected.
		1	Parity error is detected in idle state.
			This error is routed to the ESM. Refer to the device data manual to find the group and channel on which it is routed.
13-12	Reserved	0	Reads return 0. Writes have no effect.
11	ADD_TAG_ERR		Port B Address Tag Register Error Flag. This bit is set if the primary address tag has a hit but the duplicate address tag does not match the primary address tag. This bit is functional only when Port B prefetch mode is enabled (PFUENB = 1).
		0	Address Tag Register Error not detected on Port B.
		1	Address Tag Register Error detected on Port B.
			This error is routed to the ESM. Refer to the device data manual to find the group and channel on which it is routed.
10	ADD_PAR_ERR		Address Parity Error Flag.
		0	No parity error was detected on the incoming access to the L2FMC Port B.
		1	A parity error was detected on the incoming access to the L2FMC Port B. The address of the erroneous access is not stored in L2FMC.
			This error is routed to the ESM. Refer to the device data manual to find the specific group and channel on which it is routed.
9-0	Reserved	0	Reads return 0. Writes have no effect.

7.10.6 Flash Global Error and Status Register (FEDAC_GBLSTATUS)

This register applies to global error and status flags in L2FMC.

All these status bits can be cleared by writing a 1 to the bit; writing a 0 has no effect.

Figure 7-16. Flash Global Error and Status Register (FEDAC_GBLSTATUS) (offset = 1Ch)

31	24
Reserved	FSM_DONE
R-0	RCP-0

23 16
Reserved
R-0

15	14	13	12 8
RCR_ERR	IMPLICIT_COR_ERR	IMPLICIT_UNC_ERR	Reserved
RCP-0	RCP-0	RCP-0	R-0

7 0
Reserved
R-0

LEGEND: R = Read only; RCP = Read and Clear in Privilege Mode; -*n* = value after reset

Table 7-18. Flash Global Error and Status Register (FEDAC_GBLSTATUS) Field Descriptions

Bit	Field	Value	Description
31-25	Reserved	0	Reads return 0. Writes have no effect.
24	FSM_DONE		Flash State Machine Done This bit is set to 1 when the Flash state machine completes a program or erase operation. This bit will generate an interrupt on VIM channel 61 if the FSM_EVT_EN bit of the FSM_ST_MACHINE register is set. This bit must be cleared by writing a 1 to it in the interrupt routine to clear the interrupt request.
23-16	Reserved	0	Reads return 0. Writes have no effect.
15	RCR_ERR		Soft error in high integrity bits carrying implicit read data.
		0	No error detected in high-integrity bits.
		1	Error detected in high-integrity bits.
			This error is routed to the ESM. Refer to the device data manual to find the group and channel on which it is routed.
14	IMPLICIT_COR_ERR		Correctable error occurred during implicit reads.
		0	No single-bit error is detected during implicit read.
		1	Single-bit error is detected during implicit read.
			This error is routed to the ESM. Refer to the device data manual to find the group and channel on which it is routed.
13	IMPLICIT_UNC_ERR		Uncorrectable error occurred during implicit reads.
		0	No double-bit error is detected during implicit read.
		1	Double-bit error is detected during implicit read.
			This error is routed to the ESM. Refer to the device data manual to find the group and channel on which it is routed.
12-0	Reserved	0	Reads return 0. Writes have no effect.

7.10.7 Flash Error Detection and Correction Sector Disable Register (FEDACSDIS)

This register is used to disable the SECDED function for one or two sectors from the EEPROM Emulation Flash (bank 7). An additional two sectors can have SECDED disabled by the use of the FEDACSDIS2 register (see Section 7.10.28).

Figure 7-17. Flash Error Detection and Correction Sector Disable Register (FEDACSDIS) (offset = 24h)

31	29 24	23 22	21 16
Rsvd	SectorID1_inverse	Rsvd	SectorID1
R-0	R/WP-0	R-0	R/WP-0

15 14	13 8	7 6	5 0
Rsvd	SectorID0_inverse	Rsvd	SectorID0
R-0	R/WP-0	R-0	R/WP-0

LEGEND: R/W = Read/Write; R = Read only; WP = Write in Privilege Mode; -*n* = value after reset

Table 7-19. Flash Error Detection and Correction Sector Disable Register (FEDACSDIS) Field Descriptions

Bit	Field	Value	Description
31-30	Reserved	0	Reads return 0. Writes have no effect.
29-24	SectorID1_inverse	0-3Fh	The sector ID inverse bits are used with the sector ID bits to determine which sector is disabled. If the sector ID bits are not pointing to a valid sector (0-3) or the sector ID inverse bits are not an inverse of the sector ID bits, then no sector is disabled by disable ID 1.
23-22	Reserved	0	Reads return 0. Writes have no effect.
21-16	SectorID1	0-3Fh	The sector ID bits are used with the sector ID inverse bits to determine which sector is disabled. If the sector ID bits are not pointing to a valid sector (0-3) or the sector ID inverse bits are not an inverse of the sector ID bits, then no sector is disabled by disable ID 1.
15-14	Reserved	0	Reads return 0. Writes have no effect.
13-8	SectorID0_inverse	0-Fh	The sector ID inverse bits are used with the sector ID bits to determine which sector is disabled. If the sector ID bits are not pointing to a valid sector (0-3) or the sector ID inverse bits are not an inverse of the sector ID bits, then no sector is disabled by disable ID 0.
7-6	Reserved	0	Reads return 0. Writes have no effect.
5-0	SectorID0	0-3Fh	The sector ID bits are used with the sector ID inverse bits to determine which sector is disabled. If the sector ID bits are not pointing to a valid sector (0-3) or the sector ID inverse bits are not an inverse of the sector ID bits, then no sector is disabled by disable ID 0.

7.10.8 Primary Address Tag Register (FPRIM_ADD_TAG)

This register is used to test the prefetch address tag registers. (see Section 7.7.2.1)

Figure 7-18. Primary Address Tag Register (FPRIM_ADD_TAG) (offset = 28h)

31	16
PRIM_ADD_TAG	
R/WP-0	

15	5	4	0
PRIM_ADD_TAG		Reserved	
R/WP-0		R-0	

LEGEND: R/W = Read/Write; R = Read only; WP = Write in Privilege Mode; -*n* = value after reset

Table 7-20. Primary Address Tag Register (FPRIM_ADD)_TAG Field Descriptions

Bit	Field	Value	Description
31-5	PRIM_ADD_TAG	0-7FF FFFFh	Primary Address Tag Register The primary address tag register selected by the DIAG_BUF_SEL bits in the FDIAGCTRL register is memory-mapped here. This register can only be written in privileged mode when diagnostic mode is enabled with DIAG_EN_KEY = 5h and DIAGMODE = 5h in the FDIAGCTRL register. This register is not updated with new Flash data if DIAG_EN_KEY is not equal to 5h or DIAGMODE is 0 or 7h. Valid reads can occur in any mode. The register clears when an address tag error is found and when leaving DIAG_MODE 5.
4-0	Reserved	0	Reads return 0. Writes have no effect.

7.10.9 Duplicate Address Tag Register (FDUP_ADD_TAG)

This register is used to test the prefetch address tag registers. (see Section 7.7.2.1)

Figure 7-19. Duplicate Address Tag Register (FDUP_ADD_TAG) (offset = 2Ch)

31	16
DUP_ADD_TAG	
R/WP-0	

15	5	4	0
DUP_ADD_TAG		Reserved	
R/WP-0		R-0	

LEGEND: R/W = Read/Write; R = Read only; WP = Write in Privilege Mode; -*n* = value after reset

Table 7-21. Duplicate Address Tag Register (FDUP_ADD)_TAG Field Descriptions

Bit	Field	Value	Description
31-5	DUP_ADD_TAG	0-7FF FFFFh	Duplicate Address Tag Register The duplicate address tag register selected by the DIAG_BUF_SEL bits in the FDIAGCTRL register is memory-mapped here. This register can only be written in privileged mode when diagnostic mode is enabled with DIAG_EN_KEY = 5h and DIAGMODE = 5h in the FDIAGCTRL register. This register is not updated with new Flash data if DIAG_EN_KEY is not equal to 5h or DIAGMODE is 0 or 7h. Valid reads can occur in any mode. The register clears when an address tag error is found and when leaving DIAG_MODE 5.
3-0	Reserved	0	Reads return 0. Writes have no effect.

7.10.10 Flash Bank Protection Register (FBPROT)

Figure 7-20. Flash Bank Protection Register (FBPROT) (offset = 30h)

31	16
Reserved	
R-0	

15	1	0
Reserved		PROTL1DIS
R-0		R/WP-0

LEGEND: R/W = Read/Write; R = Read only; WP = Write in Privilege Mode; -*n* = value after reset

Table 7-22. Flash Bank Protection Register (FBPROT) Field Descriptions

Bit	Field	Value	Description
31-1	Reserved	0	Reads return 0. Writes have no effect.
0	PROTL1DIS		Level 1 Protection Disable bit Setting this bit disables protection from writing to the OTPPROTDIS bits in the FBAC register as well as the BSE bits for all banks in the FBSE register. Clearing this bit enables protection and disables write access to the OTPPROTDIS bits and FBSE register.
		0	Level 1 protection is enabled.
		1	Level 1 protection is disabled.

7.10.11 Flash Bank Sector Enable Register (FBSE)

FBSE provides one enable bit per sector for up to 16 sectors per bank. Each bank in the Flash module has one FBSE register. The bank is selected via the BANK bits in the FMAC register. As only one bank at a time can be selected by FMAC, only the register for the bank selected appears at this address.

Figure 7-21. Flash Bank Sector Enable Register (FBSE) (offset = 34h)

31	16
Reserved	
R-0	

15	0
BSE[15:0]	
R/WP-0	

LEGEND: R/W = Read/Write; R = Read only; WP=Write in Privilege Mode; -*n* = value after reset

Table 7-23. Flash Bank Sector Enable Register (FBSE) Field Descriptions

Bit	Field	Value	Description
31-16	Reserved	0	Reads return 0. Writes have no effect.
15-0	BSE[*n*]		Bank Sector Enable. Each bit corresponds to a Flash sector in the bank specified by the FMAC register. Bit 0 corresponds to sector 0, bit 1 corresponds to sector 1, and so on. These bits can be set only when PROTL1DIS = 1 in the FBPROT register and in privilege mode.
		0	The corresponding numbered sector is disabled for program or erase access.
		1	The corresponding numbered sector is enabled for program or erase access.

7.10.12 Flash Bank Busy Register (FBBUSY)

Figure 7-22. Flash Bank Busy Register (FBBUSY) (offset = 38h)

31	16
Reserved	
R-0	

15	8	7	0
Reserved		BUSY[7:0]	
R-0		R-0	

LEGEND: R = Read only; -*n* = value after reset

Table 7-24. Flash Bank Busy Register (FBBUSY) Field Descriptions

Bit	Field	Value	Description
31-8	Reserved	0	Reads return 0. Writes have no effect.
7-0	BUSY[*n*]		Bank Busy. Each bit corresponds to a Flash bank.
		0	The corresponding bank is not busy.
		1	The corresponding bank is busy with a state machine operation, or the bank is not implemented.

7.10.13 Flash Bank Access Control Register (FBAC)

Figure 7-23. Flash Bank Access Control Register (FBAC) (offset = 3Ch)

31	24	23	16
Reserved		OTPPROTDIS[7:0]	
R-0		R/WP-0	

15	8	7	0
BAGP		VREADST	
R/WP-0		R/WP-Fh	

LEGEND: R/W = Read/Write; R = Read only; WP = Write in Privilege Mode; -*n* = value after reset

Table 7-25. Flash Bank Access Control Register (FBAC) Field Descriptions

Bit	Field	Value	Description
31-24	Reserved	0	Reads return 0. Writes have no effect.
23-16	OTPPROTDIS[*n*]		OTP Sector Protection Disable. Each bit corresponds to a Flash bank. This bit can be set only when PROTL1DIS = 1 in the FBPROT register and in privilege mode.
		0	Programming of the OTP sector is disabled.
		1	Programming of the OTP sector is enabled.
15-8	Reserved	0	Reads return 0. Writes have no effect.
7-0	VREADST	0-FFh	VREAD Setup. VREAD is generated by the Flash pump and used for Flash read operation. The bank power up sequencing starts VREADST HCLK cycles after VREAD power supply becomes stable. **Note**: There is not a programmable Bank Sleep counter and Standby counter register. The number of clock cycles to transition from sleep to standby and standby to active is hardcoded in the Flash wrapper design.

7.10.14 Flash Bank Power Mode Register (FBPWRMODE)

Figure 7-24. Flash Bank Power Mode Register (FBPWRMODE) (offset = 40h)

31	16
Reserved	
R-505h	

15	14	13	4	3	2	1	0
BANKPWR7		Reserved		BANKPWR1		BANKPWR0	
R/WP-3h		R-3FFh		R/WP-3h		R/WP-3h	

LEGEND: R/W = Read/Write; R = Read only; WP = Write in Privilege Mode; -*n* = value after reset

Table 7-26. Flash Bank Power Mode Register (FBPWRMODE) Field Descriptions

Bit	Field	Value	Description
31-16	Reserved	505h	Do not write to these register bits.
15-14	BANKPWR7		Bank 7 Power Mode.
		0	Bank sleep mode
		1h	Bank standby mode
		2h	Reserved
		3h	Bank active mode
13-4	Reserved	3FFh	Do not write to these register bits.
3-2	BANKPWR1		Bank 1 Power Mode.
		0	Bank sleep mode
		1h	Bank standby mode
		2h	Reserved
		3h	Bank active mode
1-0	BANKPWR0		Bank 0 Power Mode.
		0	Bank sleep mode
		1h	Bank standby mode
		2h	Reserved
		3h	Bank active mode

7.10.15 Flash Bank/Pump Ready Register (FBPRDY)

FBPRDY register allows you to determine if the pump and banks are ready for performing a read access.

Figure 7-25. Flash Bank/Pump Ready Register (FBPRDY) (offset = 44h)

31 24	23	22 18	17 16
Reserved	BANKBUSY[7]	Reserved	BANKBUSY[1:0]
R-0	R-0	R-1	R-0

15	14 8	7	6 2	1 0
PUMPRDY	Reserved	BANKRDY[7]	Reserved	BANKRDY[1:0]
R-1	R-0	R-1	R-0	R-1

LEGEND: R = Read only; -*n* = value after reset

Table 7-27. Flash Bank/Pump Ready Register (FBPRDY) Register Description

Bit	Field	Value	Description
31-24	Reserved	0	Reads return 0. Writes have no effect.
23	BANKBUSY[7]		Bank 7 Busy Status
		0	Bank is not busy with any FSM or CPU operation.
		1	Bank is busy with an FSM or CPU operation.
22-18	Reserved	1	Reads return 1. Writes have no effect.
17-16	BANKBUSY[1:0]		Bank 0 (bit 16) and Bank 1 (bit 17) Busy Status
		0	Bank is not busy with any FSM or CPU operation.
		1	Bank is busy with an FSM or CPU operation.
15	PUMPRDY		Pump Ready is a read-only bit which allows software to determine if the pump is ready for Flash access before attempting the actual access. When set, it means that the charge pump is in active power state. If an access is made to a bank which is not ready then wait states are asserted until it becomes ready
		0	Pump is not ready.
		1	Pump is ready.
14-8	Reserved	0	Reads return 0. Writes have no effect.
7	BANKRDY[7]		Bank 7 Ready Status
		0	Bank is not ready for Flash access.
		1	Bank is ready for Flash access.
6-2	Reserved	0	Reads return 0. Writes have no effect.
1-0	BANKRDY[1:0]		Bank 0 (bit 0) and Bank 1 (bit 1) Ready Status
		0	Bank is not ready for Flash access.
		1	Bank is ready for Flash access.

7.10.16 Flash Pump Access Control Register 1 (FPAC1)

Figure 7-26. Flash Pump Access Control Register 1 (FPAC1) (offset = 48h)

31 27	26 16
Reserved	PSLEEP
R-0	R/WP-C8h

15 1	0
Reserved	PUMPPWR
R-0	R/WP-1

LEGEND: R/W = Read/Write; R = Read only; WP = Write in Privilege Mode; -*n* = value after reset

Table 7-28. Flash Pump Access Control Register 1 (FPAC1) Field Descriptions

Bit	Field	Value	Description
31-12	Reserved	0	Reads return 0. Writes have no effect.
26-16	PSLEEP	0-7FFh	Pump Sleep. These bits contain the starting count value for the charge pump sleep down counter. While the charge pump is in sleep mode, the power mode management logic holds the charge pump sleep counter at this value. When the charge pump exits sleep power mode, the down counter delays from 0 to PSLEEP pump sleep down clock cycles before putting the charge pump into active power mode. **Note:** Pump sleep down counter clock is a divide by 2 input of HCLK. That is, there are 2 × HCLK cycles for every PSLEEP counter cycle.
15-1	Reserved	0	Reads return 0. Writes have no effect.
0	PUMPPWR		Flash Charge Pump Fallback Power Mode
		0	Sleep (all pump circuits are disabled)
		1	Active (all pump circuits are active)

7.10.17 Flash Module Access Control Register (FMAC)

Figure 7-27. Flash Module Access Control Register (FMAC) (offset = 50h)

31	16
Reserved	
R-0	

15	3	2	0
Reserved		BANK	
R-0		R/WP-0	

LEGEND: R/W = Read/Write; R = Read only; WP = Write in Privilege Mode; -*n* = value after reset

Table 7-29. Flash Module Access Control Register (FMAC) Field Descriptions

Bit	Field	Value	Description
31-16	Reserved	0	Reads return 0. Writes have no effect.
2-0	BANK	0-7h	Bank Enable. These bits select which bank is enabled for operations such as local register access, OTP sector access, and program/erase commands. These bits select only one bank at a time from up to eight banks depending on the specific device being used. For example, a 000 selects bank 0; 011 selects bank 3. **Note:** BANK can identify up to 8 Flash banks. If BANK is selected for an un-implemented bank, then the BANK will set itself to the number of an implemented bank. To determine if a bank is implemented, write the bank number to BANK and read back the value to see if what was written can be read back.

7.10.18 Flash Module Status Register (FMSTAT)

Figure 7-28. Flash Module Status Register (FMSTAT) (offset = 54h)

31-24
Reserved
R-0

23-18	17	16
Reserved	RVSUSP	RDVER
R-0	R-0	R-0

15	14	13	12	11	10	9	8
RVF	ILA	DBT	PGV	PCV	EV	CV	BUSY
R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0

7	6	5	4	3	2	1	0
ERS	PGM	INV-DAT	CSTAT	VOLTSTAT	ESUSP	PSUSP	SLOCK
R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0

LEGEND: R = Read only; -*n* = value after reset

Table 7-30. Flash Module Status Register (FMSTAT) Field Descriptions

Bit	Field	Value	Description
31-18	Reserved	0	Reads return 0. Writes have no effect.
17	RVSUSP	1	Read Verify Suspend When set, this bit indicates that the Flash module has received and processed a suspend command during a read-verify operation. This bit remains set until the read-verify-resume command has been issued or the Clear_More command is run.
16	RVDER	1	Read verify command currently underway When set, this bit indicates that the Flash module is actively performing a read-verify operation. This bit is set when read-verify starts and is cleared when it is complete. It is also cleared when the read-verify is suspended and set when the read-verify resumes.
15	RVF	1	Read Verify Failure When set, indicates that a read verify mismatch is detected using the Read Verify command. This bit remains set until clear_status or clear_more FSM commands are run.
14	ILA	1	Illegal Address When set, indicates that an illegal address is detected. The following conditions can set the illegal address flag. 1. Writing to a hole (un-implemented logical address space) within a Flash bank. 2. Writing to an address location to an un-implemented Flash space. 3. Input address for write is decoded to select a different bank from the bank ID register. 4. The address range does not match the type of FSM command. For example, the erase_sector command must match the address regions. 5. TI-OTP address selected but CMD_EN in FSM_ST_MACHINE is not set.
13	DBT	1	Disturbance Test Fail This bit is set during a Program Sector command when the FSM first reads an address and it is not all 1s.
12	PGV	1	Program Verify When set, indicates that a word is not successfully programmed after the maximum allowed number of program pulses are given for program operation.
11	PCV	1	Precondition Verify. When set, indicates that a sector is not successfully preconditioned (pre-erased) after the maximum allowed number of program pulses are given for precondition operation for any applied command such as Erase Sector command. During Precondition verify command, this flag is set immediately if a Flash bit is found to be 1.

Table 7-30. Flash Module Status Register (FMSTAT) Field Descriptions (continued)

Bit	Field	Value	Description
10	EV		Erase Verify
		1	When set, indicates that a sector is not successfully erased after the maximum allowed number of erase pulses are given for erase operation. During Erase verify command, this flag is set immediately if a bit is found to be 0.
9	CV		Compact Verify
		1	When set, indicates that a sector contains one or more bits in depletion after an erase operation with CMPV_ALLOWED set. During compact verify command, this flag is set immediately if a bit is found to be 1.
8	BUSY		Busy
		1	When set, this bit indicates that a program, erase, or suspend operation is being processed.
7	ERS		Erase Active
		1	When set, this bit indicates that the Flash module is actively performing an erase operation. This bit is set when erasing starts and is cleared when erasing is complete. It is also cleared when the erase is suspended and set when the erase resumes.
6	PGM		Program Active
		1	When set, this bit indicates that the Flash module is currently performing a program operation. This bit is set when programming starts and is cleared when programming is complete. It is also cleared when programming is suspended and set when programming is resumes.
5	INVDAT		Invalid Data
		1	When set, this bit indicates that the user attempted to program a 1 where a 0 was already present. This bit is cleared by the Clear Status command.
4	CSTAT		Command Status
		1	Once the FSM starts any failure will set this bit. When set, this bit informs the host that the program, erase, or validate sector command failed and the command was stopped. This bit is cleared by the Clear Status command. For some errors, this will be the only indication of an FSM error because the cause does not fall within the other error bit types.
3	VOLTSTAT		Core Voltage Status
		1	When set, this bit indicates that the core voltage generator of the pump power supply dipped below the lower limit allowable during a program or erase operation. This bit is cleared by the Clear Status command.
2	ESUSP		Erase Suspended
		1	When set, this bit indicates that the Flash module has received and processed an erase suspend operation. This bit remains set until the erase resume command has been issued or until the Clear_More command is run.
1	PSUSP		Program Suspended
		1	When set, this bit indicates that the Flash module has received and processed a program suspend operation. This bit remains set until the program resume command has been issued or until the Clear_More command is run.
0	SLOCK		Sector Lock Status
		1	When set, this bit indicates that the operation was halted because the target sector was locked for erasing and programming either by the sector protect bit or by OTP write protection disable bits. (BSE bits in the FBSE register or OTPPROTDIS bits in the FBAC register). This bit is cleared by the Clear Status command. No SLOCK FSM error will occur if all sectors in a bank erase operation are set to 1. All the sectors will be checked but no SLOCK will be set if no operation occurs due to the SECT_ERASED bits being set to all 1s. A SLOCK error will occur if attempting to do a sector erase with either BSE is cleared or SECT_ERASED is set.

7.10.19 EEPROM Emulation Data MSW Register (FEMU_DMSW)

Figure 7-29. EEPROM Emulation Data MSW Register (FEMU_DMSW) (offset = 58h)

31	16
EMU_DMSW[63:48]	
R/WP-0h	

15	0
EMU_DMSW[47:32]	
R/WP-0h	

LEGEND: R/W = Read/Write; WP = Write in Privilege mode; -*n* = value after reset

Table 7-31. EEPROM Emulation Data MSW Register (FEMU_DMSW) Field Descriptions

Bit	Field	Description
31-0	EMU_DMSW	This register can be written by the CPU in any mode. This register is used in diagnostic mode 7 to XOR the upper 32 bits of the data being delivered to the bus master.

7.10.20 EEPROM Emulation Data LSW Register (FEMU_DLSW)

Figure 7-30. EEPROM Emulation Data LSW Register (FEMU_DLSW) (offset = 5Ch)

31	16
EMU_DLSW[31:16]	
R/WP-0h	

15	0
EMU_DLSW[15:0]	
R/WP-0h	

LEGEND: R/W = Read/Write; WP = Write in Privilege mode; -*n* = value after reset

Table 7-32. EEPROM Emulation Data LSW Register (FEMU_DLSW) Field Descriptions

Bit	Field	Description
31-0	EMU_DLSW	This register can be written by the CPU in any mode. This register is used in diagnostic mode 7 to XOR the lower 32 bits of the data being delivered to the bus master.

7.10.21 EEPROM Emulation ECC Register (FEMU_ECC)

Figure 7-31. EEPROM Emulation ECC Register (FEMU_ECC) (offset = 60h)

31	16
Reserved	
R-0	

15	8	7	0
Reserved		EMU_ECC	
R-0		R/WP-0h	

LEGEND: R/W = Read/Write; R = Read only; WP = Write in Privilege mode; -*n* = value after reset

Table 7-33. EEPROM Emulation ECC Register (FEMU_ECC) Field Descriptions

Bit	Field	Value	Description
31-8	Reserved	0	Reads return 0. Writes have no effect.
7-0	EMU_ECC	0-FFh	This register can be written by the CPU in any mode. This register is used in diagnostic mode 7 to XOR the ECC being delivered to the bus master.

7.10.22 Flash Lock Register (FLOCK)

Figure 7-32. Flash Lock Register (FLOCK) (offset = 64h)

31	16
Reserved	
R-0	

15	0
ENCOM	
R/WP-55AAh	

LEGEND: R/W = Read/Write; R = Read only; WP = Write in Privilege Mode; -*n* = value after reset

Table 7-34. Flash Lock Register (FLOCK) Field Descriptions

Bit	Field	Value	Description
31-16	Reserved	0	Reads return 0. Writes have no effect.
15-0	ENCOM	AA55h	Enable writes to EE_FEDACCTRL1 register (see Section 7.10.3).
		All other values	Writes to EE_FEDACCTRL1 are ignored. It is recommended to leave this register as 55AAh when not writing to the FEDACCTRL1 register.

7.10.23 Diagnostic Control Register (FDIAGCTRL)

First set the DIAGMODE and the DIAG_EN_KEY bits before setting up the other registers to block the other registers from causing a false error. The final write should set the DIAG_TRIG bit to activate the test. Running out of RAM will prevent problems with the diagnostic test corrupting the Flash access in some of the modes.

Figure 7-33. Diagnostic Control Register (FDIAGCTRL) (offset = 6Ch)

31 — 25	24	23 — 20	19 — 16
Reserved	DIAG_TRIG	Reserved	DIAG_EN_KEY
R-0	R/WP-0	R-0	R/WP-Ah

15 — 11	10 — 8	7 — 3	2 — 0
Reserved	DIAG_BUF_SEL	Reserved	DIAGMODE
R-0	R/WP-0	R-0	R/WP-0

LEGEND: R/W = Read/Write; R = Read only; WP = Write in Privilege mode; -*n* = value after reset

Table 7-35. Diagnostic Control Register (FDIAGCTRL) Field Descriptions

Bit	Field	Value	Description
31-25	Reserved	0	Reads return 0. Writes have no effect.
24	DIAG_TRIG		Diagnostic Trigger Diagnostic trigger is the final qualifier for the diagnostic result. After setting all the other diagnostic register values, the DIAG_TRIG is set to 1. This activates the diagnostic logic for one access and then automatically clears the DIAG_TRIG value. The DIAG_EN_KEY and DIAGMODE bits must be set before setting DIAG_TRIG. This bit always reads as 0.
23-20	Reserved	0	Reads return 0. Writes have no effect.
19-16	DIAG_EN_KEY		Diagnostic Enable Key
		5h	Diagnostic mode is enabled.
		All other values	Diagnostic mode is disabled.
15-11	Reserved	0	Reads return 0. Writes have no effect.
10-8	DIAG_BUF_ SEL		Diagnostic Buffer Select
		0	During diagnostic mode 5 the DIAG_BUF_SEL selects the buffer to read or write when accessing the FPRIM_ADD_TAG and FDUP_ADD_TAG registers. The address tags consists of matching primary and duplicate address tag registers. All the primary address tag registers are memory mapped to a common address (see Section 7.10.8) and are selected by DIAG_BUF_SEL. The same occurs for the duplicate address (see Section 7.10.9). Port A has 2 buffers and Port B has 4 buffers. During diagnostic mode 7 the value selects the port on which to perform the diagnostic.
		0	Port A Buffer 0 (diag mode 5) / Port A selected to flip data/ECC (diag mode 7)
		1h	Port A Buffer 1 (diag mode 5) / Reserved in diag mode 7
		2h	Reserved
		3h	Reserved
		4h	Port B Buffer 0 (diag mode 5) / Port B selected to flip data/ECC (diag mode 7)
		5h	Port B Buffer 1 (diag mode 5) / Reserved (diag mode 7)
		6h	Port B Buffer 2 (diag mode 5) / Reserved (diag mode 7)
		7h	Port B Buffer 3 (diag mode 5) / Reserved (diag mode 7)
7-4	Reserved	0	Reads return 0. Writes have no effect.
2-0	DIAGMODE		Diagnostic Mode (Only values 0, 5, and 7 are valid. Other values are reserved).
		0	Diagnostic mode is disabled. This is the same as DIAG_EN_KEY is not equal to 5h.
		5h	Address Tag Register test mode (see Section 7.7.2.1).
		7h	ECC Data Correction Diagnostic test mode (see Section 7.7.2.2).

7.10.24 Raw Address Register (FRAW_ADDR)

Figure 7-34. Raw Address Register (FRAW_ADDR) (offset = 74h)

31 5	4 0
RAW_DATA[31:5]	Reserved
R/WP-u	R-0

LEGEND: R/W = Read/Write; R = Read only; WP = Write in Privilege mode; -u = Unchanged value on internal reset, cleared on power up; -*n* = value after reset

Table 7-36. Raw Address Register (FRAW_ADDR) Field Descriptions

Bit	Field	Description
31-5	RAW_DATA	Raw Address. This register is used during the address tag register test mode, DIAGMODE = 5, to replace the address bus bits 31:3. The lower 5 bits are not compared during the diagnostic.
4-0	Reserved	Reads return 0. Writes have no effect.

7.10.25 Parity Override Register (FPAR_OVR)

This register allows overriding the parity that is internally computed by the L2FMC for checking the parity circuit.

Figure 7-35. Parity Override Register (FPAR_OVR) (offset = 7Ch)

<table>
<tr><td>31</td><td>18</td><td>17</td><td>16</td></tr>
<tr><td colspan="2">Reserved</td><td colspan="2">PAR_OVR_SEL</td></tr>
<tr><td colspan="2">R-0</td><td colspan="2">R/WP-0</td></tr>
</table>

<table>
<tr><td>15</td><td>12</td><td>11</td><td>9</td><td>8</td><td>0</td></tr>
<tr><td colspan="2">PAR_DIS_KEY</td><td colspan="2">PAR_OVR_KEY</td><td colspan="2">Reserved</td></tr>
<tr><td colspan="2">R/WP-5h</td><td colspan="2">R/WP-2h</td><td colspan="2">R-0</td></tr>
</table>

LEGEND: R/W = Read/Write; R = Read only; WP = Write in Privilege Mode; -*n* = value after reset

Table 7-37. Parity Override Register (FPAR_OVR) Field Descriptions

Bit	Field	Value	Description
31-18	Reserved	0	Reserved
17-16	PAR_OVR_SEL		Select which parity checker to invert the polarity of the parity.
		0	No effect.
		1h	Idle state parity checker received inverted parity polarity.
		2h	Command parity checker receives inverted parity polarity.
		3h	Internal address parity checker receives inverted parity polarity
15-12	PAR_DIS_KEY		Disable access Parity. ECC on Data is NOT affected by this setting and behaves the same way.
		Ah	The access parity error is disabled and no checking is done and no events are generated.
		All other values	Any other value enables the parity checking on the access.
11-9	PAR_OVR_KEY		Parity Override
		5h	The selected parity checker selected through PAR_OVR_SEL will receive inverted SYS_ODD_PARITY.
		All other values	Any other value causes the module to use the global system parity bit in the system register DEVCR1.
8-0	Reserved	0	Reads return 0. Writes have no effect.

7.10.26 Reset Configuration Valid Register (RCR_VALID)

This register reflects the validity of the implicit read.

Figure 7-36. Reset Configuration Valid Register (RCR_VALID) (offset = B4h)

31	16
Reserved	
R-0	

15	2	1	0
Reserved		JSM_VALID	RCR_VALID
R-0		R-1	R-1

LEGEND: R = Read only; -*n* = value after reset

Table 7-38. Reset Configuration Valid Register (RCR_VALID) Field Descriptions

Bit	Field	Value	Description
31-2	Reserved	0	Reserved
1	JSM_VALID		When the L2FMC finishes the implicit read, it sets this bit to indicate that the contents of RCR_VALUE0 and RCR_VALUE1 are valid. This bit will be cleared in case there was a double-bit error during implicit reads.
		0	The implicit read has failed. The device level settings may not be correct.
		1	Implicit read is successful. Device level settings are correct.
0	RCR_VALID		When the L2FMC finishes the implicit read, it sets this bit to indicate that the contents of RCR_VALUE0 and RCR_VALUE1 are valid. This bit will be cleared in case there was a double-bit error during implicit reads.
		0	The implicit read has failed. The device level settings may not be correct.
		1	Implicit read is successful. Device level settings are correct.

7.10.27 Crossbar Access Time Threshold Register (ACC_THRESHOLD)

Figure 7-37. Crossbar Access Time Threshold Register (ACC_THRESHOLD) (offset = B8h)

31	16
Reserved	
R-0	

15	12	11	0
Reserved		ACC_THRESH_CNT	
R-0		R/WP-5FFh	

LEGEND: R/W = Read/Write; R = Read only; WP = Write in Privilege Mode; -*n* = value after reset

Table 7-39. Crossbar Access Time Threshold Register (ACC_THRESHOLD) Field Descriptions

Bit	Field	Value	Description
31-12	Reserved	0	Reserved
11-0	ACC_THRESH_CNT	5FFh	Configures maximum number of clocks beyond which the L2FMC internal switch will timeout the access. This can occur due to soft error in internal logic. It is NOT recommended to modify this register unless a crossbar diagnostic is being performed.

7.10.28 Flash Error Detection and Correction Sector Disable Register 2 (FEDACSDIS2)

This register is used to disable the SECDED function on additional two sectors on the EEPROM Emulation Flash (bank 7).

Figure 7-38. Flash Error Detection and Correction Sector Disable Register 2 (FEDACSDIS2) (offset = C0h)

31 30	29 24	23 22	21 16
Rsvd	SectorID3_inverse	Rsvd	SectorID3
R-0	R/WP-0	R-0	R/WP-0

15 14	13 8	7 6	5 0
Rsvd	SectorID2_inverse	Rsvd	SectorID2
R-0	R/WP-0	R-0	R/WP-0

LEGEND: R/W = Read/Write; R = Read only; WP = Write in Privilege Mode; -*n* = value after reset

Table 7-40. Flash Error Detection and Correction Sector Disable Register 2 (FEDACSDIS2) Field Descriptions

Bit	Field	Value	Description
31-30	Reserved	0	Reads return 0. Writes have no effect.
29-24	SectorID3_inverse	0-3Fh	The sector ID inverse bits are used with the sector ID bits to determine which sector is disabled. If the sector ID bits are not pointing to a valid sector (0-3) or the sector ID inverse bits are not an inverse of the sector ID bits, then no sector is disabled by disable ID 3.
23-22	Reserved	0	Reads return 0. Writes have no effect.
21-16	SectorID3	0-3Fh	The sector ID bits are used with the sector ID inverse bits to determine which sector is disabled. If the sector ID bits are not pointing to a valid sector (0-3) or the sector ID inverse bits are not an inverse of the sector ID bits, then no sector is disabled by disable ID 3.
15-14	Reserved	0	Reads return 0. Writes have no effect.
13-8	SectorID2_inverse	0-3Fh	The sector ID inverse bits are used with the sector ID bits to determine which sector is disabled. If the sector ID bits are not pointing to a valid sector (0-3) or the sector ID inverse bits are not an inverse of the sector ID bits, then no sector is disabled by disable ID 2.
7-6	Reserved	0	Reads return 0. Writes have no effect.
5-0	SectorID2	0-3Fh	The sector ID bits are used with the sector ID inverse bits to determine which sector is disabled. If the sector ID bits are not pointing to a valid sector (0-3) or the sector ID inverse bits are not an inverse of the sector ID bits, then no sector is disabled by disable ID 2.

7.10.29 Lower Word of Reset Configuration Read Register (RCR_VALUE0)

When L2FMC completes the implicit read, it populates this register with the lower 32 bits of the data. This is useful to perform a software diagnostic of the SECDED module

Figure 7-39. Lower Word of Reset Configuration Read Register (RCR_VALUE0) (offset = D0h)

31	16
RCR_VALUE[31:16]	
R-u	

15	0
RCR_VALUE[15:0]	
R-u	

LEGEND: R = Read only; -u = Unchanged value on internal reset, cleared on power up; -*n* = value after reset

Table 7-41. Lower Word of Reset Configuration Read Register (RCR_VALUE0) Field Descriptions

Bit	Field	Value	Description
31-0	RCR_VALUE	0	Value of the lower 32 bits of the implicit read. Valid only if RCR_VALID is set.

7.10.30 Upper Word of Reset Configuration Read Register (RCR_VALUE1)

When L2FMC completes the implicit read, it populates this register with the upper 32 bits of the data. This is useful to perform a software diagnostic of the SECDED module

Figure 7-40. Upper Word of Reset Configuration Read Register (RCR_VALUE1) (offset = D4h)

31	16
RCR_VALUE[63:48]	
R-u	

15	0
RCR_VALUE[47:32]	
R-u	

LEGEND: R = Read only; -u = Unchanged value on internal reset, cleared on power up; -*n* = value after reset

Table 7-42. Upper Word of Reset Configuration Read Register (RCR_VALUE1) Field Descriptions

Bit	Field	Value	Description
31-0	RCR_VALUE	Varies with device	Value of the upper 32 bits of the implicit read. Valid only if RCR_VALID is set.

7.10.31 FSM Register Write Enable Register (FSM_WR_ENA)

Figure 7-41. FSM Register Write Enable Register (FSM_WR_ENA) (offset = 288h)

31	16
Reserved	
R-0	

15	3	2	0
Reserved		WR_ENA	
R-0		R/WP-2h	

LEGEND: R/W = Read/Write; R = Read only; WP = Write in Privilege Mode; -*n* = value after reset

Table 7-43. FSM Register Write Enable Register (FSM_WR_ENA) Field Descriptions

Bit	Field	Value	Description
31-3	Reserved	0	Reads return 0. Writes have no effect.
2-0	WR_ENA		FSM Write Enable
		5h	This register must contain 5h in order to write to any other register in the range FFF8 7200h to FFF8 72FFh. This is the first register to be written when setting up the FSM.
		All other values	For all other values, the FSM registers cannot be written.

7.10.32 EEPROM Emulation Configuration Register (EEPROM_CONFIG)

Figure 7-42. EEPROM Emulation Configuration Register (EEPROM_CONFIG) (offset = 2B8h)

31	20	19	16
Reserved		EWAIT	
R-0		R/WP-1	

15	0
Reserved	
R-0	

LEGEND: R/W = Read/Write; R = Read only; WP = Write in Privilege Mode; -*n* = value after reset

Table 7-44. EPROM Emulation Configuration Register (EEPROM_CONFIG) Field Descriptions

Bit	Field	Value	Description
31-20	Reserved	0	Reads return 0. Writes have no effect.
19-16	EWAIT	0-Fh	EEPROM Wait state Counter Replaces the RWAIT count in the EEPROM register. The same formulas that apply to RWAIT apply to EWAIT in the EEPROM bank.
15-0	Reserved	0	Reads return 0. Writes have no effect.

7.10.33 FSM Sector Register 1 (FSM_SECTOR1)

This is a banked register. A separate register is implemented for each bank, but they all occupy the same address. The correct bank must be selected in the FMAC register before reading or writing this register. See Section 7.10.17.

Figure 7-43. FSM Sector Register 1 (FSM_SECTOR1) (offset = 2C0h)

31	16
SECT_ERASED[31:16]	
R/WP-1	

15	0
SECT_ERASED[15:0]	
R/WP-1	

LEGEND: R/W = Read/Write; WP = Write in Privilege Mode; -*n* = value after reset

Table 7-45. FSM Sector Register 1 (FSM_SECTOR1) Field Descriptions

Bit	Field	Value	Description
31-0	SECT_ERASED		Sectors Erased. Each bit corresponds to a Flash sector in the bank specified by the FMAC register. Bit 0 corresponds to sector 0, bit 1 corresponds to sector 1, and so on.
		0	During bank erase, each sector whose corresponding bit is 0 will be erased. After bank erase, the bit corresponding to each sector that is erased will be changed from 0 to 1.
		1	During bank erase, each sector whose corresponding bit is 1 will not be erased.

NOTE: If the bank has less than 32 sectors, only those many LSB bits of FSM_SECTOR1 are valid. For EEPROM bank having more than 32 sectors, use this register in conjunction with FSM_SECTOR2.

7.10.34 FSM Sector Register 2 (FSM_SECTOR2)

This register is applicable to EEPROM bank having more than 32 sectors only. Refer to the device datasheet to find the number of EEPROM sectors in a particular device.

Figure 7-44. FSM Sector Register 2 (FSM_SECTOR2) (offset = 2C4h)

31	16
SECT_ERASED[63:48]	
R/WP-1	

15	0
SECT_ERASED[47:32]	
R/WP-1	

LEGEND: R/W = Read/Write; WP = Write in Privilege Mode; -*n* = value after reset

Table 7-46. FSM Sector Register 2 (FSM_SECTOR2) Field Descriptions

Bit	Field	Value	Description
31-0	SECT_ERASED		Sectors Erased. Each bit corresponds to a Flash sector in the bank specified by the FMAC register. Bit 0 corresponds to sector 32, bit 1 corresponds to sector 33, and so on.
		0	During bank erase, each sector whose corresponding bit is 0 will be erased. After bank erase, the bit corresponding to each sector that is erased will be changed from 0 to 1.
		1	During bank erase, each sector whose corresponding bit is 1 will not be erased.

7.10.35 Flash Bank Configuration Register (FCFG_BANK)

Figure 7-45. Flash Bank Configuration Register (FCFG_BANK) (offset = 400h)

31 — 20	19 — 16
EE_BANK_WIDTH	Reserved
R-48h	R-1

15 — 4	3 — 0
MAIN_BANK_WIDTH	Reserved
R-90h	R-2h

LEGEND: R = Read only; -*n* = value after reset

Table 7-47. Flash Bank Configuration Register (FCFG_BANK) Field Descriptions

Bit	Field	Value	Description
31-20	EE_BANK_WIDTH	48h	Bank 7 width (72-bits wide) This read-only value indicates the maximum number of bits that can be programmed in the bank in one operation. The 72 bits includes 64 data bits and 8 ECC bits.
19-16	Reserved	1	Writes have no effect.
15-4	MAIN_BANK_WIDTH	90h	Width of main Flash banks (288-bits wide) This read-only value indicates the maximum number of bits that can be programmed in the bank in one operation. The 288 bits includes 256 data bits and 32 ECC bits.
3-0	Reserved	2h	Writes have no effect.

7.11 POM Control Registers

This section details the POM module registers listed in Table 7-48.

The POM module control registers can only be read and/or written while in privileged or debug mode. Each register begins on a word boundary. All registers are 32-bit, 16-bit and 8-bit accessible. The start address of the POM module is FFA0 4000h.

Table 7-48. POM Control Registers

Offset	Acronym	Register Description	Section
00h	POMGLBCTRL	POM Global Control Register	Section 7.11.1
04h	POM_REVID	POM Revision ID Register	Section 7.11.2
0Ch	POMFLG	POM Flag Register	Section 7.11.3
200h, 210h, ...	PROMPROGSTARTx	POM Region Start Address Register	Section 7.11.4
204h, 214h,...	POMOVLSTARTx	POM Overlay Start Address Register	Section 7.11.5
208h, 218h,...	POMREGSIZEx	POM Region Size Register	Section 7.11.6

7.11.1 POM Global Control Register (POMGLBCTRL)

Contains enable control for the POM module.

Figure 7-46. POM Global Control Register (POMGLBCTRL) (offset = 00h)

31 22	21 16
OTADDR	Reserved
R/WP-01 1000 0000	R-0

15 4	3 0
Reserved	ON_OFF
R-0	R/WP-5h

LEGEND: R/W = Read/Write; R = Read only; WP = Write in Privilege Mode; -*n* = value after reset

Table 7-49. POM Global Control Register (POMGLBCTRL) Field Descriptions

Bit	Field	Value	Description
31-22	OTADDR		Overlay Target Address. These bits determine the upper address bits of the remapped address. Writing a different value to this bitfield will steer the access to a different location in the 4GB address space. Care has to be taken that the value written represents actual memory.
21-4	Reserved	0	Reads return 0. Writes have no effect.
3-0	ON_OFF		POM enable
		except Ah	POM is disabled.
		Ah	POM is enabled.

7.11.2 POM Revision ID Register (POMREV)

Figure 7-47. POM Revision ID Register (POMREV) (offset = 04h)

31	16
REVID	
R-0108h	

15	0
REVID	
R-CA03h	

LEGEND: R = Read only; -*n* = value after reset

Table 7-50. POM Revision ID Register (POMREV) Field Descriptions

Bit	Field	Value	Description
31-0	REVID	0108CA03h	Revision ID of POM

7.11.3 POM Flag Register (POMFLG)

This register conveys status bits that get set during POM accesses.

All these error status bits can be cleared by writing a 1 to the bit; writing a 0 has no effect.

Figure 7-48. POM Flag Register (POMFLG) (offset = 0Ch)

31	16
Reserved	
R-0	

15 11	10	9	8
Reserved	PERR_SRESP_IDLE	PERR_PB	PERR_PA
R-0	R/W1CP-u	R/W1CP-u	R/W1CP-u

7	0
Reserved	
R-0	

LEGEND: R/W = Read/Write; R = Read only; W1CP = Write 1 to clear in Privilege Mode; -u = unchanged value on internal reset, cleared on power up; -*n* = value after reset

Table 7-51. POM Flag Register (POMFLG) Field Descriptions

Bit	Field	Value	Description
31-11	Reserved	0	Reads return 0. Writes have no effect.
10	PERR_SRESP_IDLE		Idle response parity error on POM access.
		0	Idle response parity error on POM access has NOT occurred.
		1	Idle response parity error on POM access has occurred.
9	PERR_PB		Parity Error on POM access due to remapping request on Port B.
		0	Parity error on POM Port B remap request has NOT occurred.
		1	Parity error on POM Port B remap request has occurred.
8	PERR_PA		Parity Error on POM access due to remapping request on Port A.
		0	Parity error on POM Port A remap request has NOT occurred.
		1	Parity error on POM Port A remap request has occurred.
7-0	Reserved	0	Reads return 0. Writes have no effect.

7.11.4 POM Region Start Address Register (POMPROGSTARTx)

This set of registers contains the start address of each region which is to be remapped. These registers are at an offset 200h + (10h x region number). Region numbers are counted from 0 onwards.

Figure 7-49. POM Region Start Address Register (POMPROGSTARTx) (offset = 200h, 210h,..)

<table>
<tr><td>31</td><td>23</td><td>22</td><td>16</td></tr>
<tr><td colspan="2">Reserved</td><td colspan="2">STARTADDRESS</td></tr>
<tr><td colspan="2">R-0</td><td colspan="2">R/WP-0</td></tr>
</table>

<table>
<tr><td>15</td><td>6</td><td>5</td><td>0</td></tr>
<tr><td colspan="2">STARTADDRESS</td><td colspan="2">Reserved</td></tr>
<tr><td colspan="2">R/WP-0</td><td colspan="2">R-0</td></tr>
</table>

LEGEND: R/W = Read/Write; R = Read only; WP = Write in Privilege Mode; -*n* = value after reset

Table 7-52. POM Region Start Address Register (POMPROGSTARTx) Field Descriptions

Bit	Field	Value	Description
31-23	Reserved	0	Reads return 0. Writes have no effect.
22-6	STARTADDRESS		Start address of the program memory region.
5-0	Reserved	0	Reads return 0. Writes have no effect.

7.11.5 POM Overlay Region Start Address Register (POMOVLSTARTx)

Contains the start address of the overlay region in volatile memory.

Figure 7-50. POM Overlay Region Start Address Register (POMOVLSTARTx) (offset = 204h, 214h,...)

<table>
<tr><td>31</td><td>23</td><td>22</td><td>16</td></tr>
<tr><td colspan="2">Reserved</td><td colspan="2">STARTADDRESS</td></tr>
<tr><td colspan="2">R-0</td><td colspan="2">R/WP-0</td></tr>
</table>

<table>
<tr><td>15</td><td>6</td><td>5</td><td>0</td></tr>
<tr><td colspan="2">STARTADDRESS</td><td colspan="2">Reserved</td></tr>
<tr><td colspan="2">R/WP-0</td><td colspan="2">R-0</td></tr>
</table>

LEGEND: R/W = Read/Write; R = Read only; WP = Write in Privilege Mode; -*n* = value after reset

Table 7-53. POM Overlay Region Start Address Register (POMOVLSTARTx) Field Descriptions

Bit	Field	Value	Description
31-23	Reserved	0	Reads return 0. Writes have no effect.
22-6	STARTADDRESS		Start address of the program memory region.
5-0	Reserved	0	Reads return 0. Writes have no effect.

7.11.6 POM Region Size Register (POMREGSIZEx)

Contains the size of the program memory and overlay memory region.

Figure 7-51. POM Region Size Register (POMREGSIZEx) (offset = 208h, 218h, ...)

31 16
Reserved
R-0

15 4	3 0
Reserved	SIZE
R-0	R/WP-0

LEGEND: R/W = Read/Write; R = Read only; WP=Write in Privilege Mode; -*n* = value after reset

Table 7-54. POM Region Size Register (POMREGSIZEx) Field Descriptions

Bit	Field	Value	Description
31-4	Reserved	0	Reads return 0. Writes have no effect.
3-0	SIZE	0	Region is disabled.
		1h	64 bytes
		2h	128 bytes
		:	:
		Ch	128K bytes
		Dh	256K bytes
		Eh-Fh	Reserved

Chapter 8
SPNU563A–March 2018

Level 2 RAM (L2RAMW) Module

This chapter describes the Level II RAM (L2RAM) module.

Topic **Page**

8.1 Overview

The Level 2 RAM (L2RAM) module controls and decodes RAM memory accesses on this device. Features of the L2RAM are:

- Controls read/write accesses to the data RAM
- Decodes addresses within the memory region allocated for the RAM
- Performs ECC check on all incoming CPU writes to ensure that data is intact
- Supports read and write accesses in 64-bit, 32-bit, 16-bit, or 8-bit access sizes
  - Performs redundant ECC check on merged data during read-modify-write operations
  - Does not support bit-wise operations
- Safety Features:
  - Single-Error-Correction Double-Error-Detection (SECDED) on data
    - Uses the CPU's Event bus and maintains the SECDED status in memory-mapped registers
    - Captures the number of occurrences of single-bit or multi-bit errors as well as the RAM address that has the fault
    - Generates error signals for single-bit and multi-bit errors to the Error Signaling Module (ESM)
  - Performs Memory Scrubbing to Identify and Correct Single Bit RAM errors in the L2RAM memory
  - SECDED Malfunction Checking to Verify that L2RAMW ECC is functioning correctly
  - Parity Protection of the Address Bus and Control Signals
    - Generates error signals for parity error to the Error Signaling Module (ESM)
  - Redundant Address Decode Scheme
    - Checks the decoding of CPU address lines and generation of correct memory selects for the RAM banks
- Exclusive access support
- Supports auto-initialization of the CPU data RAM banks
- Supports the RAM Trace Port (RTP) Interface
  - Traces out all RAM read and write accesses to the RTP module

8.2 Module Operation

8.2.1 RAM Memory Map

The L2RAMW decodes 8MB of data space. Up to 512kB of implemented data space is supported. Check the specific part's datasheet to identify the actual amount of RAM supported on the device. This RAM is protected by ECC, allowing the CPU to correct any single-bit errors and detect multi-bit errors within a 64-bit value. The error correction code (ECC) values are stored in the RAM memory space as well. The memory map for the RAM and the corresponding ECC space is shown in Figure 8-1. Any access to an unimplemented RAM location results in an error response from the L2RAMW module.

Each RAM data word is 64-bits wide. These 64 bits are divided into 32 bits per RAM bank. The 8 bits of ECC are also divided into 4 bits per RAM bank.

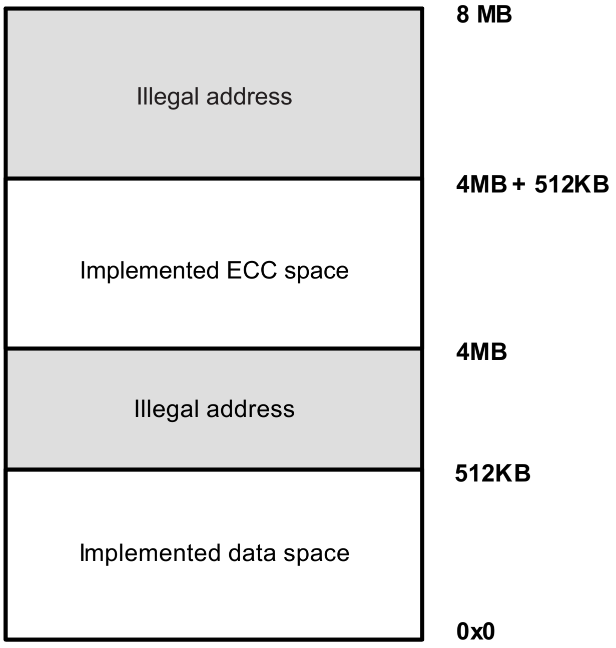
For every 64-bit read from the RAM, an 8-bit ECC is also read by the CPU on its ECC bus. Similarly, for every 64-bit write to the RAM, the CPU also writes an 8-bit ECC using the same ECC bus.

The ECC memory can also be directly accessed via memory-mapped offset addresses. A read from the ECC space results in the 8-bit ECC value appearing on each byte of the 64-bit CPU data. Writes to ECC memory must be 64-bit aligned. Writes to the ECC space must also first be enabled via the RAM Control Register (RAMCTRL).

Accesses to the ECC space are not traced out to the RAM Trace Port (RTP).

NOTE: **No ECC Error Generated for Accesses to ECC Memory:** A read from the ECC memory itself would generate an ECC value on both the read data bus as well as the 8-bit ECC bus. This could result in the detection of a multi-bit error by the SECDED logic inside the CPU. The L2RAMW interface module ignores the ECC errors that are indicated by the CPU when accessing ECC space.

Figure 8-1. RAM Memory Map



8.2.2 Safety Features

The L2RAMW module incorporates some features that are designed specifically with safety considerations.

8.2.2.1 ECC Handling on 8-, 16-, and 32-Bit Writes

ECC calculation is handled by the R5F CPU except in the case of sub-64bit writes. If an 8-, 16-, or a 32-bit write is performed, L2RAMW handles the ECC calculation along with read-modify-write operation. This is to minimize the latency between CPU and L2RAMW in the case of sub-64bit write.

When a sub-64 bit write is performed with ECC enabled, the RAM Error Status Register flags any errors that are detected by the ECC logic of the L2RAMW.

NOTE: The RAM Error Status Register does not indicate ECC errors that are detected by the Cortex R5F CPU. These errors and handled and flagged in the R5F registers

8.2.2.2 Memory Scrubbing

To increase memory reliability, the L2RAMW has an optional "memory scrubbing" feature, which automatically corrects single bit errors whenever they are detected during any RAM read. The reason for performing this action is that if a single bit error occurs on the RAM, and no immediate action is taken to correct it, it is possible that a nearby bit cell will be corrupted as well at some point. If this were to occur, the two corrupted bits would result in a double-bit error that can no longer be corrected by the SECDED algorithm.

Memory scrubbing can be enabled by setting the Memory Scrubbing Enable (MSE) bit in the L2RAMW Module Control Register (RAMCTRL). Note that the ECC Detect Enable (ECC_DETECT_EN) field in RAMCTRL must be set to Ah before enabling memory scrubbing, since memory scrubbing uses the L2RAMW SECDED logic.

8.2.2.3 SECDED Malfunction

To enhance device safety, the L2RAMW has a SECDED malfunction detection feature to ensure that the SECDED logic is functioning correctly. Every time ECC is calculated for a CPU write data or a read data for a read-modify-write operation, the results of the ECC correction are compared back again to the original data value to ensure that the SECDED logic is working correctly. If an error in the SECDED logic is detected, it will be flagged in the RAMERRSTATUS Register (RAM Error Status).

8.2.2.4 L2RAMW Error Types and Responses

Table 8-1. L2RAMW Error Types

Error Source	Corresponding RAMERRSTATUS Bit	ESM Group
CPU Write ECC single error (correctable)	CPUWE (0)	Group 1
ECC double bit errors:		Group 3, bus error
Read-Modify-Write (RMW) ECC double bit error	RMWDE (7)	
CPU Write ECC double bit error	CPUWDE (5)	
Uncorrectable error Type A:		Group 3, bus error
Write SECDED malfunction error	WEME (3)	
Redundant address decode error	ADE (2)	
Read SECDED malfunction error	REME (1)	
Uncorrectable error Type B:		Group 2
Memory scrubbing SECDED malfunction error	MSSM (18)	
Memory scrubbing redundant address decode error	MSRA (17)	
Memory scrubbing address / control parity error	MSACP (16)	
ECC single bit and double bit diagnostic errors	DRDE(22), DRSE(21), DWDE(20), DWSE(19)	
Merged mux diagnostic error	MMDE (12)	
Write SECDED malfunction diagnostic error	WEMDE (11)	
Read SECDED malfunction diagnostic error	REMDE (10)	
Write data merged mux error	MME (9)	
Redundant address decode diagnostic error	ADDE (4)	
Command parity error on idle	CPEOI (15)	
Address / Control parity error	PACE(8)	Group 3, bus error
Level 2 RAM illegal address error	n/a	n/a (bus error only)
Memory initialization error	MIE (13)	n/a (bus error only)

8.2.2.5 Support for Cortex-R5F CPU's Address and Control Bus Parity Checking

The Cortex-R5F CPU provides parity bits for the address and control signals going to L2RAMW. The L2RAMW module also computes the parity bits based on the CPU's address bus and control signals. The computed parity bits are compared against the parity bits received from the CPU. A mismatch is recorded as Address/Control parity error (bit 8) in the RAMERRSTATUS register and signaled as an Address Parity Failure to the Error Signaling Module (ESM). It also generates a bus error.

The error flag in the RAMERRSTATUS register must be cleared by the application in order for the L2RAMW interface module to continue capturing subsequent errors and error addresses.

NOTE: **No Change Of Parity Scheme On-The-Fly:** The L2RAMW interface module does not support on-the-fly change to the parity scheme being used for checking the CPU address bus and control bus. The application must ensure that the parity polarity (odd or even) is not changed while there is an ongoing access to the L2RAM.

8.2.2.6 Redundant Address Decode

The L2RAMW module generates the memory selects for each of the L2RAMW banks as well as the ECC memory based on the CPU address. The logic to generate these memory selects is duplicated and the outputs compared to detect any address decode errors. A mismatch is indicated as an Address Error to the Error Signaling Module (ESM). The L2RAMW or ECC address that caused the fault is captured in the RAMUERRADDR register. This is a 64-bit address that is stored as an offset from the base of the L2RAMW or ECC memory.

As described earlier, each individual physical RAM bank is 36 bits wide. Each RAM bank contributes 32 bits of data and 4 bits of ECC when the bus master performs a 64-bit read from the L2RAM. Each L2RAMW bank receives a memory select and the address from the L2RAMW interface module. Any difference between the address and the memory selects results in wrong data and ECC pair being sent to the CPU. The CPU's SECDED block will detect this data error.

The L2RAMW interface module also supports a mechanism to test the operation of the redundant address decode logic and the compare logic. This testing is supported by providing a test stimulus, and can be triggered by the application by configuring the RAMTEST register. The address of any error identified during testing of the redundant address decode and compare logic is not captured in the RAMUERRADDR register.

NOTE: **Address decode checking when in compare logic test mode:** When the address decode and compare logic test mode is enabled, the redundant address decode and compare logic is not available for checking the proper generation of the memory selects for the L2RAMW and ECC memory.

8.2.3 L2RAMW Auto-Initialization

The RAM memory can be initialized by using the dedicated auto-initialization hardware. The L2RAMW module initializes the entire memory when the auto-init is enabled for the RAM. All RAM data memory is initialized to zeros and the ECC memory is initialized to the correct ECC value for zeros, that is, 0Ch.

8.2.4 Trace Module Support

The L2RAMW module traces out the following signals to the RAM Trace Port (RTP) module, thereby providing RAM dataport trace capability.

- 18-bit address line
- 64-bit data bus
- Byte strobe information
- Current access master identification number
- Access type: Opcode or data fetch
- Read or Write access

No data is traced for an access to ECC memory.

8.2.5 Emulation/Debug Mode Behavior

The following describes the behavior of the L2RAMW Module when in debug mode:

- No single-bit error interrupt is generated nor is any single-bit error address captured even when the RAMOCCUR counter reaches the programmed single-bit error correction threshold.
- No uncorrectable error interrupt is generated nor is any double-bit error address captured.
- No address parity error interrupt is generated nor is any parity error address captured.
- The RAMUERRADDR register is not cleared by a read in debug mode.
  - That is, if a double-bit error address is captured and is not read by the CPU before entering debug mode, then it remains frozen during debug mode even if it is read.
- The RAMPERRADDR register is not cleared by a read in debug mode.

8.2.6 Diagnostic Test Procedure

1. Write test vectors DIAG_DATA_VECTOR_H, DIAG_DATA_VECTOR_L, DIAG_ECC, and RAMADDRDEC_VECT with desire test irritants.
2. In RAMTEST, write TEST_ENABLE field with Ah and TEST_MODE field with the choice of inequality or equality testing for redundant address decoding and SECDED multifunction diagnostics. Set up proper values in DIAG_ECC, DIAG_DATA_VECTOR_L and DIAG_DATA_VECTOR_H registers. ECC single bit or double bit read and write diagnostic errors will be generated if the values do not match.
3. In RAMTEST, write TRIGGER bit. Remember the trigger can only be enabled when TEST_ENABLE is equal to Ah and RAMERRSTATUS[22,21,20,19,12,11,10, 4] bits are zero. Triggering diagnostic test while the memory banks are busy will force the test to wait until the banks are free. Note all diagnostic testing for two SECDEDs and compare logics of redundant address decode, two SECDED malfunctions, data merging block are completed in one HCLK cycle even though the TRIGGER bit can last one VCLK cycle.
4. Read back register bits RAMERRSTATUS[22,21,20,19,12,11,10, 4] and observe pass/fail status. No error bit will be set if no error is detected in the diagnostic test. The diagnostic errors will also be sent to ESM group 2 as "uncorrectable error type B".

8.3 Control and Status Registers

The L2RAMW Module registers listed in Table 8-2 are accessed through the system module register space in the Cortex-R5F CPUs memory map. All registers are 32-bit wide and are located on a 32-bit boundary. Reads and writes to registers are supported in 8-, 16-, and 32-bit accesses. The base address for the L2RAMW control registers is FFFF F900h.

Table 8-2. L2RAMW Module Control and Status Registers

Offset	Acronym	Register Description	Section
00h	RAMCTRL	L2RAMW Module Control Register	Section 8.3.1
10h	RAMERRSTATUS	L2RAMW Module Error Status Register	Section 8.3.2
24h	DIAG_DATA_VECTOR_H	Diagnostic Data Vector High Register	Section 8.3.3
28h	DIAG_DATA_VECTOR_L	Diagnostic Data Vector Low Register	Section 8.3.4
2Ch	DIAG_ECC	Diagnostic ECC Vector Register	Section 8.3.5
30h	RAMTEST	L2RAMW RAM Test Register	Section 8.3.6
38h	RAMADDRDEC_VECT	L2RAMW RAM Address Decode Vector Test Register	Section 8.3.7
3Ch	MEMINIT_DOMAIN	L2RAMW Memory Initialization Domain Register	Section 8.3.8
44h	BANK_DOMAIN_MAP0	Bank to Domain Mapping Register 0	Section 8.3.9
48h	BANK_DOMAIN_MAP1	Bank to Domain Mapping Register 1	Section 8.3.10

8.3.1 L2RAMW Module Control Register (RAMCTRL)

The RAMCTRL register, shown in Figure 8-2 and described in Table 8-3, controls the safety features supported by the L2RAMW Module.

Figure 8-2. L2RAMW Module Control Register (RAMCTRL) (offset = 00h)

31	30	29 28	27 24
Reserved	EMU_TRACE_DIS	Reserved	ADDR_PARITY_OVERRIDE
R-0	R/WP-0	R-0	R/WP-5h

23 21	20	19 16
Reserved	MSE	ADDR_PARITY_DISABLE
R-0	R/WP-0	R/WP-5h

15 13	12	11 7	8
Reserved	EEMMS	Reserved	ECC_WR_EN
R-0	R/WP-0	R-0	R/WP-0

7 5	4	3 0
Reserved	CPUWSC	ECC_DETECT_EN
R-0	R/WP-0	R/WP-Ah

LEGEND: R/W = Read/Write; R=Read only; WP = Write allowed in privileged mode only; -*n* = value after reset

Table 8-3. L2RAMW Module Control Register (RAMCTRL) Field Descriptions

Bit	Field	Value	Description
31	Reserved	0	Reads return 0. Writes have no effect.
30	EMU_TRACE_DIS		Emulation Mode Trace Disable. This bit, when set, disables the tracing of read data to RAM Trace Port (RTP) module during emulation mode access.
		0	Data is allowed to be traced out to the trace modules for emulation mode accesses.
		1	Data is blocked from being traced out to the trace modules for emulation mode accesses.
29-28	Reserved	0	Reads return 0. Writes have no effect.

Table 8-3. L2RAMW Module Control Register (RAMCTRL) Field Descriptions (continued)

Bit	Field	Value	Description
27-24	ADDR_PARITY_OVERRIDE		Address Parity Override. This field, when set to Ah, will invert the parity scheme selected by the device global parity selection. The address parity checker would then work on the inverted parity scheme. By default, the parity scheme is the same as the global device parity scheme.
		Ah	Parity scheme is opposite to the device global parity scheme.
		All other values	Parity scheme is the same as the device global parity scheme.
23-21	Reserved	0	Reads return 0. Writes have no effect.
20	MSE		MSE: Memory Scrubbing Enable. This bit enables or disables memory scrubbing of single-bit errors on read operations. **Note: The ECC_DETECT_EN field must be set to Ah before enabling memory scrubbing, since memory scrubbing uses the L2RAMW SECDED logic.**
		0	Memory scrubbing is disabled.
		1	Memory scrubbing is enabled.
19-16	ADDR_PARITY_DISABLE		Address/Control Bus Parity Detect Disable. This field, when set to Ah, disables the parity checking for the address and control bus. The parity checking is enabled when this field is set to any other value. **Note: The application must ensure that PACE field in RAMERRSTATUS register is cleared before enabling address/control bus parity checking.**
		Ah	Address parity checking is disabled.
		All other values	Address parity checking is enabled.
15-13	Reserved	0	Reads return 0. Writes have no effect.
12	EEMMS		Enable ESM notification (Parity, Redundant Address Decode, SECDED malfunction) for write back during memory scrubbing.
		0	ESM will not be signaled when an error occurs during memory scrubbing write back.
		1	ESM will be signaled when an error occurs during memory scrubbing write back.
11-9	Reserved	0	Reads return 0. Writes have no effect.
8	ECC_WR_EN		ECC Memory Write Enable. This bit is provided to prevent accidental writes to the ECC memory. A write access to the ECC memory is allowed only when the ECC_WR_EN bit is set to 1. If this bit is cleared, then any writes to ECC memory are ignored. Note: Reads are allowed from the ECC memory regardless of the state of the ECC_WR_EN.
		0	ECC memory writes are disabled.
		1	ECC memory writes are enabled.
7-5	Reserved	0	Reads return 0. Writes have no effect.
4	CPUWSC		CPUWSC: CPU Write SERR Capture. By default, single bit error are not signaled to ESM module. This bit allows the option to capture the status and notify ESM. **Note: This feature is only applicable to CPU write data.**
		0	Disable single bit error status capture and ESM notification.
		1	Enable single bit error status capture and ESM notification.
3-0	ECC_DETECT_EN		ECC Detect Enable. This is a 4-bit key to enable the ECC detection feature in the L2RAMW Module. Error detection, status updates, and data correction are performed by the L2RAMW logic only if ECC detection is enabled. ECC detection is enabled by default after reset. **Note: Disabling ECC on the L2RAMW module will disable ECC error checking only for the ECC functions that the L2RAM handles (sub 64-bit Write Operations). All other ECC handling is done by the R5F CPU. ECC error checking cannot be disabled on the R5F CPU.**
		5h	ECC detection is disabled.
		All other values	ECC detection is enabled.

8.3.2 L2RAMW Error Status Register (RAMERRSTATUS)

The RAMERRSTATUS register, shown in Figure 8-3 and described in Table 8-4, indicates the status of the various error conditions monitored by the L2RAMW Module.

Figure 8-3. L2RAMW Module Error Status Register (RAMERRSTATUS) (offset = 10h)

31-24
Reserved
R-0

23	22	21	20	19	18	17	16
Reserved	DRDE	DRSE	DWDE	DWSE	MSSM	MSRA	MSACP
R-0	R/W1CP-0	R/W1CP-0	R/W1CP-0	R/W1CP-0	R/W1CP-0	R/W1CP-0	R/W1CP-0

15	14	13	12	11	10	9	8
CPEOI	Reserved	MIE	MMDE	WEMDE	REMDE	MME	PACE
R/W1CP-0	R-0	R/W1CP-0	R/W1CP-0	R/W1CP-0	R/W1CP-0	R/W1CP-0	R/W1CP-0

7	6	5	4	3	2	1	0
RMWDE	Reserved	CPUWDE	ADDE	WEME	ADE	REME	CPUWE
R/W1CP-0	R-0	R/W1CP-0	R/W1CP-0	R/W1CP-0	R/W1CP-0	R/W1CP-0	R/W1CP-0

LEGEND: R/W = Read/Write; R = Read only; W1CP = Privilege Write 1 to Clear; -*n* = value after reset

Table 8-4. L2RAMW Module Error Status Register (RAMERRSTATUS) Field Descriptions

Bit	Field	Value	Description
31-23	Reserved	0	Reads return 0. Writes have no effect.
22	DRDE		Diagnostic Read Double-bit Error. This bit indicates that a double-bit error has occurred during diagnostic of the L2RAMW SECDED logic that is used to handle read of read-modify write operations. This bit must be cleared by writing a 1 to it before any new error can be generated.
		0	A double-bit error did not occur during diagnostic.
		1	A double-bit error occurred during diagnostic.
21	DRSE		Diagnostic Read Single-bit Error. This bit indicates that a single-bit error has occurred during diagnostic of the L2RAMW SECDED logic that is used to handle read of read-modify write operations. This bit must be cleared by writing a 1 to it before any new error can be generated.
		0	A single-bit error did not occur during diagnostic.
		1	A single-bit error occurred during diagnostic.
20	DWDE		Diagnostic Write Double-bit Error. This bit indicates that a double-bit error has occurred during diagnostic of the L2RAMW SECDED logic that handles write operations. This bit must be cleared by writing a 1 to it before any new error can be generated.
		0	A double-bit error did not occur during diagnostic.
		1	A double-bit error occurred during diagnostic.
19	DWSE		Diagnostic Write Single-bit Error. This bit indicates that a single-bit error has occurred during diagnostic of the L2RAMW SECDED logic that handles write operations. This bit must be cleared by writing a 1 to it before any new error can be generated.
		0	A single-bit error did not occur during diagnostic.
		1	A single-bit error occurred during diagnostic.
18	MSSM		Memory Scrubbing write back SECDED Malfunction. This indicates that a SECDED malfunction occurred during memory scrubbing write back. This bit must be cleared by writing a 1 to it before any new error can be generated.
		0	A SECDED malfunction did not occur during scrubbing write back.
		1	A SECDED malfunction occurred during scrubbing write back.
17	MSRA		Memory Scrubbing write back Redundant Address decode error. This bit indicates that a redundant address decode error occurred during memory scrubbing write back. This bit must be cleared by writing a 1 to it before any new error can be generated.
		0	An address decode error did not occur during scrubbing write back.
		1	An address decode error occurred during scrubbing write back.

Table 8-4. L2RAMW Module Error Status Register (RAMERRSTATUS) Field Descriptions (continued)

Bit	Field	Value	Description
16	MSACP		Memory Scrubbing write back Redundant Address decode error. This bit indicates that an address-control parity error occurred during memory scrubbing write back. This bit must be cleared by writing a 1 to it before any new error can be generated.
		0	An address control parity error did not occur during memory scrubbing write back.
		1	An address control parity error occurred during memory scrubbing write back.
15	CPEOI		Command Parity Error on Idle. This bit indicates an error occurred for an idle command with parity error. This bit must be cleared by writing a 1 to it before any new error can be generated.
		0	An error did not occur.
		1	An error occurred.
14	Reserved	0	Reads return 0. Writes have no effect.
13	MIE		Memory Initialization Error. This bit indicated an error occurred for an access to a bank under memory initialization. Access to a bank under memory initialization is not allowed. It will result in a false double bit error. This bit must be cleared by writing a 1 to it before any new error can be generated.
		0	An error did not occur.
		1	An error occurred.
12	MMDE		Merged MUX Diagnostic Error. This bit indicates a error was detected on the compare logic of the mux logic used for data merging of a read modify write operation during diagnostic test. This bit must be cleared by writing a 1 to it before any new error can be generated.
		0	An error did not occur.
		1	An error occurred.
11	WEMDE		Write ECC Malfunction Diagnostic Error. This bit indicated an error was detected on the compare logic of the write ECC malfunction during diagnostic test. This bit must be cleared by writing a 1 to it before any new error can be generated.
		0	An error did not occur.
		1	An error occurred.
10	REMDE		Read ECC Malfunction Diagnostic Error. This bit indicated an error was detected on the compare logic of the read ECC malfunction during diagnostic test. This bit must be cleared by writing a 1 to it before any new error can be generated.
		0	An error did not occur.
		1	An error occurred.
9	MME		Merged Mux Error. This bit indicates an error was detected on the mux logic that is used to merge the corrected read and write data for a read modify write operation. This bit must be cleared by writing a 1 to it before any new error can be generated.
		0	An error did not occur.
		1	An error occurred.
8	PACE		Address and/or Control bus Parity Error. This bit must cleared by writing a 1 to it before any new error can be generated.
		0	An error did not occur.
		1	An error occurred.
7	RMWDE		Read-Modify-Write Double Bit Error. This bit indicates that an ECC uncorrectable (double bit) error was detected during read access of the read modify write operation. This bit must be cleared by writing a 1 to it before any new error can be generated.
		0	An error did not occur.
		1	An error occurred.
6	Reserved	0	Reads return 0. Writes have no effect.
5	CPUWDE		CPU Write Double-bit Error. This bit indicates that an ECC uncorrectable (double bit) error was detected during write access. This bit must be cleared by writing a 1 to it before any new error can be generated.
		0	An error did not occur.
		1	An error occurred.

Table 8-4. L2RAMW Module Error Status Register (RAMERRSTATUS) Field Descriptions (continued)

Bit	Field	Value	Description
4	ADDE		Redundant address decoding diagnostic error. This bit indicates that the redundant address decode logic diagnostic test has detected that a compare element has malfunctioned during the testing of the logic. This bit is only set in test mode. This bit must be cleared by writing a 1 to it for generation of any new uncorrectable error interrupt in non-test mode.
		0	An error did not occur.
		1	An error occurred.
3	WEME		Write ECC Malfunction Error. This bit Indicates that the SECDED logic failed to correct a single bit error during a CPU write operation. This bit must be cleared by writing a 1 to it before any new error can be generated.
		0	An error did not occur.
		1	An error occurred.
2	ADE		Address Decode Error. This bit indicates than an address error was generated by the redundant address decode logic due to a functional failure. This bit must be cleared by writing a 1 to it before any new error can be generated.
		0	An error did not occur.
		1	An error occurred.
1	REME		Read ECC Malfunction Error. Indicates that the SECDED logic failed to correct a single bit error on the read of a read-modify-write operation. This bit must be cleared by writing a 1 to it before any new error can be generated.
		0	An error did not occur.
		1	An error occurred.
0	CPUWE		CPU Write Single Error. This bit indicates that a single-bit error occurred during write access. This bit must be cleared by writing 1 to it in order to clear the interrupt request and to enable subsequent single-bit error interrupt generation.
		0	An error did not occur.
		1	An error occurred.

8.3.3 L2RAMW Diagnostic Data Vector High Register (DIAG_DATA_VECTOR_H)

The DIAG_DATA_VECTOR_H register, shown in Figure 8-4 and described in Table 8-5, is used in conjunction with the RAMTEST register to perform diagnostic tests.

Figure 8-4. L2RAMW Diagnostic Data Vector High Register (DIAG_DATA_VECTOR_H) (offset = 24h)

31 … 0
DIAGNOSTIC_VECTOR[63:32]
R/WP-0

LEGEND: R/W = Read/Write; WP = Write in privilege mode only; -*n* = value after reset

Table 8-5. L2RAMW Diagnostic Data Vector High Register (DIAG_DATA_VECTOR_H) Field Descriptions

Bit	Field	Description
31-0	DIAGNOSTIC_VECTOR	Used in conjunction with DIAG_DATA_VECTOR_L to form a 64-bit test vector used for diagnostic test of two SECDEDs (read and write) and compare logic of the two SECDED malfunctions and merged mux. This register is the upper 32 bits. This register is used in conjunction with the RAMTEST register to perform diagnostic tests. See Section 8.2.6 for details on how to start a diagnostic test.

8.3.4 L2RAMW Diagnostic Data Vector Low Register (DIAG_DATA_VECTOR_L)

The DIAG_DATA_VECTOR_L, shown in Figure 8-5 and described in Table 8-6, is used in conjunction with the RAMTEST register to perform diagnostic tests.

Figure 8-5. L2RAMW Diagnostic Vector Low Register (DIAG_DATA_VECTOR_L) (offset = 28h)

31 … 0
DIAGNOSTIC_VECTOR[31:0]
R/WP-0

LEGEND: R/W = Read/Write; WP = Write in privilege mode only; -*n* = value after reset

Table 8-6. L2RAMW Diagnostic Vector Low Register (DIAG_DATA_VECTOR_L) Field Descriptions

Bit	Field	Description
31-0	DIAGNOSTIC_VECTOR	Used in conjunction with DIAG_DATA_VECTOR_H to form a 64-bit test vector used for diagnostic test of two SECDEDs (read and write) and compare logic of the two SECDED malfunctions and merged mux. This register is the lower 32 bits. This register is used in conjunction with the RAMTEST register to perform diagnostic tests. See Section 8.2.6 for details on how to start a diagnostic test.

8.3.5 L2RAMW Diagnostic ECC Vector Register (DIAG_ECC)

The DIAG_ECC register, shown in Figure 8-6 and described in Table 8-7, captures the address for which the Cortex-R5F CPU detected a multi-bit error.

Figure 8-6. L2RAMW Diagnostic ECC Vector Register (DIAG_ECC) (offset = 2Ch)

31–16
Reserved
R-0

15–8	7–0
Reserved	DIAG_ECC_VECTOR
R-0	R/WP-U

LEGEND: R/W = Read/Write; R=Read only; WP = Write in privilege mode only; U = Unknown; -*n* = value after reset

Table 8-7. L2RAMW Diagnostic ECC Vector Register (DIAG_ECC) Field Descriptions

Bit	Field	Value	Description
31-8	Reserved	0	Reads return 0. Writes have no effect.
7-0	DIAG_ECC_VECTOR	0-FFh	Diagnostic ECC Vector. This field provides an 8-bit ECC test vector used for diagnostic test of the two SECDEDs and compare logic for two SECDED malfunctions and merged mux. This register is used in conjunction with DIAG_DATA_VECTOR_H and DIAG_DATA_VECTOR_L registers to form a data/ECC pair in the diagnostic ECC checking test. See Section 8.2.6 for details on how to start a diagnostic test.

8.3.6 L2RAMW RAM Test Mode Control Register (RAMTEST)

The RAMTEST register, shown in Figure 8-7 and described in Table 8-8, controls the test mode of the L2RAMW Module.

Figure 8-7. L2RAMW Module Test Mode Control Register (RAMTEST) (offset = 30h)

31	16
Reserved	
R-0	

15	9	8	7	6	5	4	3	0
Reserved		TRIGGER	TEST_MODE		Reserved		TEST_ENABLE	
R-0		R/WP-0	R/WP-0		R-0		R/WP-5h	

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privileged mode only; -*n* = value after reset

Table 8-8. L2RAMW Module Test Mode Control Register (RAMTEST) Field Descriptions

Bit	Field	Value	Description
31-9	Reserved	0	Reads return 0. Writes have no effect.
8	TRIGGER		Test Trigger. This is an auto clear test trigger used to test the redundant address decode, data merging mux, SECDED malfunction compare logic, and ECC checking logics. The diagnostic test is executed when test mode is enabled and the test trigger is applied by writing a 1 to this bit. The trigger is valid only if test mode is enabled, the correct mode is configured in the TEST_MODE field, and all diagnostic error bits in the RAMERRSTATUS register are in the cleared state. The trigger bit is auto clear after the test and has to be written again for a new test.
7-6	TEST_MODE		Test Mode. This field selects either equality or inequality testing schemes for redundant address decoding and SECDED malfunction diagnostics. If TEST_MODE is set to 2h, equality check is done. The test stimulus stored in RAMADDRDEC_VECT register is fed directly to both the channels of the comparator. If the XOR of these two inputs **is not zero**, then UERR interrupt is generated and ADDE flag is set in RAMERRSTATUS register. If TEST_MODE is set to 1h, inequality check is done. The test stimulus stored in RAMADDRDEC_VECT register is inverted and fed into one channel and the non-inverted vector is fed into the other channel. If the XOR of these inputs **is zero**, then the UERR interrupt is generated and ADDE flag is set in RAMERRSTATUS register.
5-4	Reserved	0	Reads return 0. Writes have no effect.
3-0	TEST_ENABLE		Test Enable. This is a 4-bit key to enable the redundant address decode, SECDED malfunction, data merging mux and ECC checking diagnostics. If the test scheme is enabled, then the compare logic uses the test vector inputs from the RAMADDRDEC_VECT, DIAG_ECC, DIAG_DATA_VECTOR_L, and DIAG_DATA_VECTOR_H registers. The functional path comparison is disabled when test mode is enabled.
		Ah	Test mode is enabled.
		All other values	Test mode is disabled.

8.3.7 L2RAMW RAM Address Decode Vector Test Register (RAMADDRDEC_VECT)

The RAMADDRDEC_VECT register, shown in Figure 8-8 and described in Table 8-9, is used for testing the redundant address decode and compare logic of the L2RAMW Module.

Figure 8-8. L2RAMW RAM Address Decode Vector Test Register (RAMADDRDEC_VECT) (offset = 38h)

31 27	26	25 16
Reserved	DESV	Reserved
R-0	R/WP-0	R-0

15 0
RAM_CHIP_SELECT
R/WP-0

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privileged mode only; -*n* = value after reset

Table 8-9. L2RAMW RAM Address Decode Vector Test Register (RAMADDRDEC_VECT) Field Descriptions

Bit	Field	Value	Description
31-27	Reserved	0	Reads return 0. Writes have no effect.
26	DESV		Diagnostic ECC Select Vector. This bit is used to store the ECC select test vector for the redundant address decode test logic. The stored value is passed as test stimulant for the built in test scheme.
25-16	Reserved	0	Reads return 0. Writes have no effect.
15-0	RAM_CHIP_SELECT	0-FFFFh	RAM Chip Select. This field is used to store the RAM chip select value for the redundant address decode and compare logic. The stored value is passed as test stimulus for the built-in test scheme.

8.3.8 L2RAMW Memory Initialization Domain Register (MEMINIT_DOMAIN)

The MEMINIT_DOMAIN register, shown in Figure 8-9 and described in Table 8-10, stores the address for which an address-parity error was detected.

Figure 8-9. L2RAMW Memory Initialization Domain Register (MEMINIT_DOMAIN) (offset = 3Ch)

31	16
Reserved	
R-0	

15	8	7	0
Reserved		MEMINIT_ENA	
R-0		R/WP-FFh	

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privileged mode only; -*n* = value after reset

Table 8-10. L2RAMW Memory Initialization Domain Register (MEMINIT_DOMAIN) Field Descriptions

Bit	Field	Value	Description
31-8	Reserved	0	Reads return 0. Writes have no effect.
7-0	MEMINIT_ENA[*n*]		Memory Initialization Enable. Each bit *n* corresponds to an individual memory domain. If the corresponding bit is set to 1 when an initialization of the RAM memory is executed, then that section of the RAM memory will be initialized. If the corresponding bit is cleared to 0 when an initialization of the RAM memory is executed, then that section of the RAM memory will not be affected. After reset, all memory power domains are enabled (set to 1) by default. Bit 0: enable bit for power domain 0. Bit 1: enable bit for power domain 1. : Bit 7: enable bit for power domain 7.
		1	Enable the memory in this power domain to be initialized.
		0	Disable the memory in this power domain from being initialized.

8.3.9 L2RAMW Bank to Domain Mapping Register0 (BANK_DOMAIN_MAP0)

The BANK_DOMAIN_MAP0 register, shown in Figure 8-10 and described in Table 8-11, stores the address for which an address-parity error was detected.

Figure 8-10. L2RAMW Bank to Domain Mapping Register0 (BANK_DOMAIN_MAP0) (offset = 44h)

31	30 28	27	26 24	23	22 20	19	18 16
Rsvd	BANK7_MAP	Rsvd	BANK6_MAP	Rsvd	BANK5_MAP	Rsvd	BANK4_MAP
R-0	R-DS	R-0	R-DS	R-0	R-DS	R-0	R-DS

15	14 12	11	10 8	7	6 4	3	2 0
Rsvd	BANK3_MAP	Rsvd	BANK2_MAP	Rsvd	BANK1_MAP	Rsvd	BANK0_MAP
R-0	R-DS	R-0	R-DS	R-0	R-DS	R-0	R-DS

LEGEND: R = Read only; DS = Device Specific; -*n* = value after reset

Table 8-11. L2RAMW Bank to Domain Mapping Register0 (BANK_DOMAIN_MAP0) Field Descriptions

Bit	Field	Value	Description
31	Reserved	0	Reads return 0. Writes have no effect.
30-28	BANK7_MAP	0-7h	This 3-bit field allows the software to read the memory power domain number that bank 7 is associated.
27	Reserved	0	Reads return 0. Writes have no effect.
26-24	BANK6_MAP	0-7h	This 3-bit field allows the software to read the memory power domain number that bank 6 is associated.
23	Reserved	0	Reads return 0. Writes have no effect.
22-20	BANK5_MAP	0-7h	This 3-bit field allows the software to read the memory power domain number that bank 5 is associated.
19	Reserved	0	Reads return 0. Writes have no effect.
18-16	BANK4_MAP	0-7h	This 3-bit field allows the software to read the memory power domain number that bank 4 is associated.
15	Reserved	0	Reads return 0. Writes have no effect.
14-12	BANK3_MAP	0-7h	This 3-bit field allows the software to read the memory power domain number that bank 3 is associated.
11	Reserved	0	Reads return 0. Writes have no effect.
10-8	BANK2_MAP	0-7h	This 3-bit field allows the software to read the memory power domain number that bank 2 is associated.
7	Reserved	0	Reads return 0. Writes have no effect.
6-4	BANK1_MAP	0-7h	This 3-bit field allows the software to read the memory power domain number that bank 1 is associated.
3	Reserved	0	Reads return 0. Writes have no effect.
2-0	BANK0_MAP	0-7h	This 3-bit field allows the software to read the memory power domain number that bank 0 is associated.

8.3.10 L2RAMW Bank to Domain Mapping Register1 (BANK_DOMAIN_MAP1)

The BANK_DOMAIN_MAP1 register, shown in Figure 8-11 and described in Table 8-12, stores the address for which an address-parity error was detected.

Figure 8-11. L2RAMW Bank to Domain Mapping Register1 (BANK_DOMAIN_MAP1) (offset = 48h)

31	30 28	27	26 24	23	22 20	19	18 16
Rsvd	BANK15_MAP	Rsvd	BANK14_MAP	Rsvd	BANK13_MAP	Rsvd	BANK12_MAP
R-0	R-DS	R-0	R-DS	R-0	R-DS	R-0	R-DS

15	14 12	11	10 8	7	6 4	3	2 0
Rsvd	BANK11_MAP	Rsvd	BANK10_MAP	Rsvd	BANK9_MAP	Rsvd	BANK8_MAP
R-0	R-DS	R-0	R-DS	R-0	R-DS	R-0	R-DS

LEGEND: R = Read only; DS = Device Specific; -*n* = value after reset

Table 8-12. L2RAMW Bank to Domain Mapping Register1 (BANK_DOMAIN_MAP1) Field Descriptions

Bit	Field	Value	Description
31	Reserved	0	Reads return 0. Writes have no effect.
30-28	BANK15_MAP	0-7h	This 3-bit field allows the software to read the memory power domain number that bank 15 is associated.
27	Reserved	0	Reads return 0. Writes have no effect.
26-24	BANK14_MAP	0-7h	This 3-bit field allows the software to read the memory power domain number that bank 14 is associated.
23	Reserved	0	Reads return 0. Writes have no effect.
22-20	BANK13_MAP	0-7h	This 3-bit field allows the software to read the memory power domain number that bank 13 is associated.
19	Reserved	0	Reads return 0. Writes have no effect.
18-16	BANK12_MAP	0-7h	This 3-bit field allows the software to read the memory power domain number that bank 12 is associated.
15	Reserved	0	Reads return 0. Writes have no effect.
14-12	BANK11_MAP	0-7h	This 3-bit field allows the software to read the memory power domain number that bank 11 is associated.
11	Reserved	0	Reads return 0. Writes have no effect.
10-8	BANK10_MAP	0-7h	This 3-bit field allows the software to read the memory power domain number that bank 10 is associated.
7	Reserved	0	Reads return 0. Writes have no effect.
6-4	BANK9_MAP	0-7h	This 3-bit field allows the software to read the memory power domain number that bank 9 is associated.
3	Reserved	0	Reads return 0. Writes have no effect.
2-0	BANK8_MAP	0-7h	This 3-bit field allows the software to read the memory power domain number that bank 8 is associated.

Chapter 9

SPNU563A – March 2018

Programmable Built-In Self-Test (PBIST) Module

This chapter describes the programmable built-in self-test (PBIST) controller module used for testing the on-chip memories.

Topic **Page**

9.1 Overview

The PBIST (Programmable Built-In Self-Test) controller architecture provides a run-time-programmable memory BIST engine for varying levels of coverage across many embedded memory instances.

9.1.1 Features of PBIST

- Information regarding on-chip memories, memory groupings, memory background patterns and test algorithms stored in dedicated on-chip PBIST ROM
- Host processor interface to configure and start BIST of memories
- Supports testing of PBIST ROM itself as well
- Supports testing of each memory at its maximum access speed in application
- Implements intelligent clock gating to conserve power
- Execution of microcode from PBIST ROM supported for ROM clock speeds up to 100 MHz

9.1.2 PBIST vs. Application Software-Based Testing

The PBIST architecture consists of a small coprocessor with a dedicated instruction set targeted specifically toward testing memories. This coprocessor executes test routines stored in the PBIST ROM and runs them on multiple on-chip memory instances. The on-chip memory configuration information is also stored in the PBIST ROM.

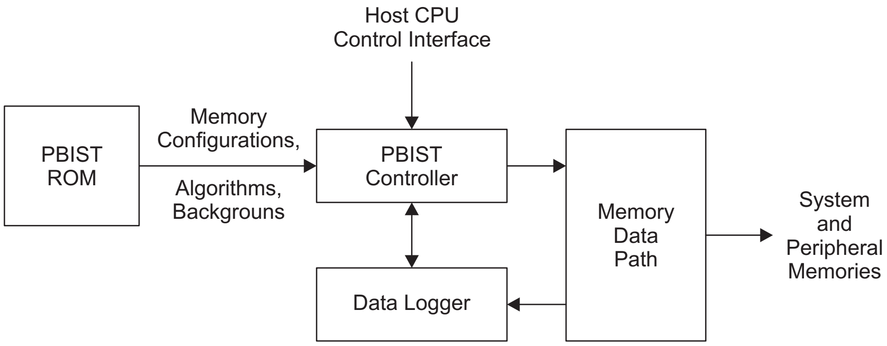
The PBIST Controller architecture offers significant advantages over tests running on the main Cortex-R5F processor (application software-based testing):

- Embedded CPUs have a long access path to memories outside the tightly-couple memory sub-system, while the PBIST controller has a dedicated path to the memories specifically for the self-test
- Embedded CPUs are designed for their targeted use and are often not easily programmed for memory test algorithms.
- The memory test algorithm code on embedded CPUs is typically significantly larger than that needed for PBIST.
- The embedded CPU is significantly larger than the PBIST controller.

9.1.3 PBIST Block Diagram

Figure 9-1 illustrates the basic PBIST blocks and its wrapper logic for the device.

Figure 9-1. PBIST Block Diagram



9.1.3.1 On-chip ROM

The on-chip ROM contains the information regarding the algorithms and memories to be tested.

9.1.3.2 Host Processor Interface to the PBIST Controller Registers

The Cortex-R5F CPU can select the algorithm and RAM groups for the memories' self-test from the on-chip ROM based on the application requirements. Once the self-test has executed, the CPU can query the PBIST controller registers to identify any memories that failed the self-test and to then take appropriate next steps as required by the application's author.

9.1.3.3 Memory Data Path

This is the read and write data path logic between different system and peripheral memories tightly coupled to the PBIST memory interface. The PBIST controller executes each selected algorithm on each valid memory group sequentially until all the algorithms are executed.

NOTE: Not all algorithms are designed to run on all RAM groups. If an algorithm is selected to run on an incompatible memory, this will result in a failure. Refer to Table 2-5 and Table 2-6 for RAM grouping and algorithm information.

9.2 RAM Grouping and Algorithm

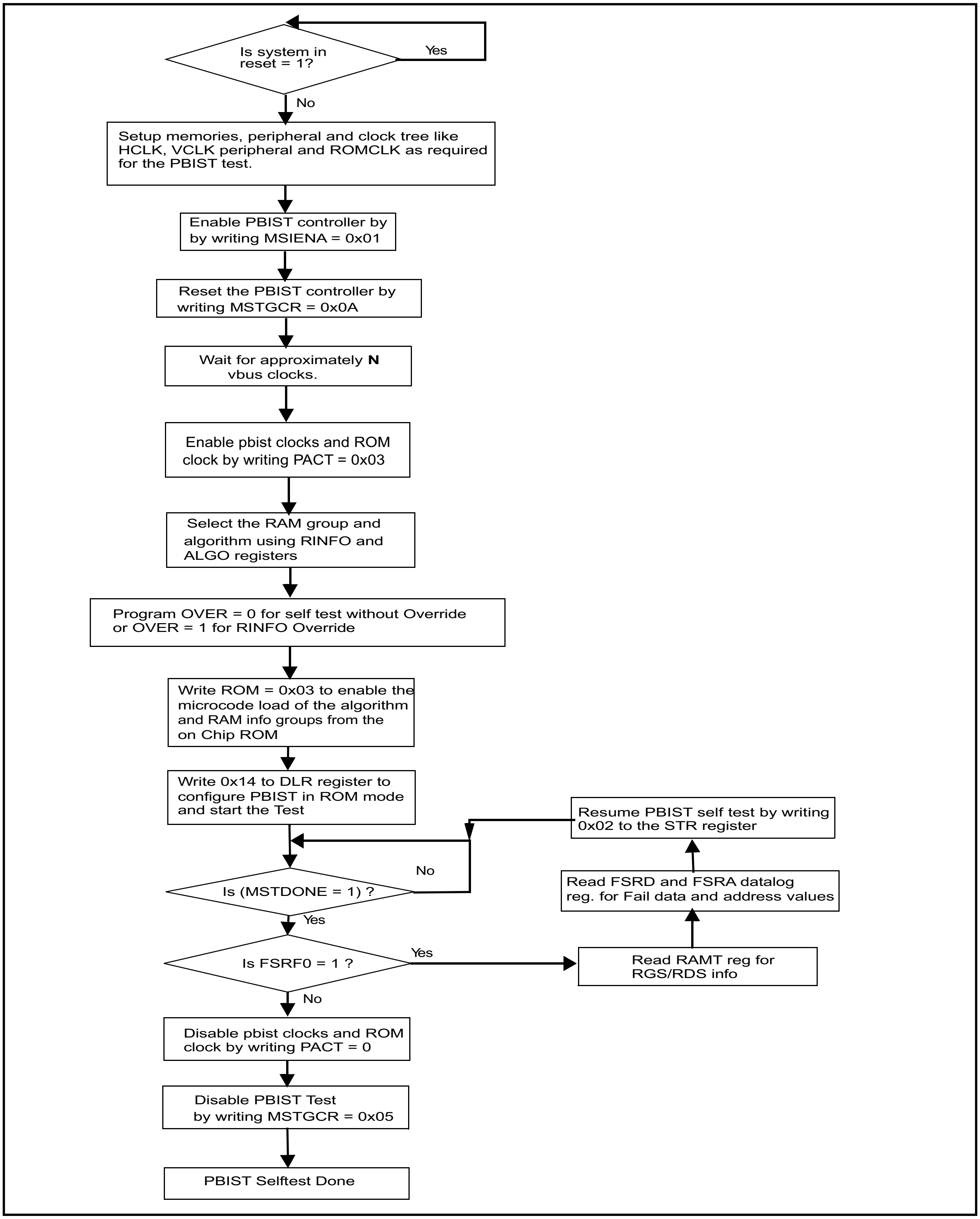
Table 2-5 gives the list of RAM groups and their types supported on the device. Table 2-6 maps the different algorithms supported in application mode for the RAM groups with the background patterns used for the particular algorithm.

NOTE: March13 is the most recommended algorithm for the memory self-test.

9.3 PBIST Flow

Figure 9-2 illustrates the memory self-test flow.

Figure 9-2. PBIST Memory Self-Test Flow Diagram


Is system in reset = 1?
Yes
No
Setup memories, peripheral and clock tree like HCLK, VCLK peripheral and ROMCLK as required for the PBIST test.
Enable PBIST controller by by writing MSIENA = 0x01
Reset the PBIST controller by writing MSTGCR = 0x0A
Wait for approximately **N** vbus clocks.
Enable pbist clocks and ROM clock by writing PACT = 0x03
Select the RAM group and algorithm using RINFO and ALGO registers
Program OVER = 0 for self test without Override or OVER = 1 for RINFO Override
Write ROM = 0x03 to enable the microcode load of the algorithm and RAM info groups from the on Chip ROM
Write 0x14 to DLR register to configure PBIST in ROM mode and start the Test
Resume PBIST self test by writing 0x02 to the STR register
No
Is (MSTDONE = 1) ?
Read FSRD and FSRA datalog reg. for Fail data and address values
Yes
Yes
Is FSRF0 = 1 ?
Read RAMT reg for RGS/RDS info
No
Disable pbist clocks and ROM clock by writing PACT = 0
Disable PBIST Test by writing MSTGCR = 0x05
PBIST Selftest Done

9.3.1 PBIST Sequence

Before starting the PBIST sequence, you should ensure that both the instruction cache and data cache are disabled. By default, PBIST will test all on-chip SRAMs including both the instruction and data cache memories. After reset, cache is disabled by default. If cache has been enabled, use the following code example to disable the cache.

```
MRC   p15, #0, R1, c1, c0, #0   ; Read System Control Register configuration data
BIC   R1, R1, #0x1 <<12         ; instruction cache disable
BIC   R1, R1, #0x1 <<2          ; data cache disable
DSB
MCR   p15, #0, R1, c1, c0, #0   ; disabled cache RAMs ISB
```

1. Configure the device clock sources and domains so that they are running at their target frequencies.
2. Program the GCLK1 to PBIST ROM clock ratio by configuring the ROM_DIV field (bits 9:8) of the MSTGCR register of the system module. This device supports a max PBIST ROM clock frequency of 82.5MHz.
3. Enable PBIST Controller by setting bit 1 of MSIENA register in system module.
4. Enable the PBIST self-test by writing a value of 0x0A to bits 3:0 of the MSTGCR in the system module.
5. Wait for N VBUS clock cycles based on the HCLK to PBIST ROM clock ratio:

   N = 16 when GCLK1:PBIST ROM clock is 1:1
   N = 32 when GCLK1:PBIST ROM clock is 1:2
   N = 64 when GCLK1:PBIST ROM clock is 1:4
   N = 64 when GCLK1:PBIST ROM clock is 1:8
6. Write 1h to PACT register to enable the PBIST internal clocks.
7. Program the ALGO register to decide which algorithm from the instruction ROM must be selected (the default value of ALGO register is all 1's, meaning all algorithms are selected). Similarly, program the RINFOL and RINFOU registers to indicate whether a particular RAM group in the instruction ROM would get executed or not.

   NOTE: In case of RAM Override (Override Register (OVER) = 00), the user should make sure that only the algorithms that run on similar RAMs are selected. If a single port algorithm is selected in ROM Algorithm Mask Register (ALGO), the RAM Info Mask Lower Register (RINFOL) and RAM Info Mask Upper Register (RINFOU) must select only the single port RAM's. The same applies for two port RAM's. Check Table 2-5 for information on the memory types.

8. Program OVER = 1h to run PBIST self-test without RAM override. Program OVER = 0 to run PBIST self-test with RAM Override.
9. Write a value of 3h to the ROM mask register should the microcode for the Algorithms as well as the RAM groups loaded from the on-chip PBIST ROM.
10. Write DLR (Data Logger register) with 14h to configure the PBIST run in ROM mode and to enable the configuration access. This starts the memory self-tests.
11. Wait for the PBIST self-test done by polling MSTDONE bit of MSTCGSTAT register in System Module.
12. Once self-test is completed, check the Fail Status register FSRF0.

    In case there is a failure (FSRF0 = 1h):
    a. Read RAMT register that indicates the RGS and RDS values of the failure RAM.
    b. Read FSRC0 and FSRC1 registers that contain the failure count.
    c. Read FSRA0 and FSRA1 registers that contain the address of first failure.
    d. Read FSRDL0 and FSRDL1 registers that contain the failure data.
    e. Write a value of 2h to the STR register to resume the test.

    In case there is no failure (FSRF0 = 0), the memory self-test is completed.
    a. Disable the PBIST internal clocks by writing a 0 to the PACT register.
    b. Disable the PBIST self-test by writing a value of 5h to bits 3:0 of the MSTGCR in the system module.

13. Repeat steps 2 through 9 for subsequent runs with different RAM group and algorithm configurations.
14. After required Memory tests are completed, Resume or Start the Normal Application software.

NOTE: The contents of the selected memory before the test will be completely lost. User software must take care of data backup if required. Typically the PBIST tests are carried out at the beginning of Application software.

NOTE: Memory test fail information is reported in terms of RGS:RDS and not RAM GROUP. Check Table 2-5 for information on the RGS:RDS information applicable to each memory being tested.

If cache memory is selected to be part of the PBIST test then the contents will become incoherent with respect to the level 2 memory after the PBIST test. The cache will need to be invalidated before cache can be enabled for use by the CPU. In addition, if you are using ECC error checking scheme in the cache, you must enable this by programming the CEC bits in the Auxiliary Control Register before invalidating the cache, to ensure that the correct error code bits are calculated when the cache is invalidated. For more information on the CEC bits in the Auxiliary Control Register, refer to the ARM® Cortex®-R5F Technical Reference Manual.

Use the following example code to invalidate cache and enable cache.

```
MRC p15, #0, R1, c1, c0, #1   ; Read auxiliary control register
BIC R1, R1, #0x1, <<5         ; bit is default set to disable ECC. Clearing bit 5
MCR p15, #0, R1, c1, c0, #1   ; enable ECC, generate abort on ECC errors, enable
                              ; hardware recovery
MRC p15, #0, R1, c1, c0, #0   ; Read system control register configuration data
ORR R1, R1, #0x1 <<12         ; instruction cache enable
ORR R1, R1, #0x1 <<2          ; data cache enable
DSB
MCR p15, #0, R0, c15, c5, #0  ; invalidate entire data cache
MCR p15, #0, R0, c7, c5, #0   ; invalidate entire instruction cache
MCR p15, #0, R1, c1, c0, #0   ; enable cache RAM
ISB                           ; You must issue an ISB instruction to flush the pipeline.
                              ; This ensures that all subsequent instruction fetches
                              ; see the effect of enabling the instruction cache
```

9.4 Memory Test Algorithms on the On-chip ROM

This section provides a brief description for some of the test algorithms used for memory self-test.

1. **March13N:**
   - March13N is the baseline test algorithm for SRAM testing. It provides the highest overall coverage. The other algorithms provide additional coverage of otherwise missed boundary conditions of the SRAM operation.
   - The concept behind the general march algorithm is to indicate:
     - The bit cell can be written and read as both a 1 and a 0.
     - The bits around the bit cell do not affect the bit cell.
   - The basic operation of the march is to initialize the array to a know pattern, then march a different pattern through the memory.
   - Type of faults detected by this algorithm:
     - Address decoder faults
     - Stuck-At faults
     - Coupled faults
     - State coupling faults
     - Parametric faults
     - Write recovery faults
     - Read/write logic faults

9.5 PBIST Control Registers

PBIST controller uses configuration registers for programming the algorithm and its execution. All the configuration registers are memory mapped for access by the CPU through the Peripheral Bus interface. The base address for the control registers is FFFF E400h.

NOTE: There is no watchdog functionality implemented in the PBIST controller. If a bad code is executed, the PBIST runs forever. The PBIST controller does not guard against this situation.

Registers are accessible only when the clock to the PBIST controller is active. The clock is activated by first writing 1h to the PACT register.

Table 9-1. PBIST Registers

Offset	Acronym	Register Description	Section
160h	RAMT	RAM Configuration Register	Section 9.5.1
164h	DLR	Datalogger Register	Section 9.5.2
180h	PACT	PBIST Activate/Clock Enable Register	Section 9.5.3
184h	PBISTID	PBIST ID Register	Section 9.5.4
188h	OVER	Override Register	Section 9.5.5
190h	FSRF0	Fail Status Fail Register 0	Section 9.5.6
198h	FSRC0	Fail Status Count Register 0	Section 9.5.7
19Ch	FSRC1	Fail Status Count Register 1	Section 9.5.7
1A0h	FSRA0	Fail Status Address Register 0	Section 9.5.8
1A4h	FSRA1	Fail Status Address Register 1	Section 9.5.8
1A8h	FSRDL0	Fail Status Data Register 0	Section 9.5.9
1B0h	FSRDL1	Fail Status Data Register 1	Section 9.5.9
1C0h	ROM	ROM Mask Register	Section 9.5.10
1C4h	ALGO	ROM Algorithm Mask Register	Section 9.5.11
1C8h	RINFOL	RAM Info Mask Lower Register	Section 9.5.12
1CCh	RINFOU	RAM Info Mask Upper Register	Section 9.5.13

9.5.1 RAM Configuration Register (RAMT)

This register is divided into following internal registers, none of which have a default value after reset. Figure 9-3 and Table 9-2 illustrate this register.

This register provides the information regarding the memory being currently tested. In case of a PBIST failure, the application can read this register to identify the RGS:RDS values for the memory that failed the self-test.

Figure 9-3. RAM Configuration Register (RAMT) [offset = 0160h]

31 24	23 16
RGS	RDS
R/W-X	R/W-X

15 8	7 6	5 2	1 0
DWR	SMS	PLS	RLS
R/W-X	R/W-X	R/W-X	R/W-X

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 9-2. RAM Configuration Register (RAMT) Field Descriptions

Bit	Field	Description
31-24	RGS	Ram Group Select. Refer to Table 2-5 for information on the RGS value for each memory.
23-16	RDS	Return Data Select. Refer to Table 2-5 for information on the RDS values for each memory. **Note:** In the current version of the PBIST, only 5 bits are used for RDS.
15-8	DWR	Data Width Register
7-6	SMS	Sense Margin Select Register
5-2	PLS	Pipeline Latency Select
1-0	RLS	RAM Latency Select

9.5.2 Datalogger Register (DLR)

This register puts the PBIST controller into the appropriate comparison modes for data logging. Figure 9-4 and Table 9-3 illustrate this register.

Figure 9-4. Datalogger Register (DLR) [offset = 0164h]

31–16
Reserved
R-0

15–5	4	3	2	1–0
Reserved	DLR4	Rsvd	DLR2	Reserved
R-0	R/W-0	R/W-1	R/W-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 9-3. Datalogger Register (DLR) Field Descriptions

Bit	Field	Value	Description
31-5	Reserved	0	Reads return 0. Do not change these bits from their default value.
4	DLR4		Config access: setting this bit allows the host processor to configure the PBIST controller registers.
3	Reserved	1	Do not change this bit from its default value of 1.
2	DLR2		ROM-based testing: setting this bit enables the PBIST controller to execute test algorithms that are stored in the PBIST ROM.
1-0	Reserved	00	Do not change these bits from their default value of 00.

- **DLR2: ROM-based testing mode**

Writing a 1 to this register starts the ROM-based testing. This register is used to initiate ROM-based testing from Config and ATE interfaces. Also, since a 1 in this bit position means the instruction ROM is used for memory testing, all the intermediate interrupts and PBIST done signal after each memory test are masked until all the selected algorithms in the ROM are executed for all RAM groups. However, a failure would stop the test and report the status immediately.

- **DLR4: Config access mode**

This mode, when set, indicates the CPU is being used to access PBIST.

9.5.3 PBIST Activate/Clock Enable Register (PACT)

This is the first register that needs to be programmed to activate the PBIST controller. Bit [0] is used for static clock gating, and unless a 1 is written to this bit, all the internal PBIST clocks are shut off. Figure 9-5 and Table 9-4 illustrate this register.

NOTE: This register must be programmed to 1h during application self-test.

Figure 9-5. PBIST Activate/ROM Clock Enable Register (PACT) [offset = 0180h]

31	16
Reserved	
R-0	

15	1	0
Reserved		PACT0
R-0		R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 9-4. PBIST Activate/ROM Clock Enable Register (PACT) Field Descriptions

Bit	Field	Value	Description
31-1	Reserved	0	Reads return 0. Writes have no effect.
0	PACT0		PBIST internal clocks enable.
		0	Disable PBIST internal clocks.
		1	Enable PBIST internal clocks.

- **PACT0**

This bit must be set to 1 to turn on the PBIST internal clocks. Setting this bit asserts an internal signal that is used as the clock gate enable. As long as this bit is 0, any access to the PBIST will not go through and the PBIST will remain in an almost zero-power mode.

9.5.4 PBIST ID Register

Functionality of the register is described in Figure 9-6 and Table 9-5.

Figure 9-6. PBIST ID Register [offset = 184h]

31	16
Reserved	
R-0	

15	8	7	0
Reserved		PBIST ID	
R-0		R/W-0	

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 9-5. PBIST ID Register Field Descriptions

Bit	Field	Value	Description
31-8	Reserved	0	Reads return 0. Writes have no effect.
7-0	PBIST ID		This is a unique ID assigned to each PBIST controller in a device with multiple PBIST controllers.

9.5.5 Override Register (OVER)

Functionality of the register is described in Figure 9-7 and Table 9-6.

Figure 9-7. Override Register (OVER) [offset = 0188h]

31	16
Reserved	
R-0	

15	3	2	1	0
Reserved		Reserved		OVER0
R-0		R-0		R/W-1

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 9-6. Override Register (OVER) Field Descriptions

Bit	Field	Value	Description
31-3	Reserved	0	Reads return 0. Writes have no effect.
2-1	Reserved	0	Reserved. This bit must not be changed from its default value of 0.
0	OVER0		RINFO Override Bit
		0	The RAM info registers RINFOL and RINFOU are used to select the memories for test.
		1	The memory information available from ROM will override the RAM selection from the RAM info registers RINFOL and RINFOU.

- **OVER0**

While doing ROM-based testing, each algorithm downloaded from the ROM has a memory mask associated with it that defines the applicable memory groups the algorithm will be run on. By default, this bit is set to 1, which means the memory mask that is downloaded from the ROM will overwrite the RAM info registers. The override bit can be reset by writing a 0 to it. In this case, the application can select the RAM groups to be tested by configuring the RAM info registers.

NOTE: When this override bit = 0, each algorithm selected in ALGO register will run on each RAM selected in RINFOL and RINFOU register. It must be ensured that:

1. Only the same type of memories (single port or two port) are selected, and
2. Only memories that are valid for all algorithms enabled via the ALGO register are selected.

If the above two requirements are not met, the memory self-test will fail.

9.5.6 Fail Status Fail Register (FSRF0)

This register indicates if a failure occurred during a memory self-test. Bit [0] gets set whenever a failure occurs. Figure 9-8 and Table 9-7 illustrate the FSRF0 register.

Figure 9-8. Fail Status Fail Register 0 (FSRF0) [offset = 0190h]

31	16
Reserved	
R-0	

15	1	0
Reserved		FSRF0
R-0		R-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 9-7. Fail Status Fail Register 0 (FSRF0) Field Descriptions

Bit	Field	Value	Description
31-1	Reserved	0	Reads return 0. Writes have no effect.
0	FSRF0		Fail Status 0. This bit would be cleared by reset of the module using MSTGCR register in system module.
		0	No failure occurred.
		1	Failure occurred on port 0.

9.5.7 Fail Status Count Registers (FSRC0 and FSRC1)

These registers keep count of the number of failures observed during the memory self-test. The PBIST controller stops executing the memory self-test whenever a failure occurs in any memory instance for any of the test algorithms. The value in FSRC0 / FSRC1 gets incremented by one whenever a failure occurs and gets decremented by one when the failure is processed. FSRC0 is for Port 0 and FSRC1 is for Port 1. Figure 9-9 and Table 9-8 illustrate the FSRC0 register, while Figure 9-10 and Table 9-9 illustrate the FSRC1 register.

Figure 9-9. Fail Status Count 0 Register (FSRC0) [offset = 0198h]

31	16
Reserved	
R-0	

15	8	7	0
Reserved		FSRC0	
R-0		R-0	

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 9-8. Fail Status Count 0 Register (FSRC0) Field Descriptions

Bit	Field	Value	Description
31-8	Reserved	0	Reads return 0. Writes have no effect.
7-0	FSRC0		Fail Status Count 0. Indicates the number of failures on port 0.

Figure 9-10. Fail Status Count Register 1 (FSRC1) [offset = 019Ch]

31	16
Reserved	
R-0	

15	8	7	0
Reserved		FSRC1	
R-0		R-0	

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 9-9. Fail Status Count Register 1 (FSRC1) Field Descriptions

Bit	Field	Value	Description
31-8	Reserved	0	Reads return 0. Writes have no effect.
7-0	FSRC1		Fail Status Count 1. Indicates the number of failures on port 1.

9.5.8 Fail Status Address Registers (FSRA0 and FSRA1)

These registers capture the memory address of the first failure on port 0 and port 1, respectively. Figure 9-11 and Table 9-10 illustrate the FSRA0 register, while Figure 9-12 and Table 9-11 illustrate the FSRA1 register.

Figure 9-11. Fail Status Address Register 0 (FSRA0) [offset = 01A0h]

31	16
Reserved	
R-0	

15	0
FSRA0	
R-0	

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 9-10. Fail Status Address Register 0 (FSRA0) Field Descriptions

Bit	Field	Value	Description
31-16	Reserved	0	Reads return 0. Writes have no effect.
15-0	FSRA0		Fail Status Address 0. Contains the address of the first failure.

Figure 9-12. Fail Status Address Register 1 (FSRA1) [offset = 01A4h]

31	16
Reserved	
R-0	

15	0
FSRA1	
R-0	

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 9-11. Fail Status Address Register 1 (FSRA1) Field Descriptions

Bit	Field	Value	Description
31-16	Reserved	0	Reads return 0. Writes have no effect.
15-0	FSRA1		Fail Status Address 1. Contains the address of the first failure.

9.5.9 Fail Status Data Registers (FSRDL0 and FSRDL1)

These registers are used to capture the failure data in case of a memory self-test failure. FSRDL0 corresponds to Port 0, while FSRDL1 corresponds to Port 1. Figure 9-13 and Table 9-12 illustrate the FSRDL0 register, while Figure 9-14 and Table 9-13 illustrate the FSRDL1 register.

Figure 9-13. Fail Status Data Register 0 (FSRDL0) [offset = 01A8h]

31 16
FSRDL0
R-AAAAh

15 0
FSRDL0
R-AAAAh

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 9-12. Fail Status Data Register 0 (FSRDL0) Field Descriptions

Bit	Field	Description
31-0	FSRDL0	Failure data on port 0.

Figure 9-14. Fail Status Data Register 1 (FSRDL1) [offset = 01B0h]

31 16
FSRDL1
R-AAAAh

15 0
FSRDL1
R-AAAAh

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 9-13. Fail Status Data Register 1 (FSRDL1) Field Descriptions

Bit	Field	Description
31-0	FSRDL1	Failure data on port 1.

9.5.10 ROM Mask Register (ROM)

This two-bit register sets appropriate ROM access modes for the PBIST controller. The default value is 11b. This register is illustrated in Figure 9-15. It can be programmed according to Table 9-14.

Figure 9-15. ROM Mask Register (ROM) [offset = 01C0h]

31–16
Reserved
R-0

15–2	1–0
Reserved	ROM
R-0	R/W-3h

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 9-14. ROM Mask Register (ROM) Field Descriptions

Bit	Field	Value	Description
31-2	Reserved	0	Reads return 0. Writes have no effect.
1-0	ROM		ROM Mask
		0	No information is used from ROM.
		1h	Only RAM Group information from ROM.
		2h	Only Algorithm information from ROM.
		3h	Both Algorithm and RAM Group information from ROM. This option should be selected for application self-test.

9.5.11 ROM Algorithm Mask Register (ALGO)

This register is used to indicate the algorithm(s) to be used for the memory self-test routine. Each bit corresponds to a specific algorithm. For example, bit [0] controls whether algorithm 1 is enabled or not. Figure 9-16 and Table 9-15 illustrate this register.

Figure 9-16. ROM Algorithm Mask Register (ALGO) [offset = 01C4h]

31 24	23 16
ALGO3	ALGO2
R/W-FFh	R/W-FFh

15 8	7 0
ALGO1	ALGO0
R/W-FFh	R/W-FFh

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 9-15. Algorithm Mask Register (ALGO) Field Descriptions

Bit	Field	Value	Description
31		0	Algorithm 32 is not selected.
		1	Selects algorithm 32 for PBIST run.
30		0	Algorithm 31 is not selected.
		1	Selects algorithm 31 for PBIST run.
:			:
0		0	Algorithm 1 is not selected.
		1	Selects algorithm 1 for PBIST run.
31-0		0	None of the algorithms are selected.

NOTE: Please refer to Table 2-6 for available algorithms and the memories on which each algorithm can be run.

9.5.12 RAM Info Mask Lower Register (RINFOL)

This register is used to select the RAM groups 1 to 32 to run the algorithms selected in the ALGO register. For an algorithm to be executed on a particular RAM group, the corresponding bit in this register must be set to 1. The default value of this register is all 1s, which means all the RAM Groups are selected. Figure 9-17 and Table 9-16 illustrate this register.

The information from this register is used only when bit 0 in OVER register is not set.

Figure 9-17. RAM Info Mask Lower Register (RINFOL) [offset = 01C8h]

31 24	23 16
RINFOL3	RINFOL2
R/W-FFh	R/W-FFh

15 8	7 0
RINFOL1	RINFOL0
R/W-FFh	R/W-FFh

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 9-16. RAM Info Mask Lower Register (RINFOL) Field Descriptions

Bit	Field	Value	Description
31		0	RAM Group 32 is not selected.
		1	Selects group 32 for PBIST run.
30		0	RAM Group 31 is not selected.
		1	Selects RAM group 31 for PBIST run.
:			:
0		0	RAM Group 1 is not selected.
		1	Selects RAM Group 1 for PBIST run.
31-0		0	None of the RAM Groups 1 to 32 are selected.

NOTE: Please refer to Table 2-5 for RAM info groups.

9.5.13 RAM Info Mask Upper Register (RINFOU)

This register is used to select the RAM groups 33 to 64 to run the algorithms selected in the ALGO register. For an algorithm to be executed on a particular RAM group, the corresponding bit in this register should be set to 1. The default value of this register is all 1s, which means all the RAM Info Groups would be selected. Figure 9-18 and Table 9-17 illustrate this register.

Figure 9-18. RAM Info Mask Upper Register (RINFOU) [offset = 01CCh]

31 24	23 16
RINFOU3	RINFOU2
R/W-FFh	R/W-FFh

15 8	7 0
RINFOU1	RINFOU0
R/W-FFh	R/W-FFh

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 9-17. RAM Info Mask Upper Register (RINFOU) Field Descriptions

Bit	Field	Value	Description
31		0	RAM Group 64 is not selected.
		1	Selects group 64 for PBIST run.
30		0	RAM Group 63 is not selected.
		1	Selects RAM group 63 for PBIST run.
:			:
0		0	RAM Group 33 is not selected.
		1	Selects RAM Group 33 for PBIST run.
31-0		0	None of RAM Groups 33 to 64 are selected.

9.6 PBIST Configuration Example

The following examples assume that the PLL is locked and selected as clock source with GCLK1 = 300 MHz and VCLK = 75 MHz.

9.6.1 Example 1 : Configuration of PBIST Controller to Run Self-Test on DCAN1 RAM

This example explains the configurations for running March13 algorithm on DCAN1.

1. Program the GCLK1 to PBIST ROM clock ratio to 1:4 in System Module.
   MSTGCR[9:8] = 2
2. Enable PBIST Controller in System Module.
   MSIENA[31:0] = 0x00000001
3. Enable the PBIST self-test in System Module.
   MSTGCR[3:0] = 0xA
4. Wait for at least 64 VCLK cycles in a software loop.
5. Enable the PBIST internal clocks.
   PACT = 0x1
6. Disable RAM Override. This will make the PBIST controller use the information provided by the application in the RINFOx and ALGO registers for the memory self-test.
   OVER = 0x0
7. Select the Algorithm (refer to Table 2-6).
   ALGO = 0x00000004 (Algo 3 = March13N for two-port DCAN1 RAM)
8. Program the RAM group Info to select DCAN1 (DCAN1 RAM is Group 3, refer to Table 2-5).
   RINFOL = 0x00000004 (select RAM Group 3)
   RINFOU = 0x00000000 (since we are testing only DCAN1)
9. Select both Algorithm and RAM information from on-chip PBIST ROM.
   ROM = 0x3
10. Configure PBIST to run in ROM Mode and start PBIST run.
    DLR = 0x14
11. Wait for PBIST test to complete by polling MSTDONE bit in System Module.
    while (MSTDONE !=1)
12. Once self-test is completed, check the Fail Status register FSRF0.
    In case there is a failure (FSRF0 = 1):
    a. Read RAMT register that indicates the RGS and RDS values of the failure RAM.
    b. Read FSRC0 and FSRC1 registers that contain the failure count.
    c. Read FSRA0 and FSRA1 registers that contain the address of first failure.
    d. Read FSRDL0 and FSRDL1 registers that contain the failure data.
    e. Resume the Test if required using Program Control register (offset = 0x16C) STR = 2.

    In case there is no failure (FSRF0 = 0), the memory self-test is completed.
    a. Disable the PBIST internal clocks.
       PACT = 0
    b. Disable the PBIST self-test.
       MSTGCR[3:0] = 0x5

9.6.2 Example 2 : Configuration of PBIST Controller to Run Self-Test on ALL RAM Groups

This example explains the configurations for running March13 algorithm on all RAM groups defined in the PBIST ROM.

1. Program the GCLK1 to PBIST ROM clock ratio to 1:4 in System Module.
   MSTGCR[9:8] = 2
2. Enable PBIST Controller in System Module.
   MSIENA[31:0] = 0x00000001
3. Enable the PBIST self-test in System Module.
   MSTGCR[3:0] = 0xA
4. Wait for at least 64 VCLK cycles in a software loop.
5. Enable the PBIST internal clocks.
   PACT = 0x1
6. Enable RAM Override.
   OVER = 0x1
7. Select the Algorithms to be run (refer to Table 2-6).
   ALGO = 0x0000000C (select March13N for single-port and two-port RAMs)
8. Select both Algorithm and RAM information from on-chip PBIST ROM.
   ROM = 0x3
9. Configure PBIST to run in ROM Mode and kickoff PBIST test.
   DLR = 0x14
10. Wait for PBIST test to complete by polling MSTDONE bit in System Module.
    while (MSTDONE !=1)
11. Once self-test is completed, check the Fail Status register FSRF0.
    In case there is a failure (FSRF0 = 1):
    a. Read RAMT register that indicates the RGS and RDS values of the failure RAM.
    b. Read FSRC0 and FSRC1 registers that contain the failure count.
    c. Read FSRA0 and FSRA1 registers that contain the address of first failure.
    d. Read FSRDL0 and FSRDL1 registers that contain the failure data.
    e. Resume the Test if required using Program Control register (offset = 0x16C) STR = 2.

    In case there is no failure (FSRF0 = 0), the Memory self-test is completed.
    a. Disable the PBIST internal clocks.
       PACT = 0
    b. Disable the PBIST self-test.
       MSTGCR[3:0] = 0x5

Self-Test Controller (STC) Module

This chapter describes the basics and configuration of the on chip self-test controller (STC) modules.

Topic **Page**

10.1 General Description

The self-test controller (STC) module is used to test the ARM CPU core and other complex digital IPs using the 'Deterministic Logic Built-in Self-Test' (LBIST) controller as the test engine. To achieve better coverage for the self-test of complex cores like Cortex-R5F, on-chip logic BIST is the preferred solution over software based self-test.

There are two STC modules implemented on this device. STC1 for redundant CPUs and their µSCU block. STC2 for the two nHET modules. The STC module provides the capability to test redundant IPs in parallel or individually.

10.1.1 Self-Test Controller Features

The self-test controller has the following features:

- Capable of running the complete test as well as running a single or multiple test sets (intervals) at a time.
  - Ability to continue from the last executed interval as well as the ability to restart from the beginning (first interval).
- Support of two logical segments. Figure 10-1 shows the implementation with multiple segments. Each interval can be mapped to a logical segment. A segment identifier corresponding to each interval is stored in the self-test ROM.
  - Segment 0: Segment 0 has the additional capability to test redundant logic or cores in one of the following modes:
    - Parallel Mode: Redundant logic cores are tested in parallel with the same patterns but have a dedicated signature generator. This is used in the safety critical redundant logic that runs in lock-step. Figure 10-2 and Figure 10-3 show this configuration for STC1 and STC2.
    - Split Mode: Each redundant logic is tested individually. Figure 10-4 and Figure 10-5 show this configuration for STC1.
  - All redundant cores (or IPs) within a segment have their own dedicated DBIST controllers.
  - Other segment (segment 1) can test only a single logic segment during the self-test run.
  - Ability to select segment for which the first interval is selected for run.
- Complete isolation of the self-tested core from the rest of the system during the self-test run
  - The self-tested CPU core master bus transaction signals are configured to be in idle mode during the self-test run
  - Any master access to the CPU core under self-test (example: DMA access to CPU TCM) will be held until the completion of the self-test
- Ability to capture the failure segment and interval number
- Timeout counter for the self-test run as a fail-safe feature
- Able to read the MISR data (shifted from LBIST controller) of the last executed interval of the self-test run for debugging purposes
- STCCLK determines the self-test execution speed, STC clock divider (STCCLKDIV) register is used to divide one of the system clocks to generate STCCLK. The divider can be configured per segment. For STC1, GCLK1 is divided down; for STC2, VCLK2 is divided down to generate STCCLK.
- Low-frequency shift. Programmable clock divider register inside STC to reduce the shift frequency in order to reduce the shift power.

10.1.2 Terminology

Interval: An interval corresponds to a test set that is the basic test unit for the STC module

Segment: A self-test segment corresponds to a portion of the unique/discrete safety critical logic which can be tested in isolation from the rest of the system by the self-test controller and DBIST logic. A self-test segment may correspond to a logic like CPU core (for example, Cortex-R5F) or an IP (for example, µSCU or nHET) or a sub-system.

The assignment of segments to digital logic is device dependent.

NOTE: All segments need to run sequentially during the self-test run. It is not recommended to switch from one segment to another before the self-test for the current segment is completed. The segment intervals in the STC ROM are organized sequentially.

10.1.3 STC Block Diagram

STC module provides an interface to the LBIST controller implemented on the CPUs and the nHET modules. There are two separate STC modules implemented: one for redundant Cortex-R5F CPUs and µSCU and another one for the nHET modules. Each STC module comprises of the same basic blocks and has same features and functionality.

The STC module is composed of following blocks of logic:

- ROM Interface
- FSM and Sequence Control
- Register Block
- Peripheral Bus Interface (VBUSP Interface)

10.1.3.1 ROM Interface

This block handles the ROM address and control signal generation to read the self-test microcode from the ROM. The test microcode and golden signature value for each interval are stored in ROM.

10.1.3.1.1 FSM and Sequence Control

This block generates control signal and data to the LBIST controller based on the seed, test_type and scan chain depth.

10.1.3.1.2 Clock Control

The clock controller sub-block handles the internal clock selection and generation for the ROM, LBIST controller and logic under test.

The clock control ratio can be programmed in STC module by programming STCCLKDIV register.

10.1.3.2 Register Block

This block handles the control of the self-test controller. This block contains various configuration and status registers that provide the result of a self-test run. These registers are memory mapped and accessible through the Peripheral Bus (VBUSP) Interface. This block controls the reseeding (reloading the existing seed of the PRPG) in the LBIST controller.

10.1.3.3 Peripheral Bus (VBUSP) Interface

STC control registers are accessed through Peripheral Bus (VBUSP) Interface. During application programming, configuration registers are programmed through the Peripheral Bus Interface to enable and run the self-test controller.

Figure 10-1. Block Diagram for STC With Multiple Segments

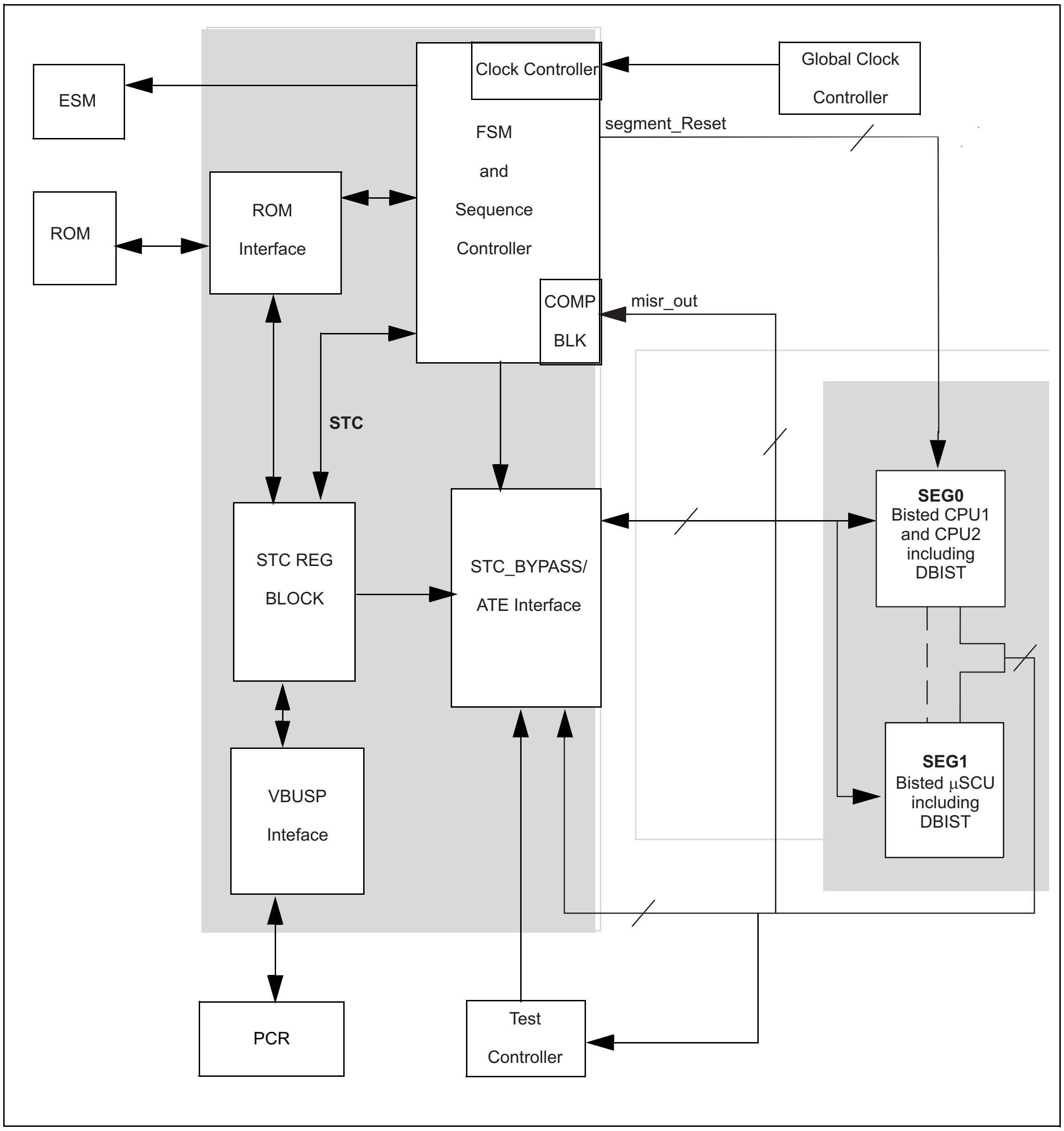


Figure 10-2. STC1 - Segment 0 Redundant Core Architecture With CCM-R5F (Parallel Mode)

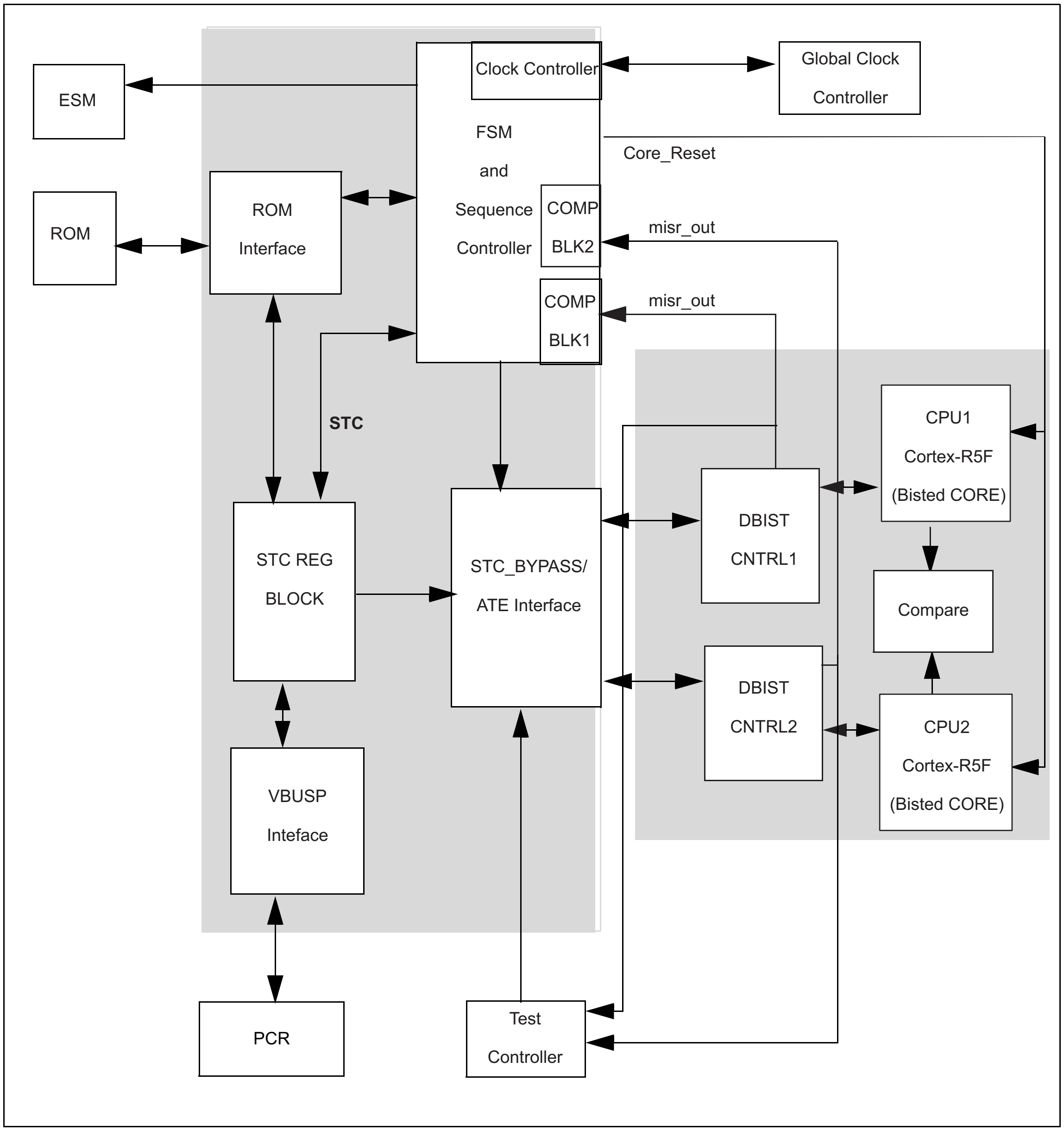


Figure 10-3. STC2 - Segment 0 Redundant Architecture (Parallel Mode)

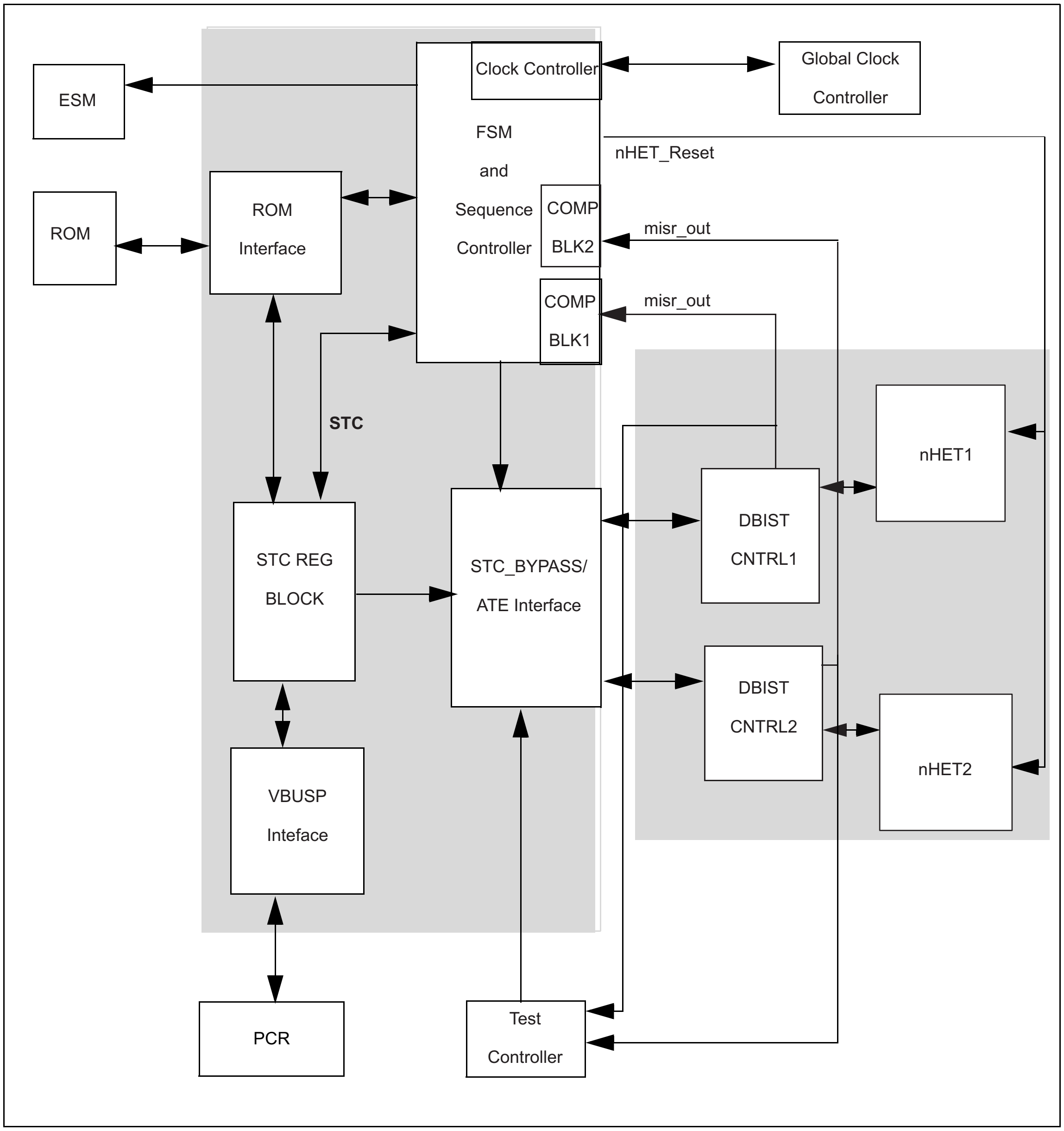
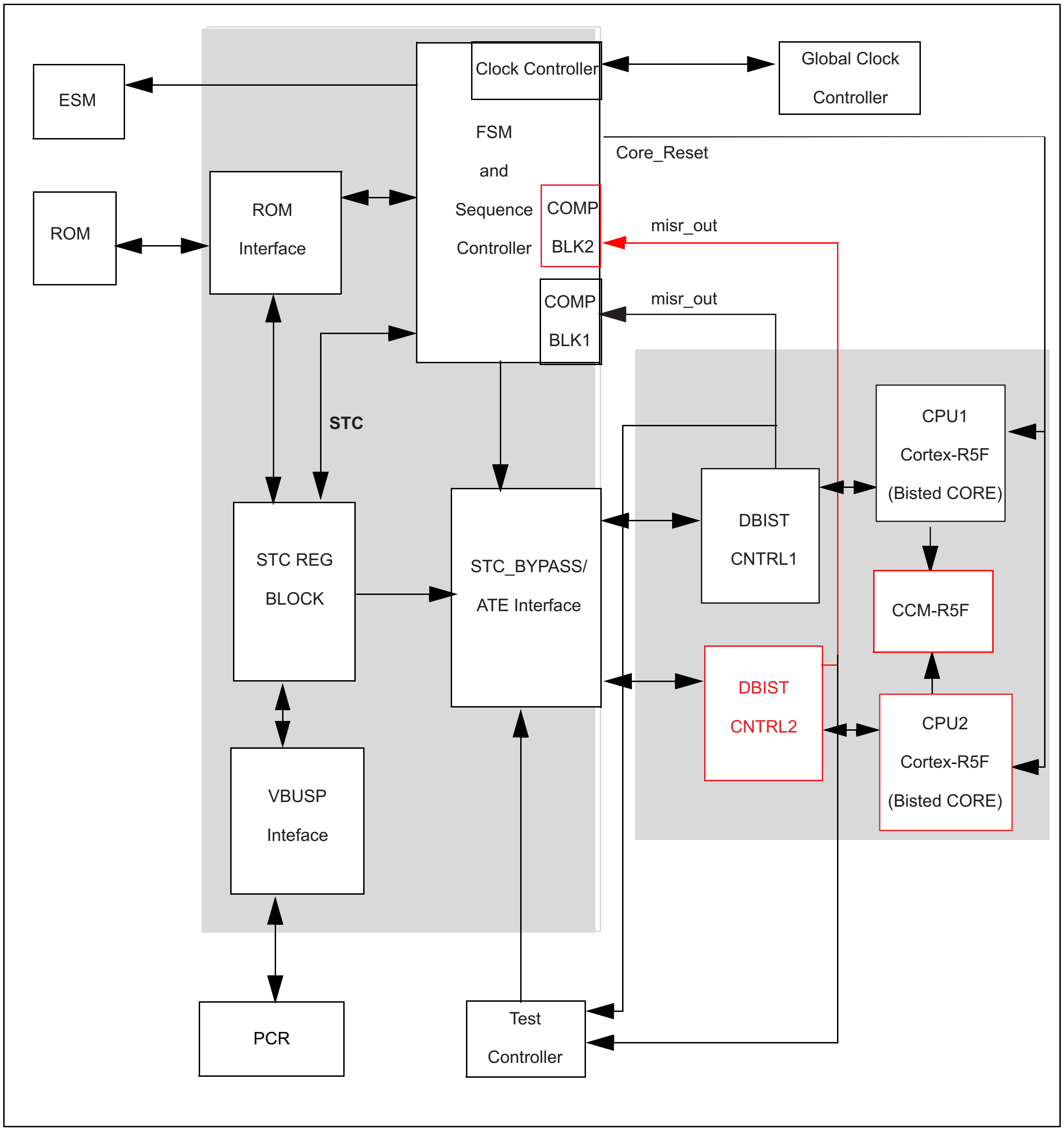
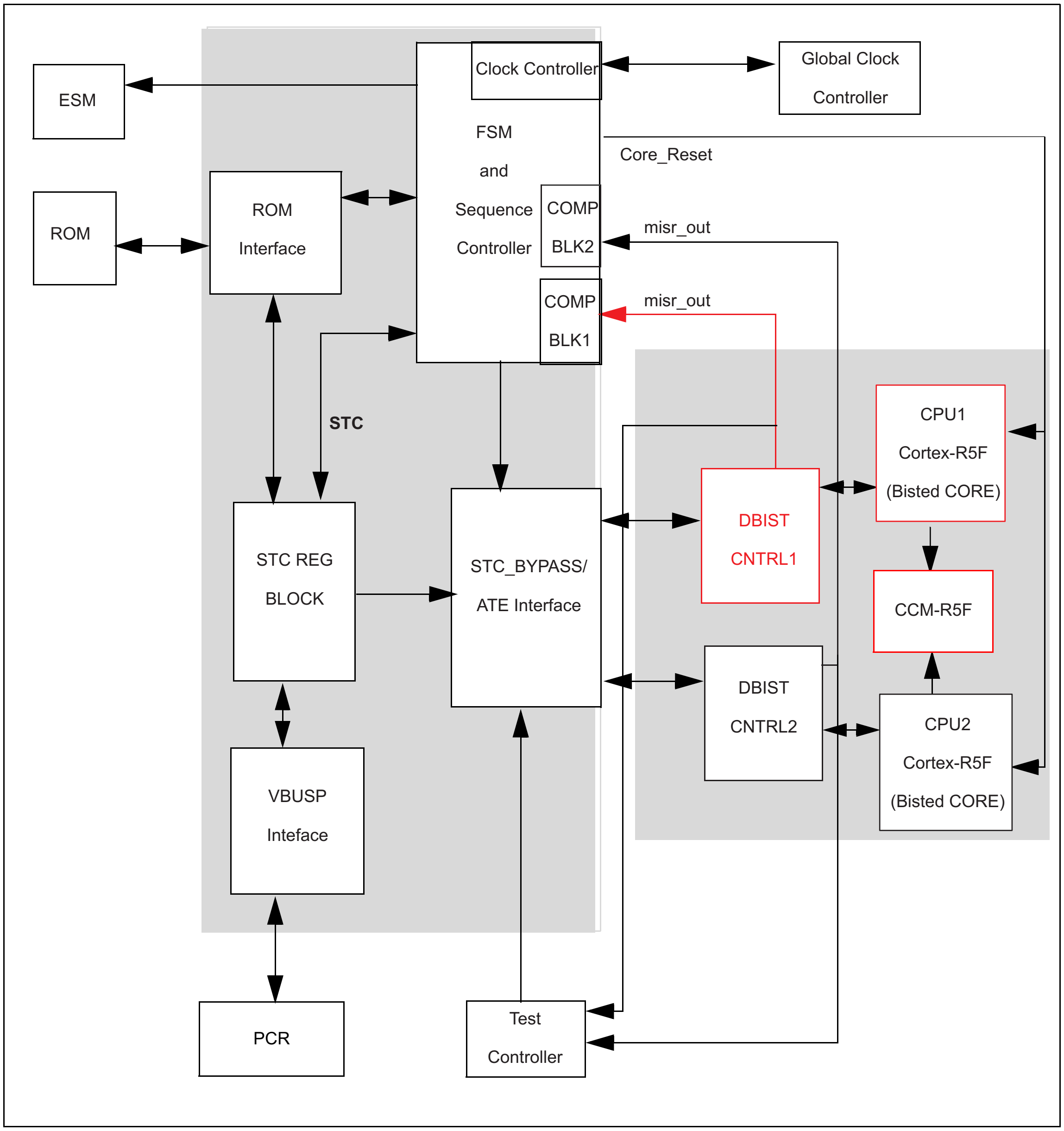


Figure 10-4. STC1 - Segment 0 Redundant Core Architecture With Only CPU1 Selected



Modules highlighted in red will not be enabled for test while testing CORE1 only in a redundant system.

Figure 10-5. STC1 - Segment 0 Redundant Core Architecture With Only CPU2 Selected



Modules highlighted in red will not be enabled for test while testing CORE2 only in a redundant system.

10.2 STC Module Assignments

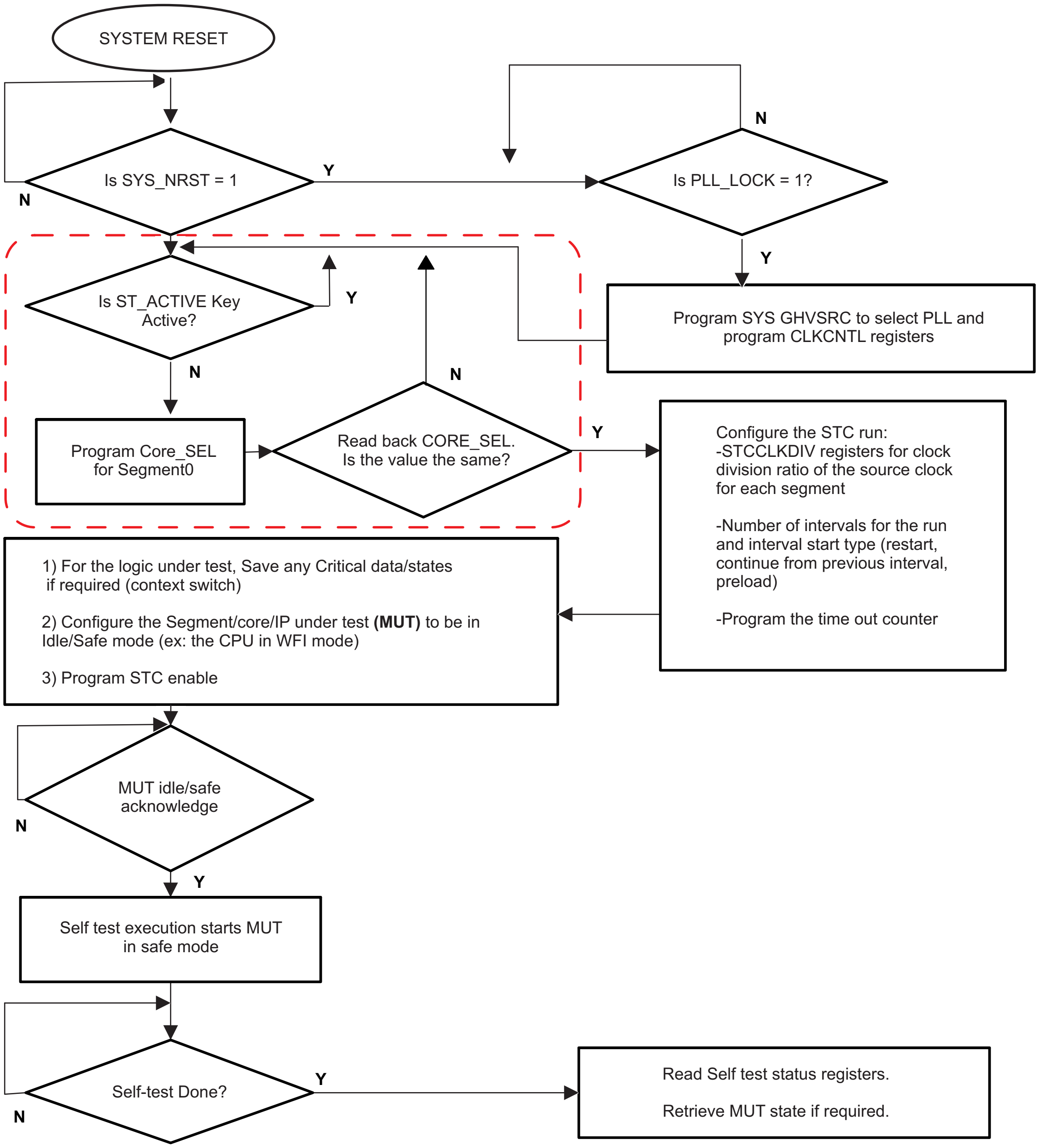
There are two instances of STC modules available on this device, see Table 10-1. STC1 is used for running self-test on the redundant CPUs and µSCU. STC2 is used for running self-test on the two nHET modules. The two instances are independent of each other.

Table 10-1. STC Module Assignments

Module	Segments	Targeted IP	Number of Intervals	STCCLK Derived From	Note
STC1	Segment 0	CPU1 and CPU2	125	GCLK1	Segment 0 allows CPU1 and CPU2 to be tested in parallel or individually.
	Segment 1	µSCU (ACP Block)	3	GCLK1	None
STC2	Segment 0	nHET1 and nHET2	57	VCLK2	Segment 0 allows nHET1 and nHET2 to be tested in parallel or individually.

10.3 STC Programmers Flow

Figure 10-6. STC Programmers Flow Chart


SYSTEM RESET
Is SYS_NRST = 1
Y
N
Is PLL_LOCK = 1?
N
Y
Program SYS GHVSRC to select PLL and program CLKCNTL registers
Is ST_ACTIVE Key Active?
Y
N
Program Core_SEL for Segment0
Read back CORE_SEL. Is the value the same?
N
Y
Configure the STC run:
-STCCLKDIV registers for clock division ratio of the source clock for each segment
-Number of intervals for the run and interval start type (restart, continue from previous interval, preload)
-Program the time out counter
1) For the logic under test, Save any Critical data/states if required (context switch)
2) Configure the Segment/core/IP under test (MUT) to be in Idle/Safe mode (ex: the CPU in WFI mode)
3) Program STC enable
MUT idle/safe acknowledge
N
Y
Self test execution starts MUT in safe mode
Self-test Done?
Y
N
Read Self test status registers.
Retrieve MUT state if required.

The steps shown in red can be bypassed for self-test with single core only.

10.4 Application Self-Test Flow

This section describes the STC module configuration and the application self-test flow that you should follow for successful execution. The following two configurations must be part of the STC initialization code:

- STC clock rate configuration, STC clock divider (STCCLKDIV) register is used to divide system clock to generate STCCLK for each segment.
- Clear SYSESR register before triggering an STC test.

10.4.1 STC Module Configuration

- Configure the test interval count using STCGCR0[31:16] register. STC1, segment 0 supports a maximum of 125 intervals, STC1 segment 1 supports a maximum of 3 and STC2 supports a maximum of 57 intervals. The intervals within each group can be ran individually or sequentially at one time. If the test intervals are run individually, the user software can specify to the self-test controller whether to continue the run from the next interval or to restart from interval 0 using bit STCGCR0[0]. This bit gets reset after the completion of the self-test run.
- Configure self-test run timeout counter preload register STCTPR. This register contains the total number of VBUS clock cycles it will take before a self-test timeout error (TO_ERR) will be triggered after the initiation of the self-test run.
- Configure Segment 0 for parallel or serial execution for each of the 2 elements to be tested (primary and redundant logic).
- Enable self-test by writing the enable key to STCGCR1 register.

10.4.2 Context Saving - CPU

STC generates a CPU reset after completion of each test regardless of pass or fail. You can run the STC test during startup or can divide STC into subsets of 1 or more intervals and executed during application run time.

The STC test is a destructive test such that content within the element being tested may need to be preserved.

If STC is run only on startup, the user software need not save the CPU contents since the reset at the completion of the test will be followed by normal device initializations/startup configuration. During startup, the user code should check the STCGSTAT register for the self-test status before going to the application software.

If STC is divided into intervals and ran during application run time, the user software must save the CPU contents and reload them after each CPU reset caused by the completion of the STC test interval. The check for STC status should bypass the STC run if the reset is caused by a completed test execution. The STCGSTAT register should be checked for the self-test status before returning to the application software.

Following are some of the registers that are required to be backed up before and restored after self-test:

1. CPU core registers: all modes R0-R15, PC, CPSR
2. CP15 System Control Coprocessor registers: MPU control and configuration registers, Auxiliary Control Register used to Enable ECC, Fault Status Register
3. CP13 Coprocessor Registers: FPU configuration registers, General Purpose Registers
4. Hardware Break Point and watch point registers: BVR, BSR, WVR, WSR

For more information on the CPU reset, refer to the ARM® Cortex®-R5F Technical Reference Manual.

NOTE: Check all reset source flags in the SYSESR register after a CPU BIST execution. If a flag in addition to CPU reset is set, clear the CPU reset flag and service the other reset sources accordingly.

10.4.3 Entering CPU Idle Mode

After enabling the STC test by writing the STC enable key, the test is triggered only after the CPU is taken to idle mode by executing the CPU Idle Instruction **asm(" WFI").**

10.4.4 Entering nHET Idle Mode

After enabling the STC test by writing the STC enable key, the test is triggered only after the nHET module is put in reset state by writing to bit 0 the HETGCR Global Configuration Register in the nHET module.

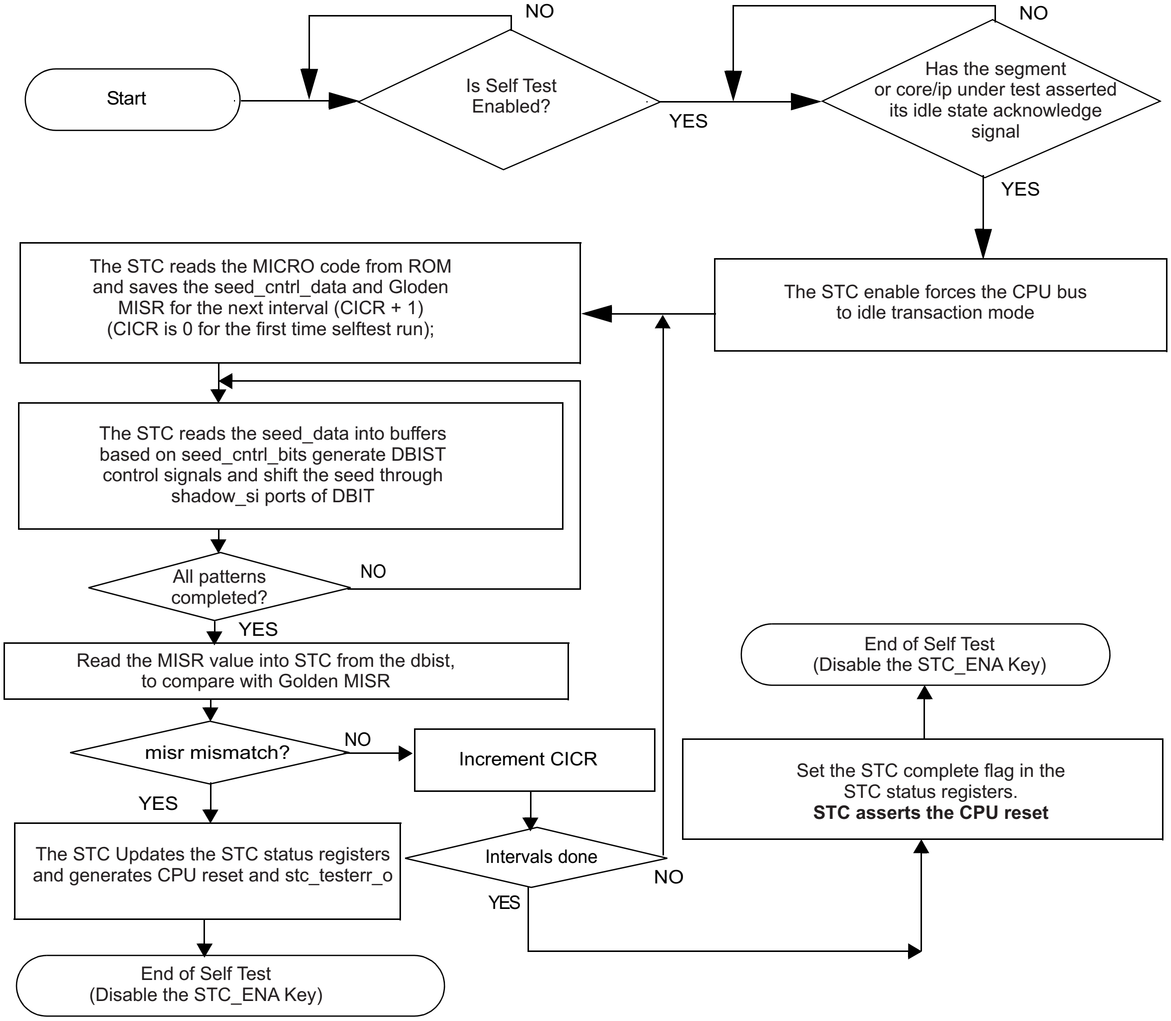
10.4.5 Self-Test Completion and Error Generation

At the end of each interval, the 128 bit MISR value (reflected in registers CPUx_CURMISR[3:0]) from the DBIST controller is shifted into the STC. This is compared with the golden MISR value stored in the ROM.

At the end of a CPU self-test, the STC controller updates the status flags in the Global Status Register (STCGSTAT) and resets the CPU. In case of a MISR mismatch or a test timeout, an error is generated through the ESM module. TEST_ERR signal is asserted when an MISR miscompare occurs during the self-test. A TO_ERR is asserted when a timeout occurs during the self-test, meaning the test could not complete within the time specified in the timeout counter preload register STCTPR. However, at the device level, these two errors are combined and mapped to a single ESM channel. To identify which error occurred, user software must check the global status register (STCGSTAT) and fail status register STCFSTAT in the ESM interrupt service routine.

Figure 10-7 illustrates the self-test hardware execution flow chart, based on the assumption that the device has gone through startup, necessary clocks initialized and SYSESR register bits cleared.

Figure 10-7. Self-Test Hardware Execution Flow Chart


Start
Is Self Test Enabled?
NO
YES
Has the segment or core/ip under test asserted its idle state acknowledge signal
NO
YES
The STC enable forces the CPU bus to idle transaction mode
The STC reads the MICRO code from ROM and saves the seed_cntrl_data and Gloden MISR for the next interval (CICR + 1) (CICR is 0 for the first time selftest run);
The STC reads the seed_data into buffers based on seed_cntrl_bits generate DBIST control signals and shift the seed through shadow_si ports of DBIT
All patterns completed?
NO
YES
Read the MISR value into STC from the dbist, to compare with Golden MISR
misr mismatch?
NO
Increment CICR
YES
The STC Updates the STC status registers and generates CPU reset and stc_testerr_o
Intervals done
NO
YES
End of Self Test (Disable the STC_ENA Key)
Set the STC complete flag in the STC status registers.
STC asserts the CPU reset
End of Self Test (Disable the STC_ENA Key)

10.5 STC1 Segment 0 (CPU) Test Coverage and Duration

The test coverage and number of test execution cycles (STCCLK) for each test interval are shown in Table 10-2.

Table 10-2. STC1 Segment 0 Test Coverage and Duration

Intervals	Test Coverage (%)	Test Time (Cycles)
0	0	0
1	56.85	1629
2	64.19	3258
3	68.76	4887
4	71.99	6516
5	75.00	8145
6	76.61	9774
7	78.08	11403
8	79.20	13032
9	80.18	14661
10	81.03	16290
11	81.90	17919
12	82.58	19548
13	83.24	21177
14	83.73	22806
15	84.15	24435
16	84.52	26064
17	84.90	27693
18	85.26	29322
19	85.68	30951
20	86.05	32580
21	86.40	34209
22	86.68	35838
23	86.94	37467
24	87.21	39096
25	87.48	40725
26	87.74	42354
27	87.98	43983
28	88.18	45612
29	88.38	47241
30	88.56	48870
31	88.75	50499
32	88.93	52128
33	89.10	53757
34	89.23	55386
35	89.41	57015
36	89.55	58644
37	89.70	60273
38	89.83	61902
39	89.96	63531
40	90.10	65160
41	90.23	66789
42	90.33	68418
43	90.43	70047

Table 10-2. STC1 Segment 0 Test Coverage and Duration (continued)

Intervals	Test Coverage (%)	Test Time (Cycles)
44	90.57	71676
45	90.67	73305
46	90.77	74934
47	90.89	76563
48	91.00	78192
49	91.08	79821
50	91.17	81450
51	91.26	83079
52	91.35	84708
53	91.42	86337
54	91.52	87966
55	91.63	89595
56	91.73	91224
57	91.81	92853
58	91.89	94482
59	91.97	96111
60	92.05	97740
61	92.11	99369
62	92.17	100998
63	92.24	102627
64	92.31	104256
65	92.38	105885
66	92.44	107514
67	92.51	109143
68	92.57	110772
69	92.63	112401
70	92.70	114030
71	92.76	115659
72	92.82	117288
73	92.92	118917
74	92.98	120546
75	93.06	122175
76	93.12	123804
77	93.20	125433
78	93.25	127062
79	93.31	128691
80	93.36	130320
81	93.42	131949
82	93.48	133578
83	93.55	135207
84	93.60	136836
85	93.66	138465
86	93.71	140094
87	93.76	141723
88	93.81	143352
89	93.86	144981
90	93.91	146610

Table 10-2. STC1 Segment 0 Test Coverage and Duration (continued)

Intervals	Test Coverage (%)	Test Time (Cycles)
91	93.96	148239
92	94.01	149868
93	94.07	151497
94	94.12	153126
95	94.17	154755
96	94.22	156384
97	94.27	158013
98	94.32	159642
99	94.37	161271
100	94.41	162900
101	94.46	164529
102	94.50	166158
103	94.54	167787
104	94.60	169416
105	94.64	171045
106	94.68	172674
107	94.72	174303
108	94.78	175932
109	94.82	177561
110	94.86	179190
111	94.91	180819
112	94.95	182448
113	94.99	184077
114	95.04	185706
115	95.08	187335
116	95.15	188964
117	95.19	190593
118	95.23	192222
119	95.27	193851
120	95.31	195480
121	95.35	197109
122	95.39	198738
123	95.43	200367
124	95.47	201996
125	95.51	203625

Table 10-3 gives the typical STC execution times for 40 intervals and 125 intervals at different clock rates. You can choose the number of intervals to be run based on the coverage needed and allowed time for STC execution.

Table 10-3. Typical Execution Times for STC1 Segment 0

Number of Intervals	Coverage	@ GCLK1 = 330 MHz STCCLK = 110 MHz	@ GCLK1 = 300 MHz STCCLK = 100 MHz
40	>90%	592.4 µS	651.6 µS
125	>95%	1.8511 mS	2.036 mS

10.6 STC1 Segment 1 (µSCU) Test Coverage and Duration

The test coverage and number of test execution cycles (STCCLK) for each test interval are shown in Table 10-4.

Table 10-4. STC1 Segment 1 Test Coverage and Duration

Intervals	Test Coverage (%)	Test Time (Cycles)
0	0	0
1	84.79	1629
2	87.96	3258
3	88.33	4887

Table 10-5 gives the typical STC execution times for 3 intervals at different clock rates. You can choose the number of intervals to be run based on the coverage needed and allowed time for STC execution.

Table 10-5. Typical Execution Times for STC1 Segment 1

Number of Intervals	Coverage	@ GCLK1 = 330 MHz STCCLK = 110 MHz	@ GCLK1 = 300 MHz STCCLK = 100 MHz
3	>88%	44.43 µS	48.87 µS

10.7 STC2 (nHET) Test Coverage and Duration

The test coverage and number of test execution cycles (STCCLK) for each test interval are shown in Table 10-6.

Table 10-6. STC2 Test Coverage and Duration

Intervals	Test Coverage (%)	Test Time (Cycles)
0	0	0
1	70.01	1365
2	77.89	2730
3	81.73	4095
4	84.11	5460
5	86.05	6825
6	87.78	8190
7	88.96	9555
8	89.95	10920
9	90.63	12285
10	91.20	13650
11	91.60	15015
12	92.02	16380
13	92.37	17745
14	92.66	19110
15	92.87	20475
16	93.04	21840
17	93.26	23205
18	93.47	24570
19	93.67	25935
20	93.82	27300
21	93.96	28665
22	94.12	30030

Table 10-6. STC2 Test Coverage and Duration (continued)

Intervals	Test Coverage (%)	Test Time (Cycles)
23	94.24	31395
24	94.38	32760
25	94.50	34125
26	94.72	35490
27	94.80	36855
28	94.90	38220
29	94.97	39585
30	95.03	40950
31	95.10	42315
32	95.16	43680
33	95.22	45045
34	95.27	46410
35	95.33	47775
36	95.42	49140
37	95.49	50505
38	95.54	51870
39	95.66	53235
40	95.69	54600
41	95.75	55965
42	95.79	57330
43	95.82	58695
44	95.85	60060
45	95.91	61425
46	95.95	62790
47	95.99	64155
48	96.01	65520
49	96.04	66885
50	96.07	68250
51	96.09	69615
52	96.12	70980
53	96.15	72345
54	96.19	73710
55	96.24	75075
56	96.29	76440
57	96.41	77805

Table 10-7 gives the typical STC execution times for 9 intervals and 57 intervals at different clock rates. You can choose the number of intervals to be run based on the coverage needed and allowed time for STC execution.

Table 10-7. Typical Execution Times for STC2

Number of Intervals	Coverage	@ VCLK = 110 MHz STCCLK = 110 MHz	@ VCLK = 150 MHz STCCLK = 75 MHz
9	>90%	111.68 µS	163.8 µS
57	>96%	707.3 µS	1.038 mS

10.8 STC Control Registers

STC control registers are accessed through Peripheral Bus (VBUSP) interface. Read and write access in 8,16, and 32 bit are supported.

The base address for the control registers of STC1 is FFFF E600h. The base address for the control registers of STC2 is FFFF 0800h.

NOTE: In suspend mode, all registers can be written irrespective of user or privilege mode and reads will not clear the 'read-clear' (RC) bits.

Table 10-8. STC Control Registers

Offset	Acronym	Register Description	Section
00h	STCGCR0	STC Global Control Register 0	Section 10.8.1
04h	STCGCR1	STC Global Control Register 1	Section 10.8.2
08h	STCTPR	Self-Test Run Timeout Counter Preload Register	Section 10.8.3
0Ch	STCCADDR1	STC Current ROM Address Register - CORE1	Section 10.8.4
10h	STCCICR	STC Current Interval Count Register	Section 10.8.5
14h	STCGSTAT	Self-Test Global Status Register	Section 10.8.6
18h	STCFSTAT	Self-Test Fail Status Register	Section 10.8.7
1Ch	CORE1_CURMISR3	CORE1 Current MISR Register	Section 10.8.8
20h	CORE1_CURMISR2	CORE1 Current MISR Register	Section 10.8.8
24h	CORE1_CURMISR1	CORE1 Current MISR Register	Section 10.8.8
28h	CORE1_CURMISR0	CORE1 Current MISR Register	Section 10.8.8
2Ch	CORE2_CURMISR3	CORE2 Current MISR Register	Section 10.8.9
30h	CORE2_CURMISR2	CORE2 Current MISR Register	Section 10.8.9
34h	CORE2_CURMISR1	CORE2 Current MISR Register	Section 10.8.9
38h	CORE2_CURMISR0	CORE2 Current MISR Register	Section 10.8.9
3Ch	STCSCSCR	Signature Compare Self-Check Register	Section 10.8.10
40h	STCCADDR2	STC Current ROM Address Register - CORE2	Section 10.8.11
44h	STCCLKDIV	STC Clock Divider Register	Section 10.8.12
48h	STCSEGPLR	STC Segment First Preload Register	Section 10.8.13

10.8.1 STC Global Control Register 0 (STCGCR0)

This register is described in Figure 10-8 and Table 10-9.

NOTE: On a power-on reset or system reset, this register gets reset to its default values.

Figure 10-8. STC Global Control Register 0 (STCGCR0) [offset = 00h]

31-16
INTCOUNT
R/WP-1

15-11	10-8	7-2	1-0
Reserved	CAP_IDLE_CYCLE	Reserved	RS_CNT
R-0	R/WP-1	R-0	R/WP-0

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privilege mode only; -*n* = value after reset

Table 10-9. STC Global Control Register 0 (STCGCR0) Field Descriptions

Bit	Field	Value	Description
31-16	INTCOUNT		Number of intervals of self-test run.
		0-FFFFh	This register specifies the number of intervals to run for the self-test run. This corresponds to the number of intervals to be run from the value reflected in the current interval counter.
15-11	Reserved	0	Reads return 0. Writes have no effect.
10-8	CAP_IDLE_CYCLE		Idle cycle before and after the capture clock.
		0	Disabled
		1	Enabled
7-2	Reserved	0	Reads return 0. Writes have no effect.
1-0	RS_CNT		Restart or Continue
			This bit specifies whether to continue the run from next interval onwards or to restart from interval 0. This bit gets reset after the completion of a self-test run.
		0	Continue STC run from the previous interval.
		1h	Restart STC run from interval 0.
		2h-3h	Reserved

10.8.2 STC Global Control Register 1 (STCGCR1)

This register is described in Figure 10-9 and Table 10-10.

NOTE: On a power-on reset or system reset, this register resets to its default values. Also, this register automatically resets to its default values at the completion of a self-test run.

The SEG0_CORE_SEL bits must be written first before the STC_ENA bits are written, in order for the STC to properly initiate the selected core for self-test.

Figure 10-9. STC Global Control Register 1 (STCGCR1) [offset = 04h]

31-16
Reserved
R-0

15-12	11-8	7-4	3-0
Reserved	SEG0_CORE_SEL	Reserved	STC_ENA
R-0	R/WP-0	R-0	R/WP-5h

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privilege mode only; -*n* = value after nPORST (power on reset) or System reset

Table 10-10. STC Global Control Register 1 (STCGCR1) Field Descriptions

Bit	Field	Value	Description
31-12	Reserved	0	Reads return 0. Writes have no effect.
11-8	SEG0_CORE_SEL		Selects cores in Segment 0 for self-test. These bits can be programmed only when SEG0_CORE_SEL is 0000. Once the field is written it ignores all further writes until the self-test sequence completes. This is to maintain coherency for self-test runs.
		5h	Select only Core1 for self-test.
		Ah	Select only Core2 for self-test.
		All other values	Select both cores for self-test in parallel.
7-4	Reserved	0	Reads return 0. Writes have no effect.
3-0	STC_ENA		Self-test run enable key.
		Ah	Self-test run is enabled.
		All other values	Self-test run is disabled.

10.8.3 Self-Test Run Timeout Counter Preload Register (STCTPR)

This register is described in Figure 10-10 and Table 10-11.

NOTE: On a power-on reset or system reset, this register gets reset to its default values.

Figure 10-10. Self-Test Run Timeout Counter Preload Register (STCTPR) [offset = 08h]

31 ... 0
RTOD
R/WP-FFFF FFFFh

LEGEND: R/W = Read/Write; WP = Write in privilege mode only; -*n* = value after nPORST (power on reset) or System reset

Table 10-11. Self-Test Run Timeout Counter Preload Register (STCTPR)

Bit	Field	Description
31-0	RTOD	Self-test timeout count preload. This register contains the total number of VBUS clock cycles it will take before an self-test timeout error (TO_ERR) will be triggered after the initiation of the self-test run. This is a fail safe feature to prevent the device from hanging up due to a run away test during the self-test. The preload count value gets loaded into the self-test time out down counter whenever a self-test run is initiated (STC_KEY is enabled) and gets disabled on completion of a self-test run.

10.8.4 STC Current ROM Address Register - CORE1 (STCCADDR1)

This register is described in Figure 10-11 and Table 10-12.

NOTE: When the RS_CNT bit in STCGCR0 is set to a 1 on the start of a self-test run, or on a power-on reset or system reset, this register resets to all zeroes.

Figure 10-11. STC Current ROM Address Register (STCCADDR1) [offset = 0Ch]

31 ... 0
ADDR
R-0

LEGEND: R = Read only; -*n* = value after nPORST (power on reset) or System reset

Table 10-12. STC Current ROM Address Register (STCCADDR1) Field Descriptions

Bit	Field	Description
31-0	ADDR	Current ROM Address This register reflects the current ROM address (address or micro code load) accessed during self-test Segment0 -Core1 and other segments. This is the current value of the STC program counter.

10.8.5 STC Current Interval Count Register (STCCICR)

This register is described in Figure 10-12 and Table 10-13.

NOTE: When the RS_CNT bit in STCGCR0 is set to a 1 or on a power-on reset, the current interval counter resets to the default value.

Figure 10-12. STC Current Interval Count Register (STCCICR) [offset = 10h]

31 ... 16
CORE2_ICOUNT
R-0

15 ... 0
CORE1_ICOUNT
R-0

LEGEND: R = Read only; -*n* = value after reset

Table 10-13. STC Current Interval Count Register (STCCICR) Field Descriptions

Bit	Field	Description
31-16	CORE2_ICOUNT	Interval Number This specifies the last executed interval number for Core2 in case of self-test being run on Segement0 redundant cores.
15-0	CORE1_ICOUNT	Interval Number This specifies the last executed interval number for Core1 in case of self-test being run on Segment0 or any other segments.

10.8.6 Self-Test Global Status Register (STCGSTAT)

This register is described in Figure 10-13 and Table 10-14.

NOTE: The two status bits can be cleared to their default values on a write of 1 to the bits. Additionally when the STC_ENA key in STCGCR1 is written from a disabled state to an enabled state, the two status flags get cleared to their default values. This register gets reset to its default value with power-on reset assertion.

Figure 10-13. Self-Test Global Status Register (STCGSTAT) [offset = 14h]

31-16
Reserved
R-0

15-12	11-8	7-2	1	0
Reserved	ST_ACTIVE	Reserved	TEST_FAIL	TEST_DONE
R-0	R-5h	R-0	R/W1CP-0	R/W1CP-0

LEGEND: R/W = Read/Write; R = Read only; W1CP = Write 1 to clear in privilege mode; -*n* = value after reset

Table 10-14. Self-Test Global Status Register (STCGSTAT) Field Descriptions

Bit	Field	Value	Description
31-12	Reserved	0	Reads return 0. Writes have no effect.
11-8	ST_ACTIVE		This field indicates if the self-test is active.
		Ah	Self-test is active.
		All other values	Self-test is not active.
			This will be set in the cycle after CORE_SEL is programmed. This will be reset once the STC generated the CPU reset after completion of the self-test.
7-2	Reserved	0	Reads return 0. Writes have no effect.
1	TEST_FAIL		Test Fail
		0	Self-test run has not failed.
		1	Self-test run has failed.
0	TEST_DONE		Test Done
		0	Self-test run is not completed.
		1	Self-test run is completed.
			The test done flag is set to a 1 for any of the following conditions: 1. When the STC run is complete without any failure 2. When a failure occurs on a STC run 3. When a timeout failure occurs
			Reset is generated to the CPU on which the STC run is being performed when TEST_DONE goes high (the test is completed).

10.8.7 Self-Test Fail Status Register (STCFSTAT)

This register is described in Figure 10-14 and Table 10-15.

NOTE: The three status bits can be cleared to their default values on a write of 1 to the bits. Additionally when the STC_ENA key in STCGCR1 is written from a disabled state to an enabled state, the three status bits get cleared to their default values. This register gets reset to its default value with power-on reset assertion.

When multiple segments are enabled in a self-test run, the STC will indicate the self-test complete on the first failed interval corresponding to a segment. The subsequent segments will not be run. FSEG_ID bits in this register indicate which segment failed.

Figure 10-14. Self-Test Fail Status Register (STCFSTAT) [offset = 18h]

31	16
Reserved	
R-0	

15	5	4	3	2	1	0
Reserved		FSEG_ID		TO_ERR	CORE2_FAIL	CORE1_FAIL
R-0		RCP-0		R/W1CP-0	R/W1CP-0	R/W1CP-0

LEGEND: R/W = Read/Write; R = Read only; W1CP = Write 1 to clear in privilege mode; -*n* = value after reset

Table 10-15. Self-Test Fail Status Register (STCFSTAT) Field Descriptions

Bit	Field	Value	Description
31-5	Reserved	0	Reads return 0. Writes have no effect.
4-3	FSEG_ID		Failed Segment Number
		0	Segment 0 Failed
		1	Segment 1 Failed
		All other values	Reserved
2	TO_ERR		Timeout Error
		0	No time out error occurred.
		1	Self-test run failed due to a timeout error.
1	CORE2_FAIL		CORE2 failure info for segment 0 only.
		0	No MISR mismatch for CORE2.
		1	Self-test run failed due to MISR mismatch for CORE2.
0	CORE1_FAIL		CORE1 failure info for segment 0 only.
		0	No MISR mismatch for CORE1.
		1	Self-test run failed due to MISR mismatch for CORE1.

10.8.8 CORE1 Current MISR Registers (CORE1_CURMISR[3:0])

This register is described in Figure 10-15 through Figure 10-18 and Table 10-16.

NOTE: This register gets reset to its default value with power-on or system reset assertion.

Figure 10-15. CORE1 Current MISR Register (CORE1_CURMISR3) [offset = 1Ch]

31	16
MISR[31:16]	
R-0	

15	0
MISR[15:0]	
R-0	

LEGEND: R = Read only; -*n* = value after reset

Figure 10-16. CORE1 Current MISR Register (CORE1_CURMISR2) [offset = 20h]

31	16
MISR[63:48]	
R-0	

15	0
MISR[47:32]	
R-0	

LEGEND: R = Read only; -*n* = value after reset

Figure 10-17. CORE1 Current MISR Register (CORE1_CURMISR1) [offset = 24h]

31	16
MISR[95:80]	
R-0	

15	0
MISR[79:64]	
R-0	

LEGEND: R = Read only; -*n* = value after reset

Figure 10-18. CORE1 Current MISR Register (CORE1_CURMISR0) [offset = 28h]

31	16
MISR[127:112]	
R-0	

15	0
MISR[111:96]	
R-0	

LEGEND: R = Read only; -*n* = value after reset

Table 10-16. CORE1 Current MISR Register (CORE1_CURMISRn) Field Descriptions

Bit	Field	Description
127-0	MISR	MISR data from CORE1 This register contains the MISR data from the CORE1 for the most recent interval in case of segment 0 and all other segments. This value is compared with the GOLDEN MISR value copied from ROM.

10.8.9 CORE2 Current MISR Registers (CORE2_CURMISR[3:0])

This register is described in Figure 10-19 through Figure 10-22 and Table 10-17.

NOTE: This register gets reset to its default value with power-on or system reset assertion.

Figure 10-19. CORE2 Current MISR Register (CORE2_CURMISR3) [offset = 2Ch]

31	16
MISR[31:16]	
R-0	

15	0
MISR[15:0]	
R-0	

LEGEND: R = Read only; -*n* = value after reset

Figure 10-20. CORE2 Current MISR Register (CORE2_CURMISR2) [offset = 30h]

31	16
MISR[63:48]	
R-0	

15	0
MISR[47:32]	
R-0	

LEGEND: R = Read only; -*n* = value after reset

Figure 10-21. CORE2 Current MISR Register (CORE2_CURMISR1) [offset = 34h]

31	16
MISR[95:80]	
R-0	

15	0
MISR[79:64]	
R-0	

LEGEND: R = Read only; -*n* = value after reset

Figure 10-22. CORE2 Current MISR Register (CORE2_CURMISR0) [offset = 38h]

31	16
MISR[127:112]	
R-0	

15	0
MISR[111:96]	
R-0	

LEGEND: R = Read only; -*n* = value after reset

Table 10-17. CORE2 Current MISR Register (CORE2_CURMISRn) Field Descriptions

Bit	Field	Description
127-0	MISR	MISR data from CORE2 This register contains the MISR data from the CORE2 for the most recent interval in case of segment 0l. This value is compared with the GOLDEN MISR value copied from ROM.

10.8.10 Signature Compare Self-Check Register (STCSCSCR)

This register is described in Figure 10-23. This register is used to enable the self-check feature of the CPU Self-Test Controller's (STC) signature compare logic. Self-check can only be done for the STC interval 0 by setting the RS_CNT bit in STCGCR0 to 1 to restart the self-test. The STC run will fail for signature miss-compare, provided the signature compare logic is operating correctly. To proceed with regular CPU self-test, STCSCSCR should be programmed to disable the self-check feature and clear the RS_CNT bit in STCGCR0 to 0. This register gets reset to its default value with any system reset assertion.

Figure 10-23. Signature Compare Self-Check Register (STCSCSCR) [offset = 3Ch]

31-16
Reserved
R-0

15-5	4	3-0
Reserved	FAULT_INS	SELF_CHECK_KEY
R-0	R/WP-0	R/WP-5h

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privilege mode only; -*n* = value after nPORST (power on reset) or System reset

Table 10-18. Signature Compare Self-Check Regsiter (STCSCSCR) Field Descriptions

Bit	Field	Value	Description
31-5	Reserved	0	Reads return 0. Writes have no effect.
4	FAULT_INS		Enable fault insertion.
		0	No fault is inserted.
		1	Insert stuck-at-fault inside CPU so that STC signature compare will fail.
3-0	SELF_CHECK_KEY		Signature compare logic self-check enable key.
		Ah	Signature compare logic self-check is enabled. This allows a fault to be inserted using the FAULT_INS field.
		All other values	Signature compare logic self-check is disabled The FAULT_INS field has no effect in this case.

10.8.11 STC Current ROM Address Register - CORE2 (STCCADDR2)

This register is described in Figure 10-24 and Table 10-19.

NOTE: When the RS_CNT bit in STCGCR0 is set to a 1 on the start of a self-test run, or on a power-on reset or system reset, this register resets to all zeroes.

Figure 10-24. STC Current ROM Address Register (STCCADDR2) [offset = 40h]

31-0
ADDR
R-0

LEGEND: R = Read only; -*n* = value after nPORST (power on reset) or System reset

Table 10-19. STC Current ROM Address Register (STCCADDR2) Field Descriptions

Bit	Field	Description
31-0	ADDR	Current ROM Address This register reflects the current ROM address (address or micro code load) accessed during self-test Segment0 -Core2. This is the current value of the STC program counter.

10.8.12 STC Clock Prescalar Register (STCCLKDIV)

This register is described in Figure 10-25. This register is used to configure STC clock divider ratio for each segment. STCCLK is derived from the system clock (GCLK1 for STC1 and VCLK2 for STC2) and the configured ratio is applied when the corresponding segment is under test. The division ratio programmed in this register will have effect only when the value in the CLKDIV field of the STCLKDIV register (FFFF E108h) from SYS2 module is zero. Else the division ratio will be taken from SYS2. This is done for software compatibility.

NOTE: The clock divider ratio is applied when the corresponding segment is under test.

Figure 10-25. STC Clock Prescalar Register (STCCLKDIV) [offset = 44h]

31 27	26 24	23 19	18 16
Reserved	CLKDIV0	Reserved	CLKDIV1
R-0	R/WP-0	R-0	R/WP-0

15 0
Reserved
R-0

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privilege mode only; -*n* = value after nPORST (power on reset) or System reset

Table 10-20. STC Clock Prescalar Register (STCCLKDIV) Field Descriptions

Bit	Field	Value	Description
31-27	Reserved	0	Reads return 0. Writes have no effect.
26-24	CLKDIV0		STCCLK divider for segment 0.
		0-7h	Division ratio of segment 0 will be n+1. STCCLK clock will be divided by (n+1) for segment 0.
23-19	Reserved	0	Reads return 0. Writes have no effect.
18-16	CLKDIV1		STCCLK divider for segment 1.
		0-7h	Division ratio of segment 1 will be n+1. STCCLK clock will be divided by (n+1) for segment 1.
15-0	Reserved	0	Reads return 0. Writes have no effect.

10.8.13 Segment Interval Preload Register (STCSEGPLR)

This register is described in Figure 10-26. This register is used to specify the segment for which the first interval will be run. The address of the first interval of the selected segment is loaded to the STC ROM address counter before the test is started.

Figure 10-26. Segment Interval Preload Register (STCSEGPLR) [offset = 48h]

31–16
Reserved
R-0

15–2	1–0
Reserved	SEGID_PLOAD
R-0	RWP-0

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privilege mode only; -*n* = value after nPORST (power on reset) or System reset

Table 10-21. Segment Interval Preload Register (STCSEGPLR) Field Descriptions

Bit	Field	Value	Description
31-2	Reserved	0	Reads return 0. Writes have no effect.
1-0	SEGID_PLOAD		Specifies the segment for the first interval to be run.
		0	Preload the address of the 1st interval for Segment 0.
		1	Preload the address of the 1st interval for Segment 1.
		All other values	Reserved

10.9 STC Configuration Example

The following example provides steps to configure STC1 to run self-test on CPUs and the µSCU unit. It that the PLL is locked and selected as the system clock source with GCLK1 = 330 MHz and HCLK = 110 MHz prior to going through the following configurations.

10.9.1 Example: STC1 Self-Test Run

This example explains the configurations for running STC Test for on 40 test intervals.

1. Maximum STC clock rate support at 330 MHz GCLK1 is 110 MHz. Divide GCLK1 by 3 to achieve this clock rate. Bits STCCLKDIV[26:24] and STCCLKDIV[18:16] need to be configured.

   STCCLKDIV[26:24] = 2, STCCLKDIV[18:16] = 2
2. Clear CPU RST status bit in the System Exception Status Register in the system module.

   SYSESR[5] = 1
3. Configure the test interval count in STC module. Note that in case of multiple segments, segments run sequentially, one after another depending on the number of intervals selected.

   STCGCR0[31:16] = 40.
4. Configure self-test run time out counter preload register.

   STCTPR[31:0] = 0xFFFFFFFF
5. Optionally, configure SEG0_CORE_SEL bits in register STCGCR1 to select one of the redundant cores. By default bits SEG0_CORE_SEL are clear, which configures the STC to run both redundant cores in parallel.
6. Enable CPU self-test.

   STCGCR1[3:0]= 0xA;
7. Perform a context save of CPU state and configuration registers that get reset on CPU reset.
8. Put the CPU in idle mode by executing the CPU idle instruction.
   asm(“ WFI”)
9. Upon CPU reset, verify the CPU RST status bit in the System Exception Status Register is set. This also verifies that no other resets occurred during the self-test.

   SYSESR[5] == 1
10. Check the STCGSTAT register for the self-test status.

    Check TEST_DONE bit before evaluating TEST_FAIL bit.

    If (TEST_DONE = 1 and TEST_FAIL = 1), the self-test is completed and Failed.

    - Read STC Fail Status Register STCFSTAT[2:0] to identify the type of Failure (Timeout, CORE1 fail, CORE2 fail, FSEG_ID).

    In case there is no failure (TEST_DONE = 1 and TEST_FAIL = 0), the CPU self-test is completed successfully.

    - Recover the CPU status, configuration registers and continue the application software.

10.10 Self-Test Controller Diagnostics

This section provides the recommended flow for the self-test controller diagnostics. This test is recommended to be done at the application startup only, not with individual interval runs during the application.

Step 1: Configure the interval count to 1 in STCGCR0 register.

Segment 0

Step 2: Enable the SELF_CHECK_KEY and FAULT_INS bits in the STCSCSCR register and kick off the self-test by enabling the first interval of segment 0. On the completion of self-test, TEST_FAIL bit will be set in the STCGSTAT register. Check if the FSEGID bits in the STCFSTAT register are set to 00. Depending on the segment 0 configuration (parallel or individual cores), the CORE1_FAIL or CORE2_FAIL bits would be set.

Step 3: Disable one or both of the SELF_CHECK_KEY and FAULT_INS bits in the STCSCSCR register. Then restart the self-test by programming bit 0 of the STCGCR0 register to 1. On the completion of the test, the TEST_FAIL bit will be cleared in the STCGSTAT register.

Segment 1 (for STC1 only)

Step 4: Configure the SEGID_PLOAD bits in STCSEGPLR register to select the first interval of segment 1. Configure RS_CNT bit in STCGCR0 register to 1. This will start the self-test from the first interval of the selected segment. On the completion of self-test, TEST_FAIL bit will be set in the STCGSTAT register. Check if the FSEGID bits in the STCFSTAT register are set to 01.

Step 5: Disable one or both of the SELF_CHECK_KEY and FAULT_INS bits in the STCSCSCR register. Then restart the self-test by programming bit 0 of the STCGCR register to 1. On the completion of the test, the TEST_FAIL bit will be cleared in the STCGSTAT register.

After the diagnostics, the application can continue with the self-test as described in Section 10.4.

Chapter 11

SPNU563A–March 2018

System Memory Protection Unit (NMPU)

This chapter describes the System Memory Protection Unit (NMPU).

Topic **Page**

11.1 Overview

The System Memory Protection Unit module(s) provide an mechanism to control the memory access rights of bus masters in the system other than the host CPU. The programmer's model for the System Memory Protection unit is similar to but a subset of the host CPU's own memory protection unit. It allows memory partition into multiple regions and allows individual access protection for each region from a bus master point of view. An access from bus master is checked against each memory region access permission to make sure that the access from bus master does not alter the unintended memory region that could cause a system failure.

11.1.1 Features

NMPU offers the following main features:

- Software programmer model is similar to but a subset of the host CPUs own memory protection unit.
- Provide protection to memory regions ranging from 32-bytes to 4GB in size
- Up to 8 memory protection regions. Note that the number of memory region is different for each bus master IP that the NMPU is dedicated for. Each region is defined by the base address and region size that are programmable in NMPU control registers. Table 11-1 defines the number of region available for the corresponding bus master IP.
- Programmable access permissions for each region such as full access, read-only, write-only, and no access.
- Different access permissions for user and privilege mode.
- On access violation, NMPU can notify ESM if ERRENA key in MPUCTRL1 register (Section 11.4.7) is enabled.

Table 11-1. NMPU Region

NMPU Module	Number of Available Regions
DMA-NMPU	8
Peripheral Interconnect Subsystem-NMPU	8
EMAC-NMPU	2

11.1.2 Safety Diagnostic

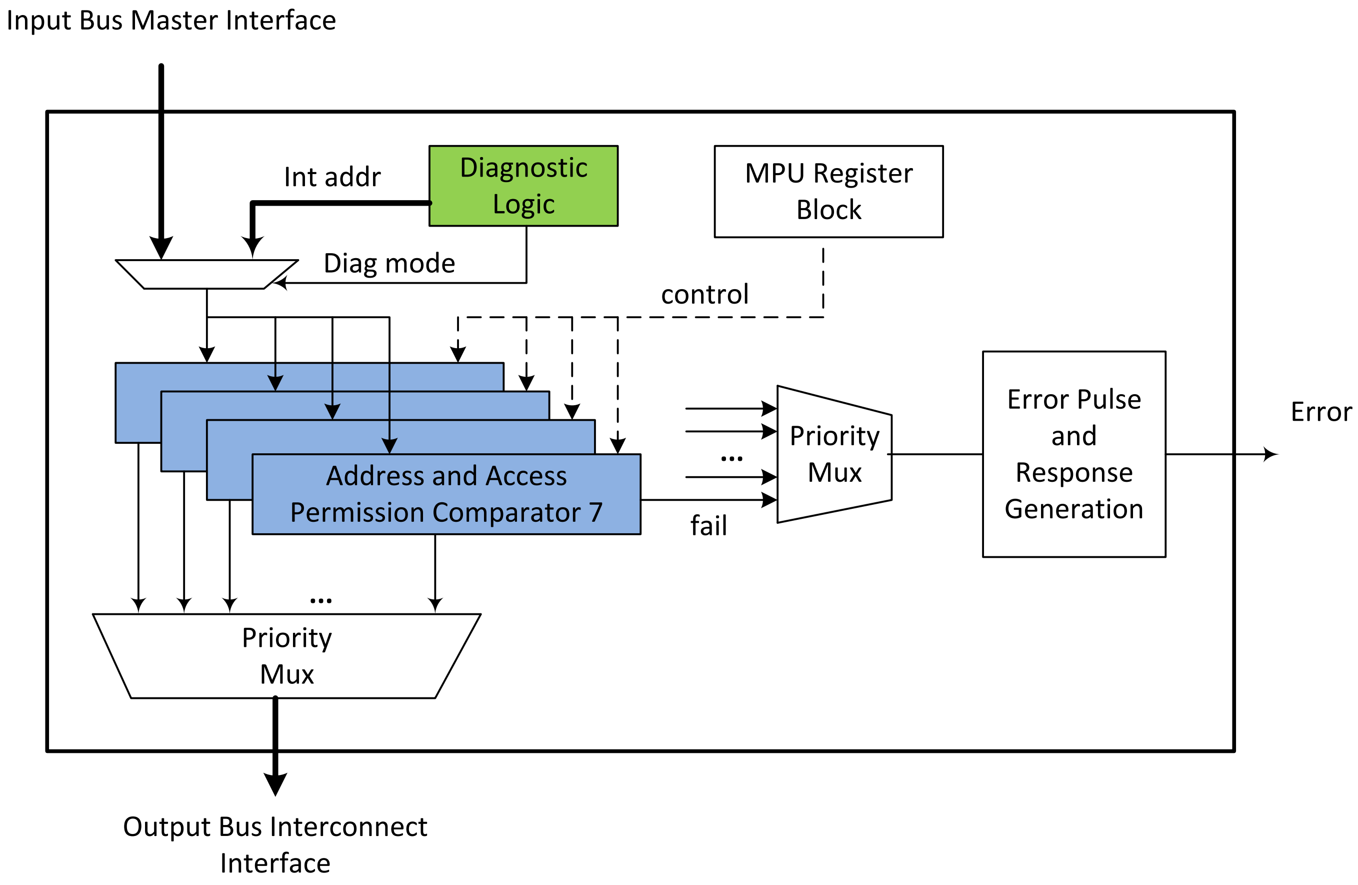
NMPU offers the following safety diagnostic capabilities:

- Provide a lock mechanism to avoid unintentional changes to NMPU control registers.
- Provide diagnostic capability to check the MPU region access permission logic.

11.1.3 Block Diagram

Figure 11-1 shows the block diagram of NMPU.

Figure 11-1. NMPU Block Diagram


Input Bus Master Interface
Int addr
Diagnostic Logic
MPU Register Block
Diag mode
control
Address and Access Permission Comparator 7
...
Priority Mux
fail
Error Pulse and Response Generation
Error
...
Priority Mux
Output Bus Interconnect Interface

11.2 Module Operation

11.2.1 Functional Mode

On reset, NMPU is disabled and no filtering will be done on the bus. User must ensure that no bus transaction from the master is on going while NMPU is getting disabled or enabled. This is similar to the need to flush transactions using memory barrier instructions on the CPU before changing CPU MPU setting.

The MPU can be enabled by writing 0xAh to the MPUENA key bits of MPUCTRL1 register and can be disabled by writing 0x5h to the same bits.

Access permission (AP) for each MPU region is defined in AP field in the MPU region access control register (MPUREGACR), see Table 11-2.

Table 11-2. Access Permission

AP Field	Privilege Mode Permissions	User Mode Permissions
000	No Access	No Access
001	Read/Write	No Access
010	Read/Write	Read Only
011	Read/Write	Read/Write
100	No Access	No Access
101	Read only	No Access
110	Read only	Read only
111	No Access	No Access

Each MPU region has three control registers:

- MPUREGBASE: MPU based address register. It defines the base address for a particular MPU region
- MPUREGSENA: MPU region size and enable register. It defines the size of a particular MPU region and allows you to enable the region
- MPUREGACR: MPU region access control register. It defines the MPU region accessing permission for user or privilege mode

NMPU has one region register that you have to configure to determine which MPU region user is programming the corresponding MPUREGBASE, MPUREGSENA, and MPUREGACR registers. In this scheme, the MPUREGBASE, MPUREGSENA, and MPUREGACR can share the same memory map offset from user programming point of view.

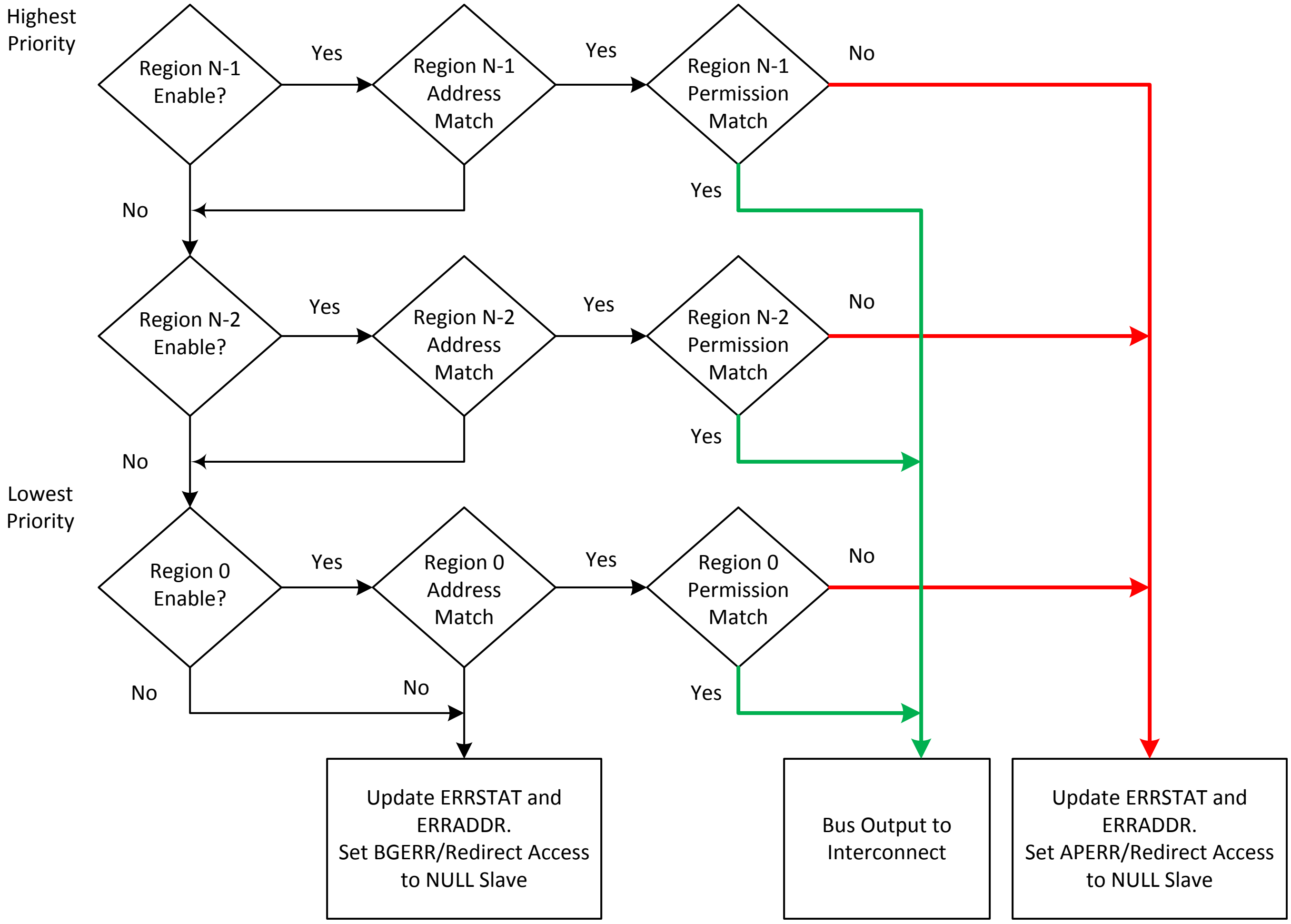
Size of each MPU region can vary from 32 bytes to 4 GB. Region based address must start at an offset that is a multiple of region size. In case the base address does not start at an offset that is a multiple of region size, the region size takes priority and MPU ignores the LSB bits of base address.

Overlapping regions enable efficient programming of memory map. When the incoming address hits multiple MPU regions, access permission is decided by the highest numbered region for which there was an address compare match. In MPU configuration with 8 regions, region 0 has the lowest priority and region 7 has the highest priority.

Figure 11-2 shows how the region priority is resolved in a high-level abstraction.

MPU does not support default background memory map. If memory protection is enabled without region configuration, all transactions will result in bus error response. Before the protection unit is enabled, care needs to be taken to ensure that at least one valid protection region is specified and its access permission fields are defined.

Figure 11-2. MPU Region Priority


Highest Priority
Region N-1 Enable?
Yes
Region N-1 Address Match
Yes
Region N-1 Permission Match
No
No
Yes
Region N-2 Enable?
Yes
Region N-2 Address Match
Yes
Region N-2 Permission Match
No
No
Yes
Lowest Priority
Region 0 Enable?
Yes
Region 0 Address Match
Yes
Region 0 Permission Match
No
No
No
Yes
Update ERRSTAT and ERRADDR. Set BGERR/Redirect Access to NULL Slave
Bus Output to Interconnect
Update ERRSTAT and ERRADDR. Set APERR/Redirect Access to NULL Slave

11.2.2 Diagnostic Mode

Diagnostic mode can be used to verify the MPU address and access permission comparator logic. Entering or exiting the diagnostic mode will automatically clear the MPUERRSTAT and MPUERRADDR registers. Memory protection must be disabled while entering or exiting diagnostic mode. There are two different diagnostic modes: internal diagnostic mode and external diagnostic mode.

11.2.2.1 Internal Diagnostic Mode

In internal diagnostic mode, diagnostic logic inside the NMPU module drives the input of the MPU address and access permission comparator logic. You can program the address for which comparison needs to be performed and the type of transaction (read/write and user/privilege). For every write to the MPUDIAGADDR register, an address and access permission comparison is performed and the results are stored in MPUERRSTAT and MPUERRADDR registers. ERROR output to ESM will be generated if ERRENA key in MPUCTRL2 register is Ah. You must ensure that no bus transactions from the master are going on while NMPU is in internal diagnostic mode. NMPU does not accept any access originated from the bus master and ensures that the internal diagnostic logic will not result in any bus transactions on to the bus interconnect.

How to use the internal diagnostic mode is discussed in Section 11.3.

11.2.2.2 External Diagnostic Mode

In external diagnostic mode, the actual bus master initiates the access to the NMPU. Address of the access from the bus master is replaced by the address in MPUDIAGADDR register before the address reaches the address comparator logic. In this mode, both bus error response and ERROR pulse to ESM (if ERRENA = Ah) are generated for accesses that violate the access permissions. This diagnostic mode is useful to test the full signal chain from bus master access generation logic to NMPU comparator logic.

How to use the external diagnostic mode is discussed in Section 11.3.

11.2.3 Functional Fail Safe

Since NMPU module check and manipulate address or mode of bus master transaction, it is important to have functional fail safe features to guarantee that faults in MPU region checking, address translation, or user mode translation can be detected.

11.2.3.1 Run-time Diagnostics for Functional Features

Since features like input address masking, address translation and mode translation are integrated along with a critical function like memory protection, NMPU needs to have the following hardware logic for run-time diagnostics. This logic is implemented using 1oo1D safety architecture.

- There are two independent blocks (primary and checker) running in lock-step and compare address masking output every cycle. Outputs from NMPU are driven by the primary block.
- There are two independent blocks (primary and checker) running in lock-step and compare address translation output every cycle. Outputs from NMPU are driven by the primary block.
- There are two independent blocks (primary and checker) running in lock-step and compare mode translation output every cycle. Outputs from NMPU are driven by the primary block.
- If there is a lockstep comparison error, DIAGERR bit in MPUERRSTAT register is set to 1. ERRFLAG bit in the same register is also set. ERROR pulse output to ESM is generated irrespective of ERRENA key value in MPUCTRL2 register.
- A fault insertion allows user verifying that the individual lockstep comparator logic is functional and avoid latent fault. User can program the fault insertion bits in MPUDIAGCTRL register to introduce a fault in one of the lockstep comparator inputs for input address masking, address translation or mode translation during start up or shut down of the device.

11.2.3.2 Lock Feature for NMPU

Lock feature prevents unintentional updates to NMPU registers. Writes to registers other than MPUERRSTAT is possible only when NMPU is unlocked by writing 0xAh to LOCK key bits of MPULOCK register. On reset, these bits are set to 0x5h and hence the NMPU registers are in the locked state. All NMPU registers are writable only in privilege mode. There is no built in protection based on master ID. It is user responsibility to ensure that only a single valid privilege master updates the MPU registers.

11.2.3.3 Multibit Keys for Feature Enable/Disable

4-bit key is used to protect critical function enters enable or disable state from soft error. These key are updated only if the write data is 0x5h or 0xAh. Register write is ignored for all other write values. A built in correction logic detects single bit soft error on this field and corrects the value in the next cycle. Functionality and register read data remain the same during the correction cycle.

11.3 How to Use NMPU

11.3.1 How to Use NMPU in Functional Mode

The NMPU is used to configure the bus master MPU region in such a way that the bus master does not interfere with the memory region reserved for other tasks and not belonging to the system partitioning for the IP.

Once user determines the architectural memory partitioning of the IP bus masters on memory system frame according to their application, user should configure the corresponding MPU region for each bus master accordingly.

Figure 11-3 shows the example recommended memory setting for a bus master in the device, for example, DMA.

Assume the DMA bus master has 3 MPU regions.

The lowest priority MPU region1 is programmed to enable full read and write to peripheral memory frame.

MPU region 2 is programmed to allow read and write to a lower 10KB portion of the system RAM starting from 0x0800_0000.

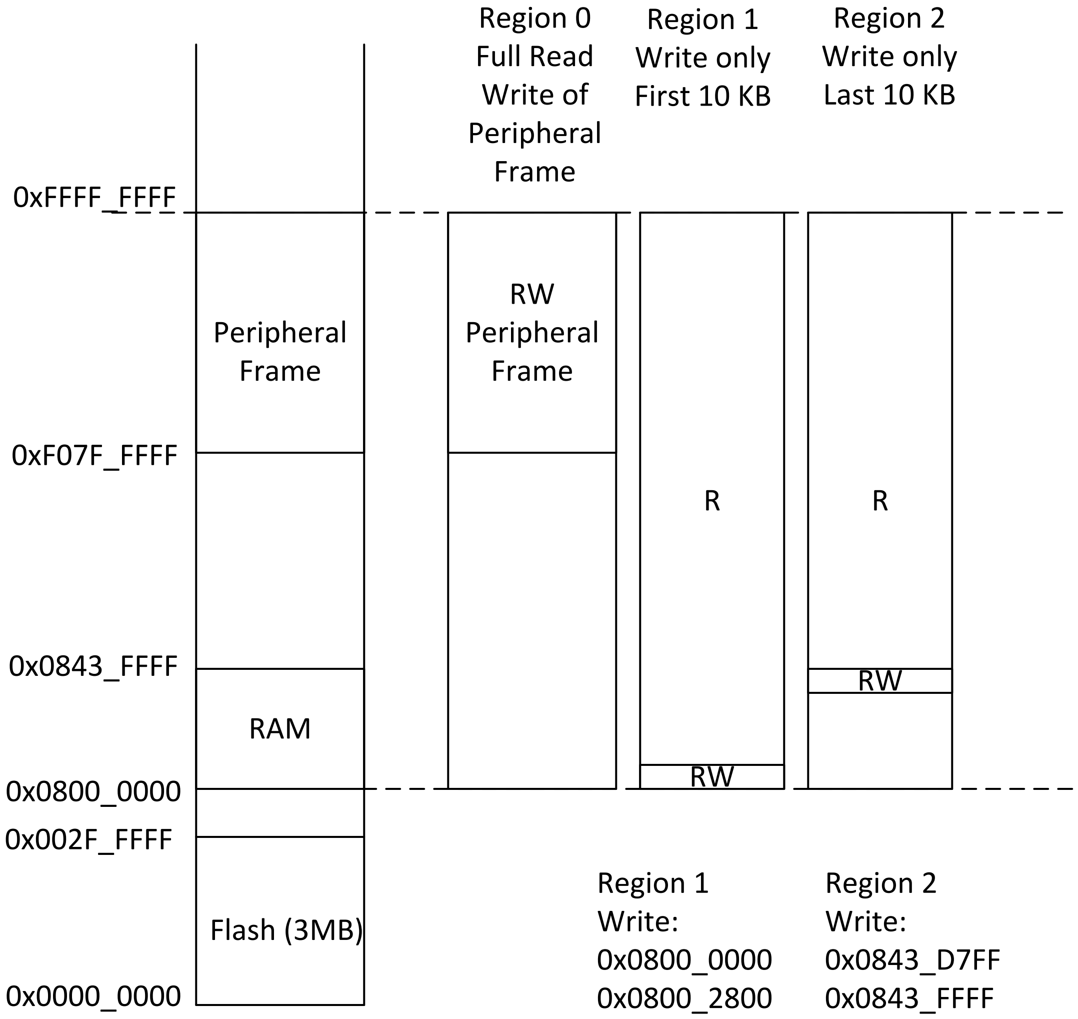
MPU region 3 is programmed to allow read and write to the upper 10KB away from 0x0843_FFFF portion of the system RAM.

Any access in between 0x0800_2800 and 0x0843_D7FF is a read only mode for DMA.

With this configuration, DMA can have read or write access to the entire peripheral frame and only able to write to upper or lower 10KB of the system RAM.

The rest of the system RAM is reserved for other tasks in which the DMA should not interfere with.

Figure 11-3. Example of DMA 3 MPU Region Set Up



This will allow DMA to be able to create transfer from any location within peripheral frame to a specific allocation in system RAM to avoid corrupting the system memory RAM reserved for other tasks.

Following is the recommended generic software sequence to setup the MPU regions:

1. Make sure the bus master is idle and not sending any transaction. Please follow the bus master TRM on how to idle the bus interface. it will be different from one bus master to another.
2. Write 0xA to unlocked field LOCK of MPULOCK register (Section 11.4.2) to allow update to NMPU control registers.
3. Enable MPU error pulse event to ESM by writing 0xA to field ERRENA field of MPUCTRL2 register (Section 11.4.8). Program this step if and only if the bus master has no capability to capture the MPU transaction error from NMPU. If bus master has the ability to report transaction error, disable the ERRENA. Software will rely on bus master to trigger error event causing interrupt to the CPU.
4. Read MPUTYPE register (Section 11.4.9) to identify how many regions are implemented for this bus master in a particular device.
5. Program the MPUREGNUM register (Section 11.4.13) to indicate that MPU region number to write starting address, size, permission, and so on.
6. Program the MPUREGBASE register (Section 11.4.10) to set the base address for the particular MPU region number that was set in step 5.

7. Program the MPUREGSENA register (Section 11.4.11) to set the size, and enable MPU region number indicated by step 5 above. Notice that this is not yet enabling the NMPU module.
8. Program the MPUREGACR register (Section 11.4.12) to set access permission for user or privilege mode.
9. Repeat steps 5 to 8 for the remaining MPU regions available for the particular NMPU instance.
10. Write 0xA to MPUENA field of MPUCTRL1 register (Section 11.4.7) to enable NMPU to start access permission check.
11. Write 0x5 to unlocked key field LOCK of MPULOCK register (Section 11.4.2) to avoid unintentional write to NMPU configuration registers.
12. Enable bus master to start transaction. Please follow the bus master TRM on how to start the bus interface. it will be different from one bus master to another.

During application run time, if the NMPU detects a memory access violation or the functional fail safe logic detects a lock step compare error, you will be interrupted. Refer to TRM to find out which ESM group and channel the NMPU or bus master error event is mapped (based on step 3 above) and which VIM channel the ESM interrupt output is mapped.

Following is the recommended generic software sequence to find out what causes MPU error:

1. Read the MPUERRSTAT register:
   - If RERR (read error), WERR (write error), BGERR (background error), or APERR (access permission error) is set and ERRFLAG is also set, this indicates that the bus master tries to access the address location that violates the memory protection setting. This can happen due to software bug, transient fault, or permanent fault.
   - If DIAGERR (safety diagnostic error) bit is set and ERRFLAG is also set, this indicates that the 1oo1D diagnostic architect for input address masking, address translation, or mode translation has detected an error. This can happen due to transient fault or permanent fault.
2. You can further read additional information from REGION, MASTERID, or ERRFLAG bit fields of MPUERRSTAT register and MPUERRADDR register to narrow down the causes.
3. In fault case, it is up to end application to decide on whether to bring the system to safe state or ignore NMPU error or try to recover by retrying transaction if master is able to support it. If bus master does not support retry of a particular transaction, use can halt bus master, starts NMPU internal diagnostic (see Section 11.2.2.1). Assume diagnostic result passes, you can restart bus master operation. If the recover attempt still fails, you can decide to bring the system to safe state.

11.3.2 How to Use Diagnostics

Diagnostic mode can be used to verify the MPU address and access permission comparator logic working properly at either start up time or during application run time with/without error encountered. This is achieved by internal or external diagnostic mode. Entering or exiting the diagnostic mode will automatically clear the MPUERRSTAT (Section 11.4.5) and MPUERRADDR (Section 11.4.6) registers. It is recommended that you back up the values of MPUERRSTAT (Section 11.4.5) and MPUERRADDR (Section 11.4.6) registers to system RAM prior to start diagnostic during run time.

Another diagnostic mode of NMPU is for functional fail safe diagnostic. Features like input address masking, address translation and mode translation are integrated along with a critical function like memory protection, thus NMPU needs to have the hardware logic for run-time diagnostics. This logic is implemented using 1oo1D safety architecture.

11.3.2.1 How to Run Internal Diagnostic Mode

In internal diagnostic mode, diagnostic logic inside the NMPU module drives the input of the MPU address and access permission comparator logic. You can program the address for which comparison needs to be performed and the type of transaction (read/write and user/privilege). For every write to the MPUDIAGADDR register, an address and access permission comparison is performed and the results are stored in MPUERRSTAT and MPUERRADDR registers. ERROR output to ESM will be generated if ERRENA field in MPUCTRL2 register is set to 0xAh. The following is the recommended sequence for internal diagnostic mode:

1. User must ensure that no bus transactions from the master are going on while NMPU is in internal diagnostic mode by disabling bus master access. Please follow the bus master TRM on how to idle the bus interface. it will be different from one bus master to another.
2. Unlock the MPU registers by writing 0xA to LOCK field of MPULOCK register.
3. Disable memory protection by writing 0x5 to MPUENA key of MPUCTRL1 register.
4. Program the different MPU regions in MPUREGBASE0-7, MPUREGSENA0-7 and MPUREGACR0-7 registers.
5. In MPUDIAGCTRL register, program the INT/EXT bit as 0.
6. Enable the DIAGKEY in MPUDIAGCTRL register by writing 0xA.
7. Program the diagnostic transaction type as read/write in R/W bit in MPUDIAGCTRL register. User/Privilege mode is set in U/P bit in the same register.
8. Program the diagnostic address in MPUDIAGADDR register.
9. If there should be an access permission violation according to the diagnostic test, error flag is set.
10. Read the MPUERRSTAT and MPUERRADDR registers and verify the expected results.
11. Clear the ERRFLAG bit in MPUERRSTAT register.
12. Repeat steps 6 to 10 for different values of diagnostic address and R/W bit.
13. Exit the diagnostic mode by writing 0x5 to DIAGKEY field in MPUDIAGCTRL register.
14. Lock the MPU registers by writing 0xA to LOCK field of MPULOCK register.
15. User enables bus master. Please follow the bus master TRM on how to enable the bus interface. it will be different from one bus master to another.

11.3.2.2 How to Run External Diagnostic Mode

In external diagnostic mode, the actual bus master initiates the memory transaction. Address from the bus master is replaced by the address in MPUDIAGADDR register before the address reaches the address comparator logic. In this mode, both bus error response and ERROR pulse to ESM (if ERRENA field of MPUCTRL2 register is set to 0xAh) are generated for accesses that violate the access permissions. This mode is useful for a complete signal chain testing from bus master internal bus generation logic to NMPU comparator logics. The following is the recommended sequence for external diagnostic mode:

1. User must ensure that no bus transactions from the master are going on while NMPU is configuring in external diagnostic mode by disabling bus master access. Please follow the bus master TRM on how to idle the bus interface. it will be different from one bus master to another.
2. Unlock the MPU registers by writing 0xA to LOCK field of MPULOCK register.
3. Disable memory protection by writing 0x5 to MPUENA field of MPUCTRL1 register.
4. Program the different MPU regions in MPUREGBASE0-7, MPUREGSENA0-7 and MPUREGACR0-7 registers.
5. In MPUDIAGCTRL register, program the INT/EXT bit as 1.
6. Enable the DIAGKEY field of MPUDIAGCTRL register by writing 0xA.
7. Program the diagnostic address in MPUDIAGADDR register.
8. Enable memory protection by writing 0xA to MPUENA field of MPUCTRL1 register.
9. User enables bus master. Please follow the bus master TRM on how to enable the bus interface. it will be different from one bus master to another.
10. Initiate ONE bus transactions using the actual bus master.
11. Read the MPUERRSTAT and MPUERRADDR registers and verify the expected results.
12. Clear the ERRFLAG bit in MPUERRSTAT register.
13. Repeat steps 7,10, 11, 12 for different values of diagnostic address.
14. Disable memory protection by writing 0x5 to MPUENA field of MPUCTRL1 register.
15. Exit the diagnostic mode by writing 0x5 to DIAGKEY field in MPUDIAGCTRL register.
16. Lock the MPU registers by writing 0xA to LOCK field of MPULOCK register.
17. Restart the bus master functional operation. Please follow the bus master TRM on how to enable the bus interface. it will be different from one bus master to another.

11.4 NMPU Registers

The new memory protection unit (NMPU) registers listed in Table 11-3 are accessed through the system module register space in the Cortex-R5F CPUs memory-map. All registers are 32-bit wide and are located on a 32-bit boundary. Reads and writes to registers are supported in 8-, 16-, and 32-bit accesses. Refer to the device specific datasheet for the base address of each instance of NMPU in the device.

NOTE: If a register is not implemented, corresponding address location behaves like a reserved location, that is, reads return 0 and writes have no effect. User mode writes to NMPU registers are ignored. No error response is given for such an access. Writes to registers other than MPUERRSTAT register are ignored, when NMPU registers are locked (LOCK = 5h in MPULOCK register). No error response is given for such an access.

Table 11-3. NMPU Registers

Offset	Acronym	Register Description	Section
00h	MPUREV	MPU Revision ID Register	Section 11.4.1
04h	MPULOCK	MPU Lock Register	Section 11.4.2
08h	MPUDIAGCTRL	MPU Diagnostics Control Register	Section 11.4.3
0Ch	MPUDIAGADDR	MPU Diagnostic Address Register	Section 11.4.4
10h	MPUERRSTAT	MPU Error Status Register	Section 11.4.5
14h	MPUERRADDR	MPU Error Address Register	Section 11.4.6
20h	MPUCTRL1	MPU Control Register 1	Section 11.4.7
24h	MPUCTRL2	MPU Control Register 2	Section 11.4.8
2Ch	MPUTYPE	MPU Type Register	Section 11.4.9
30h	MPUREGBASE	MPU Region Base Address Register	Section 11.4.10
34h	MPUREGSENA	MPU Region Size and Enable Register	Section 11.4.11
38h	MPUREGACR	MPU Region Access Control Register	Section 11.4.12
3Ch	MPUREGNUM	MPU Region Number Register	Section 11.4.13

11.4.1 MPU Revision ID Register (MPUREV)

Figure 11-4. MPU Revision ID Register (MPUREV) [offset = 00h]

31 30	29 28	27 16
SCHEME	Reserved	FUNC
R-1	R-0	R-A0Ch

15 11	10 8	7 6	5 0
RTL	MAJOR	CUSTOM	MINOR
R-0	R-1	R-0	R-0

LEGEND: R = Read only; -*n* = value after reset

Table 11-4. MPU Revision ID Register (MPUREV) Field Descriptions

Bit	Field	Value	Description
31-30	SCHEME	1	Identification scheme.
29-28	Reserved	0	Reserved. Reads return 0.
27-16	FUNC	A0Ch	Indicates functionally equivalent module family. This value is dedicated to Hercules family from other general Texas Instruments MCU or MPU family.
15-11	RTL	0	RTL version number.
10-8	MAJOR	1	Major revision number.
7-6	CUSTOM	0	Indicates device-specific implementation.
5-0	MINOR	0	Minor revision number.

11.4.2 MPU Lock Register (MPULOCK)

Figure 11-5. MPU Lock Register (MPULOCK) [offset = 04h]

31 16
Reserved
R-0

15 4	3 0
Reserved	LOCK
R-0	R/WP-0

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privileged mode only; -*n* = value after reset

Table 11-5. MPU Lock Register (MPULOCK) Field Descriptions

Bit	Field	Value	Description
31-4	Reserved	0	Reserved. Reads return 0.
3-0	LOCK		MPU Register Lock Key. Lock feature prevents unintentional updates to MPU registers. Writes to registers other than MPUERRSTAT is possible only when MPU is unlocked. This field is updated only if the write data is 5h or Ah. Register writes are ignored for all other values of write data. A built-in correction logic detects single bit soft error on this field and corrects the value in the next cycle. Functionality and register read data remain the same during the correction cycle. **Read:** Returns current value of LOCK bits.
			Write in Privilege:
		5h	Writes to other MPU registers are blocked.
		Ah	Writes to other MPU registers are allowed.
		All other values	Reserved. The bits remain unchanged.

11.4.3 MPU Diagnostics Control Register (MPUDIAGCTRL)

Figure 11-6. MPU Diagnostics Control Register (MPUDIAGCTRL) [offset = 08h]

31-24
Reserved
R-0

23-19	18	17	16
Reserved	U_P	R_W	INT_EXT
R-0	R/WP-0	R/WP-0	R/WP-0

15-8	7-4	3-0
Reserved	DIAGKEY	Reserved
R-0	R/WP-5h	R-0

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privileged mode only; -*n* = value after reset

Table 11-6. MPU Diagnostics Control Register (MPUDIAGCTRL) Field Descriptions

Bit	Field	Value	Description
31-19	Reserved	0	Reserved. Reads return 0.
18	U_P		User/Privilege transaction in internal diagnostic mode. This field is used only in internal diagnostic mode. **Read:** Returns the current value of U_P.
			Write in Privilege:
		0	For access permission checks, treat the transaction as user mode access.
		1	For access permission checks, treat the transaction as privilege mode access.
17	R_W		Read/Write transaction in internal diagnostic mode. This field is used only in internal diagnostic mode. **Read:** Returns the current value of R_W.
			Write in Privilege:
		0	For access permission checks, treat the transaction as read.
		1	For access permission checks, treat the transaction as write.
16	INT_EXT		Internal/External diagnostic mode. **Read:** Returns the current value of INT_EXT.
			Write in Privilege:
		0	Enable internal diagnostic mode.
		1	Enable external diagnostic mode.
15-8	Reserved	0	Reserved. Reads return 0.
7-4	DIAGKEY		Diagnostics mode key. This is the key for enabling diagnostics mode. All other diagnostic configuration fields must be programmed before enabling this key. Diagnostic mode is entered by writing Ah to this key. Entering or exiting the diagnostic mode automatically clears the MPUERRSTAT and MPUERRADDR registers. This field is updated only if the write data is 5h or Ah. Register writes are ignored for all other values of write data. A built-in correction logic detects single bit soft error on this field and corrects the value in the next cycle. Functionality and register read data remain the same during the correction cycle. **Read:** Returns the current value of DIAGKEY.
			Write in Privilege:
		5h	Diagnostics mode is disabled.
		Ah	Diagnostics mode is enabled.
		All other values	Reserved. The bits remain unchanged.
3-0	Reserved	0	Reserved. Reads return 0.

11.4.4 MPU Diagnostic Address Register (MPUDIAGADDR)

Figure 11-7. MPU Diagnostic Address Register (MPUDIAGADDR) [offset = 0Ch]

31 0
DIAG ADDRESS
R/WP-0

LEGEND: R/W = Read/Write; WP = Write in privileged mode only; -*n* = value after reset

Table 11-7. MPU Diagnostic Address Register (MPUDIAGADDR) Field Descriptions

Bit	Field	Description
31-0	DIAG ADDRESS	Diagnostic address. This register is used in diagnostic mode. **Read:** Returns the current value of diagnostic address. **Write in Privilege:** Address to be used for diagnostic mode to check the address comparator logic.

11.4.5 MPU Error Status Register (MPUERRSTAT)

Figure 11-8. MPU Error Status Register (MPUERRSTAT) [offset = 10h]

31 29	28	27	26	25	24
Reserved	RERR	WERR	BGERR	APERR	Reserved
R-0	R-0	R-0	R-0	R-0	R-0

23 19	18 16
Reserved	REGION
R-0	R-0

15 14	13 8	7 1	0
Reserved	MASTERID	Reserved	ERRFLAG
R-0	R-0	R-0	R/W1CP-0

LEGEND: R/W = Read/Write; R = Read only; W1CP = Write 1 to clear in privilege mode only; -*n* = value after reset

Table 11-8. MPU Error Status Register (MPUERRSTAT) Field Descriptions

Bit	Field	Value	Description
31-29	Reserved	0	Reserved. Reads return 0.
28	RERR		Read Error. This field is valid only when the APERR or BGERR bit is 1. This field is read only and is automatically reset by clearing the ERRFLAG bit. This field is not updated when the ERRFLAG bit is set. Writes have no effect.
		0	MPU compare fail did not occur on a read access.
		1	MPU compare fail occurred on a read access.
27	WERR		Write Error. This field is valid only when the APERR or BGERR bit is 1. This field is read only and is automatically reset by clearing the ERRFLAG bit. This field is not updated when the ERRFLAG bit is set. Writes have no effect.
		0	MPU compare fail did not occur on a write access.
		1	MPU compare fail occurred on a write access.
26	BGERR		Background Error. This field is read only and is automatically reset by clearing the ERRFLAG bit. This field is not updated when the ERRFLAG bit is set. Writes have no effect.
		0	There was no memory access to addresses that are outside all the enabled MPU regions.
		1	MPU compare fail generated because of access to an address that is outside all the enabled MPU regions.

Table 11-8. MPU Error Status Register (MPUERRSTAT) Field Descriptions (continued)

Bit	Field	Value	Description
25	APERR		Access Permission Error. This field is read only and is automatically reset by clearing the ERRFLAG bit. This field is not updated when the ERRFLAG bit is set. Writes have no effect.
		0	Access permission violation did not occur in any of the enabled MPU regions.
		1	MPU compare fail generated because of access permission violation in one of the enabled MPU regions.
24-19	Reserved	0	Reserved. Reads return 0.
18-16	REGION		Region. This field is valid only when the APERR bit is 1. This field indicates the highest priority MPU region for which an access permission error was detected. This field is read only and is automatically reset by clearing the ERRFLAG bit. This field is not updated when the ERRFLAG bit is set. Writes have no effect.
		0	MPU compare fail generated because of access permission violation in region-0.
		1h	MPU compare fail generated because of access permission violation in region-1.
		2h	MPU compare fail generated because of access permission violation in region-2.
		3h	MPU compare fail generated because of access permission violation in region-3.
		4h	MPU compare fail generated because of access permission violation in region-4.
		5h	MPU compare fail generated because of access permission violation in region-5.
		6h	MPU compare fail generated because of access permission violation in region-6.
		7h	MPU compare fail generated because of access permission violation in region-7.
15-14	Reserved	0	Reserved. Reads return 0.
13-8	MASTERID		Master ID for MPU compare fail. This field is valid only when APERR or BGERR bit is 1. This field is read only and is automatically reset by clearing the ERRFLAG bit. This field is not updated when the ERRFLAG bit is set. Writes have no effect. Shows the master ID for the first transaction that resulted in a compare fail. Master ID is taken from MReqInfo[8:3] bits.
7-1	Reserved	0	Reserved. Reads return 0.
0	ERRFLAG		MPU compare error flag.
			Read:
		0	No MPU compare fail was detected.
		1	At least one MPU compare fail was detected.
			Write in Privilege:
		0	Reserved. The bit remains unchanged.
		1	Clears the bit.

11.4.6 MPU Error Address Register (MPUERRADDR)

Figure 11-9. MPU Error Address Register (MPUERRADDR) [offset = 14h]

31 … 0
COMPARE FAIL ADDRESS
R-0

LEGEND: R = Read only; -*n* = value after reset

Table 11-9. MPU Error Address Register (MPUERRADDR) Field Descriptions

Bit	Field	Description
31-0	COMPARE FAIL ADDRESS	Address for MPU compare fail. This field is valid only when the ERRFLAG bit in the MPU error status register (MPUERRSTAT) is set and the APERR or BGERR bit in MPUERRSTAT register is 1. This field is read only and is automatically reset by clearing the ERRFLAG bit in MPUERRSTAT register. This field is not updated when the ERRFLAG bit is set. Once the ERRFLAG bit is cleared, this field gets updated for the next MPU compare fail after clearing the flag. Writes have no effect. Shows the address for the first transaction that resulted in a compare fail.

11.4.7 MPU Control Register 1 (MPUCTRL1)

Figure 11-10. MPU Control Register 1 (MPUCTRL1) [offset = 20h]

31 … 16
Reserved
R-0

15 … 4	3 … 0
Reserved	MPUENA
R-0	R/WP-5h

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privileged mode only; -*n* = value after reset

Table 11-10. MPU Control Register 1 (MPUCTRL1) Field Descriptions

Bit	Field	Value	Description
31-4	Reserved	0	Reserved. Reads return 0.
3-0	MPUENA		MPU Enable Key. This is the key for enabling memory protection. This field is updated only if the write data is 5h or Ah. Register writes are ignored for all other values of write data. All other configuration registers must be programmed before enabling the MPU. A built-in correction logic detects single bit soft error on this field and corrects the value in the next cycle. Functionality and register read data remain the same during the correction cycle. **Read:** Returns current value of MPUENA.
			Write in Privilege:
		5h	Memory protection is disabled.
		Ah	Memory protection is enabled.
		All other values	Reserved. The bits remain unchanged.

11.4.8 MPU Control Register 2 (MPUCTRL2)

Figure 11-11. MPU Control Register 2 (MPUCTRL2) [offset = 24h]

31	16
Reserved	
R-0	

15	4	3	0
Reserved		ERRENA	
R-0		R/WP-5h	

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privileged mode only; -*n* = value after reset

Table 11-11. MPU Control Register 2 (MPUCTRL2) Field Descriptions

Bit	Field	Value	Description
31-4	Reserved	0	Reserved. Reads return 0.
3-0	ERRENA		MPU Error Pulse Enable. This is the key for enabling ERROR pulse output generation for the Error Signaling Module. This field is updated only if the write data is 5h or Ah. Register writes are ignored for all other values of write data. A built-in correction logic detects single bit soft error on this field and corrects the value in the next cycle. Functionality and register read data remain the same during the correction cycle. **Read:** Returns current value of ERRENA.
			Write in Privilege:
		5h	Error pulse output to ESM is disabled.
		Ah	Error pulse output to ESM is enabled.
		All other values	Reserved. The bits remain unchanged.

11.4.9 MPU Type Register (MPUTYPE)

Figure 11-12. MPU Type Register (MPUTYPE) [offset = 2Ch]

31	16
Reserved	
R-0	

15	8	7	0
NUMREG		Reserved	
R-x		R-0	

LEGEND: R = Read only; -*n* = value after reset; -*x* = value is implementation defined

Table 11-12. MPU Type Register (MPUTYPE) Field Descriptions

Bit	Field	Value	Description
31-16	Reserved	0	Reserved. Reads return 0.
15-8	NUMREG		Number of MPU Regions. Indicates the number of implemented MPU regions.
		0	Reserved
		1h	1 MPU region is implemented.
		2h	2 MPU regions are implemented.
		3h	3 MPU regions are implemented.
		4h	4 MPU regions are implemented.
		5h	5 MPU regions are implemented.
		6h	6 MPU regions are implemented.
		7h	7 MPU regions are implemented.
		8h	8 MPU regions are implemented.
		All other values	Reserved
7-0	Reserved	0	Reserved. Reads return 0.

11.4.10 MPU Region Base Address Register (MPUREGBASE)

NOTE: MPUREGBASE0-7 registers are memory-mapped to the same address. Which region register is selected for read/write access is decided by the REGION field in the MPU region number register (MPUREGNUM).

Figure 11-13. MPU Region Base Address Register (MPUREGBASE) [offset = 30h]

31-16
BASE_ADDRESS
R/WP-0

15-5	4-0
BASE_ADDRESS	Reserved
R/WP-0	R-0

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privileged mode only; -*n* = value after reset

Table 11-13. MPU Region Base Address Register (MPUREGBASE) Field Descriptions

Bit	Field	Value	Description
3-0	BASE_ADDRESS		Base address. Defines the base address for an MPU region. **Read:** Returns current value of base address. **Write in Privilege:** Defines the base address for an MPU region.
4-0	Reserved	0	Reserved. Reads return 0.

11.4.11 MPU Region Size and Enable Register (MPUREGSENA)

NOTE: MPUREGSENA0-7 registers are memory-mapped to the same address. Which region register is selected for read/write access is decided by the REGION field in the MPU region number register (MPUREGNUM).

Figure 11-14. MPU Region Size and Enable Register (MPUREGSENA) [offset = 34h]

31-16
Reserved
R-0

15-6	5-1	0
Reserved	REG_SIZE	REGENA
R-0	R/WP-1Fh	R/WP-0

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privileged mode only; -*n* = value after reset

Table 11-14. MPU Region Size and Enable Register (MPUREGSENA) Field Descriptions

Bit	Field	Value	Description
31-6	Reserved	0	Reserved. Reads return 0.

Table 11-14. MPU Region Size and Enable Register (MPUREGSENA) Field Descriptions (continued)

Bit	Field	Value	Description
5-1	REG_SIZE		MPU Region size. This field determines the size of an MPU region. **Read:** Returns current value of REG_SIZE.
			Write in Privilege: Defines the size of an MPU region.
		0-3h	Reserved
		4h	32 bytes
		5h	64 bytes
		6h	128 bytes
		7h	256 bytes
		8h	512 bytes
		9h	1 KB
		Ah	2 KB
		Bh	4 KB
		Ch	8 KB
		Dh	16 KB
		Eh	32 KB
		Fh	64 KB
		10h	128 KB
		11h	256 KB
		12h	512 KB
		13h	1 MB
		14h	2 MB
		15h	4 MB
		16h	8 MB
		17h	16 MB
		18h	32 MB
		19h	64 MB
		1Ah	128 MB
		1Bh	256 MB
		1Ch	512 MB
		1Dh	1 GB
		1Eh	2 GB
		1Fh	4 GB
0	REGENA		MPU Region Enable. This is the register bit for enabling an MPU region. **Read:**
		0	MPU region is disabled.
		1	MPU region is enabled.
			Write in Privilege:
		0	Disable MPU region.
		1	Enable MPU region.

11.4.12 MPU Region Access Control Register (MPUREGACR)

NOTE: MPUREGACR0-7 registers are memory-mapped to the same address. Which region register is selected for read/write access is decided by the REGION field in the MPU region number register (MPUREGNUM).

Figure 11-15. MPU Region Access Control Register (MPUREGACR) [offset = 38h]

31 — 16
Reserved
R-0

15 — 11	10 — 8	7 — 0
Reserved	AP	Reserved
R-0	R/WP-0	R-0

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privileged mode only; -*n* = value after reset

Table 11-15. MPU Region Access Control Register (MPUREGACR) Field Descriptions

Bit	Field	Value	Description
31-11	Reserved	0	Reserved. Reads return 0.
10-8	AP		MPU Region Access Permission. This field determines the access permission for memory accesses to addresses that are in an MPU region. **Read:** Returns current value of AP.
			Write in Privilege: Defines access permissions.
		0	No access.
		1h	Read/write in privileged mode; No access in user mode.
		2h	Read/write in privileged mode; Read only in user mode.
		3h	Read/write.
		4h	No access.
		5h	Read only in privileged mode; No access in user mode.
		6h	Read only.
		7h	No access.
7-0	Reserved	0	Reserved. Reads return 0.

11.4.13 MPU Region Number Register (MPUREGNUM)

NOTE: MPUREGBASE0-7, MPUREGSENA0-7, MPUREGACR0-7, MPUREGAM0-7, MPUREGTA0-7, and MPUREGMT0-7 registers are memory-mapped to just six different addresses. Which region register is selected for read/write access is decided by the REGION field in the MPU region number register (MPUREGNUM).

Figure 11-16. MPU Region Number Register (MPUREGNUM) [offset = 3Ch]

31–16
Reserved
R-0

15–4	3–0
Reserved	REGION
R-0	R/WP-0

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privileged mode only; -*n* = value after reset

Table 11-16. MPU Region Number Register (MPUREGNUM) Field Descriptions

Bit	Field	Value	Description
31-4	Reserved	0	Reserved. Reads return 0.
3-0	REGION		MPU Region Number. This field determines which MPU region registers are accessed. Writing this register with a value greater than or equal to the number of implemented MPU regions (indicated by MPUTYPE register) does not affect the NMPU functionality. Behavior will be same as that of reserved space. **Read:** Returns current value of REGION.
			Write in Privilege:
		0	Access MPU region 0 registers.
		1h	Access MPU region 1 registers.
		2h	Access MPU region 2 registers.
		3h	Access MPU region 3 registers.
		4h	Access MPU region 4 registers.
		5h	Access MPU region 5 registers.
		6h	Access MPU region 6 registers.
		7h	Access MPU region 7 registers.

Chapter 12

SPNU563A – March 2018

Error Profiling Controller (EPC)

This chapter describes overall functionality and how to use the Error Profiling Controller (EPC).

Topic **Page**

12.1 Overview

The EPC is used as a diagnostic for functional safety purposes.

The primary goal of this module is to provide a unified correctable ECC error (single-bit ECC fault) profiling capability and error address cache on ECC failures in system bus memory slaves like Flash, FEE, and SRAM.

The secondary goal of this module is to provide an ECC error reporting capability for bus masters which are not natively built to manage ECC error like the interconnect.

The EPC can not distinguish between memory failure and interconnect failure. A fault address may not always mean there is a true issue in the memory. EPC captures both the correctable and uncorrectable information.

The EPC supports the following features:

- Traps the correctable and uncorrectable faults from RAM, CPU, and Interconnect.
- For correctable fault, EPC will keep track of unique addresses through the usage of Content Address Memory (CAM).
- Allow CPU accessing CAM to set or clear any CAM entry index and/or content during execution run time as well as diagnosing the CAM.
- Trigger error event to Error Signaling Module (ESM) and keep track of error in status register for user query.

12.2 Module Operation

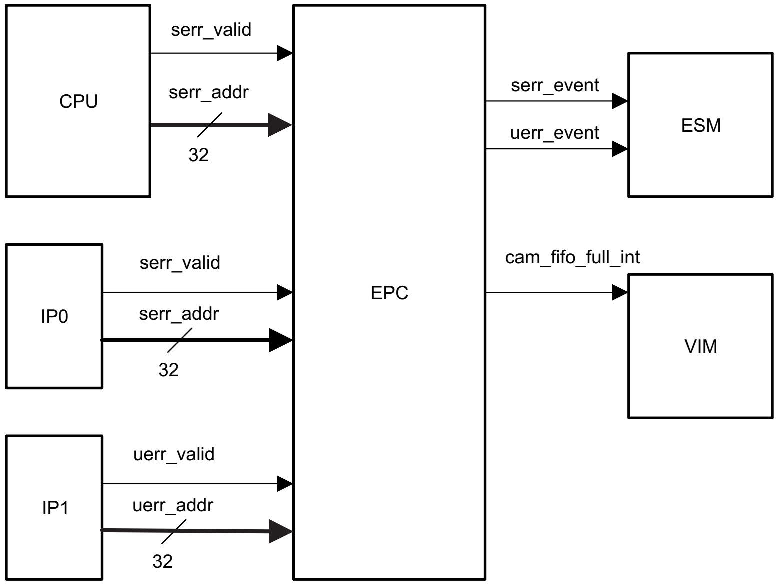
Figure 12-1 shows the typical usage of EPC in device architecture. In the EPC chapter, the CPU, RAM, or Interconnect is referred to as IP. The Error Profiling Module section in the Architecture chapter indicates which IP correctable and uncorrectable event are hooked up to EPC. Each IP can provide either or both correctable and uncorrectable fault event to EPC. The EPC chapter will mention IP correctable or uncorrectable fault event with generic description of how EPC process these fault events.

EPC captures the uncorrectable address from IP that are not natively built to manage ECC error like interconnect and triggers uerr_event to ESM. See Section 12.2.1 for more details.

EPC performs error profiling on the correctable fault and trigger serr_event to ESM if the address of the correctable fault is not part of the CAM and SERRENA control bits are set to enable values. Detail description of error profiling definition is described in Section 12.2.2. Each single fault correctable IP has a FIFO to buffer correctable fault address input. Following is the behavior of FIFO and CAM operation:

1. If FIFO overflow happens on a particular ECC correctable IP, EPC will set the corresponding FIFO Overflow Bit in the Overflow Status Register (OVRFLWSTAT) and trigger serr_event.
2. If any of the FIFOs is full (any FIFOFULLSTAT(x) is set), EPC will trigger the cam_fifo_full_int port if CAM/FIFO full interrupt enable (EPCCNTRL(24)) is set.
3. If CAM indexes are all occupied, EPC will set the CAM Full Bit in EPCERRSTAT register and trigger the cam_fifo_full_int port if CAM/FIFO full interrupt enable is set.
4. If CAM overflow happens, EPC will set CAM overflow status bit (cam ovflw) in EPCERRSTAT register.
5. You can access CAM content and CAM index during functional and diagnostic run time.

Figure 12-1. EPC System Block Diagram



12.2.1 Uncorrectable Fault Operation

EPC will capture full 32-bit addresses of uncorrectable fault from interconnect and RAM IP modules to UERRADDR_(0,1) registers and set the corresponding uncorrectable status bit in UERRSTAT register. The first uncorrectable address from RAM and interconnect IP will be captured to the corresponding UERRADDR_(0,1) and frozen until CPU in privilege mode write clears to the corresponding UERRSTAT bit.

Whenever any status bit in ERRSTAT just got set by the presence of a new uncorrectable fault, EPC will trigger uncorrectable fault event to ESM. The bits in UERRSTAT can only be cleared by device power on reset or CPU write clears in privilege mode.

12.2.2 Correctable Fault Operation

12.2.2.1 Functional Mode

CPU, Interconnect, and RAM IP can trigger correctable fault event to EPC. The EPC provides a 4-entry FIFO to each of these IP(s) to capture correctable event and its 64-bit aligned addresses.

A FIFO full condition happens when all 4 entries of a particular FIFO are occupied. In this case, the corresponding FIFO full status bit of the IP will be set in FIFOFULLSTAT register. An interrupt will be generated if CAMFIFOFULLENA bit is set in EPCCNTRL register.

A FIFO overflow can happen when all entries are occupied and there is a new correctable fault event just arrives to the same FIFO. In this case, the new correctable fault event and address will be discarded, but the overflow bit remained to be set. If the SERRENA bits in EPCCNTRL register are enabled, the single-bit error correctable fault event will be triggered to ESM.

A FIFO full or overflow interrupt indicates to you that there is an abnormal condition on the number of correctable faults happening to the particular IP. It is up to application software to handle this situation by either putting the system in safe stage if the IP causing full or overflow interrupt happens to be a critical IP in safety application or doing extra diagnostic of the corresponding IP memory during diagnostic time. You can write clears the corresponding FIFO full or overflow status bit in privilege mode.

The 64-bit aligned address of the correctable fault from each IP FIFO is sent to the CAM to check if the correctable fault is unique or repetitive. If it is a repetitive address for the correctable fault, then the correctable fault and its address are discarded and no further indication to the CPU. If it is a unique address, then the address will be remembered in the CAM content and CAM index will be set to occupied. It is software configurable to raise an error event to ESM if SERRENA bits in EPCCNTRL are enable.

If all CAM indexes are occupied, EPC will set the CAM full status bit in EPCERRSTAT register and trigger an interrupt to VIM if CAMFIFOFULLENA bit in EPCCNTRL register is set. You can inspect CAM and set its indexes to available.

If all CAM indexes are occupied and there is a new correctable fault event to be checked, the EPC will set the CAM overflow status bit in EPCERRSTAT register and trigger an error event to ESM if correctable error event enable bits are set in EPCCNTRL register. The CAM content and index are not updated when CAM overflow happens.

Reading a CAM index value of 5h indicates the CAM entry is available and reading a value of Ah indicates the CAM entry is occupied. You can also inspect the number of CAM indexes that are still in available state by reading the CAMAVAILSTAT register.

CAM content and index can only be updated in privilege mode.

In functional mode, CPU can only set CAM index to available state but not occupied state. Occupied state setting by CPU will be ignored.

CPU can also update the CAM content. In this case, once the CAM content is updated, the CAM index will auto set to occupied state, but there is no correctable error event generation to ESM. This is mainly used as a way to avoid correctable error event generation for hard fault on single (correctable) bit error address in functional mode. An example usage would be: Assume address 0x0800_0000 has a hard fault single-bit error in the RAM. You can write 0x0800_0000 to the EPC CAM content. This write will update the corresponding CAM index to “occupied” by EPC hardware. You can avoid EPC generation of single-bit error event every time the CPU accessing address 0x0800_0000.

12.2.2.2 Diagnostic Mode

EPC allows you to diagnose the CAM content, CAM index, and correctable event generation to ensure the CAM operates correctly and to avoid latent fault.

You need to set DIAG_ENA_KEY in EPCCNTRL register to Ah to enter diagnostic mode.

Once in diagnostic mode, you can change any CAM index to available or occupied state. Setting all CAM indexes to occupied will result in CAM full status bit to be set in EPCCNTRL register. In this case, EPC will generate the CAM full interrupt if CAM/FIFO full interrupt enable is set. The NUMCAMAVAIL bits of CAMAVAILSTAT will also reflect the number of CAM index available when you change the CAM index values between available and occupied.

Writes to CAM content will also set the corresponding CAM index to occupied and trigger correctable error event to ESM. This is done to test the signal chain in CAM content update for unique address and triggering correctable event in functional mode.

12.3 How to Use EPC

12.3.1 Functional Mode

Following steps are the recommended sequences to initialize EPC:

1. Set up correct values for SERRENA and CAMFIFO FULL ENA bits in EPCCNTRL. Setting SERRENA will enable correctable error event generation to ESM. Setting CAMFIFO FULL ENA will enable CAM full or FIFO full interrupt to the CPU. You need to set these values in according to their safety application requirement.
2. Read CAMAVAILSTAT to ensure that all CAM indexes are available after system reset.

On CAM or FIFO full interrupt, following sequences are recommended to query the EPC:

1. Read EPCERRSTAT and FIFOFULLSTAT to determine if this is CAM or FIFO full.
   a. If it is FIFO full, the FIFOFULLSTAT indicates which IP FIFO is full so you can make a decision on whether to put the system in safe state if the particular IP happens to be a critical IP in safety application or doing extra diagnostic of the corresponding IP RAM during diagnostic time. Clear the FIFO full by write clear to the FIFOFULLSTAT register.
   b. If it is CAM full, you need to read CAM content to find out if most of the correctable fault happens to be in the same IP or scatter among IP in order to take decision on whether to put the system in safe state or decides to run certain RAM test during diagnostic time. You can also keep track of the correctable fault in system RAM in order to clear the CAM index to avoid CAM overflow condition.

On correctable error event or CAM overflow or FIFO overflow from ESM interrupt CPU, following sequences are recommended to query the EPC:

1. Read EPCERRSTAT and OVRFLWSTAT registers to determine if this is CAM or FIFO overflow or a registration of new correctable fault event in CAM.
   a. If it is FIFO overflow, the OVRFLWSTAT indicates which IP FIFO has overflow so you can make decision on whether to put the system in safe state if the particular IP happens to be a critical IP in safety application or doing extra diagnostic of the corresponding IP RAM during diagnostic time. Clear the FIFO overflow by write clear to the OVRFLWSTAT register.
   b. If it is CAM overflow, that means you do not service the CAM full interrupt in time. You need to read CAM content to find out if most of the correctable fault happens to be in the same IP or scatter among IP in order to take decision on whether to put the system in safe state or decides to run certain RAM test during diagnostic time. You can also keep track of the correctable fault in system RAM in order to clear the CAM index to avoid CAM overflow condition.
   c. If it is none of the two above cases, then it is a new correctable fault event register on CAM. You can read the CAM index registers and CAM content registers to determine which IP RAM or RAM location that has the correctable fault and does a quick diagnose of that RAM location by backing up location content, write and read back new RAM value. If it is a transient fault, restores RAM backup data and clear the CAM index. Otherwise, mark it as permanent fault by not clearing the index to available so that it does not generate correctable error event again.

On uncorrectable error event, following sequences are recommended to query the EPC:

1. Read UERRSTAT register to determine which IP_(n) causing uncorrectable fault.
2. Read the corresponding UERR_ADDR_(n) to determine the location of the fault.
3. Diagnose the corresponding location to determine if this is a permanent or transient fault. Depending on the criticality of this uncorrectable fault in safety application, it is up to you to bring the system to safe state or not.

12.3.2 CAM Diagnostic Mode

In order to test the CAM logic and error event generation functionality, you need to diagnose the CAM at diagnostic time in their control loop.

Following sequences are recommended to diagnose the CAM and error event generation:

1. Configure EPC in diagnostic mode by setting the DIAG_ENA_KEY to Ah in EPCCNTRL register.
2. Backing up the CAM content and CAM index to system RAM.
3. Change CAM index to available state from occupied state and vice versa and check the number of CAM available status correctly reflected in CAMAVAILSTAT register as well as the CAM index registers correctly reflecting the new state.
4. Write to CAM content of any available index and should observe a correctable error event set in ESM as well as CAM index set to occupied.
5. Restore the CAM content and CAM index values.
6. Exit diagnostic mode by writing 5h to DIAG_ENA_KEY.

12.4 EPC Control Registers

The error profiling controller registers listed in Table 12-1 are accessed through the system module register space in the Cortex-R5F CPUs memory-map. All registers are 32-bit wide and are located on a 32-bit boundary. Reads and writes to registers are supported in 8-, 16-, and 32-bit accesses. The base address for the control registers is FFFF 0C00h.

Table 12-1. EPC Control Registers

Offset	Acronym	Register Description	Section
00h	EPCREVID	EPC REVID Register	Section 12.4.1
04h	EPCCNTRL	EPC Control Register	Section 12.4.2
08h	UERRSTAT	Uncorrectable Error Status Register	Section 12.4.3
0Ch	EPCERRSTAT	EPC Error Status Register	Section 12.4.4
10h	FIFOFULLSTAT	FIFO Full Status Register	Section 12.4.5
14h	OVRFLWSTAT	IP Interface FIFO Overflow Status Register	Section 12.4.6
18h	CAMAVAILSTAT	CAM Index Available Status Register	Section 12.4.7
20h-24h	UERRADDR	Uncorrectable Error Address Registers	Section 12.4.8
A0h-11Ch	CAM_CONTENT	CAM Content Update Registers	Section 12.4.9
200h	CAM_INDEX0	CAM Index Register 0	Section 12.4.10
204h	CAM_INDEX1	CAM Index Register 1	Section 12.4.10
208h	CAM_INDEX2	CAM Index Register 2	Section 12.4.10
20Ch	CAM_INDEX3	CAM Index Register 3	Section 12.4.10
210h	CAM_INDEX4	CAM Index Register 4	Section 12.4.10
214h	CAM_INDEX5	CAM Index Register 5	Section 12.4.10
218h	CAM_INDEX6	CAM Index Register 6	Section 12.4.10
21Ch	CAM_INDEX7	CAM Index Register 7	Section 12.4.10

12.4.1 EPC REVID Register (EPCREVID)

Figure 12-2. EPC REVID Register (EPCREVID) (offset = 00h)

31 30	29 28	27 16
SCHEME	Reserved	FUNC
R-1	R-0	R-A0Ah

15 11	10 8	7 6	5 0
RTL	MAJOR	CUSTOM	MINOR
R-0	R-0	R-0	R-0

LEGEND: R = Read only; -*n* = value after synchronous reset on system reset

Table 12-2. EPC REVID Register (EPCREVID) Field Descriptions

Bit	Field	Value	Description
31-30	SCHEME	1	Identification scheme.
29-28	Reserved	0	Reserved. Reads return 0.
27-16	FUNC	A0Ah	Indicates functionally equivalent module family.
15-11	RTL	0	RTL version number.
10-8	MAJOR	0	Major revision number.
7-6	CUSTOM	0	Indicates device-specific implementation.
5-0	MINOR	0	Minor revision number.

12.4.2 EPC Control Register (EPCCNTRL)

Figure 12-3. EPC Control Register (EPCCNTRL) (offset = 04h)

31 25	24	23 16
Reserved	CAM/FIFO_ FULL_ENA	Reserved
R-0	R/WP-0	R-0

15 12	11 8	7 4	3 0
Reserved	DIA_ENA_KEY	Reserved	SERRENA
R-0	R/WP-5h	R-0	R/WP-5h

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privileged mode only; -*n* = value after synchronous reset on system reset

Table 12-3. EPC Control Register (EPCCNTRL) Field Descriptions

Bit	Field	Value	Description
31-25	Reserved	0	Reserved. Reads return 0.
24	CAM/FIFO_FULL_ENA		CAM or FIFO full interrupt enable. If this bit is set and CAM is full, CAM Full Interrupt is generated.
			Read:
		0	CAM/FIFO full interrupt is disabled.
		1	CAM/FIFO full interrupt is enabled.
			Write in Privilege:
		0	Disable CAM/FIFO full interrupt.
		1	Enable CAM/FIFO full interrupt.
23-12	Reserved	0	Reserved. Reads return 0.
11-8	DIA_ENA_KEY		CAM diagnostic enable key. These bits (when enabled) allow the CPU to access the CAM content to clear or set any entry (CAM index) or write any pattern to CAM content. Internal RTL will implement self-correction logic to avoid single bit flipping.
			Read:
		5h	CAM diagnostic is disabled.
		Ah	CAM diagnostic is enabled.
		All other values	Reserved
			Write in Privilege:
		5h	CAM diagnostic is disabled.
		Ah	CAM diagnostic is enabled.
		All other values	Reserved
7-4	Reserved	0	Reserved. Reads return 0.
3-0	SERRENA		Single (correctable) bit error event enable. These bits (when enable) cause EPC to generate the serr_event if there is a correctable ECC fault address arrives from one of the EPC-IP interface and the CAM has an empty entry. These bits also allow EPC to generate the serr_event if there is a correctable ECC fault address arrives from one of the EPC-IP interface and the CAM is full. In this case, CAM FULL status bit is set in EPCERRSTAT. Internal RTL will implement self-correction logic to avoid single bit flipping.
			Read:
		5h	serr_event generation is disabled.
		Ah	serr_event generation is enabled.
		All other values	Reserved
			Write in Privilege:
		5h	serr_event generation is disabled.
		Ah	serr_event generation is enabled.
		All other values	Reserved

12.4.3 Uncorrectable Error Status Register (UERRSTAT)

Figure 12-4. Uncorrectable Error Status Register (UERRSTAT) (offset = 08h)

31	16
Reserved	
R-0	

15	2	1	0
Reserved		UE1	UE0
R-0		R/W1CP-0	R/W1CP-0

LEGEND: R/W = Read/Write; R = Read only; W1CP = Write 1 to clear in privilege mode only; -*n* = value after asynchronous reset by power-on reset

Table 12-4. Uncorrectable Error Status Register (UERRSTAT) Field Descriptions

Bit	Field	Value	Description
31-2	Reserved	0	Reserved. Reads return 0.
1-0	UE*n*		Uncorrectable ECC Fault Status Bit for interface *n*. Each bit corresponds to one uncorrectable EPC-IP interface. If the IP triggers uncorrectable error, one of these bits gets set. Once it is set, it can only be cleared by power-on reset or CPU write-clear in privilege mode or by reading the corresponding UERRADDR register. Any of these bits set causes an uncorrectable error event (uerr_event) to be triggered to ESM. The number of implemented bits depends on the number of implemented EPC IP uncorrectable address ports. Unimplemented bits are reserved and are not writable. Reserved bits are read as 0. **Read:**
		0	Uncorrectable ECC fault status bit is not active for interface *n*.
		1	Uncorrectable ECC fault status bit is active for interface *n*.
			Write in Privilege:
		0	No effect.
		1	Clear this flag bit.

12.4.4 EPC Error Status Register (EPCERRSTAT)

Figure 12-5. EPC Error Status Register (EPCERRSTAT) (offset = 0Ch)

31 … 16
Reserved
R-0

15 … 3	2	1	0
Reserved	CAM_FULL	BUS_ERR	CAM_OVFLW
R-0	R/W1CP-0	R/W1CP-0	R/W1CP-0

LEGEND: R/W = Read/Write; R = Read only; W1CP = Write 1 to clear in privilege mode only; -*n* = value after asynchronous reset by power-on reset

Table 12-5. EPC Error Status Register (EPCERRSTAT) Field Descriptions

Bit	Field	Value	Description
31-3	Reserved	0	Reserved. Reads return 0.
2	CAM_FULL		CAM full status bit. This bit is set when CAM has no more available index available to accept new correctable address.
			Read:
		0	CAM is not full.
		1	CAM is full.
			Write in Privilege:
		0	No effect.
		1	Clear this flag bit.
1	BUS_ERR		MMR interface bus error status bit. This bit is set if MMR interface receives unsupported bus commands like ReadLink-WriteConditional.
			Read:
		0	No MMR bus error.
		1	MMR unsupported bus command is detected.
			Write in Privilege:
		0	No effect.
		1	Clear this flag bit.
0	CAM_OVFLW		CAM overflow status bit. CAM is full and there is another correctable address arrives.
			Read:
		0	No CAM overflow.
		1	CAM overflow is detected.
			Write in Privilege:
		0	No effect.
		1	Clear this flag bit.

12.4.5 FIFO Full Status Register (FIFOFULLSTAT)

Figure 12-6. FIFO Full Status Register (FIFOFULLSTAT) (offset = 10h)

31	16
Reserved	
R-0	

15	5	4	3	2	1	0
Reserved		FULL4	FULL3	FULL2	FULL1	FULL0
R-0		R/W1CP-0	R/W1CP-0	R/W1CP-0	R/W1CP-0	R/W1CP-0

LEGEND: R/W = Read/Write; R = Read only; W1CP = Write 1 to clear in privilege mode only; -*n* = value after asynchronous reset by power-on reset

Table 12-6. FIFO Full Status Register (FIFOFULLSTAT) Field Descriptions

Bit	Field	Value	Description
31-5	Reserved	0	Reserved. Reads return 0.
4-0	FULL*n*		FIFO interface *n* is full. If there is a FIFO full occurs on a particular interface, the corresponding bit is set. If any of these bits is set and the CAM/FIFO full ena (enabled) bits are set, EPC triggers cam_fifo_full_int. The number of implemented bits depends on the number of implemented EPC IP correctable address ports. Unimplemented bits are reserved and are not writable. Reserved bits are read as 0. **Read:**
		0	FIFO interface *n* is not full.
		1	FIFO interface *n* full occurred.
			Write in Privilege:
		0	No effect.
		1	Clear this flag bit.

12.4.6 IP Interface FIFO Overflow Status Register (OVRFLWSTAT)

Figure 12-7. IP Interface FIFO Overflow Status Register (OVRFLWSTAT) (offset = 14h)

31	16
Reserved	
R-0	

15 5	4	3	2	1	0
Reserved	OVFL4	OVFL3	OVFL2	OVFL1	OVFL0
R-0	R/W1CP-0	R/W1CP-0	R/W1CP-0	R/W1CP-0	R/W1CP-0

LEGEND: R/W = Read/Write; R = Read only; W1CP = Write 1 to clear in privilege mode only; -*n* = value after asynchronous reset by power-on reset

Table 12-7. IP Interface FIFO Overflow Status Register (OVRFLWSTAT) Field Descriptions

Bit	Field	Value	Description
31-5	Reserved	0	Reserved. Reads return 0.
4-0	OVFL*n*		Correctable EPC-IP interface *n* FIFO overflow. Each bit corresponds to one correctable EPC-IP interface FIFO status. If there is a FIFO overflow occurs, this bit is set. If any of these bits is set and the FIFO overflow interrupt enable bit is set, EPC triggers FIFO overflow interrupt. The number of implemented bits depends on the number of implemented EPC IP correctable address ports. Unimplemented bits are reserved and are not writable. Reserved bits are read as 0.
			Read:
		0	No FIFO overflow.
		1	FIFO overflow occurred.
			Write in Privilege:
		0	No effect.
		1	Clear this flag bit.

12.4.7 CAM Index Available Status Register (CAMAVAILSTAT)

Figure 12-8. CAM Index Available Status Register (CAMAVAILSTAT) (offset = 18h)

31	16
Reserved	
R-0	

15 6	5 0
Reserved	NUMCAMAVAIL
R-0	R-20h

LEGEND: R = Read only; -*n* = value after synchronous reset on system reset

Table 12-8. CAM Index Available Status Register (CAMAVAILSTAT) Field Descriptions

Bit	Field	Value	Description
31-6	Reserved	0	Reserved. Reads return 0.
5-0	NUMCAMAVAIL		Number of current available CAM index. These bits indicate (binary encoded value) the number of currently available CAM index.
		0	Reserved
		1h	1 CAM index is available.
		2h	2 CAM index is available.
		:	:
		20h	32 CAM index is available.

12.4.8 Uncorrectable Error Address Register n (UERR_ADDR)

Figure 12-9. Uncorrectable Error Address Register n (UERR_ADDR) (offset = 20h-24h)

<table>
<tr><td>31</td><td>0</td></tr>
<tr><td colspan="2">UERR_ADDR</td></tr>
<tr><td colspan="2">R-0</td></tr>
</table>

LEGEND: R = Read only; -*n* = value after asynchronous reset on power-on reset

Table 12-9. Uncorrectable Error Address Register n (UERR_ADDR) Field Descriptions

Bit	Field	Description
31-0	UERR_ADDR	Register *n* corresponds to uncorrectable port *n*. The number of uncorrectable ports is configured through the generic parameter: Number of EPC uncorrectable ports. This 32-bit register captures the uncorrectable address error from each EPC-IP uncorrectable interface. Once EPC-IP interface receives the uerr_valid_x, the corresponding address is captured and frozen. CPU read (privilege mode) of this address unfreezes this register. EMUDBG read access is non-intrusive (not unfreeze). Power-on reset, write-clear to the corresponding UERRSTAT bit or reading of the ERRADDR register also unfreezes this register for each interface. Unfreeze means that the register content can be updated whenever there is the next uncorrectable error address becomes active on this interface.

12.4.9 CAM Content Update Register n (CAM_CONTENT)

Figure 12-10. CAM Content Update Register n (CAM_CONTENT) (offset = A0h-11Ch)

<table>
<tr><td>31</td><td>16</td></tr>
<tr><td colspan="2">CAM_CONTENT</td></tr>
<tr><td colspan="2">R/WP-0</td></tr>
</table>

<table>
<tr><td>15</td><td>3</td><td>2</td><td>0</td></tr>
<tr><td colspan="2">CAM_CONTENT</td><td colspan="2">Reserved</td></tr>
<tr><td colspan="2">R/WP-0</td><td colspan="2">R-0</td></tr>
</table>

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privileged mode only; -*n* = value after synchronous reset on system reset

Table 12-10. CAM Content Update Register n (CAM_CONTENT) Field Descriptions

Bit	Field	Value	Description
31-3	CAM_CONTENT		CAM content register *n*. CPU writes to this field in functional or diagnostic mode. The write data is masked with byten and stored into CAM on each index. Address A0h corresponds to index 0; address 11Ch corresponds to index 31. The number of active registers changes depending on the number of CAM indexes available upon configuration during device integration.
2-0	Reserved	0	Reserved. Reads return 0.

12.4.10 CAM Index Registers (CAM_INDEX[0-7])

Figure 12-11. CAM Index Registers (CAM_INDEXn) (offset = 200h-21Ch)

31 28	27 24	23 20	19 16
Reserved	index n × 4 + 3	Reserved	index n × 4 + 2
R-0	R/WP-5h	R-0	R/WP-5h

15 12	11 8	7 4	3 0
Reserved	index n × 4 + 1	Reserved	index n × 4
R-0	R/WP-5h	R-0	R/WP-5h

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privileged mode only; -*n* = value after synchronous reset on system reset

Table 12-11. CAM Index Registers (CAM_INDEXn) Field Descriptions

Field	Value	Description
Reserved	0	Reserved. Reads return 0.
index n		Entry valid tag for index n. See Table 12-12. This is the 4-bit key that determines whether the current entry in CAM is occupied or not. A write (privilege mode) to A0h to 11Ch in diagnostic mode or functional mode does not only update the CAM content in corresponding index but also updates the corresponding bit field in the entry valid tags registers CAM_INDEXn. The index has a self correction mechanism as follows: • Key active if valid key = 1010 or 1011 or 1000 or 1110 or 0010 • Key inactive if valid key = 0101 or 0100 or 0111 or 1101 or 0001 **Read:**
	5h	Entry is clear and available for future CAM usage.
	Ah	Entry is occupied.
	All other values	Reserved
		Write in Diagnostic Mode:
	5h	Entry is cleared and available for future CAM usage.
	Ah	Entry is set and occupied.
	All other values	Reserved
		Write in Functional Mode:
	5h	Entry is cleared and available for future CAM usage.
	All other values	Reserved

Table 12-12. CAM Index Register n

		CAM Index Register Bits			
Address Offset	CAM Index Register	Bits 27-24	Bits 19-16	Bits 11-8	Bits 3-0
200h	CAM Index Register 0	index 3	index 2	index 1	index 0
204h	CAM Index Register 1	index 7	index 6	index 5	index 4
208h	CAM Index Register 2	index 11	index 10	index 9	index 8
20Ch	CAM Index Register 3	index 15	index 14	index 13	index 12
210h	CAM Index Register 4	index 19	index 18	index 17	index 16
214h	CAM Index Register 5	index 23	index 22	index 21	index 20
218h	CAM Index Register 6	index 27	index 26	index 25	index 24
21Ch	CAM Index Register 7	index 31	index 30	index 29	index 28

Chapter 13

SPNU563A – March 2018

CPU Compare Module for Cortex-R5F (CCM-R5F)

This chapter describes the CPU compare module for the ARM® Cortex®-R5F (CCM-R5F). This device implements two instances of the Cortex-R5F CPU that are running in lockstep to detect faults that may result in unsafe operating conditions. The CCM-R5F detects faults and signals them to an error signaling module (ESM).

NOTE: In general, the R5F term is used when referencing the Cortex-R5F CPU used in the Hercules family of devices; however, the floating-point functionality is a device-specific option and may not be included in some devices. Consult your device-specific datasheet to determine which core is included on your specific device being used.

Topic **Page**

13.1 Overview

Safety-critical applications require run-time detection of faults in critical components in the device such as the Central Processing Unit (CPU) and the Vectored Interrupt Controller Module (VIM). For this purpose, the CPU Compare Module for Cortex-R5F (CCM-R5F) compares the core bus outputs of two Cortex-R5F CPUs running in a 1oo1D (one-out-of-one, with diagnostics) lockstep configuration. This microcontroller also implements two VIM modules in 1oo1D (one-out-of-one, with diagnostic) lockstep configuration. Any difference in the core compare bus outputs of the CPUs or the VIMs is flagged as an error. For diagnostic purposes, the CCM-R5F also incorporates a self-test capability to allow for boot time checking of hardware faults within the CCM-R5F itself.

In addition to comparing the CPU's and VIM's outputs for fault detection during run-time, the CCM-R5F also incorporates two additional run-time diagnostic features.

The first additional measure is the Checker CPU Inactivity Monitor which will monitor the checker CPU's key bus signals to the interconnect. When the two CPUs are in lockstep configuration, several key bus signals from the checker CPU which would have indicated a valid bus transaction to the interconnect on the microcontroller will be monitored. A list of the signals to be monitored is provided in Table 13-5. These signals from the checker CPU are expected to be inactive. All transactions between the lockstep CPUs and the rest of the system should only go through the main CPU. Any signals which indicate activity will be flagged as an error.

The second feature is the Power Domain (PD) Inactivity Monitor. Similar to the Checker CPU Inactivity Monitor in concept, the Power Domain Inactivity Monitor will monitor key bus signals for bus masters residing in power domains which are turned off. When a power domain is turned off, the boundary of the power domain is isolated from the rest of the system. Bus signals which would have indicated a valid bus transaction onto the interconnect are monitored. Any signals which indicate active state will be flagged as an error.

13.1.1 Main Features

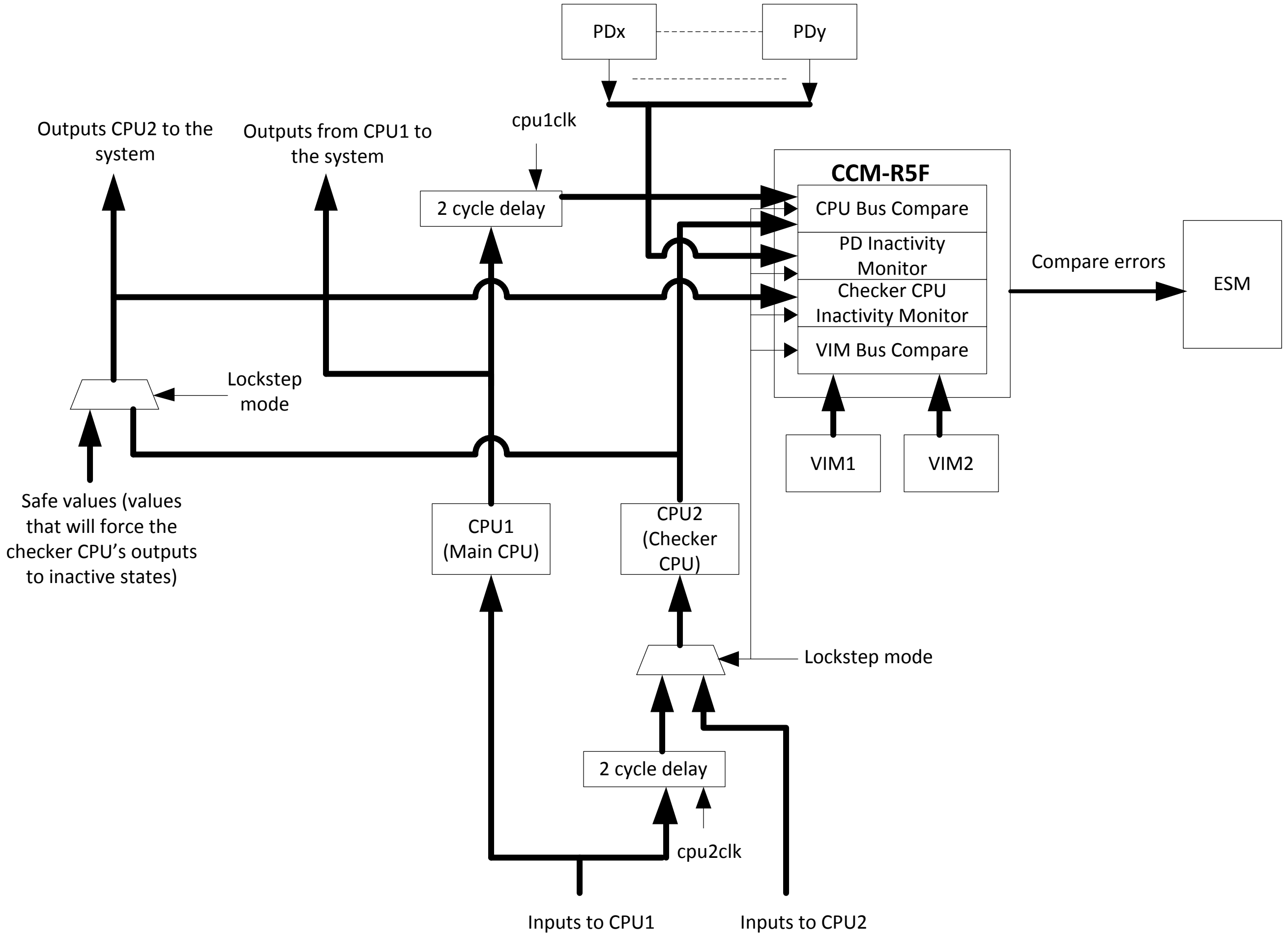
The main features of the CCM-R5F are:

- Run-time detection of faults
  - Run-time compare of CPU's outputs
  - Run-time compare of VIM's outputs
  - Run-time inactivity monitor on the checker CPU's bus signals to the interconnect
  - Run-time inactivity monitor on the power domains' bus signals to the interconnect
- self-test capability
- error forcing capability

13.1.2 Block Diagram

Figure 13-1 shows the interconnect diagram of the CCM-R5F with the two Cortex-R5F CPUs and the two VIMs. The core bus outputs of the CPUs are compared in the CCM-R5F. To avoid common mode impacts, the signals of the CPUs to be compared are temporally diverse. The output signals of the master CPU are delayed 2 cycles while the input signals of checker CPU are delayed 2 cycles. The two cycle delay strategy is also deployed between the two VIM modules. While in lockstep mode, the checker CPU's output signals to the system are clamped to inactive safe values. Key signals which would have indicated a valid bus transaction to the interconnect are monitored by the CCM-R5F. The same approach is used for the key power domains if inactive signals indicate that bus masters inside these power domains are asserting valid bus transactions.

Figure 13-1. Block Diagram



13.2 Module Operation

As described in Section 13.1, there are four different run-time diagnostics supported by the CCM-R5F. The CCM-R5F compares the core bus outputs of the master and checker Cortex-R5F CPUs on the microcontroller and signals an error on any mismatch. This comparison is started 6 CPU clock cycles after the CPU comes out of reset to ensure that CPU output signals have propagated to a known value after reset. Once comparison is started, the CCM module continues to monitor the outputs of the two CPUs without any software intervention. If an error is detected by the CCM-R5F, a software handler is necessary to implement the appropriate response to the error dependent on application needs. The module principles of operation are applicable to both the CPU output compare as described above as well as to the VIM output compare.

13.2.1 CPU/VIM Output Compare Diagnostic

CPU / VIM Output Compare Diagnostic can run in one of the following four operating modes:

1. Active compare lockstep mode
2. Self-test
3. Error forcing
4. Self-test error forcing

The operating mode can be selected by writing a dedicated key to the key register (MKEYx) of the corresponding diagnostic.

NOTE: MKEY1 and MKEY2 are used to select the operating mode for the CPU Output Compare Diagnostic and VIM Output Compare Diagnostic, respectively.

13.2.1.1 Active Compare lockstep Mode

This is the default mode on start-up. In lockstep mode, the bus output signals of both CPUs and VIMs are compared. A difference in the CPU compare bus outputs is indicated by signaling an error to the ESM, which sets the error flag "CCM-R5F - CPU compare" and "CCM-R5F - VIM compare", respectively.

- CPU types of output signals to be compared:
  - Global signals
  - Interrupt signals
  - All L1 cache interface signals
  - All cache coherency signals
  - All L1 TCM interface signals
  - All L2 AXI interface signals
  - ETM interface signals
  - FPU signals
  - All ACP interface signals
  - All AXI Peripheral port interface signals
  - All AHB Peripheral port interface signals
  - All status and control signals
- VIM output signals to be compared:
  - nFIQ
  - nIRQ
  - IRQADDRV
  - IRQVECTADDR

NOTE: The CPU compare error asserts “CCM-R5F self-test error” flag as well. By doing this, the CPU compare error has two paths (“CCM-R5F - CPU compare” and “CCM-R5F self-test error” flag) to the ESM, so that even if one of the paths fails, the error is still propagated to the ESM. This is also true for "CCM-R5F - VIM compare" error flag.

Not all internal registers of the Cortex-R5F CPU have fixed values upon reset. To avoid an erroneous CCM-R5F compare error, the application software needs to ensure that the CPU registers of both CPUs are initialized with the same values before the registers are used, including function calls where the register values are pushed onto the stack.

13.2.1.2 Self-Test Mode

In self-test mode, the CCM-R5F checks itself for faults. During self-test, the compare error module output signal is deactivated. Any fault detected inside the CCM-R5F will be flagged by ESM error “CCM-R5F - self-test”.

In self-test mode, the CCM-R5F automatically generates test patterns to look for any hardware faults. If a fault is detected, then a self-test error flag is set, a self-test error signal is asserted and sent to the ESM, and the self-test is terminated immediately. If no fault is found during self-test, the self-test complete flag is set. In both cases, the CCM-R5F CPU / VIM Output Compare Diagnostic remains in self-test mode after the test has been terminated or completed, and the application needs to switch the CCM-R5F mode by writing another key to the mode key register (MKEY1 or MKEY2 depending which diagnostic is selected for self-test). During the self-test operation, the compare error signal output to the ESM is inactive irrespective of the compare result.

There are two types of patterns generated by CCM-R5F during self-test mode:

1. Compare Match Test
2. Compare Mismatch Test

CCM-R5F first generates Compare Match Test patterns, followed by Compare Mismatch Test patterns. Each test pattern is applied on both CPU signal inputs of the CCM-R5F’s compare block and clocked for one cycle. The duration of self-test for CPU Output Compare Diagnostic is 4947 CPU clock cycles (GCLK1) and 151 system peripheral clock cycles (VCLK) for VIM Output Compare Diagnostic.

NOTE: During self-test, both CPUs can execute normally, but the compare logic will not be checking any CPU signals. Also during self-test, only the compare unit logic is tested and not the memory-mapped register controls for the CCM-R5F. The self-test is not interruptible.

Self-test of all different diagnostics can be run at the same time.

13.2.1.2.1 Compare Match Test

During the Compare Match Test, there are four different test patterns generated to stimulate the CCM-R5F. An identical vector is applied to both input ports at the same time expecting a compare match. These patterns cause the self-test logic to exercise every CPU compare bus output signal in parallel. If the compare unit produces a compare mismatch then the self-test error flag is set, the self-test error signal is generated, and the Compare Match Test is terminated.

The four test patterns used for the Compare Match Test are:

- All 1s on both CPU / VIM signal ports
- All 0s on both CPU / VIM signal ports
- 0xAs on both CPU / VIM signal ports
- 0x5s on both CPU / VIM signal ports

These four test patterns will take four clock cycles to complete. Table 13-1 illustrates the sequence of Compare Match Test.

Table 13-1. Compare Match Test Sequence

CPU 1 (Main CPU) Signal Position									CPU 2 (Checker CPU) Signal Position									Cycle
n:8	7	6	5	4	3	2	1	0	n:8	7	6	5	4	3	2	1	0	
1s	1	1	1	1	1	1	1	1	1s	1	1	1	1	1	1	1	1	0
0s	0	0	0	0	0	0	0	0	0s	0	0	0	0	0	0	0	0	1
0xA	1	0	1	0	1	0	1	0	0xA	1	0	1	0	1	0	1	0	2
0x5	0	1	0	1	0	1	0	1	0x5	0	1	0	1	0	1	0	1	3

13.2.1.2.2 Compare Mismatch Test

During the Compare Mismatch Test, the number of test patterns is equal to twice the number of CPU output signals to compare in lockstep mode. An all 1s vector is applied to the CCM-R5F's CPU1 / VIM1 input port and the same pattern is also applied to the CCM-R5F's CPU2 /VIM2 input port but with one bit flipped starting from signal position 0. The un-equal vector will cause the CCM-R5F to expect a compare mismatch at signal position 0, if the CCM-R5F logic is working correctly. If, however, the CCM-R5F logic reports a compare match, the self-test error flag is set, the self-test error signal is asserted, and the Compare Mismatch Test is terminated.

This Compare Mismatch Test algorithm repeats in a domino fashion with the next signal position flipped while forcing all other signals to logic level 1. This sequence is repeated until every single signal position is verified on both CPU signal ports.

The Compare Mismatch Test is terminated if the CCM-R5F reports a compare match versus the expected compare mismatch. This test ensures that the compare unit is able to detect a mismatch on every CPU signal being compared. Table 13-2 illustrates the sequence of Compare Mismatch Test. There is no error signal sent to ESM if the expected errors are seen with each pattern.

Table 13-2. CPU / VIM Compare Mismatch Test Sequence

CPU 1 (Main CPU) Signal Position											CPU 2 (Checker CPU) Signal Position											Cycle
n	n–1:8		7	6	5	4	3	2	1	0	n	n–1:8		7	6	5	4	3	2	1	0	
1	1	1s	1	1	1	1	1	1	1	1	1	1	1s	1	1	1	1	1	1	1	0	0
1	1	1s	1	1	1	1	1	1	1	1	1	1	1s	1	1	1	1	1	1	0	1	1
1	1	1s	1	1	1	1	1	1	1	1	1	1	1s	1	1	1	1	1	0	1	1	2
1	1	1s	1	1	1	1	1	1	1	1	1	1	1s	1	1	1	1	0	1	1	1	3
::																						
1	1	1s	1	1	1	1	1	1	1	1	1	0	1s	1	1	1	1	1	1	1	1	–1
1	1	1s	1	1	1	1	1	1	1	1	0	1	1s	1	1	1	1	1	1	1	1	n
1	1	1s	1	1	1	1	1	1	1	0	1	1	1s	1	1	1	1	1	1	1	1	n+1
1	1	1s	1	1	1	1	1	1	0	1	1	1	1s	1	1	1	1	1	1	1	1	n+2
1	1	1s	1	1	1	1	1	0	1	1	1	1	1s	1	1	1	1	1	1	1	1	n+3
1	1	1s	1	1	1	1	0	1	1	1	1	1	1s	1	1	1	1	1	1	1	1	n+4
::																						
1	0	1s	1	1	1	1	1	1	1	1	1	1	1s	1	1	1	1	1	1	1	1	2n-1
0	1	1s	1	1	1	1	1	1	1	1	1	1	1s	1	1	1	1	1	1	1	1	2n

13.2.1.3 Error Forcing Mode

In error forcing mode, a test pattern is applied to the CPU / VIM related inputs of the CCM-R5F compare logic to force an error in the compare error output signal of the compare unit. Depending if error forcing mode is applied to the CPU Output Compare Diagnostic or VIM Output Compare Diagnostic, the ESM error flag “CCM-R5F - CPU compare” or “CCM-R5F - VIM compare” is expected after the error forcing mode completes. As a side effect, the “CCM-R5F self-test error” flag is also asserted whenever the CPU compare error is asserted.

Error forcing mode is similar to the Compare Mismatch Test operation of self-test mode in which an unequal vector is applied to the CCM-R5F CPU signal ports. The error forcing mode forces the compare mismatch to actually assert the compare error output signal. This ensures that a fault in the path between CCM-R5F and ESM is detected.

Only one hardcoded test pattern is applied into CCM-R5F during error forcing mode. A repeated 0x5 pattern is applied to CPU1 / VIM1 signal port of CCM-R5F input while a repeated 0xA pattern is applied to the CPU2 / VIM2 signal port of CCM-R5F input. The error forcing mode takes one cycle to complete. Hence, the failing signature is presented for one clock cycle. After that, the mode is automatically switched to lockstep mode. The key register (MKEY1 for CPU output compare and MKEY2 for VIM output compare) will indicate the lockstep key mode once it is switched to lockstep mode. During the one cycle required by the error forcing test, the CPU / VIM output signals are not compared. The user should expect the ESM to trigger a response (report the CCM-R5F fail). If no error is detected by the ESM, then a hardware fault is present.

13.2.1.4 Self-Test Error Forcing Mode

In self-test error forcing mode, an error is forced at the self-test error signal. The compare unit is still running in lockstep mode and the key is switched to lockstep after one clock cycle. The ESM error flag “CCM-R5F - self-test” is expected after the self-test error forcing mode completes. Once the expected errors are seen, the application can clean the error through the ESM module.

Table 13-3 shows what error signals and flags are asserted in different operating mode. The behavior of different modes in this table for CPU compare is also valid for other diagnostics such as VIM compare, Checker CPU Inactivity Monitor and Power Domain monitor.

Table 13-3. Error Flags and Error Signals Generation in Each Mode

Mode	Key	Self Test Error Signal	Compare Error Signal	CMPE	STC	STET	STE
Active Compare Lockstep	0000	Enabled	Enabled	Enabled	Disabled	Disabled	Disabled
Self-Test	0110	Enabled	Disabled	Disabled	Enabled	Enabled	Enabled
Error Forcing	1001	Error	Error	Disabled	Disabled	Disabled	Disabled
Self-Test Error Forcing	1111	Error	Enabled	Enabled	Disabled	Disabled	Disabled

13.2.2 CPU Input Inversion Diagnostic

There is another way to intentionally create a mismatch between the two CPUs' outputs as a diagnostic test to self-test the CCM-R5F's CPU Output Compare Diagnostic block. Before the CPU1's outputs are taken to the CCM-R5F, eight of the output signals are first exclusive-ORed bitwise with the 8-bit POLARITYINVERT register. After reset, the default value of the POLARITYINVERT register is all zeros. The resultant values of the 8 signals after the XOR logic with the POLARITYINVERT register will still be the same as the original 8 signal values. However, by programming the POLARITYINVERT to a non-zero values it will have the effect to invert the signal values. This intentional inversion on the inputs to the CCM-R5F will cause the CPU Output Compare Diagnostic to detect a compare error. See Figure 13-2 for illustration.

Figure 13-2. CPU Input Inversion Scheme

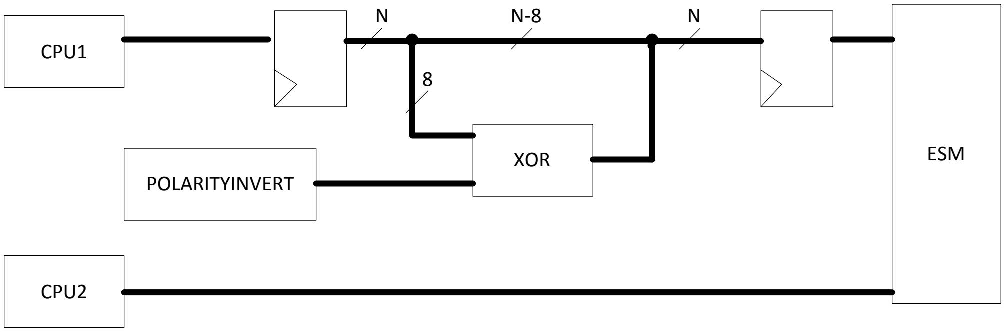


Table 13-4. CPU1 (Main CPU) Signals Being Inverted Before Being Compared

Signals	Remark
AWVALIDM	Indicates write address and control are valid
ARVALIDM	Indicates write address and control are valid
AWVALIDP	Indicates write address and control are valid
ARVALIDP	Indicates write address and control are valid
HTRANSP[1:0]	Indicates write address and control are valid

13.2.3 Checker CPU Inactivity Monitor

Similar to the CPU / VIM Output Compare Diagnostic, the Checker CPU Inactivity Monitor can also run in one of the following four operating modes:

1. Active compare
2. Self-test
3. Error forcing
4. Self-test error forcing

The operating mode can be selected by writing a dedicated key to the key register (MKEY3).

13.2.3.1 Active Compare Mode

This is the default mode on start-up. In this mode, several key bus signals such as the bus valid control signals from the checker CPU that would have indicated a valid bus transaction onto the interconnect are compared against their clamped safe values. While the two CPUs are in lockstep configuration, the outputs of the checker CPU are supposed to clamp to the inactive state that is all zeros. A difference between the checker CPU compare bus outputs and their respective inactive states is indicated by signaling an error to the ESM which sets the error flag "CCM-R5F - CPU1 AXIM Bus Monitor Failure".

Table 13-5. Checker CPU Signals to Monitor

Signals	Remark
AWVALIDM	When asserted, indicates address and control are valid on the Checker CPU's AXI master port for write transaction.
ARVALIDM	When asserted, indicates address and control are valid on the Checker CPU's AXI master port for read transaction.
AWVALIDP	When asserted, indicates address and control are valid on the Checker CPU's AXI peripheral port for write transaction.
ARVALIDP	When asserted, indicates address and control are valid on the Checker CPU's AXI peripheral port for read transaction.
BVALIDS	When asserted, indicates that a valid write response is available on the Checker CPU's AXI slave port for write transaction
RVALIDS	When asserted, indicates address and control are valid on the Checker CPU's AXI slave port for read transaction

13.2.3.2 Self-Test Mode

Similar to the other self-test described for CPU / VIM Output Compare Diagnostic, the Checker CPU Inactivity Monitor can be placed in self-test mode. In self-test mode, the CCM-R5F checks the Checker CPU Inactivity Monitor itself for faults. During self-test, the compare error module output signal is deactivated. Any fault detected inside the CCM-R5F will be flagged by ESM error “CCM-R5F - self-test”.

In self-test mode, the CCM-R5F automatically generates test patterns to look for any hardware faults. If a fault is detected, then a self-test error flag is set, a self-test error signal is asserted and sent to the ESM, and the self-test is terminated immediately. If no fault is found during self-test, the self-test complete flag is set. In both cases, the CCM-R5F Checker CPU Inactivity Monitor Diagnostic remains in self-test mode after the test has been terminated or completed, and the application needs to switch the CCM-R5F mode by writing another key to the mode key register (MKEY3). During the self-test operation, the compare error signal output to the ESM is inactive irrespective of the compare result.

There are also two types of patterns generated by CCM-R5F during self-test mode for Check CPU Inactivity Monitor. The difference here is the number of test patterns applied during self-test.

i. Compare Match Test

ii. Compare Mismatch Test

CCM-R5F first generates Compare Match Test patterns, followed by Compare Mismatch Test patterns.

13.2.3.2.1 Compare Match Test

Since the comparison is done against the clamped values, and all compared signals are clamped to zero, only one test pattern is applied for the compare match test. A pattern of all-zeros are applied for the compare match test. The test will take one cycle. If the compare unit produces a compare mismatch then the self-test error flag is set, the self-test error signal is generated, and the Compare Match Test is terminated.

13.2.3.2.2 Compare Mismatch Test

During the Compare Mismatch Test, the number of test patterns is equal to the number of bus signals on the checker CPU to be monitored. There are a total of 6 signals being monitored on the checker CPU's level 2 interface and hence it takes 6 test patterns for the mismatch test. The mismatch test will take a total of 6 cycles to complete. An all 0's test vector is applied to the CCM-R5F’s but with one bit flipped starting from signal position 0. The un-equal vector will cause the CCM-R5F to expect a compare mismatch at signal position 0, if the CCM-R5F logic is working correctly. If, however, the CCM-R5F logic reports a compare match, the self-test error flag is set, the self-test error signal is asserted, and the Compare Mismatch Test is terminated.

This Compare Mismatch Test algorithm repeats in a domino fashion with the next signal position flipped while forcing all other signals to logic level 0. This sequence is repeated until every inactivity monitor signal position is verified on the checker CPU .

Table 13-6 shows the sequence of Compare Mismatch Test. There is no error signal sent to ESM if the expected errors are seen with each pattern.

Table 13-6. Checker CPU Inactivity Monitor Compare Mismatch Test

Signal Position						
5	**4**	**3**	**2**	**1**	**0**	**Cycle**
0	0	0	0	0	1	0
0	0	0	0	1	0	1
0	0	0	1	0	0	2
0	0	1	0	0	0	3
0	1	0	0	0	0	4
1	0	0	0	0	0	5

13.2.3.3 Error Forcing Mode

In error forcing mode, a test pattern of all 1's is applied to the check CPU's compare logic to force an error in the compare error output signal of the compare unit. The ESM error flag “CCM-R5F - CPU1 AXIM Bus Inactivity failure” is expected after the error forcing mode completes. As a side effect, the “CCM-R5F self-test error” flag is also asserted whenever the CPU compare error is asserted.

The error forcing mode takes one cycle to complete. Hence, the failing signature is presented for one clock cycle. After that, the mode is automatically switched to active compare mode. The key register (MKEY3) will indicate the active compare mode once it is switched to active compare mode. During the one cycle required by the error forcing test, the checker CPU Inactivity Monitor is deactivated. User should expect the ESM to trigger a response (report the CCM-R5F fail). If no error is detected by ESM, then a hardware fault is present.

13.2.3.4 Self-Test Error Forcing Mode

In self-test error forcing mode, an error is forced at the self-test error signal. The compare unit is still running in active compare mode and the key is switched to active compare after one clock cycle. The ESM error flag “CCM-R5F - self-test” is expected after the self-test error forcing mode completes. Once the expected errors are seen, the application can clean the error through the ESM module.

13.2.4 Power Domain Inactivity Monitor

The Power Domain Inactivity Monitor is very similar to the Checker CPU Inactivity Monitor in concept. When a power domain is turned off, its outputs are isolated from the rest of the system. The outputs are clamped to inactive safe values. Depending on the signals, the clamp value of a signal may be 0 or 1. Some bus masters may be residing in the turned off power domains. Key bus signals from the power domains which would have indicated that the bus master is generating a valid bus transaction are compared against their clamped values.

The Power Domain Inactivity Monitor Diagnostic can also run in one of the following four operating modes:

1. Active compare
2. Self-test
3. Error forcing
4. Self-test error forcing

The operating mode can be selected by writing a dedicated key to the key register (MKEY4).

13.2.4.1 Active Compare Mode

This is the default mode on start-up.

In this mode, several critical bus signals such as the bus request control signals from the power domains which would have indicated a valid bus transaction onto the interconnect are compared against their clamped safe values. If a power domain is turned off, the outputs of the power domain are expected to clamp to the inactive states. A difference between the power domain compare bus outputs and their respective inactive states is indicated by signaling an error to the ESM which sets the error flag "CCM-R5F - Power Domain Monitor Failure". In addition, the corresponding bus masters for which the compare block detected the monitor failure are also captured in the CCMPDSTAT0 register.

Self-test mode, Error forcing mode and Self-test error forcing mode for Power Domain Inactivity Monitor Diagnostic are the same as Checker CPU Inactivity Monitor Diagnostic. See Section 13.2.3.2, Section 13.2.3.3, and Section 13.2.3.4 for details.

13.2.5 Operation During CPU Debug Mode

Certain debug operations place the CPU in a halting debug state where the code execution is halted. Because halting debug events are asynchronous, there is a possibility for the debug requests to cause loss of lockstep. CCM-R5F will disable all functional diagnostics upon detection of halting debug requests. Core compare error will not be generated and flags will not update. A CPU reset is needed to ensure the CPUs are again in lockstep and will also re-enable the CCM-R5F.

13.3 Control Registers

Table 13-7 lists the CCM-R5F registers. Each register begins on a 32-bit word boundary. The registers support 32-bit, 16-bit, and 8-bit accesses. The base address for the control registers is FFFF F600h.

Table 13-7. Control Registers

Offset	Acronym	Register Description	Section
00h	CCMSR1	CCM-R5F Status Register 1	Section 13.3.1
04h	CCMKEYR1	CCM-R5F Key Register 1	Section 13.3.2
08h	CCMSR2	CCM-R5F Status Register 2	Section 13.3.3
0Ch	CCMKEYR2	CCM-R5F Key Register 2	Section 13.3.4
10h	CCMSR3	CCM-R5F Status Register 3	Section 13.3.5
14h	CCMKEYR3	CCM-R5F Key Register 3	Section 13.3.6
18h	CCMPOLCNTRL	Polarity Control Register	Section 13.3.7
1Ch	CCMSR4	CCM-R5F Status Register 4	Section 13.3.8
20h	CCMKEYR4	CCM-R5F Key Register 4	Section 13.3.9
24h	CCMPDSTAT0	CCM-R5F Power Domain Status Register 0	Section 13.3.10

13.3.1 CCM-R5F Status Register 1 (CCMSR1)

The contents of this register should be interpreted in context of what test was selected. That is, what mode is CCM operating.

Figure 13-3. CCM-R5F Status Register 1 (CCMSR1) (Offset = 00h)

31 17	16
Reserved	CPME1
R-0	R/W1CP-0

15 9	8	7 2	1	0
Reserved	STC1	Reserved	STET1	STE1
R-0	R-0	R-0	R-0	R-0

LEGEND: R/W = Read/Write; R = Read only; W1CP = Write 1 to clear in privilege mode only; -*n* = value after reset

Table 13-8. CCM-R5F Status Register 1 (CCMSR1) Field Descriptions

Bit	Field	Value	Description
31-17	Reserved	0	Reads return 0. Writes have no effect.
16	CMPE1		Compare Error for CPU Output Compare Diagnostic. **Read in User and Privileged mode. Write in Privileged mode only.**
		0	Read: CPU signals are identical. Write: Leaves the bit unchanged.
		1	Read: CPU signal compare mismatch. Write: Clears the bit.
15-9	Reserved		Reads return 0. Writes have no effect.
8	STC1		Self-test Complete for CPU Output Compare Diagnostic. **Note:** This bit is always 0 when not in self-test mode. Once set, switching from self-test mode to other modes will clear this bit. **Read/Write in User and Privileged mode.**
		0	Read: Self-test on-going if self-test mode is entered. Write: Writes have no effect.
		1	Read: Self-test is complete. Write: Writes have no effect.
7-2	Reserved		Reads return 0. Writes have no effect.
1	STET1		Self-test Error Type for CPU Output Compare Diagnostic. **Read/Write in User and Privileged mode.**
		0	Read: Self-test failed during Compare Match Test if STE1 = 1. Write: Writes have no effect.
		1	Read: Self-test failed during Compare Mismatch Test if STE1 = 1. Write: Writes have no effect.
0	STE1		Self-test Error for CPU Output Compare Diagnostic. **Note:** This bit gets updated when the self-test is complete or an error is detected. **Read/Write in User and Privileged mode.**
		0	Read: Self-test passed. Write: Writes have no effect.
		1	Read: Self-test failed. Write: Writes have no effect.

13.3.2 CCM-R5F Key Register 1 (CCMKEYR1)

Figure 13-4. CCM-R5F Key Register 1 (CCMKEYR1) (Offset = 04h)

31	16
Reserved	
R-0	

15	4	3	0
Reserved		MKEY1	
R-0		R/WP-0	

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privileged mode only; -*n* = value after reset

Table 13-9. CCM-R5F Key Register 1 (CCMKEYR1) Field Descriptions

Bit	Field	Value	Description
31-4	Reserved	0	Reads return 0. Writes have no effect.
3-0	MKEY1	0	Mode Key to select operation for CPU Output Compare Diagnostic . **Read in User and Privileged mode. Write in Privileged mode only.** Read: Returns current value of the MKEY1. Write: Active Compare Lockstep mode.
		6h	Read: Returns current value of the MKEY1. Write: Self-test mode.
		9h	Read: Returns current value of the MKEY1. Write: Error Forcing mode.
		Fh	Read: Returns current value of the MKEY1. Write: Self-test Error Forcing mode.
		Other values	**Note:** It is recommended to not write any other key combinations. Invalid keys will result in switching operation to lockstep mode.

13.3.3 CCM-R5F Status Register 2 (CCMSR2)

Figure 13-5. CCM-R5F Status Register 2 (CCMSR2) (Offset = 08h)

31 … 17	16
Reserved	CPME2
R-0	R/W1CP-0

15 … 9	8	7 … 2	1	0
Reserved	STC2	Reserved	STET2	STE2
R-0	R-0	R-0	R-0	R-0

LEGEND: R/W = Read/Write; R = Read only; W1CP = Write 1 to clear in privilege mode only; -*n* = value after reset

Table 13-10. CCM-R5F Status Register 2 (CCMSR2) Field Descriptions

Bit	Field	Value	Description
31-17	Reserved	0	Reads return 0. Writes have no effect.
16	CMPE2		Compare Error for VIM Output Compare Diagnostic. **Read in User and Privileged mode. Write in Privileged mode only.**
		0	Read: CPU signals are identical. Write: Leaves the bit unchanged.
		1	Read: CPU signal compare mismatch. Write: Clears the bit.
15-9	Reserved		Reads return 0. Writes have no effect.
8	STC2		Self-test Complete for VIM Output Compare Diagnostic. **Note:** This bit is always 0 when not in self-test mode. Once set, switching from self-test mode to other modes will clear this bit. **Read/Write in User and Privileged mode.**
		0	Read: Self-test on-going if self-test mode is entered. Write: Writes have no effect.
		1	Read: Self-test is complete. Write: Writes have no effect.
7-2	Reserved		Reads return 0. Writes have no effect.
1	STET2		Self-test Error Type for VIM Output Compare Diagnostic. **Read/Write in User and Privileged mode.**
		0	Read: Self-test failed during Compare Match Test if STE2 = 1. Write: Writes have no effect.
		1	Read: Self-test failed during Compare Mismatch Test if STE2 = 1. Write: Writes have no effect.
0	STE2		Self-test Error for VIM Output Compare Diagnostic. **Note:** This bit gets updated when the self-test is complete or an error is detected. **Read/Write in User and Privileged mode.**
		0	Read: Self-test passed. Write: Writes have no effect.
		1	Read: Self-test failed. Write: Writes have no effect.

13.3.4 CCM-R5F Key Register 2 (CCMKEYR2)

Figure 13-6. CCM-R5F Key Register 2 (CCMKEYR2) (Offset = 0Ch)

31 16
Reserved
R-0

15 4	3 0
Reserved	MKEY2
R-0	R/WP-0

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privileged mode only; -*n* = value after reset

Table 13-11. CCM-R5F Key Register 2 (CCMKEYR2) Field Descriptions

Bit	Field	Value	Description
31-4	Reserved	0	Reads return 0. Writes have no effect.
3-0	MKEY2		Mode Key to select operation for VIM Output Compare Diagnostic. **Read in User and Privileged mode. Write in Privileged mode only.**
		0	Read: Returns current value of the MKEY2. Write: Active Compare Lockstep mode.
		6h	Read: Returns current value of the MKEY2. Write: Self-test mode.
		9h	Read: Returns current value of the MKEY2. Write: Error Forcing mode.
		Fh	Read: Returns current value of the MKEY2. Write: Self-test Error Forcing mode.
		Other values	**Note:** It is recommended to not write any other key combinations. Invalid keys will result in switching operation to lockstep mode.

13.3.5 CCM-R5F Status Register 3 (CCMSR3)

Figure 13-7. CCM-R5F Status Register 3 (CCMSR3) (Offset = 10h)

31 17	16
Reserved	CPME3
R-0	R/W1CP-0

15 9	8	7 2	1	0
Reserved	STC3	Reserved	STET3	STE3
R-0	R-0	R-0	R-0	R-0

LEGEND: R/W = Read/Write; R = Read only; W1CP = Write 1 to clear in privilege mode only; -*n* = value after reset

Table 13-12. CCM-R5F Status Register 3 (CCMSR3) Field Descriptions

Bit	Field	Value	Description
31-17	Reserved	0	Reads return 0. Writes have no effect.
16	CMPE3	0	Compare Error for Checker CPU Inactivity Monitor. **Read in User and Privileged mode. Write in Privileged mode only.** Read: CPU signals are identical. Write: Leaves the bit unchanged.
		1	Read: CPU signal compare mismatch. Write: Clears the bit.
15-9	Reserved		Reads return 0. Writes have no effect.
8	STC3	0	Self-test Complete for Checker CPU Inactivity Monitor. **Note:** This bit is always 0 when not in self-test mode. Once set, switching from self-test mode to other modes will clear this bit. **Read/Write in User and Privileged mode.** Read: Self-test on-going if self-test mode is entered. Write: Writes have no effect.
		1	Read: Self-test is complete. Write: Writes have no effect.
7-2	Reserved		Reads return 0. Writes have no effect.
1	STET3	0	Self-test Error Type for Checker CPU Inactivity Monitor. **Read/Write in User and Privileged mode.** Read: Self-test failed during Compare Match Test if STE3 = 1. Write: Writes have no effect.
		1	Read: Self-test failed during Compare Mismatch Test if STE3 = 1. Write: Writes have no effect.
0	STE3	0	Self-test Error for Checker CPU Inactivity Monitor. **Note:** This bit gets updated when the self-test is complete or an error is detected. **Read/Write in User and Privileged mode.** Read: Self-test passed. Write: Writes have no effect.
		1	Read: Self-test failed. Write: Writes have no effect.

13.3.6 CCM-R5F Key Register 3 (CCMKEYR3)

Figure 13-8. CCM-R5F Key Register 3 (CCMKEYR3) (Offset = 14h)

31	16
Reserved	
R-0	

15	4	3	0
Reserved		MKEY3	
R-0		R/WP-0	

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privileged mode only; -*n* = value after reset

Table 13-13. CCM-R5F Key Register 2 (CCMKEYR2) Field Descriptions

Bit	Field	Value	Description
31-4	Reserved	0	Reads return 0. Writes have no effect.
3-0	MKEY3		Mode Key to select operation for Checker CPU Inactivity Monitor. **Read in User and Privileged mode. Write in Privileged mode only.**
		0	Read: Returns current value of the MKEY3. Write: Active Compare Lockstep mode.
		6h	Read: Returns current value of the MKEY3. Write: Self-test mode.
		9h	Read: Returns current value of the MKEY3. Write: Error Forcing mode.
		Fh	Read: Returns current value of the MKEY3. Write: Self-test Error Forcing mode.
		Other values	**Note:** It is recommended to not write any other key combinations. Invalid keys will result in switching operation to lockstep mode.

13.3.7 CCM-R5F Polarity Control Register (CCMPOLCNTRL)

Figure 13-9. CCM-R5F Polarity Control Register (CCMPOLCNTRL) (Offset = 18h)

31	16
Reserved	
R-0	

15	8	7	0
Reserved		POLARITYINVERT	
R-0		R/WP-0	

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privileged mode only; -*n* = value after reset

Table 13-14. CCM-R5F Polarity Control Register (CCMPOLCNTRL) Field Descriptions

Bit	Field	Value	Description
31-8	Reserved	0	Reads return 0. Writes have no effect.
3-0	POLARITYINVERT		Polarity Inversion. This value is used to invert one of the 8 output compare signals from the CPU1 to the CCM-R5F. Inverting any one signal will lead to compare error by the CPU Output Compare Diagnostic. **Read in User and Privileged mode. Write in Privileged mode only.**

13.3.8 CCM-R5F Status Register 4 (CCMSR4)

Figure 13-10. CCM-R5F Status Register 4 (CCMSR4) (Offset = 1Ch)

31	17	16
Reserved		CPME4
R-0		R/W1CP-0

15	9	8	7	2	1	0
Reserved		STC4	Reserved		STET4	STE4
R-0		R-0	R-0		R-0	R-0

LEGEND: R/W = Read/Write; R = Read only; W1CP = Write 1 to clear in privilege mode only; -*n* = value after reset

Table 13-15. CCM-R5F Status Register 4 (CCMSR4) Field Descriptions

Bit	Field	Value	Description
31-17	Reserved	0	Reads return 0. Writes have no effect.
16	CMPE4	0	Compare Error for Power Domain Inactivity Monitor. **Read in User and Privileged mode. Write in Privileged mode only.** Read: CPU signals are identical. Write: Leaves the bit unchanged.
		1	Read: CPU signal compare mismatch. Write: Clears the bit.
15-9	Reserved		Reads return 0. Writes have no effect.
8	STC4	0	Self-test Complete for Power Domain Inactivity Monitor. **Note:** This bit is always 0 when not in self-test mode. Once set, switching from self-test mode to other modes will clear this bit. **Read/Write in User and Privileged mode.** Read: Self-test on-going if self-test mode is entered. Write: Writes have no effect.
		1	Read: Self-test is complete. Write: Writes have no effect.
7-2	Reserved		Reads return 0. Writes have no effect.
1	STET4	0	Self-test Error Type for Power Domain Inactivity Monitor. **Read/Write in User and Privileged mode.** Read: Self-test failed during Compare Match Test if STE4 = 1. Write: Writes have no effect.
		1	Read: Self-test failed during Compare Mismatch Test if STE4 = 1. Write: Writes have no effect.
0	STE4	0	Self-test Error for Power Domain Inactivity Monitor. **Note:** This bit gets updated when the self-test is complete or an error is detected. **Read/Write in User and Privileged mode.** Read: Self-test passed. Write: Writes have no effect.
		1	Read: Self-test failed. Write: Writes have no effect.

13.3.9 CCM-R5F Key Register 4 (CCMKEYR4)

Figure 13-11. CCM-R5F Key Register 4 (CCMKEYR4) (Offset = 20h)

31	16
Reserved	
R-0	

15	4	3	0
Reserved		MKEY4	
R-0		R/WP-0	

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privileged mode only; -*n* = value after reset

Table 13-16. CCM-R5F Key Register 4 (CCMKEYR4) Field Descriptions

Bit	Field	Value	Description
31-4	Reserved	0	Reads return 0. Writes have no effect.
3-0	MKEY4		Mode Key to select operation for Power Domain Inactivity Monitor. **Read in User and Privileged mode. Write in Privileged mode only.**
		0	Read: Returns current value of the MKEY4. Write: Active Compare Lockstep mode.
		6h	Read: Returns current value of the MKEY4. Write: Self-test mode.
		9h	Read: Returns current value of the MKEY4. Write: Error Forcing mode.
		Fh	Read: Returns current value of the MKEY4. Write: Self-test Error Forcing mode.
		Other values	**Note:** It is recommended to not write any other key combinations. Invalid keys will result in switching operation to lockstep mode.

13.3.10 CCM-R5F Power Domain Status Register 0 (CCMPDSTAT0)

Figure 13-12. CCM-R5F Power Domain Status Register 0 (CCMPDSTAT0) (Offset = 24h)

31	16
Reserved	
R-0	

15	6	5	4	3	2	1	0
Reserved		DMM_TRANS		HTU2_TRANS		FTU_TRANS	
R-0		R-0		R-0		R-0	

LEGEND: R = Read only; -*n* = value after reset

Table 13-17. CCM-R5FPower Domain Status Register 0 (CCMPDSTAT0) Field Descriptions

Bit	Field	Value	Description
31-6	Reserved	0	Reads return 0. Writes have no effect.
5-4	DMM_TRNS		DMM Transaction. When the power domain in which the DMM resides is turned off, an unexpected bus transaction is detected on DMM master. **Read in User and Privileged mode. Write has no effect.**
		0	Read: No bus transaction on the master when the power domain is turned off. Write: Writes have no effect.
		Any non-zero value	Read: An unexpected bus transaction is detected on the master. Write: Writes have no effect.
3-2	HTU2_TRNS		HTU2 Transaction. When the power domain in which the HTU2 resides is turned off, an unexpected bus transaction is detected on HTU2 master. **Read in User and Privileged mode. Write has no effect.**
		0	Read: No bus transaction on the master when the power domain is turned off. Write: Writes have no effect.
		Any non-zero value	Read: An unexpected bus transaction is detected on the master. Write: Writes have no effect.
1-0	FTU_TRNS		FTU Transaction. When the power domain in which the FTU resides is turned off, an unexpected bus transaction is detected on FTU master. **Read in User and Privileged mode. Write has no effect.**
		0	Read: No bus transaction on the master when the power domain is turned off. Write: Writes have no effect.
		Any non-zero value	Read: An unexpected bus transaction is detected on the master. Write: Writes have no effect.

Chapter 14

SPNU563A–March 2018

Oscillator and PLL

This chapter describes the oscillator and PLL clock source paths for the device.

Topic **Page**

14.1 Introduction

The oscillator macro will pass a signal driven into the OSCIN pin to clock source 0 that is the device default clock source on reset. When a crystal or resonator with appropriate load circuitry is connected to OSCIN and OSCOUT, the oscillator macro drives the crystal/resonator to generate the input waveform. In addition to being directly usable as clock source 0, the oscillator clock is the input to the PLL.

The oscillator frequency is continuously monitored by a dedicated clock detect circuit. If the frequency falls out of a fixed range, the clock detect switches the clock from the oscillator to an internally generated, free-running frequency (generated by the low power oscillator (LPO)).

The phase lock loop (PLL), a circuit in the microcontroller, is used to multiply the input frequency to some higher (device operation) frequency. This frequency synthesis is useful for generating higher frequencies than can be conveniently achieved with an external crystal or resonator. Additionally, the PLL allows the flexibility to be able to synthesize one of multiple frequency options from a given crystal or resonator.

Frequency modulation can be superimposed on the synthesized frequency. The modulation provides a means to reduce the impact of electromagnetic radiation from the device; this reduction in measured radiation can be useful in sensitive applications.

14.1.1 Features

The main features of the source clock path are:

- The oscillator may drive a crystal/resonator or be driven from an external source
- The clock detect provides continuous monitoring of the oscillator frequency and provides an automatic switch over to a free-running clock in case of oscillator failure.
- The FM-PLL module can be operated in either modulation or non-modulation mode.
- The phase-frequency detector assures lock to the fundamental reference frequency.
- $$f_{PLL} = \frac{f_{OSCIN}}{NR} \times \frac{NF}{OD \times R} \quad (1)$$
  - Configurable prescale divider (NR) for the input clock
  - Configurable multiplier (NF)
  - Configurable postscale dividers (OD, R)
- The PLL may be used with modulation enabled.
  - Configurable modulation frequency (NS)
  - Configurable modulation depth (NV)
- The slip control circuitry provides flexible response to a PLL failure (slip) including reset or automatic switch over to oscillator.

14.2 Quick Start

The purpose of this section is to provide an overview of how to configure the oscillator and PLL clock paths on power-up. More detailed descriptions are presented in later sections. Figure 14-1 shows the oscillator and PLL clock paths.

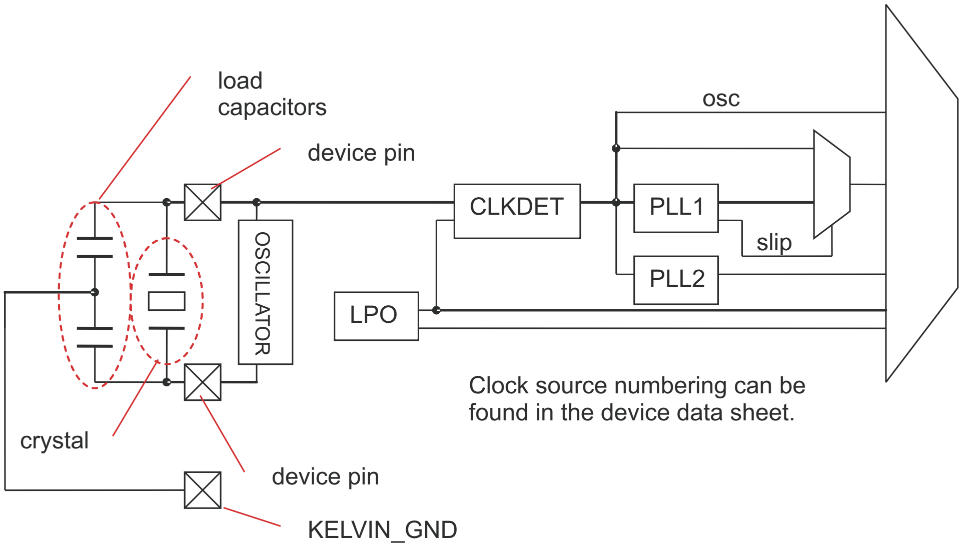
While power-on reset is asserted (low), the oscillator and low power oscillator (LPO) are enabled and start-up by default. After power-on reset is released to a high level, the clock detect circuit (CLKDET) begins to monitor the oscillator. If the oscillator is within a valid range, the oscillator becomes the default clock for the device as it exits reset; if the oscillator is not within a valid range, the clock detect selects the high-frequency low power oscillator as the default clock for the device.

The low power oscillator has a wide frequency range which also creates a large valid window for the clock detect; in order to refine the clock detect window, the low power oscillator can be trimmed. The initial trim value is stored in one-time programmable section of the flash memory, address 0xF008_01B4. Bits 31:16 of this word contain a 16 bit value that may be programmed into LPOMONCTL(15:0) in order to initialize the trim for both HF LPO and LF LPO. Software should read the initial trim values from flash and write them to the control register.

The PLL is disabled by default on power-up. The PLL control registers (PLLCTL1 and PLLCTL2) must be configured to set the desired output frequency. Then, the system PLL may be enabled (CSDISCLR.1). Similarly, the second PLL must be configured in PLLCTL3 and enabled (CSDISCLR.6). Each PLL has a valid bit that indicates the PLL is locked (CLKSRnV bit in the Clock Source Valid Status Register (CSVSTAT) of the System and Peripheral Control Registers).

Prior to selecting the PLL clock as the source for a clock domain (GCLK1, HCLK, VCLKA1), the domain and modules on the domain must be configured to accept the new frequency. An example of a module that should be configured prior to selecting the PLL as clock source for GCLK1 and HCLK is the memory wrapper to insure that access times are maintained correctly.

Figure 14-1. Clock Path from Oscillator through PLL to Device



Clock source numbering can be found in the device data sheet.

14.3 Oscillator

The clock generation path through the PLL begins with the oscillator. The oscillator consists of three separate pads -- OSCIN, OSCOUT, and Kelvin_GND (see Figure 14-2).

The oscillator is responsible for two independent functions:

1. The oscillator is responsible for generating positive feedback in the external crystal/resonator with appropriate load and tank circuitry. At start-up, the oscillator amplifies random noise. The external circuitry acts like a band-pass and selects the crystal/resonator frequency to provide as positive feedback into the amplifier. The positive feedback increases the amplitude of the output waveform into the crystal/resonator (and the load circuitry), and the voltage waveform shows an envelope of increasing amplitude. The oscillator can drive a crystal frequency that is within the data sheet range $t_{c(OSC)}$.

   Looking at the input waveform into OSCIN, the voltage waveform is an AC-coupled, filtered version of the OSCOUT waveform. The band-pass functionality of the crystal/resonator removes distortion from the OSCOUT waveform, leaving a sinusoidal input waveform.

NOTE: Vendor Validation of Resonators/Crystals

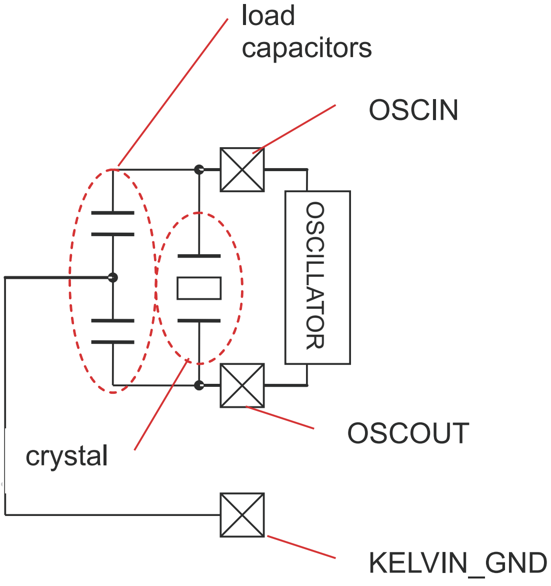
The crystal is a very tight bandpass filter while a resonator is a somewhat wider bandpass. The load circuitry pulls the center frequency of the bandpass.

Texas Instruments strongly encourages each customer to submit samples of the device to the resonator/crystal vendor for validation. The vendor is equipped to determine what load capacitances will best tune their resonator/crystal to the microcontroller device for optimum start-up and operation over temperature and voltage extremes. The vendor also factors in margins for variations in the microcontroller process.

2. The oscillator is also responsible for squaring-up the input waveform. This squaring-up converts the sinusoid into a square wave at the core logic levels. The input path limits the input frequency range as a low-pass filter with a cutoff frequency.

   The oscillator has a frequency range that is determined by the driving capability of external crystals/resonators (feedback path). If a clock is driven directly into the oscillator, then the feedback path is not relevant and the frequency range is determined solely by the forward path (which typically allows a higher frequency); the device can support inputs within the data sheet range $t_{c(OSC\_Sqr)}$.

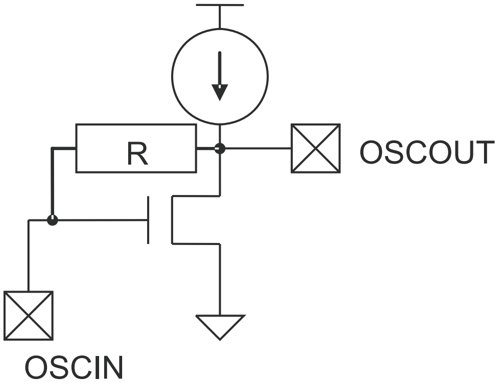
Figure 14-2. Clock Generation Path



14.3.1 Oscillator Implementation

The oscillator operates at 3.3V and uses a constant current source to drive current onto the OSCOUT node. An internal transistor shunts the current (and current from the external circuitry) to GND. This current steering drives the voltage waveform on OSCOUT.

Figure 14-3. Oscillator Implementation



14.3.2 Oscillator Enable

The oscillator is enabled asynchronously when nPORRST is low.

The oscillator is enabled by clearing bit 0 in the Clock Source Disable Register (CSDIS) or setting bit 0 in the Clock Source Disable Clear Register (CSDISCLR) of the System and Peripheral Control Registers. The bit sends a start signal to the oscillator. Bit 0 of CSDIS is cleared to 0 by default on a system or power-on reset so that the oscillator starts-up by default. After the oscillator swings at a high-enough amplitude to pass an input clock into the core domain and nPORRST is released, 1024 oscillator periods are counted before setting the CLKSR0V bit in the Clock Source Valid Status Register (CSVSTAT) of the System and Peripheral Control Registers. The oscillator generates clock source 0 in the global clock module (GCM).

14.3.3 Oscillator Disable

The clock sources (for example, OSC, PLL) are disabled by setting the appropriate bit in the Clock Source Disable Register (CSDIS) or setting the appropriate bit in the Clock Source Disable Set Register (CSDISSET) of the System and Peripheral Control Registers. These bits *allow* the clock source to disable but do not force the behavior until the clock is no longer used as the source for a clock domain (for example, GCLK1, VCLK, VCLK2, RTICLK1). The CLKSR0V bit in the Clock Source Valid Status Register (CSVSTAT), of the System and Peripheral Control Registers, is cleared after clock disable is asserted (which occurs after all clock domains are stopped).

The oscillator disable signal places the oscillator into a low-power state, disconnects the feedback (bias) resistor between OSCIN and OSCOUT, and OSCIN is grounded.

14.4 Low Power Oscillator and Clock Detect (LPOCLKDET)

The Low Power Oscillator (LPO) is comprised of two oscillators -- HF LPO and LF LPO -- in a single macro. The low power oscillator and clock detect (LPOCLKDET) uses a relaxation oscillator to generate an internal clock whose frequency is NOT tightly controlled. This frequency is used to monitor the oscillator input frequency and is also available as an independent clock source in the GCM.

The LPO produces two frequencies:

- High-frequency low-power oscillator (HF LPO) with a nominal frequency of 9.6MHz and a range from 5.5MHz to 19.5MHz; the HF LPO generates clock source 5 in the GCM.
- Low-frequency low-power oscillator (LF LPO) with a nominal frequency of 85kHz; the LF LPO generates clock source 4 in the GCM.

A single current source drives current onto a capacitor; when the voltage on the capacitor exceeds some threshold, the clock toggles. The LPO uses a single current source and the two different comparators to generate the HF LPO and LF LPO frequencies. The LPO is controlled by 4 different bit fields -- CSDIS.(5:4), HFTRIM(4:0), LFTRIM(4:0), and BIASEN.

- CSDIS.5 enables/disables the comparator that generates HF LPO.
- CSDIS.4 enables/disables the comparator that generates LF LPO.
- The HF TRIM and LF TRIM bit fields vary the current into the comparator to independently trim the HF LPO and LF LPO frequencies.
- BIAS ENABLE (LPOMONCTL.24) enables/disables the current source which drives the LPO.

14.4.1 Clock Detect

The LPO HF clock frequency is typically near 9.6MHz, but ranges from 5.5MHz to 19.5MHz. The clock detect establishes a window for the oscillator by:

OSCIN > HF LPO_{min} / 4	OSCIN / 4 < HF LPO_{max}
OSCIN > 5.5[MHz] / 4 = 1.375[MHz]	OSCIN < 4 × 19.5 = 78[MHz]

The clock detect circuit works by checking for a rising edge on one clock (oscillator or HF LPO) between rising edges of the other clock. The result is that in addition to flagging incorrect, repeating frequencies, the circuit also fails due to transient conditions.

The low end of the clock detect window ignores a transient low phase of at least 12 HF LPO cycles.

NOTE: Clock Detection of Oscillator MUST be Disabled Before Disabling HF LPO

The HF LPO frequency is the comparison frequency for the oscillator. The clock detection must be disabled prior to disabling the HF LPO frequency.

If the clock detection is NOT disabled prior to disabling the HF LPO, the clock detect circuitry will fail the oscillator as too fast (compared to the non-existent HF LPO). The clock detect circuitry will switch to the non-existent clock, leaving the device without a valid clock.

14.4.2 Behavior on Oscillator Failure

If the oscillator frequency fails, the clock detects supplies:

- the HF LPO clock to GCM clock source 0 instead of the oscillator
- the HF LPO clock to GCM clock source 1 instead of the PLL

The HF LPO signal will be available as three different clock sources:

- GCM clock source 0 (replacing the oscillator)
- GCM clock source 1 (replacing the PLL)
- GCM clock source 5 as HF LPO

The automatic switch-over from oscillator to HF LPO allows the application to execute at a reduced frequency and respond to a problem with the external crystal/resonator. During and after an oscillator failure, the oscillator CLKSRnV bit in the Clock Source Valid Status Register (CSVSTAT), of the System and Peripheral Control Registers, is set along with the OSCFAIL flag in the Global Status Register (GLBSTAT), of the System and Peripheral Control Registers.

It is useful to explicitly change the GHVSRC register, defining the current clock source for GCLK1/HCLK/VCLK domains, to the HF LPO after an oscillator failure.

When reset on oscillator failure is set, PLLCTL1.23 (ROF), the device responds to an oscillator failure by generating a device reset.

14.4.3 Recovery from Oscillator Failure

If the oscillator fails, the clock detect switches the HF LPO frequency onto the oscillator source into the GCM. The OSCFAIL flag in the Global Status Register (GLBSTAT) of the System and Peripheral Control Registers is also set.

The oscillator may be re-enabled (though if the failure was caused by a hard-fault, the re-enable will fail) through the following procedure:

1. Switch all clock domains from the oscillator to the HF LPO (for example, GHVSRC uses HF LPO, VCLKAn uses HF LPO or VCLK, and so on).
2. If the PLL is used, disable the PLL by setting the appropriate bit in the Clock Source Disable Set Register (CSDISSET) of the System and Peripheral Control Registers.
3. Disable the oscillator by setting the appropriate bit in the Clock Source Disable Set Register (CSDISSET). This action resets the clock detect and allows the oscillator to propagate through GCM clock source 0.
4. Re-enable the oscillator by setting the appropriate bit in the Clock Source Disable Clear Register (CSDISCLR) of the System and Peripheral Control Registers.
5. Clear the OSCFAIL flag in the Global Status Register (GLBSTAT) by writing a 1 to the bit. The PLL slip bits may also be set on an oscillator failure. These can also be cleared.
6. Switch the clock domains back to the oscillator.
7. Re-enable the PLL by setting the appropriate bit in the Clock Source Disable Clear Register (CSDISCLR).

NOTE: Clock Re-Enable Procedure Will Fail If Caused by a Hard Failure

Although it is possible to re-enable the oscillator after a failure, if the oscillator failure was caused by a hard fault (for example, disconnected crystal/resonator terminal), the re-enable process will fail.

14.4.4 LPOCLKDET Enable

The LPO is enabled by default while nPORRST is low. During this time, the current source initializes, holding the relaxation oscillator in reset until initialized. After the current source releases the HF LPO and the LF LPO, these clock frequencies slew to their final frequencies; the final frequency may be achieved while nPORRST is active or after its release. After, nPORRST is released, the HF LPO Valid signal is set 32 HF LPO clock cycles later.

The clock detect is enabled once the oscillator and HF LPO are valid. Because an oscillator failure could occur from reset, the clock detect logic must provide an override path. If the HF LPO is valid and the oscillator is not valid, the clock detect circuitry will become active (overriding the oscillator invalid signal) after 16K LF LPO cycles (about 200 ms).

14.4.5 LPOCLKDET Disable

14.4.5.1 Disable Clock Detect

It is possible to disable the clock detect circuitry. For protection, this clock detect disable employs a 2-bit key:

- RANGE DET ENA SSET (CLKTEST.24) must be set to 1
- RANGE DET CTRL (CLKTEST.25) must be cleared to 0

In this case, the LPO HF and LF clocks are still active but the clock detect circuitry is disabled. The clock detect unconditionally switches GCM_CLK_SRC(0) back to the oscillator so care should be taken to insure that the oscillator is good before disabling the clock detect circuitry.

14.4.5.2 Disable LPO HF and LF Clocks

The LPO may be disabled by holding the relaxation oscillator clocks (HF and LF) in reset. The clock detect must be disabled, and any clock domains using either HF or LF clocks must be switched to a different clock source. The LPO HF clock is reset by setting CSDIS.5; CSDISSET.5 is an easy way to set specific bits without disturbing the rest of the register. The HF LPO clock disables several HF LPO cycles after CSDIS is set.

Similarly, the LPO LF clock is reset by setting CSDIS.4, and in a similar way CSDISSET.4 can set the specific CSDIS register bit without using a read-modify-write construction. The LF LPO disables several LF LPO cycles after CSDIS is set.

Restarting the LPO clocks from this condition is fast and is known as a warm re-start. The CSDISCLR register allows the user to clear CSDIS bits without using a read-modify-write code-construct.

14.4.5.3 Disable LPO Current Bias

The LPO current source may be disabled after the clock detect is disabled and HF and LF clock sources are disabled. Turning off this current source places the LPOCLKDET into its lowest power configuration. The bias may be disabled by clearing the BIAS ENABLE bit (LPOMONCTL.24).

Restarting the LPO when the bias current has been disabled requires the current source to initialize first and is, therefore slower than a warm re-start; re-enabling the LPO from this condition is known as a warm re-start (similar to what happens during nPORRST active).

14.4.6 Trimming the HF LPO Oscillator

The HF LPO range varies considerably around 9.6MHz from device to device. In order to provide tighter monitoring of the crystal/resonator, it is useful to trim the oscillator. During device test, a trim value is written into the one-time programmable section of the flash memory (OTP), address 0xF008_01B4. Bits 31:16 of this OTP word contain a 16 bit value that may be programmed into LPOMONCTL(15:0) in order to initialize the trim for both HF LPO and LF LPO.

When trimming the HF LPO, it is recommended to step the trim value so as not to make a large change to any TRIM setting.

After the initial trim, further trimming may be done in LPOMONCTL, using the dual clock compare module (DCC, please see Dual Clock Compare User’s Guide) in order to determine the resultant frequency. This module allows for comparison of two clock frequencies. Once the HF LPO is determined to be in-range with the initial HFTRIM setting from the OTP, the crystal oscillator may be used as a reference against which the HF LPO and LF LPO may be further adjusted.

14.5 PLL

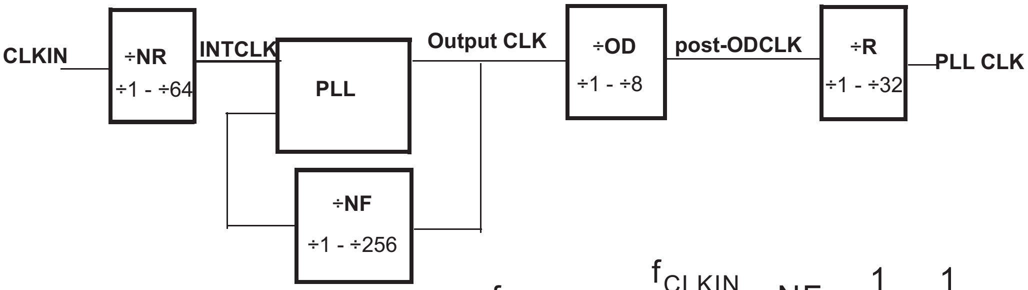
The following bit fields from PLLCTL1 and PLLCTL2 configure the PLL:

- REFCLKDIV[5:0]
- PLLMUL[15:0]
- ODPLL[2:0]
- PLLDIV[4:0]
- SPR_AMOUNT[8:0]
- SPREADINGRATE[8:0]
- FMENA

The PLL is responsible for synthesizing an output frequency from the input clock (from the oscillator); Figure 14-4 shows a simple block diagram of the PLL. The FM-PLL divides the reference input for a lower frequency input into the PLL ($f_{INTCLK} = f_{CLKIN}/NR$). The PLL multiplies this internal frequency by NF to get the VCO output clock frequency ($f_{Output\ CLK} = f_{INTCLK} \times NF$). The PLL output is subsequently divided by two prescale values (OD and R). The value of OD is an integer from 1-8 and R is an integer from 1 to 32. This output clock, PLL CLK, sources GCM clock source 1. Valid frequencies are shown in Table 14-1 while Table 14-2 shows how that encoding is generated from the PLL bit fields.

[$f_{(post\_ODCLK)}$ and $f_{(GCLK)}$ are data sheet parameters.]

Figure 14-4. Operation of the FM-PLL Module



$$f_{PLLCLK} = \frac{f_{CLKIN}}{NR} \times NF \times \frac{1}{OD} \times \frac{1}{R}$$

Table 14-1. Valid Frequency Ranges for PLL

	Frequency Limit
f_{CLKIN}	$f_{(OSC\_Sqr)}$
f_{INTCLK}	1MHz - $f_{(OSC\_Sqr)}$
$f_{Output\ CLK}$	150MHz - 550MHz
$f_{post\text{-}ODCLK}$	$f_{(post\_ODCLK)}$
$f_{PLL\ CLK}$	$f_{(GCLK)}$

Table 14-2. PLL Value Encoding

	PLL
NR	$NR = REFCLKDIV[5...0]+1$ (2)
NF	Non-modulated: $NF = \frac{(PLLMUL[15...0] + 256)}{256}$ (3) Modulated: $NF = \frac{(PLLMUL[15...0] + MULMOD[8...0] + 256)}{256}$ (4)
NV	$NV = \frac{(SPR\_AMOUNT[8...0]+1)}{2048}$ (5)
NS	$NS = SPRRATE[8...0]+1$ (6)
OD	$OD = ODPLL[2...0]+1$ (7)

NOTE: ODPLL change should occur prior to enabling asynchronous clock domains

Since changing the ODPLL bit-field causes the PLL CLK to be gated, these changes to ODPLL should be completed before configuring a clock domain for an asynchronous clock source. Some clock domains (RTICLK1, VCLK2) require a frequency relationship to the VCLK.

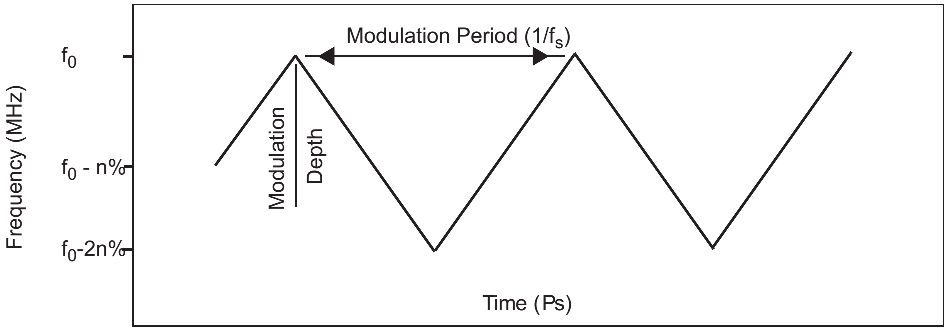
$$f_{VCLK} \geq 3 \times \frac{f_{RTISRC}}{RTIDIV}$$

If the PLL is clocking VCLK and it is stopped for some cycles, then the frequency relationship is temporarily violated.

Many asynchronous domains require frequency relationships between VCLK and the asynchronous domain. Therefore, if the PLL clock is the source for GCLK1, HCLK, and VCLK, then the gating produces a short-term change in the PLL clock frequency (and hence also the VCLK frequency). As such, this frequency change could violate the requirements for an asynchronous clock domain.

14.5.1 Modulation

Optionally, the frequency can be modulated, that is, a controlled jitter is introduced onto the baseline frequency of the PLL. This modulation mechanism is not shown in Figure 14-4. When the PLL is used in the modulating mode, the programmable modulation block varies the PLL frequency from the baseline frequency ($f_{baseline}$ = (f_{CLKIN}/NR) × NF/(OD × R)) to $f_{baseline}$ × (1 - 2 × Depth) in a period defined by 1/f_s; the modulation waveform is triangular and should be enabled after lock.



The modulation is digital and the spreading profile is triangular, down-spread which implies:

- the modulation waveform is composed of a series of frequency steps.
- the modulation frequency and modulation depth are both well controlled due to their digital character.
- the average frequency during modulation is lower than the average frequency prior to enabling modulation. The depth of modulation, however, sets the new average frequency.
- the modulation frequency must be selected slower than the loop bandwidth. From a practical perspective, NS should be near 20.

The modulation fields have a simple geometric meaning:

- the modulation step size is:

$$\frac{NV}{NF} \times f_{OutputCLK} \tag{8}$$

- the number of steps per modulation period is 2 × NS
- the modulation depth is given by:

$$\Delta f = \frac{NS}{2} \times \frac{NV}{NF} \times f_{OutputCLK}$$

$$Depth[\%] = \frac{NS}{2} \times \frac{NV}{NF} \tag{9}$$

- the modulation frequency is:

$$T_{\text{mod}} = \frac{f_{OSC}}{2 \times NR \times NS} \tag{10}$$

- MULMOD minimizes frequency offset when programmed as:

$$\frac{(SPR\_AMOUNT[8 \ldots 0] + 1)(SPRRATE[8 \ldots 0] + 1)}{16} \tag{11}$$

NOTE: Modulation should be enabled after Lock

Enable modulation after the lock is completed.

14.5.2 PLL Output Control

The outputs from the PLL are the output clock, slip signals and VALID.

- RFSLIP -- the RFSLIP signal indicates that the Output CLK is running *too fast* relative to INTCLK and sets a RFSLIP status flag in the Global Status Register (GLBSTAT), of the System and Peripheral Control Registers, if the slip signal is active during normal PLL operation; the RFSLIP flag is masked off while the PLL is not active and during the PLL's lock period.
- FBSLIP -- the FBSLIP signal indicates that the Output CLK is running *too slow* relative to INTCLK and sets a FBSLIP status flag in the Global Status Register (GLBSTAT), of the System and Peripheral Control Registers, if the slip signal is active during normal PLL operation; the FBSLIP flag is masked off while the PLL is not active and during the PLL's lock period.
- PLL Slip -- Logical-OR of the two PLL slip signals. Typically this signal is used to generate a consolidated slip signal to the device (for example, error logic or exception generation). Also used to gate VALID.

NOTE: Clearing Slip Bits

In order to clear any of the slip bits, it is necessary to disable the PLL first.

- VALID -- is driven based upon whether the output clock, PLL CLK, is gated or is not gated. However, the VALID signal is dependent upon the PLL Slip signals so that VALID cannot be set if either slip signal is active.
- PLL Clock -- The PLL output clock runs at the programmed frequency. When enabled, it takes some time to acquire the programmed frequency (see Section 14.5.2.1). Similarly, the disable has some timing and constraints (see Section 14.5.2.2).

14.5.2.1 PLL Enable

After setting the PLL control registers, the clock source is enabled by clearing the appropriate bit in the Clock Source Disable Register (CSDIS) or setting the appropriate bit in the Clock Source Disable Clear Register (CSDISCLR) of the System and Peripheral Control Registers. The bit sends a signal to the PLL that starts the process of enabling the PLL.

1. The PLL checks to make sure that the oscillator is ON. If not, it turns the oscillator ON.
2. The PLL begins a locking process in which the PLL slews from a starting frequency point to the programmed frequency. During this lock period, the PLL slip signals are typically active, and the PLL masks off the signals during this phase. The lock phase takes the following length of time:

Parameter	Value
Lock	$T_{Lock} = (512 \times T_{OSCIN}) + (1024 \times NR \times T_{OSCIN})$
Enable clocks after lock	$T_{Enable} = 6 \times T_{OSCIN}$

3. After the lock phase is complete (when lock counters expire), the PLL releases the slip signals to the system.
4. Then after the slip signals are released and a delay to enable the clocks, the clock is released to the system and the appropriate CLKSRnV bit for the PLL is set in the Clock Source Valid Status Register (CSVSTAT) of the System and Peripheral Control Registers.

14.5.2.2 PLL Disable

The clock sources (for example, OSC, PLL) are disabled by setting the appropriate bit in the Clock Source Disable Register (CSDIS) or setting the appropriate bit in the Clock Source Disable Set Register (CSDISSET) of the System and Peripheral Control Registers. These bit allow the clock to disable but do not force the behavior until the clock is no longer used as the source for a clock domain (for example, GCLK1, VCLK, VCLK2, RTICLK1).

The PLL receives a signal to disable after the clock is no longer used by any clock domain. Within the PLL, the clock is disabled and the appropriate CLKSRnV bit for the PLL in the Clock Source Valid Status Register (CSVSTAT), of the System and Peripheral Control Registers, becomes inactive. Then the PLL is placed into a low power state after the following length of time: $T_{Enable} = 150 \times T_{OSCIN}$

14.5.2.3 OD-Divider Change

The PLL gates the clock if the ODPLL bit-field is changed while the PLL is active. The output clock from the PLL is gated for 3 or 12 OSCIN clock cycles. As the post-ODCLK is gated in the low phase, the output clock to the device -- PLL CLK -- may be gated in a high or low phase though the transition is always glitchless: $T_{ODPLL} = 3 \times T_{OSCIN}$

NOTE: ODPLL change should occur prior to enabling asynchronous clock domains

Since changing the ODPLL bit-field causes the PLL CLK to be gated, these changes to ODPLL should be completed before configuring a clock domain for an asynchronous clock source. Some clock domains (RTICLK1, VCLK2) require a frequency relationship to the VCLK.

$$f_{VCLK} \geq 3 \times \frac{f_{RTISRC}}{RTIDIV}$$

If the PLL is clocking VCLK and it is stopped for some cycles, then the frequency relationship is temporarily violated.

Many asynchronous domains require frequency relationships between VCLK and the asynchronous domain. Therefore, if the PLL clock is the source for GCLK1, HCLK, and VCLK, then the gating produces a short-term change in the PLL clock frequency (and hence also the VCLK frequency). As such, this frequency change could violate the requirements for an asynchronous clock domain.

14.5.2.4 Changing the PLL Operating Point While the PLL is Active

Once the valid bit (CLKSRnV bit in the Clock Source Valid Status Register (CSVSTAT) of the System and Peripheral Control Registers) is set, software may change values to the PLL. If the change of values results in a small percentage change to the VCO frequency ($\Delta f_{OutputCLK} < 0.1 \times f_{OutputCLK}$), then these changes can be done on-the-fly. In this mode, the values are updated into the PLL synchronously, and the PLL re-locks to the new value without gating the clocks or the slip bits. If the operating point change is too large, then the slip bits will be set.

Conversely, if the changes to the VCO frequency are large, then the PLL should be disabled prior to changing the values. Typically, any change to the REFCLKDIV field or large changes to the PLLMUL field in the PLL Control Register 1 (PLLCTL1) of the System and Peripheral Control Registers requires a complete disable-and-relock strategy.

14.5.2.5 Summary of PLL Timings

In addition to controlling the lock period and disabling the clock during an ODPLL change, the PLL also generates reset delays. When power-on reset is released (nPORRST 0 --> 1), that release is delayed by 1024 OSCIN cycles so that it is released at the same time that the oscillator valid is asserted. The system reset release is delayed by an additional 8 oscillator clock cycles.

Table 14-3. Summary of PLL Timings

Parameter	Value
nPORRST delay	$T_{nPORRST}$ = 1024 x T_{OSCIN}
nRST delay	T_{nRST} = 1032 x T_{OSCIN}
OSC valid	$T_{OSCVALID}$ = 1024 x T_{OSCIN}
Lock	T_{Lock} = (512 x T_{OSCIN}) + (1024 x NR x T_{OSCIN})
Enable clocks after lock	T_{Enable} = 6 x T_{OSCIN}
Disable clocks after lock	T_{Enable} = 150 x T_{OSCIN}
Change ODPLL	T_{ODPLL} = 3 x T_{OSCIN}

14.5.3 Behavior on PLL Fail

The PLL allows flexible response to a PLL failure (slip). Like the oscillator, the PLL clock is configured by default to automatically switch-over to the oscillator in case of a PLL slip. (In this case, the oscillator sources GCM clock source 1 as well as GCM clock source 0. Also, if the oscillator fails, LPO HF is sourced to both GCM clock sources 0 and 1.)

The PLL slip outputs indicate that the PLL is running either too fast or too slow. These error output toggle when the PLL is locking and when the PLL is disabling. The PLL blocks these slip outputs during these times, leaving them active only while the PLL is active.

A slip after the PLL has locked and while it is active is an indication of a PLL failure. The PLL provides slip-filtering which enhances the flexibility of the PLL's response to failure. The slip-filtering circuit samples the slip based on HF LPO. The filter defines the number of consecutive HF LPO cycles for which the slip signal must be active before the slip is recognized. This slip is latched in the RFSLIP and FBSLIP status flags in the Global Status Register (GLBSTAT) of the System and Peripheral Control Registers.

The PLL may enable/disable the automatic switch over as well as the error signaling; if the error signaling is enabled, a PLL slip may be configured to generate a reset. The automatic switch-over and suppression of the error signals are controlled by the bypass on slip bit field -- BPOS[1:0] (PLLCTL1.(30:29)). When BPOS[1:0] is disabled (BPOS[1:0] = 10b):

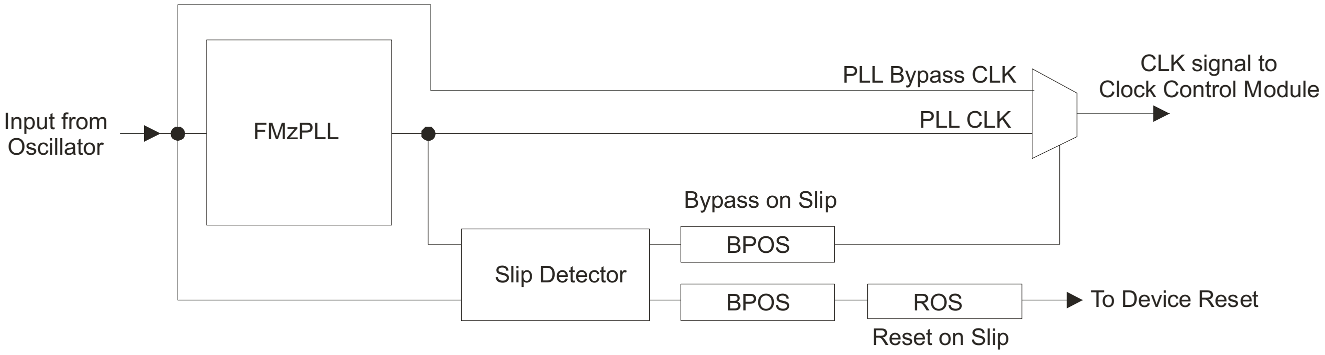
- automatic response to the PLL slip is prevented
- ESM/exception is NOT generated
- reset on slip is not generated regardless of the state of the ROS bit
- status bits are set on a PLL slip independent of BPOS[1:0]

When BPOS[1:0] is enabled (BPOS[1:0] = 00b OR 01b OR 11b):

- PLL slip causes the clock source into GCM clock source 1 to shift from the PLL to the oscillator
- ESM/exception is generated
- reset on slip is generated if ROS is set

The effect of BPOS[1:0] on the system is shown in Figure 14-5.

Figure 14-5. PLL Slip Detection and Reset/Bypass Block Diagram



14.5.4 Recovery from a PLL Failure

If PLL1 fails, the PLL's slip causes the valid flag to be locked and causes the clock source into GCM clock source 1 to shift from the PLL to the oscillator. The RFSLIP or FBSLIP status flags in the Global Status Register (GLBSTAT) of the System and Peripheral Control Registers are also set. PLL1 may be re-enabled (though if the failure was caused by a hard-fault, the re-enable will fail) through the following procedure:

1. Switch all clock domains from PLL1 to the oscillator (for example, GHVSRC uses oscillator, VCLKAn uses oscillator or VCLK, and so on).
2. Disable PLL1 by setting the appropriate bit in the Clock Source Disable Set Register (CSDISSET) of the System and Peripheral Control Registers. This action disables the PLL and causes the slip signal to no longer be driven. Valid is not released until the slip is cleared.
3. Clear the RFSLIP or FBSLIP status flags in the Global Status Register (GLBSTAT) of the System and Peripheral Control Registers by writing a 1 to the bit. After this step, the valid flag is unlocked and cleared if it was previously set.
4. Re-enable PLL1 by setting the appropriate bit in the Clock Source Disable Clear Register (CSDISCLR).
5. Switch the clock domains back to PLL1.

If PLL2 fails, the PLL's slip causes the valid flag to be locked. There is no autonomous change of clock source for PLL2. Neither the RFSLIP or FBSLIP status flags in the Global Status Register (GLBSTAT) of the System and Peripheral Control Registers are set. PLL2 may be re-enabled in a similar procedure to re-enabling PLL1 (though if the failure was caused by a hard-fault, the re-enable will fail):

1. Switch all clock domains from PLL2 to the oscillator (for example, GHVSRC uses oscillator, VCLKAn uses oscillator or VCLK, and so on).
2. Disable PLL2 by setting the appropriate bit in the Clock Source Disable Set Register (CSDISSET) of the System and Peripheral Control Registers. This action disables the PLL and causes the slip signal to no longer be driven. Valid is not released until the slip is cleared.
3. Reset PLL2 Valid by writing a 1 to both RFSLIP and FBSLIP status flags in the Global Status Register (GLBSTAT) of the System and Peripheral Control Registers (even though they are not set by the slip). After this step, the valid flag is unlocked and cleared if it was previously set.
4. Re-enable PLL2 by setting the appropriate bit in the Clock Source Disable Clear Register (CSDISCLR).
5. Switch the clock domains back to PLL2.

14.5.5 PLL Modulation Depth Measurement

The PLL contains a circuit for estimating the depth of the modulation. The circuit counts clock edges over a fixed window of the modulation waveform (SSW_CAPTURE_COUNT in SSWPLL2) and clock edges over the entire waveform (SSW_CLKOUT_COUNT in SSWPLL3). The capture ends after a pre-determined number of clock edges in SSW_CLKOUT_COUNTER as set in TAP_COUNTER_DIS. There are 2 × NR windows per modulation waveform. The procedure for estimating the modulation depth is:

1. While GCLK1 is sourced by the oscillator and the PLL is enabled with modulation, configure SSWPLL1 as follows:
   a. CAPTURE_WINDOW_INDEX is set equal to NR.
   b. COUNTER_RESET is set.
   c. TAP_COUNTER_DIS is set to disable the measurement after SSW_CLKOUT_COUNT captures this number of clocks. The measurement is disabled after the set tap is set AND the modulation cycle ends.
   d. Ensure that EXT_COUNTER_EN is cleared.
2. Ensure that both SSW_CAPTURE_COUNT and SSW_CLKOUT_COUNT are cleared (by the COUNTER_RESET).
3. Set COUNTER_EN and clear COUNTER_RESET. This step releases the reset and enables the counter to begin counting.
4. After a wait loop, poll for COUNTER_READ_READY to set. After the bit is set, read SSW_CAPTURE_COUNT and SSW_CLKOUT_COUNT.
5. Compute the modulation depth as:

$$Depth = abs\left(1 - \frac{2 \times NR \times SSW\_CAPTURE\_COUNT}{SSW\_CLKOUT\_COUNT}\right) \tag{12}$$

14.5.6 PLL Frequency Measurement Circuit

The same circuit that is used to measure modulation depth is also available to measure the average frequency of the PLL. In this mode, the PLL output (before the R-divider) is captured in SSW_CLKOUT_COUNT while the oscillator is captured in SSW_CAPTURE_COUNT. The procedure for using the PLL frequency measurement circuit is:

1. While the PLL is enabled, set EXT_COUNTER_EN.
2. Set COUNTER_EN. This bit clears both SSW_CAPTURE_COUNT and SSW_CLKOUT_COUNT and then immediately enables for counting.
3. Wait for some software delay loop.
4. Clear COUNTER_EN. Wait for COUNTER_READ_READY to set. Read both SSW_CAPTURE_COUNT and SSW_CLKOUT_COUNT and compute the ratio of PLL multiplication as:

$$\frac{NF}{NR \times OD} = \frac{SSW\_CLKOUT\_COUNT}{SSW\_CAPTURE\_COUNT} \tag{13}$$

5. Note that CAPTURE_WINDOW_INDEX, COUNTER_RESET, TAP_COUNTER_DIS are not used in this procedure

14.5.7 PLL2

PLL2 drives GCM clock source 6.

The PLL is identical to PLL1, except modulation is disabled on this instance of the PLL. Also, the PLL typically does not clock the system, there is no automatic switch over feature. Any PLL error can be handled by the CPU.

PLL2 is programmed through PLLCTL3.

14.6 PLL Control Registers

The clock module has two registers (PLLCTL1 and PLLCTL2) located within the System and Peripheral Control Registers, plus it has four bits located in other System and Peripheral Control Registers.

The FM-PLL is off at power-on. The clock source is enabled by clearing the appropriate bit in the Clock Source Disable Register (CSDIS) or setting the appropriate bit in the Clock Source Disable Clear Register (CSDISCLR) of the System and Peripheral Control Registers. [CSDISCLR and Clock Source Disable Set Register (CSDISSET) also enable/disable the PLL and oscillator (and other clock sources).]

The LPOCLKDET module generates the OSCFAIL flag in the Global Status Register (GLBSTAT), of the System and Peripheral Control Registers, if a problem with the reference oscillator is detected. The slip signals are also registered in the RFSLIP and FBSLIP status flags in the Global Status Register (GLBSTAT), of the System and Peripheral Control Registers, in order to indicate the source of a clock failure.

The appropriate CLKSRnV bit for the PLL is set in the Clock Source Valid Status Register (CSVSTAT) of the System and Peripheral Control Registers.

The following sections describe the two PLL registers used in the system module. These registers support 8-, 16-, and 32-bit write accesses. The reset values for these registers are configured so that an input frequency in the range from 5 MHz to 20 MHz generates a valid clock.

Table 14-4. PLL Module Registers

Address	Acronym	Register Description	Section
FFFF FF30h	CSDIS	Clock Source Disable Register	Section 2.5.1.10
FFFF FF34h	CSDISSET	Clock Source Disable Set Register	Section 2.5.1.11
FFFF FF38h	CSDISCLR	Clock Source Disable Clear Register	Section 2.5.1.12
FFFF FF54h	CSVSTAT	Clock Source Valid Status Register	Section 2.5.1.19
FFFF FF70h	PLLCTL1	PLL Control 1 Register	Section 2.5.1.25
FFFF FF74h	PLLCTL2	PLL Control 2 Register	Section 2.5.1.26
FFFF E100h	PLLCTL3	PLL Control 3 Register	Section 2.5.2.1
FFFF FFA0h	GPREG1	General Purpose Register	Section 2.5.1.34
FFFF FFECh	GLBSTAT	Global Status Register	Section 2.5.1.48
FFFF E170h	CLKSLIP	Clock Slip Control Register	Section 2.5.2.7
FFFF FF24h	SSWPLL1	PLL Modulation Depth Measurement Control Register	Section 14.6.1
FFFF FF28h	SSWPLL2	SSW PLL BIST Control Register 2	Section 14.6.2
FFFF FF2Ch	SSWPLL3	SSW PLL BIST Control Register 3	Section 14.6.3

Table 14-5. LPOCLKDET Module Registers

Address	Acronym	Register Description	Section
FFFF FF88h	LPOMONCTL	LPO/CLock Monitor Control Register	Section 2.5.1.31
FFFF FF8Ch	CLKTEST	Clock Test Register	Section 2.5.1.31

14.6.1 PLL Modulation Depth Measurement Control Register (SSWPLL1)

Figure 14-6 illustrates this register and Table 14-6 provides the bit descriptions. This register applies to PLL1, but does not apply to PLL2.

Figure 14-6. SSW PLL BIST Control Register 1 (SSWPLL1) [offset = 24h]

31 ... 16
Reserved
R-0

15 ... 8
CAPTURE_WINDOW_INDEX
R/W-0

7	6	5	4	3 ... 1	0
Reserved	COUNTER_READ_ READY	COUNTER_ RESET	COUNTER_EN	TAP_COUNTER_DIS	EXT_COUNTER_ EN
R-0	R-0	R/W-1	R/W-0	R/W-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 14-6. SSW PLL BIST Control Register 1 (SSWPLL1) Field Descriptions

Bit	Field	Value	Description
31-16	Reserved	0	Reads return 0. Writes have no effect.
15-8	CAPTURE_WINDOW_INDEX	0-FFh	The capture counter present in the PLL wrapper will count the PLL clock edges when the current modulation phase capture window value is equal to these bits. Should be set equal to NR.
7	Reserved	0	Reads return 0. Writes have no effect.
6	COUNTER_READ_READY		Counter read ready. Indicates that SSW_CAPTURE_COUNT (SSWPLL2) and SSW_CLKOUT_COUNT (SSWPLL3) can be read.
		0	Counter registers in SSWPLL2 and SSWPLL3 are not ready to read.
		1	Counter registers in SSWPLL2 and SSWPLL3 are ready to read.
5	COUNTER_RESET		Counter reset. If EXT_COUNTER_EN = 0, COUNTER_RESET resets SSW_CAPTURE_COUNT (SSWPLL2) and SSW_CLKOUT_COUNT (SSWPLL3). If EXT_COUNTER_EN = 1, this bit is ignored.
		0	No impact to counters.
		1	If the EXT_COUNTER_EN bit is 0, then counters SSW_CAPTURE_COUNT and SSW_CLKOUT_COUNT will be held in the reset state. If EXT_COUNTER_EN bit is 1, then this bit will be ignored by the PLL wrapper.
4	COUNTER_EN		Counter enable. If EXT_COUNTER_EN = 0, COUNTER_EN initializes the modulation depth measurement. (In this mode, the disable is set to occur automatically.) If EXT_COUNTER_EN = 1, the counters are enabled/disabled with COUNTER_EN.
		0	If EXT_COUNTER_EN = 0, COUNTER_EN = 0 indicates that the counters are inactive. If EXT_COUNTER_EN = 1, COUNTER_EN = 0 disables the counters.
		1	If EXT_COUNTER_EN = 0, COUNTER_EN = 1 indicates that the counters are still active. If EXT_COUNTER_EN = 1, COUNTER_EN = 1 enables the counters.

Table 14-6. SSW PLL BIST Control Register 1 (SSWPLL1) Field Descriptions (continued)

Bit	Field	Value	Description
3-1	TAP_COUNTER_DIS		The value in this register is used to program a particular bit in CLKOUT counter. When that particular bit in CLKOUT counter becomes 1, then both the CLKOUT counter and the CAPTURE counter will stop counting when EXT_COUNTER_EN = 0. When EXT_COUNTER_EN = 1, this bit field is not used.
		0	Bit 16 of CLKOUT counter is selected. When this bit is set and the modulation period finishes, the counters are disabled and READ_READY_FLAG is set.
		1h	Bit 18 of CLKOUT counter is selected.
		2h	Bit 20 of CLKOUT counter is selected.
		3h	Bit 22 of CLKOUT counter is selected.
		4h	Bit 24 of CLKOUT counter is selected.
		5h	Bit 26 of CLKOUT counter is selected.
		6h	Bit 28 of CLKOUT counter is selected.
		7h	Bit 30 of CLKOUT counter is selected.
0	EXT_COUNTER_EN		Measurement mode.
		0	Modulation Depth Measurement mode.
		1	Frequency Measurement mode.

14.6.2 SSW PLL BIST Control Register 2 (SSWPLL2)

This is an observation register used to log counter value for the capture counter inside the PLL wrapper. The SSWPLL2 register is shown in Figure 14-7 and described in Table 14-7. This register applies to PLL1, but does not apply to PLL2.

Figure 14-7. SSW PLL BIST Control Register 2 (SSWPLL2) [offset = 28h]

31	16
SSW_CAPTURE_COUNT	
R-0	

15	0
SSW_CAPTURE_COUNT	
R-0	

LEGEND: R = Read only; -*n* = value after reset

Table 14-7. SSW PLL BIST Control Register 2 (SSWPLL2) Field Descriptions

Bit	Field	Description
31-0	SSW_CAPTURE_COUNT	Capture count. This register returns the value of the capture count. When EXT_COUNTER_EN = 0, this counter increments within a fixed modulation window. When EXT_COUNTER_EN = 1, this counter increments based upon the oscillator.

14.6.3 SSW PLL BIST Control Register 3 (SSWPLL3)

This is observation register used to log counter value for CLKOUT counter inside PLL wrapper. The SSWPLL3 register is shown in Figure 14-8 and described in Table 14-8. This register applies to PLL1, but does not apply to PLL2.

Figure 14-8. SSW PLL BIST Control Register 3 (SSWPLL3) [offset = 2Ch]

31 16
SSW_CLKOUT_COUNT
R-0

15 0
SSW_CLKOUT_COUNT
R-0

LEGEND: R = Read only; -*n* = value after reset

Table 14-8. SSW PLL BIST Control Register 3 (SSWPLL3) Field Descriptions

Bit	Field	Description
31-0	SSW_CAPTURE_COUNT	Value of CLKout count register. This counter increments based upon the PLL output (prior to the R-divider).

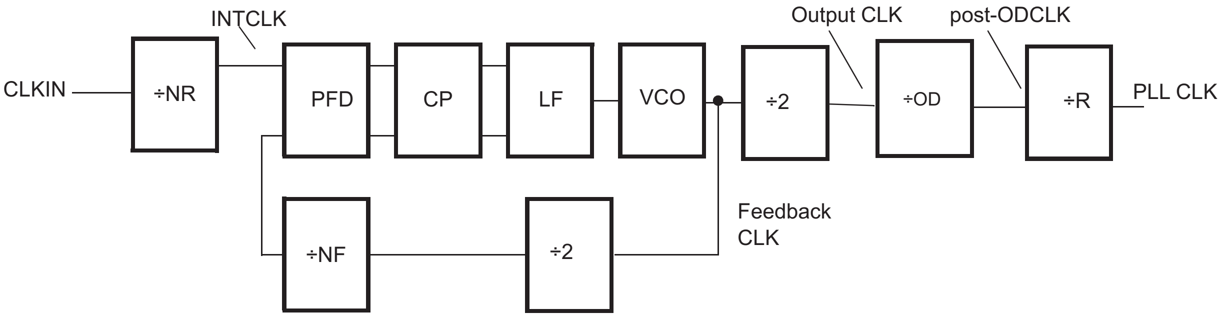
14.7 Phase-Locked Loop Theory of Operation

The PLL block consists of six logical sub-blocks:

- Phase-Frequency Detector (PFD)
- Charge Pump (CP)
- Loop Filter (LF)
- Voltage-Controlled Oscillator (VCO)
- Frequency Modulation
- Slip Detector

Figure 14-9 illustrates the sub-blocks in a basic PLL circuit. The VCO adjusts its frequency until the two signals into the PFD have the same phase and frequency. The feedback path (from VCO to PFD) divides the frequency of the feedback signal by 2 × NF; this feedback divider requires the VCO to generate a frequency 2 × NF times greater than the internal frequency (OSCIN/NR). In the forward path (from VCO to PLL CLK), the /2 block creates a clean duty cycle.

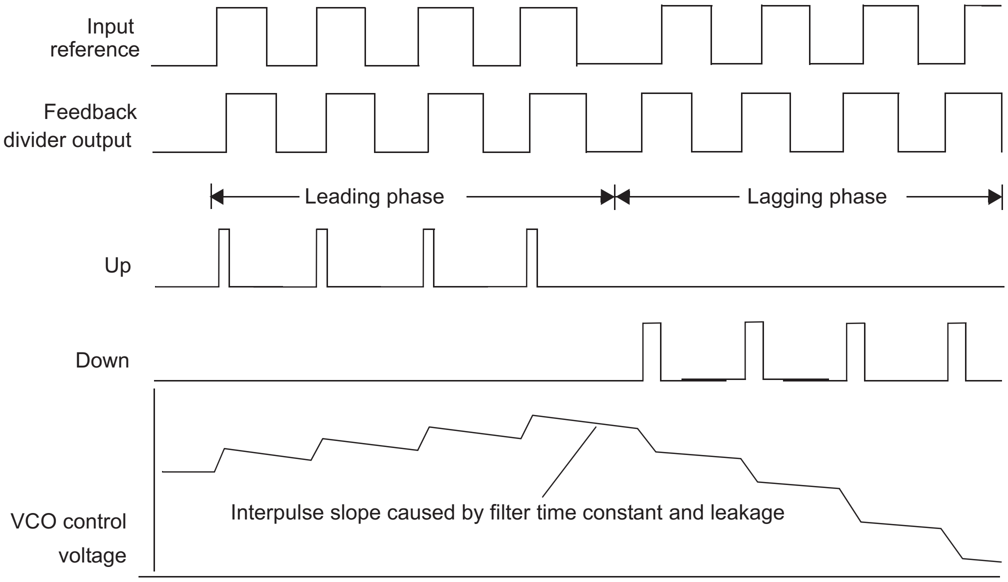
Figure 14-9. Basic PLL Circuit



14.7.1 Phase-Frequency Detector

The phase-frequency detector (PFD) compares the input reference phase/frequency to the phase/frequency of the feedback divider and generates two signals: an *up* pulse and a *down* pulse that drive a charge pump. The resulting charge, when integrated by the circuit at the LF pin, provides a VCO control voltage, as shown in Figure 14-10.

Figure 14-10. PFD Timing



The width of the up pulse and the down pulse depends on the difference in phase between the two inputs. For example, when the reference input leads the feedback input by 10 ns, then an up pulse of approximately 10 ns is generated (see Figure 14-10). On the other hand, when the reference input lags the feedback input by 10 ns, then a down pulse of approximately 10 ns is generated. When the two inputs are exactly in phase, the up pulse and down pulse become essentially zero-width. These pulses are fed to the charge pump block, which meters charge into the low-pass loop filter.

The advantage of a phase-frequency detector over a phase-only detector is that it cannot lock to a harmonic or subharmonic of the reference. This important property also ensures that the output frequency of the VCO is always exactly 2 × NF times the reference frequency.

The reference feedback frequency is based upon the VCO frequency and the feedback divider. Fractional multiplication is achieved by changing the feedback divider real-time in order to create the fractional multiplication. As an example, if a multiplier of 100.5 is selected, the feedback divider divides by 100 and 101 in equal proportions; in this case, the PLLMUL bit field would be programmed as 99.5 (0x6380). This fractional multiplication is useful when trying to achieve final frequencies that are non-integer to the input frequency (a final frequency that is a prime number). The fractional portion of the divider should be small compared to the multiplier and so it is recommended that the fractional portion relate to parts in 16, implying that the last 4 bits should always be 0.

14.7.2 Charge Pump and Loop Filter

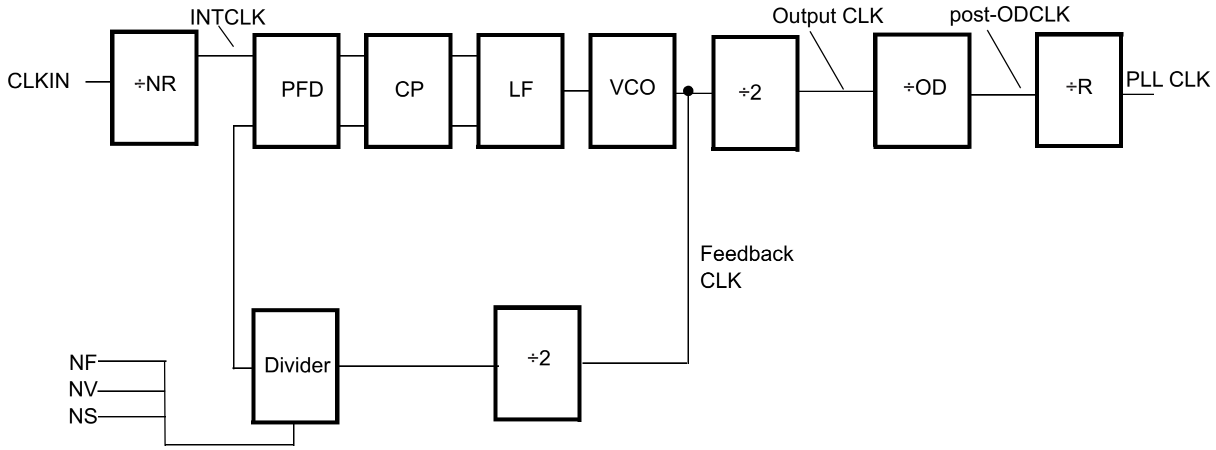
The charge pump (CP) add or remove charge from the loop filter based on the pulses coming from the phase-frequency detector (PFD).

Two components of the filter output signal are summed together: an integral component and a proportional component. The integral component maintains a DC level going to the VCO to set its frequency, and the proportional component makes the VCO track changes in phase to minimize jitter. The capacitors and resistors required for the filter are integrated in silicon.

14.7.3 Voltage-Controlled Oscillator

The output frequency of the VCO is proportional to its input control voltage, which is generated by the charge pump via the integrated loop filter. If the VCO oscillates too slowly, the feedback phase begins to lag the reference phase at the PFD, which increases the control voltage at the VCO. Conversely, if the VCO oscillates too fast, the feedback phase begins to lead the reference phase at the PFD, which decreases the control voltage at the VCO. These two actions keep the VCO running at the correct frequency multiple of the reference.

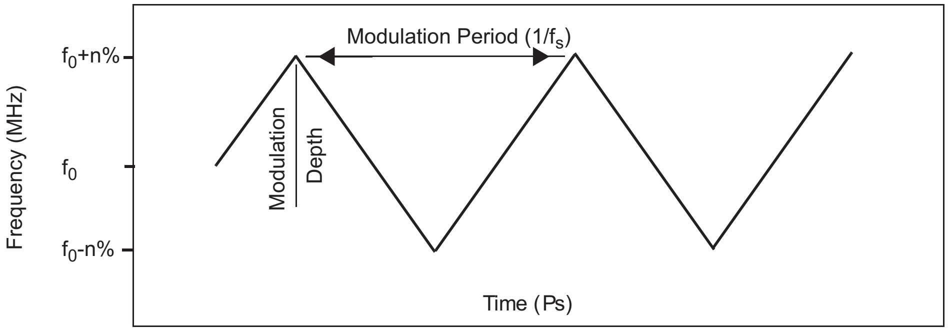
Figure 14-11. PLL Modulation Block Diagram



14.7.4 Frequency Modulation

The output clock of the PLL changes frequency in a controlled way, centered around the unmodulated output frequency. The modulation block directly modulates the VCO frequency at the loop filter, and creates the triangular frequency modulation (see Figure 14-12).

Figure 14-12. Frequency versus Time



14.8 Programming Example

This section provides an example of how to program the PLL. For non-modulation settings, the PLLCTL1 and PLLCTL2 settings from 130nm process devices can be used without modification.

Suppose that, using a 20 MHz crystal, the application requires:

- 180 MHz GCLK1 (and HCLK) frequency
- 100 kHz spreading frequency
- 0.5% spreading depth

1. Choose an NR and NS such that:
   - $$\frac{f_{CLKIN}}{NR \times f_s} \geq 40 \quad (14)$$
   - $$f_s \equiv \frac{f_{CLKIN}}{2 \times NR \times NS} \quad (15)$$
   - $$2 \times NS = \frac{f_{CLKIN}}{NR \times f_s} \geq 40 \quad (16)$$
   - (NR,NS) = {(5,20), (4,25), (2,50), (1,100)}
   - Either NR = 5 and NS = 20 or NR = 4 and NS = 25 are reasonable. Another choice (NR = 3 and NS = 33) is possible, if the modulation frequency can vary from 100 KHz.
2. Choose Output CLK frequency as integer divider of output frequency near to 330 MHz. Output CLK frequency shall not exceed 550 MHz or fall below 150 MHz.

   The integer values for 180 MHz are 360 MHz or 540 MHz. 360 MHz is close to the target frequency of 330 MHz and we use this frequency.
3. In this case, either of the following equations are suitable choices for getting to 360 MHz. Choose NR = 5, NS = 20 and set NF = 90 or choose NR = 4, NS = 25 and set NF = 72.

   $$\frac{f_{CLKIN}}{NR} = \frac{20[MHz]}{4} = 5[MHz] \;\; or \;\; \frac{f_{CLKIN}}{NR} = \frac{20[MHz]}{5} = 4[MHz] \quad (17)$$
4. Select the output divider OD so that the post-ODCLK frequency does not exceed the maximum frequency of output divider R (device-specific frequency). In this case, choose OD = 2 and R = 1.

5. Compute the divider value NV:

$$Depth = \frac{0.5}{100} = \frac{NV}{NF} \times \frac{NS}{2} = \frac{NV}{90} \times \frac{20}{2} \tag{18}$$

   NV = 0.045

6. If it is important to maintain the same average frequency in modulation as in non-modulation, either NF should be modified OR program the MULMOD bit field. The modulation fields create a multiplier offset equal to:

$$\Delta NF = \frac{NV \times NS}{2} \tag{19}$$

   If using MULMOD[8:0], then:

$$\Delta NF = \frac{MULMOD[8...0]}{256} = \frac{NV \times NS}{2} = \frac{0.045 \times 20}{2} \tag{20}$$

$$MULMOD[8...0] = \frac{0.045 \times 20}{2} \times 256 = 115.2 \tag{21}$$

   MULMOD will be set to 115.

7. Convert the PLL parameters into bit field values:
   - NR = 5, implies that REFCLKDIV[5:0] = 4
   - NS = 20, implies that SPRATE[8:0] = 19 = 0x13
   - NF = 90, implies that PLLMUL[15:0] = 0x5900
   - OD = 2, implies that ODPLL[2:0] = 1
   - R = 1, implies that PLLDIV[4:0] = 0
   - NV = 0.045, implies that SPR_AMOUNT[8:0] = 91 = 0x5B
   - MULMOD[8:0] = 115 = 0x73

8. Setting only these fields (that is, not BPOS, ROF, or ROS) yields:

   PLLCTL1 = 0x00045900

   PLLCTL2 = 0x04C7325B

   When FM ENA is turned on, PLLCTL2 = 0x84C7325B.

   The Output CLK is centered in the range from 150 MHz to 550 MHz at 360 MHz.

   NF = 90 falls within the multiplier range from 1 to 256.

   OD is selected so that post-ODCLK meets the device specification.

Dual-Clock Comparator (DCC) Module

This chapter describes the dual-clock comparator (DCC) module.

Topic **Page**

15.1 Introduction

The primary purpose of a DCC module is to measure the frequency of a clock signal using a second known clock signal as a reference. This capability can be used to ensure the correct frequency range for several different device clock sources, thereby enhancing the system safety metrics.

15.1.1 Main Features

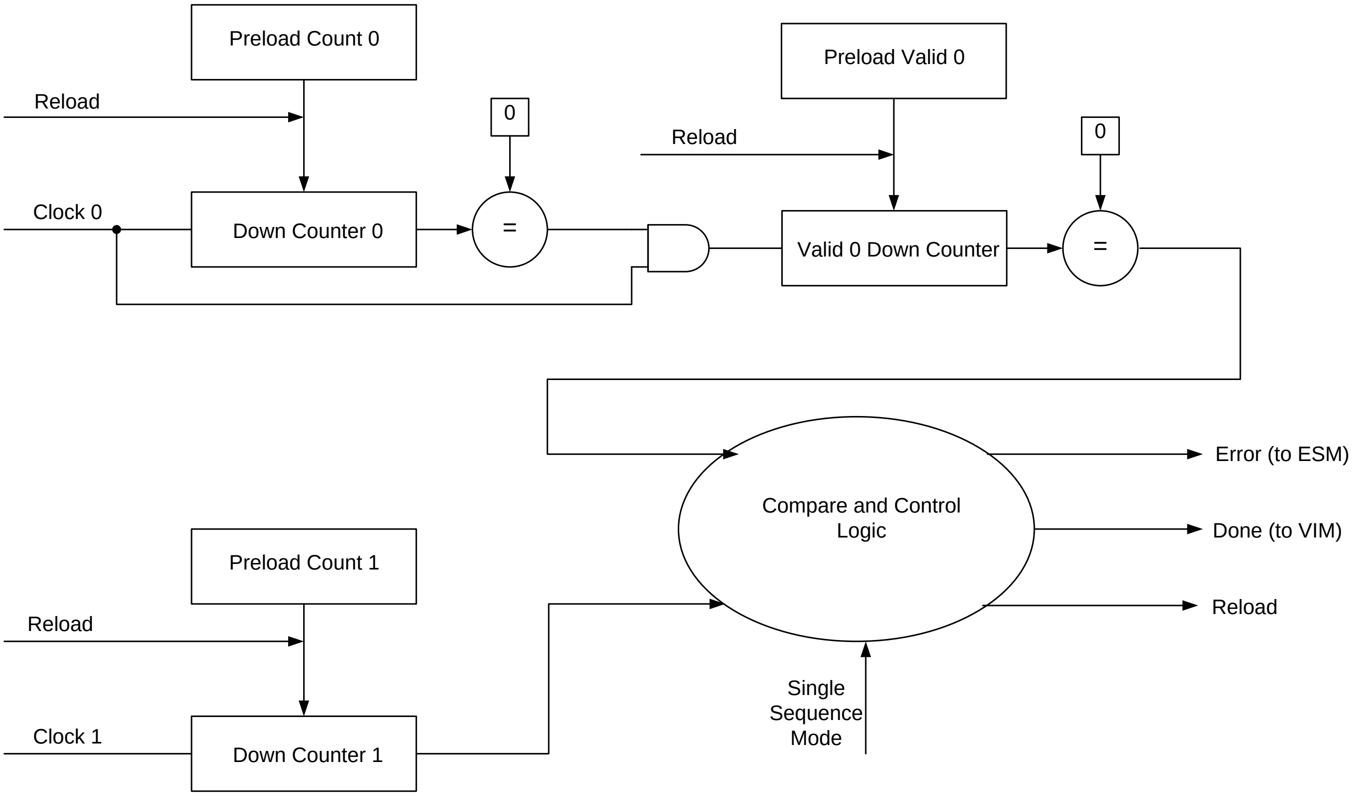
The main features of each of the DCC modules are:

- Allows application to ensure that a fixed ratio is maintained between frequencies of two clock signals
- Supports the definition of a programmable tolerance window in terms of number of reference clock cycles
- Supports continuous monitoring without requiring application intervention
- Also supports a single-sequence mode for spot measurements
- Allows selection of clock source for each of the counters resulting in several specific use cases

15.1.2 Block Diagram

Figure 15-1 illustrates the main concept of the DCC module.

Figure 15-1. DCC Operation



15.2 Module Operation

As shown in Figure 15-1, the DCC contains two counters – counter0 and counter1, which are driven by two signals – clock0 and clock1. The application programs the seed values for both these counters. The application also configures the tolerance window time by configuring the valid counter for clock0.

Counter0 and counter1 both start counting simultaneously once the DCC is enabled. When counter0 counts down to zero, this automatically triggers the count down of the tolerance window counter (valid0).

The DCC module can be used in two different operating modes:

15.2.1 Continuous Monitoring Mode

In this mode, the DCC is used by the application to ensure that two clock signals maintain the correct frequency ratio. Suppose the application wants to ensure that the PLL output signal (clock source # 1) always maintains a fixed frequency relationship with the main oscillator (clock source # 0).

- In this case, the application can use the main oscillator as the clock0 signal (for counter0 and valid0) and the PLL output as the clock1 (for counter1).
- The seed values of counter0, valid0 and counter1 are selected such that if the actual frequencies of clock0 and clock1 are equal to their expected frequencies, then the counter1 will reach zero either at the same time as counter0 or during the count down of the valid0 counter.
- If the counter1 reaches zero during the count down of the valid0 counter, then all the counters (counter0, valid0, counter1) are reloaded with their initial seed values once valid0 has also counted down to zero.
- This sequence of counting down and checking then continues as long as there is no error, or until the DCC module is disabled.
- The counters also all get reloaded if the application resets and restarts the DCC module.

Error Conditions:

An error condition is generated by any one of the following:

1. Counter1 counts down to 0 before Counter0 reaches 0. This means that clock1 is faster than expected, or clock0 is slower than expected. It includes the case when clock0 is stuck at 1 or 0.
2. Counter1 does not reach 0 even when Counter0 and Valid0 have both reached 0. This means that clock1 is slower than expected. It includes the case when clock1 is stuck at 1 or 0.

   Any error freezes the counters from counting. An application may then read out the counter values to help determine what caused the error.

15.2.1.1 Error Conditions

While operating in continuous mode, the counters get reloaded with the seed values and continue counting down under the following conditions:

- The module is reset or restarted by the application, OR
- Counter0, Valid 0 and Counter1 all reach 0 without any error

Figure 15-2. Counter Relationship

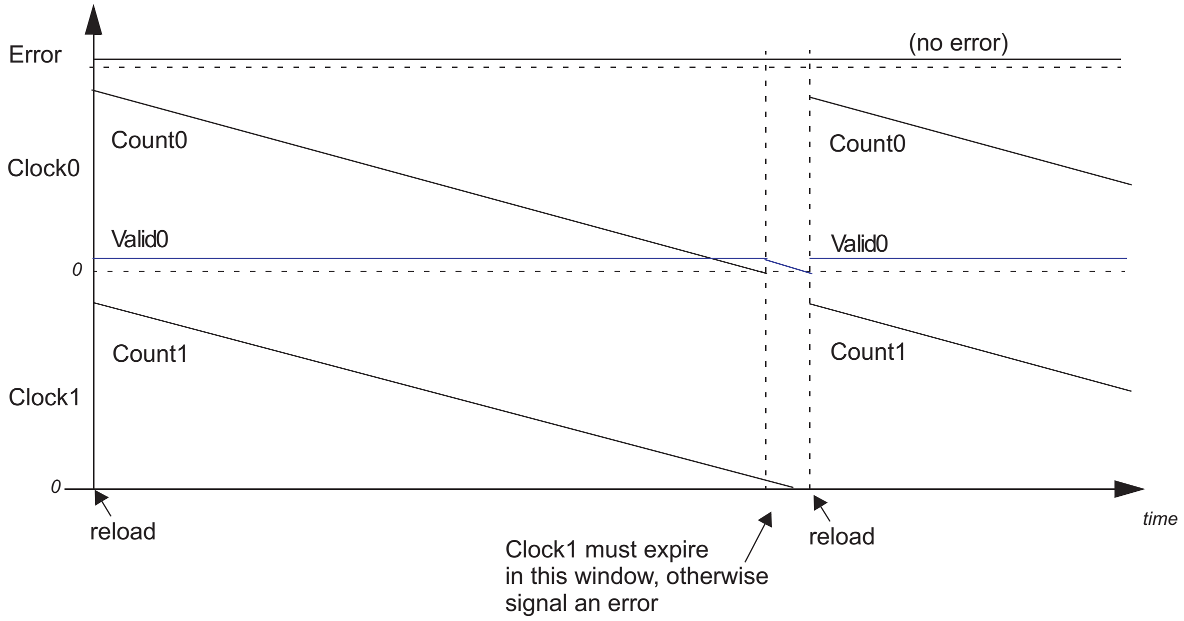


Figure 15-3. Clock1 Slower Than Clock0 - Results in an Error and Stops Counting

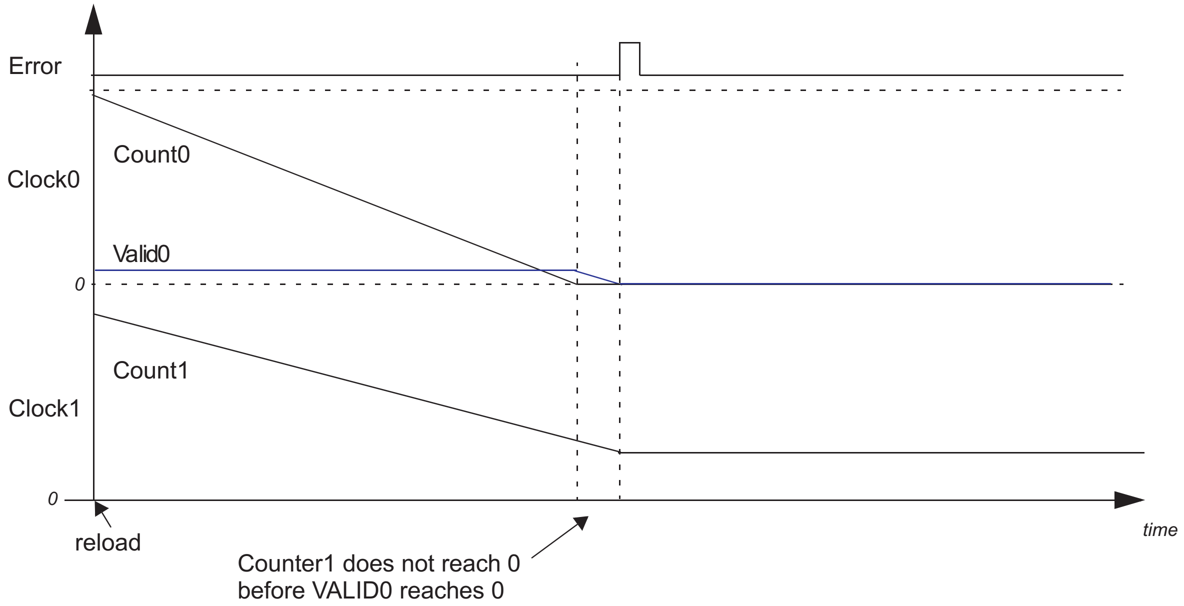


Figure 15-4. Clock1 Faster Than Clock0 - Results in an Error and Stops Counting

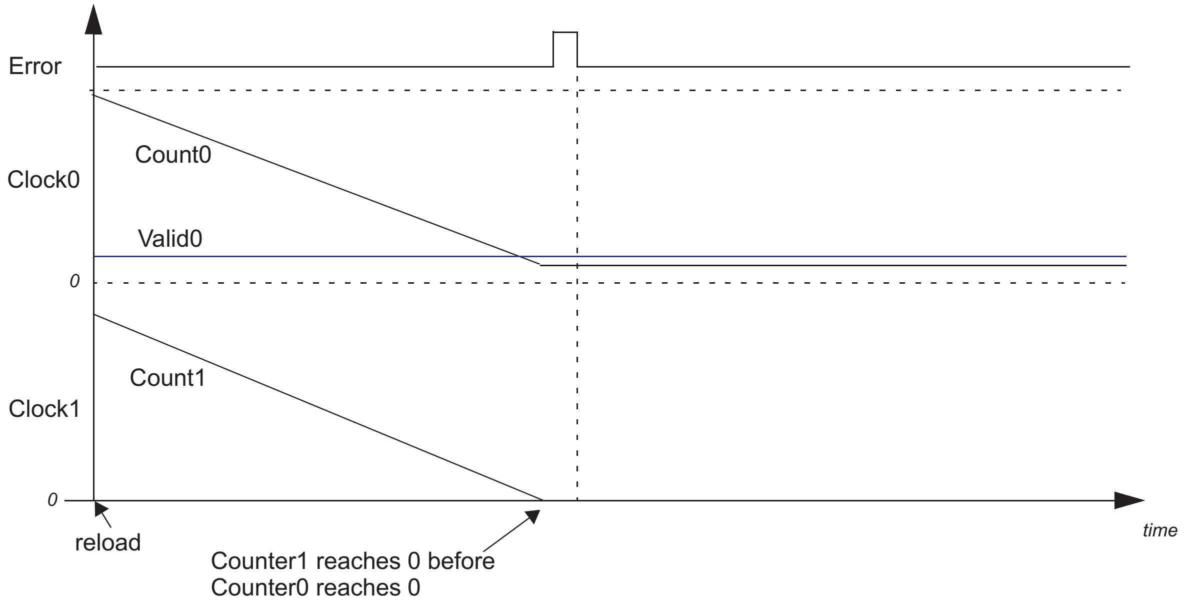


Figure 15-5. Clock1 Not Present - Results in an Error and Stops Counting

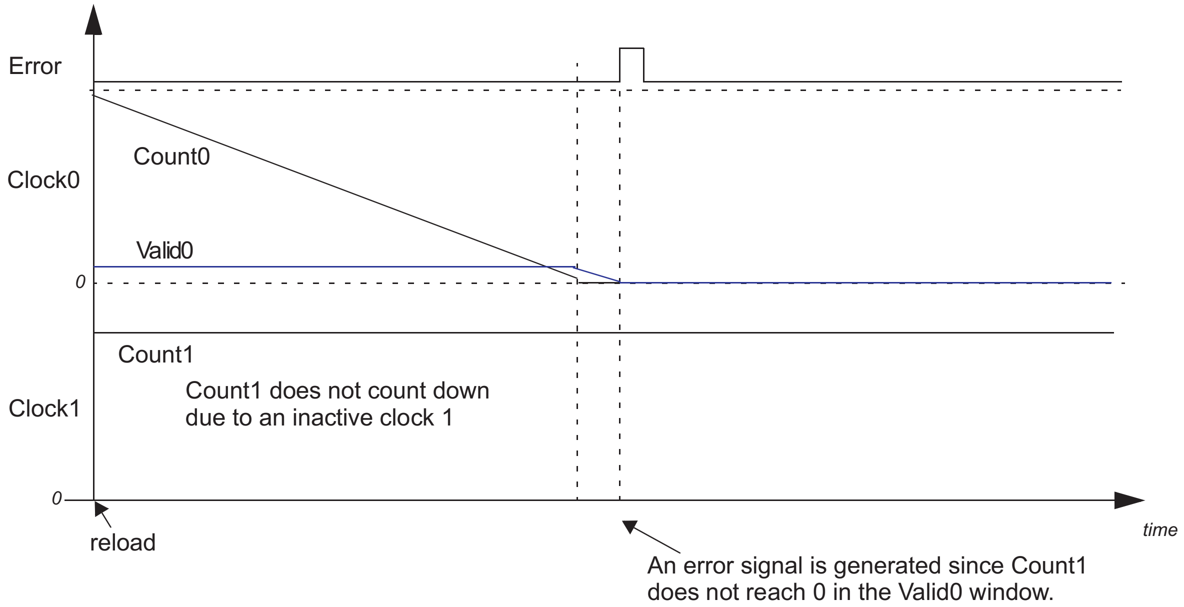
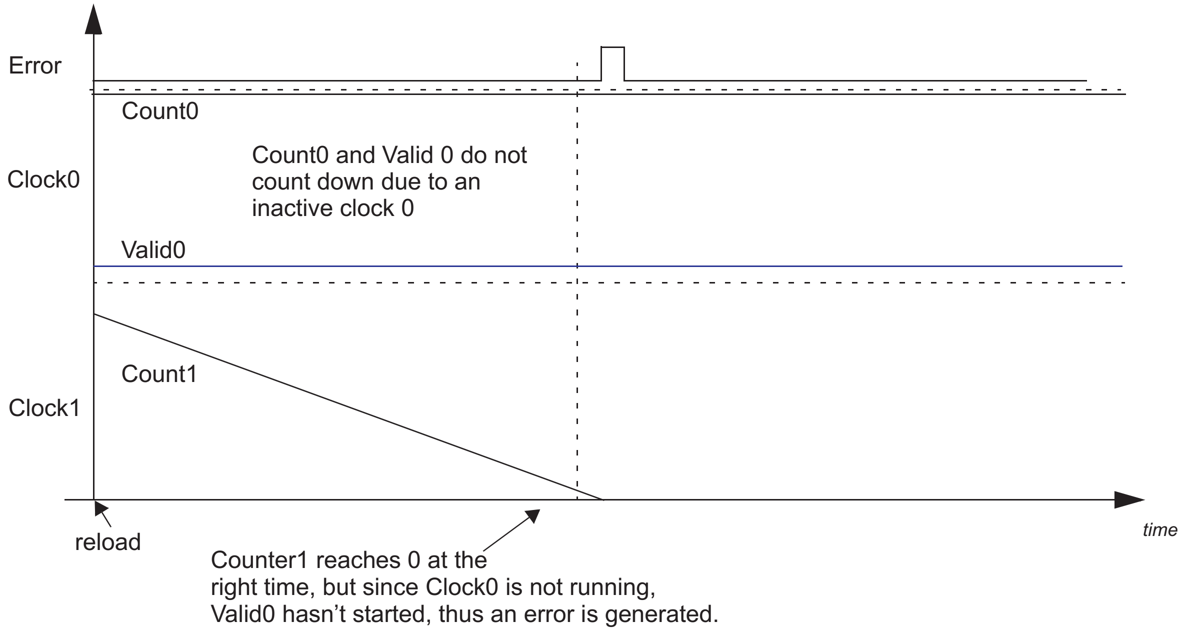

Error
Clock0
Count0
Valid0
0
Count1
Count1 does not count down
due to an inactive clock 1
Clock1
0
reload
time
An error signal is generated since Count1
does not reach 0 in the Valid0 window.

Figure 15-6. Clock0 Not Present - Results in an Error and Stops Counting


Error
Count0
Clock0
Count0 and Valid 0 do not count down due to an inactive clock 0
Valid0
Count1
Clock1
reload
time
Counter1 reaches 0 at the right time, but since Clock0 is not running, Valid0 hasn't started, thus an error is generated.

15.2.2 Single-Shot Measurement Mode

The DCC module can be programmed to count down one time by enabling the single-shot mode. In this mode, the DCC stops operating when the down counter0 and the valid counter0 reach 0. Alternatively, the DCC can be programmed to stop counting when the down counter1 reaches 0.

At the end of one sequence of counting down in this single-shot mode, the DCC gets disabled automatically, which prevents further counting. This mode is typically used for spot measurements of the frequency of a signal. This frequency could be an unknown for the application before the measurement.

Example Usage of Single-Shot Measurement Mode: Trimming the High-Frequency Low-Power Oscillator

A practical example of the usage of the spot measurement mode is in trimming the HF LPO (clock source # 5) using the main oscillator as a reference. This measurement sequence would proceed as follows:

- The application sets up the seed values for counter0 and valid0 for the duration of the measurement. Suppose the main oscillator frequency is 10MHz and the intended duration of the measurement is 500µs. The application needs to configure a seed value of 5000.
- These 5000 counts need to be divided between the counter0 and the valid0 counters. The minimum value for the valid0 seed is 4, so the application can configure counter0 seed value as 4996 and the valid0 seed value as 4.
- Suppose the HF LPO frequency is truly unknown. In this case the application can choose the maximum allowed seed value for counter1. This increases the probability of counter0 and valid0 counting down while the counter1 has still not fully counted down to zero. The maximum allowed seed value for counter1 is 1048575.
- Once the DCC is enabled, the counters counter0 and counter1 both start counting down from their seed values.
- When counter0 reaches zero, it automatically triggers the valid0 counter.
- When valid0 reaches zero, if counter1 is not zero as well, an ERROR status flag is set and a "DCC error" is sent to the ESM. Counter1 is also frozen so that it stops counting down any further. The application can enable an interrupt to be generated from the ESM whenever this DCC error is indicated. Refer the device datasheet to identify the ESM group and channel where the DCC error is connected.

- The DCC error interrupt service routine can then check the value of counter1 when the error was generated. Suppose that the counter1 now reads 1044575. This means that counter1 has counted 1048575 - 1044575, or 4000 cycles within the 500µs measurement period. This means that the average frequency of the HF LPO over this 500µs period was 4000 cycles / 500µs, or 8MHz.
- The application then needs to clear the ERROR status flag and restart the DCC module so that it is ready for the next spot measurement.

If there is no error generated at the end of the sequence, then the DONE status flag is set and a DONE interrupt is generated. The application must clear the DONE flag before restarting the DCC.

The conditions that cause a DCC error are identical between the continuous monitoring mode and the single-shot measurement mode.

Error Conditions:

An error condition is generated by any one of the following:

1. Counter1 counts down to 0 before Counter0 reaches 0. This means that clock1 is faster than expected, or clock0 is slower than expected. It includes the case when clock0 is stuck at 1 or 0.
2. Counter1 does not reach 0 even when Counter0 and Valid0 have both reached 0. This means that clock1 is slower than expected. It includes the case when clock1 is stuck at 1 or 0.

   Any error freezes the counters from counting. An application may then read out the counter values to help determine what caused the error.

Freezing Counters when Counter1 Reaches Zero:

The DCC module also allows the counters to be frozen when the counter1 reaches zero. This allows one of the clock sources for counter1 to be used as a reference for measuring one of the clock sources for counter0. The error conditions are the same as those where (counter0=0 and valid0=0) define the condition when the DCC counters are frozen. That is, an error is indicated if coutner0 and valid0 become zero while counter1 is still non-zero. In this case, however, the application would typically set up the seed values such that the counter1 will become zero before counter0. Essentially the measurement period is defined by the seed value of the counter1. Note that this is also an error condition, and the interrupt service routine can use the measurement period and the actual cycles counted by counter1 to determine the frequency of the clock0 signal.

15.3 Clock Source Selection for Counter0 and Counter1

Refer to the device datasheet to identify the available options for selecting the clock sources for both counters of the DCC module. Some microcontrollers may include multiple instances of the DCC module. This will also be identified in the device datasheet.

The selection of the clock sources for counter0 and coutner1 is done by a combination of the KEY, CNT0 CLKSRC, and CNT1 CLKSRC control fields of the CNT0CLKSRC and CNT1CLKSRC registers.

15.4 DCC Control Registers

This section describes the dual-clock comparator (DCC) module control and status registers. The registers support 8-bit, 16-bit or 32-bit writes and are aligned on a word (32-bit) boundary. Table 15-1 shows address offsets from the module base address. The base address for the control registers is FFFF EC00h for DCC1 and FFFF F400h for DCC2.

Table 15-1. DCC Control Registers

Offset	Acronym	Register Description	Section
00h	DCCGCTRL	DCC Global Control Register	Section 15.4.1
04h	DCCREV	DCC Revision Id Register	Section 15.4.2
08h	DCCCNT0SEED	DCC Counter0 Seed Register	Section 15.4.3
0Ch	DCCVALID0SEED	DCC Valid0 Seed Register	Section 15.4.4
10h	DCCCNT1SEED	DCC Counter1 Seed Register	Section 15.4.5
14h	DCCSTAT	DCC Status Register	Section 15.4.6
18h	DCCCNT0	DCC Counter0 Value Register	Section 15.4.7
1Ch	DCCVALID0	DCC Valid0 Value Register	Section 15.4.8
20h	DCCCNT1	DCC Counter1 Value Register	Section 15.4.9
24h	DCCCNT1CLKSRC	DCC Counter1 Clock Source Selection Register	Section 15.4.10
28h	DCCCNT0CLKSRC	DCC Counter0 Clock Source Selection Register	Section 15.4.11

15.4.1 DCC Global Control Register (DCCGCTRL)

Figure 15-7 and Table 15-2 describe the DCC Global Control register.

Figure 15-7. DCC Global Control Register (DCCGCTRL) [offset = 00]

31	16
Reserved	
R-0	

15 12	11 8	7 4	3 0
DONE INT ENA	SINGLE SHOT	ERR ENA	DCC ENA
R/WP-5h	R/WP-5h	R/WP-5h	R/WP-5h

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privileged mode only; -*n* = value after reset

Table 15-2. DCC Global Control Register (DCCGCTRL) Field Descriptions

Bit	Field	Value	Description
31-16	Reserved	0	Reads return 0. Writes have no effect.
15-12	DONE INT ENA		Done Interrupt Enable.
			Any operation mode read, privileged mode write:
		5h	No interrupt is generated when the DONE flag is set in the DCC Status (DCCSTAT) register.
		Others	DONE interrupt is generated when the DONE flag is set in the DCC Status (DCCSTAT) register.
11-8	SINGLE SHOT		Single-Shot Mode Enable.
			Any operation mode read, privileged mode write:
		Ah	DCC stops counting when counter0 and valid0 both reach zero.
		Bh	DCC stops counting when counter1 reaches zero.
		Others	DCC counts continuously and only stops when an error occurs.
7-4	ERR ENA		Error Interrupt Enable.
			Any operation mode read, privileged mode write:
		5h	No interrupt is generated when the ERR flag is set in the DCC Status (DCCSTAT) register.
		Others	ERROR interrupt is generated when the ERR flag is set in the DCC Status (DCCSTAT) register.
3-0	DCC ENA		DCC Enable.
			Any operation mode read, privileged mode write:
		5h	All DCC counters are stopped and error-checking is disabled. When an error occurs, the counters stop and this field is set to 5h automatically disabling the DCC counter in hardware.
		Others	Read: Counters are enabled.
			Write: Load counters with their seed values and begin counting. It is recommended to write Ah to enable counters to protect against single-bit errors.

15.4.2 DCC Revision Id Register (DCCREV)

Figure 15-8 and Table 15-3 describe the DCC Revision Id register.

Figure 15-8. DCC Revision Id Register (DCCREV) [offset = 4h]

31 30	29 28	27 16
SCHEME	Reserved	FUNC
R-01	R-0	R-0

15 11	10 8	7 6	5 0
RTL	MAJOR	CUSTOM	MINOR
R-0	R-2h	R-0	R-4h

LEGEND: R = Read only; -*n* = value after reset

Table 15-3. DCC Revision Id Register (DCCREV) Field Descriptions

Bit	Field	Value	Description
31-30	SCHEME	01	Reads return 01, writes have no effect.
29-28	Reserved	0	Reads return 0. Writes have no effect.
27-16	FUNC	0	Functional release number. Reads return 0x000, writes have no effect.
15-11	RTL	0	Design release number. Reads return 0x00, writes have no effect.
10-8	MAJOR	2h	Major revision number. Reads return 0x2, writes have no effect.
7-6	CUSTOM	0	Custom version number. Reads return 0x0, writes have no effect.
5-0	MINOR	4h	Minor revision number. Reads return 0x4, writes have no effect.

15.4.3 DCC Counter0 Seed Register (DCCCNT0SEED)

Figure 15-9 and Table 15-4 describe the DCC Counter0 Seed register.

Figure 15-9. DCC Counter0 Seed Register (DCCCNT0SEED) [offset = 8h]

31 20	19 16
Reserved	COUNT0 SEED
R-0	R/WP-0

15 0
COUNT0 SEED
R/WP-0

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privileged mode only; -*n* = value after reset

Table 15-4. DCC Counter0 Seed Register (DCCCNT0SEED) Field Descriptions

Bit	Field	Value	Description
31-20	Reserved	0	Reads return 0. Writes have no effect.
19-0	COUNT0 SEED		Seed value for DCC counter0. Reads in any operating mode return the current value of counter0. Writing in privileged mode only sets the current seed value for counter0.

NOTE: Seed for Counter0 must be non-zero

The DCC must only be enabled after programming a non-zero value in the COUNT0 SEED register.

15.4.4 DCC Valid0 Seed Register (DCCVALID0SEED)

Figure 15-10 and Table 15-5 describe the DCC Valid0 Seed register.

Figure 15-10. DCC Valid0 Seed Register (DCCVALID0SEED) [offset = Ch]

31 - 16
Reserved
R-0

15 - 0
VALID0 SEED
R/WP-0

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privileged mode only; -*n* = value after reset

Table 15-5. DCC Valid0 Seed Register (DCCVALID0SEED) Field Descriptions

Bit	Field	Value	Description
31-16	Reserved	0	Reads return 0. Writes have no effect.
15-0	VALID0 SEED		Seed value for DCC Valid0. This value defines the window within which the counter1 must reach 0. This window needs to be at least 4 cycles wide. Reads in any operating mode return the current value of seed for Valid0. Writing in privileged mode only sets the current seed value for Valid0. Writes in user mode are ignored.

NOTE: Seed for Valid0 must be at least 0x4

The DCC must only be enabled after programming a value greater than or equal to 0x4 in the VALID0 SEED register.

15.4.5 DCC Counter1 Seed Register (DCCCNT1SEED)

Figure 15-11 and Table 15-6 describe the DCC Counter1 Seed register.

Figure 15-11. DCC Counter1 Seed Register (DCCCNT1SEED) [offset = 10h]

31 - 20	19 - 16
Reserved	COUNT1 SEED
R-0	R/WP-0

15 - 0
COUNT1 SEED
R/WP-0

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privileged mode only; -*n* = value after reset

Table 15-6. DCC Counter1 Seed Register (DCCCNT0SEED) Field Descriptions

Bit	Field	Value	Description
31-20	Reserved	0	Reads return 0. Writes have no effect.
19-0	COUNT1 SEED		Seed value for DCC counter1. Reads in any operating mode return the current value of seed for counter1. Writing in privileged mode only sets the current seed value for counter1. Writes in user mode are ignored.

NOTE: Seed for Counter0 must be non-zero

The DCC must only be enabled after programming a non-zero value in the COUNT1 SEED register.

15.4.6 DCC Status Register (DCCSTAT)

Figure 15-7 and Table 15-2 describe the DCC Status register.

Figure 15-12. DCC Status Register (DCCSTAT) [offset = 14h]

31–16
Reserved
R-0

15–2	1	0
Reserved	DONE	ERR
R-0	R/W1CP-0	R/W1CP-0

LEGEND: R/W = Read/Write; R = Read only; W1CP = Write 1 to clear in privilege mode only; -*n* = value after reset

Table 15-7. DCC Status Register (DCCSTAT) Field Descriptions

Bit	Field	Value	Description
31-2	Reserved	0	Reads return 0. Writes have no effect.
1	DONE		Single-Shot Sequence Done flag. Indicates that a single-shot DCC sequence is done without any error.
		0	Read: Single-shot sequence is not done. Write: Writing 0 has no effect.
		1	Read: Single-shot sequence is done without any error. Write: Writing 1 in privileged mode clears the DONE flag.
0	ERR		Error flag. Indicates that a DCC error has occurred.
		0	Read: DCC error has not occurred. Write: Writing 0 has no effect.
		1	Read: An error has occurred. Write: Writing 1 in privileged mode clears the ERR flag.

15.4.7 DCC Counter0 Value Register (DCCCNT0)

Figure 15-13 and Table 15-8 describe the DCC Counter0 Value register.

Figure 15-13. DCC Counter0 Value Register (DCCCNT0) [offset = 18h]

31 20	19 16
Reserved	COUNT0
R-0	R-0

15 0
COUNT0
R-0

LEGEND: R = Read only; -*n* = value after reset

Table 15-8. DCC Counter0 Value Register (DCCCNT0) Field Descriptions

Bit	Field	Value	Description
31-20	Reserved	0	Reads return 0. Writes have no effect.
19-0	COUNT0		Current value of DCC counter0. Reads in any operating mode return the current value of counter0. Writes have no effect.

NOTE: Reads may not return exact current value of counter

Reading the counter0 value while counting is enabled may not return the exact value of the counter0.

15.4.8 DCC Valid0 Value Register (DCCVALID0)

Figure 15-14 and Table 15-9 describe the DCC Valid0 Value register.

Figure 15-14. DCC Valid0 Value Register (DCCVALID0) [offset = 1Ch]

31	16
Reserved	
R-0	

15	0
VALID0	
R-0	

LEGEND: R = Read only; -*n* = value after reset

Table 15-9. DCC Valid0 Value Register (DCCVALID0) Field Descriptions

Bit	Field	Value	Description
31-16	Reserved	0	Reads return 0. Writes have no effect.
15-0	VALID0		Current value for DCC Valid0. Reads in any operating mode return the current value of Valid0. Writes have no effect.

NOTE: Reads may not return exact current value of Valid0

Reading the Valid0 value while counting is enabled may not return the exact value of the Valid0.

15.4.9 DCC Counter1 Value Register (DCCCNT1)

Figure 15-15 and Table 15-10 describe the DCC Counter1 Value register.

Figure 15-15. DCC Counter1 Value Register (DCCCNT1) [offset = 20h]

31-20	19-16
Reserved	COUNT1
R-0	R/WP-0

15	0
COUNT1	
R/WP-0	

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privileged mode only; -*n* = value after reset

Table 15-10. DCC Counter1 Value Register (DCCCNT1) Field Descriptions

Bit	Field	Value	Description
31-20	Reserved	0	Reads return 0. Writes have no effect.
19-0	COUNT1		Current value for DCC counter1. Reads in any operating mode return the current value of counter1. Writes have no effect.

NOTE: Reads may not return exact current value of counter

Reading the counter1 value while counting is enabled may not return the exact value of the counter1.

15.4.10 DCC Counter1 Clock Source Selection Register (DCCCNT1CLKSRC)

Figure 15-15 and Table 15-10 describe the DCC Counter1 Clock Source Selection register.

Figure 15-16. DCC Counter1 Clock Source Selection Register (DCCCNT1CLKSRC) [offset = 24h]

31 16
Reserved
R-0

15 12	11 4	3 0
KEY	Reserved	CNT1 CLKSRC
R/WP-5h	R-0	R/WP-0

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privileged mode only; -*n* = value after reset

Table 15-11. DCC Counter1 Clock Source Selection Register (DCCCNT1CLKSRC) Field Descriptions

Bit	Field	Value	Description
31-16	Reserved	0	Reads return 0. Writes have no effect.
15-12	KEY		Key to enable clock source selection for counter1. Reads in any operating mode return the current value of the key.
			Writes in privileged mode set the key value.
		Ah	Writing Ah as the key enables the CNT1 CLKSRC field to define the clock source for counter1.
		Any other value	Writing any other value as the key disables the clock source selection for counter1. In this case, the N2HET signal is used as the source for counter1.
			Refer to the device datasheet for available clock source options and the KEY required to enable these options for counter1.
11-4	Reserved	0	Reads return 0. Writes have no effect.
3-0	CNT1 CLKSRC		Clock source for counter1 when KEY is programmed to Ah. Reads in any operating mode return the current value of CLKSRC. Writes in privileged mode select the clock source for counter1. Refer to the device datasheet for available clock source options and the KEY required to enable these options for counter1.

15.4.11 DCC Counter0 Clock Source Selection Register (DCCCNT0CLKSRC)

Figure 15-15 and Table 15-10 describe the DCC Counter0 Clock Source Selection register.

Figure 15-17. DCC Counter0 Clock Source Selection Register (DCCCNT0CLKSRC) [offset = 28h]

31	16
Reserved	
R-0	

15	4	3	0
Reserved		CNT0 CLKSRC	
R-0		R/WP-5h	

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privileged mode only; -*n* = value after reset

Table 15-12. DCC Counter0 Clock Source Selection Register (DCCCNT0CLKSRC) Field Descriptions

Bit	Field	Value	Description
31-4	Reserved	0	Reads return 0. Writes have no effect.
3-0	CNT0 CLKSRC		Clock source for counter0 . Reads in any operating mode return the current value of CLKSRC. Writes in privileged mode select the clock source for counter0. Refer to the device datasheet for available clock source options for counter0.

Chapter 16

SPNU563A–March 2018

Error Signaling Module (ESM)

This chapter provides the details of the error signaling module (ESM) that aggregates device errors and provides internal and external error response based on error severity.

Topic **Page**

16.1 Overview

The Error Signaling Module (ESM) collects and reports the various error conditions on the microcontroller. The error condition is categorized based on a severity level. Error response is then generated based on the category of the error. Possible error responses include a low priority interrupt, high priority interrupt, and an external pin action.

16.1.1 Feature List

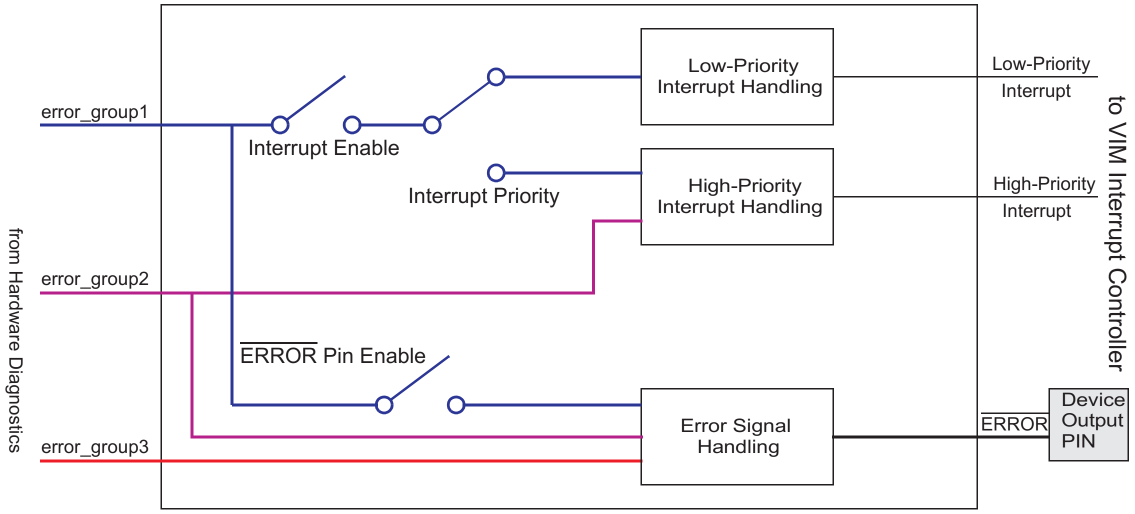
- Up to 160 error channels are supported, divided into 3 different groups:
  - 96 Group1 (low severity) channels with configurable interrupt generation and configurable ERROR pin behavior
  - 32 Group2 (high severity) channels with predefined interrupt generation and predefined ERROR pin behavior
  - 32 Group3 (high severity) channels with no interrupt generation and predefined ERROR pin behavior. These channels have no interrupt response as they are reserved for CPU based diagnostics that generate aborts directly to the CPU.
- Dedicated device ERROR pin to signal an external observer
- Configurable timebase for ERROR pin output
- Error forcing capability for latent fault testing

16.1.2 Block Diagram

As shown in Figure 16-1, the ESM channels are divided into three groups. Group1 channels are considered to be low severity. Group1 errors have a configurable interrupt response and configurable ERROR pin behavior. Note that the ESM Status Register 1 (ESMSR1) for error group 1 gets updated, regardless if the interrupt enable is active or not. Group2 channels are ERROR high severity. Group2 errors always generate a high priority interrupt and an output on the ERROR pin. Group3 channels indicate errors of the highest severity. Check the specific part's datasheet for identifying group3 errors and their expected responses. Group3 errors always generate an ERROR pin output.

The ESM interrupt and ERROR pin behavior are also summarized in Table 16-1.

Figure 16-1. Block Diagram



Note that the ESM Status Register 1 (ESMSR1) for error_group1 gets updated, regardless if the interrupt enable is active or not.

Table 16-1. ESM Interrupt and ERROR Pin Behavior

Error Group	Interrupt Generated	Interrupt Priority	ERROR Pin Response Generated
1	configurable interrupt	configurable priority	configurable output generation
2	interrupt generated	high priority	output generated
3	no interrupt	NA	output generated

Figure 16-2 and Figure 16-3 show the interrupt response handling and ERROR pin response handling with register configuration. The total active time of the ERROR pin is controlled by the Low-Time Counter Preload register (LTCP) and the key register (ESMEPSR) as shown in Figure 16-3. See Section 16.2.2 for details.

Figure 16-2. Interrupt Response Handling

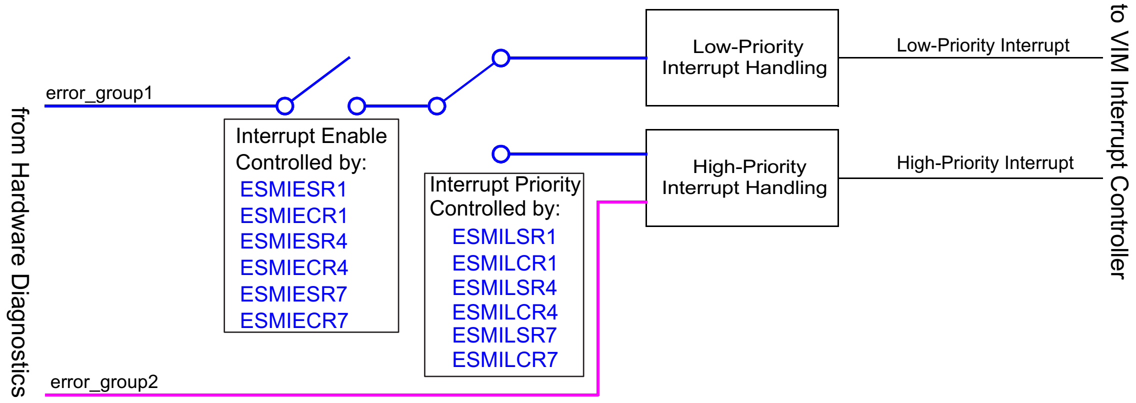
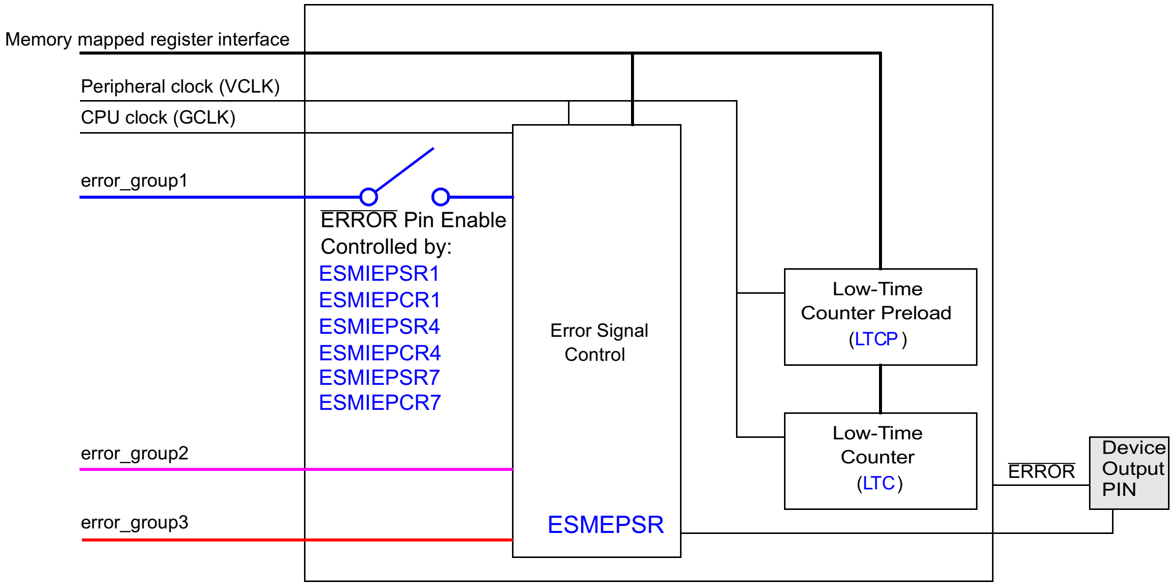


Figure 16-3. ERROR Pin Response Handling



16.2 Module Operation

This device has 160 error channels, divided into 3 different error groups. Please refer to the device datasheet for ESM channel assignment details.

The ESM module has error flags for each error channel. The error status registers ESMSR1, ESMSR4, ESMSR7, ESMSR2, ESMSR3 provide status information on a pending error of Group1 (Channel 0-31), Group1 (Channel 32-63), Group1 (Channel 64-95), Group2, and Group3, respectively. The ESMEPSR register provides the current $\overline{\text{ERROR}}$ status. The module also provides a status shadow register, ESMSSR2, which maintains the error flags of Group2 until power-on reset ($\overline{\text{PORRST}}$) is asserted. See Section 16.2.1 for details of their behavior during power on reset and warm reset.

Once an error occurs, the ESM module will set the corresponding error flags. In addition, it can trigger an interrupt, $\overline{\text{ERROR}}$ pin outputs low depending on the ESM settings. Once the $\overline{\text{ERROR}}$ pin outputs low, a power on reset or a write of 0x5 to ESMEKR is required to release the ESM error pin back to normal state. See Section 16.2.2 for details. The application can read the error status registers (ESMSR1, ESMSR4, ESMSR7, ESMSR2, and ESMSR3) to debug the error. If an $\overline{\text{RST}}$ is triggered or the error interrupt has been served, the error flag of Group2 should be read from ESMSSR2 because the error flag in ESMSR2 will be cleared by $\overline{\text{RST}}$.

You can also test the functionality of the $\overline{\text{ERROR}}$ pin by forcing an error. See Section 16.2.3 for details.

16.2.1 Reset Behavior

Power on reset:

- $\overline{\text{ERROR}}$ pin behavior

  When nPORRST is active, the $\overline{\text{ERROR}}$ pin is in a high impedance state (output drivers disabled).
- Register behavior

  After $\overline{\text{PORRST}}$, all registers in ESM module will be re-initialized to the default value. All the error status registers are cleared to zero.

Warm reset ($\overline{\text{RST}}$):

- $\overline{\text{ERROR}}$ pin behavior

  During $\overline{\text{RST}}$, the $\overline{\text{ERROR}}$ pin is in “output active” state with pull-down disabled. The $\overline{\text{ERROR}}$ pin remains unchanged after $\overline{\text{RST}}$.
- Register behavior

  After $\overline{\text{RST}}$, ESMSR1, ESMSR4, ESMSR7, ESMSSR2, ESMSR3 and ESMEPSR register values remains un-changed. Since $\overline{\text{RST}}$ does not clear the critical failure registers, the user can read those registers to debug the failures after $\overline{\text{RST}}$ pin goes back to high.

  After $\overline{\text{RST}}$, if one of the flags in ESMSR1, ESMSR4 and ESMSR7 is set, the interrupt service routine will be called once the corresponding interrupt is enabled.

NOTE: ESMSR2 is cleared after $\overline{\text{RST}}$. The flag in ESMSR2 gets cleared when reading the appropriate vector in the ESMIOFFHR offset register. Reading ESMIOFFHR will not clear the ESMSR1, ESMSR4, ESMSR7 and the shadow register ESMSSR2. Reading ESMIOFFLR will also not clear the ESMSR1, ESMSR4 and ESMSR7.

16.2.2 ERROR Pin Timing

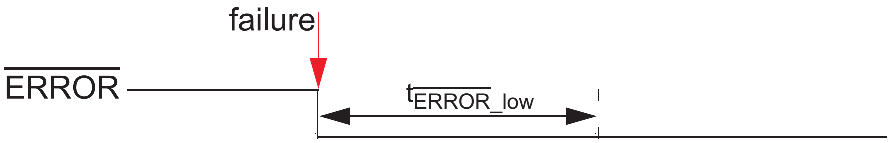
The ERROR pin is an active low function. The state of the pin is also readable from ERROR Pin Status Register (ESMEPSR). A warm reset (RST) does not affect the state of the pin. The pin is in a high-impedance state during power-on reset. Once the ESM module drives the ERROR pin low, it remains in this state for the time specified by the Low-Time Counter Preload register (LTCPR). Based on the time period of the peripheral clock (VCLK), the total active time of the ERROR pin can be calculated as:

$$t_{\overline{ERROR}\_low} = t_{VCLK} \times (LTCP + 1) \qquad (22)$$

Once this period expires, the ERROR pin is set to high in case the reset of the ERROR pin was requested. This request is done by writing an appropriate key (0x5) to the key register (ESMEKR) during the ERROR pin low time. Here are a few examples:

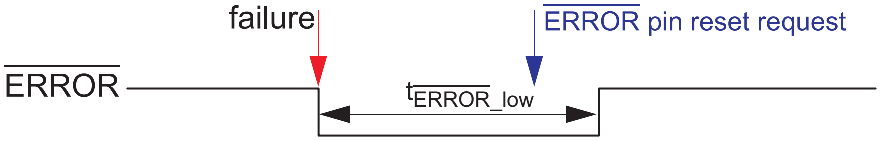
Example 1: ESM detects a failure and drives the ERROR pin low. No ERROR pin reset is requested. The ERROR pin continues outputting low until power on reset occurs.

Figure 16-4. ERROR Pin Timing - Example 1



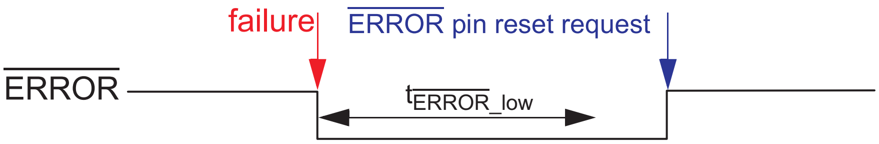
Example 2: ESM detects a failure and drives the ERROR pin low. An ERROR pin reset request is received before $t_{\overline{ERROR}\_low}$ expires. In this case, the ERROR pin is set to high immediately after $t_{\overline{ERROR}\_low}$ expires.

Figure 16-5. ERROR Pin Timing - Example 2



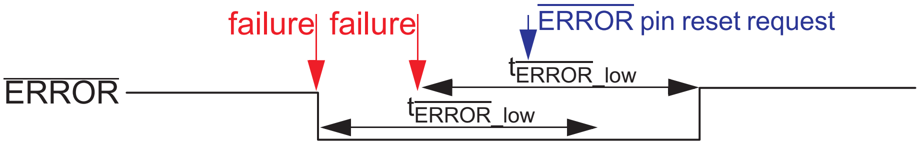
Example 3: ESM detects a failure and drives the ERROR pin low. An ERROR pin reset request is received after $t_{\overline{ERROR}\_low}$ expires. In this case, the ERROR pin is set to high immediately after ERROR pin reset request is received.

Figure 16-6. ERROR Pin Timing - Example 3



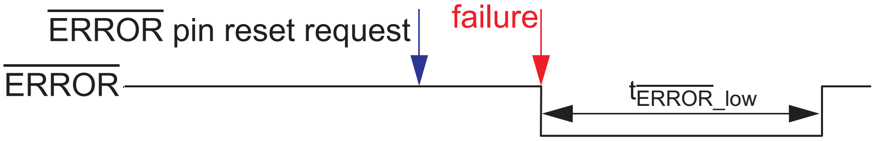
Example 4: ESM detects a failure and drives the $\overline{\text{ERROR}}$ pin low. Another failure occurs within the time the pin stays low. In this case, the low time counter will be reset when the other failure occurs. In other words, $t_{\overline{\text{ERROR}}\_low}$ should be counted from whenever the most recent failure occurs.

Figure 16-7. $\overline{\text{ERROR}}$ Pin Timing - Example 4



Example 5: The reset of the $\overline{\text{ERROR}}$ pin was requested by the software even before the failure occurs. In this case, the $\overline{\text{ERROR}}$ pin is set to high immediately after $t_{\overline{\text{ERROR}}\_low}$ expires. This case is not recommended and should be avoided by the application.

Figure 16-8. $\overline{\text{ERROR}}$ Pin Timing - Example 5



16.2.3 Forcing an Error Condition

The error response generation mechanism is testable by software by forcing an error condition. This allows testing the $\overline{\text{ERROR}}$ pin functionality. By writing a dedicated key to the error forcing key register (ESMEKR), the $\overline{\text{ERROR}}$ pin is set to low for the specified time. The following steps describe how to force an error condition:

1. Check $\overline{\text{ERROR}}$ Pin Status Register (ESMEPSR). This register must be 1 to switch into the error forcing mode.

   The ESM module cannot be switched into the error forcing mode if a failure has already been detected in functional mode. The application command to switch to error forcing mode is ignored.

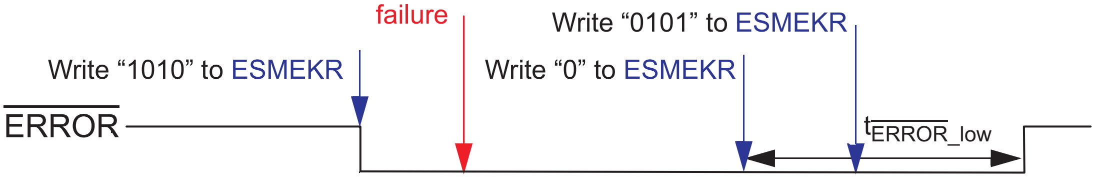
2. Write “1010b” to the error forcing key register (ESMEKR). After that, the $\overline{\text{ERROR}}$ pin should output low (error force mode).

   Once the application puts the ESM module in the error forcing mode, the $\overline{\text{ERROR}}$ pin cannot indicate the normal error functionality. If a failure occurs during this time, it gets still latched and the LTC is reset and stopped. The error output pin is already driven low on account of the error forcing mode. When the ESM is forced back to normal functional mode, the LTC becomes active and forces the $\overline{\text{ERROR}}$ pin low until the expiration of the LTC (see Figure 16-9).

3. Write “0000” to the error forcing key register (ESMEKR) back to the active normal mode.

   If there are no errors detected while the ESM module is in the error forcing mode, the $\overline{\text{ERROR}}$ pin goes high immediately after exiting the error forcing mode.

Figure 16-9. ERROR Pin Timing - Example 6

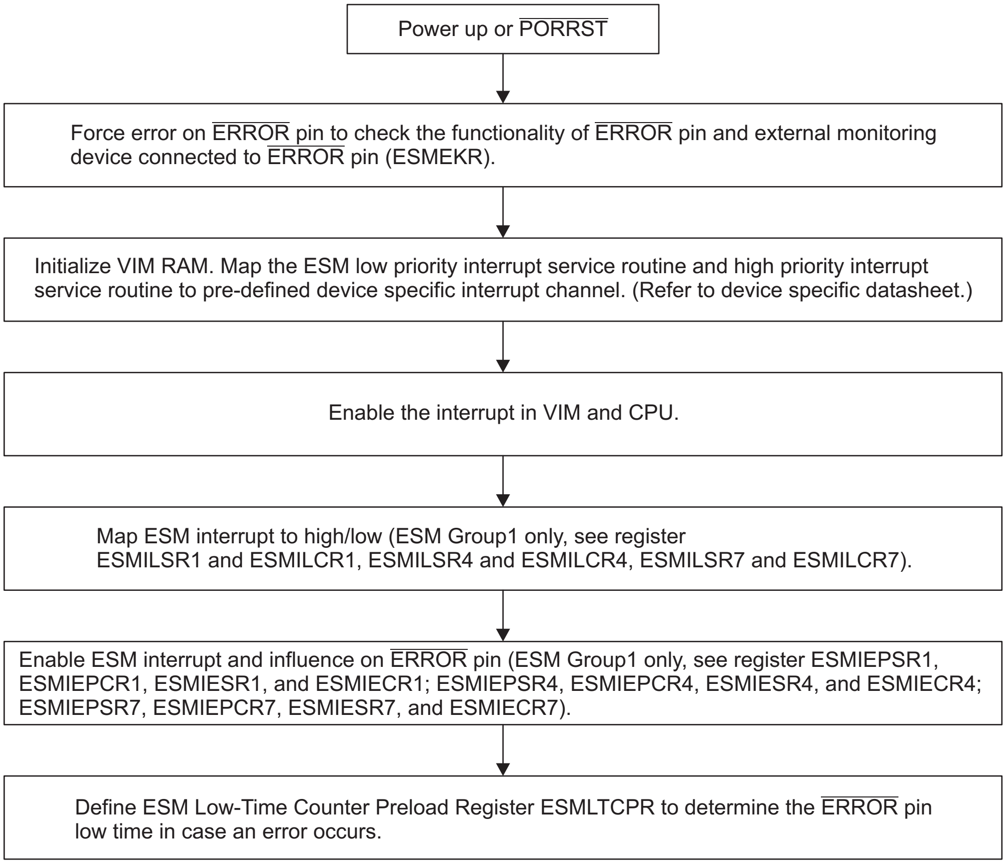


16.3 Recommended Programming Procedure

During the initialization stage, the application code should follow the recommendations in Figure 16-10 to initialize the ESM.

Once an error occurs, it can trigger an interrupt, $\overline{\text{ERROR}}$ pin outputs low depending on the ESM settings. Once the $\overline{\text{ERROR}}$ pin outputs low, a power on reset or a write of 0x5 to ESMEKR is required to release the ESM back to normal state. The application can read the error status registers (ESMSR1, ESMSR4, ESMSR7, ESMSR2, and ESMSR3) to debug the error. If an $\overline{\text{RST}}$ is triggered or the error interrupt has been served, the error flag of Group2 should be read from ESMSSR2 because the error flag in ESMSR2 will be cleared by $\overline{\text{RST}}$.

Figure 16-10. ESM Initialization


Power up or $\overline{\text{PORRST}}$
Force error on $\overline{\text{ERROR}}$ pin to check the functionality of $\overline{\text{ERROR}}$ pin and external monitoring device connected to $\overline{\text{ERROR}}$ pin (ESMEKR).
Initialize VIM RAM. Map the ESM low priority interrupt service routine and high priority interrupt service routine to pre-defined device specific interrupt channel. (Refer to device specific datasheet.)
Enable the interrupt in VIM and CPU.
Map ESM interrupt to high/low (ESM Group1 only, see register ESMILSR1 and ESMILCR1, ESMILSR4 and ESMILCR4, ESMILSR7 and ESMILCR7).
Enable ESM interrupt and influence on $\overline{\text{ERROR}}$ pin (ESM Group1 only, see register ESMIEPSR1, ESMIEPCR1, ESMIESR1, and ESMIECR1; ESMIEPSR4, ESMIEPCR4, ESMIESR4, and ESMIECR4; ESMIEPSR7, ESMIEPCR7, ESMIESR7, and ESMIECR7).
Define ESM Low-Time Counter Preload Register ESMLTCPR to determine the $\overline{\text{ERROR}}$ pin low time in case an error occurs.

16.4 ESM Control Registers

Table 16-2 lists the ESM registers. Each register begins on a 32-bit word boundary. The registers support 8-, 16-, and 32-bit accesses. The base address for the control registers is FFFF F500h.

Table 16-2. ESM Control Registers

Offset	Acronym	Register Description	Section
00h	ESMEEPAPR1	ESM Enable $\overline{\text{ERROR}}$ Pin Action/Response Register 1	Section 16.4.1
04h	ESMDEPAPR1	ESM Disable $\overline{\text{ERROR}}$ Pin Action/Response Register 1	Section 16.4.2
08h	ESMIESR1	ESM Interrupt Enable Set/Status Register 1	Section 16.4.3
0Ch	ESMIECR1	ESM Interrupt Enable Clear/Status Register 1	Section 16.4.4
10h	ESMILSR1	Interrupt Level Set/Status Register 1	Section 16.4.5
14h	ESMILCR1	Interrupt Level Clear/Status Register 1	Section 16.4.6
18h	ESMSR1	ESM Status Register 1	Section 16.4.7
1Ch	ESMSR2	ESM Status Register 2	Section 16.4.8
20h	ESMSR3	ESM Status Register 3	Section 16.4.9
24h	ESMEPSR	ESM $\overline{\text{ERROR}}$ Pin Status Register	Section 16.4.10
28h	ESMIOFFHR	ESM Interrupt Offset High Register	Section 16.4.11
2Ch	ESMIOFFLR	ESM Interrupt Offset Low Register	Section 16.4.12
30h	ESMLTCR	ESM Low-Time Counter Register	Section 16.4.13
34h	ESMLTCPR	ESM Low-Time Counter Preload Register	Section 16.4.14
38h	ESMEKR	ESM Error Key Register	Section 16.4.15
3Ch	ESMSSR2	ESM Status Shadow Register 2	Section 16.4.16
40h	ESMIEPSR4	ESM Influence $\overline{\text{ERROR}}$ Pin Set/Status Register 4	Section 16.4.17
44h	ESMIEPCR4	ESM Influence $\overline{\text{ERROR}}$ Pin Clear/Status Register 4	Section 16.4.18
48h	ESMIESR4	ESM Interrupt Enable Set/Status Register 4	Section 16.4.19
4Ch	ESMIECR4	ESM Interrupt Enable Clear/Status Register 4	Section 16.4.20
50h	ESMILSR4	Interrupt Level Set/Status Register 4	Section 16.4.21
54h	ESMILCR4	Interrupt Level Clear/Status Register 4	Section 16.4.22
58h	ESMSR4	ESM Status Register 4	Section 16.4.23
80h	ESMIEPSR7	ESM Influence $\overline{\text{ERROR}}$ Pin Set/Status Register 7	Section 16.4.24
84h	ESMIEPCR7	ESM Influence $\overline{\text{ERROR}}$ Pin Clear/Status Register 7	Section 16.4.25
88h	ESMIESR7	ESM Interrupt Enable Set/Status Register 7	Section 16.4.26
8Ch	ESMIECR7	ESM Interrupt Enable Clear/Status Register 7	Section 16.4.27
90h	ESMILSR7	Interrupt Level Set/Status Register 7	Section 16.4.28
94h	ESMILCR7	Interrupt Level Clear/Status Register 7	Section 16.4.29
98h	ESMSR7	ESM Status Register 7	Section 16.4.30

16.4.1 ESM Enable $\overline{\text{ERROR}}$ Pin Action/Response Register 1 (ESMEEPAPR1)

This register is dedicated for Group1 Channel[31:0].

Figure 16-11. ESM Enable $\overline{\text{ERROR}}$ Pin Action/Response Register 1 (ESMEEPAPR1) [offset = 00h]

31	16
IEPSET[31:16]	
R/WP-0	

15	0
IEPSET[15:0]	
R/WP-0	

LEGEND: R/W = Read/Write; WP = Write in privileged mode only; -*n* = value after reset

Table 16-3. ESM Enable $\overline{\text{ERROR}}$ Pin Action/Response Register 1 (ESMEEPAPR1) Field Descriptions

Bit	Field	Value	Description
31-0	IEPSET		Enable $\overline{\text{ERROR}}$ Pin Action/Response on Group 1. **Read in User and Privileged mode. Write in Privileged mode only.**
		0	Read: Failure on channel x has no influence on $\overline{\text{ERROR}}$ pin. Write: Leaves the bit and the corresponding clear bit in the ESMIEPCR1 register unchanged.
		1	Read: Failure on channel x has influence on $\overline{\text{ERROR}}$ pin. Write: Enables failure influence on $\overline{\text{ERROR}}$ pin and sets the corresponding clear bit in the ESMIEPCR1 register.

16.4.2 ESM Disable $\overline{\text{ERROR}}$ Pin Action/Response Register 1 (ESMDEPAPR1)

This register is dedicated for Group1 Channel[31:0].

Figure 16-12. ESM Disable $\overline{\text{ERROR}}$ Pin Action/Response Register 1 (ESMDEPAPR1) [offset = 04h]

31	16
IEPCLR[31:16]	
R/WP-0	

15	0
IEPCLR[15:0]	
R/WP-0	

LEGEND: R/W = Read/Write; WP = Write in privileged mode only; -*n* = value after reset

Table 16-4. ESM Disable $\overline{\text{ERROR}}$ Pin Action/Response Register 1 (ESMDEPAPR1) Field Descriptions

Bit	Field	Value	Description
31-0	IEPCLR		Disable $\overline{\text{ERROR}}$ Pin Action/Response on Group 1. **Read in User and Privileged mode. Write in Privileged mode only.**
		0	Read: Failure on channel x has no influence on $\overline{\text{ERROR}}$ pin. Write: Leaves the bit and the corresponding set bit in the ESMIEPSR1 register unchanged.
		1	Read: Failure on channel x has influence on $\overline{\text{ERROR}}$ pin. Write: Disables failure influence on $\overline{\text{ERROR}}$ pin and clears the corresponding set bit in the ESMIEPSR1 register.

16.4.3 ESM Interrupt Enable Set/Status Register 1 (ESMIESR1)

This register is dedicated for Group1 Channel[31:0].

Figure 16-13. ESM Interrupt Enable Set/Status Register 1 (ESMIESR1) [offset = 08h]

31	16
INTENSET[31:16]	
R/WP-0	

15	0
INTENSET[15:0]	
R/WP-0	

LEGEND: R/W = Read/Write; WP = Write in privileged mode only; -*n* = value after reset

Table 16-5. ESM Interrupt Enable Set/Status Register 1 (ESMIESR1) Field Descriptions

Bit	Field	Value	Description
31-0	INTENSET		Set interrupt enable. **Read in User and Privileged mode. Write in Privileged mode only.**
		0	Read: Interrupt is disabled. Write: Leaves the bit and the corresponding clear bit in the ESMIECR1 register unchanged.
		1	Read: Interrupt is enabled. Write: Enables interrupt and sets the corresponding clear bit in the ESMIECR1 register.

16.4.4 ESM Interrupt Enable Clear/Status Register 1 (ESMIECR1)

This register is dedicated for Group1 Channel[31:0].

Figure 16-14. ESM Interrupt Enable Clear/Status Register 1 (ESMIECR1) [offset = 0Ch]

31	16
INTENCLR[31:16]	
R/WP-0	

15	0
INTENCLR[15:0]	
R/WP-0	

LEGEND: R/W = Read/Write; WP = Write in privileged mode only; -*n* = value after reset

Table 16-6. ESM Interrupt Enable Clear/Status Register 1 (ESMIECR1) Field Descriptions

Bit	Field	Value	Description
31-0	INTENCLR		Clear interrupt enable. **Read in User and Privileged mode. Write in Privileged mode only.**
		0	Read: Interrupt is disabled. Write: Leaves the bit and the corresponding set bit in the ESMIESR1 register unchanged.
		1	Read: Interrupt is enabled. Write: Disables interrupt and clears the corresponding set bit in the ESMIESR1 register.

16.4.5 ESM Interrupt Level Set/Status Register 1 (ESMILSR1)

This register is dedicated for Group1 Channel[31:0].

Figure 16-15. ESM Interrupt Level Set/Status Register 1 (ESMILSR1) [offset = 10h]

31	16
INTLVLSET[31:16]	
R/WP-0	
15	0
INTLVLSET[15:0]	
R/WP-0	

LEGEND: R/W = Read/Write; WP = Write in privileged mode only; -*n* = value after reset

Table 16-7. ESM Interrupt Level Set/Status Register 1 (ESMILSR1) Field Descriptions

Bit	Field	Value	Description
31-0	INTLVLSET		Set interrupt priority. **Read in User and Privileged mode. Write in Privileged mode only.**
		0	Read: Interrupt of channel x is mapped to low level interrupt line. Write: Leaves the bit and the corresponding clear bit in the ESMILCR1 register unchanged.
		1	Read: Interrupt of channel x is mapped to high level interrupt line. Write: Maps interrupt of channel x to high level interrupt line and sets the corresponding clear bit in the ESMILCR1 register.

16.4.6 ESM Interrupt Level Clear/Status Register 1 (ESMILCR1)

This register is dedicated for Group1 Channel[31:0].

Figure 16-16. ESM Interrupt Level Clear/Status Register 1 (ESMILCR1) [offset = 14h]

31	16
INTLVLCLR[31:16]	
R/WP-0	
15	0
INTLVLCLR[15:0]	
R/WP-0	

LEGEND: R/W = Read/Write; WP = Write in privileged mode only; -*n* = value after reset

Table 16-8. ESM Interrupt Level Clear/Status Register 1 (ESMILCR1) Field Descriptions

Bit	Field	Value	Description
31-0	INTLVLCLR		Clear interrupt priority. **Read in User and Privileged mode. Write in Privileged mode only.**
		0	Read: Interrupt of channel x is mapped to low-level interrupt line. Write: Leaves the bit and the corresponding set bit in the ESMILSR1 register unchanged.
		1	Read: Interrupt of channel x is mapped to high-level interrupt line. Write: Maps interrupt of channel x to low-level interrupt line and clears the corresponding set bit in the ESMILSR1 register.

16.4.7 ESM Status Register 1 (ESMSR1)

This register is dedicated for Group1 Channel[31:0]. Note that the ESMSR1 status register will get updated if an error condition occurs, regardless if the corresponding interrupt enable flag is set or not.

Figure 16-17. ESM Status Register 1 (ESMSR1) [offset = 18h]

31	16
ESF[31:16]	
R/W1CP-X/0	

15	0
ESF[15:0]	
R/W1CP-X/0	

LEGEND: R/W = Read/Write; W1CP = Write 1 to clear in privilege mode only; -*n* = value after reset/$\overline{\text{PORRST}}$; *X* = value is unchanged

Table 16-9. ESM Status Register 1 (ESMSR1) Field Descriptions

Bit	Field	Value	Description
31-0	ESF		Error Status Flag. Provides status information on a pending error. **Read in User and Privileged mode. Write in Privileged mode only.**
		0	Read: No error occurred; no interrupt is pending. Write: Leaves the bit unchanged.
		1	Read: Error occurred; interrupt is pending. Write: Clears the bit. **Note:** After $\overline{\text{RST}}$, if one of these flags are set and the corresponding interrupt are enabled, the interrupt service routine will be called.

16.4.8 ESM Status Register 2 (ESMSR2)

This register is dedicated for Group2.

Figure 16-18. ESM Status Register 2 (ESMSR2) [offset = 1Ch]

31	16
ESF2[31:16]	
R/W1CP-0	

15	0
ESF2[15:0]	
R/W1CP-0	

LEGEND: R/W = Read/Write; W1CP = Write 1 to clear in privilege mode only; -*n* = value after reset

Table 16-10. ESM Status Register 2 (ESMSR2) Field Descriptions

Bit	Field	Value	Description
31-0	ESF2		Error Status Flag. Provides status information on a pending error. **Read in User and Privileged mode. Write in Privileged mode only.**
		0	Read: No error occurred; no interrupt is pending. Write: Leaves the bit unchanged.
		1	Read: Error occurred; interrupt is pending. Write: Clears the bit. ESMSSR2 is not impacted by this action. **Note:** In normal operation the flag gets cleared when reading the appropriate vector in the ESMIOFFHR offset register. Reading ESMIOFFHR will not clear the ESMSR1 and the shadow register ESMSSR2.

16.4.9 ESM Status Register 3 (ESMSR3)

This register is dedicated for Group3.

Figure 16-19. ESM Status Register 3 (ESMSR3) [offset = 20h]

31 16
ESF3[31:0]
R/W1CP-X/0

15 0
ESF3[15:0]
R/W1CP-X/0

LEGEND: R/W = Read/Write; W1CP = Write 1 to clear in privilege mode only; -*n* = value after reset/$\overline{\text{PORRST}}$; *X* = value is unchanged

Table 16-11. ESM Status Register 3 (ESMSR3) Field Descriptions

Bit	Field	Value	Description
31-0	ESF3		Error Status Flag. Provides status information on a pending error. **Read in User and Privileged mode. Write in Privileged mode only.**
		0	Read: No error occurred. Write: Leaves the bit unchanged.
		1	Read: Error occurred. Write: Clears the bit.

16.4.10 ESM $\overline{\text{ERROR}}$ Pin Status Register (ESMEPSR)

Figure 16-20. ESM $\overline{\text{ERROR}}$ Pin Status Register (ESMEPSR) [offset = 24h]

31 16
Reserved
R-0

15 1	0
Reserved	EPSF
R-0	R-X/1

LEGEND: R = Read only; -*n* = value after reset/$\overline{\text{PORRST}}$; *X* = value is unchanged

Table 16-12. ESM $\overline{\text{ERROR}}$ Pin Status Register (ESMEPSR) Field Descriptions

Bit	Field	Value	Description
31-1	Reserved	0	Reads return 0. Writes have no effect.
0	EPSF		$\overline{\text{ERROR}}$ Pin Status Flag. Provides status information for the $\overline{\text{ERROR}}$ Pin. **Read/Write in User and Privileged mode.**
		0	Read: $\overline{\text{ERROR}}$ pin is low (active) if any error has occurred. Write: Writes have no effect.
		1	Read: $\overline{\text{ERROR}}$ pin is high if no error has occurred. Write: Writes have no effect. **Note:** This flag will be set to 1 after $\overline{\text{PORRST}}$. The value will be unchanged after $\overline{\text{RST}}$. The $\overline{\text{ERROR}}$ pin status remains unchanged during after $\overline{\text{RST}}$.

16.4.11 ESM Interrupt Offset High Register (ESMIOFFHR)

Figure 16-21. ESM Interrupt Offset High Register (ESMIOFFHR) [offset = 28h]

31	16
Reserved	
R-0	

15	8	7	0
Reserved		INTOFFH	
R-0		R-0	

LEGEND: R = Read only; -*n* = value after reset

Table 16-13. ESM Interrupt Offset High Register (ESMIOFFHR) Field Descriptions

Bit	Field	Value	Description
31-8	Reserved	0	Reads return 0. Writes have no effect.
7-0	INTOFFH		Offset High-Level Interrupt. This vector gives the channel number of the highest-pending interrupt request for the high-level interrupt line. Interrupts of error Group2 have higher priority than interrupts of error Group1. Inside a group, channel 0 has highest priority and channel 31 has lowest priority.
			User and privileged mode (read):
			Returns number of pending interrupt with the highest priority for the high-level interrupt line.
		0	No pending interrupt.
		1h	Interrupt pending for channel 0, error Group1.
		:	:
		20h	Interrupt pending for channel 31, error Group1.
		21h	Interrupt pending for channel 0, error Group2.
		:	:
		40h	Interrupt pending for channel 31, error Group2.
		41h	Interrupt pending for channel 32, error Group1.
		:	:
		60h	Interrupt pending for channel 63, error Group1.
		61h	Reserved
		:	:
		80h	Reserved
		81h	Interrupt pending for channel 64, error Group1.
		:	:
		A0h	Interrupt pending for channel 95, error Group1.
			Note: Reading the interrupt vector will clear the corresponding flag in the ESMSR2 register; will **not** clear ESMSR1 and ESMSSR2 and the offset register gets updated.
			User and privileged mode (write):
			Writes have no effect.

16.4.12 ESM Interrupt Offset Low Register (ESMIOFFLR)

Figure 16-22. ESM Interrupt Offset Low Register (ESMIOFFLR) [offset = 2Ch]

31–16
Reserved
R-0

15–8	7–0
Reserved	INTOFFL
R-0	R-0

LEGEND: R = Read only; -*n* = value after reset

Table 16-14. ESM Interrupt Offset Low Register (ESMIOFFLR) Field Descriptions

Bit	Field	Value	Description
31-8	Reserved	0	Reads return 0. Writes have no effect.
7-0	INTOFFL		Offset Low-Level Interrupt. This vector gives the channel number of the highest-pending interrupt request for the low-level interrupt line. Inside a group, channel 0 has highest priority and channel 31 has lowest priority.
			User and privileged mode (read):
			Returns number of pending interrupt with the highest priority for the low-level interrupt line.
		0	No pending interrupt.
		1h	Interrupt pending for channel 0, error Group1.
		:	:
		20h	Interrupt pending for channel 31, error Group1.
		21h	Reserved
		:	:
		40h	Reserved
		41h	Interrupt pending for channel 32, error Group1.
		:	:
		60h	Interrupt pending for channel 63, error Group1.
		61h	Reserved
		:	:
		80h	Reserved
		81h	Interrupt pending for channel 64, error Group1.
		:	:
		A0h	Interrupt pending for channel 95, error Group1.
			Note: Reading the interrupt vector will **not** clear the corresponding flag in the ESMSR1 register. Group2 interrupts are fixed to the high level interrupt line only.
			User and privileged mode (write):
			Writes have no effect.

16.4.13 ESM Low-Time Counter Register (ESMLTCR)

Figure 16-23. ESM Low-Time Counter Register (ESMLTCR) [offset = 30h]

31	16
Reserved	
R-0	

15	0
LTC	
R-3FFFh	

LEGEND: R = Read only; -*n* = value after reset

Table 16-15. ESM Low-Time Counter Register (ESMLTCR) Field Descriptions

Bit	Field	Value	Description
31-16	Reserved	0	Reads return 0. Writes have no effect.
15-0	LTC		ERROR Pin Low-Time Counter 16-bit pre-loadable down-counter to control low-time of $\overline{\text{ERROR}}$ pin. The low-time counter is triggered by the peripheral clock (VCLK). **Note:** Low time counter is set to the default pre-load value of the ESMLTCPR in the following cases: 1. Reset (power on reset or warm reset) 2. An error occurs 3. User forces an error

16.4.14 ESM Low-Time Counter Preload Register (ESMLTCPR)

Figure 16-24. ESM Low-Time Counter Preload Register (ESMLTCPR) [offset = 34h]

31	16
Reserved	
R-0	

15	14	13	0
LTCP		LTCP	
R/WP-0		R-3FFFh	

LEGEND: R/W = Read/Write; R = Read; WP = Write in privileged mode only; -*n* = value after reset

Table 16-16. ESM Low-Time Counter Preload Register (ESMLTCPR) Field Descriptions

Bit	Field	Value	Description
31-16	Reserved	0	Reads return 0. Writes have no effect.
15-0	LTCP		$\overline{\text{ERROR}}$ Pin Low-Time Counter Pre-load Value 16-bit pre-load value for the $\overline{\text{ERROR}}$ pin low-time counter. **Note:** Only LTCP.15 and LTCP.14 are configurable (privileged mode write).

16.4.15 ESM Error Key Register (ESMEKR)

Figure 16-25. ESM Error Key Register (ESMEKR) [offset = 38h]

31	16
Reserved	
R-0	

15	4	3	0
Reserved		EKEY	
R-0		R/WP-0	

LEGEND: R/W = Read/Write; R = Read; WP = Write in privileged mode only; -*n* = value after reset

Table 16-17. ESM Error Key Register (ESMEKR) Field Descriptions

Bit	Field	Value	Description
31-4	Reserved	0	Reads return 0. Writes have no effect.
3-0	EKEY		Error Key. The key to reset the $\overline{\text{ERROR}}$ pin or to force an error on the $\overline{\text{ERROR}}$ pin. **User and privileged mode (read):** Returns current value of the EKEY.
			Privileged mode (write):
		0	Activates normal mode (recommended default mode).
		5h	The $\overline{\text{ERROR}}$ pin set to high when the low time counter (LTC) has completed; then the EKEY bit will switch back to normal mode (EKEY = 0000)
		Ah	Forces error on $\overline{\text{ERROR}}$ pin.
		All other values	Activates normal mode.

16.4.16 ESM Status Shadow Register 2 (ESMSSR2)

This register is dedicated for Group2.

Figure 16-26. ESM Status Shadow Register 2 (ESMSSR2) [offset = 3Ch]

31	16
ESF	
R/W1CP-X/0	

15	0
ESF	
R/W1CP-X/0	

LEGEND: R/W = Read/Write; W1CP = Write 1 to clear in privilege mode only; -*n* = value after reset/$\overline{\text{PORRST}}$; *X* = value is unchanged

Table 16-18. ESM Status Shadow Register 2 (ESMSSR2) Field Descriptions

Bit	Field	Value	Description
31-0	ESF		Error Status Flag. Shadow register for status information on pending error. **Read in User and Privileged mode. Write in Privileged mode only.**
		0	Read: No error occurred. Write: Leaves the bit unchanged.
		1	Read: Error occurred. Write: Clears the bit. ESMSR2 is not impacted by this action. **Note:** Errors are stored until they are cleared by the software or at power-on reset ($\overline{\text{PORRST}}$).

16.4.17 ESM Influence $\overline{\text{ERROR}}$ Pin Set/Status Register 4 (ESMIEPSR4)

This register is dedicated for Group1 Channel[63:32].

Figure 16-27. ESM Influence $\overline{\text{ERROR}}$ Pin Set/Status Register 4 (ESMIEPSR4) [offset = 40h]

31	16
IEPSET[63:48]	
R/WP-0	

15	0
IEPSET[47:32]	
R/WP-0	

LEGEND: R/W = Read/Write; WP = Write in privileged mode only; -*n* = value after reset

Table 16-19. ESM Influence $\overline{\text{ERROR}}$ Pin Set/Status Register 4 (ESMIEPSR4) Field Descriptions

Bit	Field	Value	Description
63-32	IEPSET		Set influence on $\overline{\text{ERROR}}$ pin. **Read in User and Privileged mode. Write in Privileged mode only.**
		0	Read: Failure on channel x has no influence on $\overline{\text{ERROR}}$ pin. Write: Leaves the bit and the corresponding clear bit in the ESMIEPCR4 register unchanged.
		1	Read: Failure on channel x has influence on $\overline{\text{ERROR}}$ pin. Write: Enables failure influence on $\overline{\text{ERROR}}$ pin and sets the corresponding clear bit in the ESMIEPCR4 register.

16.4.18 ESM Influence $\overline{\text{ERROR}}$ Pin Clear/Status Register 4 (ESMIEPCR4)

This register is dedicated for Group1 Channel[63:32].

Figure 16-28. ESM Influence $\overline{\text{ERROR}}$ Pin Clear/Status Register 4 (ESMIEPCR4) [offset = 44h]

31	16
IEPCLR[63:48]	
R/WP-0	

15	0
IEPCLR[47:32]	
R/WP-0	

LEGEND: R/W = Read/Write; WP = Write in privileged mode only; -*n* = value after reset

Table 16-20. ESM Influence $\overline{\text{ERROR}}$ Pin Clear/Status Register 4 (ESMIEPCR4) Field Descriptions

Bit	Field	Value	Description
63-32	IEPCLR		Clear influence on $\overline{\text{ERROR}}$ pin. **Read in User and Privileged mode. Write in Privileged mode only.**
		0	Read: Failure on channel x has no influence on $\overline{\text{ERROR}}$ pin. Write: Leaves the bit and the corresponding clear bit in the ESMIEPSR4 register unchanged.
		1	Read: Failure on channel x has influence on $\overline{\text{ERROR}}$ pin. Write: Disables failure influence on $\overline{\text{ERROR}}$ pin and clears the corresponding clear bit in the ESMIEPSR4 register.

16.4.19 ESM Interrupt Enable Set/Status Register 4 (ESMIESR4)

This register is dedicated for Group1 Channel[63:32].

Figure 16-29. ESM Interrupt Enable Set/Status Register 4 (ESMIESR4) [offset = 48h]

31	16
INTENSET[63:48]	
R/WP-0	

15	0
INTENSET[47:32]	
R/WP-0	

LEGEND: R/W = Read/Write; WP = Write in privileged mode only; -*n* = value after reset

Table 16-21. ESM Interrupt Enable Set/Status Register 4 (ESMIESR4) Field Descriptions

Bit	Field	Value	Description
63-32	INTENSET		Set interrupt enable. **Read in User and Privileged mode. Write in Privileged mode only.**
		0	Read: Interrupt is disabled. Write: Leaves the bit and the corresponding clear bit in the ESMIECR4 register unchanged.
		1	Read: Interrupt is enabled. Write: Enables interrupt and sets the corresponding clear bit in the ESMIECR4 register.

16.4.20 ESM Interrupt Enable Clear/Status Register 4 (ESMIECR4)

This register is dedicated for Group1 Channel[63:32].

Figure 16-30. ESM Interrupt Enable Clear/Status Register 4 (ESMIECR4) [offset = 4Ch]

31	16
INTENCLR[63:48]	
R/WP-0	

15	0
INTENCLR[47:32]	
R/WP-0	

LEGEND: R/W = Read/Write; WP = Write in privileged mode only; -*n* = value after reset

Table 16-22. ESM Interrupt Enable Clear/Status Register 4 (ESMIECR4) Field Descriptions

Bit	Field	Value	Description
63-32	INTENCLR		Clear interrupt enable. **Read in User and Privileged mode. Write in Privileged mode only.**
		0	Read: Interrupt is disabled. Write: Leaves the bit and the corresponding clear bit in the ESMIESR4 register unchanged.
		1	Read: Interrupt is enabled. Write: Disables interrupt and clears the corresponding clear bit in the ESMIESR4 register.

16.4.21 ESM Interrupt Level Set/Status Register 4 (ESMILSR4)

This register is dedicated for Group1 Channel[63:32].

Figure 16-31. ESM Interrupt Level Set/Status Register 4 (ESMILSR4) [offset = 50h]

31	16
INTLVLSET[63:48]	
R/WP-0	

15	0
INTLVLSET[47:32]	
R/WP-0	

LEGEND: R/W = Read/Write; WP = Write in privileged mode only; -*n* = value after reset

Table 16-23. ESM Interrupt Level Set/Status Register 4 (ESMILSR4) Field Descriptions

Bit	Field	Value	Description
63-32	INTLVLSET		Set interrupt level. **Read in User and Privileged mode. Write in Privileged mode only.**
		0	Read: Read: Interrupt of channel x is mapped to low-level interrupt line. Write: Leaves the bit and the corresponding clear bit in the ESMILCR4 register unchanged.
		1	Read: Interrupt of channel x is mapped to high-level interrupt line. Write: Maps interrupt of channel x to high-level interrupt line and sets the corresponding clear bit in the ESMILCR4 register.

16.4.22 ESM Interrupt Level Clear/Status Register 4 (ESMILCR4)

This register is dedicated for Group1 Channel[63:32].

Figure 16-32. ESM Interrupt Level Clear/Status Register 4 (ESMILCR4) [offset = 54h]

31	16
INTLVLCLR[63:48]	
R/WP-0	

15	0
INTLVLCLR[47:32]	
R/WP-0	

LEGEND: R/W = Read/Write; WP = Write in privileged mode only; -*n* = value after reset

Table 16-24. ESM Interrupt Level Clear/Status Register 4 (ESMILCR4) Field Descriptions

Bit	Field	Value	Description
63-32	INTLVLCLR		Clear interrupt level. **Read in User and Privileged mode. Write in Privileged mode only.**
		0	Read: Interrupt of channel x is mapped to low-level interrupt line. Write: Leaves the bit and the corresponding set bit in the ESMILSR4 register unchanged.
		1	Read: Interrupt of channel x is mapped to high-level interrupt line. Write: Maps interrupt of channel x to low-level interrupt line and clears the corresponding set bit in the ESMILSR4 register.

16.4.23 ESM Status Register 4 (ESMSR4)

This register is dedicated for Group1 Channel[63:32].

Figure 16-33. ESM Status Register 4 (ESMSR4) [offset = 58h]

31	16
ESF[63:48]	
R/W1CP-X/0	

15	0
ESF[47:32]	
R/W1CP-X/0	

LEGEND: R/W = Read/Write; W1CP = Write 1 to clear in privilege mode only; -*n* = value after reset/$\overline{\text{PORRST}}$; *X* = value is unchanged

Table 16-25. ESM Status Register 4 (ESMSR4) Field Descriptions

Bit	Field	Value	Description
63-32	ESF		Error Status Flag. Provides status information on a pending error. **Read in User and Privileged mode. Write in Privileged mode only.**
		0	Read: No error occurred; no interrupt is pending. Write: Leaves the bit unchanged.
		1	Read: Error occurred; interrupt is pending. Write: Clears the bit. **Note:** After $\overline{\text{RST}}$, if one of these flags are set and the corresponding interrupt are enabled, the interrupt service routine will be called.

16.4.24 ESM Influence $\overline{\text{ERROR}}$ Pin Set/Status Register 7 (ESMIEPSR7)

This register is dedicated for Group1 Channel[95:64].

Figure 16-34. ESM Influence $\overline{\text{ERROR}}$ Pin Set/Status Register 7 (ESMIEPSR7) [offset = 80h]

31	16
IEPSET[95:80]	
R/WP-0	

15	0
IEPSET[79:64]	
R/WP-0	

LEGEND: R/W = Read/Write; WP = Write in privileged mode only; -*n* = value after reset

Table 16-26. ESM Influence $\overline{\text{ERROR}}$ Pin Set/Status Register 7 (ESMIEPSR7) Field Descriptions

Bit	Field	Value	Description
95-64	IEPSET		Set influence on $\overline{\text{ERROR}}$ pin. **Read in User and Privileged mode. Write in Privileged mode only.**
		0	Read: Failure on channel x has no influence on $\overline{\text{ERROR}}$ pin. Write: Leaves the bit and the corresponding clear bit in the ESMIEPCR7 register unchanged.
		1	Read: Failure on channel x has influence on $\overline{\text{ERROR}}$ pin. Write: Enables failure influence on $\overline{\text{ERROR}}$ pin and sets the corresponding clear bit in the ESMIEPCR7 register.

16.4.25 ESM Influence $\overline{\text{ERROR}}$ Pin Clear/Status Register 7 (ESMIEPCR7)

This register is dedicated for Group1 Channel[95:64].

Figure 16-35. ESM Influence $\overline{\text{ERROR}}$ Pin Clear/Status Register 7 (ESMIEPCR7) [offset = 84h]

31	16
IEPCLR[95:80]	
R/WP-0	

15	0
IEPCLR[79:64]	
R/WP-0	

LEGEND: R/W = Read/Write; WP = Write in privileged mode only; -*n* = value after reset

Table 16-27. ESM Influence $\overline{\text{ERROR}}$ Pin Clear/Status Register 7 (ESMIEPCR7) Field Descriptions

Bit	Field	Value	Description
95-64	IEPCLR		Clear influence on $\overline{\text{ERROR}}$ pin. **Read in User and Privileged mode. Write in Privileged mode only.**
		0	Read: Failure on channel x has no influence on $\overline{\text{ERROR}}$ pin. Write: Leaves the bit and the corresponding clear bit in the ESMIEPSR7 register unchanged.
		1	Read: Failure on channel x has influence on $\overline{\text{ERROR}}$ pin. Write: Disables failure influence on $\overline{\text{ERROR}}$ pin and clears the corresponding clear bit in the ESMIEPSR7 register.

16.4.26 ESM Interrupt Enable Set/Status Register 7 (ESMIESR7)

This register is dedicated for Group1 Channel[95:64].

Figure 16-36. ESM Interrupt Enable Set/Status Register 7 (ESMIESR7) [offset = 88h]

31	16
INTENSET[95:80]	
R/WP-0	

15	0
INTENSET[79:64]	
R/WP-0	

LEGEND: R/W = Read/Write; WP = Write in privileged mode only; -*n* = value after reset

Table 16-28. ESM Interrupt Enable Set/Status Register 7 (ESMIESR7) Field Descriptions

Bit	Field	Value	Description
95-64	INTENSET		Set interrupt enable. **Read in User and Privileged mode. Write in Privileged mode only.**
		0	Read: Interrupt is disabled. Write: Leaves the bit and the corresponding clear bit in the ESMIECR7 register unchanged.
		1	Read: Interrupt is enabled. Write: Enables interrupt and sets the corresponding clear bit in the ESMIECR7 register.

16.4.27 ESM Interrupt Enable Clear/Status Register 7 (ESMIECR7)

This register is dedicated for Group1 Channel[95:64].

Figure 16-37. ESM Interrupt Enable Clear/Status Register 7 (ESMIECR7) [offset = 8Ch]

31	16
INTENCLR[95:80]	
R/WP-0	

15	0
INTENCLR[79:64]	
R/WP-0	

LEGEND: R/W = Read/Write; WP = Write in privileged mode only; -*n* = value after reset

Table 16-29. ESM Interrupt Enable Clear/Status Register 7 (ESMIECR7) Field Descriptions

Bit	Field	Value	Description
95-64	INTENCLR		Clear interrupt enable. **Read in User and Privileged mode. Write in Privileged mode only.**
		0	Read: Interrupt is disabled. Write: Leaves the bit and the corresponding clear bit in the ESMIESR7 register unchanged.
		1	Read: Interrupt is enabled. Write: Disables interrupt and clears the corresponding clear bit in the ESMIESR7 register.

16.4.28 ESM Interrupt Level Set/Status Register 7 (ESMILSR7)

This register is dedicated for Group1 Channel[95:64].

Figure 16-38. ESM Interrupt Level Set/Status Register 7 (ESMILSR7) [offset = 90h]

31	16
INTLVLSET[95:80]	
R/WP-0	

15	0
INTLVLSET[79:64]	
R/WP-0	

LEGEND: R/W = Read/Write; WP = Write in privileged mode only; -*n* = value after reset

Table 16-30. ESM Interrupt Level Set/Status Register 7 (ESMILSR7) Field Descriptions

Bit	Field	Value	Description
95-64	INTLVLSET		Set interrupt level. **Read in User and Privileged mode. Write in Privileged mode only.**
		0	Read: Read: Interrupt of channel x is mapped to low-level interrupt line. Write: Leaves the bit and the corresponding clear bit in the ESMILCR7 register unchanged.
		1	Read: Interrupt of channel x is mapped to high-level interrupt line. Write: Maps interrupt of channel x to high-level interrupt line and sets the corresponding clear bit in the ESMILCR7 register.

16.4.29 ESM Interrupt Level Clear/Status Register 7 (ESMILCR7)

This register is dedicated for Group1 Channel[95:64].

Figure 16-39. ESM Interrupt Level Clear/Status Register 7 (ESMILCR7) [offset = 94h]

31	16
INTLVLCLR[95:80]	
R/WP-0	

15	0
INTLVLCLR[79:64]	
R/WP-0	

LEGEND: R/W = Read/Write; WP = Write in privileged mode only; -*n* = value after reset

Table 16-31. ESM Interrupt Level Clear/Status Register 7 (ESMILCR7) Field Descriptions

Bit	Field	Value	Description
95-64	INTLVLCLR		Clear interrupt level. **Read in User and Privileged mode. Write in Privileged mode only.**
		0	Read: Interrupt of channel x is mapped to low-level interrupt line. Write: Leaves the bit and the corresponding set bit in the ESMILSR7 register unchanged.
		1	Read: Interrupt of channel x is mapped to high-level interrupt line. Write: Maps interrupt of channel x to low-level interrupt line and clears the corresponding set bit in the ESMILSR7 register.

16.4.30 ESM Status Register 7 (ESMSR7)

This register is dedicated for Group1 Channel[95:64].

Figure 16-40. ESM Status Register 7 (ESMSR7) [offset = 98h]

31	16
ESF[95:80]	
R/W1CP-X/0	

15	0
ESF[79:64]	
R/W1CP-X/0	

LEGEND: R/W = Read/Write; W1CP = Write 1 to clear in privilege mode only; -*n* = value after reset/$\overline{\text{PORRST}}$; *X* = value is unchanged

Table 16-32. ESM Status Register 7 (ESMSR7) Field Descriptions

Bit	Field	Value	Description
95-64	ESF		Error Status Flag. Provides status information on a pending error. **Read in User and Privileged mode. Write in Privileged mode only.**
		0	Read: No error occurred; no interrupt is pending. Write: Leaves the bit unchanged.
		1	Read: Error occurred; interrupt is pending. Write: Clears the bit. **Note:** After $\overline{\text{RST}}$, if one of these flags are set and the corresponding interrupt are enabled, the interrupt service routine will be called.

Real-Time Interrupt (RTI) Module

This chapter describes the functionality of the real-time interrupt (RTI) module. The RTI is designed as an operating system timer to support a real time operating system (RTOS).

NOTE: This chapter describes a superset implementation of the RTI module that includes features and functionality related to DMA, FlexRay, and Timbase control. These features are dependent on the device-specific feature content. Consult your device-specific datasheet to determine the applicability of these features to your device being used.

Topic **Page**

17.1 Overview

The real-time interrupt (RTI) module provides timer functionality for operating systems and for benchmarking code. The RTI module can incorporate several counters that define the timebases needed for scheduling in the operating system.

The timers also allow you to benchmark certain areas of code by reading the values of the counters at the beginning and the end of the desired code range and calculating the difference between the values.

In addition the RTI provides a mechanism to synchronize the operating system to the FlexRay communication cycle. Clock supervision allows for detection of issues on the FlexRay bus with an automatic switch to an internally generated timebase when a failure with the FlexRay timebase is detected.

17.1.1 Features

The RTI module has the following features:

- Two independent 64 bit counter blocks
- Four configurable compares for generating operating system ticks or DMA requests. Each event can be driven by either counter block 0 or counter block 1.
- One counter block usable for application synchronization to FlexRay network including clock supervision
- Fast enabling/disabling of events
- Two time stamp (capture) functions for system or peripheral interrupts, one for each counter block
- Digital windowed watchdog

17.1.2 Industry Standard Compliance Statement

This module is specifically designed to fulfill the requirements for OSEK (**O**ffene **S**ysteme und deren **S**chnittstellen für die **E**lektronik im **K**raftfahrzeug, or Open Systems and the Corresponding Interfaces for Automotive Electronics) as well as OSEK/time-compliant operating systems, but is not limited to it.

17.2 Module Operation

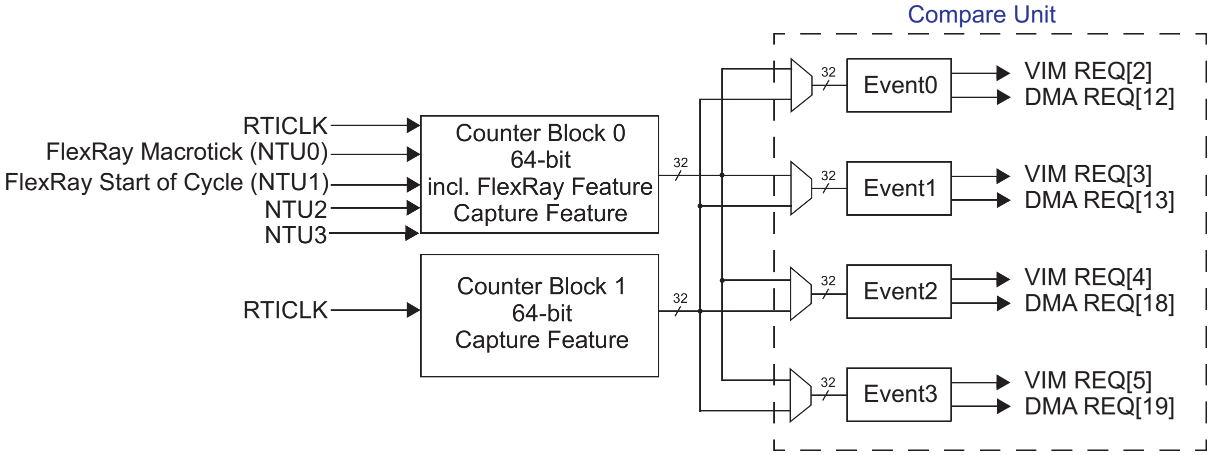
Figure 17-1 illustrates the high level block diagram of the RTI module.

The RTI module has two independent counter blocks for generating different timebases: counter block 0 and counter block 1. The two counter blocks provide the same basic functionality, but counter block 0 has the additional functionality of being able to work with the FlexRay Macrotick (NTU0) or Start of Cycle (NTU1) and perform clock supervision to detect a missing signal.

A compare unit compares the counters with programmable values and generates four independent interrupt or DMA requests on compare matches. Each of the compare registers can be programmed to be compared to either counter block 0 or counter block 1.

The following sections describe the individual functions in more detail.

Figure 17-1. RTI Block Diagram



17.2.1 Counter Operation

Each counter block consists of the following (see Figure 17-2):

- One 32-bit prescale counter (RTIUC0 or RTIUC1)
- One 32-bit free running counter (RTIFRC0 or RTIFRC1)

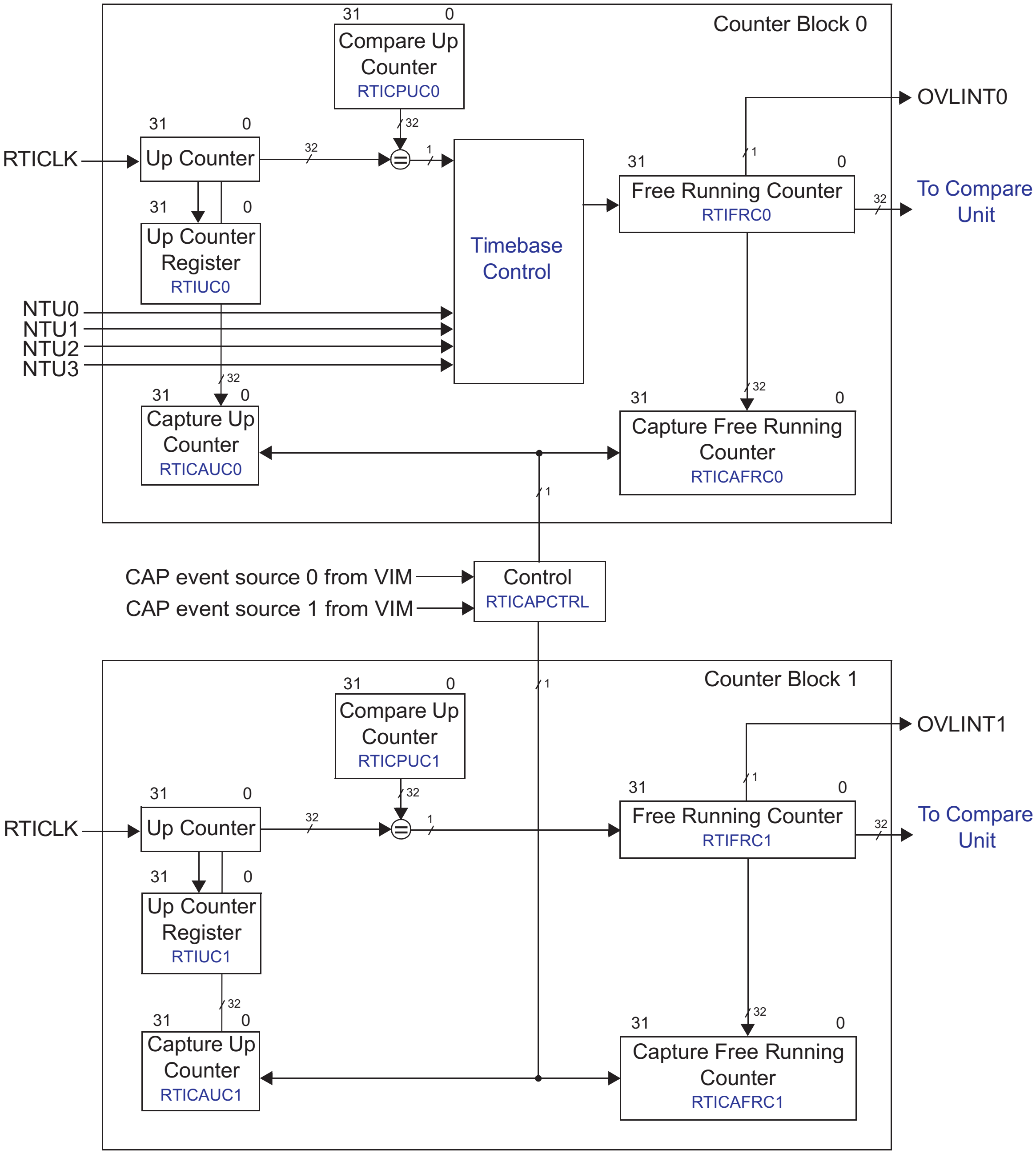
The RTIUC0/1 is driven by the RTICLK and counts up until the compare value in the compare up counter register (RTICPUC0 or RTICPUC1) is reached. When the compare matches, RTIFRC0/1 is incremented and RTIUC0/1 is reset to 0. If RTIFRC0/1 overflows, an interrupt is generated to the vectored interrupt manager (VIM). The overflow interrupt is not intended to generate the timebase for the operating system. See Section 17.2.2 for the timebase generation. The up counter together with the compare up counter value prescale the RTI clock. The resulting formula for the frequency of the free running counter (RTIFRC0/1) is:

$$f_{RTIFRCx} = \begin{cases} \dfrac{f_{RTICLK}}{RTICPUCx + 1} & \text{when RTICPUCx} \neq 0 \\[2ex] \dfrac{f_{RTICLK}}{2^{32}+1} & \text{when RTICPUCx} = 0 \end{cases} \quad (23)$$

NOTE: Setting RTICPUCx equal to zero is not recommended. Doing so will hold the Up Counter at zero for two RTICLK cycles after it overflows from 0xFFFFFFFF to zero.

The counter values can be determined by reading the respective counter registers or by generating a hardware event which captures the counter value into the respective capture register. Both functions are described in the following sections.

Figure 17-2. Counter Block Diagram



17.2.1.1 Counter and Capture Read Consistency

Portions of the device internal databus are 32-bits wide. If the application wants to read the 64-bit counters or the 64-bit capture values, a certain order of 32-bit read operations needs to be followed. This is to prevent one counter incrementing in between the two separate read operations to both counters.

Reading the Counters

The free running counter (RTIFRCx) must be read first. This priority will ensure that in the cycle when the CPU reads RTIFRCx, the up counter value is stored in its counter register (RTIUCx). The second read has to access the up counter register (RTIUCx), which then holds the value which corresponds to the number of RTICLK cycles that have elapsed at the time reading the free running counter register (RTIFRCx).

NOTE: The up counters are implemented as shadow registers. Reading RTIUCx without having read RTIFRCx first will return always the same value. RTIUCx will only be updated when RTIFRCx is read.

Reading the Capture Values

The free running counter capture register (RTICAFRCx) must be read first. This priority will ensure that in the cycle when the CPU reads RTICAFRCx, the up counter value is stored in its counter register (RTICAUCx). The second read has to access the up counter register (RTICAUCx), which then holds the value captured at the time when reading the capture free running counter register (RTICAFRCx).

NOTE: The capture up counter registers are implemented as shadow registers. Reading RTICAUCx without having read RTICAFRCx first will return always the same value. RTICAUCx will only be updated when RTICAFRCx is read.

17.2.1.2 Capture Feature

Both counter blocks also provide a capture feature on external events. Two capture sources can trigger the capture event. The source triggering the block is configurable (RTICAPCTRL). The sources originate from the Vectored Interrupt Manager (VIM) and allow the generation of capture events when a peripheral modules has generated an interrupt. Any of the peripheral interrupts can be selected as the capture event in the VIM.

When an event is detected, RTIUCx and RTIFRCx are stored in the capture up counter (RTICAUCx) and capture free running counter (RTICAFRCx) registers. The read order of the captured values must be the same as the read order of the actual counters (see Section 17.2.1.1).

17.2.2 Interrupt/DMA Requests

There are four compare registers (RTICOMPy) to generate interrupt requests to the VIM or DMA requests to the DMA controller. The interrupts can be used to generate different timebases for the operating system. Each of the compare registers can be configured to be compared to either RTIFRC0 or RTIFRC1. When the counter value matches the compare value, an interrupt is generated. To allow periodic interrupts, a certain value can be added to the compare value in RTICOMPy automatically. This value is stored in the update compare register (RTIUDCPy) and will be added after a compare is matched. The period of the generated interrupt/DMA request can be calculated with:

$$t_{COMPx} = t_{RTICLK} \times (RTICPUCy + 1) \times RTIUDCPy$$

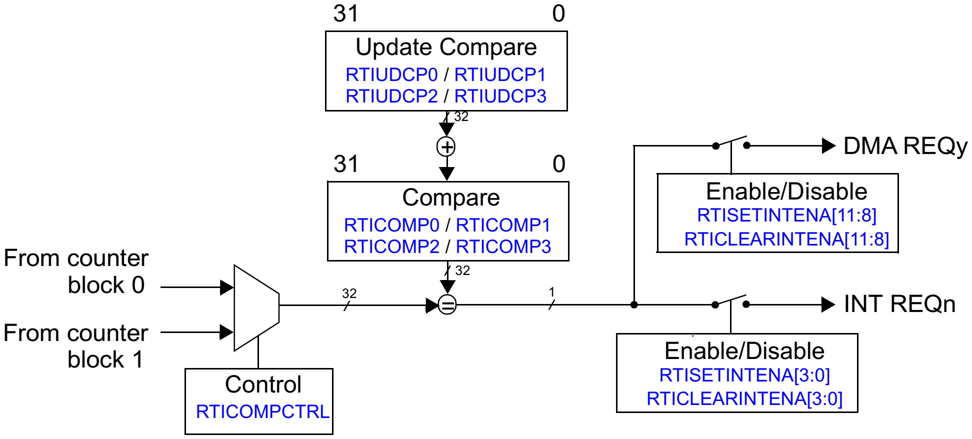
if RTICPUCy ≠ 0,

$$t_{COMPx} = t_{RTICLK} \times (2^{32}+1) \times RTIUDCPy$$

if RTIUDCPy = 0,

$$t_{COMPx} = t_{RTICLK} \times (RTICPUCy + 1) \times 2^{32} \tag{24}$$

Figure 17-3. Compare Unit Block Diagram (shows only 1 of 4 blocks for simplification)



Another interrupt that can be generated is the overflow interrupt (OVLINTx) in case the RTIFRCx counter overflows.

The interrupts/DMA requests can be enabled in the RTISETINTENA register and disabled in the RTICLEARINTENA register. The RTIINTFLAG register shows the pending interrupts.

17.2.3 RTI Clocking

The counter blocks are clocked with RTICLK (for definition see Section 2.4.2). Counter block 0 can be clocked in addition by either the FlexRay Macrotick (NTU0) or the FlexRay Start of Cycle (NTU1).

A clock supervision for the NTUx clocking scheme is implemented to avoid missing operating system ticks.

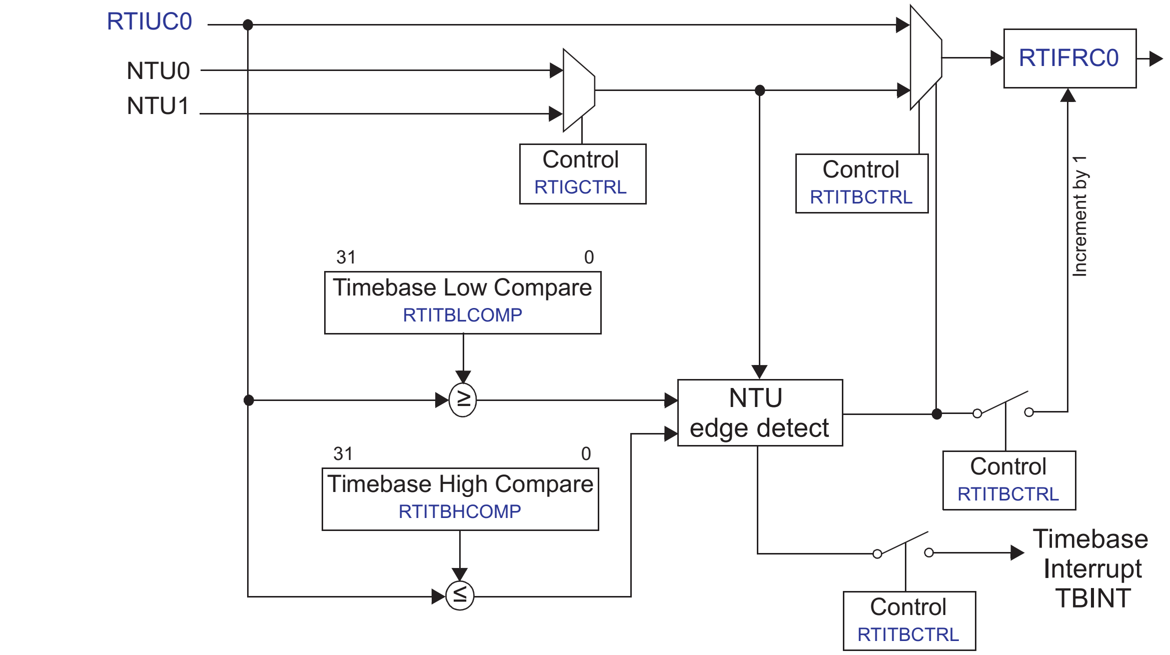
17.2.4 Synchronizing Timer Events to Network Time (NTU)

For applications which are participating on a time-triggered communication bus, it is often beneficial to synchronize the application or operating system to the network time. The RTI provides a feature to increment Free Running Counter 0 (RTIFRC0) by a periodic clock provided by the communication module. In this case two different clocks can be chosen. One is the FlexRay module Macrotick (NTU0) and the other is the Start of Cycle (NTU1) information of the same module.

The application has control over which clock (RTICLK, NTU0, NTU1) should be used for clocking RTIFRC0. If NTUx is used, a clock supervision circuit allows to monitor this clock and provides a fallback solution, should the clock be non-functional (missing). A too fast running NTUx cannot be detected.

RTIUC0 is utilized to monitor the NTUx signal. A detection window can be programmed in which a valid NTU clock pulse needs to occur. If no pulse is detected, the RTI automatically switches back to clock the Free Running Counter 0 with RTIUC0. In order to avoid a big jitter in the operating ticks, in case a switch back to RTIUC0 happens, RTICPUC0 should be set to a value so the clock frequency RTIUC0 outputs is approximately the same as the NTUx frequency.

Figure 17-4. Timebase Control



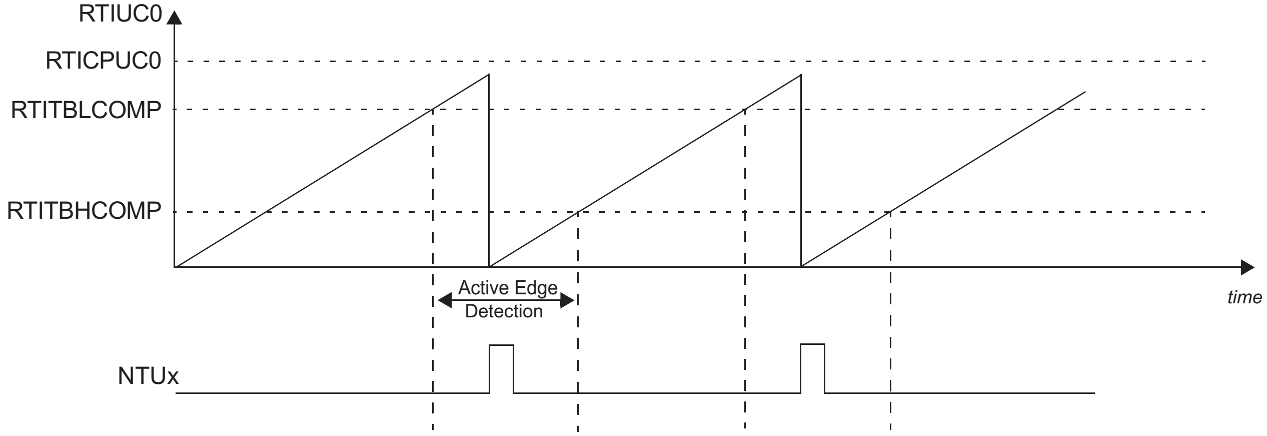
17.2.4.1 Detecting Clock Edges

To detect clock edges on the NTUx signal, the timebase low compare has to be set lower or equal than the value stored in the RTICPUC0 register and the timebase high compare has to be set higher than 0 and lower than the timebase low compare value. This effectively opens a window in which an edge of the NTUx signal is expected (see Figure 17-5). Outside this window, no edges will be detected. If no edge will occur inside the detection window, the multiplexer is switched to internal timebase. The application can select to generate a timebase interrupt (TBINT) and if the INC bit is set, also will automatically increment RTIFRC0 by one to compensate for the missed clock cycle of NTUx. If an edge occurs inside the window, RTIUC0 will be reset to synchronize the two timebases.

In order to make the edge detection work properly, the value in RTICPUC0 needs to be adapted so that RTIUC0 has a similar period as NTUx.

NOTE: To ensure the NTUx signal is properly detected, the NTUx period must be at least twice as long as the RTICLK period.

Figure 17-5. Clock Detection Scheme



17.2.4.2 Switching from Internal Source to External Source

If the application switches from an internal source to an external source, the two signals must be synchronized (see Figure 17-6). The synchronization will occur when the TBEXT bit is set. RTIUC0 will be reset and the edge detection circuit will be active for one (RTICPUC0 + RTITBHCOMP) period or until an edge is detected. If there is no pulse during this period, the source will be reset from an external clock source to an internal clock source. If an edge is detected, the windowed edge detection behavior will take place. Setting the TBEXT bit will not increment free running counter 0.

NOTE: If an external timebase is used, then the software must ensure that timebase low compare and timebase high compare are programmed to a valid state before switching TBEXT to an external source. This state is necessary to allow the timebase control circuit to operate correctly. The following condition must be met:

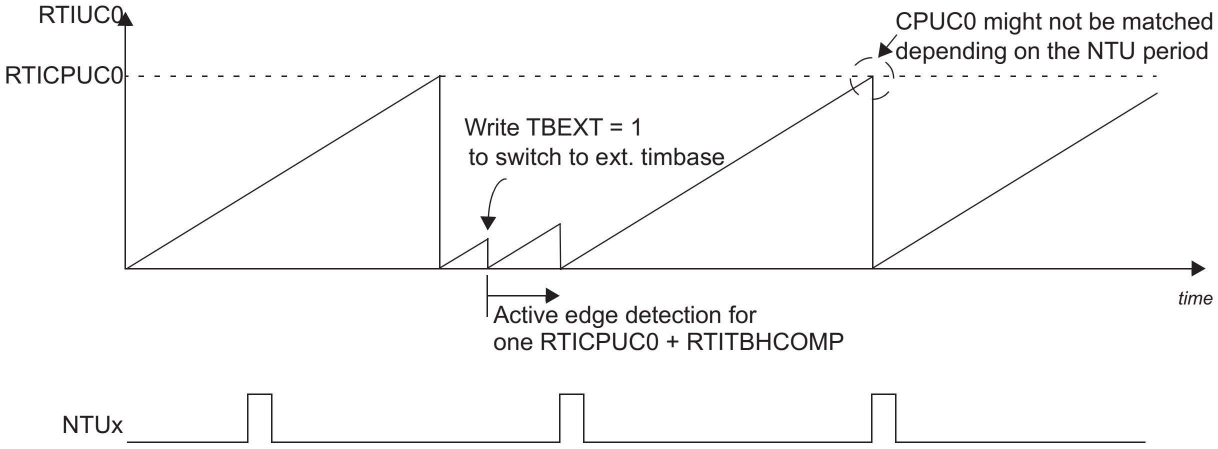
- RTITBHCOMP < RTITBLCOMP + RTICPUC0

RTITBHCOMP must be lower than RTICPUC0 because RTIUC0 will be reset if RTICPUC0 is reached. RTITBHCOMP will represent the number of RTICLK cycles from RTICPUC0 until the circuit switches to the internal timebase when no NTU edge is detected.

If an external timebase is used, RTIGCTRL[0] must be set to 1 (enable RTIUC0) to ensure that the timebase control circuit does not wait indefinitely for an incoming signal.

Figure 17-6 shows a timing example for the synchronization phase when the TBEXT bit is set.

Figure 17-6. Switch to NTUx

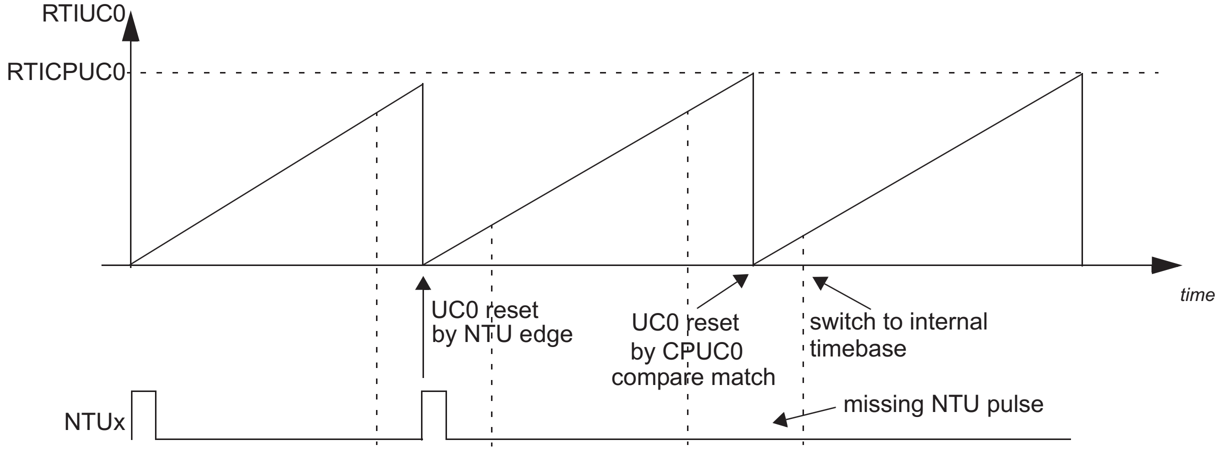

RTIUC0
RTICPUC0
CPUC0 might not be matched
depending on the NTU period
Write TBEXT = 1
to switch to ext. timebase
time
Active edge detection for
one RTICPUC0 + RTITBHCOMP
NTUx

17.2.4.3 Switching from External Source to Internal Source

When the edge detection is active (TBEXT = 1) and no clock edge of NTUx is detected inside the programmed detection window, the RTI will automatically switch the timebase to RTIUC0. Figure 17-7 shows a timing example for a missing NTU signal. In the case where the INC bit is set, RTIFRC0 will automatically be incremented by one to compensate for the missed NTU pulse.

Setting TBEXT = 0 will also switch the clock source for RTIFRC0 to RTIUC0.

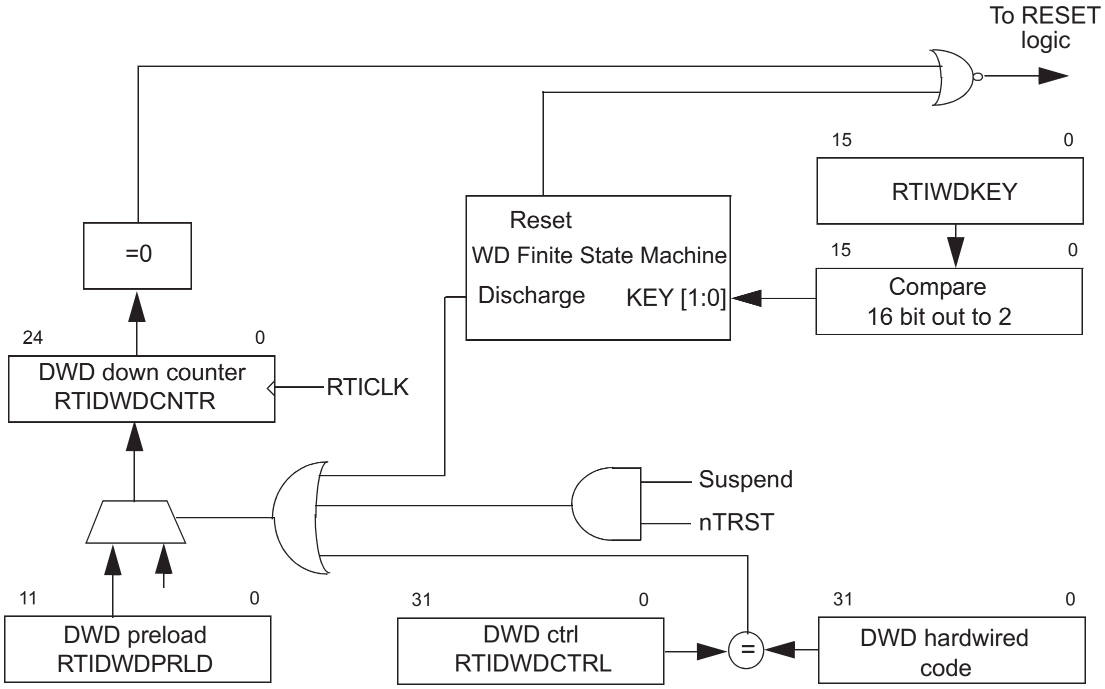
Figure 17-7. Missing NTUx Signal Example



17.2.5 Digital Watchdog (DWD)

The digital watchdog (DWD) is an optional safety diagnostic which can detect a runaway CPU and generate either a reset or NMI (non-maskable interrupt) response. It generates resets or NMIs after a programmable period, or if no correct key sequence was written to the RTIWDKEY register. Figure 17-8 illustrates the DWD.

Figure 17-8. Digital Watchdog



17.2.5.1 Digital Watchdog (DWD)

The DWD is disabled by default. If it should be used, it must be enabled by writing a 32-bit value to the RTIDWDCTRL register.

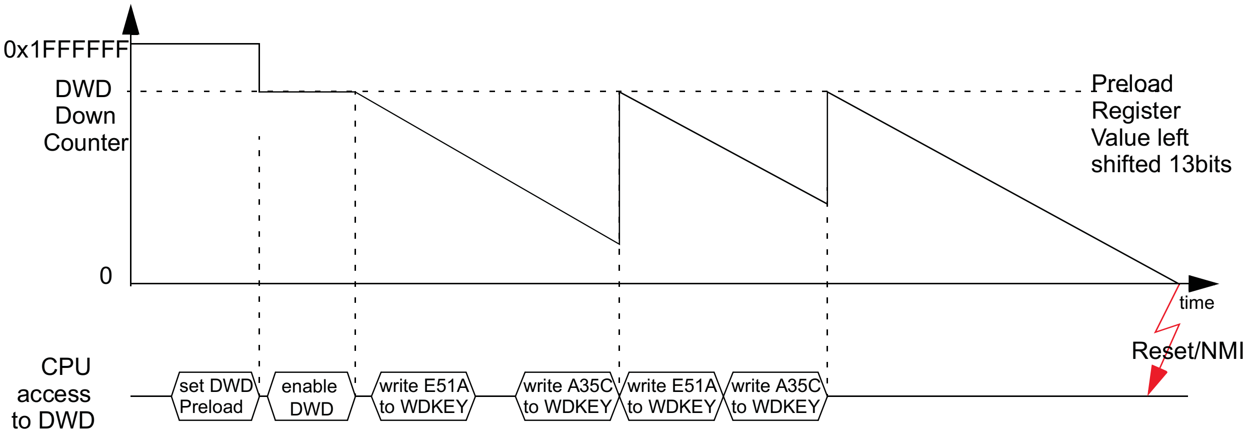
NOTE: Once the DWD is enabled, it cannot be disabled except by system reset or power on reset.

If the correct key sequence is written to the RTIWDKEY register (0xE51A followed by 0xA35C), the 25-bit DWD down counter is reloaded with the left justified 12-bit preload value stored in RTIDWDPRLD. If an incorrect value is written, a watchdog reset or NMI will occur immediately. A reset or NMI will also be generated when the DWD down counter is decremented to 0.

While the device is in suspend mode (halting debug mode), the DWD down counter keeps the value it had when entering suspend mode.

The DWD down counter will be decremented with the RTICLK frequency.

Figure 17-9. DWD Operation



The expiration time of the DWD down counter can be determined with the following equation:

$$texp = (DWDPRLD + 1) \times 2^{13}/RTICLK$$

where

DWDPRLD = 0...4095

NOTE: Care should be taken to ensure that the CPU write to the watchdog register is made allowing time for the write to propagate to the RTI.

17.2.5.2 Digital Windowed Watchdog (DWWD)

In addition to the time-out boundary configurable via the digital watchdog discussed in Section 17.2.5.1, for enhanced safety metrics it is desirable to check for a watchdog "pet" within a time window rather than using a single time threshold. This is enabled by the digital windowed watchdog (DWWD) feature.

- Functional Behavior

The DWWD opens a configurable time window in which the watchdog must be serviced. Any attempt to service the watchdog outside this time window, or a failure to service the watchdog in this time window, will cause the watchdog to generate either a reset or a NMI to the CPU. This is controlled by configuring the RTIWWDRXNCTRL register. As with the DWD, the DWWD is disabled after power on reset. When the DWWD is configured to generate a non-maskable interrupt on a window violation, the watchdog counter continues to count down. The NMI handler needs to clear the watchdog violation status flag(s) and then service the watchdog by writing the correct sequence in the watchdog key register. This service will cause the watchdog counter to get reloaded from the preload value and start counting down. If the NMI handler does not service the watchdog in time, it could count down all the way to zero and wrap around. If the NMI Handler does not service the watchdog in time, the NMI gets generated continuously, each time the counter counts to '0'.

The DWWD uses the Digital Watchdog (DWD) preload register (RTIDWDPRLD) setting to define the end-time of the window. The start-time of the window is defined by a window size configuration register(RTIWWDSIZECTRL).

The default window size is set to 100%, which corresponds to the DWD functionality of a time-out-only watchdog. The window size can be selected (through register RTIWWDSIZECTRL) from among 100%, 50%, 25%, 12.5%, 6.25% and 3.125% as shown in Figure 17-10. The window with the respective size will be opened before the end of the DWD expiration. The user has to serve the watchdog in the window. Otherwise, a reset or NMI will generate. Figure 17-11 shows an DWWD operation example (25% window).

- Configuration of DWWD

The DWWD preload value (same as DWD preload) can only be configured when the DWWD counter is disabled. The window size and watchdog reaction to a violation can be configured even after the watchdog has been enabled. Any changes to the window size and watchdog reaction configurations will only take effect after the next servicing of the DWWD. This feature can be utilized to dynamically set windows of different sizes based on task execution time, adding a program sequence element to the diagnostic which can improve fault coverage.

Figure 17-10. Digital Windowed Watchdog Timing Example

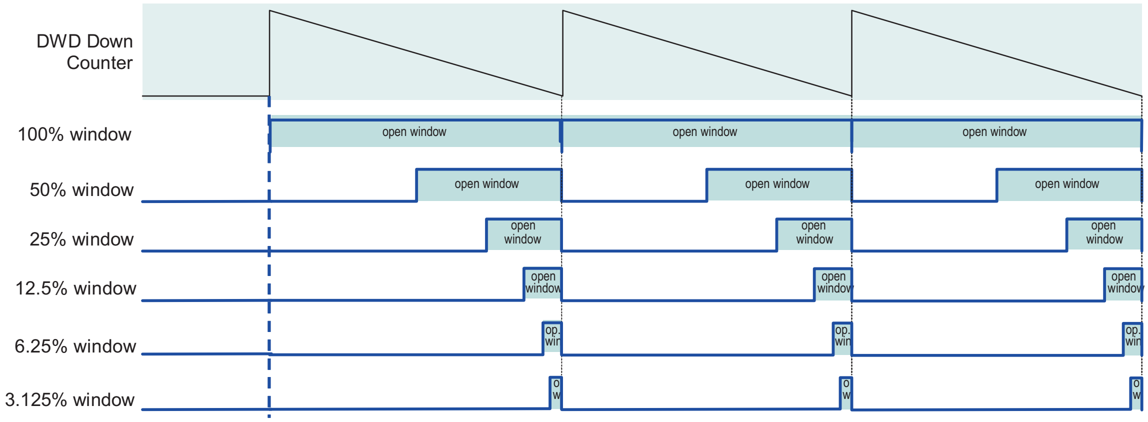
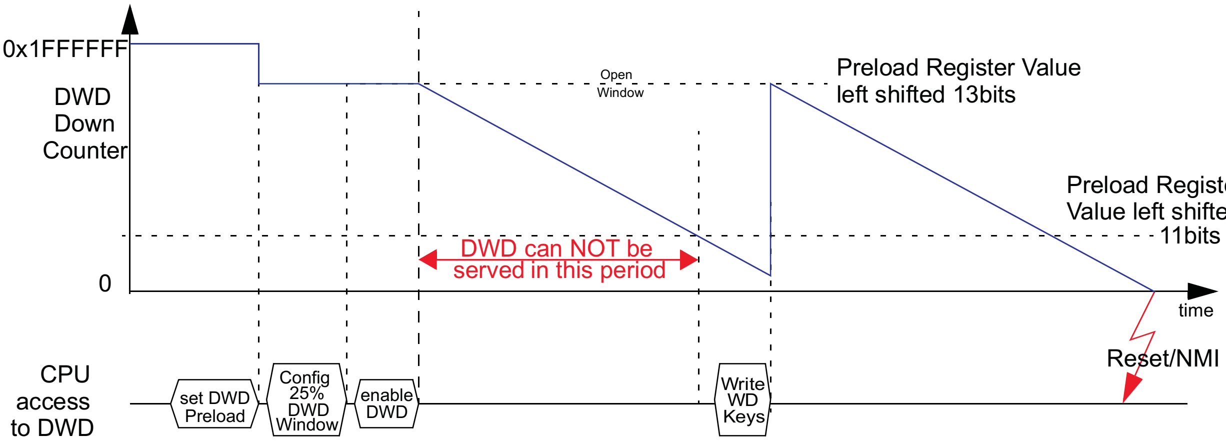


Figure 17-11. Digital Windowed Watchdog Operation Example (25% Window)



17.2.6 Low Power Modes

Low power modes allow the trade off of the current used during low power versus functionality and fast wakeup response. All low power modes have the following characteristics:

- CPU and system clocks are disabled.
- Flash banks and pump are in sleep mode.
- All peripheral modules are in low power modes and the clocks are disabled (exceptions to this may occur and would be documented in the specific device data sheet).

Flexibility in enabling and disabling clocks allows for many different low-power modes (see Section 2.4.3).

The operation of the RTI Module is guaranteed in Run, Doze and Snooze modes. In Sleep mode, all clocks will be switched off and the RTI will not work.

In Doze and Snooze modes, the RTI is active and is able to wake up the device with compare, timebase and overflow interrupts. The compare interrupts can be used to periodically wake up the device. The overflow interrupt can be used to notify the operating system that a counter overflow has occurred. Capturing events generated by the Vectored Interrupt Module (VIM) is also possible since, in both of these low power modes, the peripheral modules are able to generate interrupts that can trigger capture events. Capturing events while in Sleep mode is not supported as the clock to the RTI is not active.

When the device is put into low power mode, the peripheral which is generating the external clock NTU is no longer active, and the timebase control circuitry has to switch to an internal clocking scheme when it detects a missing clock on NTU. The timebase interrupt will wake up the device and the application software has to adapt the periodic interrupt generation to the internal clock source.

DMA transfers will be disabled, and DMA requests will not be generated after device wakeup since the DMA controller will be powered down.

NOTE: RTICLK in Doze Mode

In the special case of Doze Mode with PLL off, RTICLK might have a different period than with PLL enabled since RTICLK will be derived from the oscillator output. It has to be ensured that the VCLK to RTICLK ratio is at least 3:1.

17.2.7 Halting Debug Mode Behaviour

Once the system enters halting debug mode, the behavior of the RTI depends on the COS (continue on suspend) bit. If the bit is cleared and halting debug mode is active, all counters will stop operation. If the bit is set to one, all counters will be clocked normally and the RTI will work like in normal mode. However, if the external timebase (NTU) is used and the system is in halting debug mode, the timebase control circuit will switch to internal timebase once it detects the missing NTU signal of the suspended communication controller. This will be signaled with an TBINT interrupt so that software can resynchronize after the device exits halting debug mode.

17.3 RTI Control Registers

Table 17-1 provides a summary of the registers. The registers support 8-bit, 16-bit, and 32-bit writes. The offset is relative to the associated peripheral select. See the following sections for detailed descriptions of the registers. The base address for the control registers is FFFF FC00h. The address locations not listed are reserved.

Table 17-1. RTI Registers

Offset	Acronym	Register Description	Section
00h	RTIGCTRL	RTI Global Control Register	Section 17.3.1
04h	RTITBCTRL	RTI Timebase Control Register	Section 17.3.2
08h	RTICAPCTRL	RTI Capture Control Register	Section 17.3.3
0Ch	RTICOMPCTRL	RTI Compare Control Register	Section 17.3.4
10h	RTIFRC0	RTI Free Running Counter 0 Register	Section 17.3.5
14h	RTIUC0	RTI Up Counter 0 Register	Section 17.3.6
18h	RTICPUC0	RTI Compare Up Counter 0 Register	Section 17.3.7
20h	RTICAFRC0	RTI Capture Free Running Counter 0 Register	Section 17.3.8
24h	RTICAUC0	RTI Capture Up Counter 0 Register	Section 17.3.9
30h	RTIFRC1	RTI Free Running Counter 1 Register	Section 17.3.10
34h	RTIUC1	RTI Up Counter 1 Register	Section 17.3.11
38h	RTICPUC1	RTI Compare Up Counter 1 Register	Section 17.3.12
40h	RTICAFRC1	RTI Capture Free Running Counter 1 Register	Section 17.3.13
44h	RTICAUC1	RTI Capture Up Counter 1 Register	Section 17.3.14
50h	RTICOMP0	RTI Compare 0 Register	Section 17.3.15
54h	RTIUDCP0	RTI Update Compare 0 Register	Section 17.3.16
58h	RTICOMP1	RTI Compare 1 Register	Section 17.3.17
5Ch	RTIUDCP1	RTI Update Compare 1 Register	Section 17.3.18
60h	RTICOMP2	RTI Compare 2 Register	Section 17.3.19
64h	RTIUDCP2	RTI Update Compare 2 Register	Section 17.3.20
68h	RTICOMP3	RTI Compare 3 Register	Section 17.3.21
6Ch	RTIUDCP3	RTI Update Compare 3 Register	Section 17.3.22
70h	RTITBLCOMP	RTI Timebase Low Compare Register	Section 17.3.23
74h	RTITBHCOMP	RTI Timebase High Compare Register	Section 17.3.24
80h	RTISETINTENA	RTI Set Interrupt Enable Register	Section 17.3.25
84h	RTICLEARINTENA	RTI Clear Interrupt Enable Register	Section 17.3.26
88h	RTIINTFLAG	RTI Interrupt Flag Register	Section 17.3.27
90h	RTIDWDCTRL	Digital Watchdog Control Register	Section 17.3.28
94h	RTIDWDPRLD	Digital Watchdog Preload Register	Section 17.3.29
98h	RTIWDSTATUS	Watchdog Status Register	Section 17.3.30
9Ch	RTIWDKEY	RTI Watchdog Key Register	Section 17.3.31
A0h	RTIDWDCNTR	RTI Digital Watchdog Down Counter Register	Section 17.3.32
A4h	RTIWWDRXNCTRL	Digital Windowed Watchdog Reaction Control Register	Section 17.3.33
A8h	RTIWWDSIZECTRL	Digital Windowed Watchdog Window Size Control Register	Section 17.3.34
ACh	RTIINTCLRENABLE	RTI Compare Interrupt Clear Enable Register	Section 17.3.35
B0h	RTICOMP0CLR	RTI Compare 0 Clear Register	Section 17.3.36
B4h	RTICOMP1CLR	RTI Compare 1 Clear Register	Section 17.3.37
B8h	RTICOMP2CLR	RTI Compare 2 Clear Register	Section 17.3.38
BCh	RTICOMP3CLR	RTI Compare 3 Clear Register	Section 17.3.39

NOTE: Writes to Reserved registers may clear the pending RTI interrupt.

17.3.1 RTI Global Control Register (RTIGCTRL)

The global control register starts/stops the counters and selects the signal compared with the timebase control circuit. This register is shown in Figure 17-12 and described in Table 17-2.

Figure 17-12. RTI Global Control Register (RTIGCTRL) [offset = 00]

31 – 20	19 – 16
Reserved	NTUSEL
R-0	R/WP-0

15	14 – 2	1	0
COS	Reserved	CNT1EN	CNT0EN
R/WP-0	R-0	R/WP-0	R/WP-0

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privileged mode only; -*n* = value after reset

Table 17-2. RTI Global Control Register (RTIGCTRL) Field Descriptions

Bit	Field	Value	Description
31-20	Reserved	0	Reads return 0. Writes have no effect.
19-16	NTUSEL		Select NTU signal. These bits determine which NTU input signal is used as external timebase
		0h	NTU0
		5h	NTU1
		Ah	NTU2
		Fh	NTU3
		All other values	Tied to 0
15	COS		Continue on suspend. This bit determines if both counters are stopped when the device goes into halting debug mode or if they continue counting.
		0	Counters are stopped while in halting debug mode.
		1	Counters are running while in halting debug mode.
14-2	Reserved	0	Reads return 0. Writes have no effect.
1	CNT1EN		Counter 1 enable. This bit starts and stops counter block 1 (RTIUC1 and RTIFRC1).
		0	Counter block 1 is stopped.
		1	Counter block 1 is running.
0	CNT0EN		Counter 0 enable. This bit starts and stops counter block 0 (RTIUC0 and RTIFRC0).
		0	Counter block 0 is stopped.
		1	Counter block 0 is running.

NOTE: If the application uses the timebase circuit for synchronization between the communications controller and the operating system and the device enters halting debug mode, the synchronization may be lost depending on the COS setting in the RTI module and the halting debug mode behavior of the communications controller.

17.3.2 RTI Timebase Control Register (RTITBCTRL)

The timebase control register selects if the free running counter 0 is incremented by RTICLK or NTU. This register is shown in Figure 17-13 and described in Table 17-3.

Figure 17-13. RTI Timebase Control Register (RTITBCTRL) [offset = 04h]

31 … 8
Reserved
R-0

7 … 2	1	0
Reserved	INC	TBEXT
R-0	R/WP-0	R/WP-0

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privileged mode only; -*n* = value after reset

Table 17-3. RTI Timebase Control Register (RTITBCTRL) Field Descriptions

Bit	Field	Value	Description
31-2	Reserved	0	Reads return 0. Writes have no effect.
1	INC		Increment free running counter 0. This bit determines whether the free running counter 0 (RTIFRC0) is automatically incremented if a failing clock on the NTU signal is detected.
		0	RTIFRC0 will not be incremented on a failing external clock.
		1	RTIFRC0 will be incremented on a failing external clock.
0	TBEXT		Timebase external. This bit selects whether the free running counter 0 (RTIFRC0) is clocked by the internal up counter 0 (RTIUC0) or from the external signal NTU. Setting the TBEXT bit from 0 to 1 will not increment RTIFRC0, since RTIUC0 is reset. When the timebase supervisor circuit detects a missing clock edge, then the TBEXT bit is reset. Only the software can select whether the external signal should be used.
		0	RTIUC0 clocks RTIFRC0.
		1	NTU clocks RTIFRC0.

17.3.3 RTI Capture Control Register (RTICAPCTRL)

The capture control register controls the capture source for the counters. This register is shown in Figure 17-14 and described in Table 17-4.

Figure 17-14. RTI Capture Control Register (RTICAPCTRL) [offset = 08h]

31 … 8
Reserved
R-0

7 … 2	1	0
Reserved	CAPCNTR1	CAPCNTR0
R-0	R/WP-0	R/WP-0

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privileged mode only; -*n* = value after reset

Table 17-4. RTI Capture Control Register (RTICAPCTRL) Field Descriptions

Bit	Field	Value	Description
31-2	Reserved	0	Reads return 0. Writes have no effect.
1	CAPCNTR1		Capture counter 1. This bit determines which external interrupt source triggers a capture event of RTIUC1 and RTIFRC1.
		0	Capture of RTIUC1/ RTIFRC1 is triggered by capture event source 0.
		1	Capture of RTIUC1/ RTIFRC1 is triggered by capture event source 1.
0	CAPCNTR0		Capture counter 0. This bit determines which external interrupt source triggers a capture event of RTIUC0 and RTIFRC0.
		0	Capture of RTIUC0/ RTIFRC0 is triggered by capture event source 0.
		1	Capture of RTIUC0/ RTIFRC0 is triggered by capture event source 1.

17.3.4 RTI Compare Control Register (RTICOMPCTRL)

The compare control register controls the source for the compare registers. This register is shown in Figure 17-15 and described in Table 17-5.

Figure 17-15. RTI Compare Control Register (RTICOMPCTRL) [offset = 0Ch]

31	16
Reserved	
R-0	

15	13	12	11	9	8
Reserved		COMPSEL3	Reserved		COMPSEL2
R-0		R/WP-0	R-0		R/WP-0

7	5	4	3	1	0
Reserved		COMPSEL1	Reserved		COMPSEL0
R-0		R/WP-0	R-0		R/WP-0

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privileged mode only; -*n* = value after reset

Table 17-5. RTI Compare Control Register (RTICOMPCTRL) Field Descriptions

Bit	Field	Value	Description
31-13	Reserved	0	Reads return 0. Writes have no effect.
12	COMPSEL3		Compare select 3. This bit determines the counter with which the compare value held in compare register 3 (RTICOMP3) is compared.
		0	Value will be compared with RTIFRC0.
		1	Value will be compared with RTIFRC1.
11-9	Reserved	0	Reads return 0. Writes have no effect.
8	COMPSEL2		Compare select 2. This bit determines the counter with which the compare value held in compare register 2 (RTICOMP2) is compared.
		0	Value will be compared with RTIFRC0.
		1	Value will be compared with RTIFRC1.
7-5	Reserved	0	Reads return 0. Writes have no effect.
4	COMPSEL1		Compare select 1. This bit determines the counter with which the compare value held in compare register 1 (RTICOMP1) is compared.
		0	Value will be compared with RTIFRC0.
		1	Value will be compared with RTIFRC1.
3-1	Reserved	0	Reads return 0. Writes have no effect.
0	COMPSEL0		Compare select 0. This bit determines the counter with which the compare value held in compare register 0 (RTICOMP0) is compared.
		0	Value will be compared with RTIFRC0.
		1	Value will be compared with RTIFRC1.

17.3.5 RTI Free Running Counter 0 Register (RTIFRC0)

The free running counter 0 register holds the current value of free running counter 0. This register is shown in Figure 17-16 and described in Table 17-6.

Figure 17-16. RTI Free Running Counter 0 Register (RTIFRC0) [offset = 10h]

31	16
FRC0	
R/WP-0	

15	0
FRC0	
R/WP-0	

LEGEND: R/W = Read/Write; WP = Write in privileged mode only; -*n* = value after reset

Table 17-6. RTI Free Running Counter 0 Register (RTIFRC0) Field Descriptions

Bit	Field	Value	Description
31-0	FRC0	0-FFFF FFFFh	Free running counter 0. This registers holds the current value of the free running counter 0. A read of this counter returns the current value of the counter. The counter can be preset by writing (in privileged mode only) to this register. The counter increments then from this written value upwards. **Note: If counters must be preset, they must be disabled in the RTIGCTRL register to ensure consistency between RTIUC0 and RTIFRC0.**

17.3.6 RTI Up Counter 0 Register (RTIUC0)

The up counter 0 register holds the current value of prescale counter. This register is shown in Figure 17-17 and described in Table 17-7.

Figure 17-17. RTI Up Counter 0 Register (RTIUC0) [offset = 14h]

31	16
UC0	
R/WP-0	

15	0
UC0	
R/WP-0	

LLEGEND: R/W = Read/Write; WP = Write in privileged mode only; -*n* = value after reset

Table 17-7. RTI Up Counter 0 Register (RTIUC0) Field Descriptions

Bit	Field	Value	Description
31-0	UC0	0-FFFF FFFFh	Up counter 0. This register holds the current value of the up counter 0 and prescales the RTI clock. It will be only updated by a previous read of free running counter 0 (RTIFRC0). This method of updating effectively gives a 64-bit read of both counters, without having the problem of a counter being updated between two consecutive reads on up counter 0 (RTIUC0) and free running counter 0 (RTIFRC0). A read of this counter returns the value of the counter at the time RTIFRC0 was read. A write to this counter presets it with a value. The counter then increments from this written value upwards. Note: If counters must be preset, they must be disabled in the RTIGCTRL register to ensure consistency between RTIUC0 and RTIFRC0. **Note: If the preset value is bigger than the compare value stored in register RTICPUC0, then it can take a long time until a compare matches, since RTIUC0 has to count up until it overflows.**

17.3.7 RTI Compare Up Counter 0 Register (RTICPUC0)

The compare up counter 0 register holds the value to be compared with prescale counter 0 (RTIUC0). This register is shown in Figure 17-18 and described in Table 17-8.

Figure 17-18. RTI Compare Up Counter 0 Register (RTICPUC0) [offset = 18h]

31	16
CPUC0	
R/WP-0	

15	0
CPUC0	
R/WP-0	

LEGEND: R/W = Read/Write; WP = Write in privileged mode only; -*n* = value after reset

Table 17-8. RTI Compare Up Counter 0 Register (RTICPUC0) Field Descriptions

Bit	Field	Value	Description
31-0	CPUC0	0-FFFF FFFFh	Compare up counter 0. This register holds the value that is compared with the up counter 0. When the compare shows a match, the free running counter 0 (RTIFRC0) is incremented. RTIUC0 is set to 0 when the counter value matches the RTICPUC0 value. The value set in this register prescales the RTI clock. If CPUC0 = 0, then f_{FRC0} = RTICLK/(2^{32}+1) (Setting CPUC0 equal to 0 is not recommended. Doing so will hold the Up Counter at 0 for 2 RTICLK cycles after it overflows from FFFF FFFFh to 0.) If CPUC0 ≠ 0, then f_{FRC0} = RTICLK/(RTICPUC0+1) A read of this register returns the current compare value. A write to this register: • If TBEXT = 0, the compare value is updated. • If TBEXT = 1, the compare value is unchanged.

17.3.8 RTI Capture Free Running Counter 0 Register (RTICAFRC0)

The capture free running counter 0 register holds the free running counter 0 on external events. This register is shown in Figure 17-19 and described in Table 17-9.

Figure 17-19. RTI Capture Free Running Counter 0 Register (RTICAFRC0) [offset = 20h]

31	16
CAFRC0	
R-0	

15	0
CAFRC0	
R-0	

LEGEND: R = Read only; -*n* = value after reset

Table 17-9. RTI Capture Free Running Counter 0 Register (RTICAFRC0) Field Descriptions

Bit	Field	Value	Description
31-0	CAFRC0	0-FFFF FFFFh	Capture free running counter 0. This register captures the current value of the free running counter 0 (RTIFRC0) when an event occurs, controlled by the external capture control block. A read of this register returns the value of RTIFRC0 on a capture event.

17.3.9 RTI Capture Up Counter 0 Register (RTICAUC0)

The capture up counter 0 register holds the current value of prescale counter 0 on external events. This register is shown in Figure 17-20 and described in Table 17-10.

Figure 17-20. RTI Capture Up Counter 0 Register (RTICAUC0) [offset = 24h]

31	16
CAUC0	
R-0	

15	0
CAUC0	
R-0	

LEGEND: R = Read only; -*n* = value after reset

Table 17-10. RTI Capture Up Counter 0 Register (RTICAUC0) Field Descriptions

Bit	Field	Value	Description
31-0	CAUC0	0-FFFF FFFFh	Capture up counter 0. This register captures the current value of the up counter 0 (RTIUC0) when an event occurs, controlled by the external capture control block. **Note: The read sequence must be the same as with RTIUC0 and RTIFRC0. Therefore, the RTICAFRC0 register must be read before the RTICAUC0 register is read. This sequence ensures that the value of the RTICAUC0 register is the corresponding value to the RTICAFRC0 register, even if another capture event happens in between the two reads.** A read of this register returns the value of RTIUC0 on a capture event.

17.3.10 RTI Free Running Counter 1 Register (RTIFRC1)

The free running counter 1 register holds the current value of the free running counter 1. This register is shown in Figure 17-21 and described in Table 17-11.

Figure 17-21. RTI Free Running Counter 1 Register (RTIFRC1) [offset = 30h]

31	16
FRC1	
R/WP-0	

15	0
FRC1	
R/WP-0	

LEGEND: R/W = Read/Write; WP = Write in privileged mode only; -*n* = value after reset

Table 17-11. RTI Free Running Counter 1 Register (RTIFRC1) Field Descriptions

Bit	Field	Value	Description
31-0	FRC1	0-FFFF FFFFh	Free running counter 1. This register holds the current value of the free running counter 1 and will be updated continuously. A read of this register returns the current value of the counter. A write to this register presets the counter. The counter increments then from this written value upwards. **Note: If counters must be preset, they must be disabled in the RTIGCTRL register to ensure consistency between RTIUC1 and RTIFRC1.**

17.3.11 RTI Up Counter 1 Register (RTIUC1)

The up counter 1 register holds the current value of the prescale counter 1. This register is shown in Figure 17-22 and described in Table 17-12.

Figure 17-22. RTI Up Counter 1 Register (RTIUC1) [offset = 34h]

31 … 16
UC1
R/WP-0

15 … 0
UC1
R/WP-0

LEGEND: R/W = Read/Write; WP = Write in privileged mode only; -*n* = value after reset

Table 17-12. RTI Up Counter 1 Register (RTIUC1) Field Descriptions

Bit	Field	Value	Description
31-0	UC1	0-FFFF FFFFh	Up counter 1. This register holds the current value of the up counter 1 and prescales the RTI clock. It will be only updated by a previous read of free running counter 1 (RTIFRC1). This method of updating effectively gives a 64-bit read of both counters, without having the problem of a counter being updated between two consecutive reads on RTIUC1 and RTIFRC1. A read of this register will return the value of the counter when the RTIFRC1 was read. A write to this register presets the counter. The counter then increments from this written value upwards. **Note: If counters must be preset, they must be disabled in the RTIGCTRL register to ensure consistency between RTIUC1 and RTIFRC1.** **Note: If the preset value is bigger than the compare value stored in register RTICPUC1, then it can take a long time until a compare matches, since RTIUC1 has to count up until it overflows.**

17.3.12 RTI Compare Up Counter 1 Register (RTICPUC1)

The compare up counter 1 register holds the value compared with prescale counter 1. This register is shown in Figure 17-23 and described in Table 17-13.

Figure 17-23. RTI Compare Up Counter 1 Register (RTICPUC1) [offset = 38h]

31	16
CPUC1	
R/WP-0	

15	0
CPUC1	
R/WP-0	

LEGEND: R/W = Read/Write; WP = Write in privileged mode only; -*n* = value after reset

Table 17-13. RTI Compare Up Counter 1 Register (RTICPUC1) Field Descriptions

Bit	Field	Value	Description
31-0	CPUC1	0-FFFF FFFFh	Compare up counter 1. This register holds the compare value, which is compared with the up counter 1. When the compare matches, the free running counter 1 (RTIFRC1) is incremented. The up counter is cleared to 0 when the counter value matches the CPUC1 value. The value set in this prescales the RTI clock according to the following formula: If CPUC1 = 0, then f_{FRC1} = RTICLK/(2^{32}+1) (Setting CPUC1 equal to 0 is not recommended. Doing so will hold the Up Counter at 0 for 2 RTICLK cycles after it overflows from FFFF FFFFh to 0.) If CPUC1 ≠ 0, then f_{FRC1} = RTICLK/(RTICPUC1+1) A read of this register returns the current compare value. A write to this register updates the compare value.

17.3.13 RTI Capture Free Running Counter 1 Register (RTICAFRC1)

The capture free running counter 1 register holds the current value of free running counter 1 on external events. This register is shown in Figure 17-24 and described in Table 17-14.

Figure 17-24. RTI Capture Free Running Counter 1 Register (RTICAFRC1) [offset = 40h]

31	16
CAFRC1	
R-0	

15	0
CAFRC1	
R-0	

LEGEND: R = Read only; -*n* = value after reset

Table 17-14. RTI Capture Free Running Counter 1 Register (RTICAFRC1) Field Descriptions

Bit	Field	Value	Description
31-0	CAFRC1	0-FFFF FFFFh	Capture free running counter 1. This register captures the current value of the free running counter 1 (RTIFRC1) when an event occurs, controlled by the external capture control block. A read of this register returns the value of RTIFRC1 on a capture event.

17.3.14 RTI Capture Up Counter 1 Register (RTICAUC1)

The capture up counter 1 register holds the current value of prescale counter 1 on external events. This register is shown in Figure 17-25 and described in Table 17-15.

Figure 17-25. RTI Capture Up Counter 1 Register (RTICAUC1) [offset = 44h]

31	16
CAUC1	
R-0	

15	0
CAUC1	
R-0	

LEGEND: R = Read only; -*n* = value after reset

Table 17-15. RTI Capture Up Counter 1 Register (RTICAUC1) Field Descriptions

Bit	Field	Value	Description
31-0	CAUC1	0-FFFF FFFFh	Capture up counter 1. This register captures the current value of the up counter 1 (RTIUC1) when an event occurs, controlled by the external capture control block. **Note: The RTICAFRC1 register must be read before the RTICAUC1 register is read. This sequence ensures that the value of the RTICAUC1 register is the corresponding value to the RTICAFRC1 register, even if another capture event happens in between the two reads.** A read of this register returns the value of RTIUC1 on a capture event.

17.3.15 RTI Compare 0 Register (RTICOMP0)

The compare 0 register holds the value to be compared with the counters. This register is shown in Figure 17-26 and described in Table 17-16.

Figure 17-26. RTI Compare 0 Register (RTICOMP0) [offset = 50h]

31	16
COMP0	
R/WP-0	

15	0
COMP0	
R/WP-0	

LEGEND: R/W = Read/Write; WP = Write in privileged mode only; -*n* = value after reset

Table 17-16. RTI Compare 0 Register (RTICOMP0) Field Descriptions

Bit	Field	Value	Description
31-0	COMP0	0-FFFF FFFFh	Compare 0. This registers holds a value that is compared with the counter selected in the compare control logic. If RTIFRC0 or RTIFRC1, depending on the counter selected, matches the compare value, an interrupt is flagged. With this register it is also possible to initiate a DMA request. A read of this register will return the current compare value. A write to this register (in privileged mode only) will update the compare register with a new compare value.

17.3.16 RTI Update Compare 0 Register (RTIUDCP0)

The update compare 0 register holds the value to be added to the compare register 0 value on a compare match. This register is shown in Figure 17-27 and described in Table 17-17.

Figure 17-27. RTI Update Compare 0 Register (RTIUDCP0) [offset = 54h]

31	16
UDCP0	
R/WP-0	

15	0
UDCP0	
R/WP-0	

LEGEND: R/W = Read/Write; WP = Write in privileged mode only; -*n* = value after reset

Table 17-17. RTI Update Compare 0 Register (RTIUDCP0) Field Descriptions

Bit	Field	Value	Description
31-0	UDCP0	0-FFFF FFFFh	Update compare 0. This register holds a value that is added to the value in the compare 0 (RTICOMP0) register each time a compare matches. This function allows periodic interrupts to be generated without software intervention. A read of this register will return the value to be added to the RTICOMP0 register on the next compare match. A write to this register will provide a new update value.

17.3.17 RTI Compare 1 Register (RTICOMP1)

The compare 1 register holds the value to be compared to the counters. This register is shown in Figure 17-28 and described in Table 17-18.

Figure 17-28. RTI Compare 1 Register (RTICOMP1) [offset = 58h]

31 … 16
COMP1
R/WP-0

15 … 0
COMP1
R/WP-0

LEGEND: R/W = Read/Write; WP = Write in privileged mode only; -*n* = value after reset

Table 17-18. RTI Compare 1 Register (RTICOMP1) Field Descriptions

Bit	Field	Value	Description
31-0	COMP1	0-FFFF FFFFh	Compare 1. This register holds a value that is compared with the counter selected in the compare control logic. If RTIFRC0 or RTIFRC1, depending on the counter selected, matches this compare value, an interrupt is flagged. With this register, it is possible to initiate a DMA request. A read of this register will return the current compare value. A write to this register will update the compare register with a new compare value.

17.3.18 RTI Update Compare 1 Register (RTIUDCP1)

The update compare 1 register holds the value to be added to the compare register 1 value on a compare match. This register is shown in Figure 17-29 and described in Table 17-19.

Figure 17-29. RTI Update Compare 1 Register (RTIUDCP1) [offset = 5Ch]

31 … 16
UDCP1
R/WP-0

15 … 0
UDCP1
R/WP-0

LEGEND: R/W = Read/Write; WP = Write in privileged mode only; -*n* = value after reset

Table 17-19. RTI Update Compare 1 Register (RTIUDCP1) Field Descriptions

Bit	Field	Value	Description
31-0	UDCP1	0-FFFF FFFFh	Update compare 1. This register holds a value that is added to the value in the RTICOMP1 register each time a compare matches. This process allows periodic interrupts to be generated without software intervention. A read of this register will return the value to be added to the RTICOMP1 register on the next compare match. A write to this register will provide a new update value.

17.3.19 RTI Compare 2 Register (RTICOMP2)

The compare 2 register holds the value to be compared to the counters. This register is shown in Figure 17-30 and described in Table 17-20.

Figure 17-30. RTI Compare 2 Register (RTICOMP2) [offset = 60h]

31	16
COMP2	
R/WP-0	

15	0
COMP2	
R/WP-0	

LEGEND: R/W = Read/Write; WP = Write in privileged mode only; -*n* = value after reset

Table 17-20. RTI Compare 2 Register (RTICOMP2) Field Descriptions

Bit	Field	Value	Description
31-0	COMP2	0-FFFF FFFFh	Compare 2. This register holds a value that is compared with the counter selected in the compare control logic. If RTIFRC0 or RTIFRC1, depending on the counter selected, matches this compare value, an interrupt is flagged. With this register, it is possible to initiate a DMA request. A read of this register will return the current compare value. A write to this register (in privileged mode only) will provide a new compare value.

17.3.20 RTI Update Compare 2 Register (RTIUDCP2)

The update compare 2 register holds the value to be added to the compare register 2 value on a compare match. This register is shown in Figure 17-31 and described in Table 17-21.

Figure 17-31. RTI Update Compare 2 Register (RTIUDCP2) [offset = 64h]

31	16
UDCP2	
R/WP-0	

15	0
UDCP2	
R/WP-0	

LEGEND: R/W = Read/Write; WP = Write in privileged mode only; -*n* = value after reset

Table 17-21. RTI Update Compare 2 Register (RTIUDCP2) Field Descriptions

Bit	Field	Value	Description
31-0	UDCP2	0-FFFF FFFFh	Update compare 2. This register holds a value that is added to the value in the RTICOMP2 register each time a compare matches. This process makes it possible to generate periodic interrupts without software intervention. A read of this register will return the value to be added to the RTICOMP2 register on the next compare match. A write to this register will provide a new update value.

17.3.21 RTI Compare 3 Register (RTICOMP3)

The compare 3 register holds the value to be compared to the counters. This register is shown in Figure 17-32 and described in Table 17-22.

Figure 17-32. RTI Compare 3 Register (RTICOMP3) [offset = 68h]

31	16
COMP3	
R/WP-0	

15	0
COMP3	
R/WP-0	

LEGEND: R/W = Read/Write; WP = Write in privileged mode only; -*n* = value after reset

Table 17-22. RTI Compare 3 Register (RTICOMP3) Field Descriptions

Bit	Field	Value	Description
31-0	COMP3	0-FFFF FFFFh	Compare 3. This register holds a value that is compared with the counter selected in the compare control logic. If RTIFRC0 or RTIFRC1, depending on the counter selected, matches this compare value, an interrupt is flagged. With this register, it is possible to initiate a DMA request. A read of this register will return the current compare value. A write to this register will provide a new compare value.

17.3.22 RTI Update Compare 3 Register (RTIUDCP3)

The update compare 3 register holds the value to be added to the compare register 3 value on a compare match. This register is shown in Figure 17-33 and described in Table 17-23.

Figure 17-33. RTI Update Compare 3 Register (RTIUDCP3) [offset = 6Ch]

31	16
UDCP3	
R/WP-0	

15	0
UDCP3	
R/WP-0	

LEGEND: R/W = Read/Write; WP = Write in privileged mode only; -*n* = value after reset

Table 17-23. RTI Update Compare 3 Register (RTIUDCP3) Field Descriptions

Bit	Field	Value	Description
31-0	UDCP3	0-FFFF FFFFh	Update compare 3. This register holds a value that is added to the value in the RTICOMP3 register each time a compare matches. This process makes it possible to generate periodic interrupts without software intervention. A read of this register will return the value to be added to the RTICOMP3 register on the next compare match. A write to this register will provide a new update value.

17.3.23 RTI Timebase Low Compare Register (RTITBLCOMP)

The timebase low compare register holds the value to activate the edge detection circuit. This register is shown in Figure 17-34 and described in Table 17-24.

Figure 17-34. RTI Timebase Low Compare Register (RTITBLCOMP) [offset = 70h]

31	16
TBLCOMP	
R/WP-0	

15	0
TBLCOMP	
R/WP-0	

LEGEND: R/W = Read/Write; WP = Write in privileged mode only; -*n* = value after reset

Table 17-24. RTI Timebase Low Compare Register (RTITBLCOMP) Field Descriptions

Bit	Field	Value	Description
31-0	TBLCOMP	0-FFFF FFFFh	Timebase low compare value. This value determines when the edge detection circuit starts monitoring the NTU signal. It will be compared with RTIUC0. A read of this register will return the current compare value. A write to this register has the following effects: If TBEXT = 0: The compare value is updated. If TBEXT = 1: The compare value is not changed.

17.3.24 RTI Timebase High Compare Register (RTITBHCOMP)

The timebase high compare register holds the value to deactivate the edge detection circuit. This register is shown in Figure 17-35 and described in Table 17-25.

Figure 17-35. RTI Timebase High Compare Register (RTITBHCOMP) [offset = 74h]

31	16
TBHCOMP	
R/WP-0	

15	0
TBHCOMP	
R/WP-0	

LEGEND: R/W = Read/Write; WP = Write in privileged mode only; -*n* = value after reset

Table 17-25. RTI Timebase High Compare Register (RTITBHCOMP) Field Descriptions

Bit	Field	Value	Description
31-0	TBHCOMP	0-FFFF FFFFh	Timebase high compare value. This value determines when the edge detection circuit will stop monitoring the NTU signal. It will be compared with RTIUC0. RTITBHCOMP must be less than RTICPUC0 because RTIUC0 will be reset when RTICPUC0 is reached. Example: The NTU edge detection circuit should be active ± 10 RTICLK cycles around RTICPUC0. • RTICPUC0 = 0050h • RTITBLCOMP = 0046h • RTITBHCOMP = 0009h A read of this register will return the current compare value. A write to this register has the following effects: If TBEXT = 0: The compare value is updated. If TBEXT = 1: The compare value is not changed.

17.3.25 RTI Set Interrupt Enable Register (RTISETINTENA)

This register prevents the necessity of a read-modify-write operation if a particular interrupt should be enabled. This register is shown in Figure 17-36 and described in Table 17-26.

Figure 17-36. RTI Set Interrupt Control Register (RTISETINTENA) [offset = 80h]

31 24
Reserved
R-0

23 19	18	17	16
Reserved	SETOVL1INT	SETOVL0INT	SETTBINT
R-0	R/WP-0	R/WP-0	R/WP-0

15 12	11	10	9	8
Reserved	SETDMA3	SETDMA2	SETDMA1	SETDMA0
R-0	R/WP-0	R/WP-0	R/WP-0	R/WP-0

7 4	3	2	1	0
Reserved	SETINT3	SETINT2	SETINT1	SETINT0
R-0	R/WP-0	R/WP-0	R/WP-0	R/WP-0

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privileged mode only; -*n* = value after reset

Table 17-26. RTI Set Interrupt Control Register (RTISETINTENA) Field Descriptions

Bit	Field	Value	Description
31-19	Reserved	0	Reads return 0. Writes have no effect.
18	SETOVL1INT		Set free running counter 1 overflow interrupt.
		0	*Read:* Interrupt is disabled. *Write:* Corresponding bit is unchanged.
		1	*Read or Write:* Interrupt is enabled.
17	SETOVL0INT		Set free running counter 0 overflow interrupt.
		0	*Read:* Interrupt is disabled. *Write:* Corresponding bit is unchanged.
		1	*Read or Write:* Interrupt is enabled.
16	SETTBINT		Set timebase interrupt.
		0	*Read:* Interrupt is disabled. *Write:* Corresponding bit is unchanged.
		1	*Read or Write:* Interrupt is enabled.
15-12	Reserved	0	Reads return 0. Writes have no effect.
11	SETDMA3		Set compare DMA request 3.
		0	*Read:* DMA request is disabled. *Write:* DMA request is unchanged.
		1	*Read or Write:* DMA request is enabled.
10	SETDMA2		Set compare DMA request 2.
		0	*Read:* DMA request is disabled. *Write:* DMA request is unchanged.
		1	*Read or Write:* DMA request is enabled.
9	SETDMA1		Set compare DMA request 1.
		0	*Read:* DMA request is disabled. *Write:* DMA request is unchanged.
		1	*Read or Write:* DMA request is enabled.

Table 17-26. RTI Set Interrupt Control Register (RTISETINTENA) Field Descriptions (continued)

Bit	Field	Value	Description
8	SETDMA0		Set compare DMA request 0.
		0	*Read:* DMA request is disabled. *Write:* DMA request is unchanged.
		1	*Read or Write:* DMA request is enabled.
7-4	Reserved	0	Reads return 0. Writes have no effect.
3	SETINT3		Set compare interrupt 3.
		0	*Read:* Interrupt is disabled. *Write:* Corresponding bit is unchanged.
		1	*Read or Write:* Interrupt is enabled.
2	SETINT2		Set compare interrupt 2.
		0	*Read:* Interrupt is disabled. *Write:* Corresponding bit is unchanged.
		1	*Read or Write:* Interrupt is enabled.
1	SETINT1		Set compare interrupt 1.
		0	*Read:* Interrupt is disabled. *Write:* Corresponding bit is unchanged.
		1	*Read or Write:* Interrupt is enabled.
0	SETINT0		Set compare interrupt 0.
		0	*Read:* Interrupt is disabled. *Write:* Corresponding bit is unchanged.
		1	*Read or Write:* Interrupt is enabled.

17.3.26 RTI Clear Interrupt Enable Register (RTICLEARINTENA)

This register prevents the necessity of a read-modify-write operation if a particular interrupt should be disabled. This register is shown in Figure 17-37 and described in Table 17-27.

Figure 17-37. RTI Clear Interrupt Control Register (RTICLEARINTENA) [offset = 84h]

31-24
Reserved
R-0

23-19	18	17	16
Reserved	CLEAROVL1INT	CLEAROVL0INT	CLEARTBINT
R-0	R/WP-0	R/WP-0	R/WP-0

15-12	11	10	9	8
Reserved	CLEARDMA3	CLEARDMA2	CLEARDMA1	CLEARDMA0
R-0	R/WP-0	R/WP-0	R/WP-0	R/WP-0

7-4	3	2	1	0
Reserved	CLEARINT3	CLEARINT2	CLEARINT1	CLEARINT0
R-0	R/WP-0	R/WP-0	R/WP-0	R/WP-0

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privileged mode only; -*n* = value after reset

Table 17-27. RTI Clear Interrupt Control Register (RTICLEARINTENA) Field Descriptions

Bit	Field	Value	Description
31-19	Reserved	0	Reads return 0. Writes have no effect.
18	CLEAROVL1INT		Clear free running counter 1 overflow interrupt.
		0	*Read:* Interrupt is disabled. *Write:* Corresponding bit is unchanged.
		1	*Read:* Interrupt is enabled. *Write:* Interrupt is disabled.
17	CLEAROVL0INT		Clear free running counter 0 overflow interrupt.
		0	*Read:* Interrupt is disabled. *Write:* Corresponding bit is unchanged.
		1	*Read:* Interrupt is enabled. *Write:* Interrupt is disabled.
16	CLEARTBINT		Clear timebase interrupt.
		0	*Read:* Interrupt is disabled. *Write:* Corresponding bit is unchanged.
		1	*Read:* Interrupt is enabled. *Write:* Interrupt is disabled.
15-12	Reserved	0	Reads return 0. Writes have no effect.
11	CLEARDMA3		Clear compare DMA request 3.
		0	*Read:* DMA request is disabled. *Write:* Corresponding bit is unchanged.
		1	*Read:* DMA request is enabled. *Write:* DMA request is disabled.
10	CLEARDMA2		Clear compare DMA request 2.
		0	*Read:* DMA request is disabled. *Write:* Corresponding bit is unchanged.
		1	*Read:* DMA request is enabled. *Write:* DMA request is disabled.

Table 17-27. RTI Clear Interrupt Control Register (RTICLEARINTENA) Field Descriptions (continued)

Bit	Field	Value	Description
9	CLEARDMA1		Clear compare DMA request 1.
		0	*Read:* DMA request is disabled. *Write:* Corresponding bit is unchanged.
		1	*Read:* DMA request is enabled. *Write:* DMA request is disabled.
8	CLEARDMA0		Clear compare DMA request 0.
		0	*Read:* DMA request is disabled. *Write:* Corresponding bit is unchanged.
		1	*Read:* DMA request is enabled. *Write:* DMA request is disabled.
7-4	Reserved	0	Reads return 0. Writes have no effect.
3	CLEARINT3		Clear compare interrupt 3.
		0	*Read:* Interrupt is disabled. *Write:* Corresponding bit is unchanged.
		1	*Read:* Interrupt is enabled. *Write:* Interrupt is disabled.
2	CLEARINT2		Clear compare interrupt 2.
		0	*Read:* Interrupt is disabled. *Write:* Corresponding bit is unchanged.
		1	*Read:* Interrupt is enabled. *Write:* Interrupt is disabled.
1	CLEARINT1		Clear compare interrupt 1.
		0	*Read:* Interrupt is disabled. *Write:* Corresponding bit is unchanged.
		1	*Read:* Interrupt is enabled. *Write:* Interrupt is disabled.
0	CLEARINT0		Clear compare interrupt 0.
		0	*Read:* Interrupt is disabled. *Write:* Corresponding bit is unchanged.
		1	*Read:* Interrupt is enabled. *Write:* Interrupt is disabled.

17.3.27 RTI Interrupt Flag Register (RTIINTFLAG)

The corresponding flags are set at every compare match of the RTIFRCx and RTICOMPx values, whether the interrupt is enabled or not. This register is shown in Figure 17-38 and described in Table 17-28.

Figure 17-38. RTI Interrupt Flag Register (RTIINTFLAG) [offset = 88h]

31 – 19	18	17	16
Reserved	OVL1INT	OVL0INT	TBINT
R-0	R/W1CP-0	R/W1CP-0	R/W1C P-0

15 – 4	3	2	1	0
Reserved	INT3	INT2	INT1	INT0
R-0	R/W1C P-0	R/W1C P-0	R/W1C P-0	R/W1C P-0

LEGEND: R/W = Read/Write; R = Read only; W1CP = Write 1 to clear in privilege mode only; -*n* = value after reset

Table 17-28. RTI Interrupt Flag Register (RTIINTFLAG) Field Descriptions

Bit	Field	Value	Description
31-19	Reserved	0	Reads return 0. Writes have no effect.
18	OVL1INT		Free running counter 1 overflow interrupt flag. This bit determines if an interrupt is pending.
		0	*Read:* No interrupt is pending. *Write:* Bit is unchanged.
		1	*Read:* Interrupt is pending. *Write:* Bit is cleared to 0.
17	OVL0INT		Free running counter 0 overflow interrupt flag. This bit determines if an interrupt is pending.
		0	*Read:* No interrupt is pending. *Write:* Bit is unchanged.
		1	*Read:* Interrupt is pending. *Write:* Bit is cleared to 0.
16	TBINT		Timebase interrupt flag. This flag is set when the TBEXT bit is cleared by detection of a missing external clock edge. It will not be set by clearing TBEXT by software. It determines if an interrupt is pending.
		0	*Read:* No interrupt is pending. *Write:* Bit is unchanged.
		1	*Read:* Interrupt is pending. *Write:* Bit is cleared to 0.
15-4	Reserved	0	Reads return 0. Writes have no effect.
3	INT3		Interrupt flag 3. These bits determine if an interrupt due to a Compare 3 match is pending.
		0	*Read:* No interrupt is pending. *Write:* Bit is unchanged.
		1	*Read:* Interrupt is pending. *Write:* Bit is cleared to 0.
2	INT2		Interrupt flag 2. These bits determine if an interrupt due to a Compare 2 match is pending.
		0	*Read:* No interrupt is pending. *Write:* Bit is unchanged.
		1	*Read:* Interrupt is pending. *Write:* Bit is cleared to 0.
1	INT1		Interrupt flag 1. These bits determine if an interrupt due to a Compare 1 match is pending.
		0	*Read:* No interrupt is pending. *Write:* Bit is unchanged.
		1	*Read:* Interrupt is pending. *Write:* Bit is cleared to 0.

Table 17-28. RTI Interrupt Flag Register (RTIINTFLAG) Field Descriptions (continued)

Bit	Field	Value	Description
0	INT0		Interrupt flag 0. These bits determine if an interrupt due to a Compare 0 match is pending.
		0	*Read:* No interrupt is pending. *Write:* Bit is unchanged.
		1	*Read:* Interrupt is pending. *Write:* Bit is cleared to 0.

17.3.28 Digital Watchdog Control Register (RTIDWDCTRL)

The software has to write to the DWDCTRL field in order to enable the DWD, as described below. Once enabled, the watchdog can only be disabled by a system reset. The application cannot disable the watchdog. However should the RTICLK source be changed to a source that is unimplemented it will have the same effect as disabling the watchdog. This register is shown in Figure 17-38 and described in Table 17-28.

Figure 17-39. Digital Watchdog Control Register (RTIDWDCTRL) [offset = 90h]

31	16
DWDCTRL	
R/WP-5312h	

15	0
DWDCTRL	
R/WP-ACEDh	

LEGEND: R/W = Read/Write; WP = Write in privileged mode only; -*n* = value after reset

Table 17-29. Digital Watchdog Control Register (RTIDWDCTRL) Field Descriptions

Bit	Field	Value	Description
31-0	DWDCTRL		Digital Watchdog Control.
		5312 ACEDh	*Read:* DWD counter is disabled. *Write:* State of DWD counter is unchanged (stays enabled or disabled).
		A985 59DAh	*Read:* DWD counter is enabled. *Write:* DWD counter is enabled.
		All other values	*Read:* DWD counter state is unchanged (enabled or disabled). *Write:* State of DWD counter is unchanged (stays enabled or disabled). **Note:** Once the enable value is written, all other future writes are blocked. In other words, once DWD is enabled, it can only be disabled by system reset or power on reset. However should the RTICLK source be changed to a source that is unimplemented it will have the same effect as disabling the watchdog.

17.3.29 Digital Watchdog Preload Register (RTIDWDPRLD)

This register sets the expiration time of the DWD. This register is shown in Figure 17-38 and described in Table 17-28.

Figure 17-40. Digital Watchdog Preload Register (RTIDWDPRLD) [offset = 94h]

31 16
Reserved
R-0

15 12	11 0
Reserved	DWDPRLD
R-0	R/WP-FFFh

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privileged mode only; -*n* = value after reset

Table 17-30. Digital Watchdog Preload Register (RTIDWDPRLD) Field Descriptions

Bit	Field	Value	Description
31-12	Reserved	0	Reads return 0 and writes have no effect.
11-0	DWDPRLD	0-FFFh	Digital Watchdog Preload Value. *Read:* The current preload value *Write:* Set the preload value. The DWD preload register can be configured only when the DWD is disabled. Therefore, the application can only configure the DWD preload register before it enables the DWD down counter. The expiration time of the DWD Down Counter can be determined with following equation: texp = (DWDPRLD+1) x 2^{13} / RTICLK1 where: DWDPRLD = 0...4095

17.3.30 Watchdog Status Register (RTIWDSTATUS)

This register records the status of the DWD. The values of the following status bits will not be affected by a soft reset. These bits are cleared by a power-on reset, or by a write of 1. These bits can be used for debug purposes. This register is shown in Figure 17-38 and described in Table 17-28.

Figure 17-41. Watchdog Status Register (RTIWDSTATUS) [offset = 98h]

31	8
Reserved	
R-0	

7	6	5	4	3	2	1	0
Reserved		DWWD ST	END TIME VIOL	START TIME VIOL	KEY ST	DWD ST	Reserved
R-0		R/W1CP-x	R/W1CP-x	R/W1CP-x	R/W1CP-x	R/W1CP-x	R-0

LEGEND: R/W = Read/Write; R = Read only; W1CP = Write 1 to clear in privilege mode only; -*n* = value after reset

Table 17-31. Watchdog Status Register (RTIWDSTATUS) Field Descriptions

Bit	Field	Value	Description
31-6	Reserved	0	Reads return 0. Writes have no effect.
5	DWWD ST	0	Windowed Watchdog Status *Read:* No time-window violation has occurred. *Write:* Leaves the current value unchanged.
		1	*Read:* Time-window violation has occurred. The watchdog has generated either a system reset or a non-maskable interrupt to the CPU in this case. *Write:* Bit is cleared to 0. This will also clear all other status flags in the RTIWDSTATUS register. Clearing of the status flags will deassert the non-maskable interrupt generated due to violation of the DWWD.
4	END TIME VIOL	0	Windowed Watchdog End Time Violation Status. This bit indicates whether the Watchdog counter expired. *Read:* No end-time window violation has occurred. *Write:* Leaves the current value unchanged.
		1	*Read*: End-time defined by the windowed watchdog configuration has been violated. *Write:* Bit is cleared to 0.
3	START TIME VIOL	0	Windowed Watchdog Start Time Violation Status. This bit indicates whether the key is written before the watchdog window opened up. *Read:* No start-time window violation has occurred. *Write:* Leaves the current value unchanged.
		1	*Read:* Start-time defined by the windowed watchdog configuration has been violated. *Write:* Bit is cleared to 0.
2	KEY ST	0	Watchdog key status. This bit indicates a reset or NMI generated by a wrong key or key sequence written to the RTIWDKEY register. *Read:* No wrong key or key-sequence written. *Write:* Bit is unchanged.
		1	*Read:* Wrong key or key-sequence written to RTIWDKEY register. *Write:* Bit is cleared to 0.
1	DWD ST	0	DWD status. This bit is equivalent to bit END TIME VIOL. *Read:* No reset or NMI was generated. *Write:* Bit is unchanged.
		1	*Read:* Reset or NMI was generated. *Write:* Bit is cleared to 0.
0	Reserved	0	Reads return 0. Writes have no effect.

17.3.31 RTI Watchdog Key Register (RTIWDKEY)

This register must be written with the correct written key values to serve the watchdog. This register is shown in Figure 17-42 and described in Table 17-32.

NOTE: It has to be taken into account that the write to the RTIWDKEY register takes 3 VCLK cycles.

Figure 17-42. RTI Watchdog Key Register (RTIDWDKEY) [offset = 9Ch]

31	16
Reserved	
R-0	

15	0
WDKEY	
R/WP-A35Ch	

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privileged mode only; -*n* = value after reset

Table 17-32. RTI Watchdog Key Register (RTIDWDKEY) Field Descriptions

Bit	Field	Value	Description
31-16	Reserved	0	Reads return 0 and writes have no effect.
15-0	WDKEY	0-FFFFh	Watchdog key. These bits provide the key sequence location. Reads returns the current WDKEY value. A write of E51Ah followed by A35Ch in two separate write operations defines the key sequence and reloads the DWD. Writing any other value causes a reset or NMI, as shown in Table 17-33. Writing any other value will cause the WDKEY to reset to A35Ch.

Table 17-33. Example of a WDKEY Sequence

Step	Value Written to WDKEY	Result
1	A35Ch	No action
2	A35Ch	No action
3	E51Ah	WDKEY is enabled for reset or NMI by next A35Ch.
4	E51Ah	WDKEY is enabled for reset or NMI by next A35Ch.
5	E51Ah	WDKEY is enabled for reset or NMI by next A35Ch.
6	A35Ch	Watchdog is reset.
7	A35Ch	No action
8	E51Ah	WDKEY is enabled for reset or NMI by next A35Ch.
9	A35Ch	Watchdog is reset.
10	E51Ah	WDKEY is enabled for reset or NMI by next A35Ch.
11	2345h	System reset or NMI; incorrect value written to WDKEY.

17.3.32 RTI Digital Watchdog Down Counter (RTIDWDCNTR)

This register provides the current value of the DWD down counter. This register is shown in Figure 17-43 and described in Table 17-34.

Figure 17-43. RTI Watchdog Down Counter Register (RTIDWDCNTR) [offset = A0h]

31–25	24–16
Reserved	DWDCNTR
R-0	R-1FFh

15–0
DWDCNTR
R-FFFFh

LEGEND: R = Read only; -*n* = value after reset

Table 17-34. RTI Watchdog Down Counter Register (RTIDWDCNTR) Field Descriptions

Bit	Field	Value	Description
31-25	Reserved	0	Reads return 0 and writes have no effect.
24-0	DWDCNTR	0-1FF FFFFh	DWD down counter. Reads return the current counter value.

17.3.33 Digital Windowed Watchdog Reaction Control (RTIWWDRXNCTRL)

This register selects the DWWD reaction if the watchdog is serviced outside the time window. This register is shown in Figure 17-44 and described in Table 17-35.

Figure 17-44. Digital Windowed Watchdog Reaction Control (RTIWWDRXNCTRL) [offset = A4h]

31–16
Reserved
R-0

15–4	3–0
Reserved	WWDRXN
R-0	R/WP-5h

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privileged mode only; -*n* = value after reset

Table 17-35. Digital Windowed Watchdog Reaction Control (RTIWWDRXNCTRL) Field Descriptions

Bit	Field	Value	Description
31-4	Reserved	0	Reads return 0 and writes have no effect.
3-0	WWDRXN		The DWWD reaction
		5h	The windowed watchdog will cause a reset if the watchdog is serviced outside the time window defined by the configuration, or if the watchdog is not serviced at all.
		Ah	The windowed watchdog will generate a non-maskable interrupt to the CPU if the watchdog is serviced outside the time window defined by the configuration, or if the watchdog is not serviced at all.
		All other values	The windowed watchdog will cause a reset if the watchdog is serviced outside the time window defined by the configuration, or if the watchdog is not serviced at all.
			Note: The DWWD reaction can be selected by the application even when the DWWD counter is already enabled. If a change to the WWDRXN is made before the watchdog service window is opened, then the change in the configuration takes effect immediately. If a change to the WWDRXN is made when the watchdog service window is already open, then the change in configuration takes effect only after the watchdog is serviced.

17.3.34 Digital Windowed Watchdog Window Size Control (RTIWWDSIZECTRL)

This register selects the DWWD window size. This register is shown in Figure 17-45 and described in Table 17-36.

Figure 17-45. Digital Windowed Watchdog Window Size Control (RTIWWDSIZECTRL) [offset = A8h]

31	16
WWDSIZE	
R/WP-0000	

15	0
WWDSIZE	
R/WP-0005h	

LEGEND: R/W = Read/Write; WP = Write in privileged mode only; -*n* = value after reset

Table 17-36. Digital Windowed Watchdog Window Size Control (RTIWWDSIZECTRL) Field Descriptions

Bit	Field	Value	Description
31-0	WWDSIZE	0	The DWWD window size
		0000 0005h	100% (The functionality is the same as the standard time-out digital watchdog.)
		0000 0050h	50%
		0000 0500h	25%
		0000 5000h	12.5%
		0005 0000h	6.25%
		All other values	3.125%
			Note: The DWWD window size can be selected by the application even when the DWWD counter is already enabled. If a change to the WWDSIZE is made before the watchdog service window is opened, then the change in the configuration takes effect immediately. If a change to the WWDSIZE is made when the watchdog service window is already open, then the change in configuration takes effect only after the watchdog is serviced.

17.3.35 RTI Compare Interrupt Clear Enable Register (RTIINTCLRENABLE)

When the RTI compare event is configured to generate a DMA request or triggers (all triggered by RTI compare interrupt request flag) to other peripherals, it is often desirable to clear the RTI compare flag automatically so that the requests can be generated repeatedly without any CPU intervention. This register works with the RTI compare clear registers to enable an "auto-clear" of the compare interrupt enable bit after a compare equal event. This register is shown in Figure 17-46 and described in Table 17-37.

Figure 17-46. RTI Compare Interrupt Clear Enable Register (RTIINTCLRENABLE) [offset = ACh]

31 28	27 24	23 20	19 16
Reserved	INTCLRENABLE3	Reserved	INTCLRENABLE2
R-0	R/WP-5h	R-0	R/WP-5h

15 12	11 8	7 4	3 0
Reserved	INTCLRENABLE1	Reserved	INTCLRENABLE0
R-0	R/WP-5h	R-0	R/WP-5h

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privileged mode only; -*n* = value after reset

Table 17-37. RTI Compare Interrupt Clear Enable Register (RTIINTCLRENABLE) Field Descriptions

Bit	Field	Value	Description
31-28	Reserved	0	Reads return 0. Writes have no effect.
27-24	INTCLRENABLE3		Enables the auto-clear functionality on the compare 3 interrupt.
		5h	*Read:* Auto-clear for compare 3 interrupt is disabled. *Privileged Write:* Auto-clear for compare 3 interrupt becomes disabled.
		All other values	*Read:* Auto-clear for compare 3 interrupt is enabled. *Privileged Write:* Auto-clear for compare 3 interrupt becomes enabled.
23-20	Reserved	0	Reads return 0. Writes have no effect.
19-16	INTCLRENABLE2		Enables the auto-clear functionality on the compare 2 interrupt.
		5h	*Read:* Auto-clear for compare 2interrupt is disabled. *Privileged Write:* Auto-clear for compare 2 interrupt becomes disabled.
		All other values	*Read:* Auto-clear for compare 2 interrupt is enabled. *Privileged Write:* Auto-clear for compare 2 interrupt becomes enabled.
15-12	Reserved	0	Reads return 0. Writes have no effect.
11-8	INTCLRENABLE1		Enables the auto-clear functionality on the compare 1 interrupt.
		5h	*Read:* Auto-clear for compare 1 interrupt is disabled. *Privileged Write:* Auto-clear for compare 1 interrupt becomes disabled.
		All other values	*Read:* Auto-clear for compare 1 interrupt is enabled. *Privileged Write:* Auto-clear for compare 1 interrupt becomes enabled.
7-4	Reserved	0	Reads return 0. Writes have no effect.
3-0	INTCLRENABLE0		Enables the auto-clear functionality on the compare 0 interrupt.
		5h	*Read:* Auto-clear for compare 0 interrupt is disabled. *Privileged Write:* Auto-clear for compare 0 interrupt becomes disabled.
		All other values	*Read:* Auto-clear for compare 0 interrupt is enabled. *Privileged Write:* Auto-clear for compare 0 interrupt becomes enabled.

17.3.36 RTI Compare 0 Clear Register (RTICMP0CLR)

This registers holds an initial value which is larger than the value in the RTI Compare 0 register Section 17.3.4. The user needs to choose the value such that the compare clear 0 event occurs before next compare 0 event. If the Free Running Counter matches the compare value, the compare 0 interrupt request flag is cleared and the value in the RTIUDCP0 register Section 17.3.16 is added to this register. This register is shown in Figure 17-47 and described in Table 17-38.

Figure 17-47. RTI Compare 0 Clear Register (RTICMP0CLR) [offset = B0h]

31	16
CMP0CLR	
R/WP-0	

15	0
CMP0CLR	
R/WP-0	

LEGEND: R/W = Read/Write; WP = Write in privileged mode only; -*n* = value after reset

Table 17-38. RTI Compare 0 Clear Register (RTICMP0CLR) Field Descriptions

Bit	Field	Value	Description
31-0	CMP0CLR	0-FFFF FFFFh	Compare 0 clear. This registers holds a compare value. If the Free Running Counter matches the compare value, the compare 0 interrupt request flag is cleared and the value in the RTIUDCP0 register Section 17.3.16 is added to this register. Reads return the current compare clear value. A privileged write to this register updates the compare clear value.

17.3.37 RTI Compare 1 Clear Register (RTICMP1CLR)

This registers holds an initial value which is larger than the value in the RTI Compare 1 register Section 17.3.4. The user needs to choose the value such that the compare clear 1 event occurs before next compare 1 event. If the Free Running Counter matches the compare value, the compare 1 interrupt request flag is cleared and the value in the RTIUDCP1 register Section 17.3.18 is added to this register. This register is shown in Figure 17-48 and described in Table 17-39.

Figure 17-48. RTI Compare 1 Clear Register (RTICMP1CLR) [offset = B4h]

31	16
CMP1CLR	
R/WP-0	

15	0
CMP1CLR	
R/WP-0	

LEGEND: R/W = Read/Write; WP = Write in privileged mode only; -*n* = value after reset

Table 17-39. RTI Compare 1 Clear Register (RTICMP1CLR) Field Descriptions

Bit	Field	Value	Description
31-0	CMP0CLR	0-FFFF FFFFh	Compare 1 clear. This registers holds a compare value. If the Free Running Counter matches the compare value, the compare 1 interrupt request flag is cleared and the value in the RTIUDCP1 register Section 17.3.18 is added to this register. Reads return the current compare clear value. A privileged write to this register updates the compare clear value.

17.3.38 RTI Compare 2 Clear Register (RTICMP2CLR)

This registers holds an initial value which is larger than the value in the RTI Compare 2 register Section 17.3.4. The user needs to choose the value such that the compare clear 2 event occurs before next compare 2 event. If the Free Running Counter matches the compare value, the compare 2 interrupt request flag is cleared and the value in the RTIUDCP2 register Section 17.3.20 is added to this register. This register is shown in Figure 17-49 and described in Table 17-40.

Figure 17-49. RTI Compare 2 Clear Register (RTICMP2CLR) [offset = B8h]

31	16
CMP2CLR	
R/WP-0	

15	0
CMP2CLR	
R/WP-0	

LEGEND: R/W = Read/Write; WP = Write in privileged mode only; -*n* = value after reset

Table 17-40. RTI Compare 2 Clear Register (RTICMP2CLR) Field Descriptions

Bit	Field	Value	Description
31-0	CMP2CLR	0-FFFF FFFFh	Compare 2 clear. This registers holds a compare value. If the Free Running Counter matches the compare value, the compare 2 interrupt request flag is cleared and the value in the RTIUDCP2 register Section 17.3.20 is added to this register. Reads return the current compare clear value. A privileged write to this register updates the compare clear value.

17.3.39 RTI Compare 3 Clear Register (RTICMP3CLR)

This registers holds an initial value which is larger than the value in the RTI Compare 3 register Section 17.3.4. The user needs to choose the value such that the compare clear 3 event occurs before next compare 3 event. If the Free Running Counter matches the compare value, the compare 3 interrupt request flag is cleared and the value in the RTIUDCP3 register Section 17.3.22 is added to this register. This register is shown in Figure 17-50 and described in Table 17-41.

Figure 17-50. RTI Compare 3 Clear Register (RTICMP3CLR) [offset = BCh]

31	16
CMP3CLR	
R/WP-0	

15	0
CMP3CLR	
R/WP-0	

LEGEND: R/W = Read/Write; WP = Write in privileged mode only; -*n* = value after reset

Table 17-41. RTI Compare 3 Clear Register (RTICMP3CLR) Field Descriptions

Bit	Field	Value	Description
31-0	CMP3CLR	0-FFFF FFFFh	Compare 3 clear. This registers holds a compare value. If the Free Running Counter matches the compare value, the compare 3 interrupt request flag is cleared and the value in the RTIUDCP3 register Section 17.3.22 is added to this register. Reads return the current compare clear value. A privileged write to this register updates the compare clear value.

Cyclic Redundancy Check (CRC) Controller Module

This chapter describes the cyclic redundancy check (CRC) controller module.

NOTE: This chapter describes a superset implementation of the CRC module that includes features and functionality that require DMA. Since not all devices have DMA capability, consult your device-specific datasheet to determine applicability of these features and functions to your device being used.

Topic **Page**

18.1 Overview

The CRC controller is a module that is used to perform CRC (Cyclic Redundancy Check) to verify the integrity of memory system. A signature representing the contents of the memory is obtained when the contents of the memory are read into CRC controller. The responsibility of CRC controller is to calculate the signature for a set of data and then compare the calculated signature value against a pre-determined good signature value. CRC controller supports two channels to perform CRC calculation on multiple memories in parallel and can be used on any memory system.

18.1.1 Features

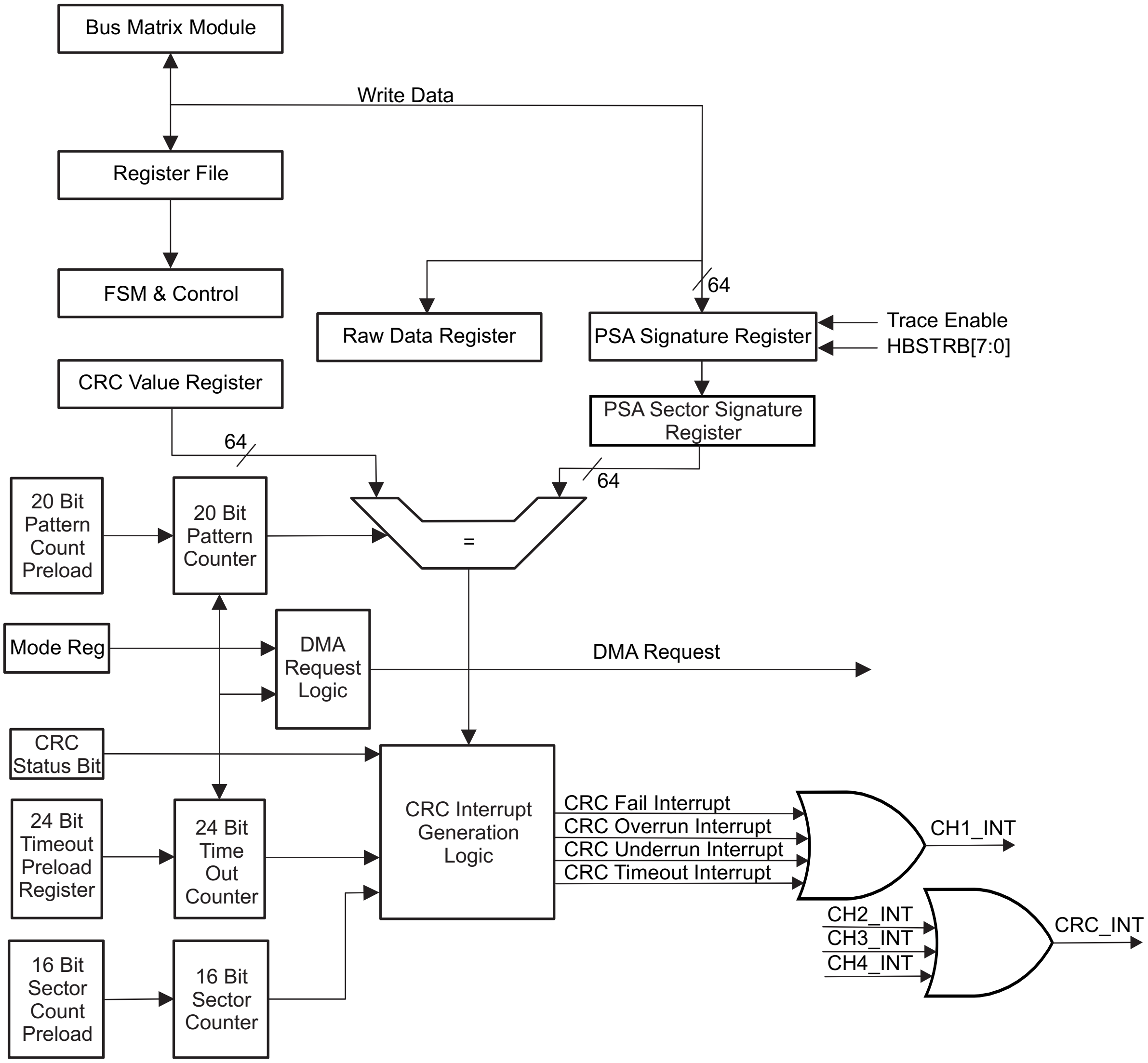
The CRC controller offers:

- Two channels to perform background signature verification on any memory sub-system.
- Data compression on 8, 16, 32, and 64 bit data size.
- Maximum-length PSA (Parallel Signature Analysis) register constructed based on 64 bit primitive polynomial.
- Each channel has a CRC Value Register that contains the pre-determined CRC value.
- Use timed base event trigger from timer to initiate DMA data transfer.
- Programmable 20-bit pattern counter per channel to count the number of data patterns for compression.
- Three modes of operation. Auto, Semi-CPU and Full-CPU.
- For each channel, CRC can be performed either by CRC Controller or by CPU.
- Automatically perform signature verification without CPU intervention in AUTO mode.
- Generate interrupt to CPU in Semi-CPU mode to allow CPU to perform signature verification itself.
- Generate CRC fail interrupt in AUTO mode if signature verification fails.
- Generate Timeout interrupt if CRC is not performed within the time limit.
- Generate DMA request per channel to initiate CRC value transfer.

18.1.2 Block Diagram

Figure 18-1 shows a block diagram of the CRC controller.

Figure 18-1. CRC Controller Block Diagram For One Channel



18.2 Module Operation

18.2.1 General Operation

There are two channels in CRC controller and for each channel there is a memory mapped PSA (Parallel Signature Analysis) Signature Register and a memory mapped CRC (Cyclic Redundancy Check) Value register. A memory can be organized into multiple sectors with each sector consisting of multiple data patterns. A data pattern can be 8-, 16-, 32-, or 64-bit data. CRC module performs the signature calculation and compares the signature to a pre-determined value. The PSA Signature Register compresses an incoming data pattern into a signature when it is written. When one sector of data patterns are written into PSA Signature Register, a final signature corresponding to the sector is obtained. CRC Value Register stores the pre-determined signature corresponding to one sector of data patterns. The calculated signature and the pre-determined signature are then compared to each other for signature verification. To minimize CPU’s involvement, data patterns transfer can be carried out at the background of CPU using DMA controller. DMA is setup to transfer data from memory from which the contents to be verified to the memory mapped PSA Signature Register. When DMA transfers data to the memory mapped PSA Signature Register, a signature is generated. A programmable 20-bit data pattern counter is used for each channel to define the number of data patterns to calculate for each sector. Signature verification can be performed automatically by CRC controller in AUTO mode or by CPU itself in Semi-CPU or Full-CPU mode. In AUTO mode, a self sustained CRC signature calculation can be achieved without any CPU intervention.

18.2.2 CRC Modes of Operation

CRC Controller can operate in AUTO, Semi-CPU, and Full-CPU modes.

18.2.2.1 AUTO Mode

In AUTO mode, CRC Controller in conjunction with DMA controller can perform CRC totally without CPU intervention. A sustained transfer of data to both the PSA Signature Register and CRC Value Register are performed in the background of CPU. When a mismatch is detected, an interrupt is generated to CPU. A 16 bit current sector ID register is provided to identify which sector causes a CRC failure.

18.2.2.2 Semi-CPU Mode

In Semi-CPU mode, DMA controller is also utilized to perform data patterns transfer to PSA Signature Register. Instead of performing signature verification automatically, the CRC controller generates an compression complete interrupt to CPU after each sector is compressed. Upon responding to the interrupt the CPU performs the signature verification by reading the calculated signature stored at the PSA Sector Signature Register and compare it to a pre-determined CRC value.

18.2.2.3 Full CPU Mode

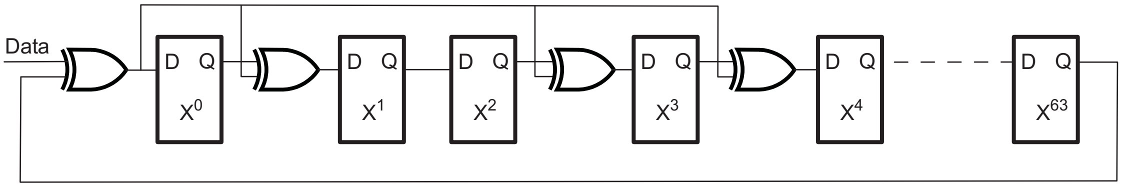
In Full-CPU mode, the CPU does the data patterns transfer and signature verification all by itself. When CPU has enough throughput, it can perform data patterns transfer by reading data from the memory system to the PSA Signature Register. After certain number of data patterns are compressed, the CPU can read from the PSA Signature Register and compare the calculated signature to the pre-determined CRC signature value. In Full-CPU mode, neither interrupt nor DMA request is generated. All counters are also disabled.

18.2.3 PSA Signature Register

The 64-bit PSA Signature Register is based on the primitive polynomial (as in the following equation) to produce the maximum length LFSR (Linear Feedback Shift Register), as shown in Figure 18-2.

$$f(x) = x^{64} + x^4 + x^3 + x + 1 \tag{25}$$

Figure 18-2. Linear Feedback Shift Register (LFSR)



The serial implementation of LFSF has a limitation that, it requires ‘n’ clock cycles to calculate the CRC values for an ‘n’ bit data stream. The idea is to produce the same CRC value operating on a multi-bit data stream, as would occur if the CRC were computed one bit at a time over the whole data stream. The algorithm involves looping to simulate the shifting, and concatenating strings to build the equations after ‘n’ shift.

The parallel CRC calculation based on the polynomial can be illustrated in the following HDL code:

```
for i in 63 to 0 loop
   NEXT_CRC_VAL(0) := CRC_VAL(63) xor DATA(i);
     for j in 1 to 63 loop
       case j is
          when 1|3|4 =>
               NEXT_CRC_VAL(j) :=
                   CRC_VAL(j - 1) xor CRC_VAL(63) xor DATA(i);
          when others =>
               NEXT_CRC_VAL(j) := CRC_VAL(j - 1);
       end case;
     end loop;
   CRC_VAL := NEXT_CRC_VAL;
end loop;
```

NOTE: 1) The inner loop is to calculate the next value of each shift register bit after one cycle

2) The outer loop is to simulate 64 cycles of shifting. The equation for each shift register bit is thus built before it is compressed into the shift register.

3) MSB of the DATA is shifted in first

There is one PSA Signature Register per CRC channel. PSA Signature Register can be both read and written. When it is written, it can either compress the data or just capture the data depending on the state of CHx_MODE bits. If CHx_MODE=Data Capture, a seed value can be planted in the PSA Signature Register without compression. Other modes other than Data Capture will result with the data compressed by PSA Signature Register when it is written. Each channel can be planted with different seed value before compression starts. When PSA Signature Register is read, it gives the calculated signature.

CRC Controller should be used in conjunction with the on chip DMA controller to produce optimal system performance. The incoming data pattern to PSA Signature Register is typically initiated by the DMA master. When DMA is properly setup, it would read data from the pre-determined memory system and write them to the memory mapped PSA Signature Register. Each time PSA Signature Register is written a signature is generated. CPU itself can also perform data transfer by reading from the memory system and perform write operation to PSA Signature Register if CPU has enough throughput to handle data patterns transfer.

After system reset and when AUTO mode is enabled, CRC Controller automatically generates a DMA request to request the pre-determined CRC value corresponding to the first sector of memory to be checked.

In AUTO mode, when one sector of data patterns is compressed, the signature stored at the PSA Signature Register is first copied to the PSA Sector Signature Register and PSA Signature Register is then cleared out to all zeros. An automatic signature verification is then performed by comparing the signature stored at the PSA Sector Signature Register to the CRC Value Register. After the comparison the CRC Controller can generate a DMA request. Upon receiving the DMA request the DMA controller will update the CRC Value Register by transferring the next pre-determined signature value associated with the next sector of memory system. If the signature verification fails then CRC Controller can generate a CRC fail interrupt.

In Full-CPU mode, no DMA request and interrupt are generated at all. The number of data patterns to be compressed is determined by CPU itself. Full-CPU mode is useful when DMA controller is not available to perform background data patterns transfer. The OS can periodically generate a software interrupt to CPU and use CPU to accomplish data transfer and signature verification.

CRC Controller supports doubleword, word, half word and byte access to the PSA Signature Register. During a non-doubleword write access, all unwritten byte lanes are padded with zero’s before compression. Note that comparison between PSA Sector Signature Register and CRC Value Register is always in 64 bit because a compressed value is always expressed in 64 bit.

There is a software reset per channel for PSA Signature Register. When set, the PSA Signature Register is reset to all zeros.

PSA Signature Register is reset to zero under the following conditions:

- System reset
- PSA Software reset
- One sector of data patterns are compressed

18.2.4 PSA Sector Signature Register

After one sector of data is compressed, the final resulting signature calculated by PSA Signature Register is transferred to the PSA Sector Signature Register. PSA Signature Register is a read only register. During Semi-CPU mode, the host CPU should read from the PSA Sector Signature Register instead of reading from PSA Signature Register for signature verification to avoid data coherency issue. The PSA Signature Register can be updated with new signature before the host CPU is able to retrieve it.

In Semi-CPU mode, no DMA request is generated. When one sector of data patterns is compressed, CRC controller first generates a compression complete interrupt. Responding to the interrupt, CPU will in the ISR read the PSA Sector Signature Register and compare it to the known good signature or write the signature value to another memory location to build a signature file. In Semi-CPU mode, CPU must perform the signature verification in a manner to prevent any overrun condition. The overrun condition occurs when the compression complete interrupt is generated after one sector of data patterns is compressed and CPU has not read from the PSA Sector Signature Register to perform necessary signature verification before PSA Sector Signature Register is overridden with a new value. An overrun interrupt can be enable to generate when overrun condition occurs. During Semi-CPU mode, the host CPU should read from the PSA Sector Signature Register instead of reading from PSA Signature Register for signature verification to avoid data coherency issue. The PSA Signature Register can be updated with new signature before the host CPU is able to retrieve it.

18.2.5 CRC Value Register

Associated with each channel there is a CRC Value Register. The CRC Value Register stores the pre-determined CRC value. After one sector of data patterns is compressed by PSA Signature Register, CRC Controller can automatically compare the resulting signature stored at the PSA Sector Signature Register with the pre-determined value stored at the CRC Value Register if AUTO mode is enabled. If the signature verification fails, CRC Controller can be enabled to generate an CRC fail interrupt. When the channel is set up for Semi-CPU mode, CRC controller first generates a compression complete interrupt to CPU. Upon servicing the interrupt, CPU will then read the PSA Sector Signature Register and then read the corresponding CRC value stored at another location and compare them. CPU should not read from the CRC Value Register during Semi-CPU or Full-CPU mode because the CRC Value Register is not updated during these two modes.

In AUTO mode, for first sector's signature, DMA request is generated when mode is programmed to AUTO. For subsequent sectors, DMA request is generated after each sector is compressed. Responding to the DMA request, DMA controller reloads the CRC Value Register for the next sector of memory system to be checked.

When CRC Value Register is updated with a new CRC value, an internal flag is set to indicate that CRC Value Register contains the most current value. This flag is cleared when CRC comparison is performed. Each time at the end of the final data pattern compression of a sector, CRC Controller first checks to see if the corresponding CRC Value Register has the most current CRC value stored in it by polling the flag. If the flag is set then the CRC comparison can be performed. If the flag is not set then it means the CRC Value Register contains stale information. A CRC underrun interrupt is generated. When an underrun condition is detected, signature verification is not performed.

CRC Controller supports doubleword, word, half word and byte access to the CRC Value Register. As noted before comparison between PSA Sector Signature Register and CRC Value Register during AUTO mode is carried out in 64 bit.

18.2.6 Raw Data Register

The raw or un-compressed data written to the PSA Signature Register is also saved in the Raw Data Register. This register is read only.

18.2.7 Example DMA Controller Setup

DMA controller needs to be setup properly in either either AUTO or Semi-CPU mode as DMA controller is used to transfer data patterns. Hardware or a combination of hardware and software DMA triggering are supported.

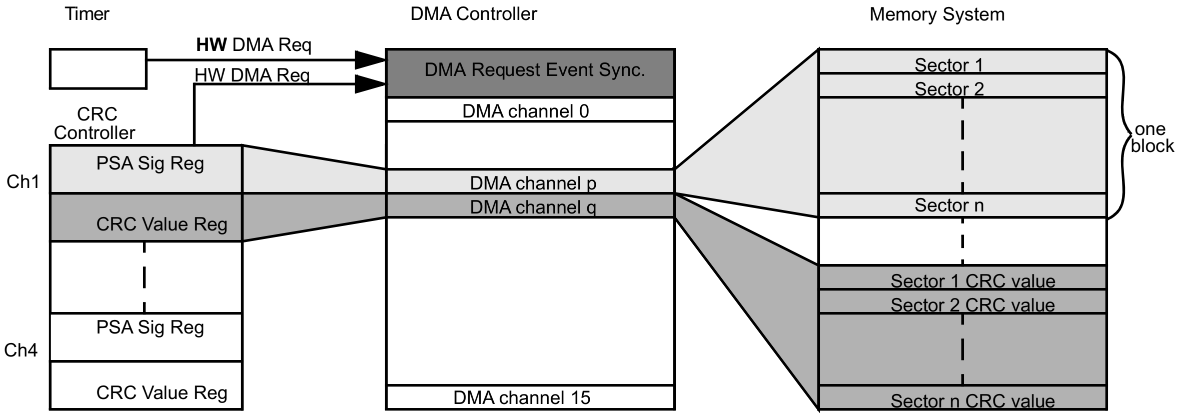
18.2.7.1 AUTO Mode Using Hardware Timer Trigger

There are two DMA channels associated with each CRC channel when in AUTO mode. One DMA channel is setup to transfer data patterns from the source memory to the PSA Signature Register. The second DMA channel is setup to transfer the pre-determined signature to the CRC Value Register. The trigger source for the first DMA channel can be either by hardware or by software. As illustrated in Figure 18-3 a timer can be used to trigger a DMA request to initiate transfer from the source memory system to PSA Signature Register. In AUTO mode, CRC Controller also generates DMA request after one sector of data patterns is compressed to initiate transfer of the next CRC value corresponding to the next sector of memory. Thus a new CRC value is always updated in the CRC Value Register by DMA synchronized to each sector of memory.

A block of memory system is usually divided into many sectors. All sectors are the same size. The sector size is programmed in the CRC_PCOUNT_REGx and the number of sectors in one block is programmed in the CRC_SCOUNT_REGx of the respective channel. CRC_PCOUNT_REGx multiplies CRC_SCOUNT_REGx and multiplies transfer size of each data pattern should give the total block size in number of bytes.

The total size of the memory system to be examined is also programmed in the respective transfer count register inside DMA module. The DMA transfer count register is divided into two parts. They are element count and frame count. Note that an HW DMA request can be programmed to trigger either one frame or one entire block transfer. In Figure 18-3, an HW DMA request from a timer is used as a trigger source to initiate DMA transfer. If all four CRC channels are active in AUTO mode then a total of four DMA requests would be generated by CRC Controller.

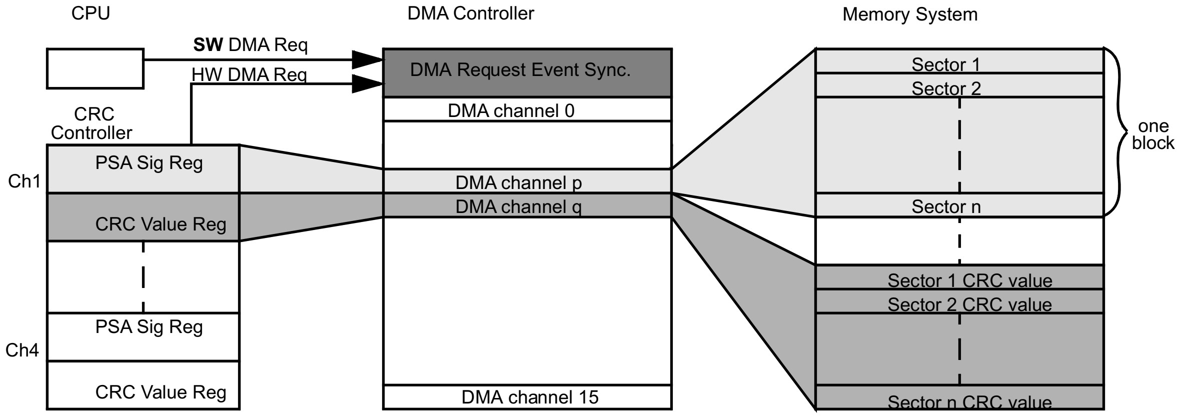
Figure 18-3. AUTO Mode Using Hardware Timer Trigger



18.2.7.2 AUTO Mode Using Software Trigger

The data patterns transfer can also be initiated by software. CPU can generate a software DMA request to activate the DMA channel to transfer data patterns from source memory system to the PSA Signature Register. To generate a software DMA request CPU needs to set the corresponding DMA channel in the DMA software trigger register. Note that just one software DMA request from CPU is enough to complete the entire data patterns transfer for all sectors. See Figure 18-4 for an illustration.

Figure 18-4. AUTO Mode With Software CPU Trigger



18.2.7.3 Semi-CPU Mode Using Hardware Timer Trigger

During semi-CPU mode, no DMA request is generated by CRC controller. Therefore, no DMA channel is allocated to update CRC Value Register. CPU should not read from CRC Value Register in semi-CPU mode as it contains stale value. Note that no signature verification is performed at all during this mode. Similar to AUTO mode, either by hardware or by software DMA request can be used as a trigger for data patterns transfer. Figure 18-5 illustrates the DMA setup using semi-CPU mode with hardware timer trigger.

Figure 18-5. Semi-CPU Mode With Hardware Timer Trigger

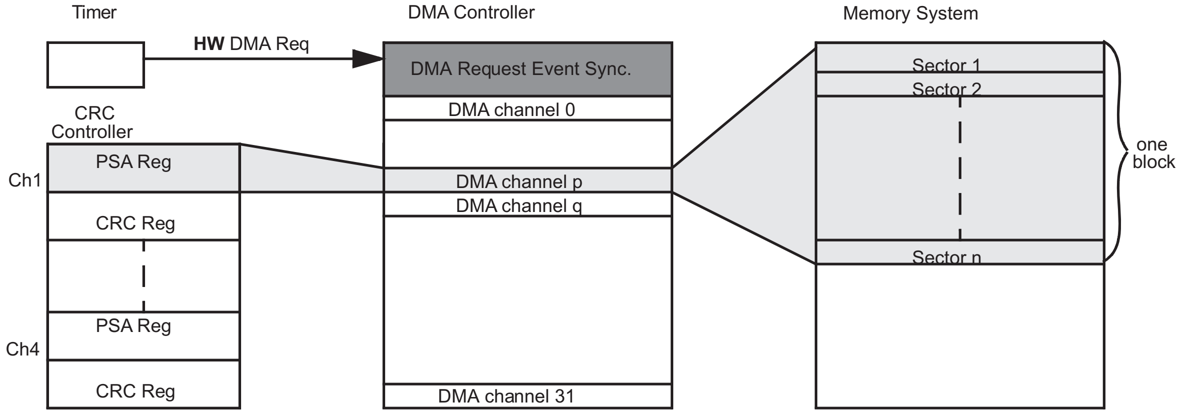


Table 18-1. CRC Modes in Which DMA Request and Counter Logic are Active or Inactive

Mode	DMA Request	Pattern Counter	Sector Counter	Timeout Counter
AUTO	Active	Active	Active	Active
Semi-CPU	Inactive	Active	Active	Active
Full-CPU	Inactive	Inactive	Inactive	Inactive

18.2.8 Pattern Count Register

There is a 20-bit data pattern counter for every CRC channel. The data pattern counter is a down counter and can be pre-loaded with a programmable value stored in the Pattern Count Register. When the data pattern counter reaches zero, a compression complete interrupt is generated in Semi-CPU mode and an automatic signature verification is performed in AUTO mode. In AUTO only, DMA request is generated to trigger the DMA controller to update the CRC Value Register.

NOTE: The data pattern count should be divisible by the total transfer count as programmed in DMA controller. The total transfer count is the product of element count and frame count.

18.2.9 Sector Count Register/Current Sector Register

Each channel contains a 16 bit sector counter. The sector count register stores the number of sectors. Sector counter is a free running counter and is incremented by one each time when one sector of data patterns is compressed. When the signature verification fails, the current value stored in the sector counter is saved into current sector register. If signature verification fails, CPU can read from the current sector register to identify the sector which causes the CRC mismatch. To aid and facilitate the CPU in determining the cause of a CRC failure, it is advisable to use the following equation during CRC and DMA setup:

$$CRC\ Pattern\ Count \times CRC\ Sector\ Count = DMA\ Element\ Count \times DMA\ Frame\ Count$$

The current sector register is frozen from being updated until both the current sector register is read and CRC fail status bit is cleared by CPU. If CPU does not respond to the CRC failure in a timely manner before another sector produces a signature verification failure, the current sector register is not updated with the new sector number. An overrun interrupt is generate instead. If current sector register is already frozen with an erroneous sector and emulation is entered with SUSPEND signal goes to high then the register still remains frozen even it is read.

In Semi-CPU mode, the current sector register is used to indicate the sector for which the compression complete has last happened.

The current sector register is reset when the PSA software reset is enabled.

NOTE: Both data pattern count and sector count registers must be greater than or equal to one for the counters to count. After reset, pattern count and sector count registers default to zero and the associated counters are inactive.

18.2.10 Interrupt

The CRC controller generates several types of interrupts per channel. Associated with each interrupt, there is an interrupt enable bit. No interrupt is generated in Full-CPU mode.

- Compression complete interrupt
- CRC fail interrupt
- Overrun interrupt
- Underrun interrupt
- Timeout interrupt

Table 18-2. Modes in Which Interrupt Condition Can Occur

	AUTO	Semi-CPU	Full-CPU
Compression Complete	no	yes	no
CRC Fail	yes	no	no
Overrun	yes	yes	no
Underrun	yes	no	no
Timeout	yes	yes	no

18.2.10.1 Compression Complete Interrupt

Compression complete interrupt is generated in Semi-CPU mode only. When the data pattern counter reaches zero, the compression complete flag is set and the interrupt is generated.

18.2.10.2 CRC Fail Interrupt

CRC fail interrupt is generated in AUTO mode only. When the signature verification fails, the CRC fail flag is set,. CPU should take action to address the fail condition and clear the CRC fail flag after it resolves the CRC mismatch.

18.2.10.3 Overrun Interrupt

Overrun interrupt is generated in either AUTO or Semi-CPU mode. During AUTO mode, if a CRC fail is detected then the current sector number is recorded in the current sector register. If CRC fail status bit is not cleared and current sector register is not read by the host CPU before another CRC fail is detected for another sector then an overrun interrupt is generated. During Semi-CPU mode, when the data pattern counter finishes counting, it generates a compression complete interrupt. At the same time the signature is copied into the PSA Sector Signature Register. If the host CPU does not read the signature from PSA Sector Signature Register before it is updated again with a new signature value then an overrun interrupt is generated.

18.2.10.4 Underrun Interrupt

Underrun interrupt only occurs in AUTO mode. The interrupt is generated when the CRC Value Register is not updated with the corresponding signature when the data pattern counter finishes counting. During AUTO mode, CRC Controller generates DMA request to update CRC Value Register in synchronization to the corresponding sector of the memory. Signature verification is also performed if underrun condition is detected. And CRC fail interrupt is generated at the same time as the underrun interrupt.

18.2.10.5 Timeout Interrupt

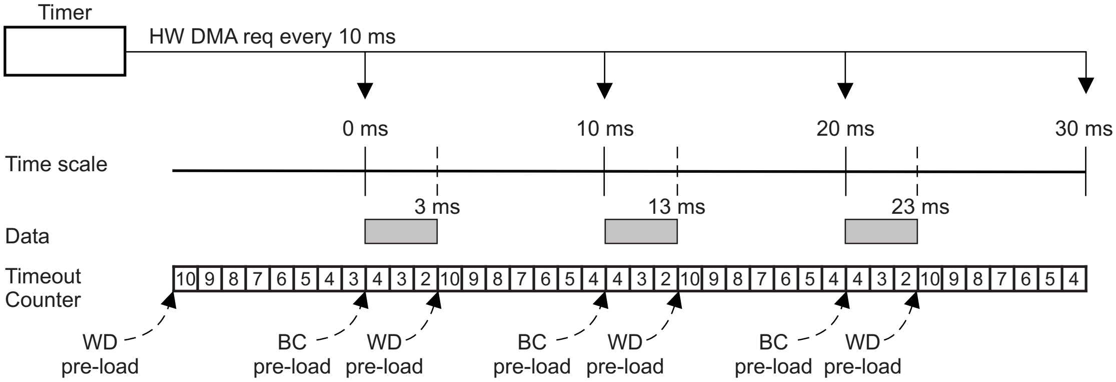
To ensure that the memory system is examined within a pre-defined time frame and no loss of incoming data there is a 24 bit timeout counter per CRC channel. The 24 bit timeout down counter can be pre-loaded with two different pre-load values, watchdog timeout pre-load value (CRC_WDTOPLDx) and block complete timeout pre-load value (CRC_BCTOPLDx). The timeout counter is clocked by a prescaler clock which is permanently running at division 64 of HCLK clock.

First pattern of data must be transferred by the DMA before the timeout counter expires, Watchdog timeout pre-load register (CRC_WDTOPLDx) is used as timeout counter. Block complete timeout pre-load register (CRC_BCTOPLDx) is used to check if one complete block of data patterns are compressed within a specific time frame. The timeout counter is first pre-loaded with CRC_WDTOPLDx after either AUTO or Semi-CPU mode is selected and starts to down count. If the timeout counter expires before DMA transfers any data pattern to PSA Signature Register then a timeout interrupt is generated. An incoming data pattern before the timeout counter expires will automatically pre-load the timeout counter with CRC_BCTOPLDx the block complete timeout pre-load value.

Block complete timeout pre-load value is used to check it one block of data patterns are compressed within a given time limit. If the timeout counter pre-loaded with CRC_BCTOPLDx value expires before one block of data patterns are compressed a timeout interrupt is generated. When one block (pattern count x sector count) of data patterns are compressed before the counter has expired, the counter is pre-loaded with CRC_WDTOPLDx value again. If the timeout counter is pre-loaded with zero then the counter is disable and no timeout interrupt is generated.

In Figure 18-6, a timer generates DMA request every 10ms to trigger one block (pattern count x sector count) transfer. Since we want to make sure that DMA does start to transfer a block every 10 ms we would set the first pre-load value to 10ms in CRC_WDTOPLDx. We also want to make sure that one block of data patterns are compressed within 4ms. With such a requirement, we would set the second pre-load value to 4ms in CRC_BCTOPLDx register.

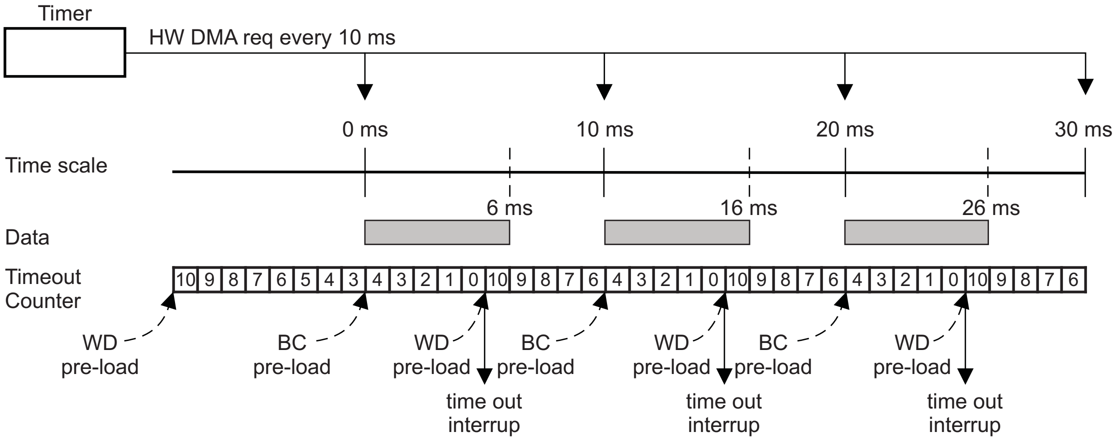
Figure 18-6. Timeout Example 1



WD pre-load = watchdog timeout pre-load (CRC_WDTOPLDx)
BC pre-load = block complete timeout pre-load (CRC_BCTOPLDx)

Note: No timeout interrupt is generated in this example since each block of data patterns are compressed in 3 ms and DMA does initiate a block transfer every 10 ms.

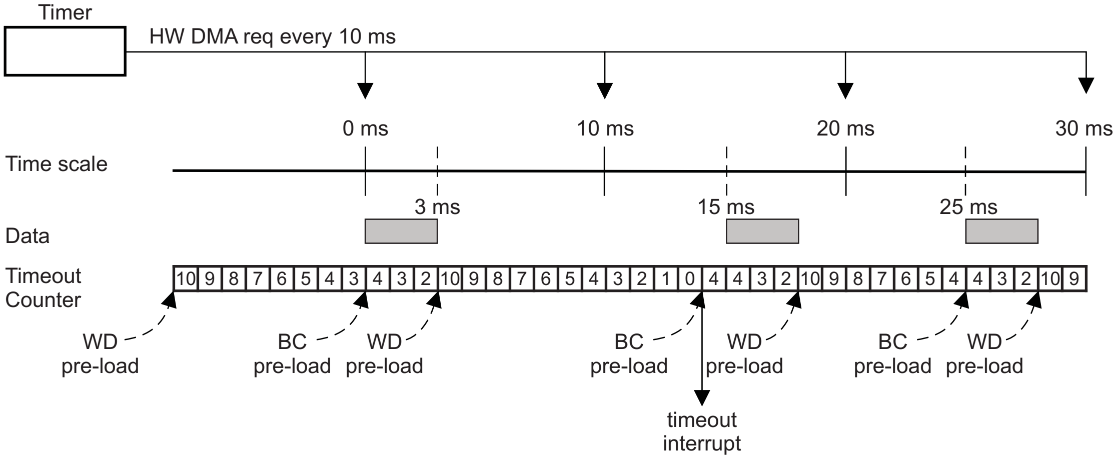
Figure 18-7. Timeout Example 2



WD pre-load = watchdog timeout pre-load (CRC_WDTOPLDx)
BC pre-load = block complete timeout pre-load (CRC_BCTOPLDx)

Note: Timeout interrupt is generated in this example since each block of data patterns are compressed in 6 ms and this is out of the 4ms time frame.

Figure 18-8. Timeout Example 3



WD pre-load = watchdog timeout pre-load (CRC_WDTOPLDx)
BC pre-load = block complete timeout pre-load (CRC_BCTOPLDx)

Note: Timeout interrupt is generated in this example since DMA can not transfer the second block of data within 10ms time limit and the reason may be that DMA is set up in fixed priority scheme and DMA is serving other higher priority channels at the time before it can service the timer request.

18.2.10.6 Interrupt Offset Register

CRC Controller only generates one interrupt request to interrupt manager. A interrupt offset register is provided to indicate the source of the pending interrupt with highest priority. Table 18-3 shows the offset interrupt vector address of each interrupt condition in an ascending order of priority.

Table 18-3. Interrupt Offset Mapping

Offset Value	Interrupt Condition
0	Phantom
1h	Ch1 CRC Fail
2h	Ch2 CRC Fail
3h-8h	Reserved
9h	Ch1 Compression Complete
Ah	Ch2 Compression Complete
Bh-10h	Reserved
11h	Ch1 Overrun
12h	Ch2 Overrun
13h-18h	Reserved
19h	Ch1 Underrun
1Ah	Ch2 Underrun
1Bh-20h	Reserved
21h	Ch1 Timeout
22h	Ch2 Timeout
23h-24h	Reserved

18.2.10.7 Error Handling

When an interrupt is generated, host CPU should take appropriate actions to identify the source of error and restart the respective channel in DMA and CRC module. To restart a CRC channel, the user should perform the following steps in the ISR:

1. Write to software reset bit in CRC_CTRL register to reset the respective PSA Signature Register.
2. Reset the CHx_MODE bits to 00 in CRC_CTRL register as Data capture mode.
3. Set the CHx_MODE bits in CRC_CTRL register to desired new mode again.
4. Release software reset.

The host CPU should use byte write to restart each individual channel.

18.2.11 Power Down Mode

CRC module can be put into power down mode when the power down control bit PWDN is set. The module wakes up when the PWDN bit is cleared.

18.2.12 Emulation

A read access from a register in functional mode can sometimes trigger a certain internal event to follow. For example, reading an interrupt offset register triggers an event to clear the corresponding interrupt status flag. During emulation when SUSPEND signal is high, a read access from any register should only return the register contents to the bus and should not trigger or mask any event as it would have in functional mode. This is to prevent debugger from reading the interrupt offset register during refreshing screen and cause the corresponding interrupt status flag to get cleared. Timeout counters are stopped to generate timeout interrupts in emulation mode. No Peripheral Master bus error should be generated if reading from the unimplemented locations.

18.2.13 Peripheral Bus Interface

CRC is a Peripheral slave module. The register interface is similar to other peripheral modules. CRC supports following features:

- Different sizes of burst operation.
- Aligned and unaligned accesses.
- Abort is generated for any illegal address accesses.

18.3 Example

This section illustrates several of the ways in which the CRC Controller can be utilized to perform CRC.

18.3.1 Example: Auto Mode Using Time Based Event Triggering

A large memory area with 2Mbyte (256k doubleword) is to be checked in the background of CPU. CRC is to be performed every 1K byte (128 doubleword). Therefore there should be 2048 pre-recorded CRC values. For illustration purpose, we map channel 1 CRC Value Register to DMA channel 1 and channel 1 PSA Signature Register to DMA channel 2. Assume all DMA transfers are carried out in 64-bit transfer size.

18.3.1.1 DMA Setup

- Set up DMA channel 1 with the starting address from which the pre-determined CRC values are stored. Set up the destination address to the memory mapped channel 1 CRC Value Register. Put the source address at post increment addressing mode and put the destination address at constant addressing mode. Use ***hardware*** DMA request for channel 1 to trigger a ***frame*** transfer.
- Set up DMA channel 2 with the source address from which the contents of memory to be verified. Set up the destination address to the memory mapped channel 1 PSA Signature Register. Program the element transfer count to 128 and the frame transfer count to 2048. Put the source address at post increment addressing mode and put the destination address at constant address mode. Use ***hardware*** DMA request for channel 2 to trigger an entire ***block*** transfer.

18.3.1.2 Timer Setup

The timer can be any general purpose timer which is capable of generating a time-based DMA request.

- Set up timer to generate DMA request associated with DMA channel 2. For example, an OS can set up the timer to generate a DMA request every 10ms.

18.3.1.3 CRC Setup

- Program the pattern count to 128.
- Program the sector count to 2048.
- For example, we want the entire 2Mbytes to be compressed within 5ms. We can program the block complete timeout pre-load (CRC_BCTOPLDx) value to 15625 (5 ms / (1 HCLK period × 64)) if CRC is operating at 200 MHz.
- Enable AUTO mode and all interrupts.

After AUTO mode is selected, CRC Controller automatically generates a DMA request on channel 1. Around the same time the timer module also generates a DMA request on DMA channel 2. When the first incoming data pattern arrives at the PSA Signature Register, the CRC Controller will compress it. After some time, the DMA controller would update the CRC Value Register with a pre-determined value matching the calculated signature for the first sector of 128 64 bit data patterns. After one sector of data patterns are compressed, the CRC Controller generate a CRC fail interrupt if signature stored at the PSA Sector Signature Register does not match the CRC Value Register. CRC Controller generates a DMA request on DMA channel 1 when one sector of data patterns are compressed. This routine will continue until the entire 2Mbyte are consumed. If the timeout counter reached zero before the entire 2Mbytes are compressed a timeout interrupt is generated. After 2MBytes are transferred, the DMA can generate an interrupt to CPU. The entire operation will continue again when DMA responds to the DMA request from both the timer and CRC Controller. The CRC is performed totally without any CPU intervention.

18.3.2 Example: Auto Mode Without Using Time Based Triggering

A small but highly secured memory area with 1kbytes is to be checked in the background of CPU. CRC is to be performed every 1Kbytes. Therefore there is only one pre-recorded CRC value. For illustration purpose, we map channel 1 CRC Value Register to DMA channel 1 and channel 1 PSA Signature Register to DMA channel 2. Assume all transfers carried out by DMA are in 64 bit transfer size.

18.3.2.1 DMA Setup

- Set up DMA channel 1 with the source address from which the pre-determined CRC value is stored. Set up the destination address to the memory mapped channel 1 CRC Value Register. Put the source address at constant addressing mode and put the destination address at constant addressing mode. Use ***hardware*** DMA request for channel 1.
- Set up DMA channel 2 with the source address from which the memory area to be verified. Set up the destination address to the memory mapped channel 1 PSA Signature Register. Program the element transfer count to 128 and the frame transfer count to 1. Put the source address at post increment addressing mode and put the destination address at constant address mode. Generate a ***software*** DMA request on channel 2 after CRC has completed its setup. Enable autoinitiation for DMA channel 2.

18.3.2.2 CRC Setup

- Program the pattern count to 128.
- Program the sector count to 1.
- Leaving the timeout count register with the reset value of zero means no timeout interrupt is generated.
- Enable AUTO mode and all interrupts.

After AUTO mode is selected, the CRC Controller automatically generates a DMA request on channel 1. At the same time the CPU generates a ***software*** DMA request on DMA channel 2. When the first incoming data pattern arrives at the PSA Signature Register, the CRC Controller will compress it. After some time, the DMA controller would update the CRC Value Register with a pre-determined value matching the calculated signature for the first sector of 128 64 bit data patterns. After one sector of data patterns are compressed, the CRC Controller generates a CRC fail interrupt if signature stored at the PSA Sector Signature Register does not match the CRC Value Register. CRC Controller generates a DMA request on DMA channel 1 again after one sector is compressed. After 1kbytes are transferred, the DMA can generate an interrupt to CPU. Responding to the DMA interrupt CPU can restart the CRC routine by generating a software DMA request onto channel 2 again.

18.3.3 Example: Semi-CPU Mode

If DMA controller is available in a system, the CRC module can also operate in semi-CPU mode. This means that CPU can still make use of the DMA to perform data patterns transfer to CRC controller in the background. The difference between semi-CPU mode and AUTO mode is that CRC controller does not automatically perform the signature verification. CRC controllers generates a compression complete interrupt to CPU when the one sector of data patterns are compressed. CPU needs to perform the signature verification itself.

A memory area with 2Mbyte is to be verified with the help of the CPU. CRC operation is to be performed every 1K byte. Since there are 2Mbyte (256k doublewords) of memory to be check and we want to perform a CRC every 1Kbyte (128 doublewords) and therefore there should be 2048 pre-recorded CRC values. In Semi-CPU mode, the CRC Value Register is not updated and contains indeterminate data.

18.3.3.1 DMA Setup

Set up DMA channel 1 with the source address from which the memory area to be verified are mapped. Set up the destination address to the memory mapped channel 1 PSA Signature Register. Put the starting address at post increment addressing mode and put the destination address at constant address mode. Use hardware DMA request to trigger an entire block transfer for channel 1. Disable autoinitiation for DMA channel 1.

18.3.3.2 Timer Setup

The timer can be any general purpose timer which is capable of generating a time based DMA request.

Set up timer to generate DMA request associated with DMA channel 1. For example, an OS can set up the timer to generate a DMA request every 10ms.

18.3.3.3 CRC Setup

- Program the pattern count to 128.
- Program the sector count to 2048.
- For example, we want the entire 2Mbytes to be compressed within 5ms. We can program the block complete timeout pre-load value to 15625 (5 ms / (1 HCLK period × 64)) if CRC is operating at 200 MHz.
- Enable Semi-CPU mode and enable all interrupts.

The timer module first generates a DMA request on DMA channel 1 when it is enabled. When the first incoming data pattern arrives at the PSA Signature Register, the CRC controller will compress it. After one sector of data patterns are compressed, the CRC controller generate a compression complete interrupt. Upon responding to the interrupt the CPU would read from the PSA Sector Signature Register. It is up to the CPU on how to deal with the PSA value just read. It can compare it to a known signature value or it can write it to another memory location to build a signature file or even transfer the signature out of the device via SCI or SPI. This routine will continue until the entire 2Mbyte are consumed. The latency of the interrupt response from CPU can cause overrun condition. If CPU does not read from PSA Sector Signature Register before the PSA value is overridden with the signature of the next sector of memory, an overrun interrupt will be generated by CRC controller.

18.3.4 Example: Full-CPU Mode

In a system without the availability of DMA controller, the CRC routine can be operated by CPU provided the CPU has enough throughput. CPU needs to read from the memory area from which CRC is to be performed.

A memory area with 2Mbyte is to be checked with the help of the CPU. CRC verification is to be performed every 1K byte. In CPU mode, the CRC Value Register is not updated and contains indeterminate data.

18.3.4.1 CRC Setup

- All control registers can be left in their reset state. Only enable Full-CPU mode.

CPU itself reads from the memory and write the data to the PSA Signature Register inside CRC Controller. When the first incoming data pattern arrives at the PSA Signature Register, the CRC Controller will compress it. After ***2MBytes*** data patterns are compressed, CPU can read from the PSA Signature Register. It is up to the CPU on how to deal with the PSA signature value just read. It can compare it to a known signature value stored at another memory location.

18.4 CRC Control Registers

All registers are in word boundary. 64, 32, 16, and 8 bit write accesses are supported to all registers. The base address for the control registers is FE00 0000h for CRC1 and FB00 0000h for CRC2.

Table 18-4. CRC Control Registers

Offset	Acronym	Register Description	Section
0h	CRC_CTRL0	CRC Global Control Register	Section 18.4.1
8h	CRC_CTRL1	CRC Global Control Register 1	Section 18.4.2
10h	CRC_CTRL2	CRC Global Control Register 2	Section 18.4.3
18h	CRC_INTS	CRC Interrupt Enable Set Register	Section 18.4.4
20h	CRC_INTR	CRC Interrupt Enable Reset Register	Section 18.4.5
28h	CRC_STATUS	CRC Interrupt Status Register	Section 18.4.6
30h	CRC_INT_OFFS_ET_REG	CRC Interrupt Offset Register	Section 18.4.7
38h	CRC_BUSY	CRC Busy Register	Section 18.4.8
40h	CRC_PCOUNT_REG1	CRC Channel 1 Pattern Counter Preload Register	Section 18.4.9
44h	CRC_SCOUNT_REG1	CRC Channel 1 Sector Counter Preload Register	Section 18.4.10
48h	CRC_CURSEC_REG1	CRC Channel 1 Current Sector Register	Section 18.4.11
4Ch	CRC_WDTOPLD1	CRC Channel 1 Watchdog Timeout Preload Register	Section 18.4.12
50h	CRC_BCTOPLD1	CRC Channel 1 Block Complete Timeout Preload Register	Section 18.4.13
60h	PSA_SIGREGL1	Channel 1 PSA Signature Low Register	Section 18.4.14
64h	PSA_SIGREGH1	Channel 1 PSA Signature High Register	Section 18.4.15
68h	CRC_REGL1	Channel 1 CRC Value Low Register	Section 18.4.16
6Ch	CRC_REGH1	Channel 1 CRC Value High Register	Section 18.4.17
70h	PSA_SECSIGREGL1	Channel 1 PSA Sector Signature Low Register	Section 18.4.18
74h	PSA_SECSIGREGH1	Channel 1 PSA Sector Signature High Register	Section 18.4.19
78h	RAW_DATAREGL1	Channel 1 Raw Data Low Register	Section 18.4.20
7Ch	RAW_DATAREGH1	Channel 1 Raw Data High Register	Section 18.4.21
80h	CRC_PCOUNT_REG2	CRC Channel 2 Pattern Counter Preload Register	Section 18.4.22
84h	CRC_SCOUNT_REG2	CRC Channel 2 Sector Counter Preload Register	Section 18.4.23
88h	CRC_CURSEC_REG2	CRC Current Sector Register 2	Section 18.4.24
8Ch	CRC_WDTOPLD2	CRC Channel 2 Watchdog Timeout Preload Register A	Section 18.4.25
90h	CRC_BCTOPLD2	CRC Channel 2 Block Complete Timeout Preload Register B	Section 18.4.26
A0h	PSA_SIGREGL2	Channel 2 PSA Signature Low Register	Section 18.4.27
A4h	PSA_SIGREGH2	Channel 2 PSA Signature High Register	Section 18.4.28
A8h	CRC_REGL2	Channel 2 CRC Value Low Register	Section 18.4.29
ACh	CRC_REGH2	Channel 2 CRC Value High Register	Section 18.4.30
B0h	PSA_SECSIGREGL2	Channel 2 PSA Sector Signature Low Register	Section 18.4.31
B4h	PSA_SECSIGREGH2	Channel 2 PSA Sector Signature High Register	Section 18.4.32
B8h	RAW_DATAREGL2	Channel 2 Raw Data Low Register	Section 18.4.33
BCh	RAW_DATAREGH2	Channel 2 Raw Data High Register	Section 18.4.34

18.4.1 CRC Global Control Register 0 (CRC_CTRL0)

Figure 18-9. CRC Global Control Register 0 (CRC_CTRL0) [offset = 00h]

31 16
Reserved
R-0

15 9	8
Reserved	CH2_PSA_SWREST
R-0	R/W-0

7 1	0
Reserved	CH1_PSA_SWREST
R-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 18-5. CRC Global Control Register 0 (CRC_CTRL0) Field Descriptions

Bit	Field	Value	Description
31-9	Reserved	0	Reads return 0. Writes have no effect.
8	CH2_PSA_SWREST		Channel 2 PSA Software Reset. When set, the PSA Signature Register is reset to all zero. Software reset does not reset software reset bit itself. Therefore, CPU is required to clear this bit by writing a 0.
		0	PSA Signature Register is not reset.
		1	PSA Signature Register is reset.
7-1	Reserved	0	Reads return 0. Writes have no effect.
0	CH1_PSA_SWREST		Channel 1 PSA Software Reset. When set, the PSA Signature Register is reset to all zero. Software reset does not reset software reset bit itself. Therefore, CPU is required to clear this bit by writing a 0.
		0	PSA Signature Register is not reset.
		1	PSA Signature Register is reset.

18.4.2 CRC Global Control Register (CRC_CTRL1)

Figure 18-10. CRC Global Control Register 1 (CRC_CTRL1) [offset = 08h]

31 16
Reserved
R-0

15 1	0
Reserved	PWDN
R-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 18-6. CRC Global Control Register 1 (CRC_CTRL1) Field Descriptions

Bit	Field	Value	Description
31-1	Reserved	0	Reads return 0. Writes have no effect.
0	PWDN		Power Down. When set, CRC module is put in power-down mode.
		0	CRC is not in power-down mode.
		1	CRC is in power-down mode.

18.4.3 CRC Global Control Register 2 (CRC_CTRL2)

Figure 18-11. CRC Global Control Register 2 (CRC_CTRL2) [offset = 10h]

31–16
Reserved
R-0

15–10	9–8
Reserved	CH2_MODE
R-0	R/WP-0

7–2	1–0
Reserved	CH1_MODE
R-0	R/WP-0

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privileged mode only; -*n* = value after reset

Table 18-7. CRC Global Control Register 2 (CRC_CTRL2) Field Descriptions

Bit	Field	Value	Description
31-10	Reserved	0	Reads return 0. Writes have no effect.
9-8	CH2_MODE		Channel 2 Mode Selection.
		0	Data Capture mode. In this mode, the PSA Signature Register does not compress data when it is written. Any data written to PSA Signature Register is simply captured by PSA Signature Register without any compression. This mode can be used to plant seed value into the PSA register.
		1h	AUTO Mode
		2h	Semi-CPU Mode
		3h	Full-CPU Mode
7-2	Reserved	0	Reads return 0. Writes have no effect.
1-0	CH1_MODE		Channel 1 Mode Selection.
		0	Data Capture mode. In this mode, the PSA Signature Register does not compress data when it is written. Any data written to PSA Signature Register is simply captured by PSA Signature Register without any compression. This mode can be used to plant seed value into the PSA register.
		1h	AUTO Mode
		2h	Semi-CPU Mode
		3h	Full-CPU Mode

18.4.4 CRC Interrupt Enable Set Register (CRC_INTS)

Figure 18-12. CRC Interrupt Enable Set Register (CRC_INTS) [offset = 18h]

31-16
Reserved
R-0

15-13	12	11	10	9	8
Reserved	CH2_ TIMEOUTENS	CH2_ UNDERENS	CH2_ OVERENS	CH2_ CRCFAILENS	CH2_ CCITENS
R-0	R/WP-0	R/WP-0	R/WP-0	R/WP-0	R/WP-0

7-5	4	3	2	1	0
Reserved	CH1_ TIMEOUTENS	CH1_ UNDERENS	CH1_ OVERENS	CH1_ CRCFAILENS	CH1_ CCITENS
R-0	R/WP-0	R/WP-0	R/WP-0	R/WP-0	R/WP-0

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privileged mode only; -*n* = value after reset

Table 18-8. CRC Interrupt Enable Set Register (CRC_INTS) Field Descriptions

Bit	Field	Value	Description
31-13	Reserved	0	Reads return 0. Writes have no effect.
12	CH2_TIMEOUTENS		Channel 2 Timeout Interrupt Enable Bit.
			User and Privileged mode (read):
		0	Timeout Interrupt is disabled.
		1	Timeout Interrupt is enabled.
			Privileged mode (write):
		0	No effect.
		1	Timeout Interrupt is enabled.
11	CH2_UNDERENS		Channel 2 Underrun Interrupt Enable Bit.
			User and Privileged mode (read):
		0	Underrun Interrupt is disabled.
		1	Underrun Interrupt is enabled.
			Privileged mode (write):
		0	No effect.
		1	Underrun Interrupt is enabled.
10	CH2_OVERENS		Channel 2 Overrun Interrupt Enable Bit.
			User and Privileged mode (read):
		0	Overrun Interrupt is disabled.
		1	Overrun Interrupt is enabled.
			Privileged mode (write):
		0	No effect.
		1	Overrun Interrupt is enabled.
9	CH2_CRCFAILENS		Channel 2 CRC Compare Fail Interrupt Enable Bit.
			User and Privileged mode (read):
		0	CRC Fail Interrupt is disabled.
		1	CRC Fail Interrupt is enabled.
			Privileged mode (write):
		0	No effect.
		1	CRC Fail Interrupt is enabled.

Table 18-8. CRC Interrupt Enable Set Register (CRC_INTS) Field Descriptions (continued)

Bit	Field	Value	Description
8	CH2_CCITENS		Channel 2 Compression Complete Interrupt Enable Bit.
			User and Privileged mode (read):
		0	Compression Complete Interrupt is disabled.
		1	Compression Complete Interrupt is enabled.
			Privileged mode (write):
		0	No effect.
		1	Compression Complete Interrupt is enabled.
7-5	Reserved	0	Reads return 0. Writes have no effect.
4	CH1_TIMEOUTENS		Channel 1 Timeout Interrupt Enable Bit.
			User and Privileged mode (read):
		0	Timeout Interrupt is disabled.
		1	Timeout Interrupt is enabled.
			Privileged mode (write):
		0	No effect.
		1	Timeout Interrupt is enabled.
3	CH1_UNDERENS		Channel 1 Underrun Interrupt Enable Bit.
			User and Privileged mode (read):
		0	Underrun Interrupt is disabled.
		1	Underrun Interrupt is enabled.
			Privileged mode (write):
		0	No effect.
		1	Underrun Interrupt is enabled.
2	CH1_OVERENS		Channel 1 Overrun Interrupt Enable Bit.
			User and Privileged mode (read):
		0	Overrun Interrupt is disabled.
		1	Overrun Interrupt is enabled.
			Privileged mode (write):
		0	No effect.
		1	Overrun Interrupt is enabled.
1	CH1_CRCFAILENS		Channel 1 CRC Compare Fail Interrupt Enable Bit.
			User and Privileged mode (read):
		0	CRC Fail Interrupt is disabled.
		1	CRC Fail Interrupt is enabled.
			Privileged mode (write):
		0	No effect.
		1	CRC Fail Interrupt is enabled.
0	CH1_CCITENS		Channel 1 Compression Complete Interrupt Enable Bit.
			User and Privileged mode (read):
		0	Compression Complete Interrupt is disabled.
		1	Compression Complete Interrupt is enabled.
			Privileged mode (write):
		0	No effect.
		1	Compression Complete Interrupt is enabled.

18.4.5 CRC Interrupt Enable Reset Register (CRC_INTR)

Figure 18-13. CRC Interrupt Enable Reset Register (CRC_INTR) [offset = 20h]

31 – 16
Reserved
R-0

15 – 13	12	11	10	9	8
Reserved	CH2_ TIMEOUTENR	CH2_ UNDERENR	CH2_ OVERENR	CH2_ CRCFAILENR	CH2_ CCITENR
R-0	R/WP-0	R/WP-0	R/WP-0	R/WP-0	R/WP-0

7 – 5	4	3	2	1	0
Reserved	CH1_ TIMEOUTENR	CH1_ UNDERENR	CH1_ OVERENR	CH1_ CRCFAILENR	CH1_ CCITENR
R-0	R/WP-0	R/WP-0	R/WP-0	R/WP-0	R/WP-0

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privileged mode only; -*n* = value after reset

Table 18-9. CRC Interrupt Enable Reset Register (CRC_INTR) Field Descriptions

Bit	Field	Value	Description
31-13	Reserved	0	Reads return 0. Writes have no effect.
12	CH2_TIMEOUTENR		Channel 2 Timeout Interrupt Enable Reset Bit.
			User and Privileged mode (read):
		0	Timeout Interrupt is disabled.
		1	Timeout Interrupt is enabled.
			Privileged mode (write):
		0	No effect.
		1	Timeout Interrupt is disabled.
11	CH2_UNDERENR		Channel 2 Underrun Interrupt Enable Reset Bit.
			User and Privileged mode (read):
		0	Underrun Interrupt is disabled.
		1	Underrun Interrupt is enabled.
			Privileged mode (write):
		0	No effect.
		1	Underrun Interrupt is disabled.
10	CH2_OVERENR		Channel 2 Overrun Interrupt Enable Reset Bit.
			User and Privileged mode (read):
		0	Overrun Interrupt is disabled.
		1	Overrun Interrupt is enabled.
			Privileged mode (write):
		0	No effect.
		1	Overrun Interrupt is disabled.
9	CH2_CRCFAILENR		Channel 2 CRC Compare Fail Interrupt Enable Reset Bit.
			User and Privileged mode (read):
		0	CRC Fail Interrupt disabled.
		1	CRC Fail Interrupt is enabled.
			Privileged mode (write):
		0	No effect.
		1	CRC Fail Interrupt is disabled.

Table 18-9. CRC Interrupt Enable Reset Register (CRC_INTR) Field Descriptions (continued)

Bit	Field	Value	Description
8	CH2_CCITENR		Channel 2 Compression Complete Interrupt Enable Reset Bit.
			User and Privileged mode (read):
		0	Compression Complete Interrupt is disabled.
		1	Compression Complete Interrupt is enabled.
			Privileged mode (write):
		0	No effect.
		1	Compression Complete Interrupt is disabled.
7-5	Reserved	0	Reads return 0. Writes have no effect.
4	CH1_TIMEOUTENR		Channel 1 Timeout Interrupt Enable Reset Bit.
			User and Privileged mode (read):
		0	Timeout Interrupt is disabled.
		1	Timeout Interrupt is enabled.
			Privileged mode (write):
		0	No effect.
		1	Timeout Interrupt is disabled.
3	CH1_UNDERENR		Channel 1 Underrun Interrupt Enable Reset Bit.
			User and Privileged mode (read):
		0	Underrun Interrupt is disabled.
		1	Underrun Interrupt is enabled.
			Privileged mode (write):
		0	No effect.
		1	Underrun Interrupt is disabled.
2	CH1_OVERENR		Channel 1 Overrun Interrupt Enable Reset Bit.
			User and Privileged mode (read):
		0	Overrun Interrupt is disabled.
		1	Overrun Interrupt is enabled.
			Privileged mode (write):
		0	No effect.
		1	Overrun Interrupt is disabled.
1	CH1_CRCFAILENR		Channel 1 CRC Compare Fail Interrupt Enable Reset Bit.
			User and Privileged mode (read):
		0	CRC Fail Interrupt is disabled.
		1	CRC Fail Interrupt is enabled.
			Privileged mode (write):
		0	No effect.
		1	CRC Fail Interrupt is disabled.
0	CH1_CCITENR		Channel 1 Compression Complete Interrupt Enable Reset Bit.
			User and Privileged mode (read):
		0	Compression Complete Interrupt is disabled.
		1	Compression Complete Interrupt is enabled.
			Privileged mode (write):
		0	No effect.
		1	Compression Complete Interrupt is disabled.

18.4.6 CRC Interrupt Status Register (CRC_STATUS)

Figure 18-14. CRC Interrupt Status Register (CRC_STATUS) [offset = 28h]

31	16
Reserved	
R-0	

15	13	12	11	10	9	8
Reserved		CH2_TIMEOUT	CH2_UNDER	CH2_OVER	CH2_CRCFAIL	CH2_CCIT
R-0		R/W1CP-0	R/W1CP-0	R/W1CP-0	R/W1CP-0	R/W1CP-0

7	5	4	3	2	1	0
Reserved		CH1_TIMEOUT	CH1_UNDER	CH1_OVER	CH1_CRCFAIL	CH1_CCIT
R-0		R/W1CP-0	R/W1CP-0	R/W1CP-0	R/W1CP-0	R/W1CP-0

LEGEND: R/W = Read/Write; R = Read only; W1CP = Write 1 to clear in privilege mode only; -*n* = value after reset

Table 18-10. CRC Interrupt Status Register (CRC_STATUS) Field Descriptions

Bit	Field	Value	Description
31-13	Reserved	0	Reads return 0. Writes have no effect.
12	CH2_TIMEOUT		Channel 2 CRC Timeout Interrupt Status Flag. This bit is set in both AUTO and Semi-CPU mode.
			User and Privileged mode (read):
		0	No timeout interrupt is active.
		1	Timeout interrupt is active.
			Privileged mode (write):
		0	No effect.
		1	Bit is cleared.
11	CH2_UNDER		Channel 2 CRC Underrun Interrupt Status Flag. This bit is set in AUTO mode only.
			User and Privileged mode (read):
		0	No Underrun Interrupt is active.
		1	Underrun Interrupt is active.
			Privileged mode (write):
		0	No effect.
		1	Bit is cleared.
10	CH2_OVER		Channel 2 CRC Overrun Interrupt Status Flag. This bit is set in either AUTO or Semi-CPU mode.
			User and Privileged mode (read):
		0	No Overrun Interrupt is active.
		1	Overrun Interrupt is active.
			Privileged mode (write):
		0	No effect.
		1	Bit is cleared.
9	CH2_CRCFAIL		Channel 2 CRC Compare Fail Interrupt Status Flag. This bit is set in AUTO mode only.
			User and Privileged mode (read):
		0	No CRC Fail Interrupt is active
		1	CRC Fail Interrupt is active
			Privileged mode (write):
		0	No effect
		1	Bit is cleared

Table 18-10. CRC Interrupt Status Register (CRC_STATUS) Field Descriptions (continued)

Bit	Field	Value	Description
8	CH2_CCIT		Channel 2 CRC Pattern Compression Complete Interrupt Status Flag. This bit is only set in Semi-CPU mode.
			User and Privileged mode (read):
		0	No Compression Complete Interrupt is active.
		1	Compression Complete Interrupt is active.
			Privileged mode (write):
		0	No effect.
		1	Bit is cleared.
7-5	Reserved	0	Reads return 0. Writes have no effect.
4	CH1_TIMEOUT		Channel 1 CRC Timeout Interrupt Status Flag.
			User and Privileged mode (read):
		0	No timeout interrupt is active.
		1	Timeout interrupt is active.
			Privileged mode (write):
		0	No effect.
		1	Bit is cleared.
3	CH1_UNDER		Channel 1 Underrun Interrupt Status Flag.
			User and Privileged mode (read):
		0	No Underrun Interrupt is active.
		1	Underrun Interrupt is active.
			Privileged mode (write):
		0	No effect.
		1	Bit is cleared.
2	CH1_OVER		Channel 1 Overrun Interrupt Status Flag.
			User and Privileged mode (read):
		0	No Overrun Interrupt is active.
		1	Overrun Interrupt is active.
			Privileged mode (write):
		0	No effect.
		1	Bit is cleared.
1	CH1_CRCFAIL		Channel 1 CRC Compare Fail Interrupt Status Flag.
			User and Privileged mode (read):
		0	No CRC Fail Interrupt is active.
		1	CRC Fail Interrupt is active.
			Privileged mode (write):
		0	No effect.
		1	Bit is cleared.
0	CH1_CCIT		Channel 1 CRC Pattern Compression Complete Interrupt Status Flag.
			User and Privileged mode (read):
		0	No Compression Complete Interrupt is active.
		1	Compression Complete Interrupt is active.
			Privileged mode (write):
		0	No effect.
		1	Bit is cleared.

18.4.7 CRC Interrupt Offset (CRC_INT_OFFSET_REG)

Figure 18-15. CRC Interrupt Offset (CRC_INT_OFFSET_REG) [offset = 30h]

31-16
Reserved
R-0

15-8	7-0
Reserved	OFSTREG
R-0	R-0

LEGEND: R = Read only; -*n* = value after reset

Table 18-11. CRC Interrupt Offset (CRC_INT_OFFSET_REG) Field Descriptions

Bit	Field	Value	Description
31-8	Reserved	0	Reads return 0. Writes have no effect.
7-0	OFSTREG		CRC Interrupt Offset. This register indicates the highest priority pending interrupt vector address. Reading the offset register automatically clears the respective interrupt flag.
		0	Phantom
		1h	Ch1 CRC Fail
		2h	Ch2 CRC Fail
		3h-8h	Reserved
		9h	Ch1 Compression Complete
		Ah	Ch2 Compression Complete
		Bh-10h	Reserved
		11h	Ch1 Overrun
		12h	Ch2 Overrun
		13h-18h	Reserved
		19h	Ch1 Underrun
		1Ah	Ch2 Underrun
		1Bh-20h	Reserved
		21h	Ch1 Timeout
		22h	Ch2 Timeout
		23h-FFh	Reserved

18.4.8 CRC Busy Register (CRC_BUSY)

Figure 18-16. CRC Busy Register (CRC_BUSY) [offset = 38h]

31 16
Reserved
R-0

15 9	8	7 1	0
Reserved	CH2_BUSY	Reserved	CH1_BUSY
R-0	R-0	R-0	R-0

LEGEND: R = Read only; -*n* = value after reset

Table 18-12. CRC Busy Register (CRC_BUSY) Field Descriptions

Bit	Field	Value	Description
31-9	Reserved	0	Reads return 0. Writes have no effect.
8	CH2_BUSY		CH2_BUSY. During AUTO or Semi-CPU mode, the busy flag is set when the first data pattern of the block is compressed and remains set until the last data pattern of the block is compressed. The flag is cleared when the last data pattern of the block is compressed.
7-1	Reserved	0	Reads return 0. Writes have no effect.
0	CH1_BUSY		CH1_BUSY. During AUTO or Semi-CPU mode, the busy flag is set when the first data pattern of the block is compressed and remains set until the last data pattern of the block is compressed. The flag is cleared when the last data pattern of the block is compressed.

18.4.9 CRC Pattern Counter Preload Register 1 (CRC_PCOUNT_REG1)

Figure 18-17. CRC Pattern Counter Preload Register 1 (CRC_PCOUNT_REG1) [offset = 40h]

31 20	19 16
Reserved	CRC_PAT_COUNT1
R-0	R/W-0

15 0
CRC_PAT_COUNT1
R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 18-13. CRC Pattern Counter Preload Register 1 (CRC_PCOUNT_REG1) Field Descriptions

Bit	Field	Value	Description
31-20	Reserved	0	Reads return 0. Writes have no effect.
19-0	CRC_PAT_COUNT1		Channel 1 Pattern Counter Preload Register. This register contains the number of data patterns in one sector to be compressed before a CRC is performed.

18.4.10 CRC Sector Counter Preload Register 1 (CRC_SCOUNT_REG1)

Figure 18-18. CRC Sector Counter Preload Register 1 (CRC_SCOUNT_REG1) [offset = 44h]

31	16
Reserved	
R-0	

15	0
CRC_SEC_COUNT1	
R/W-0	

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 18-14. CRC Sector Counter Preload Register 1 (CRC_SCOUNT_REG1) Field Descriptions

Bit	Field	Value	Description
31-16	Reserved	0	Reads return 0. Writes have no effect.
15-0	CRC_SEC_COUNT1		Channel 1 Sector Counter Preload Register. This register contains the number of sectors in one block of memory.

18.4.11 CRC Current Sector Register 1 (CRC_CURSEC_REG1)

Figure 18-19. CRC Current Sector Preload Register 1 (CRC_CURSEC_REG1) [offset = 48h]

31	16
Reserved	
R-0	

15	0
CRC_CURSEC1	
R-0	

LEGEND: R = Read only; -*n* = value after reset

Table 18-15. CRC Current Sector Register 1 (CRC_CURSEC_REG1) Field Descriptions

Bit	Field	Value	Description
31-16	Reserved	0	Reads return 0. Writes have no effect.
15-0	CRC_CURSEC1		Channel 1 Current Sector ID Register. In AUTO mode, this register contains the current sector number of which the signature verification fails. The sector counter is a free running up counter. When a sector fails, the erroneous sector number is logged into current sector ID register and the CRC fail interrupt is generated The sector ID register is frozen until it is read and the CRC fail status bit is cleared by CPU. While it is frozen, it does not capture another erroneous sector number. When this condition happens, an overrun interrupt is generated instead. Once the register is read and the CRC fail interrupt flag is cleared it can capture new erroneous sector number. In Semi-CPU mode, this register is used to indicate the sector number for which the compression complete has last happened.

18.4.12 CRC Channel 1 Watchdog Timeout Preload Register A (CRC_WDTOPLD1)

Figure 18-20. CRC Channel 1 Watchdog Timeout Preload Register A (CRC_WDTOPLD1) [offset = 4Ch]

31 24	23 16
Reserved	CRC_WDTOPLD1
R-0	R/W-0

15 0
CRC_WDTOPLD1
R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 18-16. CRC Channel 1 Watchdog Timeout Preload Register A (CRC_WDTOPLD1) Field Descriptions

Bit	Field	Value	Description
31-24	Reserved	0	Reads return 0. Writes have no effect.
23-0	CRC_WDTOPLD1		Channel 1 Watchdog Timeout Counter Preload Register. This register contains the number of clock cycles within which the DMA must transfer the next block of data patterns. In Semi-CPU mode, this register is used to indicate the sector number for which the compression complete has last happened.

18.4.13 CRC Channel 1 Block Complete Timeout Preload Register B (CRC_BCTOPLD1)

Figure 18-21. CRC Channel 1 Block Complete Timeout Preload Register B (CRC_BCTOPLD1) [offset = 50h]

31 24	23 16
Reserved	CRC_BCTOPLD1
R-0	R/W-0

15 0
CRC_BCTOPLD1
R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 18-17. CRC Channel 1 Block Complete Timeout Preload Register B (CRC_BCTOPLD1) Field Descriptions

Bit	Field	Value	Description
31-24	Reserved	0	Reads return 0. Writes have no effect.
23-0	CRC_BCTOPLD1		Channel 1 Block Complete Timeout Counter Preload Register. This register contains the number of clock cycles within which the CRC for an entire block needs to complete before a timeout interrupt is generated.

18.4.14 Channel 1 PSA Signature Low Register (PSA_SIGREGL1)

Figure 18-22. Channel 1 PSA Signature Low Register (PSA_SIGREGL1) [offset = 60h]

31	0
PSASIG1	
R/W-0	

LEGEND: R/W = Read/Write; -*n* = value after reset

Table 18-18. Channel 1 PSA Signature Low Register (PSA_SIGREGL1) Field Descriptions

Bit	Field	Description
31-0	PSASIG1	Channel 1 PSA Signature Low Register. This register contains the value stored at PSASIG1[31:0] register.

18.4.15 Channel 1 PSA Signature High Register (PSA_SIGREGH1)

Figure 18-23. Channel 1 PSA Signature High Register (PSA_SIGREGH1) [offset = 64h]

31	0
PSASIG1	
R/W-0	

LEGEND: R/W = Read/Write; -*n* = value after reset

Table 18-19. Channel 1 PSA Signature High Register (PSA_SIGREGH1) Field Descriptions

Bit	Field	Description
31-0	PSASIG1	Channel 1 PSA Signature High Register. This register contains the value stored at PSASIG1[63:32] register.

18.4.16 Channel 1 CRC Value Low Register (CRC_REGL1)

Figure 18-24. Channel 1 CRC Value Low Register (CRC_REGL1) [offset = 68h]

31	0
CRC1	
R/W-0	

LEGEND: R/W = Read/Write; -*n* = value after reset

Table 18-20. Channel 1 CRC Value Low Register (CRC_REGL1) Field Descriptions

Bit	Field	Description
31-0	CRC1	Channel 1 CRC Value Low Register. This register contains the current known good signature value stored at CRC1[31:0] register.

18.4.17 Channel 1 CRC Value High Register (CRC_REGH1)

Figure 18-25. Channel 1 CRC Value High Register (CRC_REGH1) [offset = 6Ch]

31 … 0
CRC1
R/W-0

LEGEND: R/W = Read/Write; -*n* = value after reset

Table 18-21. Channel 1 CRC Value High Register (CRC_REGH1) Field Descriptions

Bit	Field	Description
31-0	CRC1	Channel 1 CRC Value Low Register. This register contains the current known good signature value stored at CRC1[63:32] register.

18.4.18 Channel 1 PSA Sector Signature Low Register (PSA_SECSIGREGL1)

Figure 18-26. Channel 1 PSA Sector Signature Low Register (PSA_SECSIGREGL1) [offset = 70h]

31 … 0
PSASECSIG1
R-0

LEGEND: R = Read only; -*n* = value after reset

Table 18-22. Channel 1 PSA Sector Signature Low Register (PSA_SECSIGREGL1) Field Descriptions

Bit	Field	Description
31-0	PSASECSIG1	Channel 1 PSA Sector Signature Low Register. This register contains the value stored at PSASECSIG1[31:0] register.

18.4.19 Channel 1 PSA Sector Signature High Register (PSA_SECSIGREGH1)

Figure 18-27. Channel 1 PSA Sector Signature High Register (PSA_SECSIGREGH1) [offset = 74h]

31 … 0
PSASECSIG1
R-0

LEGEND: R = Read only; -*n* = value after reset

Table 18-23. Channel 1 PSA Sector Signature High Register (PSA_SECSIGREGH1) Field Descriptions

Bit	Field	Description
31-0	PSASECSIG1	Channel 1 PSA Sector Signature High Register. This register contains the value stored at PSASECSIG1[63:32] register.

18.4.20 Channel 1 Raw Data Low Register (RAW_DATAREGL1)

Figure 18-28. Channel 1 Raw Data Low Register (RAW_DATAREGL1) [offset = 78h]

31 0
RAW_DATA1
R-0

LEGEND: R = Read only; -*n* = value after reset

Table 18-24. Channel 1 Raw Data Low Register (RAW_DATAREGL1) Field Descriptions

Bit	Field	Description
31-0	RAW_DATA1	Channel 1 Raw Data Low Register. This register contains bits 31:0 of the uncompressed raw data.

18.4.21 Channel 1 Raw Data High Register (RAW_DATAREGH1)

Figure 18-29. Channel 1 Raw Data High Register (RAW_DATAREGH1) [offset = 7Ch]

31 0
RAW_DATA1
R-0

LEGEND: R = Read only; -*n* = value after reset

Table 18-25. Channel 1 Raw Data High Register (RAW_DATAREGH1) Field Descriptions

Bit	Field	Description
31-0	RAW_DATA1	Channel 1 Raw Data High Register. This register contains bits 63:32 of the uncompressed raw data.

18.4.22 CRC Pattern Counter Preload Register 2 (CRC_PCOUNT_REG2)

Figure 18-30. CRC Pattern Counter Preload Register 2 (CRC_PCOUNT_REG2) [offset = 80h]

31 18	19 16
Reserved	CRC_PAT_COUNT2
R-0	R/W-0

15 0
CRC_PAT_COUNT2
R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 18-26. CRC Pattern Counter Preload Register 2 (CRC_PCOUNT_REG2) Field Descriptions

Bit	Field	Value	Description
31-20	Reserved	0	Reads return 0. Writes have no effect.
19-0	CRC_PAT_COUNT2		Channel 2 Pattern Counter Preload Register. This register contains the number of data patterns in one sector to be compressed before a CRC is performed.

18.4.23 CRC Sector Counter Preload Register 2 (CRC_SCOUNT_REG2)

Figure 18-31. CRC Sector Counter Preload Register 2 (CRC_SCOUNT_REG2) [offset = 84h]

31	16
Reserved	
R-0	

15	0
CRC_SEC_COUNT2	
R/W-0	

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 18-27. CRC Sector Counter Preload Register 2 (CRC_SCOUNT_REG2) Field Descriptions

Bit	Field	Value	Description
31-16	Reserved	0	Reads return 0. Writes have no effect.
15-0	CRC_SEC_COUNT2		Channel 2 Sector Counter Preload Register. This register contains the number of sectors in one block of memory.

18.4.24 CRC Current Sector Register 2 (CRC_CURSEC_REG2)

Figure 18-32. CRC Current Sector Register 2 (CRC_CURSEC_REG2) [offset = 88h]

31	16
Reserved	
R-0	

15	0
CRC_CURSEC2	
R-0	

LEGEND: R = Read only; -*n* = value after reset

Table 18-28. CRC Current Sector Register 2 (CRC_CURSEC_REG2) Field Descriptions

Bit	Field	Value	Description
31-16	Reserved	0	Reads return 0. Writes have no effect.
15-0	CRC_CURSEC2		Channel 2 Current Sector ID Register. In AUTO mode, this register contains the current sector number of which the signature verification fails. The sector counter is a free running up counter. When a sector fails, the erroneous sector number is logged into current sector ID register and the CRC fail interrupt is generated The sector ID register is frozen until it is read and the CRC fail status bit is cleared by CPU. While it is frozen, it does not capture another erroneous sector number. When this condition happens, an overrun interrupt is generated instead. Once the register is read and the CRC fail interrupt flag is cleared it can capture new erroneous sector number. In Semi-CPU mode, this register is used to indicate the sector number for which the compression complete has last happened.

18.4.25 CRC Channel 2 Watchdog Timeout Preload Register A (CRC_WDTOPLD2)

Figure 18-33. CRC Channel 2 Watchdog Timeout Preload Register A (CRC_WDTOPLD2) [offset = 8Ch]

31 24	23 16
Reserved	CRC_WDTOPLD2
R-0	R/W-0

15 0
CRC_WDTOPLD2
R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 18-29. CRC Channel 2 Watchdog Timeout Preload Register A (CRC_WDTOPLD2) Field Descriptions

Bit	Field	Value	Description
31-24	Reserved	0	Reads return 0. Writes have no effect.
23-0	CRC_WDTOPLD2		Channel 2 Watchdog Timeout Counter Preload Register. This register contains the number of clock cycles within which the DMA must transfer the next block of data patterns. In Semi-CPU mode, this register is used to indicate the sector number for which the compression complete has last happened.

18.4.26 CRC Channel 2 Block Complete Timeout Preload Register B (CRC_BCTOPLD2)

Figure 18-34. CRC Channel 2 Block Complete Timeout Preload Register B (CRC_BCTOPLD2) [offset = 90h]

31 24	23 16
Reserved	CRC_BCTOPLD2
R-0	R/W-0

15 0
CRC_BCTOPLD2
R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 18-30. CRC Channel 2 Block Complete Timeout Preload Register B (CRC_BCTOPLD2) Field Descriptions

Bit	Field	Value	Description
31-24	Reserved	0	Reads return 0. Writes have no effect.
23-0	CRC_BCTOPLD2		Channel 2 Block Complete Timeout Counter Preload Register. This register contains the number of clock cycles within which the CRC for an entire block needs to complete before a timeout interrupt is generated.

18.4.27 Channel 2 PSA Signature Low Register (PSA_SIGREGL2)

Figure 18-35. Channel 2 PSA Signature Low Register (PSA_SIGREGL2) [offset = A0h]

31	0
PSASIG2	
R/W-0	

LEGEND: R/W = Read/Write; -*n* = value after reset

Table 18-31. Channel 2 PSA Signature Low Register (PSA_SIGREGL2) Field Descriptions

Bit	Field	Description
31-0	PSASIG2	Channel 2 PSA Signature Low Register. This register contains the value stored at PSASIG2[31:0] register.

18.4.28 Channel 2 PSA Signature High Register (PSA_SIGREGH2)

Figure 18-36. Channel 2 PSA Signature High Register (PSA_SIGREGH2) [offset = A4h]

31	0
PSASIG2	
R/W-0	

LEGEND: R/W = Read/Write; -*n* = value after reset

Table 18-32. Channel 2 PSA Signature High Register (PSA_SIGREGH2) Field Descriptions

Bit	Field	Description
31-0	PSASIG2	Channel 2 PSA Signature High Register. This register contains the value stored at PSASIG2[63:32] register.

18.4.29 Channel 2 CRC Value Low Register (CRC_REGL2)

Figure 18-37. Channel 2 CRC Value Low Register (CRC_REGL2) [offset = A8h]

31	0
CRC2	
R/W-0	

LEGEND: R/W = Read/Write; -*n* = value after reset

Table 18-33. Channel 2 CRC Value Low Register (CRC_REGL2) Field Descriptions

Bit	Field	Description
31-0	CRC2	Channel 2 CRC Value Low Register. This register contains the current known good signature value stored at CRC2[31:0] register.

18.4.30 Channel 2 CRC Value High Register (CRC_REGH2)

Figure 18-38. Channel 2 CRC Value High Register (CRC_REGH2) [offset = ACh]

31	0
CRC2	
R/W-0	

LEGEND: R/W = Read/Write; -*n* = value after reset

Table 18-34. Channel 2 CRC Value High Register (CRC_REGH2) Field Descriptions

Bit	Field	Description
31-0	CRC2	Channel 2 CRC Value High Register. This register contains the current known good signature value stored at CRC2[63:32] register.

18.4.31 Channel 2 PSA Sector Signature Low Register (PSA_SECSIGREGL2)

Figure 18-39. Channel 2 PSA Sector Signature Low Register (PSA_SECSIGREGL2) [offset = B0h]

31	0
PSASECSIG2	
R-0	

LEGEND: R = Read only; -*n* = value after reset

Table 18-35. Channel 2 PSA Sector Signature Low Register (PSA_SECSIGREGL2) Field Descriptions

Bit	Field	Description
31-0	PSASECSIG2	Channel 2 PSA Sector Signature Low Register. This register contains the value stored at PSASECSIG2[31:0] register.

18.4.32 Channel 2 PSA Sector Signature High Register (PSA_SECSIGREGH2)

Figure 18-40. Channel 2 PSA Sector Signature High Register (PSA_SECSIGREGH2) [offset = B4h]

31	0
PSASECSIG2	
R-0	

LEGEND: R = Read only; -*n* = value after reset

Table 18-36. Channel 2 PSA Sector Signature High Register (PSA_SECSIGREGH2) Field Descriptions

Bit	Field	Description
31-0	PSASECSIG2	Channel 2 PSA Sector Signature High Register. This register contains the value stored at PSASECSIG2[63:32] register.

18.4.33 Channel 2 Raw Data Low Register (RAW_DATAREGL2)

Figure 18-41. Channel 2 Raw Data Low Register (RAW_DATAREGL2) [offset = B8h]

31 … 0
RAW_DATA2
R-0

LEGEND: R = Read only; -*n* = value after reset

Table 18-37. Channel 2 Raw Data Low Register (RAW_DATAREGL2) Field Descriptions

Bit	Field	Description
31-0	RAW_DATA2	Channel 2 Raw Data Low Register. This register contains bits 31:0 of the uncompressed raw data..

18.4.34 Channel 2 Raw Data High Register (RAW_DATAREGH2)

Figure 18-42. Channel 2 Raw Data High Register (RAW_DATAREGH2) [offset = BCh]

31 … 0
RAW_DATA2
R-0

LEGEND: R = Read only; -*n* = value after reset

Table 18-38. Channel 2 Raw Data High Register (RAW_DATAREGH2) Field Descriptions

Bit	Field	Description
31-0	RAW_DATA2	Channel 2 Raw Data High Register. This register contains bits 63:32 of the uncompressed raw data..

Vectored Interrupt Manager (VIM) Module

This chapter describes the behavior of the vectored interrupt manager (VIM) module of the device family.

19.1 Overview

The vectored interrupt manager (VIM) provides hardware assistance for prioritizing and controlling the many interrupt sources present on a device. Interrupts are caused by events outside of the normal flow of program execution. Normally, these events require a timely response from the central processing unit (CPU); therefore, when an interrupt occurs, the CPU switches execution from the normal program flow to an interrupt service routine (ISR).

The VIM module has the following features:

- Dual VIM for safety
- Supports 127 interrupt channels, in both register vectored interrupt and hardware vectored interrupt mode.
  - Provides IRQ vector directly to the CPU VIC port
  - Provides FIQ/IRQ vector through registers
  - Provides programmable priority and enable for interrupt request lines
- Provides a direct hardware dispatch mechanism for fastest IRQ dispatch.
- Provides two software dispatch mechanisms for backward compatibility with earlier generation of TI processors.
  - Index interrupt
  - Register vectored interrupt
- ECC (Error Code Correction) protected vector interrupt table against soft errors.

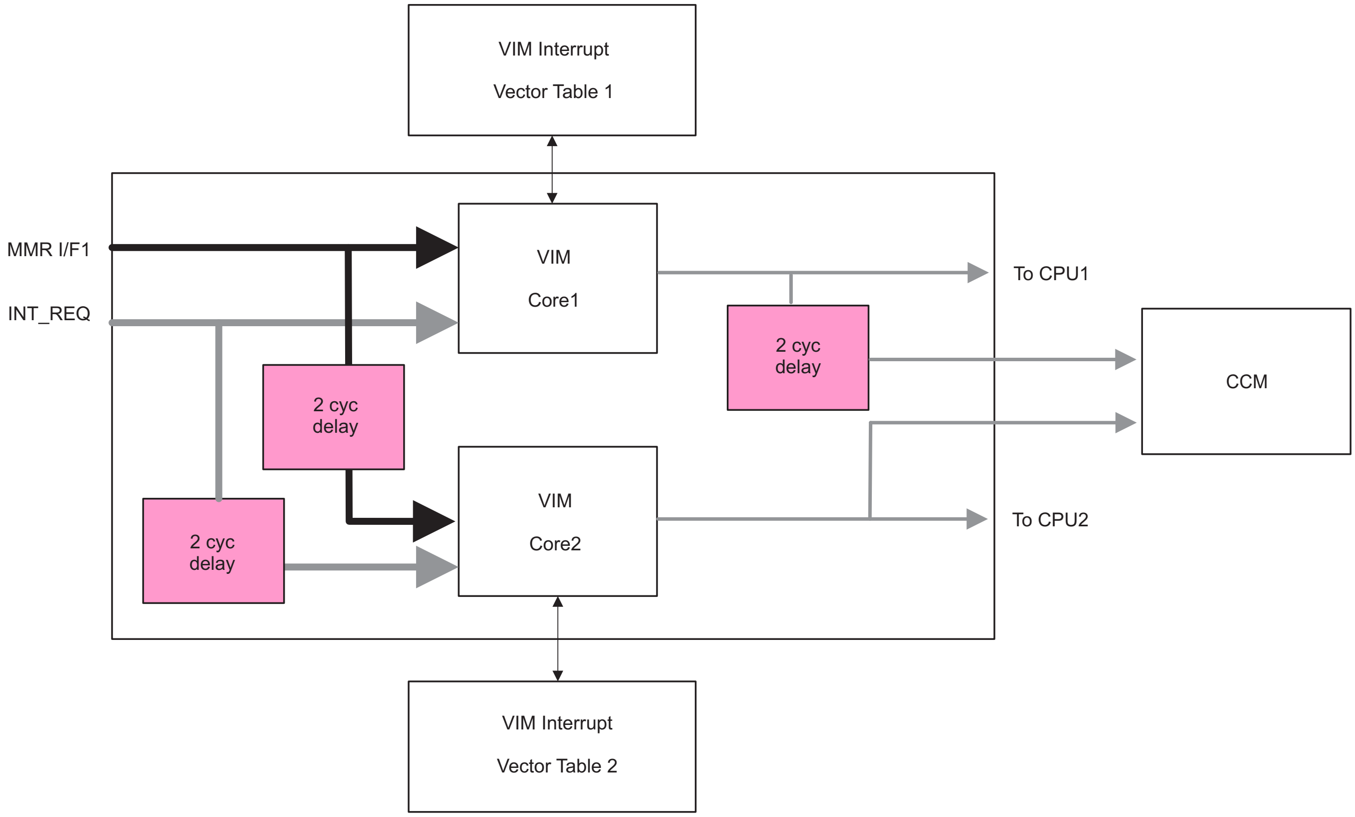
19.2 Dual VIM for Safety

A block diagram of Dual VIM for safety support is shown in Figure 19-1. To reduce probability of common cause failure, the VIM module mimics the dual CPU scheme of two cycle delayed operation of the two cores. In this case, the MMR (Memory Mapped Register) interface to the second instance is delayed by two cycles. Similarly, the interrupt inputs are also delayed by two cycles to the second instance.

A separate set of “2 cycle” delayed versions of output ports for the CPU interrupt interface of the VIM1 are provided. These will be used as one of the compare inputs of CPU Compare Module (CCM). The CPU interface signals of VIM2 are used as second set of inputs of CCM.

VIM2 uses the same address space as that of VIM1. During LockStep mode, any write to VIM1 (including the Interrupt Vector Table) will be routed to VIM2 as well so that the secondary instance is programmed exactly as the first one and provide compare diagnostic support. Auto initialization of the VIM1 Interrupt Vector Table will result in VIM2 Interrupt Vector Table getting initialized as well in this mode. In this mode, reads from VIM will return only VIM1 data.VIM2 registers and Interrupt Vector Table cannot be read out in Locked mode.

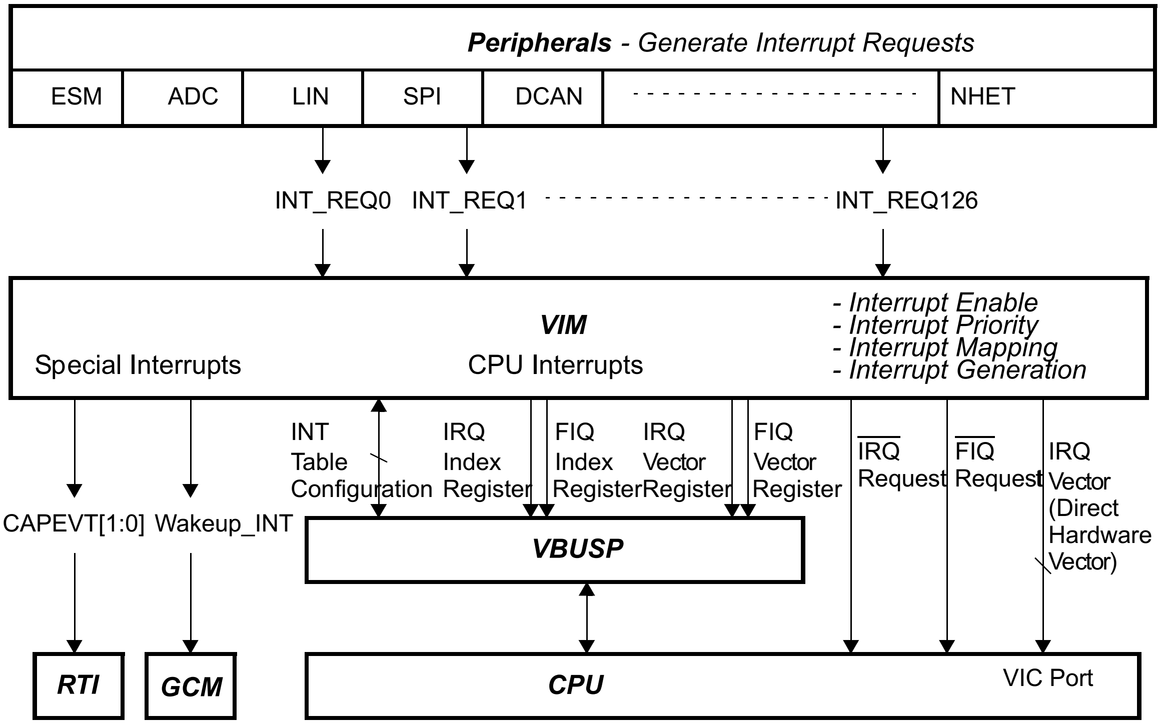
Figure 19-1. Block Diagram of Dual VIM for Safety Support



19.3 Device Level Interrupt Management

A block diagram of device level interrupt handling is shown in Figure 19-2. When an event occurs within a peripheral, the peripheral makes an interrupt request to the VIM. Then, VIM prioritizes the requests from peripherals and provides the address of the highest interrupt service routine (ISR) to the CPU. Finally, CPU starts executing the ISR instructions from that address in the ISR. Section 19.3.1 through Section 19.3.3 provide additional details about these three steps.

Figure 19-2. Device Level Interrupt Block Diagram



19.3.1 Interrupt Generation at the Peripheral

Interrupt generation begins when an event occurs within a peripheral module. Some examples of interrupt-capable events are expiration of a counter within a timer module, receipt of a character in a communications module, and completion of a conversion in an analog-to-digital converter (ADC) module. Some device peripherals are capable of requesting interrupts on more than one interrupt request line.

Interrupts are not always generated when an event occurs; the peripheral must make an interrupt request to the VIM based on the event occurrence. Typically, the peripheral contains:

- An interrupt flag bit for each event to signify the event occurrence.
- An interrupt enable bit to control whether the event occurrence causes an interrupt request to the VIM.

19.3.2 Interrupt Handling at the CPU

The ARM CPU provides two vectors for interrupt requests—fast interrupt requests (FIQs) and normal interrupt requests (IRQs). FIQs are higher priority than IRQs, and FIQ interrupts may interrupt IRQ interrupts.

NOTE: The FIQ implemented in Cortex-R4F/R5F is Non-Maskable Fast Interrupts (NMFI). Once FIQ is enabled (by clearing F bit in CPSR), it can NOT be disabled by setting F bit in CPSR. Only a reset or an FIQ will be able to set the F bit in CPSR. By hardware, Non Maskable FIQ are not reentrant.

After reset (power reset or warm reset), both FIQ and IRQ are disabled. The CPU may enable these interrupt request channels individually within the CPSR (Current Program Status Register); CPSR bits 6 and 7 must be cleared to enable the FIQ (bit 6) and IRQ (bit 7) interrupt requests at the CPU. CPSR is writable in privilege mode only. Example 19-2 shows how to enable the IRQ and FIQ through CPSR.

When the CPU receives an interrupt request, the CPSR mode field changes to either FIQ or IRQ mode. When an IRQ interrupt is received, the CPU disables other IRQ interrupts by setting CPSR bit 7. When an FIQ interrupt is received, the CPU disables both IRQ and FIQ interrupts by setting CPSR bits 6 and 7.

A write of 1 to CPSR bit 7 disables the IRQ from CPU. However, a write of 1 to CPSR bit 6 leaves it unchanged. Example 19-2 also shows how to disable the IRQ through CPSR.

19.3.3 Software Interrupt Handling Options

The device supports three different possibilities for software to handle interrupts

1. Index interrupts mode (compatible with TMS470R1x legacy code)

   After the interrupt is received by the CPU, the CPU branches to 0x18 (IRQ) or 0x1C (FIQ) to execute the main ISR. The main ISR routine reads the offset register (IRQINDEX, FIQINDEX) to determine the source of the interrupt.

   This mode is compatible with the TMS470R1x (CIM) module and provides the same interrupt registers.

   This mode could be used if legacy code needs to be reused, porting it from the TMS470R1x family. However, imported software will not benefit from the VIM improvements.

   To port legacy software, the interrupt vector at 0x18 (IRQ) or 0x1C (FIQ) only needs to be a branch statement to a software interrupt table. The software interrupt table reads the pending interrupt from a vector offset register (FIQINDEX[7:0] for FIQ interrupts and IRQINDEX[7:0] for IRQ interrupts). All pending interrupts can be viewed in the INTREQ register. Example 19-4 shows how to respond to FIQ with short latency in this mode.

2. Register vectored interrupts (automatically provide vector address to application)

   Before enabling interrupts, the application software also has to initiate the interrupt vector table (VIM RAM).

   Once the VIM receives an interrupt, it loads the address of ISR from interrupt vector table, and store it into the interrupt vector register (IRQVECREG for IRQ interrupt, FIQVECREG for FIQ interrupt).

   After the interrupt is received by the CPU, the CPU executes the instruction placed at 0x18 or 0x1C (IRQ or FIQ vector) to load the address of ISR (interrupt vector) from the interrupt vector register. Example 19-3 illustrates the configuration for the exception vectors using this mode.

3. Hardware vectored interrupts (automatically dispatch to ISR, IRQ only)

   Before enabling interrupts, the application software must initiate the interrupt vector table (VIM RAM) pointing to the ISR for each interrupt channel.

   After the interrupt (IRQ) is received by the CPU, CPU reads the address of ISR directly from the interface with VIM (VIC port) instead of branching to 0x18. The CPU will branch directly to the ISR.

   The hardware vectored interrupt behavior must be explicitly enabled by setting the vector enable (VE) bit in the CP15 R1 register. This bit resets to 0, so that the default state after reset is backward compatible to earlier ARM CPU. Example 19-1 shows how to enable the hardware vectored interrupt.

   NOTE: This mode is NOT available for FIQ.

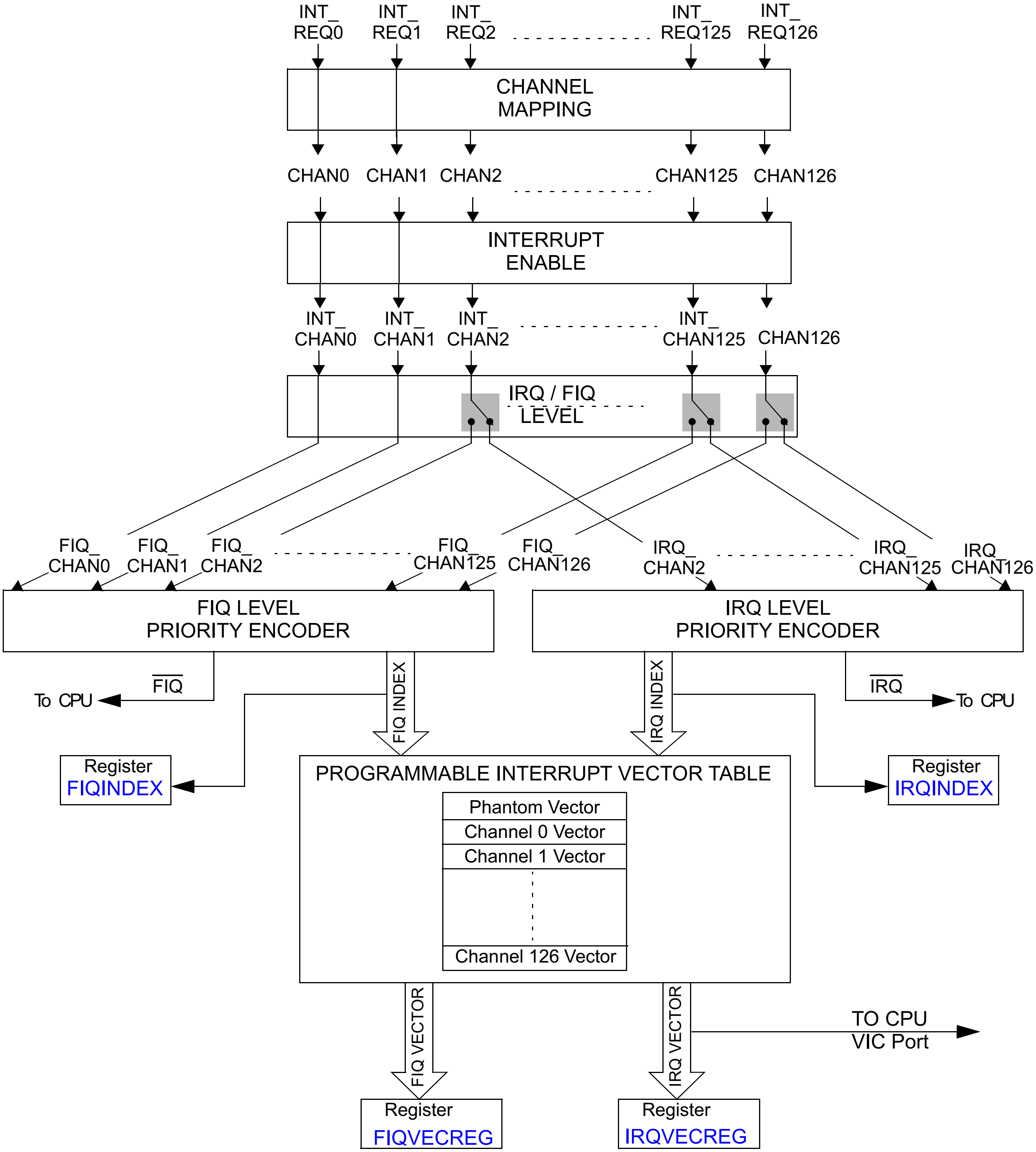
4. Software-Based Priority Decoding Scheme

   If the application uses a software-based interrupt priority decoding scheme instead of the hardware vector capabilities, then there is an additional step which was not required on earlier devices. This version of the VIM will hold an interrupt request generated by a peripheral. When the software clears the interrupt condition in the source module (for example, RTI, GIO, and so on), then it must also perform an additional clear of the interrupt request in the VIM. This can be done by reading the IRQVECREG register (Section 19.9.15) or FIQVECREG register (Section 19.9.16), or by writing a 1 to the INTREQ(i) bit (Section 19.9.10) in the VIM. This is not necessary if any of the three previous methods are used as the interrupt request bit in the VIM will be automatically cleared when the vector is read.

19.4 Interrupt Handling Inside VIM

A block diagram of the interrupt handling inside VIM is shown in Figure 19-3

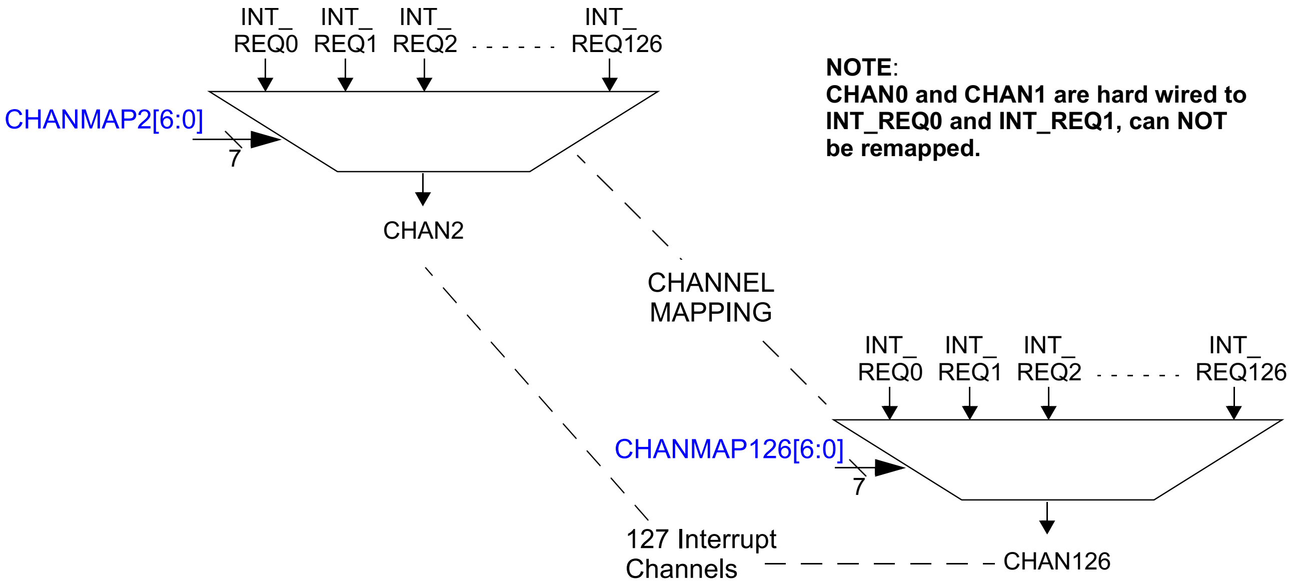
Figure 19-3. VIM Interrupt Handling Block Diagram



19.4.1 VIM Interrupt Channel Mapping

The VIM support 128 interrupt channels (including phantom interrupt). A block diagram of the VIM interrupt requests arrangement from peripheral modules to the interrupt channels is provided in Figure 19-4. Each interrupt channel (CHANx) has a corresponding mapping register bit field (CHANMAPx[6:0]). This mapping register determines which interrupt channel it maps each VIM interrupt request. With this scheme, the same request can be mapped to multiple channels. A lower numbered channel in each FIQ and IRQ has higher priority. The programmability of the VIM allows software to control the interrupt priority.

Figure 19-4. VIM Channel Mapping



NOTE:
CHAN0 and CHAN1 are hard wired to INT_REQ0 and INT_REQ1, can NOT be remapped.

NOTE: CHAN127

CHAN127 has no dedicated interrupt vector table entry. Therefore, CHAN127 shall NOT be remapped to other INT_REQ (INT_REQ127 is reserved at device level).

In the reset state, the VIM maps all of the interrupt requests in the system to their respective interrupt channels. Figure 19-5 shows the default state following the reset.

Figure 19-6 shows the VIM INT2 is remapped to both Channel 2 and 4, and INT3 is mapped to channel 3.

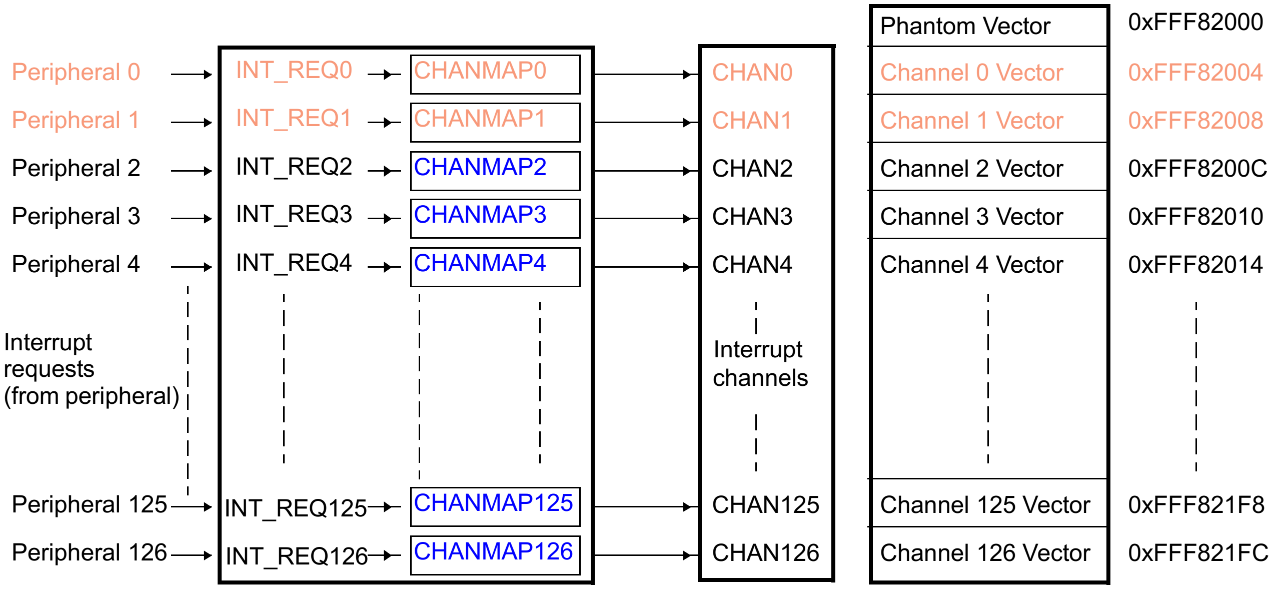
NOTE: By mapping INT2 to channel 2 and channel 4, and mapping INT3 to channel 3, it is possible for the software to change the priority dynamically by changing the ENABLE register (REQENASET and REQENACLR). When channel 2 is enabled, the priority is:

1. INT0
2. INT1
3. INT2
4. INT3

Disabling channel 2, the priority becomes:

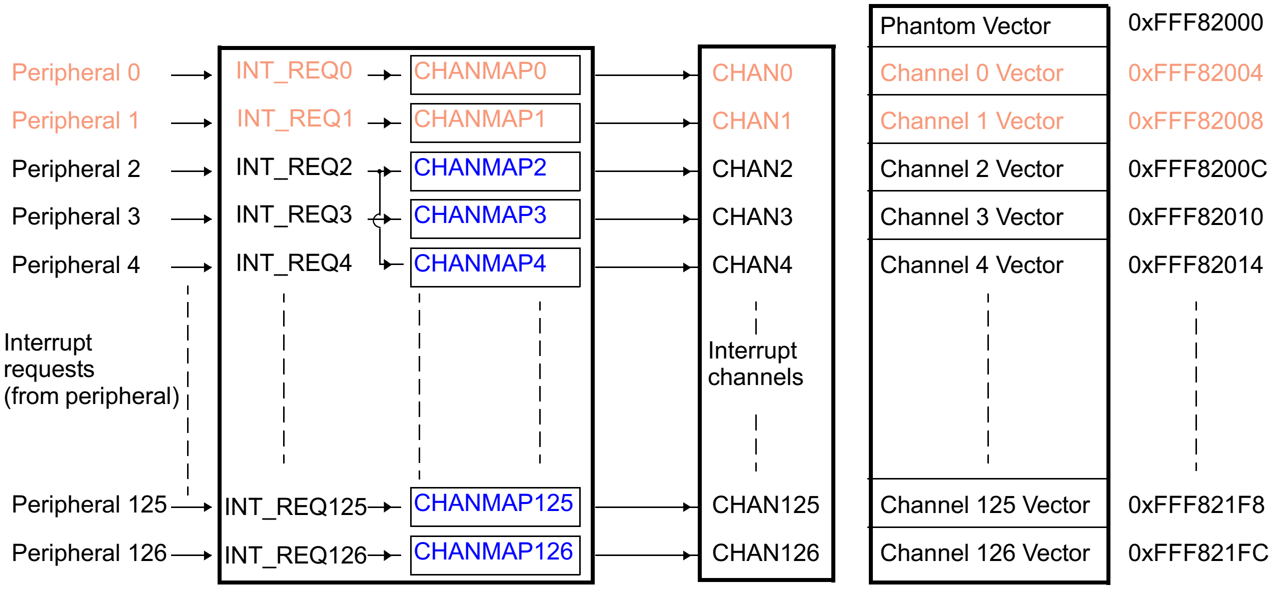
1. INT0
2. INT1
3. INT3
4. INT2

Figure 19-5. VIM in Default State



NOTE: CHAN0 and CHAN1 are hardwired to INT_REQ0 and INT_REQ1, so they cannot be remapped.

Figure 19-6. VIM in a Programmed State



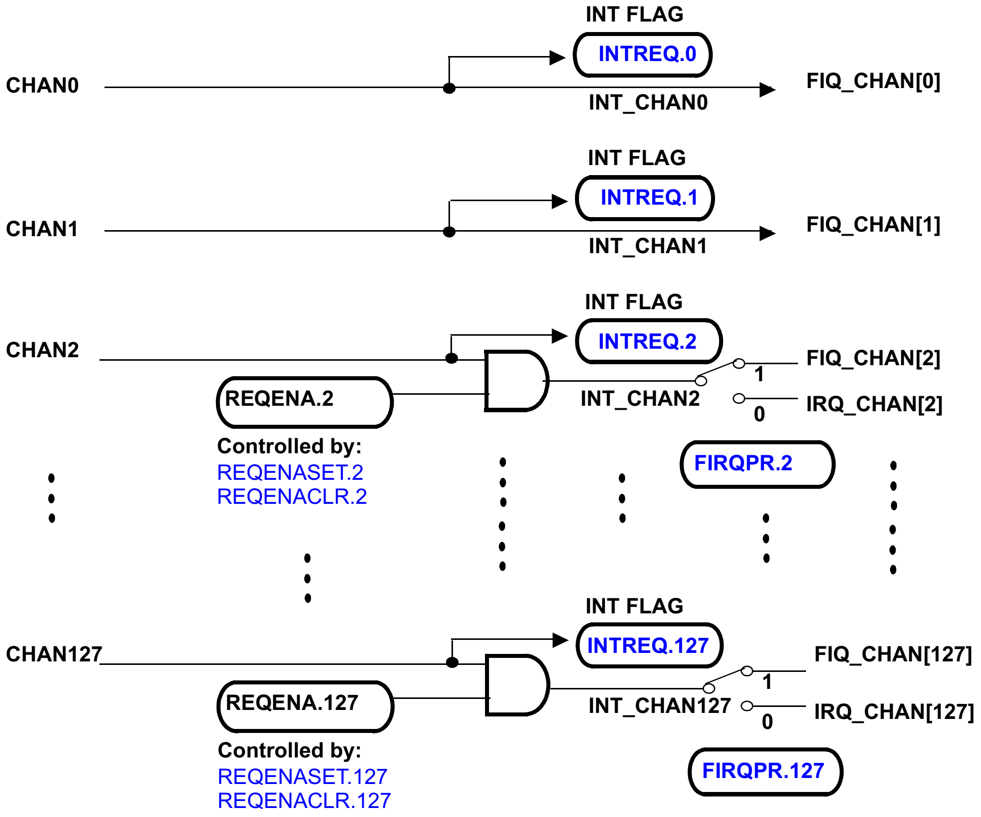
NOTE: CHAN0 and CHAN1 are hard wired to INT_REQ0 and INT_REQ1, so they cannot be remapped.

19.4.2 VIM Input Channel Management

As shown in Figure 19-7, the VIM enables channels on a channel-by-channel basis (in the REQENASET and REQENACLR registers); unused channels may be masked to prevent spurious interrupts.

NOTE: The interrupt ENABLE register does not affect the value of INTREQ.

Figure 19-7. Interrupt Channel Management



By default, interrupt CHAN0 is mapped to ESM (Error Signal Module) high level interrupt and CHAN1 is reserved for other NMI. For safety reasons, these two channels are mapped to FIQ only and can **NOT** be disabled through ENABLE registers.

NOTE: NMI Channel

Channel 0 and channel 1 are not maskable by the REQENASET / REQENACLR bit and both channel are routed exclusively to FIQ/NMI request line (FIRQPR0 and FIRQPR1 have no effect).

The VIM prioritizes the received interrupts based upon a programmed prioritization scheme. The VIM can send two interrupt requests to the CPU simultaneously—one IRQ and one FIQ. If both interrupt types are enabled at the CPU level, then the FIQ has greater priority and is handled first. Each interrupt channel, except channel 0 and 1, can be assigned to send either an FIQ or IRQ request to the CPU (in the FIRQPR register).

The VIM provides a default prioritization scheme, which sends the lowest numbered active channel (in each FIQ and IRQ classes) to the CPU. Within the FIQ and IRQ classes of interrupts, the lowest channel has the highest priority interrupt. The channel number is programmable through register CHANMAPx.

After the VIM has generated the vector corresponding to the highest active IRQ, it updates the FIQINDEX or the IRQINDEX register, depending on the class of interrupt. Then, it accesses the interrupt vector table using the vector value to fetch the address of the corresponding ISR. If the request is an FIQ class interrupt, the address read from the interrupt vector table, is written to the FIQVECREG register. If the request is an IRQ class interrupt, the address is written to the IRQVECREG register and put on the VIC port of the CPU (in case of hardware vectored interrupt is enabled).

All of the interrupt registers are updated when a new high priority interrupt line becomes active.

19.5 Interrupt Vector Table (VIM RAM)

Interrupt vector table stores the address of ISRs. During register vectored interrupt and hardware vectored interrupt, VIM accesses the interrupt vector table using the vector value to fetch the address of the corresponding ISR.

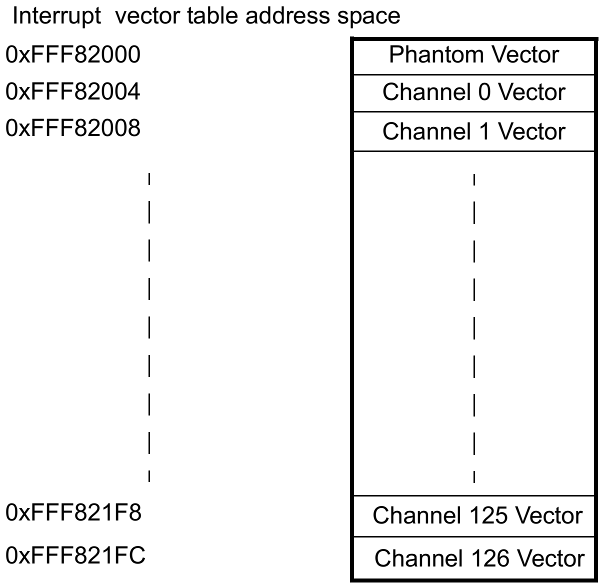
For safety reasons, the interrupt vector table has protection by ECC to indicate corruption due to soft errors. The ECC scheme is implemented as a continuous background check based on memory access. The ECC logic inside VIM supports Single-bit Error Correction and Double-bit Error Detection (SECDED). Section 19.5.1 through Section 19.5.4 describe how ECC works in the interrupt vector table.

NOTE: Writes to the interrupt vector table ECC status register (ECCSTAT) and the interrupt vector table ECC control register (ECCCTL) are in privilege mode only.

19.5.1 Interrupt Vector Table Operation

The interrupt vector table is organized in 128 words of 32 bits. Figure 19-8 shows the interrupt memory mapping. The table base address is 0xFFF82000.

Figure 19-8. VIM Interrupt Address Memory Map



NOTE: The interrupt vector table only has 128 entries, one phantom vector and 127 interrupt channels. Channel 127 does not have a dedicated vector and shall not be used.

There are seven bits of ECC per 32-bit ISR address. When a write is performed into the interrupt vector table, the ECC bits are calculated for the 32-bit word and written into the corresponding ECC region of interrupt vector table if ECC is enabled in VIM.

NOTE: Only 32-bit write/read access are allowed on interrupt vector table if ECC is required. Non 32-bit access might result in ECC errors.

When a read occurs from the CPU or VIM, the VIM calculates the ECC bits from the data coming from the interrupt vector table and compares it to the known good ECC value stored in the table. If a single-bit error is detected in the data, the SECDED block will automatically correct it. The read data will be a corrected one in this case. If double-bit errors are detected, the read data will be the uncorrected one. The access of the data and the ECC bits are performed in the same clock cycle.

The Double-Bit Error (DBE) and Single-Bit Error (SBE) events will be generated only if the ECC feature is enabled by ECCENA field. Correction of the data upon a SBE event will be done only if enabled EDAC_MODE field. Any double-bit error will be flagged out to ESM module and as UERR flag in ECCSTAT register. The address of the data for which UERR is detected will also be stored as UERRADDR register.

Any single-bit error will be registered into SBERR flag in ECCSTAT register and the corresponding address will be captured as SBERRADDR register. If SBE_INT_EN field on ECCCTL register is set to enable value, then it will be flagged out to ESM module.

Since the interrupt vector table may have an uncorrectable error (for example, DBE), the FBVECADDR register will provide to the VIC port, IRQVECREG and FIQVECREG, a fall-back address to an ISR that can restore the interrupt vector table content. The FB_VECADDR register should be set before initializing the interrupt in the interrupt vector table, to avoid branching to an unpredictable location.

The normal operation is restored when the ECCSTAT is cleared by the CPU. It is recommended to restore the content of the VIM before clearing the ECCSTAT.

19.5.2 VIM ECC Syndrome

The VIM ECC is controlled by the ECCENA bits of ECCCTL register. After reset, the SECDED feature is disabled. The SECDED feature can be enabled by writing 0xA (1010b) in the ECCENA[3:0] bit field of the ECCCTL register.

The ECC generation is done according to the ECC syndrome table as shown in Table 19-1 and Table 19-2. Each ECC bit is built by generating the parity of the XORed bits of the data word, whereas ECC bit 2 and 3 are even parity and the other bits odd parity.

Table 19-1. ECC Syndrome Table

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	ECC
x	x	x	x	x	x	x	x																	x	x	x	x	x	x	x	x	6
x	x	x	x	x	x	x	x									x	x	x	x	x	x	x	x									5
x	x							x	x	x	x	x	x			x	x							x	x	x	x	x	x			4
		x	x	x				x	x	x				x	x			x	x	x				x	x	x				x	x	3
x		x			x	x		x			x	x			x	x		x			x	x		x			x	x			x	2
			x		x		x		x		x		x	x	x				x		x		x		x		x		x	x	x	1
x		x	x		x					x		x	x	x			x			x		x	x	x	x		x				x	0

Table 19-2. ECC Error Bits for Syndrome Decode

6	5	4	3	2	1	0	ECC
x							6
	x						5
		x					4
			x				3
				x			2
					x		1
						x	0

19.5.3 Interrupt Vector Table Initialization

After reset, the interrupt vector table content, including the ECC bits is not initialized. Therefore, the interrupt vector table has to be initialized first before enabling the corresponding interrupt channel. This can be done either using the hardware initialization mechanism (in Chapter Architecture Overview) or it can be done by writing known values into the interrupt vector table by software. If ECC is required, this initialization should be done after the ECC functionality is enabled. In this way, the corresponding ECC bits will be automatically updated. This initialization is only required when vectored interrupts are used, index interrupt management does not need the table to be initialized.

19.5.4 Interrupt Vector Table ECC Testing

To test the ECC checking mechanism, the ECC bits allows manual insertion of faults. This option is implemented using the TEST_DIAG_EN bit in the ECCCTL register control bit. Once TEST_DIAG_EN is enabled, the ECC bits are mapped to 0xFFF82400. In this mode, the user can modify the ECC bits without changing the data bits. If ECCENA is disabled, writing to data bits does not automatically update ECC bits. The CPU reads and writes under different conditions are summarized in Table 19-3 and Table 19-4. After that, user can force faults into either the data or the ECC bits. Finally, the ECC error can be triggered by reading interrupt vector table (not ECC bits) from VIM or CPU. Please note that no ECC checking will be done for reads of ECC bits in test mode.

Table 19-3. CPU Reads - Address Bit 10 Selects Between Normal Data and ECC Bits

VBUSP_ ADDR(10)	TEST_DIAG_ EN	ECCENA	Action
0	x(don’t care)	x(don’t care)	Normal RAM location read
1	x	x	ECC bits read

Table 19-4. CPU Writes - Address Bit 10 Selects Between Normal Data and ECC Bits

VBUSP_ ADDR(10)	TEST_DIAG_ EN	ECCENA	Action
0	x	1	Normal RAM locations write with ECC bits
1	0	1	This write will be blocked
1	1	1	ECC bits write
0	x	0	Normal RAM locations write without ECC bits
1	0	0	This write will be blocked
1	1	0	This write is not allowed

The following sequence should be used for injecting faults to ECC bits and testing the ECC check feature.

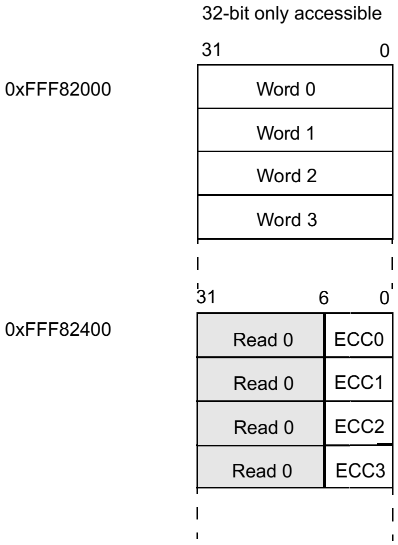
1. Write the data locations of VIM RAM with the required patterns while keeping ECCENA active. The ECC bits will be automatically initialized along with data bits.
2. Enable ECC test mode using TEST_DIAG_EN field of ECCCTRL register.
3. In this mode, it is possible to corrupt ECC bits using any of the following methods.
   - Read the ECC bits, flip one bit and write back
   - Read the ECC bits, flip 2 bits and write back
4. Depending on the kind corruption created, read back the data bits and check for the correction error (single-bit error or double-bit error or no error).
5. Read the UERRADDR and SBERRADDR registers and check for the correct address capture as well.

The following sequence should be used for injecting faults to data bits and testing the ECC check feature.

1. Write the data locations of VIM RAM with the required patterns while keeping ECCENA active. The ECC bits will be automatically initialized along with data bits.
2. Disable ECC by setting ECCENA=0 in ECCCTRL register. In this mode, writing to data bits does not automatically update ECC bits.
3. In this mode, it is possible to corrupt data bits using any of the following methods.
   - Read the data bits, flip one bit and write back
   - Read the data bits, flip 2 bits and write back
4. Depending on the kind corruption created, read back the data bits and check for the correction error (single-bit error or double-bit error or no error).
5. Read the UERRADDR and SBERRADDR registers and check for the correct address capture as well.

NOTE: After completing the tests for ECC check features, it should be ensured that VIM Interrupt Vector Table is initialized with valid data and corresponding check bits. Care should also be taken to clear the UERR and SBERR flag registers and the error address registers.

Figure 19-9. ECC Bits Mapping

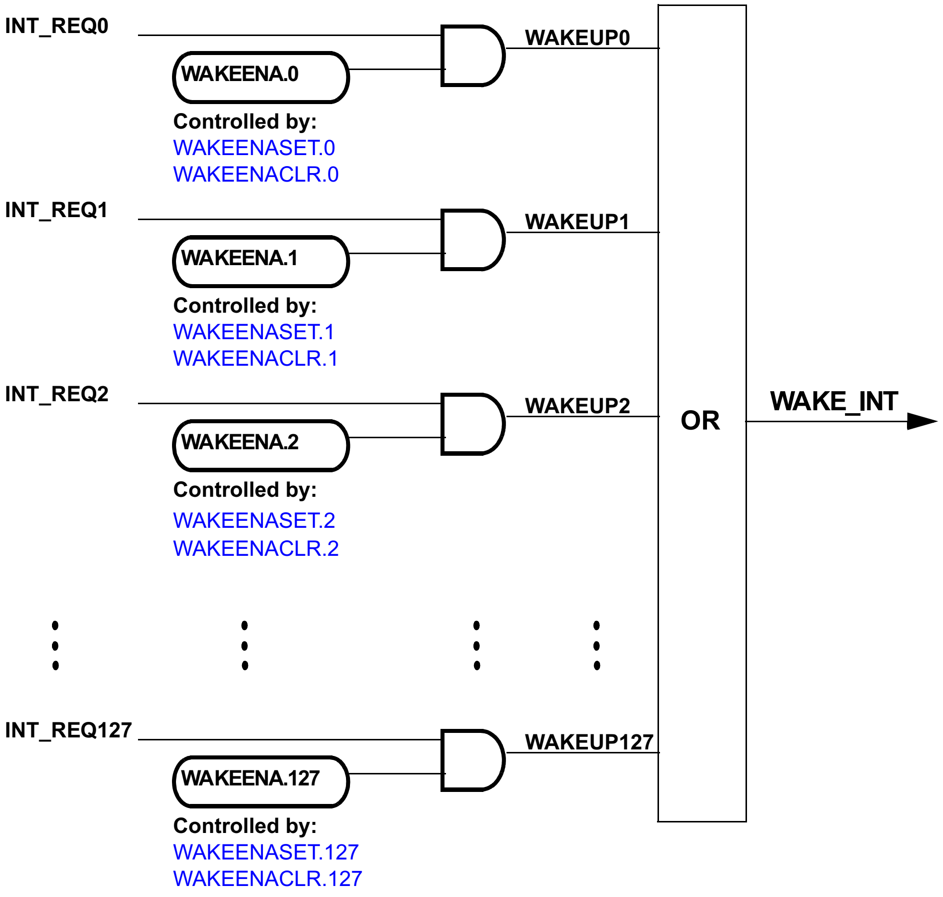


19.6 VIM Wakeup Interrupt

The wakeup interrupts are used to come out of low power mode (LPM). Any interrupt requests can be used to wake up the device. After reset, all interrupt requests are set to wake up from LPM. However, the VIM can mask unwanted interrupt lines for wake-up by using the WAKEENASET and WAKEENACLR register. The value in REQENASET / REQENACLR does NOT impact the wakeup interrupt.

As shown in Figure 19-10, the WAKEENASET and WAKEENACLR registers will enable/disable an interrupt for wake-up from low-power mode. All wake-up interrupts are “ORed” into a single signal WAKE_INT connected to the Global Clock Module.

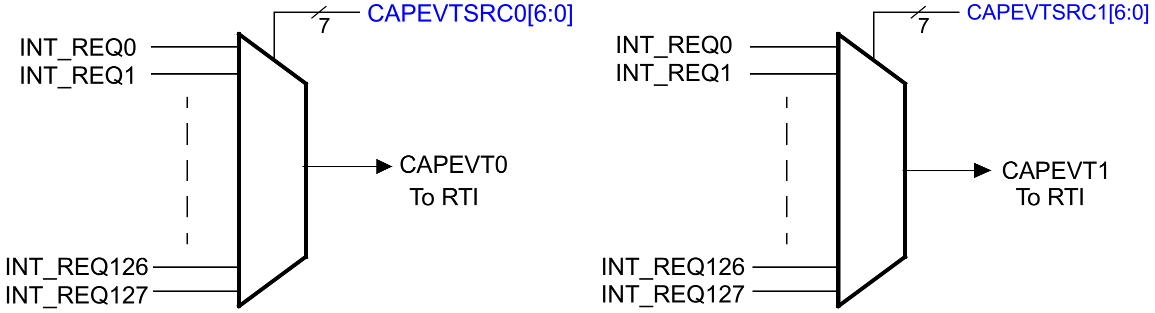
Figure 19-10. Detail of the IRQ Input



19.7 Capture Event Sources

The VIM can select any of the 128 interrupt request to generate up to two capture events for the real-time interrupt (RTI) module (see Figure 19-11). The value in REQENASET / REQENACLR does NOT impact the capture event. Two registers (Section 19.9.17) are available, one for each capture event source.

Figure 19-11. Capture Event Sources



19.8 Examples

The following sections provide examples about the operation of the VIM.

19.8.1 Examples - Configure CPU To Receive Interrupts

Example 19-1 shows how to set the vector enable (VE) bit in the CP15 R1 register to enable the hardware vector interrupt. Example 19-2 shows how to enable/disable the IRQ and FIQ through CPSR. As a convention, the program who calls these subroutines shall preserve register R1 if needed. Example 19-2 can ONLY run in privileged mode. However, in USER mode, the application software can force the program into software interrupt by instruction ‘SWI’. Then, in the software interrupt service routine, user can write register SPSR, which is the copy of CPSR in this exception mode.

Example 19-1. Enable Hardware Vector Interrupt (IRQ Only)

```
_HW_Vec_Init
    MRC p15 ,#0 ,R1 ,c1 ,c0 ,#0
    ORR R1 ,R1 ,#0x01000000       ; Mask 0-31 bits except bit 24 in Sys
                                  ; Ctrl Reg of CORTEX-R4
    MCR p15 ,#0 ,R1 ,c1 ,c0 ,#0   ; Enable bit 24
    MOV PC, LR
```

Example 19-2. Enable/Disable IRQ/FIQ through CPSR

```
FIQENABLE .equ 0x40
IRQENABLE .equ 0x80
......
_Enable_Fiq
    MRS R1, CPSR
    BIC R1, R1, #FIQENABLE
    MSR CPSR, R1
    MOV PC, LR
......
_Disable_Irq
    MRS R1, CPSR
    ORR R1, R1, #IRQENABLE
    MSR CPSR, R1
    MOV PC, LR
......
_Enable_Irq
    MRS R1, CPSR
    BIC R1, R1, #IRQENABLE
    MSR CPSR, R1
    MOV PC, LR
```

19.8.2 Examples - Register Vector Interrupt and Index Interrupt Handling

Example 19-3 illustrates the configuration for the exception vectors in Register Vector Interrupt handling. After the interrupt is received by the CPU, the CPU branches to 0x18 (IRQ) or 0x1C (FIQ). The instruction placed here should be *LDR PC, [PC,#-0x1B0]*. The pending ISR address is written into the corresponding vector register (IRQVECREG for IRQ, FIQVECREG for FIQ). The CPU reads the content of the register and branches to the ISR.

Example 19-3. Exception Vector Configuration for VIM Vector

```
          .sect ".intvecs"
00000000h b _RESET              ; RESET interrupt
00000004h b _UNDEF_INST_INT     ; UNDEFINED INSTRUCTION interrupt
00000008h b _SW_INT             ; SOFTWARE interrupt
0000000Ch b _ABORT_PREF_INT     ; ABORT (PREFETCH) interrupt
00000010h b _ABORT_DATA_INT     ; ABORT (DATA) interrupt
00000014h b #-8                 ; Reserved
00000018h ldr pc,[pc,#-0x1B0]   ; IRQ interrupt
0000001Ch ldr pc,[pc,#-0x1B0]   ; FIQ interrupt
```

NOTE: Program Counter (PC) always pointers two instructions beyond the current executed instruction. In this case, PC equals to ‘*0x18 or 0x1C + 0x08*’. The *LDR* instruction load the memory at ‘*PC - 0x1B0*’, which is ‘*0x18 or 0x1C + 0x08 - 0x1B0* = 0xFFFFFE70 or 0xFFFFFE74’. These are the address of IRQVECREG and FIQVECREG, which store the pending ISR address.

Example 19-4 shows a fast response to the FIQ interrupt in Index Interrupt and can be applied to a system that has more than one channel assigned as a FIQ. It is built in Index Interrupt compatible with TMS470R1x legacy code.

Example 19-4. How to Respond to FIQ With Short Latency

```
          .sect ".intvecs"         ; Interrupt and exception vector sector
00000000h b _RESET                 ; RESET interrupt
00000004h b _UNDEF_INST_INT        ; UNDEFINED INSTRUCTION interrupt
00000008h b _SW_INT                ; SOFTWARE interrupt
0000000Ch b _ABORT_PREF_INT        ; ABORT (PREFETCH) interrupt
00000010h b _ABORT_DATA_INT        ; ABORT (DATA) interrupt
00000014h b #-8                    ; Reserved
00000018h b _IRQ_ENTRY_0           ; IRQ interrupt
                                   ;*********************************
                                   ; INTERRUPT PROCESSING AREA
                                   ;*********************************
0000001Ch ldrb R8, [PC,#-0x21d]    ; FIQ INTERRUPT ENTRY
                                   ; R8 used to get the FIQ index
                                   ; with address pointer to the
                                   ; first FIQ banked register
00000020h ldr PC, [PC, R8, LSL#2]  ; Branch to the indexed interrupt
                                   ; routine. The prefetch
                                   ; operation causes the PC to be 2
                                   ; words (8 bytes) ahead of the
                                   ; current instruction, so
                                   ; pointing to _INT_TABLE.
00000024h nop                      ; Required due to pipeline.
                                   ;=================================
00000028h _INT_TABLE               ; FIQ INTERRUPT DISPATCH
                                   ;=================================
0000002Ch .word _FIQ_TABLE         ; beginning of FIQ Dispatch
00000030h .word _ISR1              ; dispatch to interrupt routine 1
00000034h .word _ISR2              ; dispatch to interrupt routine 2
          .
          .
```

Another way to improve the FIQ latency is to assign only one channel to the FIQ interrupt and to map the ISR code corresponding to this channel directly starting at 0x1C.

NOTE: When the CPU is in vector-enabled mode, Example 19-3 and Example 19-4 are still valid. The difference is that the CPU will not read from the 0x18 location during IRQ interrupt, but will jump directly to the corresponding ISR routine.

19.9 VIM Control Registers

Table 19-5 lists the VIM module registers. Each register begins on a word boundary. All registers are 32-bit, 16-bit, and 8-bit accessible for read and write. Write is only possible in privilege mode. The base address of the control registers is FFFF FE00h. The base address of the ECC-related VIM registers is FFFF FD00h. The address locations not listed are reserved.

Table 19-5. VIM Control Registers

Offset	Acronym	Register Description	Section
		ECC-related Registers	
ECh	ECCSTAT	Interrupt Vector Table ECC Status Register	Section 19.9.1
F0h	ECCCTL	Interrupt Vector Table ECC Control Register	Section 19.9.2
F4h	UERRADDR	Uncorrectable Error Address Register	Section 19.9.3
F8h	FBVECADDR	Fallback Vector Address Register	Section 19.9.4
FCh	SBERRADDR	Single-Bit Error Address Register	Section 19.9.5
		Control Registers	
00h	IRQINDEX	IRQ Index Offset Vector Register	Section 19.9.7
04h	FIQINDEX	FIQ Index Offset Vector Register	Section 19.9.8
10h	FIRQPR0	FIQ/IRQ Program Control Register 0	Section 19.9.9
14h	FIRQPR1	FIQ/IRQ Program Control Register 1	Section 19.9.9
18h	FIRQPR2	FIQ/IRQ Program Control Register 2	Section 19.9.9
1Ch	FIRQPR3	FIQ/IRQ Program Control Register 3	Section 19.9.9
20h	INTREQ0	Pending Interrupt Read Location Register 0	Section 19.9.10
24h	INTREQ1	Pending Interrupt Read Location Register 1	Section 19.9.10
28h	INTREQ2	Pending Interrupt Read Location Register 2	Section 19.9.10
2Ch	INTREQ3	Pending Interrupt Read Location Register 3	Section 19.9.10
30h	REQENASET0	Interrupt Enable Set Register 0	Section 19.9.11
34h	REQENASET1	Interrupt Enable Set Register 1	Section 19.9.11
38h	REQENASET2	Interrupt Enable Set Register 2	Section 19.9.11
3Ch	REQENASET3	Interrupt Enable Set Register 3	Section 19.9.11
40h	REQENACLR0	Interrupt Enable Clear Register 0	Section 19.9.12
44h	REQENACLR1	Interrupt Enable Clear Register 1	Section 19.9.12
48h	REQENACLR2	Interrupt Enable Clear Register 2	Section 19.9.12
4Ch	REQENACLR3	Interrupt Enable Clear Register 3	Section 19.9.12
50h	WAKEENASET0	Wake-up Enable Set Register 0	Section 19.9.13
54h	WAKEENASET1	Wake-up Enable Set Register 1	Section 19.9.13
58h	WAKEENASET2	Wake-up Enable Set Register 2	Section 19.9.13
5Ch	WAKEENASET3	Wake-up Enable Set Register 3	Section 19.9.13
60h	WAKEENACLR0	Wake-up Enable Clear Register 0	Section 19.9.14
64h	WAKEENACLR1	Wake-up Enable Clear Register 1	Section 19.9.14
68h	WAKEENACLR2	Wake-up Enable Clear Register 2	Section 19.9.14
6Ch	WAKEENACLR3	Wake-up Enable Clear Register 3	Section 19.9.14
70h	IRQVECREG	IRQ Interrupt Vector Register	Section 19.9.15
74h	FIQVECREG	FIQ Interrupt Vector Register	Section 19.9.16
78h	CAPEVT	Capture Event Register	Section 19.9.17
80h-FCh	CHANCTRL	VIM Interrupt Control Register	Section 19.9.18

19.9.1 Interrupt Vector Table ECC Status Register (ECCSTAT)

Figure 19-12 and Table 19-6 describe this register.

Figure 19-12. Interrupt Vector Table ECC Status Register (ECCSTAT) [offset = ECh]

31–16
Reserved
R-0

15–9	8	7–1	0
Reserved	SBERR	Reserved	UERR
R-0	R/W1CP-0	R-0	R/W1CP-0

LEGEND: R/W = Read/Write; R = Read only; W1CP = Write 1 to clear in privilege mode only; -*n* = value after reset

Table 19-6. Interrupt Vector Table ECC Status Register (ECCSTAT) Field Descriptions

Bit	Field	Value	Description
31-9	Reserved	0	Reads return 0. Writes have no effect.
8	SBERR		The SBERR indicates that a single-bit error has been detected and has been corrected by the SECDED logic and the Interrupt Vector Table is being used for normal operation (not bypassed).
		0	*Read:* No single-bit error has occurred. *Write:* No effect.
		1	*Read:* A single-bit error has occurred and was corrected by the SECDED logic. *Write:* The SBERR bit is cleared.
7-1	Reserved	0	Reads return 0. Writes have no effect.
0	UERR		The UERR indicates that a double-bit error has been found and that the Interrupt Vector Table is bypassed. The resulting vector of any IRQ/FRQ interrupt is then the value contained in the FBVECADDR register until this bit has been cleared.
		0	*Read:* No double-bit error has occurred. *Write:* No effect.
		1	*Read:* A double-bit error has occurred and the Interrupt Vector Table is bypassed. *Write:* The UERR bit is cleared and the interrupt vector can be read from the Interrupt Vector Table.

19.9.2 Interrupt Vector Table ECC Control Register (ECCCTL)

Figure 19-13. Interrupt Vector Table ECC Control Register (ECCCTL) [offset = F0h]

31 28	27 24	23 20	19 16
Reserved	SBE_EVT_EN	Reserved	EDAC_MODE
R-0	R/WP-5h	R-0	R/WP-Ah

15 12	11 8	7 4	3 0
Reserved	TEST_DIAG_EN	Reserved	ECCENA
R-0	R/WP-Ah	R-0	R/WP-5h

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privilege mode only; -*n* = value after reset

Table 19-7. Interrupt Vector Table ECC Control Register (ECCCTL) Field Descriptions

Bit	Field	Value	Description
31-28	Reserved	0	Reads return 0. Writes have no effect.
27-24	SBE_EVT_EN		These bits control the generation of Error signal out based on Single-Bit Error (SBE) indications from SECDED logic for the Interrupt Vector Table.
		5h	Disable Error Event indication upon detection of SBE on the Interrupt Vector Table.
		Ah	Enable Error Event upon detection of SBE the Interrupt Vector Table.
		All other values	Writes are ignored and the values are not updated into this field. The state of the feature remains unchanged.
23-20	Reserved	0	Reads return 0. Writes have no effect.
19-16	EDAC_MODE		These bits determine whether Single-Bit Errors (SBE) detected by the SECDED block will be corrected or not.
		5h	Disable correction of SBE detected by the SECDED block.
		Ah	Enable correction of SBE detected by the SECDED block.
		All other values	Writes are ignored and the values are not updated into this field. The state of the feature remains unchanged.
			Note: If an SBE is selected to be not corrected (using EDAC_MODE), then an SBE event will also cause VIM RAM to be bypassed just like UERR and the module to use the FBVECADDR register as the vector address.
15-12	Reserved	0	Reads return 0. Writes have no effect.
11-8	TEST_DIAG_EN		This bit maps the ECC bits into the Interrupt Vector Table frame to make them accessible by the CPU. When enabled, the ECC bits are writable as well as readable independent of data bits.
		5h	Enable memory-mapping of ECC bits for read/write operation.
		All other values	Disable memory-mapping of ECC bits for read/write operation.
			Note: To avoid soft error to disable VIM ECC mapping, it is recommended to write Ah to disable ECC bits mapping.
7-4	Reserved	0	Reads return 0. Writes have no effect.
3-0	ECCENA		VIM ECC enable.
		5h	VIM ECC is disabled.
		All other values	VIM ECC is enabled.
			Note: To avoid soft error to disable VIM ECC checking, it is recommended to write Ah to enable ECC checking.

19.9.3 Uncorrectable Error Address Register (UERRADDR)

The uncorrectable error address register gives the address of the first uncorrectable error location.

NOTE: No computation is needed when reading the complete register to retrieve the address in the Interrupt Vector Table.

This register will never be reset by a power-on reset nor any other reset source.

Figure 19-14. Uncorrectable Error Address Register (UERRADDR) [offset = F4h]

31	16
Interrupt Vector Table offset	
R-FFF8h	

15	10	9	2	1	0
Interrupt Vector Table offset		ADDERR		Word offset	
R-0010 000b		R-x		R-0	

LEGEND: R = Read only; x = value is indeterminate; -*n* = value after reset

Table 19-8. Uncorrectable Error Address Register (UERRADDR) Field Descriptions

Bit	Field	Description
31-10	Interrupt Vector Table offset	Interrupt Vector Table offset. Reads are always FFF8 2xxxh; writes have no effect.
9-2	ADDERR	Uncorrectable error address register. This register gives the address of the first encountered double-bit error since the flag has been clear. Subsequent ECC errors will not update this register until the UERR flag has been cleared. **Note: This register is valid only when PARFLG is set (see Section 19.9.1).**
1-0	Word offset	Word offset. Reads are always 0; writes have no effect.

19.9.4 Fallback Vector Address Register (FBVECADDR)

This register provides a fall-back address to the VIM if a uncorrectable error has occurred in the Interrupt Vector Table. Figure 19-15 and Table 19-9 describe this register.

NOTE: This register will never be reset by a power-on reset nor any other reset source.

Figure 19-15. Fallback Vector Address Register (FBVECADDR) [offset = F8h]

31	0
FBVECADDR	
R/WP-x	

LEGEND: R/W = Read/Write; WP = Write in privilege mode only; x = value is indeterminate; -*n* = value after reset

Table 19-9. Fallback Vector Address Register (FBVECADDR) Field Descriptions

Bit	Field	Description
31-0	FBVECADDR	Fallback Vector Address Register. This register is used by the VIM if the Interrupt Vector Table has been corrupted. The contents of the IRQVECREG and FIQVECREG registers will reflect the value programmed in FBVECADDR. The value provided to the VIC port will also reflect FBVECADDR until the UERR register has been cleared. This register provides the address of the ISR that will restore the integrity of the Interrupt Vector Table.

19.9.5 Single-Bit Error Address Register (SBERRADDR)

This register gives the address of the first single-bit ECC error detected by the ECC logic.Figure 19-16 and Table 19-10 describe this register.

NOTE: This register will never be reset by a power-on reset nor any other reset source.

Figure 19-16. Single-Bit Error Address Register (SBERRADDR) [offset = FCh]

<table>
<tr><td>31</td><td>0</td></tr>
<tr><td colspan="2">SBERRADDR</td></tr>
<tr><td colspan="2">R/WP-x</td></tr>
</table>

LEGEND: R/W = Read/Write; WP = Write in privilege mode only; x = value is indeterminate; -*n* = value after reset

Table 19-10. Single-Bit Error Address Register (SBERRADDR) Field Descriptions

Bit	Field	Description
31-0	SBERRADDR	Single-Bit Error Address Register. This register gives the address of the first single-bit error detected by the SECDED logic since the SBERR flag has been clear. Subsequent single-bit ECC errors will not update this register until the SBERR flag has been cleared. This register provides the Interrupt Vector Table address (offset from base address word aligned) of the ECC error location. This register is valid only when the SBERR flag is set.

19.9.6 VIM Offset Vector Registers

The VIM offset register provides the user with the numerical index value that represents the pending interrupt with the highest precedence. The register IRQINDEX holds the index to the highest priority IRQ interrupt; the register FIQINDEX holds the index to the highest priority FIQ interrupt. The index can be used to locate the interrupt routine in a dispatch table, as shown in Table 19-11.

Table 19-11. Interrupt Dispatch

IRQINDEX / FIQINDEX Register Bit Field	Highest Priority Pending Interrupt Enabled
0x00	No interrupt
0x01	Channel 0
:	:
0x7F	Channel 126
0x80	Channel 127

NOTE: Channel 127 has no dedicated interrupt vector table entry. Therefore, Channel 127 shall NOT be used in application.

The VIM offset registers are read only. They are updated continuously by the VIM. When an interrupt is serviced, the offset vectors show the index for the next highest pending interrupt or 0x0 if no interrupt is pending.

19.9.7 IRQ Index Offset Vector Register (IRQINDEX)

The IRQ offset register provides the user with the numerical index value that represents the pending IRQ interrupt with the highest priority. Figure 19-17 and Table 19-12 describe this register.

Figure 19-17. IRQ Index Offset Vector Register (IRQINDEX) [offset = 00h]

31–16
Reserved
R-0

15–8	7–0
Reserved	IRQINDEX
R-0	R-0

LEGEND: R = Read only; -*n* = value after reset

Table 19-12. IRQ Index Offset Vector Register (IRQINDEX) Field Descriptions

Bit	Field	Value	Description
31-8	Reserved	0	Reads return 0. Writes have no effect.
7-0	IRQINDEX	0-FFh	IRQ index vector. The least-significant bits represent the index of the IRQ pending interrupt with the highest precedence, as shown in Table 19-11. When no interrupts are pending, the least-significant byte of IRQINDEX is 0.

19.9.8 FIQ Index Offset Vector Registers (FIQINDEX)

The FIQINDEX register provides the user with a numerical index value that represents the pending FIQ interrupt with the highest priority. Figure 19-18 and Table 19-13 describe this register.

Figure 19-18. FIQ Index Offset Vector Register (FIQINDEX) [offset = F04h]

31–16
Reserved
R-0

15–8	7–0
Reserved	FIQINDEX
R-0	R-0

LEGEND: R = Read only; -*n* = value after reset

Table 19-13. FIQ Index Offset Vector Register (FIQINDEX) Field Descriptions

Bit	Field	Value	Description
31-8	Reserved	0	Reads return 0. Writes have no effect.
7-0	FIQINDEX	0-FFh	FIQ index offset vector. The least-significant bits represent the index of the FIQ pending interrupt with the highest precedence, as shown in Table 19-11. When no interrupts are pending, the least-significant byte of FIQINDEX is 0x00.

19.9.9 FIQ/IRQ Program Control Registers (FIRQPR[0:3])

The FIQ/IRQ program control registers determine whether a given interrupt request will be either FIQ or IRQ. Figure 19-19, Figure 19-20, Figure 19-21, Figure 19-22 and Table 19-14 describe these registers.

NOTE: Channel 0 and 1 are FIQ only, not impacted by this register.

Figure 19-19. FIQ/IRQ Program Control Register 0 (FIRQPR0) [offset = 10h]

31	16
FIRQPR0[31:16]	
R/WP-0	

15	2	1	0
FIRQPR0[15:2]		Reserved	
R/WP-0		R-3h	

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privilege mode only; -*n* = value after reset

Figure 19-20. FIQ/IRQ Program Control Register 1 (FIRQPR1) [offset = F14h]

31	0
FIRQPR1[63:32]	
R/WP-0	

LEGEND: R/W = Read/Write; WP = Write in privilege mode only; -*n* = value after reset

Figure 19-21. FIQ/IRQ Program Control Register 2 (FIRQPR2) [offset = 18h]

31	0
FIRQPR2[95:64]	
R/WP-0	

LEGEND: R/W = Read/Write; WP = Write in privilege mode only; -*n* = value after reset

Figure 19-22. FIQ/IRQ Program Control Register 3 (FIRQPR3) [offset = 1Ch]

31	0
FIRQPR3[127:96]	
R/WP-0	

LEGEND: R/W = Read/Write; WP = Write in privilege mode only; -*n* = value after reset

Table 19-14. FIQ/IRQ Program Control Registers (FIRQPR) Field Descriptions

Bit	Field	Value	Description
127-2	FIRQPRx[*n*]		FIQ/IRQ program control bits. These bits determine whether an interrupt request from a peripheral is of type FIQ or IRQ. Bit FIRQPRx[127:2] corresponds to request channel[127:2].
		0	Interrupt request is of IRQ type.
		1	Interrupt request is of FIQ type.
1-0	Reserved	3h	Read only. Writes have no effect.

19.9.10 Pending Interrupt Read Location Registers (INTREQ[0:3])

The pending interrupt register gives the pending interrupt requests. The register is updated every vbus clock cycle. Figure 19-23, Figure 19-24, Figure 19-25, Figure 19-26 and Table 19-15 describe this register.

Figure 19-23. Pending Interrupt Read Location Register 0 (INTREQ0) [offset = 20h]

31	0
INTREQ0[31:0]	
R/W1CP-0	

LEGEND: R/W = Read/Write; W1CP = Write 1 to clear in privilege mode only; -*n* = value after reset

Figure 19-24. Pending Interrupt Read Location Register 1 (INTREQ1) [offset = 24h]

31	0
INTREQ1[63:32]	
R/W1CP-0	

LEGEND: R/W = Read/Write; W1CP = Write 1 to clear in privilege mode only; -*n* = value after reset

Figure 19-25. Pending Interrupt Read Location Register 2 (INTREQ2) [offset = 28h]

31	0
INTREQ2[95:64]	
R/W1CP-0	

LEGEND: R/W = Read/Write; W1CP = Write 1 to clear in privilege mode only; -*n* = value after reset

Figure 19-26. Pending Interrupt Read Location Register 3 (INTREQ3) [offset = 2Ch]

31	0
INTREQ3[127:96]	
R/W1CP-0	

LEGEND: R/W = Read/Write; W1CP = Write 1 to clear in privilege mode only; -*n* = value after reset

Table 19-15. Pending Interrupt Read Location Registers (INTREQ) Field Descriptions

Bit	Field	Value	Description
127-0	INTREQx[*n*]		Pending interrupt bits. These bits determine whether an interrupt request is pending for the request channel between 0 and 127. The interrupt ENABLE register does not affect the value of the interrupt pending bit. Bit INTREQx[127:0] corresponds to request channel[127:0].
			User and Privilege Mode read:
		0	No interrupt event has occurred.
		1	An interrupt is pending.
			Privilege Mode write only:
		0	Writing 0 has no effect.
		1	Clears the interrupt pending status flag. This write-clear functionality is intended to allow clearing those interrupts which have been signaled to VIM before enabling the interrupt channel, if they are undesired.

19.9.11 Interrupt Enable Set Registers (REQENASET[0:3])

The interrupt register enable selectively enables individual request channels. Figure 19-27, Figure 19-28, Figure 19-29, Figure 19-30 and Table 19-16 describe these registers.

NOTE: Channel 0 and 1 are always enabled, not impacted by this register.

Figure 19-27. Interrupt Enable Set Register 0 (REQENASET0) [offset = 30h]

31 16
REQENASET0[31:16]
R/WP-0

15 2	1 0
REQENASET0[15:2]	Reserved
R/WP-0	R-3h

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privilege mode only; -*n* = value after reset

Figure 19-28. Interrupt Enable Set Register 1 (REQENASET1) [offset = 34h]

31 0
REQENASET1[63:32]
R/WP-0

LEGEND: R/W = Read/Write; WP = Write in privilege mode only; -*n* = value after reset

Figure 19-29. Interrupt Enable Set Register 2 (REQENASET2) [offset = 38h]

31 0
REQENASET2[95:64]
R/WP-0

LEGEND: R/W = Read/Write; WP = Write in privilege mode only; -*n* = value after reset

Figure 19-30. Interrupt Enable Set Register 3 (REQENASET3) [offset = 3Ch]

31 0
REQENASET3[127:96]
R/WP-0

LEGEND: R/W = Read/Write; WP = Write in privilege mode only; -*n* = value after reset

Table 19-16. Interrupt Enable Set Registers (REQENASET) Field Descriptions

Bit	Field	Value	Description
127-2	REQENASETx[*n*]		Request enable set bits. This vector determines whether the interrupt request channel is enabled. Bit REQENASETx[127:2] corresponds to request channel[127:2].
		0	*Read:* Interrupt request channel is disabled. *Write:* No effect.
		1	*Read or Write:* The interrupt request channel is enabled.
1-0	Reserved	3h	Read only. Writes have no effect.

19.9.12 Interrupt Enable Clear Registers (REQENACLR[0:3])

The interrupt register enable selectively disables individual request channels. Figure 19-31, Figure 19-32, Figure 19-33, Figure 19-34 and Table 19-17 describe these registers.

NOTE: Channel 0 and 1 are always enabled, not impacted by this register.

Figure 19-31. Interrupt Enable Clear Register 0 (REQENACLR0) [offset = 40h]

31	16
REQENACLR0[31:16]	
R/WP-0	

15	2	1	0
REQENACLR0[15:2]		Reserved	
R/WP-0		R-3h	

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privilege mode only; -*n* = value after reset

Figure 19-32. Interrupt Enable Clear Register 1 (REQENACLR1) [offset = 44h]

31	0
REQENACLR1[63:32]	
R/WP-0	

LEGEND: R/W = Read/Write; WP = Write in privilege mode only; -*n* = value after reset

Figure 19-33. Interrupt Enable Clear Register 2 (REQENACLR2) [offset = 48h]

31	0
REQENACLR2[95:64]	
R/WP-0	

LEGEND: R/W = Read/Write; WP = Write in privilege mode only; -*n* = value after reset

Figure 19-34. Interrupt Enable Clear Register 3 (REQENACLR3) [offset = 4Ch]

31	0
REQENACLR3[127:96]	
R/WP-0	

LEGEND: R/W = Read/Write; WP = Write in privilege mode only; -*n* = value after reset

Table 19-17. Interrupt Enable Clear Registers (REQENACLR) Field Descriptions

Bit	Field	Value	Description
127-2	REQENACLRx[*n*]		Request enable clear bits. This vector determines whether the interrupt request channel is enabled. Bit REQENACLRx[127:2] corresponds to request channel[127:2].
		0	*Read:* Interrupt request channel is disabled. *Write:* No effect.
		1	*Read:* The interrupt request channel is enabled. *Write:* The interrupt request channel is disabled.
1-0	Reserved	3h	Read only. Writes have no effect.

19.9.13 Wake-Up Enable Set Registers (WAKEENASET[0:3])

The wake-up enable registers selectively enables individual wake-up interrupt request lines. Figure 19-35, Figure 19-36, Figure 19-37, Figure 19-38 and Table 19-18 describe these registers.

Figure 19-35. Wake-Up Enable Set Register 0 (WAKEENASET0) [offset = 50h]

31	0
WAKEENASET0[31:0]	
R/WP-FFFF FFFFh	

LEGEND: R/W = Read/Write; WP = Write in privilege mode only; -*n* = value after reset

Figure 19-36. Wake-Up Enable Set Register 1 (WAKEENASET1) [offset = 54h]

31	0
WAKEENASET1[63:32]	
R/WP-FFFF FFFFh	

LEGEND: R/W = Read/Write; WP = Write in privilege mode only; -*n* = value after reset

Figure 19-37. Wake-Up Enable Set Register 2 (WAKEENASET2) [offset = 58h]

31	0
WAKEENASET2[95:64]	
R/WP-FFFF FFFFh	

LEGEND: R/W = Read/Write; WP = Write in privilege mode only; -*n* = value after reset

Figure 19-38. Wake-Up Enable Set Register 3 (WAKEENASET3) [offset = 5Ch]

31	0
WAKEENASET3[127:96]	
R/WP-FFFF FFFFh	

LEGEND: R/W = Read/Write; WP = Write in privilege mode only; -*n* = value after reset

Table 19-18. Wake-Up Enable Set Registers (WAKEENASET) Field Descriptions

Bit	Field	Value	Description
127-0	WAKEENASETx[*n*]		Wake-up enable set bits. This vector determines whether the wake-up interrupt line is enabled. Bit WAKEENASETx[127:0] corresponds to interrupt request channel[127:0].
		0	*Read:* Interrupt request channel is disabled. *Write:* No effect.
		1	*Read or Write:*The interrupt request channel is enabled.

19.9.14 Wake-Up Enable Clear Registers (WAKEENACLR[0:3])

The wake-up enable register selectively disables individual wake-up interrupt request lines. Figure 19-39, Figure 19-40, Figure 19-41, Figure 19-42 and Table 19-19 describe these registers.

Figure 19-39. Wake-Up Enable Clear Register 0 (WAKEENACLR0) [offset = 60h]

31 ... 0
WAKEENACLR0[31:0]
R/WP-FFFF FFFFh

LEGEND: R/W = Read/Write; WP = Write in privilege mode only; -*n* = value after reset

Figure 19-40. Wake-Up Enable Clear Register 1 (WAKEENACLR1) [offset = 64h]

31 ... 0
WAKEENACLR1[63:32]
R/WP-FFFF FFFFh

LEGEND: R/W = Read/Write; WP = Write in privilege mode only; -*n* = value after reset

Figure 19-41. Wake-Up Enable Clear Register 2 (WAKEENACLR2) [offset = 68h]

31 ... 0
WAKEENACLR2[95:64]
R/WP-FFFF FFFFh

LEGEND: R/W = Read/Write; WP = Write in privilege mode only; -*n* = value after reset

Figure 19-42. Wake-Up Enable Clear Register 3 (WAKEENACLR3) [offset = 6Ch]

31 ... 0
WAKEENACLR3[127:96]
R/WP-FFFF FFFFh

LEGEND: R/W = Read/Write; WP = Write in privilege mode only; -*n* = value after reset

Table 19-19. Wake-Up Enable Clear Registers (WAKEENACLR) Field Descriptions

Bit	Field	Value	Description
127-0	WAKEENACLRx[*n*]		Wake-up enable clear bits. This vector determines whether the wake-up interrupt line is enabled. Bit WAKEENACLRx[127:0] corresponds to interrupt request channel[127:0].
		0	*Read:* Wake-up interrupt channel is disabled. *Write:* No effect.
		1	*Read:* The wake-up interrupt channel is enabled. *Write:* The wake-up interrupt channel is disabled.

19.9.15 IRQ Interrupt Vector Register (IRQVECREG)

The interrupt vector register gives the address of the enabled and active IRQ interrupt. Figure 19-43 and Table 19-20 describe these registers.

Figure 19-43. IRQ Interrupt Vector Register (IRQVECREG) [offset = 70h]

31	0
IRQVECREG	
R-0	

LEGEND: R = Read only; -*n* = value after reset

Table 19-20. IRQ Interrupt Vector Register (IRQVECREG) Field Descriptions

Bit	Field	Value	Description
31-0	IRQVECREG	From Section 19.5	IRQ interrupt vector register. This vector gives the address of the ISR with the highest pending IRQ request. The CPU reads the address and branches to this location.

19.9.16 FIQ Interrupt Vector Register (FIQVECREG)

The interrupt vector register gives the address of the enabled and active FIQ interrupt. Figure 19-44 and Table 19-21 describe these registers.

Figure 19-44. IRQ Interrupt Vector Register (*FIQVECREG*) [offset = 74h]

31	0
FIQVECREG	
R-0	

LEGEND: R = Read only; -*n* = value after reset

Table 19-21. FIQ Interrupt Vector Register (FIQVECREG) Field Descriptions

Bit	Field	Value	Description
31-0	FIQVECREG	From Section 19.5	FIQ interrupt vector register. This vector gives the address of the ISR with the highest pending FIQ request. The CPU reads the address and branches to this location.

19.9.17 Capture Event Register (CAPEVT)

Figure 19-45 and Table 19-22 describe this register.

Figure 19-45. Capture Event Register (CAPEVT) [offset = 78h]

31 23	22 16
Reserved	CAPEVTSRC1
R-U	R/WP-0

15 7	6 0
Reserved	CAPEVTSRC0
R-U	R/WP-0

LEGEND: R = Read only; WP = Write in privilege mode only; U = value is undefined; -*n* = value after reset

Table 19-22. Capture Event Register (CAPEVT) Field Descriptions

Bit	Field	Value	Description
31-23	Reserved	0	Reads are indeterminate and writes have no effect.
22-16	CAPEVTSRC1		Capture event source 1 mapping control. These bits determine which interrupt request maps to the capture event source 1 of the RTI:
		0	Interrupt request 0.
		1h	Interrupt request 1.
		:	:
		7Fh	Interrupt request 127.
15-7	Reserved	0	Reads are indeterminate and writes have no effect.
6-0	CAPEVTSRC0		Capture event source 0 mapping control. These bits determine which interrupt request maps to the capture event source 0 of the RTI:
		0	Interrupt request 0.
		1h	Interrupt request 1.
		:	:
		7Fh	Interrupt request 127.

19.9.18 VIM Interrupt Control Registers (CHANCTRL[0:31])

Thirty-two interrupt control registers control the 128 interrupt channels of the VIM. Each register controls four interrupt channels: each of them is indexed from 0 to 127. Table 19-23 shows the organization of all the registers and the reset value of each. Each four fields of the register has been named with a generic index that refers to the detailed register organization. Figure 19-46 and Table 19-24 describe these registers.

Table 19-23. Interrupt Control Registers Organization

Address	Register Acronym	Register Field 31:24 CHANMAPx_0	Register Field 23:16 CHANMAPx_1	Register Field 15:8 CHANMAPx_2	Register Field 7:0 CHANMAPx_3	Reset Value
FFFF FE80h	CHANCTRL0	CHANMAP0	CHANMAP1	CHANMAP2	CHANMAP3	0001 0203h
FFFF FE84h	CHANCTRL1	CHANMAP4	CHANMAP5	CHANMAP6	CHANMAP7	0405 0607h
:	:	:	:	:	:	:
FFFF FEF8h	CHANCTRL30	CHANMAP120	CHANMAP121	CHANMAP122	CHANMAP123	7879 7A7Bh
FFFF FEFCh	CHANCTRL31	CHANMAP124	CHANMAP125	CHANMAP126	CHANMAP127	7C7D 7E7Fh

NOTE: CHANMAP0 and CHANMAP1 are not programable. CHAN0 and CHAN1 are hard wired to INT_REQ0 and INT_REQ1.

Do NOT write any value other than 0x7F to CHANMAP127. Channel 127 is reserved because no interrupt vector table entry supports this channel.

Figure 19-46. Interrupt Control Registers (CHANCTRL[0:31]) [offset = 80h-FCh]

31	30 - 24	23	22 - 16
Rsvd	CHANMAPx_0	Rsvd	CHANMAPx_1
R-U	R/WP-n	R-U	R/WP-n

15	14 - 8	7	6 - 0
Rsvd	CHANMAPx_2	Rsvd	CHANMAPx_3
R-U	R/WP-n	R-U	R/WP-n

LEGEND: R = Read only; WP = Write in privilege mode only; U = value is undefined; -*n* = value after reset (see Table 19-23)

Table 19-24. Interrupt Control Registers (CHANCTRL[0:31]) Field Descriptions

Bit	Field	Value	Description
31	Reserved	0	Reads are indeterminate and writes have no effect.
30-24	CHANMAPx_0		CHANMAPx_0(6-0). Interrupt CHANx_0 mapping control. These bits determine which interrupt request the priority channel CHANx_0 maps to:
		0	*Read:* Interrupt request 0 maps to channel priority CHANx_0. *Write:* The default value of this bit after reset is given in Table 19-23 . The channel priority CHANx_0 is set with the interrupt request.
		1h	*Read:* Interrupt request 1 maps to channel priority CHANx_0. *Write:* The default value of this bit after reset is given in Table 19-23. The channel priority CHANx_0 is set with the interrupt request.
		:	:
		7Fh	*Read:* Interrupt request 127 maps to channel priority CHANx_0. *Write:* The default value of this bit after reset is given in Table 19-23. The channel priority CHANx_0 is set with the interrupt request.
23	Reserved	0	Reads are indeterminate and writes have no effect.

Table 19-24. Interrupt Control Registers (CHANCTRL[0:31]) Field Descriptions (continued)

Bit	Field	Value	Description
22-16	CHANMAPx$_1$	 0	CHANMAPx$_1$(6-0). Interrupt CHANx$_1$ mapping control. These bits determine which interrupt request the priority channel CHANx$_1$ maps to: *Read:* Interrupt request 0 maps to channel priority CHANx$_1$. *Write:* The default value of this bit after reset is given in Table 19-23. The channel priority CHANx$_1$ is set with the interrupt request.
		1h :	*Read:* Interrupt request 1 maps to channel priority CHANx$_1$. *Write:* The default value of this bit after reset is given in Table 19-23. The channel priority CHANx$_1$ is set with the interrupt request. :
		7Fh	*Read:* Interrupt request 127 maps to channel priority CHANx$_1$. *Write:* The default value of this bit after reset is given in Table 19-23. The channel priority CHANx$_1$ is set with the interrupt request.
15	Reserved	0	Reads are indeterminate and writes have no effect.
14-8	CHANMAPx$_2$	 0	CHANMAPx$_2$(6-0). Interrupt CHANx$_2$ mapping control. These bits determine which interrupt request the priority channel CHANx$_2$ maps to: *Read:* Interrupt request 0 maps to channel priority CHANx$_2$. *Write:* The default value of this bit after reset is given in Table 19-23. The channel priority CHANx$_2$ is set with the interrupt request.
		1h :	*Read:* Interrupt request 1 maps to channel priority CHANx$_2$. *Write:* The default value of this bit after reset is given in Table 19-23. The channel priority CHANx$_2$ is set with the interrupt request. :
		7Fh	*Read:* Interrupt request 127 maps to channel priority CHANx$_2$. *Write:* The default value of this bit after reset is given in Table 19-23. The channel priority CHANx$_2$ is set with the interrupt request.
7	Reserved	0	Reads are indeterminate and writes have no effect.
6-0	CHANMAPx$_3$	 0	CHANMAPx$_3$(6-0). Interrupt CHANx$_3$ mapping control. These bits determine which interrupt request the priority channel CHANx$_3$ maps to: *Read:* Interrupt request 0 maps to channel priority CHANx$_3$. *Write:* The default value of this bit after reset is given in Table 19-23. The channel priority CHANx$_3$ is set with the interrupt request.
		1h :	*Read:* Interrupt request 1 maps to channel priority CHANx$_3$. *Write:* The default value of this bit after reset is given in Table 19-23. The channel priority CHANx$_3$ is set with the interrupt request. :
		7Fh	*Read:* Interrupt request 127 maps to channel priority CHANx$_3$. *Write:* The default value of this bit after reset is given in Table 19-23. The channel priority CHANx$_3$ is set with the interrupt request.

Chapter 20
SPNU563A–March 2018

Direct Memory Access Controller (DMA) Module

This chapter describes the direct memory access (DMA) controller.

Topic **Page**

20.1 Overview

The DMA controller is used to transfer data between two locations in the memory map in the background of CPU operations. Typically, the DMA is used to:

- Transfer blocks of data between external and internal data memories
- Restructure portions of internal data memory
- Continually service a peripheral
- Page program sections to internal program memory

Since the DMA has two master ports, the selection for the port should be made using the table mentioning the port to be used for each address region.

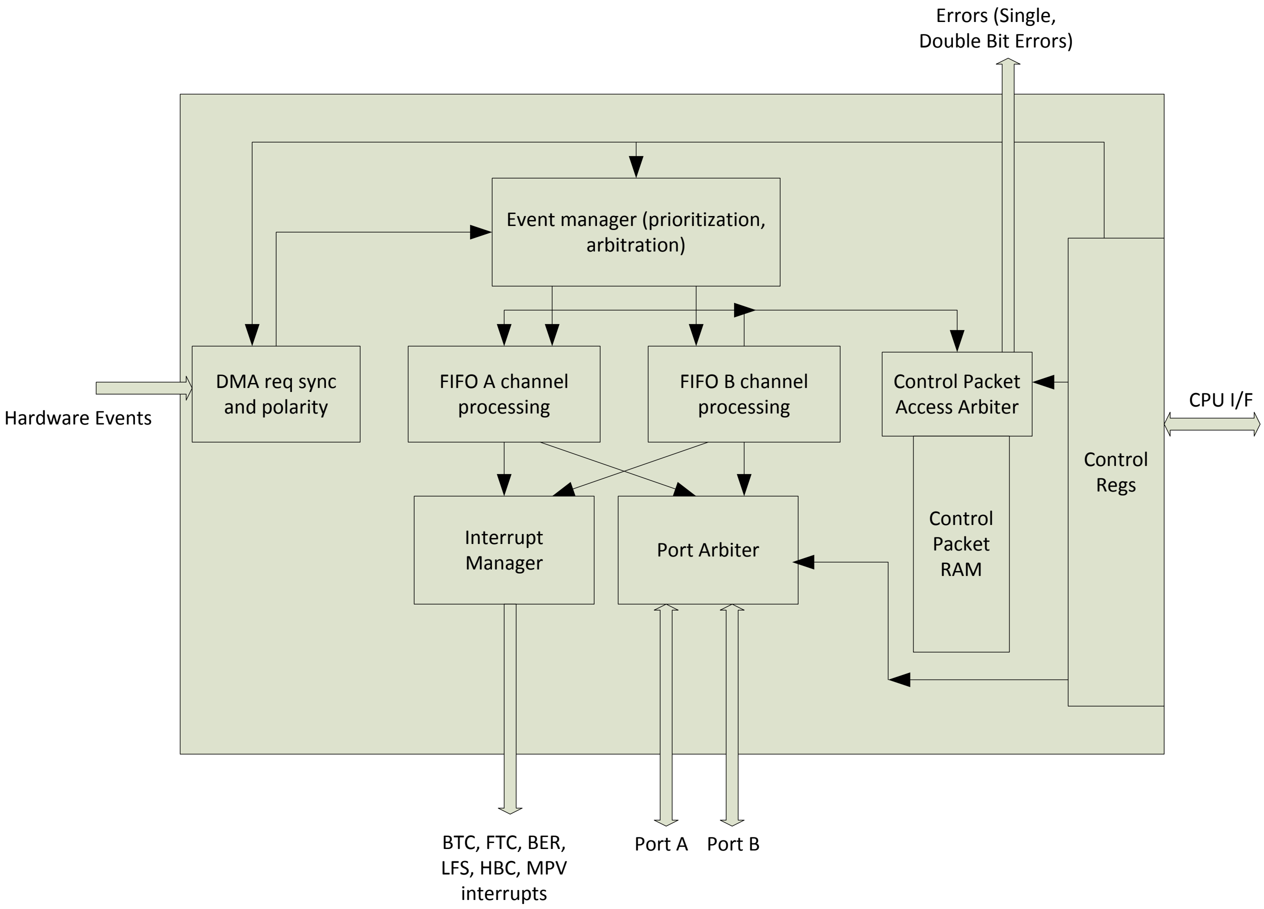
20.1.1 Main Features

- CPU independent data transfer
- Two master ports - PortA and PortB (each 64-bits wide) that interface with the Microcontroller's Bus Matrix System.
- Support for concurrent transfers on up to two different channels
- FIFO buffer (4 entries deep and each 64-bits wide)
- Channel control information is stored in RAM protected by ECC
- Multiple logical channels with individual enable (refer to the data manual for the number of channels on your device)
- Channel chaining capability
- 48 peripheral DMA requests
- Hardware and Software DMA requests
- 8-, 16-, 32-, or 64-bit transactions supported
- Multiple addressing modes for source/destination (fixed, increment, offset)
- Auto-initiation
- Power-management mode
- Memory Protection for the address range DMA can access with multiple configurable memory regions (refer datasheet for number of memory regions on your device)

20.1.1.1 Block Diagram

Figure 20-1 gives a top view of the DMA internal architecture. DMA data read and write access happen through either Port A or B. Both FIFO buffers are each 4 levels deep and 64-bits wide thus allowing a maximum of 32 bytes to be buffered inside the DMA per channel. DMA requests go into the DMA that can trigger DMA transfers. Five interrupt request lines go out of the DMA to signal that a certain transfer status is reached. Register banks hold the memory mapped DMA configuration registers. Local RAM consists of DMA control packets and is secured by ECC. All the programming / configuration of the DMA controller is done via the Peripheral bus.

Figure 20-1. DMA Block Diagram


Errors (Single, Double Bit Errors)
Event manager (prioritization, arbitration)
DMA req sync and polarity
FIFO A channel processing
FIFO B channel processing
Control Packet Access Arbiter
Hardware Events
CPU I/F
Control Regs
Control Packet RAM
Interrupt Manager
Port Arbiter
BTC, FTC, BER, LFS, HBC, MPV interrupts
Port A
Port B

20.1.2 System Resources Mapping

Table 20-1 shows how the system resources are mapped to either of the two DMA ports. In order to properly transfer data from one resource to another, the application must setup the PARx register according to Table 20-1.

Table 20-1. DMA Ports to System Resources Mapping

DMA Ports	System Resources
Port A	• L2 Flash • L2 SRAM • EMIF
Port B	• All peripherals, that is, MibSPI registers, DCAN registers • All peripheral memories, that is, MibSPI RAM, DCAN RAM

- Example 1: To transfer data from L2 Flash, L2 SRAM, or EMIF to any peripheral registers or peripheral memories, write 0x1 (Port A read, Port B write) to the respective channels in the PARx registers
- Example 2: To transfer data from any peripheral registers or peripheral memories to L2 SRAM or EMIF, write 0x0 (Port B read, Port A write) to the respective channels in the PARx registers
- Example 3: To transfer data from L2 Flash to L2 SRAM, write 0x2 (Port A) to the respective channels in the PARx registers
- Example 4: To transfer data from peripherals to another peripherals, write 0x3 (Port B) to the respective channels in the PARx registers.

20.2 Module Operation

The DMA acts as an independent master in the platform architecture. DMA will attempt to execute up to two channels at the same time to maximize system throughput. Each channel can be configured to utilize either Port A or B or both for the read and write accesses while storing the data in one of the FIFOs. Choice of Port A or Port B for a certain channel depends on the addresses chosen for the transfer and should be made by referring to Table 20-1. All DMA memory and register accesses are performed in user mode. If the DMA writes to registers which are only accessible in privileged mode, the write will not be performed.

The DMA registers and its local RAM can only be accessed in privilege mode. Therefore, it is not possible for the DMA to reprogram itself.

In order to further explain DMA operation, some terms are described below:

- Arbitration - A channel may get temporarily suspended in order to service a higher priority channel or when the channel is disabled on the fly. The channel is said to have been "arbitrated"
- Arbitration Boundary - Each time a channel finishes a chunk of transfer which can be a maximum of 32 bytes, it is said to have reached an arbitration boundary. The FIFO is empty at an arbitration boundary. The DMA will utilize this boundary to re-prioritize channels. Within an arbitration boundary, transfers can never be interrupted.

20.2.1 Memory Space

The DMA controller makes no distinction between program memory and data memory. The DMA controller can transfer to and from any space within the 4 gigabyte physical address map, by programming the absolute address for the source and destination in the control packet. Control packets store the transfer information such as source address, destination address, transfer count and control attributes for each channel.

20.2.2 DMA Data Access

The DMA controller refers to data in three levels of granularity:

- **Element:** Depending on the programmed data type, an 8-bit, 16-bit, 32-bit, or a 64-bit value. The type can be individually selected for the source (read) and destination (write). See Figure 20-2 and Figure 20-3 for an example of the use of elements. An element transfer cannot be interrupted.
- **Frame:** One or more elements to be transferred as a unit. A frame transfer can be interrupted between element transfers. See Figure 20-2 for an example. Use a frame size of one and frame transfer trigger source for transfers of one element per request.
- **Block:** One or more frames to be transferred as a unit. Each channel can transfer one block of data (once or multiple times). See Figure 20-3 for an example.

Figure 20-2. Example of a DMA Transfer Using Frame Trigger Source

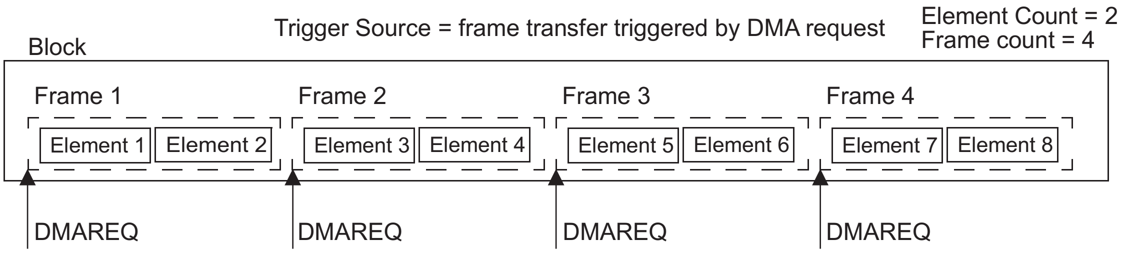
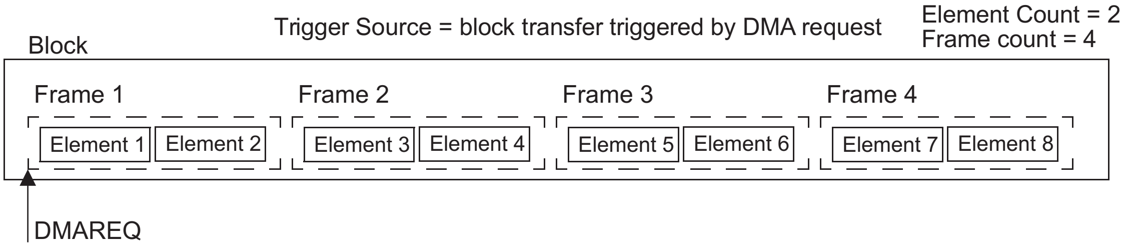


Figure 20-3. Example of a DMA Transfer Using Block Trigger Source



20.2.3 Addressing Modes

There are three addressing modes supported by the DMA controller that can be setup independent for the source and the destination address:

- Constant -- source and/or destination addresses do not change.
- Post incremented -- source and/or destination address are post-incremented by the element size.
- Indexed -- source and/or destination address is post-incremented as defined in the Element Index Offset Register (Section 20.3.2.5) and the Frame Index Offset Register (Section 20.3.2.6).

An unaligned address with respect to the element size is not supported.

20.2.4 DMA Channel Control Packets

Corresponding to each logical channel is a control packet that is mapped in fixed numerical order. For example, control packet 0 stores channel information for channel 0. The DMA requests can be mapped to the individual channels as described in Section 20.2.7. The mapping scheme between DMA requests and channels is shown in Figure 20-4. Each control packet contains nine fields. The first six fields compose the primary control packet and are programmable during DMA setup. The last three fields compose working control packet and are only readable by the CPU. The working control packets are used to support auto-initiation and prioritization of channels.. The organization of control packets is shown in Figure 20-5.

The primary control packet contains channel information such as source address, destination address, transfer count, element/frame offset value and channel configuration. Source address, destination address and transfer count also have their respective working images. The three fields of working images compose a working control packet and are not accessible to the CPU in write access.

The first time a DMA channel is selected for a transaction, the following process occurs:

1. The primary control packet is first read by the DMA state machine.
2. Once the channel is arbitrated, the current source address, destination address and transfer count are then copied to their respective working images.
3. When the channel is serviced again by the DMA, the state machine will read both the primary control packet and the working control packet to continue the DMA transaction until the end of an entire block transfer.

When the same channel is requested again, the state machine will start again by reading only the primary control packet and then continue the same process described above. The user software need not set up control packets again because the contents of the primary control packet were never lost. The working images of the control packets are reducing the software overhead and interaction with the DMA module to a minimum.

NOTE: Changing the contents of a channel control packet will clear the corresponding pending bit (Section 20.3.1.2) if the channel has a pending status. If the control packet of an active channel (as indicated in Section 20.3.1.3) is changed, then the channel will stop immediately at an arbitration boundary. When the same channel is triggered again, it will begin with the new control packet information.

Figure 20-4. DMA Request Mapping and Control Packet Organization

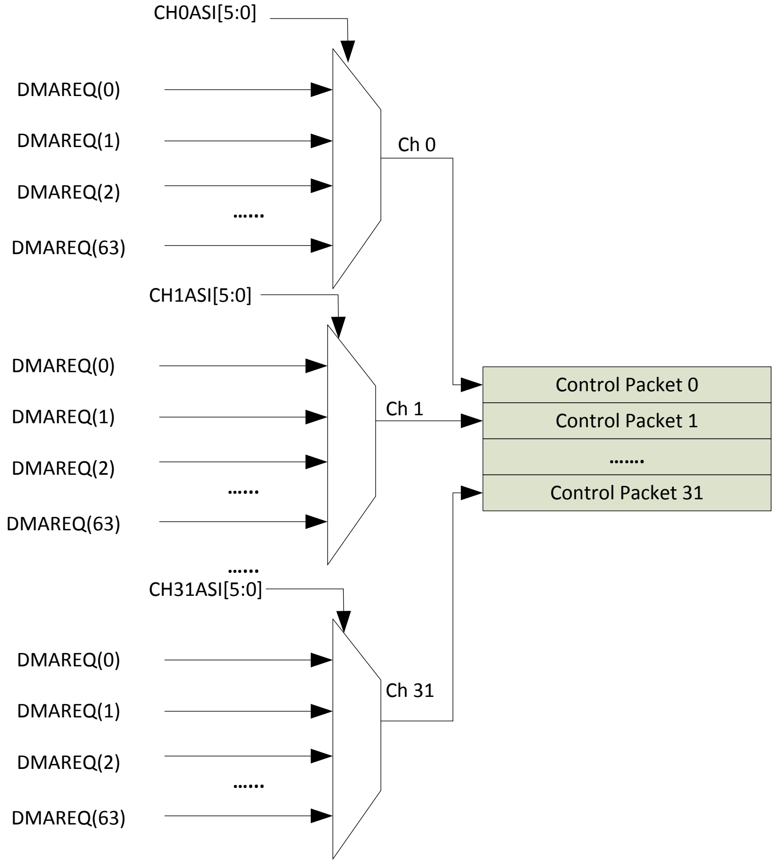


Figure 20-5. Control Packet Organization and Memory Map

	Base + 0XXX0	Base + 0xXXX4	Base + 0xXXX8		Base + 0xXXXC
Base + 0x00	Initial Source Address	Initial Destination Address	Initial Transfer Count	} Primary CP0	Reserved
0x10	Channel Configuration	Element Offset Value	Frame Offset Value		
0x20	Initial Source Address	Initial Destination Address	Initial Transfer Count	} Primary CP1	
0x30	Channel Configuration	Element Offset Value	Frame Offset Value		
0x1E0	Initial Source Address	Initial Destination Address	Initial Transfer Count	} Primary CPnn	
0x1F0	Channel Configuration	Element Offset Value	Frame Offset Value		
	Reserved	Reserved	Reserved		
0x800	Current Source Address	Current Destination Address	Current Transfer Count	} Working CP0	
0x810	Current Source Address	Current Destination Address	Current Transfer Count	} Working CP1	
0x8F0	Current Source Address	Current Destination Address	Current Transfer Count	} Working CPnn	

20.2.4.1 Initial Source Address

This field stores the absolute 32-bit source address of the DMA transfer.

20.2.4.2 Initial Destination Address

This field stores the absolute 32-bit destination address of the DMA transfer.

20.2.4.3 Initial Transfer Count

The transfer count field is composed of two parts. The frame transfer count value and the element transfer count value. Each count value is 13-bits wide. As a Single Block transfer maximum of 512 Mbytes of data can be transferred. Element count and frame count are programmed according to the source data structure.

The total transfer size is calculated as:

$$T_{sz} = E_{rsz} \bullet E_{tc} \bullet F_{tc} \tag{26}$$

where

T_{sz} = Total Transfer Size

E_{rsz} = Read Element Size

E_{tc} = Element Transfer Count

F_{tc} = Frame Transfer Count

NOTE: A zero element count with a non-zero frame count or a non-zero element count with a zero frame count are all considered as zero total transfer count. No DMA transaction is initiated with any of the counters set to 0.

20.2.4.4 Channel Configuration Word

The channel configuration defines the following individual parameters:

- Read element size
- Write element size
- Trigger type (frame or block)
- Addressing mode for source
- Addressing mode for destination
- Auto-initiation mode
- Next control packet to be triggered at control packet finish (Channel Chaining)

20.2.4.5 Element/Frame Offset Value

There are 4 offset values that allow the creation of different types of buffers in RAM and address registers in a structured manner: an element offset value for source and destination and a frame offset value for source and destination.

The element offset value for source and/or destination defines the offset to be added after each element transfer to the source and/or destination address. The frame offset value for source and/or destination defines the offset to be added to the source and/or destination address after the element count reaches zero. The element and frame offset values must be defined in terms of the number of bytes of offset. The DMA controller does not adjust the element/frame index number according to the element size. An index of 2 means *increment the address by 2* and not by 16 when the element size is 64 bits.

20.2.4.6 Current Source Address

The current source address field contains the current working source address during a DMA transaction. The current source address is incremented during post increment addressing mode or indexing mode.

20.2.4.7 Current Destination Address

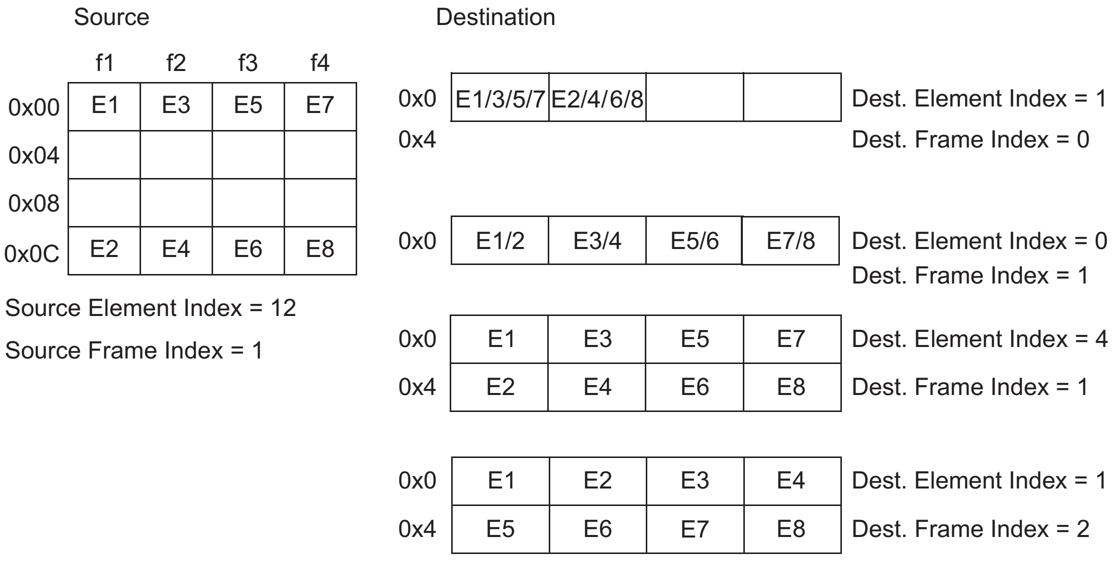
The current destination address field contains the current working destination address during a DMA transaction. The current destination address is incremented during post-increment addressing mode or indexing mode.

20.2.4.8 Current Transfer Count

The current transfer count stores the remaining number of elements to be transferred in a block. It is decremented by one for each element read from the source location.

Figure 20-6, Figure 20-7, and Figure 20-8 show some examples of DMA transfers.

Figure 20-6. DMA Transfer Example 1



Source

	f1	f2	f3	f4
0x00	E1	E3	E5	E7
0x04				
0x08				
0x0C	E2	E4	E6	E8

Source Element Index = 12

Source Frame Index = 1

Destination

0x0	E1/3/5/7	E2/4/6/8		
0x4				

Dest. Element Index = 1

Dest. Frame Index = 0

0x0	E1/2	E3/4	E5/6	E7/8

Dest. Element Index = 0

Dest. Frame Index = 1

0x0	E1	E3	E5	E7
0x4	E2	E4	E6	E8

Dest. Element Index = 4

Dest. Frame Index = 1

0x0	E1	E2	E3	E4
0x4	E5	E6	E7	E8

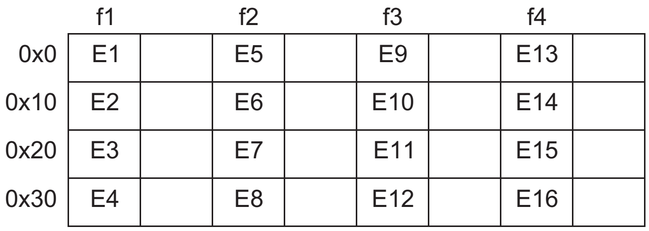
Dest. Element Index = 1

Dest. Frame Index = 2

The example assumes the following setup.

Read Element Size = 8 bit
Write Element Size = 8 bit
Element Count = 2
Frame Count = 4

Figure 20-7. DMA Indexing Example 1



	f1		f2		f3		f4	
0x0	E1		E5		E9		E13	
0x10	E2		E6		E10		E14	
0x20	E3		E7		E11		E15	
0x30	E4		E8		E12		E16	

Element Index = 16

Frame Index = 4

This example can be applied to either source or destination indexing and assumes the following setup.

Element Size = 16 bit
Element Count = 4
Frame Count = 4

Figure 20-8. DMA Indexing Example 2

0x0	E1	E4	E7	E10	E13	E16	E19	E22
0x20								
0x40	E2	E5	E8	E11	E14	E17	E20	E23
0x60								
0x80	E3	E6	E9	E12	E15	E18	E21	E23

Element Index = 64
Frame Index = 4

This example can be applied to either source or destination indexing and assumes the following setup.

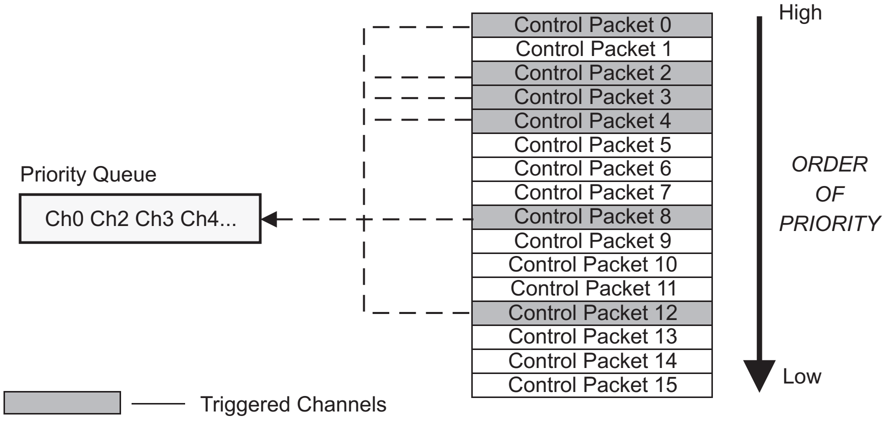
Element Size = 32 bit
Element Count = 3
Frame Count = 8

20.2.5 Priority Queue

User can assign channels in to priority queues to facilitate request handling during arbitration. The port has two priority queues: a high and a low priority queue. Each queue can be configured to follow a fixed or rotating priority scheme. Fixed priority is such that the lower the channel number (Figure 20-9), the higher its priority. Rotating priority is based on a round-robin scheme. Initially, the priority list is sorted according to the fixed priority scheme. Channels assigned to the high priority queue are always serviced first according to the selected priority scheme before channels in the low priority queue are serviced. Table 20-2 describes how arbitration is performed according to different priority schemes.

NOTE: Since the DMA controller provides the capability to map any one of the hardware DMA request lines to any channel, the numerical order of the hardware DMA request does not imply any priority. The priority of each hardware DMA request is programmed and determined by software.

Figure 20-9. Fixed Priority Scheme



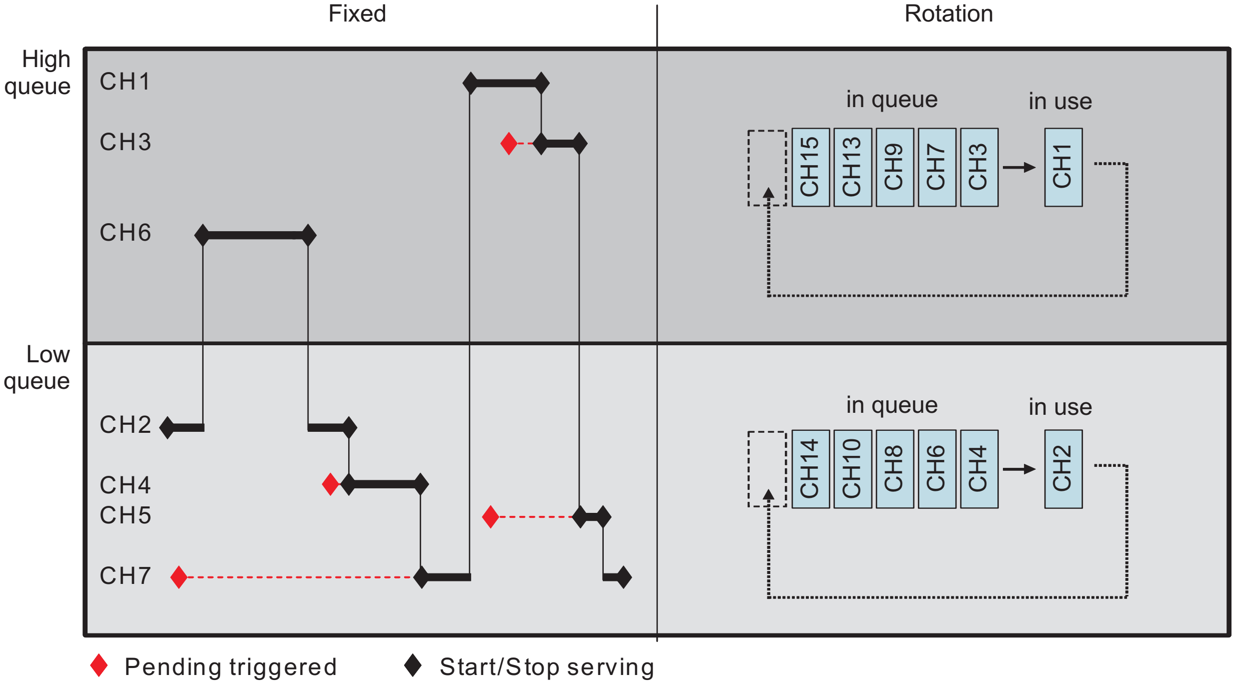
The above figure illustrates that by default Lower the channel number, higher the Priority.

Table 20-2. Arbitration According to Priority Queues and Priority Schemes

Queue	Priority Scheme	Remark
High priority	Fixed	Channels are serviced in an ascending order according to the channel number. The lower the channel number, the higher the priority. A channel will be arbitrated out whenever there is a higher pending channel. Otherwise a channel is completely serviced until its transfer count reaches zero before the next highest pending channel is serviced. When there is no pending channels left in high queue then the DMA switches to service low queue channels.
	Rotating	Channels are arbitrated by using the round-robin scheme. Arbitration is performed when the FIFO is empty. When there are no pending channels left in high queue then the DMA switches to service low queue channels.
Low priority	Fixed	Channels are serviced in an ascending order according to the channel number. The lower the channel number the higher the priority. A channel will be arbitrated out whenever there is a higher-priority pending channel. Otherwise a channel is completely serviced until its transfer count reaches zero, before the next highest pending channel is serviced. If there is a pending channel in the high-priority queue while DMA is servicing a low queue channel then DMA will switch back to service high queue channel after an arbitration boundary.
	Rotating	Channels are arbitrated by using round-robin scheme. Arbitration is performed when the FIFO is empty.

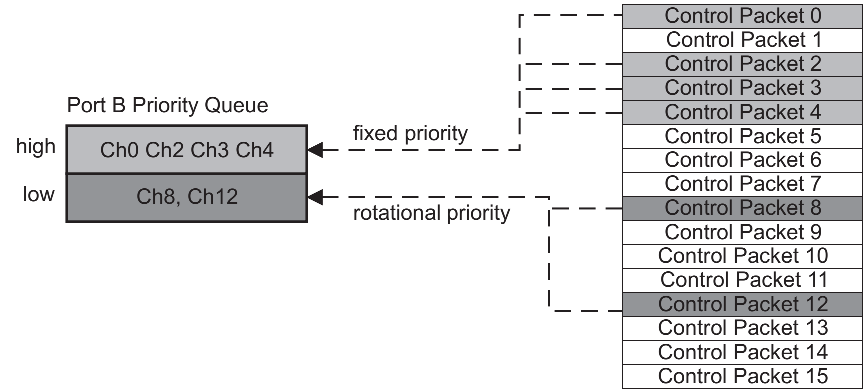
A Simple Priority Queues example in both Fixed and Rotation Scheme is shown in Figure 20-10.

Figure 20-10. Example of Priority Queues



For optimal system performance, the high priority channels should be put in fixed arbitration scheme and low priority channels in the rotating priority scheme as illustrated in Figure 20-11.

Figure 20-11. Example Channel Assignments



1 The above figure illustrates the channel assignments in a system with 16 channels. This approach can be scaled dependent on the total channels available.

20.2.6 Data Packing and Unpacking

The DMA controller automatically performs the necessary data packing and unpacking when the read element size differs from the write element size. Data packing is required when the read element size is smaller than the write element size; data unpacking is required when the read element size is larger than the write element size. When the read element size is equal to the write element size, no packing is performed during read, nor is any unpacking performed during write.

Figure 20-12 shows an example of data unpacking in which the DMA is used to transfer 128 transmit data elements to the MibSPI FIFO buffer. In this example, data unpacking is required because the read element size is 64 while the write element size is 16. The DMA first performs an 64-bit read from the source into its FIFO buffer. After the 64-bit data is read into the DMA FIFO buffer, it must unpack the data into four 16-bit data elements before writing out to the destination. Therefore the DMA would need to perform four 16 bit write operations to the destination.

NOTE: Examples are shown for big-endian scheme.

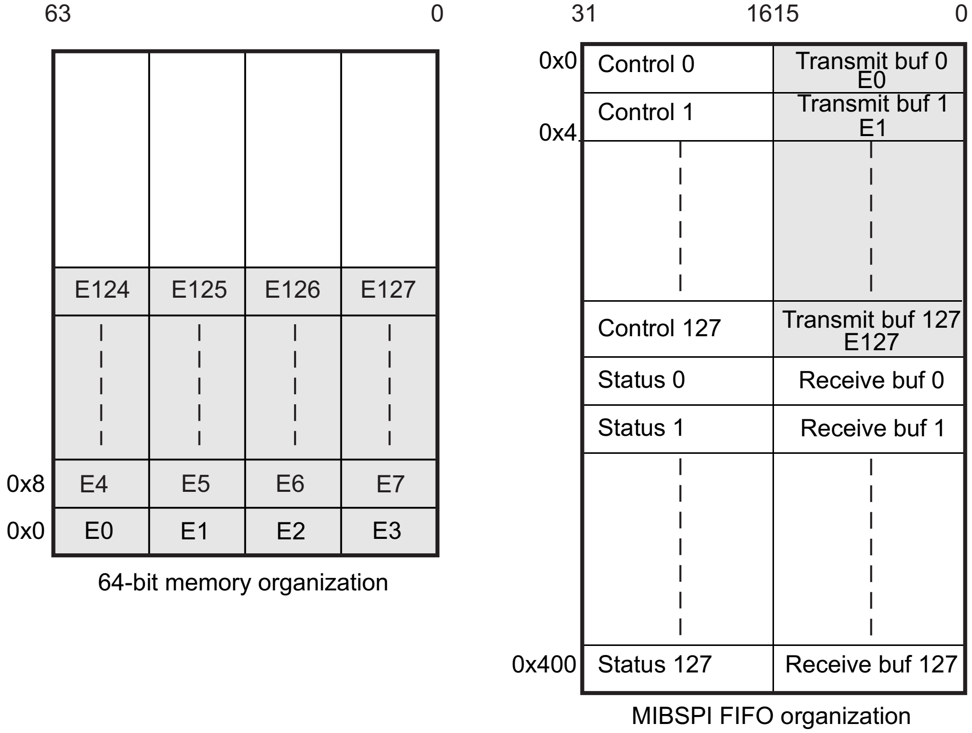
NOTE: In the example in Figure 20-12, to transmit data at the lower bits of the MibSPI, bits 15:0, the destination address should be incremented by a factor of 2.

NOTE: 1) The element Count (Section 20.3.2.3) refers only to the read element.

2) Data unpacking does not require the DMA request. Once the DMA request is received, data from Source is moved in to FIFO and unpacking happens until the FIFO is empty.

3) DMA assumes the destination is always ready and will perform write immediately. In case of data unpacking and Constant Addressing Mode write (Section 20.3.2.4 (1 - 0) = 0) the destination data will be overwritten by next data or next data might be skipped in case the destination has overflow protection (for example, SCITD register). User should configure DMA to avoid data unpacking if the Destination is configured as Constant Addressing Mode write to avoid data loss.

Figure 20-12. Example of DMA Data Unpacking



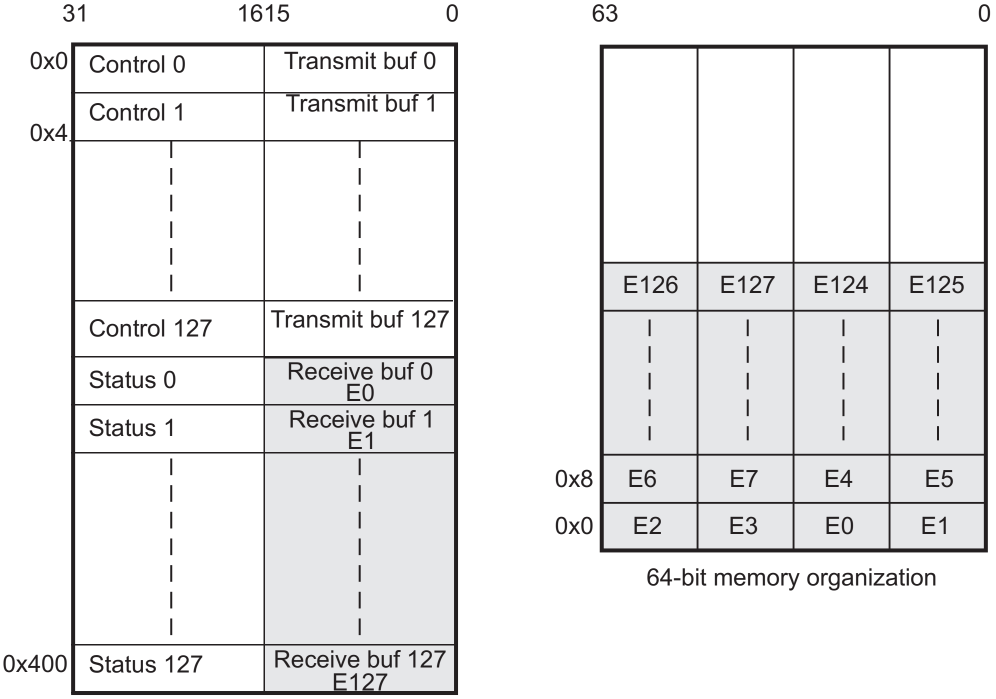
In this example, initialization of the MIBSPI FIFO is illustrated and assumes the following setup:

Read Element Size = 64 bit
Write Element Size = 16 bit
Element Count = 32
Frame Count = 1
Source Element Index = n/a, use post increment addressing mode
Source Frame Index = n/a, use post increment addressing mode
Destination Element Index = 4
Destination Frame Index = 0

When the read element size is smaller than the write element size, the DMA controller needs to perform data packing. The number of elements to pack is equal to the ratio between the write element size and read element size. In the example in Figure 20-13, the read element size is 16 bits and the write element size is 64 bits. The DMA controller would first pack the first four elements by performing four consecutive 16-bit read accesses of E0, E1, E2, and E3 into the first word of the DMA's internal FIFO. The DMA controller would then perform one single 64-bit write operation to transfer the data to the 64-bit destination memory.

Normally, the DMA controller carries out bus transactions on the bus according to the element size. For example, the DMA controller would perform a 16-bit read transaction if the read element size is programmed as 16 bits, or an 8-bit write transaction if the write element size is programmed as 8 bit. The exception is when the total transfer size is as defined in Equation 26 is not a multiple of the write element size.

Figure 20-13. Example of DMA Data Packing



MIBSPI FIFO organization

64-bit memory organization

In this example, a read of the MIBSPI FIFO is illustrated and assumes the following setup:

Read Element Size = 16 bit
Write Element Size = 64 bit
Element Count = 128
Frame Count = 1
Source Element Index = 4
Source Frame Index = 0
Destination Element Index = n/a, use post increment addressing mode
Destination Frame Index = n/a, use post increment addressing mode

For example, if the read element size is 8 bits, the element transfer count is equal to 9, and the write element size is 64 bit. The DMA controller would first perform eight 8-bit read transactions from the source. It would then perform a 64-bit write to the destination. When the same channel wins arbitration again, the DMA controller would first perform one 8-bit read from the source, followed by one 8-bit write to the destination, even though the write element size is 64 bit.

NOTE: Since peripherals are slower, it is advised to use data packing feature with caution for reading data from peripherals. Improper use might delay servicing other pending DMA channels.

20.2.7 DMA Request

There are three ways to start a DMA transfer:

- **Software request:** The transfer will be triggered by writing to SW Channel Enable Set and Status Register (Section 20.3.1.7). The software request can trigger either a block or a frame transfer depending on the setting of the TTYPE bit in the Channel Control Register (Section 20.3.2.4).
- **Hardware request**: The DMA controller can handle up to 48 DMA Request lines. A hardware request can trigger either a frame or a block transfer depending on the setting of the TTYPE bit in the Channel Control Register (Section 20.3.2.4).
- **Triggered by other control packet:** When a control packet finishes the programmed number of transfers it can trigger another channel to initiate its transfers.

Each time a DMA request is made, either one frame transfer or one block transfer can be chosen. An active DMA request signal will trigger a DMA transaction.

The DMA controller has a two-level buffer to capture HW requests per channel. When a HW request is generated and the channel is enabled, the corresponding bit in the DMA Status Register (Section 20.3.1.3) is set. The pending register acts as a first-level buffer. Typically, a peripheral acting as a source of a transfer could initiate another request after its data registers have been read out by DMA, even though that data has not been completely transferred to the destination. If a second HW request is generated by the peripheral, the DMA controller has an extra request buffer to capture this second request and service it after the first request is complete.

NOTE: The DMA cannot capture more than two requests at the same time. Additional requests are ignored until at least one pending request is completely processed.

The DMA controller also supports a mix of hardware and software requests on the same channel. Note that such interchangeable usage may result into an out of sync for DMA channel and peripheral. The application needs to be careful as the DMA does not have a built-in mechanism to protect against this loss of synchronization.

If a software request is generated, the corresponding bit in the Channel Pending Register (Section 20.3.1.2) is set accordingly. If the pending request is not completely serviced by the DMA and a hardware request is generated by a peripheral onto the same channel, the DMA will capture and recognize this hardware request into its request buffer.

NOTE: The DMA controller cannot recognize two software requests on the same channel if the first software request is still pending. If such a request occurs, the DMA will discard it. Therefore, the user software should check the pending register before issuing a new software request.

The DMA module on this microcontroller has 32 channels and up to 48 hardware DMA requests. The module contains DREQASIx registers which are used to map the DMA requests to the DMA channels. By default, channel 0 is mapped to request 0, channel 1 to request 1, and so on.

Some DMA requests have multiple sources, see Table 20-3. The application must ensure that only one of these DMA request sources is enabled at any time.

Table 20-3. DMA Request Line Connection

Modules	DMA Request Sources	DMA Request
MIBSPI1	MIBSPI1[1] [(1)]	DMAREQ[0]
MIBSPI1	MIBSPI1[0] [(2)]	DMAREQ[1]
MIBSPI2	MIBSPI2[1] [(1)]	DMAREQ[2]
MIBSPI2	MIBSPI2[0] [(2)]	DMAREQ[3]
MIBSPI1 / MIBSPI3 / DCAN2	MIBSPI1[2] / MIBSPI3[2] / DCAN2 IF3	DMAREQ[4]
MIBSPI1 / MIBSPI3 / DCAN2	MIBSPI1[3] / MIBSPI3[3] / DCAN2 IF2	DMAREQ[5]
DCAN1 / MIBSPI5	DCAN1 IF2 / MIBSPI5[2]	DMAREQ[6]
MIBADC1 / MIBSPI5	MIBADC1 event / MIBSPI5[3]	DMAREQ[7]

[(1)] SPI1, SPI2, SPI3, SPI4, SPI5 receive in compatibility mode

[(2)] SPI1, SPI2, SPI3, SPI4, SPI5 transmit in compatibility mode

Table 20-3. DMA Request Line Connection (continued)

Modules	DMA Request Sources	DMA Request
MIBSPI1 / MIBSPI3 / DCAN1	MIBSPI1[4] / MIBSPI3[4] / DCAN1 IF1	DMAREQ[8]
MIBSPI1 / MIBSPI3 / DCAN2	MIBSPI1[5] / MIBSPI3[5] / DCAN2 IF1	DMAREQ[9]
MIBADC1 / I2C / MIBSPI5	MIBADC1 G1 / I2C receive / MIBSPI5[4]	DMAREQ[10]
MIBADC1 / I2C / MIBSPI5	MIBADC1 G2 / I2C transmit / MIBSPI5[5]	DMAREQ[11]
RTI / MIBSPI1 / MIBSPI3	RTI DMAREQ0 / MIBSPI1[6] / MIBSPI3[6]	DMAREQ[12]
RTI / MIBSPI1 / MIBSPI3	RTI DMAREQ1 / MIBSPI1[7] / MIBSPI3[7]	DMAREQ[13]
MIBSPI3 / MibADC2 / MIBSPI5	MIBSPI3[1][1] / MibADC2 event / MIBSPI5[6]	DMAREQ[14]
MIBSPI3 / MIBSPI5	MIBSPI3[0][2] / MIBSPI5[7]	DMAREQ[15]
MIBSPI1 / MIBSPI3 / DCAN1 / MibADC2	MIBSPI1[8] / MIBSPI3[8] / DCAN1 IF3 / MibADC2 G1	DMAREQ[16]
MIBSPI1 / MIBSPI3 / DCAN3 / MibADC2	MIBSPI1[9] / MIBSPI3[9] / DCAN3 IF1 / MibADC2 G2	DMAREQ[17]
RTI / MIBSPI5	RTI DMAREQ2 / MIBSPI5[8]	DMAREQ[18]
RTI / MIBSPI5	RTI DMAREQ3 / MIBSPI5[9]	DMAREQ[19]
NHET1 / NHET2 / DCAN3	NHET1 DMAREQ[4] / NHET2 DMAREQ[4] / DCAN3 IF2	DMAREQ[20]
NHET1 / NHET2 / DCAN3	NHET1 DMAREQ[5] / NHET2 DMAREQ[5] / DCAN3 IF3	DMAREQ[21]
MIBSPI1 / MIBSPI3 / MIBSPI5	MIBSPI1[10] / MIBSPI3[10] / MIBSPI5[10]	DMAREQ[22]
MIBSPI1 / MIBSPI3 / MIBSPI5	MIBSPI1[11] / MIBSPI3[11] / MIBSPI5[11]	DMAREQ[23]
NHET1 / NHET2 / MIBSPI4 / MIBSPI5	NHET1 DMAREQ[6] / NHET2 DMAREQ[6] / MIBSPI4[1] [1]/ MIBSPI5[12]	DMAREQ[24]
NHET1 / NHET2 / MIBSPI4 / MIBSPI5	NHET1 DMAREQ[7] / NHET2 DMAREQ[7] / MIBSPI4[0][2] / MIBSPI5[13]	DMAREQ[25]
CRC1 / MIBSPI1 / MIBSPI3	CRC1 DMAREQ[0] / MIBSPI1[12] / MIBSPI3[12]	DMAREQ[26]
CRC1 / MIBSPI1 / MIBSPI3	CRC1 DMAREQ[1] / MIBSPI1[13] / MIBSPI3[13]	DMAREQ[27]
LIN1 / MIBSPI5	LIN1 receive / MIBSPI5[14]	DMAREQ[28]
LIN1 / MIBSPI5	LIN1 transmit / MIBSPI5[15]	DMAREQ[29]
MIBSPI1 / MIBSPI3 / SCI3 / MIBSPI5	MIBSPI1[14] / MIBSPI3[14] / SCI3 receive / MIBSPI5[1][1]	DMAREQ[30]
MIBSPI1 / MIBSPI3 / SCI3 / MIBSPI5	MIBSPI1[15] / MIBSPI3[15] / SCI3 transmit / MIBSPI5[0][2]	DMAREQ[31]
I2C2 / ePWM1 / MIBSPI2 / MIBSPI4 / GIOA	I2C2 receive / ePWM1_SOCA / MIBSPI2[2] / MIBSPI4[2] / GIOA[0]	DMAREQ[32]
I2C2 / ePWM 1 / MIBSPI2 / MIBSPI4 / GIOA	I2C2 transmit / ePWM1_SOCB / MIBSPI2[3] / MIBSPI4[3] /GIOA[1]	DMAREQ[33]
ePWM2 / MIBSPI2 / MIBSPI4 / GIOA	ePWM2_SOCA / MIBSPI2[4] / MIBSPI4[4] / GIOA[2]	DMAREQ[34]
ePWM2 / MIBSPI2 / MIBSPI4 / GIOA	ePWM2_SOCB / MIBSPI2[5] / MIBSPI4[5] / GIOA[3]	DMAREQ[35]
ePWM3 / MIBSPI2 / MIBSPI4 / GIOA	ePWM3_SOCA / MIBSPI2[6] / MIBSPI4[6] / GIOA[4]	DMAREQ[36]
ePWM3 / MIBSPI2 / MIBSPI4 / GIOA	ePWM3_SOCB / MIBSPI2[7] / MIBSPI4[7] / GIOA[5]	DMAREQ[37]
CRC2 / ePWM4 / MIBSPI2 / MIBSPI4 / GIOA	CRC2 DMAREQ[0] / ePWM4_SOCA / MIBSPI2[8] / MIBSPI4[8] / GIOA[6]	DMAREQ[38]
CRC2 / ePWM4 / MIBSPI2 / MIBSPI4 /GIOA	CRC2 DMAREQ[1] / ePWM4_SOCB / MIBSPI2[9] / MIBSPI4[9] / GIOA[7]	DMAREQ[39]
LIN2 / ePWM5 / MIBSPI2 / MIBSPI4 / GIOB	LIN2 receive / ePWM5_SOCA / MIBSPI2[10] / MIBSPI4[10] / GIOB[0]	DMAREQ[40]
LIN2 / ePWM5 / MIBSPI2 / MIBSPI4 / GIOB	LIN2 transmit / ePWM5_SOCB / MIBSPI2[11] / MIBSPI4[11] / GIOB[1]	DMAREQ[41]
SCI4 / ePWM6 / MIBSPI2 / MIBSPI4 / GIOB	SCI4 receive / ePWM6_SOCA / MIBSPI2[12] / MIBSPI4[12] / GIOB[2]	DMAREQ[42]
SCI4 / ePWM6 / MIBSPI2 / MIBSPI4 / GIOB	SCI4 transmit / ePWM6_SOCB / MIBSPI2[13] / MIBSPI4[13] / GIOB[3]	DMAREQ[43]
ePWM7 / MIBSPI2 / MIBSPI4 / GIOB	ePWM7_SOCA / MIBSPI2[14] / MIBSPI4[14] / GIOB[4]	DMAREQ[44]
ePWM7 / MIBSPI2 / MIBSPI4 / GIOB / DCAN4	ePWM7_SOCB / MIBSPI2[15] / MIBSPI4[15] / GIOB[5] / DCAN4 IF1	DMAREQ[45]
GIOB / DCAN4	GIOB[6] / DCAN4_IF2	DMAREQ[46]
GIOB / DCAN4	GIOB[7] / DCAN4_IF3	DMAREQ[47]

20.2.8 Auto-Initiation

When Auto-initiation Mode (AIM) bit of Channel Control Register (Section 20.3.2.4) is enabled for a channel and the channel is triggered by a software request for a block transfer, the channel will restart again using the same channel information stored at the respective control packet after one block transfer is completed. In the case of Hardware Request the channel needs to be retriggered each time after a block is complete even if auto-initiation is enabled.

20.2.9 Interrupts

Each channel can be configured to generate interrupts on several transfer conditions:

- Frame transfer complete (FTC) interrupt: an interrupt is issued after the last element of a frame has been transferred.
- Last frame transfer started (LFS) interrupt: an interrupt is issued before the first element of the last frame of a block transfer has started.
- First half of block complete (HBC) interrupt: an interrupt is issued if more than half of the block is transferred.
  - If the number of frames *n* is odd, then the HBC interrupt is generated at the end of the frame when *(n+1) / 2* number of frames are left in the block.
  - If the number of frames *n* is even, then the HBC interrupt is generated at the end of the frame after *n/2* number of frames are left in the block.
- Block transfer complete (BTC) interrupt: an interrupt is issued after the last element of the last frame has been transferred.
- External imprecise error on read: an interrupt can be issued when a bus error (Illegal transaction with ok response) is detected. The imprecise read error is connected to the ESM module.
- External imprecise error on write: an interrupt can be issued when a bus error (Illegal transaction with ok response) is detected. The imprecise write error is connected to the ESM module.
- Memory Protection Unit error (MPU): an interrupt is issued when the DMA detects that the access falls outside of a memory region programmed in the MPU registers of the DMA. The MPU interrupt is connected to the ESM module.
- Parity error (PAR): an interrupt is issued when the DMA detects a parity error when reading one of the control packets. The PAR interrupt is connected to the ESM module.

The DMA outputs 5 interrupt lines for control packet handling, a parity interrupt and a memory protection interrupt (Figure 20-14). Each type of transfer interrupt condition is grouped together. For example, all block-transfer complete interrupts that are routed to a port are combined (ORed). The channel that caused the interrupt is given in the corresponding interrupt channel offset register. Priority between interrupts among the same interrupt type is resolved by a fixed priority scheme. Priority between different interrupt types is resolved in the Vector Interrupt Manager. Figure 20-15 explains the Frame Transfer Complete Interrupt structure in detail.

NOTE: Each Channel Specific interrupts in DMA module are routed towards Group A or B to support two different CPUs individually. For devices with Single CPU or Dual CPU, where both CPUs are running same code in delayed lock-step as safety feature:

Group A - Interrupts (FTC, LFS, HBC, and BTC) are routed to the ARM CPU.

Group B - Interrupts (FTC, LFS, HBC, and BTC) are not routed out.

User software should configure only Group A interrupts.

Figure 20-14. DMA Interrupts

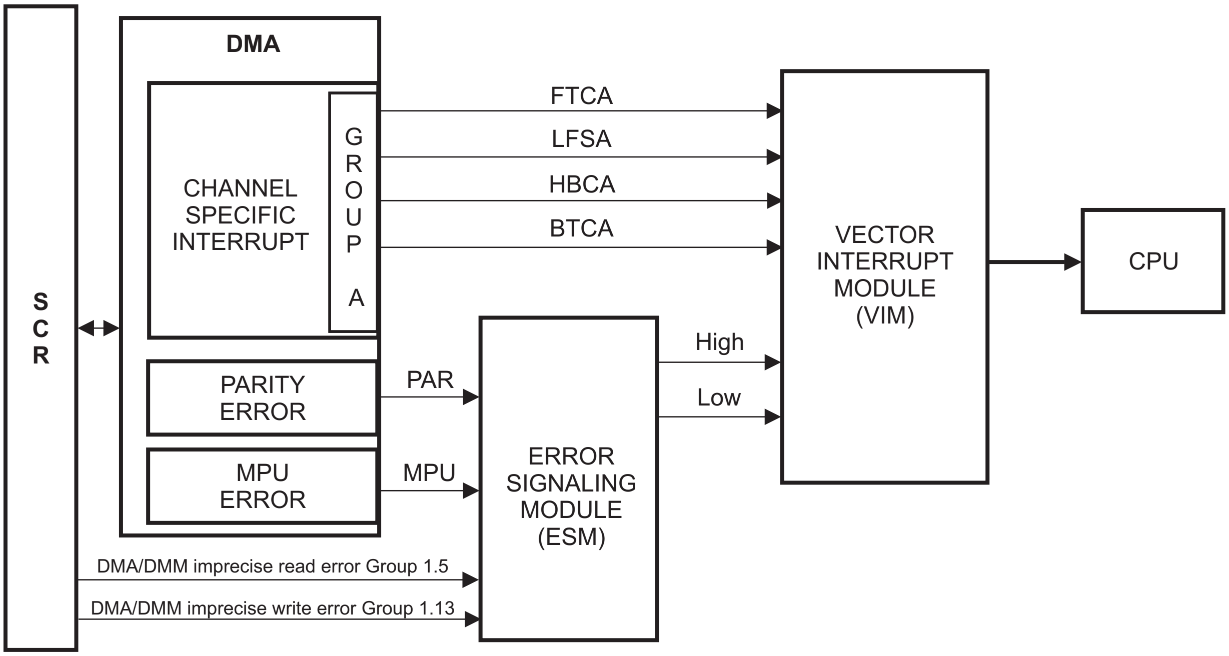
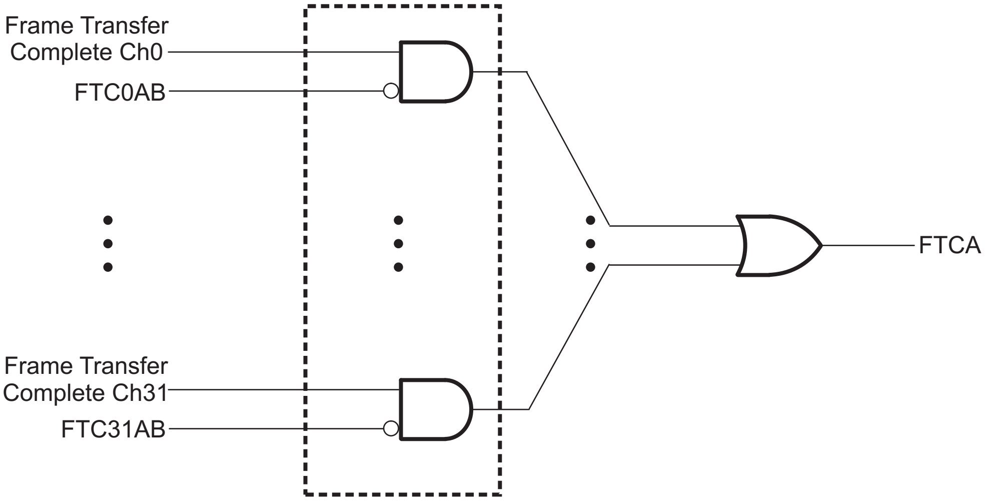

DMA
CHANNEL SPECIFIC INTERRUPT
GROUP A
PARITY ERROR
MPU ERROR
SCR
FTCA
LFSA
HBCA
BTCA
PAR
MPU
ERROR SIGNALING MODULE (ESM)
High
Low
VECTOR INTERRUPT MODULE (VIM)
CPU
DMA/DMM imprecise read error Group 1.5
DMA/DMM imprecise write error Group 1.13

Figure 20-15. Detailed Interrupt Structure (Frame Transfer Complete Path)



This figure is applicable for the HBC, LFS, and BTC interrupt.

20.2.10 Debugging

The DMA supports four different behaviors in suspend mode. These behaviors can be configured by the user as per the application requirement.

- Immediate stop at a DMA channel arbitration boundary. Please refer to Table 20-4 and Table 20-5 for arbitration boundary definition.
- Finish current frame transfer and continue after suspend ends.
- Finish current block transfer and continue after suspend ends.
- Ignore the suspend. The DMA continues to be operational as in functional mode when debug mode is active.

When the DMA controller enters suspend mode, it continues to sample incoming hardware DMA requests, but the Channel Pending Register (Section 20.3.1.2) is frozen from being updated. After the suspend ends, all new requests that were received during suspend mode are reflected in the Channel Pending Register (Section 20.3.1.2).

Except when the DMA controller is configured to ignore suspend mode, no channel arbitration is performed during suspend mode. The current channel under which suspend mode was entered will finish its entire frame or block-transfer after suspend mode ends, depending how the debug option was chosen.

To facilitate debugging, a Watch Point Register (Section 20.3.1.54) and a Watch Mask Register (Section 20.3.1.55) are used. The watch point register together with the watch mask register can be configured to watch for a unique address or a range of addresses. When the condition to watch is true, the DMA freezes its state and generates a debug request signal to the host CPU so the state of the DMA can be examined.

20.2.11 Power Management

The DMA offers two power-management modes: run and sleep. In run mode, the DMA is fully operational.

The sleep mode shuts down the DMA if no pending channels are waiting to be serviced. If a DMA request is received or a software request is generated by the user software, then the DMA wakes up immediately.

The sleep mode may be used to optimize the DMA module power consumption.

When the system module issues a global low power mode request, the DMA will respond to the system module with an acknowledge as soon as an arbitration boundary is reached. If no DMA requests are pending, it will respond with an acknowledge immediately.

NOTE: When the DMA is in global low power mode, the clock is stopped and therefore it cannot detect any DMA request. The device must be woken up before a peripheral can generate a DMA request.

20.2.12 FIFO Buffer

DMA FIFO is 4 levels deep and 64-bit wide (can hold up to 4 × 64-bits of data). They are used for Data packing and unpacking.

The DMA FIFO has two states:

- EMPTY: The FIFO contains no data.
- FULL: The FIFO is filled or the element count has reached zero; the read operation has to be stopped.

DMA channels can only be switched when the FIFO is empty. This also implies that arbitration between channels is done when the FIFO is empty.

The DMA has two FIFOs, FIFO A and FIFO B, each executing a channel that provides the capability to execute a maximum of two channels concurrently.

The FIFO buffer may be bypassed through the use of the bypass feature in the port control register; see Port Control Register (Section 20.3.1.51) for register details. Writing 1 to this bit limits the FIFO depth to the size of one element. That means if the read element size is equal to or larger than the write element size, after one element is read the write out to the destination starts. Otherwise, the write out to the destination starts after enough reads have completed to do one write of the write element size. This feature is particularly useful to minimize switching latency in-between channels. When bypass mode is enabled, the DMA performs minimal transfers within an arbitration boundary. In addition, the bypass feature allows arbitration between channels that can be carried out at a source element granularity.

However, it has to be considered that while in bypass mode, the DMA controller does not make optimal use of the bus bandwidth. Since the read and write element sizes can be different, then the number of read and write transactions will be different. Table 20-4 and Table 20-5 show a comparison between the number of read and write transactions performed by the DMA controller from one channel to another before arbitration in non-bypass and bypass mode.

Table 20-4. Maximum Number of DMA Transactions per Channel in Non-Bypass Mode

	Write Element Size	**8 bit**		**16 bit**		**32 bit**		**64 bit**	
Read Element Size	**8 bit**	4 read	4 write	4 read	2 write	4 read	1 write	8 read	1 write
	16 bit	2 read	4 write	4 read	4 write	4 read	2 write	4 read	1 write
	32 bit	1 read	4 write	2 read	4 write	4 read	4 write	4 read	2 write
	64 bit	1 read	8 write	1 read	4 write	2 read	4 write	4 read	4 write

Table 20-5. Maximum Number of DMA Transactions per Channel in Bypass Mode

	Write Element Size	**8 bit**		**16 bit**		**32 bit**		**64 bit**	
Read Element Size	**8 bit**	1 read	1 write	2 read	1 write	4 read	1 write	8 read	1 write
	16 bit	1 read	2 write	1 read	1 write	2 read	1 write	4 read	1 write
	32 bit	1 read	4 write	1 read	2 write	1 read	1 write	2 read	1 write
	64 bit	1 read	8 write	1 read	4 write	1 read	2 write	1 read	1 write

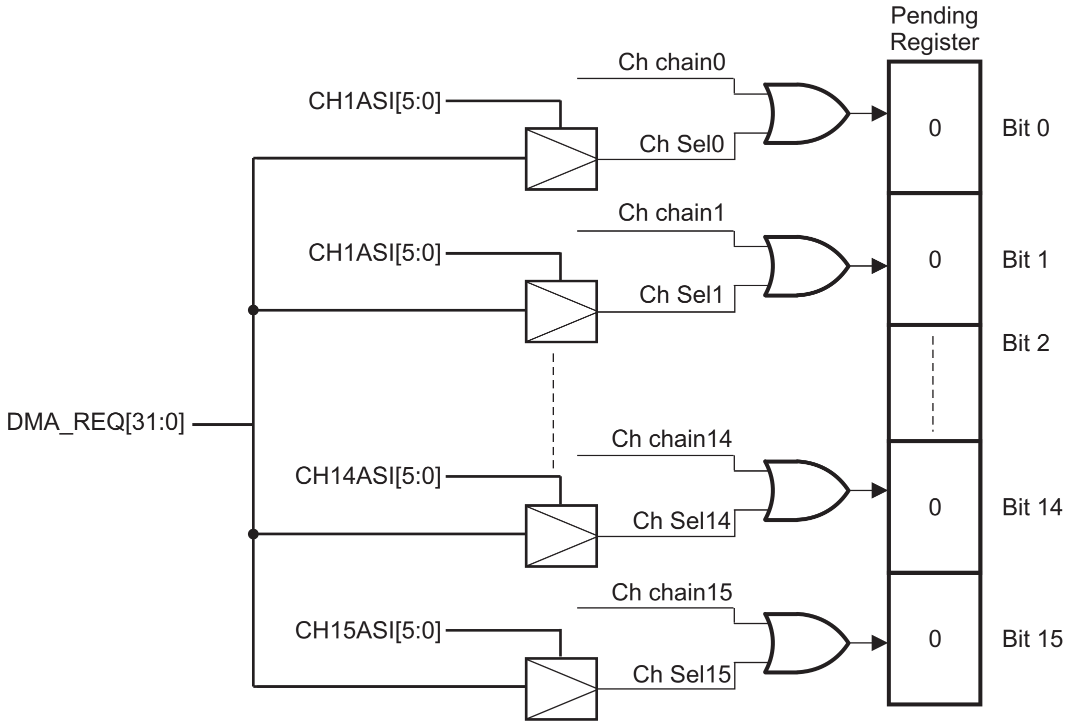
20.2.13 Channel Chaining

Channel chaining is used to trigger a single or multiple channels with out an external DMA request. This is possible by chaining one control packet to other. Chain[5:0] field of the Channel Control Register (Section 20.3.2.4) is used to program the chaining control packet. Chained control packets follow arbitration rules within the pending register. For example if CH1, CH2, CH4, CH5 are triggered together and CH3 is chained with CH1. The order of channels serviced in spite of chaining will be CH1 -> CH2 -> CH3 -> CH4 -> CH5.

In order to setup up channel chain feature, the Channel Control Register (Section 20.3.2.4) needs to be enabled for all chained channels before triggering first DMA request.

Figure 20-16 illustrates how internally chained request is generated after completing the required transfers and stored in pending register. In this example CH1 is Chained to CH0. When CH0 is triggered CH1 is captured as pending in the Channel Pending Register (Section 20.3.1.2) even when it is not triggered.

Figure 20-16. Example of Channel Chaining



20.2.14 Request Polarity

DMA supports both active high and active low hardware requests. This is configured through the registers DMAREQPS1 and DMAREQPS0.

The selection of request polarity should be done at the start of the program. In order to change the request polarity from active high to active low for a channel following sequence should be followed:

1. Disable channel for which polarity is to be changed using the HWCHENA bit.
2. Disable the peripheral in order that it may set the request line to inactive high state (since by default requests are active high).
3. Apply software reset to the DMA using the GCTRL register.
4. Program the request polarity for the channel.
5. Re-enable the DMA channel.
6. Re-enable the peripheral that triggers the DMA event.

20.2.15 Memory Protection

The DMA controller is capable of access to the full address range of the device. The protection mechanism allows the protection of multiple memory regions to restrict accesses to those address ranges. This will allow the application to protect critical application data from unintentionally being accessed by the DMA controller.

20.2.15.1 Protection Mechanism

The memory protection mechanism consists of the access privilege for a given memory region, the start and end address for the region, and notification of an access violation for the protected region.

Each region to be protected is configured by software by writing the start address and end address for each region into the DMA Memory Protection Registers, DMAMPRxS and DMAMPRxE. The definition of these registers can be found starting at Section 20.3.1.64. Any region in the valid address space can be protected from inappropriate accesses.

The access privileges can be set to one of four permission settings as shown below:

- Full access
- Read only access
- Write only access
- No access

The permissions for a given region are selected by writing the appropriate values in the DMA Memory Protection Control Register (Section 20.3.1.64).

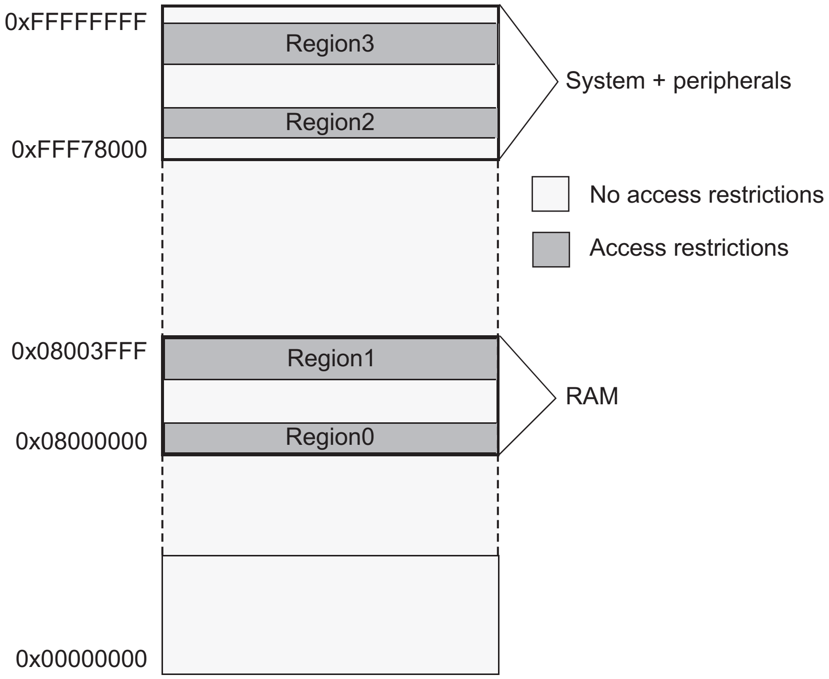
A region of memory not configured for access settings by the registers has "Full Access" privileges.

NOTE: If the regions defined by the start and end addresses overlap, the region defined first in the register space determines the access privilege. For example, if region 0 and region 1 overlap, the access permissions defined for region 0 will take precedence since region 0 registers are before region 1.

In a case where a memory protection violation occurs, a flag will be set and an interrupt will be generated, if interrupts are enabled. The DMA Memory Protection Status Register (Section 20.3.1.65) contains the status flags for the memory protection mechanism, and the DMA Memory Protection Control Register (Section 20.3.1.64) contains the interrupt enable bits. Upon detection of the memory protection violation, the DMA Channel that caused the violation will be stopped and the next available DMA channel will be serviced.

Figure 20-17 Illustrates a protection mechanism.

Figure 20-17. Example of Protection Mechanism



20.2.16 ECC Checking

The Control packet RAM is protected using a Single Error Correction Double Error Detection (SECDED) scheme. This scheme is implemented using a total of 9 ECC check bits for every 128 bits of data stored in the DMA Control Packet RAM.

ECC checking can be enabled and disabled within the module by a 4-bit key. The key is located in the ECC Control Register (Section 20.3.1.62).

During write accesses to Control Packet RAM, ECC bits are generated automatically and stored along with the data bits to the memory.

During read accesses from the Control Packet RAM, the ECC bits in memory are checked against a computed ECC value for the 128 bits of data. Following two kinds of errors can occur during the read:

- Single-Bit Error - If a single-bit error occurs during the reads to the control packet either by the CPU or by DMA logic and the EDCAMODE[3:0] in DMASECCCTRL register is 0xA, the error is automatically corrected. The SBEFLG bit in the register is also set to 1 to indicate a single-bit error was corrected. The DMAECCSBE register is updated to indicate the error address. In addition, if the SBE_EVT_EN[3:0] in DMASECCCTRL register is 0xA, the error is also indicated to ESM.
- Double-Bit Error - If a double-bit error occurs during the reads to the control packet either by the CPU or the DMA logic and the ECC_ENA[3:0] in DMAPECR register is 0xA, the error is indicated to ESM. The EDFLG bit gets set and the error address is stored in DMAPAR register.

The DMA module automatically performs read-modify-write operations to the Control Packet RAM which are required during CPU configuration of the control packet RAM. Errors occuring during these reads are also covered by the SECDED scheme. Also, reads to the Working Packet by CPU or DMA logic and writes to the Working Packet by the DMA logic are also protected by SECDED.

During double-bit errors, it is possible to configure the behavior of the channel using the ERRA bit in DMAPECR register. Two options are available:

- If ERRA bit is cleared, errors are ignored and channel operation will resume normally.
- If ERRA bit is set, errors will cause the DMA to be disabled (DMA_EN bit in GCTRL register is cleared). All channels will stop servicing at the next arbitration boundary. This action will be taken regardless of the origin of error being a CPU read or a DMA logic read.

20.2.17 ECC Testing

The ECC RAM is accessible to allow manually inserting faults so that the ECC checking feature can be tested. Test mode is entered by asserting the TEST bit in the ECC Control Register (Section 20.3.1.62). Once the bit is set, the ECC bits are mapped to the control packet RAM starting address A00h. The sequence to test the ECC is:

1. Write the data location of the Control Packet RAM while keeping ECC_ENA active. The ECC bits will get automatically written with the correct values in this step.
2. Enable ECC test mode by setting the TEST bit of the DMAPECR register.
3. To test single-bit error correction capability, read back one of the data written earlier, flip one of the bits and write it back. The same could be done for the ECC bits as well.
4. Similarly, to test double-bit detection capability, read back one of the data written earlier, flip two bits and write it back. The same could be done for the ECC bits as well.
5. Now read back the same data bits that were corrupted or for which the ECC was corrupted in the earlier steps 3-4.
6. Depending on the kind of corruption created, for double-bit error, read EDFLG and error address captured in DMAPAR; similarly for single-bit error, read SBERR in DMASECCCTRL and error address in DMAECCSBE.
7. The check is successful if the flag and error address are updated successfully.
8. Clear the flags (EDFLG or SBERR as applicable) and read the error address.
9. To exit the test mode, initialize the data and ECC that were corrupted earlier, back to their original values.
10. Finally, clear the TEST bit of the DMAPECR register.

NOTE: When in test mode, no ECC checking will be done when reading from ECC memory, but ECC checking will be performed on the normal memory.

This offsets in Table 20-6 must be used to run the ECC diagnostics.

Table 20-6. ECC Mapping

Offset	ECC of Control Packet (Only 9 bits are valid in the read)
A00h	0 (Lower 128 bits)
A04h	0 (Upper 128 bits)
A08h	1 (Lower 128 bits)
:	:
AFCh	31 (Upper 128 bits)

20.2.18 Initializing RAM with ECC

After power up, the RAM content including the ECC bits cannot be guaranteed. To avoid ECC failures when reading RAM, the RAM has to be initialized. The RAM can be initialized by writing known values into it. When the known value is written, the corresponding ECC bit will be automatically calculated and updated.

Another possibility to initialize the memory is to follow the Auto-Initialization of On-Chip SRAM Modules subsection in the *Architecture* chapter. The RAM will be initialized to 0. Depending on the even/odd parity selection, the parity bit will be calculated accordingly.

To allow for ECC calculation during initialization, the ECC functionality has to be enabled as discussed in Section 20.2.16.

20.2.19 Transaction Errors

DMA generates parity for all transactions and checks parity for responses to the transactions. Note that this feature is distinct from the ECC checking for the Control Packet RAM.

If a parity error is detected in these transactions and TER_EN bit in TERECTRL register is enabled, DMA will stop processing the current channel at the arbitration boundary and will update TER_ERR flag. The offset of the channel during which the parity failure was detected will get captured in the TERROFFSET register. Also, the error is indicated to the ESM module. This is shown in Figure 20-18.

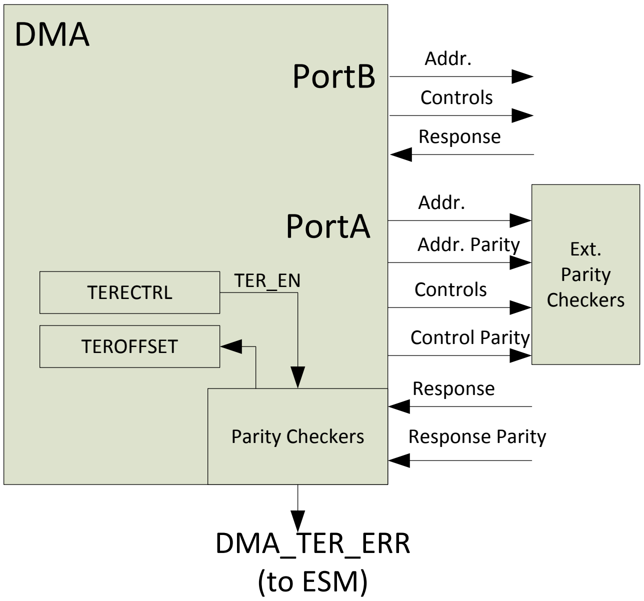
Since the channel stops due to an error and likely the peripheral and the DMA are out of synchronization, it is recommended to follow the sequence below to resume the channel:

1. Read the TEROFFSET register to find the channel number causing the transaction error. The register automatically clears to 0 once read.
2. Clear the TER_ERR flag by writing 1 to the flag.
3. Disable the peripheral that triggered the DMA event.
4. Reinitialize the control packet. Note that this does not change the channel's HWCHEN bit.
5. Re-enable the peripheral to trigger the DMA event.
6. Re-enable the DMA channel (which was previously cleared by the DMA logic due to the error).

In certain cases, it is possible that DMA sets the TER_ERR flag without updating the TEROFFSET register. This occurs due to parity errors when no channels are active. The recovery sequence in this case is to clear the TER_ERR flag.

NOTE: Handling of a parity error at a system level may require additional operations that are not detailed here.

Figure 20-18. DMA Transaction Parity



NOTE: Only PortA supports transaction parity

20.3 Control Registers and Control Packets

The DMA control registers are summarized in Table 20-7. The base address for the control registers is FFFF F000h. The control packets are summarized in Table 20-8. The base address for the control packets is FFF8 0000h. Each register begins on a word boundary. All registers and control packets are accessible in 8, 16, and 32 bit.

NOTE: The register definitions are given for a full DMA module configuration (32 channels, 64 requests, 2 Ports, Dual CPU support). Writes and Reads of bits pertaining to features not included in the DMA implementation as defined in the device-specific data manual are possible without error; however, they will have no affect on device operation.

Table 20-7. DMA Control Registers

Offset	Acronym	Register Description	Section
00h	GCTRL	Global Control Register	Section 20.3.1.1
04h	PEND	Channel Pending Register	Section 20.3.1.2
0Ch	DMASTAT	DMA Status Register	Section 20.3.1.3
10h	DMAREVID	DMA revision ID Register	Section 20.3.1.4
14h	HWCHENAS	HW Channel Enable Set and Status Register	Section 20.3.1.4
1Ch	HWCHENAR	HW Channel Enable Reset and Status Register	Section 20.3.1.6
24h	SWCHENAS	SW Channel Enable Set and Status Register	Section 20.3.1.7
2Ch	SWCHENAR	SW Channel Enable Reset and Status Register	Section 20.3.1.8
34h	CHPRIOS	Channel Priority Set Register	Section 20.3.1.9
3Ch	CHPRIOR	Channel Priority Reset Register	Section 20.3.1.10
44h	GCHIENAS	Global Channel Interrupt Enable Set Register	Section 20.3.1.11
4Ch	GCHIENAR	Global Channel Interrupt Enable Reset Register	Section 20.3.1.12
54h	DREQASI0	DMA Request Assignment Register 0	Section 20.3.1.13
58h	DREQASI1	DMA Request Assignment Register 1	Section 20.3.1.14
5Ch	DREQASI2	DMA Request Assignment Register 2	Section 20.3.1.15
60h	DREQASI3	DMA Request Assignment Register 3	Section 20.3.1.16
64h	DREQASI4	DMA Request Assignment Register 4	Section 20.3.1.13
68h	DREQASI5	DMA Request Assignment Register 5	Section 20.3.1.13
6ch	DREQASI6	DMA Request Assignment Register 6	Section 20.3.1.13
70h	DREQASI7	DMA Request Assignment Register 7	Section 20.3.1.13
94h	PAR0	Port Assignment Register 0	Section 20.3.1.21
98h	PAR1	Port Assignment Register 1	Section 20.3.1.22
9Ch	PAR2	Port Assignment Register 2	Section 20.3.1.23
A0h	PAR3	Port Assignment Register 3	Section 20.3.1.24
B4h	FTCMAP	FTC Interrupt Mapping Register	Section 20.3.1.25
BCh	LFSMAP	LFS Interrupt Mapping Register	Section 20.3.1.26
C4h	HBCMAP	HBC Interrupt Mapping Register	Section 20.3.1.27
CCh	BTCMAP	BTC Interrupt Mapping Register	Section 20.3.1.28
DCh	FTCINTENAS	FTC Interrupt Enable Set Register	Section 20.3.1.29
E4h	FTCINTENAR	FTC Interrupt Enable Reset Register	Section 20.3.1.30
ECh	LFSINTENAS	LFS Interrupt Enable Set Register	Section 20.3.1.31
F4h	LFSINTENAR	LFS Interrupt Enable Reset Register	Section 20.3.1.32
FCh	HBCINTENAS	HBC Interrupt Enable Set Register	Section 20.3.1.33
104h	HBCINTENAR	HBC Interrupt Enable Reset Register	Section 20.3.1.34
10Ch	BTCINTENAS	BTC Interrupt Enable Set Register	Section 20.3.1.35
114h	BTCINTENAR	BTC Interrupt Enable Reset Register	Section 20.3.1.36
11Ch	GINTFLAG	Global Interrupt Flag Register	Section 20.3.1.37

Table 20-7. DMA Control Registers (continued)

Offset	Acronym	Register Description	Section
124h	FTCFLAG	FTC Interrupt Flag Register	Section 20.3.1.38
12Ch	LFSFLAG	LFS Interrupt Flag Register	Section 20.3.1.39
134h	HBCFLAG	HBC Interrupt Flag Register	Section 20.3.1.40
13Ch	BTCFLAG	BTC Interrupt Flag Register	Section 20.3.1.41
144h	BERFLAG	BER Interrupt Flag Register	Section 20.3.1.42
14Ch	FTCAOFFSET	FTCA Interrupt Channel Offset Register	Section 20.3.1.43
150h	LFSAOFFSET	LFSA Interrupt Channel Offset Register	Section 20.3.1.44
154h	HBCAOFFSET	HBCA Interrupt Channel Offset Register	Section 20.3.1.45
158h	BTCAOFFSET	BTCA Interrupt Channel Offset Register	Section 20.3.1.46
160h	FTCBOFFSET	FTCB Interrupt Channel Offset Register	Section 20.3.1.47
164h	LFSBOFFSET	LFSB Interrupt Channel Offset Register	Section 20.3.1.48
168h	HBCBOFFSET	HBCB Interrupt Channel Offset Register	Section 20.3.1.49
16Ch	BTCBOFFSET	BTCB Interrupt Channel Offset Register	Section 20.3.1.50
178h	PTCRL	Port Control Register	Section 20.3.1.51
17Ch	RTCTRL	RAM Test Control Register	Section 20.3.1.52
180h	DCTRL	Debug Control Register	Section 20.3.1.53
184h	WPR	Watch Point Register	Section 20.3.1.54
188h	WMR	Watch Mask Register	Section 20.3.1.55
18Ch	FAACSADDR	FIFO A Active Channel Source Address Register	
190h	FAACDADDR	FIFO A Active Channel Destination Address Register	
194h	FAACTC	FIFO A Active Channel Transfer Address Register	
198h	FBACSADDR	FIFO B Active Channel Source Address Register	Section 20.3.1.56
19Ch	FBACDADDR	FIFO B Active Channel Destination Address Register	Section 20.3.1.57
1A0h	FBACTC	FIFO B Active Channel Transfer Address Register	Section 20.3.1.58
1A8h	DMAPECR	Parity Control Register	Section 20.3.1.62
1ACh	DMAPAR	DMA Parity Error Address Register	Section 20.3.1.63
1B0h	DMAMPCTRL1	DMA Memory Protection Control Register 1	Section 20.3.1.64
1B4h	DMAMPST1	DMA Memory Protection Status Register 1	Section 20.3.1.65
1B8h	DMAMPR0S	DMA Memory Protection Region 0 Start Address Register	Section 20.3.1.66
1BCh	DMAMPR0E	DMA Memory Protection Region 0 End Address Register	Section 20.3.1.67
1C0h	DMAMPR1S	DMA Memory Protection Region 1 Start Address Register	Section 20.3.1.68
1C4h	DMAMPR1E	DMA Memory Protection Region 1 End Address Register	Section 20.3.1.69
1C8h	DMAMPR2S	DMA Memory Protection Region 2 Start Address Register	Section 20.3.1.70
1CCh	DMAMPR2E	DMA Memory Protection Region 2 End Address Register	Section 20.3.1.71
1D0h	DMAMPR3S	DMA Memory Protection Region 3 Start Address Register	Section 20.3.1.72
1D4h	DMAMPR3E	DMA Memory Protection Region 3 End Address Register	Section 20.3.1.73
1D8h	DMAMPCTRL	DMA Memory Protection Control Register	Section 20.3.1.74
1DCh	DMAMPST2	DMA Memory Protection Status Register 2	Section 20.3.1.75
1E0h	DMAMPR4S	DMA Memory Protection Region 4 Start Address Register	Section 20.3.1.76
1E4h	DMAMPR4E	DMA Memory Protection Region 4 End Address Register	Section 20.3.1.77
1E8h	DMAMPR5S	DMA Memory Protection Region 5 Start Address Register	Section 20.3.1.78
1ECh	DMAMPR5E	DMA Memory Protection Region 5End Address Register	Section 20.3.1.79
1F0h	DMAMPR6S	DMA Memory Protection Region 6 Start Address Register	Section 20.3.1.80
1F4h	DMAMPR6E	DMA Memory Protection Region 6 End Address Register	Section 20.3.1.81
1F8h	DMAMPR7S	DMA Memory Protection Region 7 Start Address Register	Section 20.3.1.82
1FCh	DMAMPR7E	DMA Memory Protection Region 7 End Address Register	Section 20.3.1.83
228h	DMASECCCTRL	DMA Single-bit ECC Control Register	Section 20.3.1.84

Table 20-7. DMA Control Registers (continued)

Offset	Acronym	Register Description	Section
230h	DMAECCSBE	DMA ECC Single-bit Error Address Register	Section 20.3.1.85
240h	FIFOASTATREG	FIFO A Status Register	Section 20.3.1.86
244h	FIFOBSTATREG	FIFO B Status Register	Section 20.3.1.87
330h	DMAREQPS1	DMA Request Polarity Select Register 1	Section 20.3.1.88
334h	DMAREQPS0	DMA Request Polarity Select Register 0	Section 20.3.1.89
340h	TERECTRL	TER Event Control Register	Section 20.3.1.90
344h	TERFLAG	TER Event Flag Register	Section 20.3.1.91
348h	TERROFFSET	TER Event Channel Offset Register	Section 20.3.1.92

Table 20-8. Control Packet Memory Map

Offset	Acronym	Register Description	Section
		Primary Control Packet 0	
00h	ISADDR	Initial Source Address Register	Section 20.3.2.1
04h	IDADDR	Initial Destination Address Register	Section 20.3.2.2
08h	ITCOUNT	Initial Transfer Count Register	Section 20.3.2.3
10h	CHCTRL	Channel Control Register	Section 20.3.2.4
14h	EIOFF	Element Index Offset Register	Section 20.3.2.5
18h	FIOFF	Frame Index Offset Register	Section 20.3.2.6
		Working Control Packet 0	
800h	CSADDR	Current Source Address Register	Section 20.3.2.7
804h	CDADDR	Current Destination Address Register	Section 20.3.2.8
808h	CTCOUNT	Current Transfer Count Register	Section 20.3.2.9

20.3.1 Global Configuration Registers

These registers control the overall behavior of the DMA controller.

20.3.1.1 Global Control Register (GCTRL)

Figure 20-19. Global Control Register (GCTRL) [offset = 00]

31-17	16
Reserved	DMA_EN
R-0	R/WP-0

15	14	13-10	9-8
Reserved	BUS_BUSY	Reserved	DEBUGMODE
R-0	R-0	R-0	R/WP-0

7-1	0
Reserved	DMARES
R-0	R/WP-0

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privilege mode only; -*n* = value after reset

Table 20-9. Global Control Register (GCTRL) Field Descriptions

Bit	Field	Value	Description
31-17	Reserved	0	Reads return 0. Writes have no effect.
16	DMA_EN		DMA enable bit. The configuration registers and channel control packets should be setup first before DMA_EN bit is set to one to prevent state machines from carrying out bus transactions. If DMA_EN bit is cleared in the middle of an bus transaction, the state machine will stop at an arbitration boundary.
		0	The DMA is disabled.
		1	The DMA is enabled.
15	Reserved	0	Reads return 0. Writes have no effect.
14	BUS_BUSY		This bit indicates status of DMA external AHB bus status.
		0	DMAs external bus is not busy in data transfers.
		1	DMAs external bus is busy in data transfers.
13-10	Reserved	0	Reads return 0. Writes have no effect.
9-8	DEBUGMODE		Debug Mode.
		0	Ignore suspend.
		1h	Finish current block transfer.
		2h	Finish current frame transfer.
		3h	Immediately stop at an DMA arbitration boundary and continue after suspend.
7-1	Reserved	0	Reads return 0. Writes have no effect.
0	DMARES		DMA software reset. **Note:** In the event a DMA slave does not respond, the DMA module will respond to the software reset upon reaching an arbitration boundary.
		0	Read: Software reset is disabled. Write: No effect.
		1	Read and write: The DMA state machine and all control registers are in software reset. Control packets are not reset when DMA software reset is active.

20.3.1.2 Channel Pending Register (PEND)

Figure 20-20. Channel Pending Register (PEND) [offset = 04h]

31	0
PEND[31:0]	
R-0	

LEGEND: R = Read only; -*n* = value after reset

Table 20-10. Channel Pending Register (PEND) Field Descriptions

Bit	Field	Value	Description
31-0	PEND[*n*]		Channel pending register. Bit 0 corresponds to channel 0, bit 1 corresponds to channel 1, and so on. Reading from PEND gives the channel pending information no matter if the channel was initiated by SW or HW. Once set, it remains set even if the corresponding channel is disabled via HWCHENA or SWCHENA. The pending bit is automatically cleared for the following conditions: • At the end of a frame or a block transfer depending on how the channel is triggered as programmed in the TTYPE bit field of CHCTRL. • The control packet is modified after the pending bit is set. • A bus error occurs. • A transaction parity error occurs
		0	The corresponding channel is inactive.
		1	The corresponding channel is pending and is waiting for service.

20.3.1.3 DMA Status Register (DMASTAT)

Figure 20-21. DMA Status Register (DMASTAT) [offset = 0Ch]

31	0
STCH[31:0]	
R-0	

LEGEND: R = Read only; -*n* = value after reset

Table 20-11. DMA Status Register (DMASTAT) Field Descriptions

Bit	Field	Value	Description
31-0	STCH[*n*]		Status of DMA channels. Bit 0 corresponds to channel 0, bit 1 corresponds to channel 1, and so on.
		0	The channel is not being currently processed.
		1	The channel is currently being processed using one of the FIFOs.
			Note: The status of a channel currently being processed remains active even if emulation mode is entered or DMA is disabled via DMA_EN bit. Since there are two FIFOs, up to 2 bits can be set in this register at any given time.

20.3.1.4 DMA Revision ID Register (DMAREVID)

Figure 20-22. DMA Revision ID Register (DMAREVID) [offset = 10h]

<table>
<tr><td>31</td><td>30</td><td>29</td><td>28</td><td>27</td><td>16</td></tr>
<tr><td colspan="2">SCHEME</td><td colspan="2">Reserved</td><td colspan="2">FUNC</td></tr>
<tr><td colspan="2">R-1</td><td colspan="2">R-0</td><td colspan="2">R-A0Dh</td></tr>
</table>

<table>
<tr><td>15</td><td>11</td><td>10</td><td>8</td><td>7</td><td>6</td><td>5</td><td>0</td></tr>
<tr><td colspan="2">Reserved</td><td colspan="2">MAJOR</td><td colspan="2">Reserved</td><td colspan="2">MINOR</td></tr>
<tr><td colspan="2">R-0</td><td colspan="2">R-0</td><td colspan="2">R-0</td><td colspan="2">R-3h</td></tr>
</table>

LEGEND: R = Read only; -*n* = value after reset

Table 20-12. DMA Revision ID Register Description

Bit	Field	Value	Description
31-30	SCHEME	1	Identification Scheme of REVID.
29-28	Reserved	0	Reads return 0. Writes have no effect.
27-16	FUNC	A0Dh	Indicates module family.
15-11	Reserved	0	Reserved
10-8	MAJOR	0	Major revision number.
7-6	Reserved	0	Reserved
5-0	MINOR	3h	Minor revision number.

20.3.1.5 HW Channel Enable Set and Status Register (HWCHENAS)

Figure 20-23. HW Channel Enable Set and Status Register (HWCHENAS) [offset = 14h]

31	0
HWCHENA[31:0]	
R/WP-0	

LEGEND: R/W = Read/Write; WP = Write in privilege mode only; -*n* = value after reset

Table 20-13. HW Channel Enable Set and Status Register (HWCHENAS) Field Descriptions

Bit	Field	Value	Description
31-0	HWCHENA[*n*]		Hardware channel enable bit. Bit 0 corresponds to channel 0, bit 1 corresponds to channel 1, and so on. An active hardware DMA request cannot initiate a DMA transfer unless the corresponding hardware enable bit is set. The corresponding hardware enable bit is cleared automatically for the following conditions: • At the end of a block transfer if the auto-initiation bit AIM (see CHCTRL) is not active. • If a bus error is detected for an active channel. Reading from HWCHENAS gives the status (enabled/disabled) of all channels.
		0	The corresponding channel is disabled for hardware triggering.
		1	The corresponding channel is enabled for hardware triggering.

20.3.1.6 HW Channel Enable Reset and Status Register (HWCHENAR)

Figure 20-24. HW Channel Enable Reset and Status Register (HWCHENAR) [offset = 1Ch]

31	0
HWCHDIS[31:0]	
R/WP-0	

LEGEND: R/W = Read/Write; WP = Write in privilege mode only; -*n* = value after reset

Table 20-14. HW Channel Enable Reset and Status Register (HWCHENAR) Field Descriptions

Bit	Field	Value	Description
31-0	HWCHDIS[*n*]		HW channel disable bit. Bit 0 corresponds to channel 0, bit 1 corresponds to channel 1, and so on.
		0	Read: The corresponding channel is disabled for HW triggering. Write: No effect.
		1	Read: The corresponding channel is enabled for HW triggering. Write: The corresponding channel is disabled.

20.3.1.7 SW Channel Enable Set and Status Register (SWCHENAS)

Figure 20-25. SW Channel Enable Set and Status Register (SWCHENAS) [offset = 24h]

31 … 0
SWCHENA[31:0]
R/WP-0

LEGEND: R/W = Read/Write; WP = Write in privilege mode only; -*n* = value after reset

Table 20-15. SW Channel Enable Set and Status Register (SWCHENAS) Field Descriptions

Bit	Field	Value	Description
31-0	SWCHENA[*n*]		SW channel enable bit. Bit 0 corresponds to channel 0, bit 1 corresponds to channel 1, and so on. Writing a 1 to a bit triggers a SW request on the corresponding channel to start a DMA transaction. The corresponding bit is automatically cleared by the following conditions. • The corresponding bit is cleared after one frame transfer if the TTYPE bit in Channel Control Register (CHCTRL) is programmed for frame transfer. • The corresponding bit is cleared after one block transfer if the corresponding TTYPE bit is programmed for block transfer and the auto-initiation bit is not enabled. • The control packet is modified after the pending bit is set. • The corresponding bit is cleared after one block transfer when TTYPE bit is programmed for blocks transfer and if the corresponding bit in HW channel enable register (HWCHENAS) is enabled. When a channel is enabled for both HW and SW, the state machine will initiate transfers based on the SW first. After one block transfer is complete, the corresponding bit in the SWCHENA register is then cleared. The same channel is serviced again by a HW DMA request. • The corresponding bit is cleared if a bus error is detected. • A transaction parity error occurs. Reading from SWCHENAS gives the status (enabled/disabled) of channels 0 through 31.
		0	The corresponding channel is not triggered by SW request.
		1	The corresponding channel is triggered by SW request.

20.3.1.8 SW Channel Enable Reset and Status Register (SWCHENAR)

Figure 20-26. SW Channel Enable Reset and Status Register (SWCHENAR) [offset = 2Ch]

31 … 0
SWCHDIS[31:0]
R/WP-0

LEGEND: R/W = Read/Write; WP = Write in privilege mode only; -*n* = value after reset

Table 20-16. SW Channel Enable Reset and Status Register (SWCHENAR) Field Descriptions

Bit	Field	Value	Description
31-0	SWCHDIS[*n*]		SW channel disable bit. Bit 0 corresponds to channel 0, bit 1 corresponds to channel 1, and so on.
		0	Read: The corresponding channel was not triggered by SW. Write: No effect.
		1	Read: The corresponding channel was triggered by SW. Write: The corresponding channel is disabled.

20.3.1.9 Channel Priority Set Register (CHPRIOS)

Figure 20-27. Channel Priority Set Register (CHPRIOS) [offset = 34h]

31		0
	CPS[31:0]	
	R/WP-0	

LEGEND: R/W = Read/Write; WP = Write in privilege mode only; -*n* = value after reset

Table 20-17. Channel Priority Set Register (CHPRIOS) Field Descriptions

Bit	Field	Value	Description
31-0	CPS[*n*]	0	Channel priority set bit. Bit 0 corresponds to channel 0, bit 1 corresponds to channel 1, and so on. Writing a 1 to a bit assigns the corresponding channel to the high priority queue. Read: The corresponding channel is assigned to the low priority queue. Write: No effect.
		1	Read and write: The corresponding channel is assigned to high priority queue.

20.3.1.10 Channel Priority Reset Register (CHPRIOR)

Figure 20-28. Channel Priority Reset Register (CHPRIOR) [offset = 3Ch]

31		0
	CPR[31:0]	
	R/WP-0	

LEGEND: R/W = Read/Write; WP = Write in privilege mode only; -*n* = value after reset

Table 20-18. Channel Priority Reset Register (CHPRIOR) Field Descriptions

Bit	Field	Value	Description
31-0	CPR[*n*]	0	Channel priority reset bit. Bit 0 corresponds to channel 0, bit 1 corresponds to channel 1, and so on. Writing a 1 to a bit assigns the according channel to the low priority queue. Read: The corresponding channel is assigned to the low priority queue. Write: No effect.
		1	Read: The corresponding channel is assigned to the high priority queue. Write: The corresponding channel is assigned to the low priority queue.

20.3.1.11 Global Channel Interrupt Enable Set Register (GCHIENAS)

Figure 20-29. Global Channel Interrupt Enable Set Register (GCHIENAS) [offset = 44h]

31	0
GCHIE[31:0]	
R/WP-0	

LEGEND: R/W = Read/Write; WP = Write in privilege mode only; -*n* = value after reset

Table 20-19. Global Channel Interrupt Enable Set Register (GCHIENAS) Field Descriptions

Bit	Field	Value	Description
31-0	GCHIE[*n*]		Global channel interrupt enable bit. Bit 0 corresponds to channel 0, bit 1 corresponds to channel 1, and so on.
		0	Read: The corresponding channel is disabled for interrupt. Write: No effect.
		1	Read and write: The corresponding channel is enabled for interrupt.

20.3.1.12 Global Channel Interrupt Enable Reset Register (GCHIENAR)

Figure 20-30. Global Channel Interrupt Enable Reset Register (GCHIENAR) [offset = 4Ch]

31	0
GCHID[31:0]	
R/WP-0	

LEGEND: R/W = Read/Write; WP = Write in privilege mode only; -*n* = value after reset

Table 20-20. Global Channel Interrupt Enable Reset Register (GCHIENAR) Field Descriptions

Bit	Field	Value	Description
31-0	GCHID[*n*]		Global channel interrupt disable bit. Bit 0 corresponds to channel 0, bit 1 corresponds to channel 1, and so on.
		0	Read: The corresponding channel is disabled for interrupt. Write: No effect.
		1	Read: The corresponding channel is enabled for interrupt. Write: The corresponding channel is disabled for interrupt.

20.3.1.13 DMA Request Assignment Register 0 (DREQASI0)

Figure 20-31. DMA Request Assignment Register 0 (DREQASI0) [offset = 54h]

31 30	29 24	23 22	21 16
Reserved	CH0ASI	Reserved	CH1ASI
R-0	R/WP-0	R-0	R/WP-1h

15 14	13 8	7 6	5 0
Reserved	CH2ASI	Reserved	CH3ASI
R-0	R/WP-2h	R-0	R/WP-3h

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privilege mode only; -*n* = value after reset

Table 20-21. DMA Request Assignment Register 0 (DREQASI0) Field Descriptions

Bit	Field	Value	Description
31-30	Reserved	0	Reads return 0. Writes have no effect.
29-24	CH0ASI		Channel 0 assignment. This bit field chooses the DMA request assignment for channel 0.
		0	DMA request line 0 triggers channel 0.
		:	:
		2Fh	DMA request line 47 triggers channel 0.
		30h-3Fh	Reserved
23-22	Reserved	0	Reads return 0. Writes have no effect.
21-16	CH1ASI		Channel 1 assignment. This bit field chooses the DMA request assignment for channel 1.
		0	DMA request line 0 triggers channel 1.
		:	:
		2Fh	DMA request line 47 triggers channel 1.
		30h-3Fh	Reserved
15-14	Reserved	0	Reads return 0. Writes have no effect.
13-8	CH2ASI		Channel 2 assignment. This bit field chooses the DMA request assignment for channel 2.
		0	DMA request line 0 triggers channel 2.
		:	:
		2Fh	DMA request line 47 triggers channel 2.
		30h-3Fh	Reserved
7-6	Reserved	0	Reads return 0. Writes have no effect.
5-0	CH3ASI		Channel 3 assignment. This bit field chooses the DMA request assignment for channel 3.
		0	DMA request line 0 triggers channel 3.
		:	:
		2Fh	DMA request line 47 triggers channel 3.
		30h-3Fh	Reserved

20.3.1.14 DMA Request Assignment Register 1 (DREQASI1)

Figure 20-32. DMA Request Assignment Register 1 (DREQASI1) [offset = 58h]

31 30	29 24	23 22	21 16
Reserved	CH4ASI	Reserved	CH5ASI
R-0	R/WP-4h	R-0	R/WP-5h

15 14	13 8	7 6	5 0
Reserved	CH6ASI	Reserved	CH7ASI
R-0	R/WP-6h	R-0	R/WP-7h

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privilege mode only; -*n* = value after reset

Table 20-22. DMA Request Assignment Register 1 (DREQASI1) Field Descriptions

Bit	Field	Value	Description
31-30	Reserved	0	Reads return 0. Writes have no effect.
29-24	CH4ASI		Channel 4 assignment. This bit field chooses the DMA request assignment for channel 4.
		0	DMA request line 0 triggers channel 4.
		:	:
		2Fh	DMA request line 47 triggers channel 4.
		30h-3Fh	Reserved
23-22	Reserved	0	Reads return 0. Writes have no effect.
21-26	CH5ASI		Channel 5 assignment. This bit field chooses the DMA request assignment for channel 5.
		0	DMA request line 0 triggers channel 5.
		:	:
		2Fh	DMA request line 47 triggers channel 5.
		30h-3Fh	Reserved
15-14	Reserved	0	Reads return 0. Writes have no effect.
13-8	CH6ASI		Channel 6 assignment. This bit field chooses the DMA request assignment for channel 6.
		0	DMA request line 0 triggers channel 6.
		:	:
		2Fh	DMA request line 47 triggers channel 6.
		30h-3Fh	Reserved
7-6	Reserved	0	Reads return 0. Writes have no effect.
5-0	CH7ASI		Channel 7 assignment. This bit field chooses the DMA request assignment for channel 7.
		0	DMA request line 0 triggers channel 7.
		:	:
		2Fh	DMA request line 47 triggers channel 7.
		30h-3Fh	Reserved

20.3.1.15 DMA Request Assignment Register 2 (DREQASI2)

Figure 20-33. DMA Request Assignment Register 2 (DREQASI2) [offset = 5Ch]

31 30	29 24	23 22	21 16
Reserved	CH8ASI	Reserved	CH9ASI
R-0	R/WP-8h	R-0	R/WP-9h

15 14	13 8	7 6	5 0
Reserved	CH10ASI	Reserved	CH11ASI
R-0	R/WP-Ah	R-0	R/WP-Bh

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privilege mode only; -*n* = value after reset

Table 20-23. DMA Request Assignment Register 2 (DREQASI2) Field Descriptions

Bit	Field	Value	Description
31-30	Reserved	0	Reads return 0. Writes have no effect.
29-24	CH8ASI		Channel 8 assignment. This bit field chooses the DMA request assignment for channel 8.
		0	DMA request line 0 triggers channel 8.
		:	:
		2Fh	DMA request line 47 triggers channel 8.
		30h-3Fh	Reserved
23-22	Reserved	0	Reads return 0. Writes have no effect.
21-16	CH9ASI		Channel 9 assignment. This bit field chooses the DMA request assignment for channel 9.
		0	DMA request line 0 triggers channel 9.
		:	:
		2Fh	DMA request line 47 triggers channel 9.
		30h-3Fh	Reserved
15-14	Reserved	0	Reads return 0. Writes have no effect.
13-8	CH10ASI		Channel 10 assignment. This bit field chooses the DMA request assignment for channel 10.
		0	DMA request line 0 triggers channel 10.
		:	:
		2Fh	DMA request line 47 triggers channel 10.
		30h-3Fh	Reserved
7-6	Reserved	0	Reads return 0. Writes have no effect.
5-0	CH11ASI		Channel 11 assignment. This bit field chooses the DMA request assignment for channel 11.
		0	DMA request line 0 triggers channel 11.
		:	:
		2Fh	DMA request line 47 triggers channel 11.
		30h-3Fh	Reserved

20.3.1.16 DMA Request Assignment Register 3 (DREQASI3)

Figure 20-34. DMA Request Assignment Register 3 (DREQASI3) [offset = 60h]

<table>
<tr><td>31</td><td>30</td><td>29</td><td>24</td><td>23</td><td>22</td><td>21</td><td>16</td></tr>
<tr><td colspan="2">Reserved</td><td colspan="2">CH12ASI</td><td colspan="2">Reserved</td><td colspan="2">CH13ASI</td></tr>
<tr><td colspan="2">R-0</td><td colspan="2">R/WP-Ch</td><td colspan="2">R-0</td><td colspan="2">R/WP-Dh</td></tr>
<tr><td>15</td><td>14</td><td>13</td><td>8</td><td>7</td><td>6</td><td>5</td><td>0</td></tr>
<tr><td colspan="2">Reserved</td><td colspan="2">CH14ASI</td><td colspan="2">Reserved</td><td colspan="2">CH15ASI</td></tr>
<tr><td colspan="2">R-0</td><td colspan="2">R/WP-Eh</td><td colspan="2">R-0</td><td colspan="2">R/WP-Fh</td></tr>
</table>

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privilege mode only; -*n* = value after reset

Table 20-24. DMA Request Assignment Register 3 (DREQASI3) Field Descriptions

Bit	Field	Value	Description
31-30	Reserved	0	Reads return 0. Writes have no effect.
29-24	CH12ASI		Channel 12 assignment. This bit field chooses the DMA request assignment for channel 12.
		0	DMA request line 0 triggers channel 12.
		:	:
		2Fh	DMA request line 47 triggers channel 12.
		30h-3Fh	Reserved
23-22	Reserved	0	Reads return 0. Writes have no effect.
21-16	CH13ASI		Channel 13 assignment. This bit field chooses the DMA request assignment for channel 13.
		0	DMA request line 0 triggers channel 13.
		:	:
		2Fh	DMA request line 47 triggers channel 13.
		30h-3Fh	Reserved
15-14	Reserved	0	Reads return 0. Writes have no effect.
13-8	CH14ASI		Channel 14 assignment. This bit field chooses the DMA request assignment for channel 14.
		0	DMA request line 0 triggers channel 14.
		:	:
		2Fh	DMA request line 47 triggers channel 14.
		30h-3Fh	Reserved
7-6	Reserved	0	Reads return 0. Writes have no effect.
5-0	CH15ASI		Channel 15 assignment. This bit field chooses the DMA request assignment for channel 15.
		0	DMA request line 0 triggers channel 15.
		:	:
		2Fh	DMA request line 47 triggers channel 15.
		30h-3Fh	Reserved

20.3.1.17 DMA Request Assignment Register 4 (DREQASI4)

Figure 20-35. DMA Request Assignment Register 4 (DREQASI4) [offset = 64h]

31 30	29 24	23 22	21 16
Reserved	CH16ASI	Reserved	CH17ASI
R-0	R/WP-10h	R-0	R/WP-11h

15 14	13 8	7 6	5 0
Reserved	CH18ASI	Reserved	CH19ASI
R-0	R/WP-12h	R-0	R/WP-13h

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privilege mode only; -*n* = value after reset

Table 20-25. DMA Request Assignment Register 4 (DREQASI4) Field Descriptions

Bit	Field	Value	Description
31-30	Reserved	0	Reads return 0. Writes have no effect.
29-24	CH16ASI		Channel 16 assignment. This bit field chooses the DMA request assignment for channel 16.
		0 : 2Fh 30h-3Fh	DMA request line 0 triggers channel 16. : DMA request line 47 triggers channel 16. Reserved
23-22	Reserved	0	Reads return 0. Writes have no effect.
21-16	CH17ASI		Channel 17 assignment. This bit field chooses the DMA request assignment for channel 17.
		0 : 2Fh 30h-3Fh	DMA request line 0 triggers channel 17. : DMA request line 47 triggers channel 17. Reserved
15-14	Reserved	0	Reads return 0. Writes have no effect.
13-8	CH18ASI		Channel 18 assignment. This bit field chooses the DMA request assignment for channel 18.
		0 : 2Fh 30h-3Fh	DMA request line 0 triggers channel 18. : DMA request line 47 triggers channel 18. Reserved
7-6	Reserved	0	Reads return 0. Writes have no effect.
5-0	CH19ASI		Channel 19 assignment. This bit field chooses the DMA request assignment for channel 19.
		0 : 2Fh 30h-3Fh	DMA request line 0 triggers channel 19. : DMA request line 47 triggers channel 19. Reserved

20.3.1.18 DMA Request Assignment Register 5 (DREQASI5)

Figure 20-36. DMA Request Assignment Register 5 (DREQASI5) [offset = 68h]

31 30	29 24	23 22	21 16
Reserved	CH20ASI	Reserved	CH21ASI
R-0	R/WP-14h	R-0	R/WP-15h

15 14	13 8	7 6	5 0
Reserved	CH22ASI	Reserved	CH23ASI
R-0	R/WP-16h	R-0	R/WP-17h

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privilege mode only; -*n* = value after reset

Table 20-26. DMA Request Assignment Register 5 (DREQASI5) Field Descriptions

Bit	Field	Value	Description
31-30	Reserved	0	Reads return 0. Writes have no effect.
29-24	CH20ASI		Channel 20 assignment. This bit field chooses the DMA request assignment for channel 20.
		0	DMA request line 0 triggers channel 20.
		:	:
		2Fh	DMA request line 47 triggers channel 20.
		30h-3Fh	Reserved
23-22	Reserved	0	Reads return 0. Writes have no effect.
21-26	CH21ASI		Channel 21 assignment. This bit field chooses the DMA request assignment for channel 21.
		0	DMA request line 0 triggers channel 21.
		:	:
		2Fh	DMA request line 47 triggers channel 21.
		30h-3Fh	Reserved
15-14	Reserved	0	Reads return 0. Writes have no effect.
13-8	CH22ASI		Channel 22 assignment. This bit field chooses the DMA request assignment for channel 22.
		0	DMA request line 0 triggers channel 22.
		:	:
		2Fh	DMA request line 47 triggers channel 22.
		30h-3Fh	Reserved
7-6	Reserved	0	Reads return 0. Writes have no effect.
5-0	CH23ASI		Channel 23 assignment. This bit field chooses the DMA request assignment for channel 23.
		0	DMA request line 0 triggers channel 23.
		:	:
		2Fh	DMA request line 47 triggers channel 23.
		30h-3Fh	Reserved

20.3.1.19 DMA Request Assignment Register 6 (DREQASI6)

Figure 20-37. DMA Request Assignment Register 6 (DREQASI6) [offset = 6Ch]

31 30	29 24	23 22	21 16
Reserved	CH24ASI	Reserved	CH25ASI
R-0	R/WP-18h	R-0	R/WP-19h

15 14	13 8	7 6	5 0
Reserved	CH26ASI	Reserved	CH27ASI
R-0	R/WP-1Ah	R-0	R/WP-1Bh

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privilege mode only; -*n* = value after reset

Table 20-27. DMA Request Assignment Register 6 (DREQASI6) Field Descriptions

Bit	Field	Value	Description
31-30	Reserved	0	Reads return 0. Writes have no effect.
29-24	CH24ASI		Channel 24 assignment. This bit field chooses the DMA request assignment for channel 24.
		0	DMA request line 0 triggers channel 24.
		:	:
		2Fh	DMA request line 47 triggers channel 24.
		30h-3Fh	Reserved
23-22	Reserved	0	Reads return 0. Writes have no effect.
21-16	CH25ASI		Channel 25 assignment. This bit field chooses the DMA request assignment for channel 25.
		0	DMA request line 0 triggers channel 25.
		:	:
		2Fh	DMA request line 47 triggers channel 25.
		30h-3Fh	Reserved
15-14	Reserved	0	Reads return 0. Writes have no effect.
13-8	CH26ASI		Channel 26 assignment. This bit field chooses the DMA request assignment for channel 26.
		0	DMA request line 0 triggers channel 26.
		:	:
		2Fh	DMA request line 47 triggers channel 26.
		30h-3Fh	Reserved
7-6	Reserved	0	Reads return 0. Writes have no effect.
5-0	CH27ASI		Channel 27 assignment. This bit field chooses the DMA request assignment for channel 27.
		0	DMA request line 0 triggers channel 27.
		:	:
		2Fh	DMA request line 47 triggers channel 27.
		30h-3Fh	Reserved

20.3.1.20 DMA Request Assignment Register 7 (DREQASI7)

Figure 20-38. DMA Request Assignment Register 7 (DREQASI7) [offset = 70h]

31 30	29 24	23 22	21 16
Reserved	CH28ASI	Reserved	CH29ASI
R-0	R/WP-1Ch	R-0	R/WP-1Dh

15 14	13 8	7 6	5 0
Reserved	CH30ASI	Reserved	CH31ASI
R-0	R/WP-1Eh	R-0	R/WP-1Fh

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privilege mode only; -*n* = value after reset

Table 20-28. DMA Request Assignment Register 7 (DREQASI7) Field Descriptions

Bit	Field	Value	Description
31-30	Reserved	0	Reads return 0. Writes have no effect.
29-24	CH28ASI		Channel 28 assignment. This bit field chooses the DMA request assignment for channel 28.
		0	DMA request line 0 triggers channel 28.
		:	:
		2Fh	DMA request line 47 triggers channel 28.
		30h-3Fh	Reserved
23-22	Reserved	0	Reads return 0. Writes have no effect.
21-16	CH29ASI		Channel 29 assignment. This bit field chooses the DMA request assignment for channel 29.
		0	DMA request line 0 triggers channel 29.
		:	:
		2Fh	DMA request line 47 triggers channel 29.
		30h-3Fh	Reserved
15-14	Reserved	0	Reads return 0. Writes have no effect.
13-8	CH30ASI		Channel 30 assignment. This bit field chooses the DMA request assignment for channel 30.
		0	DMA request line 0 triggers channel 30.
		:	:
		2Fh	DMA request line 47 triggers channel 30.
		30h-3Fh	Reserved
7-6	Reserved	0	Reads return 0. Writes have no effect.
5-0	CH31ASI		Channel 31 assignment. This bit field chooses the DMA request assignment for channel 31.
		0	DMA request line 0 triggers channel 31.
		:	:
		2Fh	DMA request line 47 triggers channel 31.
		30h-3Fh	Reserved

20.3.1.21 Port Assignment Register 0 (PAR0)

Figure 20-39. Port Assignment Register 0 (PAR0) [offset = 94h]

31	30 28	27	26 24	23	22 20	19	18 16
Rsvd	CH0PA	Rsvd	CH1PA	Rsvd	CH2PA	Rsvd	CH3PA
R-0	R/WP-0	R-0	R/WP-0	R-0	R/WP-0	R-0	R/WP-0

15	14 12	11	10 8	7	6 4	3	2 0
Rsvd	CH4PA	Rsvd	CH5PA	Rsvd	CH6PA	Rsvd	CH7PA
R-0	R/WP-0	R-0	R/WP-0	R-0	R/WP-0	R-0	R/WP-0

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privilege mode only; -*n* = value after reset

Table 20-29. Port Assignment Register 0 (PAR0) Field Descriptions

Bit	Field	Value	Description
31	Reserved	0	Reads return 0. Writes have no effect.
30-28	CH0PA		These bit fields determine to which port(s) channel 0 is assigned.
		1h	Port A and B combined, A read/B write
		2h	Port A only
		3h	Port B only
		Others	Port A and B combined, B read/A write
27	Reserved	0	Reads return 0. Writes have no effect.
26-24	CH1PA	0-7h	These bit fields determine to which port channel 1 is assigned. Refer to CH0PA for bit value descriptions.
23	Reserved	0	Reads return 0. Writes have no effect.
22-20	CH2PA	0-7h	These bit fields determine to which port channel 2 is assigned. Refer to CH0PA for bit value descriptions.
19	Reserved	0	Reads return 0. Writes have no effect.
18-16	CH3PA	0-7h	These bit fields determine to which port channel 3 is assigned. Refer to CH0PA for bit value descriptions.
15	Reserved	0	Reads return 0. Writes have no effect.
14-12	CH4PA	0-7h	These bit fields determine to which port channel 4 is assigned. Refer to CH0PA for bit value descriptions.
11	Reserved	0	Reads return 0. Writes have no effect.
10-8	CH5PA	0-7h	These bit fields determine to which port channel 5 is assigned. Refer to CH0PA for bit value descriptions.
7	Reserved	0	Reads return 0. Writes have no effect.
6-4	CH6PA	0-7h	These bit fields determine to which port channel 6 is assigned. Refer to CH0PA for bit value descriptions.
3	Reserved	0	Reads return 0. Writes have no effect.
2-0	CH7PA	0-7h	These bit fields determine to which port channel 7 is assigned. Refer to CH0PA for bit value descriptions.

20.3.1.22 Port Assignment Register 1 (PAR1)

Figure 20-40. Port Assignment Register 1 (PAR1) [offset = 98h]

31	30 28	27	26 24	23	22 20	19	18 16
Rsvd	CH8PA	Rsvd	CH9PA	Rsvd	CH10PA	Rsvd	CH11PA
R-0	R/WP-0	R-0	R/WP-0	R-0	R/WP-0	R-0	R/WP-0

15	14 12	11	10 8	7	6 4	3	2 0
Rsvd	CH12PA	Rsvd	CH13PA	Rsvd	CH14PA	Rsvd	CH15PA
R-0	R/WP-0	R-0	R/WP-0	R-0	R/WP-0	R-0	R/WP-0

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privilege mode only; -*n* = value after reset

Table 20-30. Port Assignment Register 1 (PAR1) Field Descriptions

Bit	Field	Value	Description
31	Reserved	0	Reads return 0. Writes have no effect.
30-28	CH8PA		These bit fields determine to which port channel 8 is assigned.
		1h	Port A and B combined, A read/B write
		2h	Port A only
		3h	Port B only
		Others	Port A and B combined, B read/A write
27	Reserved	0	Reads return 0. Writes have no effect.
26-24	CH9PA	0-7h	These bit fields determine to which port channel 9 is assigned. Refer to CH8PA for bit value descriptions.
23	Reserved	0	Reads return 0. Writes have no effect.
22-20	CH10PA	0-7h	These bit fields determine to which port channel 10 is assigned. Refer to CH8PA for bit value descriptions.
19	Reserved	0	Reads return 0. Writes have no effect.
18-16	CH11PA	0-7h	These bit fields determine to which port channel 11 is assigned. Refer to CH8PA for bit value descriptions.
15	Reserved	0	Reads return 0. Writes have no effect.
14-12	CH12PA	0-7h	These bit fields determine to which port channel 12 is assigned. Refer to CH8PA for bit value descriptions.
11	Reserved	0	Reads return 0. Writes have no effect.
10-8	CH13PA	0-7h	These bit fields determine to which port channel 13 is assigned. Refer to CH8PA for bit value descriptions.
7	Reserved	0	Reads return 0. Writes have no effect.
6-4	CH14PA	0-7h	These bit fields determine to which port channel 14 is assigned. Refer to CH8PA for bit value descriptions.
3	Reserved	0	Reads return 0. Writes have no effect.
2-0	CH15PA	0-7h	These bit fields determine to which port channel 15 is assigned. Refer to CH8PA for bit value descriptions.

20.3.1.23 Port Assignment Register 2 (PAR2)

Figure 20-41. Port Assignment Register 2 (PAR2) [offset = 9Ch]

31	30	28	27	26	24	23	22	20	19	18	16
Rsvd	CH0PA		Rsvd	CH1PA		Rsvd	CH2PA		Rsvd	CH3PA	
R-0	R/WP-0		R-0	R/WP-0		R-0	R/WP-0		R-0	R/WP-0	

15	14	12	11	10	8	7	6	4	3	2	0
Rsvd	CH4PA		Rsvd	CH5PA		Rsvd	CH6PA		Rsvd	CH7PA	
R-0	R/WP-0		R-0	R/WP-0		R-0	R/WP-0		R-0	R/WP-0	

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privilege mode only; -*n* = value after reset

Table 20-31. Port Assignment Register 2 (PAR2) Field Descriptions

Bit	Field	Value	Description
31	Reserved	0	Reads return 0. Writes have no effect.
30-28	CH16PA		These bit fields determine to which port(s) channel 16 is assigned.
		1h	Port A and B combined, A read/B write
		2h	Port A only
		3h	Port B only
		Others	Port A and B combined, B read/A write
27	Reserved	0	Reads return 0. Writes have no effect.
26-24	CH17PA	0-7h	These bit fields determine to which port channel 17 is assigned. Refer to CH16PA for bit value descriptions.
23	Reserved	0	Reads return 0. Writes have no effect.
22-20	CH18PA	0-7h	These bit fields determine to which port channel 18 is assigned. Refer to CH16PA for bit value descriptions.
19	Reserved	0	Reads return 0. Writes have no effect.
18-16	CH19PA	0-7h	These bit fields determine to which port channel 19 is assigned. Refer to CH16PA for bit value descriptions.
15	Reserved	0	Reads return 0. Writes have no effect.
14-12	CH20PA	0-7h	These bit fields determine to which port channel 20 is assigned. Refer to CH16PA for bit value descriptions.
11	Reserved	0	Reads return 0. Writes have no effect.
10-8	CH21PA	0-7h	These bit fields determine to which port channel 21 is assigned. Refer to CH16PA for bit value descriptions.
7	Reserved	0	Reads return 0. Writes have no effect.
6-4	CH22PA	0-7h	These bit fields determine to which port channel 22 is assigned. Refer to CH16PA for bit value descriptions.
3	Reserved	0	Reads return 0. Writes have no effect.
2-0	CH23PA	0-7h	These bit fields determine to which port channel 23 is assigned. Refer to CH16PA for bit value descriptions.

20.3.1.24 Port Assignment Register 3 (PAR3)

Figure 20-42. Port Assignment Register 3 (PAR3) [offset = A0h]

31	30 28	27	26 24	23	22 20	19	18 16
Rsvd	CH24PA	Rsvd	CH25PA	Rsvd	CH26PA	Rsvd	CH27PA
R-0	R/WP-0	R-0	R/WP-0	R-0	R/WP-0	R-0	R/WP-0

15	14 12	11	10 8	7	6 4	3	2 0
Rsvd	CH28PA	Rsvd	CH29PA	Rsvd	CH30PA	Rsvd	CH31PA
R-0	R/WP-0	R-0	R/WP-0	R-0	R/WP-0	R-0	R/WP-0

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privilege mode only; -*n* = value after reset

Table 20-32. Port Assignment Register 3 (PAR3) Field Descriptions

Bit	Field	Value	Description
31	Reserved	0	Reads return 0. Writes have no effect.
30-28	CH24PA		These bit fields determine to which port channel 24 is assigned.
		1h	Port A and B combined, A read/B write
		2h	Port A only
		3h	Port B only
		Others	Port A and B combined, B read/A write
27	Reserved	0	Reads return 0. Writes have no effect.
26-24	CH25PA	0-7h	These bit fields determine to which port channel 25 is assigned. Refer to CH24PA for bit value descriptions.
23	Reserved	0	Reads return 0. Writes have no effect.
22-20	CH26PA	0-7h	These bit fields determine to which port channel 26 is assigned. Refer to CH24PA for bit value descriptions.
19	Reserved	0	Reads return 0. Writes have no effect.
18-16	CH27PA	0-7h	These bit fields determine to which port channel 27 is assigned. Refer to CH24PA for bit value descriptions.
15	Reserved	0	Reads return 0. Writes have no effect.
14-12	CH28PA	0-7h	These bit fields determine to which port channel 28 is assigned. Refer to CH24PA for bit value descriptions.
11	Reserved	0	Reads return 0. Writes have no effect.
10-8	CH29PA	0-7h	These bit fields determine to which port channel 29 is assigned. Refer to CH24PA for bit value descriptions.
7	Reserved	0	Reads return 0. Writes have no effect.
6-4	CH30PA	0-7h	These bit fields determine to which port channel 30 is assigned. Refer to CH24PA for bit value descriptions.
3	Reserved	0	Reads return 0. Writes have no effect.
2-0	CH31PA	0-7h	These bit fields determine to which port channel 31 is assigned. Refer to CH24PA for bit value descriptions.

20.3.1.25 FTC Interrupt Mapping Register (FTCMAP)

Figure 20-43. FTC Interrupt Mapping Register (FTCMAP) [offset = B4h]

31 … 0
FTCAB[31:0]
R/WP-0

LEGEND: R/W = Read/Write; WP = Write in privilege mode only; -*n* = value after reset

Table 20-33. FTC Interrupt Mapping Register (FTCMAP) Field Descriptions

Bit	Field	Value	Description
31-0	FTCAB[*n*]		Frame transfer complete (FTC) interrupt to Group A or Group B. Bit 0 corresponds to channel 0, bit 1 corresponds to channel 1, and so on.
		0	FTC interrupt of the corresponding channel is routed to Group A.
		1	FTC interrupt of the corresponding channel is routed to Group B.

20.3.1.26 LFS Interrupt Mapping Register (LFSMAP)

Figure 20-44. LFS Interrupt Mapping Register (LFSMAP) [offset = BCh]

31 … 0
LFSAB[31:0]
R/WP-0

LEGEND: R/W = Read/Write; WP = Write in privilege mode only; -*n* = value after reset

Table 20-34. LFS Interrupt Mapping Register (LFSMAP) Field Descriptions

Bit	Field	Value	Description
31-0	LFSAB[*n*]		Last frame started (LFS) interrupt to Group A or Group B. Bit 0 corresponds to channel 0, bit 1 corresponds to channel 1, and so on.
		0	LFS interrupt of the corresponding channel is routed to Group A.
		1	LFS interrupt of the corresponding channel is routed to Group B.

20.3.1.27 HBC Interrupt Mapping Register (HBCMAP)

Figure 20-45. HBC Interrupt Mapping Register (HBCMAP) [offset = C4h]

31 … 0
HBCAB[31:0]
R/WP-0

LEGEND: R/W = Read/Write; WP = Write in privilege mode only; -*n* = value after reset

Table 20-35. HBC Interrupt Mapping Register (HBCMAP) Field Descriptions

Bit	Field	Value	Description
31-0	HBCAB[*n*]		Half block complete (HBC) interrupt to Group A or Group B. Bit 0 corresponds to channel 0, bit 1 corresponds to channel 1, and so on.
		0	HBC interrupt of the corresponding channel is routed to Group A.
		1	HBC interrupt of the corresponding channel is routed to Group B.

20.3.1.28 BTC Interrupt Mapping Register (BTCMAP)

Figure 20-46. BTC Interrupt Mapping Register (BTCMAP) [offset = CCh]

31	0
BTCAB[31:0]	
R/WP-0	

LEGEND: R/W = Read/Write; WP = Write in privilege mode only; -*n* = value after reset

Table 20-36. BTC Interrupt Mapping Register (BTCMAP) Field Descriptions

Bit	Field	Value	Description
31-0	BTCAB[*n*]		Block transfer complete (BTC) interrupt to Group A or Group B. Bit 0 corresponds to channel 0, bit 1 corresponds to channel 1, and so on.
		0	BTC interrupt of the corresponding channel is routed to Group A.
		1	BTC interrupt of the corresponding channel is routed to Group B.

20.3.1.29 FTC Interrupt Enable Set Register (FTCINTENAS)

Figure 20-47. FTC Interrupt Enable Set Register (FTCINTENAS) [offset = DCh]

31 … 0
FTCINTENA[31:0]
R/WP-0

LEGEND: R/W = Read/Write; WP = Write in privilege mode only; -*n* = value after reset

Table 20-37. FTC Interrupt Enable Set Register (FTCINTENAS) Field Descriptions

Bit	Field	Value	Description
31-0	FTCINTENA[*n*]		Frame transfer complete (FTC) interrupt enable. Bit 0 corresponds to channel 0, bit 1 corresponds to channel 1, and so on.
		0	Read: Corresponding FTC interrupt of a channel is disabled. Write: No effect.
		1	Read and write: FTC interrupt of the corresponding channel is enabled.

20.3.1.30 FTC Interrupt Enable Reset Register (FTCINTENAR)

Figure 20-48. FTC Interrupt Enable Reset (FTCINTENAR) [offset = E4h]

31 … 0
FTCINTDIS[31:0]
R/WP-0

LEGEND: R/W = Read/Write; WP = Write in privilege mode only; -*n* = value after reset

Table 20-38. FTC Interrupt Enable Reset (FTCINTENAR) Field Descriptions

Bit	Field	Value	Description
31-0	FTCINTDIS[*n*]		Frame transfer complete (FTC) interrupt disable. Bit 0 corresponds to channel 0, bit 1 corresponds to channel 1, and so on.
		0	Read: Corresponding FTC interrupt of a channel is disabled. Write: No effect.
		1	Read: Corresponding FTC interrupt of a channel is enabled. Write: Corresponding FTC interrupt is disabled.

20.3.1.31 LFS Interrupt Enable Set Register (LFSINTENAS)

Figure 20-49. LFS Interrupt Enable Set Register (LFSINTENAS) [offset = ECh]

31 … 0
LFSINTENA[31:0]
R/WP-0

LEGEND: R/W = Read/Write; WP = Write in privilege mode only; -*n* = value after reset

Table 20-39. LFS Interrupt Enable Set Register (LFSINTENAS) Field Descriptions

Bit	Field	Value	Description
31-0	LFSINTENA[*n*]		Last frame started (LFS) interrupt enable. Bit 0 corresponds to channel 0, bit 1 corresponds to channel 1, and so on.
		0	Read: Corresponding LFS interrupt of a channel is disabled. Write: No effect.
		1	Read and write: LFS interrupt of the corresponding channel is disabled.

20.3.1.32 LFS Interrupt Enable Reset Register (LFSINTENAR)

Figure 20-50. LFS Interrupt Enable Reset Register (LFSINTENAR) [offset = F4h]

31 … 0
LFSINTDIS[31:0]
R/WP-0

LEGEND: R/W = Read/Write; WP = Write in privilege mode only; -*n* = value after reset

Table 20-40. LFS Interrupt Enable Reset Register (LFSINTENAR) Field Descriptions

Bit	Field	Value	Description
31-0	LFSINTDIS[*n*]		Last frame started (LFS) interrupt disable. Bit 0 corresponds to channel 0, bit 1 corresponds to channel 1, and so on.
		0	Read: LFS interrupt of the corresponding channel is disabled. Write: No effect.
		1	Read: LFS interrupt of the corresponding channel is enabled. Write: LFS interrupt of the corresponding channel is disabled.

20.3.1.33 HBC Interrupt Enable Set Register (HBCINTENAS)

Figure 20-51. HBC Interrupt Enable Set Register (HBCINTENAS) [offset = FCh]

31   0
HBCINTENA[31:0]
R/WP-0

LEGEND: R/W = Read/Write; WP = Write in privilege mode only; -*n* = value after reset

Table 20-41. HBC Interrupt Enable Set Register (HBCINTENAS) Field Descriptions

Bit	Field	Value	Description
31-0	HBCINTENA[*n*]	0	Half block complete (HBC) interrupt enable. Bit 0 corresponds to channel 0, bit 1 corresponds to channel 1, and so on. Read: HBC interrupt of the corresponding channel is disabled. Write: No effect.
		1	Read and write: HBC interrupt of the corresponding channel is enabled.

20.3.1.34 HBC Interrupt Enable Reset Register (HBCINTENAR)

Figure 20-52. HBC Interrupt Enable Reset Register (HBCINTENAR) [offset = 104h]

31   0
HBCINTDIS[31:0]
R/WP-0

LEGEND: R/W = Read/Write; WP = Write in privilege mode only; -*n* = value after reset

Table 20-42. HBC Interrupt Enable Reset Register (HBCINTENAR) Field Descriptions

Bit	Field	Value	Description
31-0	HBCINTDIS[*n*]	0	Half block complete (HBC) interrupt disable. Bit 0 corresponds to channel 0, bit 1 corresponds to channel 1, and so on. Read: HBC interrupt of the corresponding channel is disabled. Write: No effect.
		1	Read: HBC interrupt of the corresponding channel is enabled. Write: HBC interrupt of the corresponding channel is disabled.

20.3.1.35 BTC Interrupt Enable Set Register (BTCINTENAS)

Figure 20-53. BTC Interrupt Enable Set Register (BTCINTENAS) [offset = 10Ch]

31 ... 0
BTCINTENA[31:0]
R/WP-0

LEGEND: R/W = Read/Write; WP = Write in privilege mode only; -*n* = value after reset

Table 20-43. BTC Interrupt Enable Reset Register (BTCINTENAS) Field Descriptions

Bit	Field	Value	Description
31-0	BTCINTENA[*n*]		Block transfer complete (BTC) interrupt enable. Bit 0 corresponds to channel 0, bit 1 corresponds to channel 1, and so on.
		0	Read: BTC interrupt of the corresponding channel is disabled. Write: No effect.
		1	Read and write: BTC interrupt of the corresponding channel is enabled.

20.3.1.36 BTC Interrupt Enable Reset Register (BTCINTENAR)

Figure 20-54. BTC Interrupt Enable Reset Register (BTCINTENAR) [offset = 114h]

31 ... 0
BTCINTDIS[31:0]
R/WP-0

LEGEND: R/W = Read/Write; WP = Write in privilege mode only; -*n* = value after reset

Table 20-44. BTC Interrupt Enable Reset Register (BTCINTENAR) Field Descriptions

Bit	Field	Value	Description
31-0	BTCINTDIS[*n*]		Block transfer complete (BTC) interrupt disable. Bit 0 corresponds to channel 0, bit 1 corresponds to channel 1, and so on.
		0	Read: BTC interrupt of the corresponding channel is disabled. Write: No effect.
		1	Read: BTC interrupt of the corresponding channel is enabled. Write: BTC interrupt of the corresponding channel is disabled.

20.3.1.37 Global Interrupt Flag Register (GINTFLAG)

Figure 20-55. Global Interrupt Flag Register (GINTFLAG) [offset = 11Ch]

31	0
GINT[31:0]	
R/WP-0	

LEGEND: R/W = Read/Write; WP = Write in privilege mode only; -*n* = value after reset

Table 20-45. Global Interrupt Flag Register (GINTFLAG) Field Descriptions

Bit	Field	Value	Description
31-0	GINT[*n*]		Global interrupt flags. Bit 0 corresponds to channel 0, bit 1 corresponds to channel 1, and so on. A global interrupt flag bit is an OR function of FTC, LFS, HBC, and BTC interrupt flags.
		0	No interrupt is pending on the corresponding channel.
		1	One or more of the interrupt types (FTC, LFS, HBC, or BTC) is pending on the corresponding channel.

20.3.1.38 FTC Interrupt Flag Register (FTCFLAG)

Figure 20-56. FTC Interrupt Flag Register (FTCFLAG) [offset = 124h]

31	0
FTCI[31:0]	
R/W1CP-0	

LEGEND: R/W = Read/Write; W1CP = Write 1 to clear in privilege mode only; -*n* = value after reset

Table 20-46. FTC Interrupt Flag Register (FTCFLAG) Field Descriptions

Bit	Field	Value	Description
31-0	FTCI[*n*]		Frame transfer complete (FTC) flags. Bit 0 corresponds to channel 0, bit 1 corresponds to channel 1, and so on. **Note: Reading from the respective interrupt channel offset register also clears the corresponding flag (see Section 20.3.1.43 and Section 20.3.1.47).** **Note: The state of the flag bit can be polled even if the corresponding interrupt enable bit is cleared.**
		0	Read: FTC interrupt of the corresponding channel is not pending. Write: No effect.
		1	Read: FTC interrupt of the corresponding channel is pending. Write: The flag is cleared.

20.3.1.39 LFS Interrupt Flag Register (LFSFLAG)

Figure 20-57. LFS Interrupt Flag Register (LFSFLAG) [offset = 12Ch]

31	0
LFSI[31:0]	
R/W1CP-0	

LEGEND: R/W = Read/Write;W1CP = Write 1 to clear in privilege mode only; -*n* = value after reset

Table 20-47. LFS Interrupt Flag Register (LFSFLAG) Field Descriptions

Bit	Field	Value	Description
31-0	LFSI[*n*]		Last frame started (LFS) flags. Bit 0 corresponds to channel 0, bit 1 corresponds to channel 1, and so on. **Note: Reading from the respective interrupt channel offset register also clears the corresponding flag (see Section 20.3.1.44 and Section 20.3.1.48).** **Note: The state of the flag bit can be polled even if the corresponding interrupt enable bit is cleared.**
		0	Read: LFS interrupt of the corresponding channel is not pending. Write: No effect.
		1	Read: LFS interrupt of the corresponding channel is pending. Write: The flag is cleared.

20.3.1.40 HBC Interrupt Flag Register (HBCFLAG)

Figure 20-58. HBC Interrupt Flag Register (HBCFLAG) [offset = 134h]

31	0
HBCI[31:0]	
R/W1CP-0	

LEGEND: R/W = Read/Write; W1CP = Write 1 to clear in privilege mode only; -*n* = value after reset

Table 20-48. HBC Interrupt Flag Register (HBCFLAG) Field Descriptions

Bit	Field	Value	Description
31-0	HBCI[*n*]		Half block transfer (HBC) complete flags. Bit 0 corresponds to channel 0, bit 1 corresponds to channel 1, and so on. **Note: Reading from the respective interrupt channel offset register also clears the corresponding flag (see Section 20.3.1.45and Section 20.3.1.49).** **Note: The state of the flag bit can be polled even if the corresponding interrupt enable bit is cleared.**
		0	Read: HBC interrupt of the corresponding channel is not pending. Write: No effect.
		1	Read: HBC interrupt of the corresponding channel is pending. Write: The flag is cleared.

20.3.1.41 BTC Interrupt Flag Register (BTCFLAG)

Figure 20-59. BTC Interrupt Flag Register (BTCFLAG) [offset = 13Ch]

31	0
BTCI[31:0]	
R/W1CP-0	

LEGEND: R/W = Read/Write; W1CP = Write 1 to clear in privilege mode only; -*n* = value after reset

Table 20-49. BTC Interrupt Flag Register (BTCFLAG) Field Descriptions

Bit	Field	Value	Description
31-0	BTCI[*n*]		Block transfer complete (BTC) flags. Bit 0 corresponds to channel 0, bit 1 corresponds to channel 1, and so on. **Note: Reading from the respective interrupt channel offset register also clears the corresponding flag (see Section 20.3.1.46 and Section 20.3.1.50).** **Note: The state of the flag bit can be polled even if the corresponding interrupt enable bit is cleared.**
		0	Read: BTC interrupt of the corresponding channel is not pending. Write: No effect.
		1	Read: BTC interrupt of the corresponding channel is pending. Write: The flag is cleared.

20.3.1.42 BER Interrupt Flag Register (BERFLAG)

The BERFLAG will never be set in this device. The bus error reporting is handled by the DMA Read Imprecise Error and DMA Write Imprecise Error asserted to the ESM module directly, which are detected at the device level. See the ESM error mapping for the DMA Read/Write Imprecise Error.

20.3.1.43 FTCA Interrupt Channel Offset Register (FTCAOFFSET)

Figure 20-60. FTCA Interrupt Channel Offset Register (FTCAOFFSET) [offset = 14Ch]

31 16
Reserved
R-0

15 8	7	6	5 0
Reserved	sbz	sbz	FTCA
R-0	R-0	R-0	R-0

LEGEND: R = Read only; -*n* = value after reset

Table 20-50. FTCA Interrupt Channel Offset Register (FTCAOFFSET) Field Descriptions

Bit	Field	Value	Description
31-16	Reserved	0	Reads return 0. Writes have no effect.
7-6	sbz	0	These bits should always be programmed as zero.
5-0	FTCA		Channel causing FTC interrupt Group A. These bits contain the channel number of the pending interrupt for Group A if the corresponding interrupt enable is set. **Note: Reading this location clears the corresponding interrupt pending flag (see Section 20.3.1.38) with the highest priority.**
		0	No interrupt is pending.
		1h	Channel 0 is causing the pending interrupt Group A.
		:	:
		20h	Channel 31 is causing the pending interrupt Group A.
		21h-3Fh	Reserved

20.3.1.44 LFSA Interrupt Channel Offset Register (LFSAOFFSET)

Figure 20-61. LFSA Interrupt Channel Offset Register (LFSAOFFSET) [offset = 150h]

31	16
Reserved	
R-0	

15	8	7	6	5	0
Reserved		sbz	sbz	LFSA	
R-0		R-0	R-0	R-0	

LEGEND: R = Read only; -*n* = value after reset

Table 20-51. LFSA Interrupt Channel Offset Register (LFSAOFFSET) Field Descriptions

Bit	Field	Value	Description
31-16	Reserved	0	Reads return 0. Writes have no effect.
7-6	sbz	0	These bits should always be programmed as zero.
5-0	LFSA		Channel causing LFS interrupt Group A. These bits contain the channel number of the pending interrupt for Group A if the corresponding interrupt enable is set. **Note: Reading this location clears the corresponding interrupt pending flag (see Section 20.3.1.39) with the highest priority.**
		0	No interrupt is pending.
		1h	Channel 0 is causing the pending interrupt Group A.
		:	:
		20h	Channel 31 is causing the pending interrupt Group A.
		21h-3Fh	Reserved

20.3.1.45 HBCA Interrupt Channel Offset Register (HBCAOFFSET)

Figure 20-62. HBCA Interrupt Channel Offset Register (HBCAOFFSET) [offset = 154h]

31	16
Reserved	
R-0	

15	8	7	6	5	0
Reserved		sbz	sbz	HBCA	
R-0		R-0	R-0	R-0	

LEGEND: R = Read only; -*n* = value after reset

Table 20-52. HBCA Interrupt Channel Offset Register (HBCAOFFSET) Field Descriptions

Bit	Field	Value	Description
31-16	Reserved	0	Reads return 0. Writes have no effect.
7-6	sbz	0	These bits should always be programmed as zero.
5-0	HBCA		Channel causing HBC interrupt Group A. These bits contain the channel number of the pending interrupt for Group A if the corresponding interrupt enable is set. **Note: Reading this location clears the corresponding interrupt pending flag (see Section 20.3.1.40) with the highest priority.**
		0	No interrupt is pending.
		1h	Channel 0 is causing the pending interrupt Group A.
		:	:
		20h	Channel 31 is causing the pending interrupt Group A.
		21h-3Fh	Reserved

20.3.1.46 BTCA Interrupt Channel Offset Register (BTCAOFFSET)

Figure 20-63. BTCA Interrupt Channel Offset Register (BTCAOFFSET) [offset = 158h]

31-16
Reserved
R-0

15-8	7	6	5-0
Reserved	sbz	sbz	BTCA
R-0	R-0	R-0	R-0

LEGEND: R = Read only; -*n* = value after reset

Table 20-53. BTCA Interrupt Channel Offset Register (BTCAOFFSET) Field Descriptions

Bit	Field	Value	Description
31-16	Reserved	0	Reads return 0. Writes have no effect.
7-6	sbz	0	These bits should always be programmed as zero.
5-0	BTCA		Channel causing BTC interrupt Group A. These bits contain the channel number of the pending interrupt for Group A if the corresponding interrupt enable is set. **Note: Reading this location clears the corresponding interrupt pending flag (see Section 20.3.1.41) with the highest priority.**
		0	No interrupt is pending.
		1h	Channel 0 is causing the pending interrupt Group A.
		:	:
		20h	Channel 31 is causing the pending interrupt Group A.
		21h-3Fh	Reserved

20.3.1.47 FTCB Interrupt Channel Offset Register (FTCBOFFSET)

Figure 20-64. FTCB Interrupt Channel Offset Register (FTCBOFFSET) [offset = 160h]

31	16
Reserved	
R-0	

15	8	7	6	5	0
Reserved		sbz	sbz	FTCB	
R-0		R-0	R-0	R-0	

LEGEND: R = Read only; -*n* = value after reset

Table 20-54. FTCB Interrupt Channel Offset Register (FTCBOFFSET) Field Descriptions

Bit	Field	Value	Description
31-16	Reserved	0	Reads return 0. Writes have no effect.
7-6	sbz	0	These bits should always be programmed as zero.
5-0	FTCB		Channel causing FTC interrupt Group B. These bits contain the channel number of the pending interrupt for Group B if the corresponding interrupt enable is set. **Note: Reading this location clears the corresponding interrupt pending flag (see Section 20.3.1.38) with the highest priority.**
		0	No interrupt is pending.
		1h	Channel 0 is causing the pending interrupt Group B.
		:	:
		20h	Channel 31 is causing the pending interrupt Group B.
		21h-3Fh	Reserved

20.3.1.48 LFSB Interrupt Channel Offset Register (LFSBOFFSET)

Figure 20-65. LFSB Interrupt Channel Offset Register (LFSBOFFSET) [offset = 164h]

31-16
Reserved
R-0

15-8	7	6	5-0
Reserved	sbz	sbz	LFSB
R-0	R-0	R-0	R-0

LEGEND: R = Read only; -*n* = value after reset

Table 20-55. LFSB Interrupt Channel Offset Register (LFSBOFFSET) Field Descriptions

Bit	Field	Value	Description
31-16	Reserved	0	Reads return 0. Writes have no effect.
7-6	sbz	0	These bits should always be programmed as zero.
5-0	LFSB		Channel causing LFS interrupt Group B. These bits contain the channel number of the pending interrupt for Group B if the corresponding interrupt enable is set. **Note: Reading this location clears the corresponding interrupt pending flag (see Section 20.3.1.39) with the highest priority.**
		0	No interrupt is pending.
		1h	Channel 0 is causing the pending interrupt Group B.
		:	:
		20h	Channel 31 is causing the pending interrupt Group B.
		21h-3Fh	Reserved

20.3.1.49 HBCB Interrupt Channel Offset Register (HBCBOFFSET)

Figure 20-66. HBCB Interrupt Channel Offset Register (HBCBOFFSET) [offset = 168h]

31	16
Reserved	
R-0	

15	8	7	6	5	0
Reserved		sbz	sbz	HBCB	
R-0		R-0	R-0	R-0	

LEGEND: R = Read only; -*n* = value after reset

Table 20-56. HBCB Interrupt Channel Offset Register (HBCBOFFSET) Field Descriptions

Bit	Field	Value	Description
31-16	Reserved	0	Reads return 0. Writes have no effect.
7-6	sbz	0	These bits should always be programmed as zero.
5-0	HBCB		Channel causing HBC interrupt Group B. These bits contain the channel number of the pending interrupt for Group B if the corresponding interrupt enable is set. **Note: Reading this location clears the corresponding interrupt pending flag (see Section 20.3.1.40) with the highest priority.**
		0	No interrupt is pending.
		1h	Channel 0 is causing the pending interrupt Group B.
		:	:
		20h	Channel 31 is causing the pending interrupt Group B.
		21h-3Fh	Reserved

20.3.1.50 BTCB Interrupt Channel Offset Register (BTCBOFFSET)

Figure 20-67. BTCB Interrupt Channel Offset Register (BTCBOFFSET) [offset = 16Ch]

31-16
Reserved
R-0

15-8	7	6	5-0
Reserved	sbz	sbz	BTCB
R-0	R-0	R-0	R-0

LEGEND: R = Read only; -*n* = value after reset

Table 20-57. BTCB Interrupt Channel Offset Register (BTCBOFFSET) Field Descriptions

Bit	Field	Value	Description
31-16	Reserved	0	Reads return 0. Writes have no effect.
7-6	sbz	0	These bits should always be programmed as zero.
5-0	BTCB		Channel causing BTC interrupt Group B. These bits contain the channel number of the pending interrupt for Group B if the corresponding interrupt enable is set. **Note: Reading this location clears the corresponding interrupt pending flag (see Section 20.3.1.41) with the highest priority.**
		0	No interrupt is pending.
		1h	Channel 0 is causing the pending interrupt Group B.
		:	:
		20h	Channel 31 is causing the pending interrupt Group B.
		21h-3Fh	Reserved

20.3.1.51 Port Control Register (PTCRL)

Figure 20-68. Port Control Register (PTCRL) [offset = 178h]

<table>
<tr><td>31</td><td>25</td><td>24</td></tr>
<tr><td colspan="2">Reserved</td><td>PENDB</td></tr>
<tr><td colspan="2">R-0</td><td>R-0</td></tr>
</table>

<table>
<tr><td>23</td><td>19</td><td>18</td><td>17</td><td>16</td></tr>
<tr><td colspan="2">Reserved</td><td>BYB</td><td colspan="2">Reserved</td></tr>
<tr><td colspan="2">R-0</td><td>R/WP-0</td><td colspan="2">R-0</td></tr>
</table>

<table>
<tr><td>15</td><td>9</td><td>8</td><td>7</td><td>3</td><td>2</td><td>1</td><td>0</td></tr>
<tr><td colspan="2">Reserved</td><td>PENDA</td><td colspan="2">Reserved</td><td>BYA</td><td>PSFRHQ</td><td>PSFRLQ</td></tr>
<tr><td colspan="2">R-0</td><td>R-0</td><td colspan="2">R-0</td><td>R/WP-0</td><td>R/WP-0</td><td>R/WP-0</td></tr>
</table>

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privilege mode only; -*n* = value after reset

Table 20-58. Port Control Register (PTCRL) Field Descriptions

Bit	Field	Value	Description
31-25	Reserved	0	Reads return 0. Writes have no effect.
24	PENDB		Transfers pending for Port B. This flag determines if transfers are ongoing on port B. The flag will be cleared if no transfers are performed. It can be used to determine if there is still data transferred while DMA_EN is cleared to 0 in GCTCRL. In this case, once all transfers are finished, the flag will be cleared to 0.
		0	No transfers are pending.
		1	Transfers are pending.
23-19	Reserved	0	Reads return 0. Writes have no effect.
18	BYB		Bypass FIFO B.
		0	FIFO B is not bypassed.
		1	FIFO B is bypassed. Writing 1 to this bit limits the FIFO depth to the size of one element. That means that after one element is read, the write-out to the destination will begin. This feature is particularly useful to minimize switching latency between channels.
			Note: This feature does not make optimal use of bus bandwidth.
17-9	Reserved	0	Reads return 0. Writes have no effect.
8	PENDA		Transfers pending for Port A. This flag determines if transfers are ongoing on port A. The flag will be cleared if no transfers are performed. It can be used to determine if there is still data transferred while DMA_EN is cleared to 0 in GCTCRL. In this case, once all transfers are finished, the flag will be cleared to 0.
		0	No transfers are pending.
		1	Transfers are pending.
7-3	Reserved	0	Reads return 0. Writes have no effect.
2	BYA		Bypass FIFO A.
		0	FIFO A is not bypassed.
		1	FIFO A is bypassed. Writing 1 to this bit limits the FIFO depth to the size of one element. That means that after one element is read, the write-out to the destination will begin. This feature is particularly useful to minimize switching latency between channels.
			Note: This feature does not make optimal use of bus bandwidth.
1	PSFRHQ		Priority scheme fix or rotate for high priority queue.
		0	Fixed priority is used.
		1	Rotation priority is used.
0	PSFRLQ		Priority scheme fix or rotate for low priority queue.
		0	The fixed priority scheme is used.
		1	The rotation priority scheme is used.

20.3.1.52 RAM Test Control Register (RTCTRL)

Figure 20-69. RAM Test Control Register (RTCTRL) [offset = 17Ch]

31	16
Reserved	
R-0	

15	1	0
Reserved		RTC
R-0		R/WP-0

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privilege mode only; -*n* = value after reset

Table 20-59. RAM Test Control Register (RTCTRL) Field Descriptions

Bit	Field	Value	Description
31-1	Reserved	0	Reads return 0. Writes have no effect.
0	RTC		RAM Test Control. Writing a 1 to this bit opens the write access to the reserved locations of control packet RAM as defined in the memory-map. **Note: This bit should be cleared to 0 during normal operation.**
		0	RAM Test Control is disabled.
		1	RAM Test Control is enabled.

20.3.1.53 Debug Control Register (DCTRL)

Figure 20-70. Debug Control Register (DCTRL) [offset = 180h]

31 29	28 24	23 17	16
Reserved	CHNUM	Reserved	DMADBGS
R-0	R-0	R-0	R/W1C-0

15 1	0
Reserved	DBGEN
R-0	R/WC-0

LEGEND: R/W = Read/Write; R = Read only; W1C = Write 1 to clear; -*n* = value after reset

Table 20-60. Debug Control Register (DCTRL) Field Descriptions

Bit	Field	Value	Description
31-29	Reserved	0	Reads return 0. Writes have no effect.
28-24	CHNUM	0-1Fh	Channel Number. This bit field indicates the channel number that causes the watch point to match.
23-17	Reserved	0	Reads return 0. Writes have no effect.
16	DMADBGS		DMA debug status. When a watch point is set up to watch for a unique bus address or a range of addresses is true on one of the three bus ports, then the DMA debug status bit is set to 1 and a debug request signal is asserted to the main CPU. The CPU must write a 1 to clear this bit for the DMA controller to release the debug request signal.
		0	Read: No watch point condition is detected. Write: No effect.
		1	Read: The watch point condition is detected. Write: The bit is cleared.
15-1	Reserved	0	Reads return 0. Writes have no effect.
0	DBGEN		Debug Enable. **Note: This bit can only be set when using a debugger.** **Note: This bit is reset when Test reset ($\overline{\text{TRST}}$) is low.**
		0	Debug is disabled.
		1	The watch point checking logics is enabled.

20.3.1.54 Watch Point Register (WPR)

Figure 20-71. Watch Point Register (WPR) [offset = 184h]

31 … 0
WP
R/W-0

LEGEND: R/W = Read/Write; -*n* = value after reset

Table 20-61. Watch Point Register (WPR) Field Descriptions

Bit	Field	Description
31-0	WP	Watch point. **Note: These bits can only be set when using a debugger.** This register is only reset by a test reset ($\overline{\text{TRST}}$). A 32-bit address can be programmed into this register as a watch point. This register is used with the watch mask register (WMR). When the DBGEN bit in the DCTRL register is set and a unique address or a range of addresses are detected on the AHB address bus of Port B, a debug request signal is sent to the ARM CPU. The state machine of the port in which the watch point condition is true is frozen.

20.3.1.55 Watch Mask Register (WMR)

Figure 20-72. Watch Mask Register (WMR) [offset = 188h]

31 … 0
WM[31:0]
R/W-0

LEGEND: R/W = Read/Write; -*n* = value after reset

Table 20-62. Watch Mask Register (WMR) Field Descriptions

Bit	Field	Value	Description
31-0	WM[*n*]		Watch mask. **Note: These bits can only be set when using a debugger.** This register is only reset by a test reset ($\overline{\text{TRST}}$).
		0	Allows the bit in the WPR register to be used for address matching for a watch point.
		1	Masks the corresponding bit in the WPR register and is disregarded in the comparison.

20.3.1.56 FIFO A Active Channel Source Address Register (FAACSADDR)

Figure 20-73. FIFO A Active Channel Source Address Register (FAACSADDR) [offset = 18Ch]

31–0
FAACSA
R-0

LEGEND: R = Read only; -*n* = value after reset

Table 20-63. FIFO A Active Channel Source Address Register (FAACSADDR) Field Descriptions

Bit	Field	Description
31-0	FAACSA	FIFO B Active Channel Source Address. This register contains the current source address of the active channel as broadcasted in Section 20.3.1.3 for FIFO B.

20.3.1.57 FIFO A Active Channel Destination Address Register (FAACDADDR)

Figure 20-74. FIFO A Active Channel Destination Address Register (FAACDADDR) [offset = 190h]

31–0
FAACDA
R-0

LEGEND: R = Read only; -*n* = value after reset

Table 20-64. FIFO A Active Channel Destination Address Register (FAACDADDR) Field Descriptions

Bit	Field	Description
31-0	FAACDA	FIFO A Active Channel Destination Address. This register contains the current destination address of the active channel as broadcasted in Section 20.3.1.3 for FIFO A.

20.3.1.58 FIFO A Active Channel Transfer Count Register (FAACTC)

Figure 20-75. FIFO A Active Channel Transfer Count Register (FAACTC) [offset = 194h]

31–29	28–16
Reserved	FAFTCOUNT
R-0	R-0

15–13	12–0
Reserved	FAETCOUNT
R-0	R-0

LEGEND: R = Read only; -*n* = value after reset

Table 20-65. Port B Active Channel Transfer Count Register (FAACTC) Field Descriptions

Bit	Field	Value	Description
31-29	Reserved	0	Reads return 0. Writes have no effect.
28-16	FAFTCOUNT	0-1FFFh	FIFO A active channel frame count. These bits contain the current frame count value of the active channel as broadcasted in Section 20.3.1.3 for FIFO A.
15-13	Reserved	0	Reads return 0. Writes have no effect.
12-0	FAETCOUNT	0-1FFFh	FIFO A active channel element count. These bits contain the current element count value of the active channel as broadcasted in Section 20.3.1.3 for FIFO A.

20.3.1.59 FIFO B Active Channel Source Address Register (FBACSADDR)

Figure 20-76. FIFO B Active Channel Source Address Register (FBACSADDR) [offset = 198h]

31	0
FBACSA	
R-0	

LEGEND: R = Read only; -*n* = value after reset

Table 20-66. FIFO B Active Channel Source Address Register (FBACSADDR) Field Descriptions

Bit	Field	Description
31-0	FBACSA	FIFO B Active Channel Source Address. This register contains the current source address of the active channel as broadcasted in Section 20.3.1.3 for FIFO B.

20.3.1.60 FIFO B Active Channel Destination Address Register (FBACDADDR)

Figure 20-77. FIFO B Active Channel Destination Address Register (FBACDADDR) [offset = 19Ch]

31	0
FBACDA	
R-0	

LEGEND: R = Read only; -*n* = value after reset

Table 20-67. FIFO B Active Channel Destination Address Register (FBACDADDR) Field Descriptions

Bit	Field	Description
31-0	FBACDA	FIFO B Active Channel Destination Address. This register contains the current destination address of the active channel as broadcasted in Section 20.3.1.3 for FIFO B.

20.3.1.61 FIFO B Active Channel Transfer Count Register (FBACTC)

Figure 20-78. FIFO B Active Channel Transfer Count Register (FBACTC) [offset = 1A0h]

31 29	28 16
Reserved	FBFTCOUNT
R-0	R-0
15 13	**12 0**
Reserved	FBETCOUNT
R-0	R-0

LEGEND: R = Read only; -*n* = value after reset

Table 20-68. FIFO B Active Channel Transfer Count Register (FBACTC) Field Descriptions

Bit	Field	Value	Description
31-29	Reserved	0	Reads return 0. Writes have no effect.
28-16	FBFTCOUNT	0-1FFFh	FIFO B active channel frame count. These bits contain the current frame count value of the active channel as broadcasted in Section 20.3.1.3 for FIFO B.
15-13	Reserved	0	Reads return 0. Writes have no effect.
12-0	FBETCOUNT	0-1FFFh	FIFO B active channel element count. These bits contain the current element count value of the active channel as broadcasted in Section 20.3.1.3 for FIFO B.

20.3.1.62 ECC Control Register (DMAPECR)

Figure 20-79. ECC Control Register (DMAPECR) [offset = 1A8h]

31 15	16
Reserved	ERRA
R-0	R/WP-0

15 9	8	7 4	3 0
Reserved	TEST	Reserved	ECC_ENA
R-0	R/WP-0	R-0	R/WP-5h

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privilege mode only; *-n* = value after reset

Table 20-69. ECC Control Register (DMAPECR) Field Descriptions

Bit	Field	Value	Description
31-17	Reserved	0	Reads return 0. Writes have no effect.
16	ERRA		Error action.
		0	If a parity error is detected on control packet x (x = 0, 1, ... n), then the enable/disable state of control packet x remains unchanged.
		1	If a parity error is detected on control packet x (x = 0, 1, ...n), then the DMA controller is disabled immediately. If a frame on control packet x is processed at the time the parity error is detected, then remaining elements of this frame will not be transferred anymore. The DMA will be disabled regardless of whether the error was detected during a read to the control packet RAM performed by the DMA state machine or by a different master.
15-9	Reserved	0	Reads return 0. Writes have no effect.
8	TEST		When this bit is set, the parity bits are memory-mapped to make them accessible by the CPU.
		0	The parity bits are not memory-mapped.
		1	The parity bits are memory-mapped.
7-4	Reserved	0	Reads return 0. Writes have no effect.
3-0	ECC_ENA		ECC enable. This bit field enables or disables the ECC check on read operations and the ECC calculation on write operations. If ECC checking is enabled and an ECC double-bit error is detected the DMA_UERR signal is activated.
		5h	The ECC check is disabled.
		All other values	The ECC check is enabled. **Note: It is recommended to write Ah to enable ECC check, to guard against soft error from flipping ECC_ENA to a disable state.**

20.3.1.63 DMA ECC Error Address Register (DMAPAR)

Figure 20-80. DMA ECC Error Address Register (DMAPAR) [offset = 1ACh]

31 25	24	23 16
Reserved	EDFLAG	Reserved
R-0	R/W1C-0	R-0

15 12	11 0
Reserved	ERRORADDRESS
R-0	R-X

LEGEND: R/W = Read/Write; R = Read only; W1C = Write 1 to clear; X= value is undefined; -*n* = value after reset

Table 20-70. DMA ECC Error Address Register (DMAPAR) Field Descriptions

Bit	Field	Value	Description
31-25	Reserved	0	Reads return 0. Writes have no effect.
24	EDFLAG		ECC Error Detection Flag. This flag indicates if an ECC error occurred on reading DMA Control packet RAM.
		0	Read: No error occurred. Write: No effect.
		1	Read: Error detected and the address is captured in DMAPAR's ERROR_ADDRESS field. Write: Clears the bit.
23-12	Reserved	0	Reads return 0. Writes have no effect.
11-0	ERRORADDRESS	0-FFFh	Error address. These bits hold the address of the first ECC error generated in the RAM. This error address is frozen from being updated until it is read by the CPU. During emulation mode when SUSPEND is high, this address is frozen even when read. **Note: The error address register will not be reset by PORRST nor by any other reset source.**

20.3.1.64 DMA Memory Protection Control Register 1 (DMAMPCTRL1)

Figure 20-81. DMA Memory Protection Control Register 1 (DMAMPCTRL1) [offset = 1B0h]

31	29	28	27	26	25	24
Reserved		INT3AB	INT3ENA	REG3AP		REG3ENA
R-0		R/WP-0	R/WP-0	R/WP-0		R/WP-0
23	**21**	**20**	**19**	**18**	**17**	**16**
Reserved		INT2AB	INT2ENA	REG2AP		REG2ENA
R-0		R/WP-0	R/WP-0	R/WP-0		R/WP-0
15	**13**	**12**	**11**	**10**	**9**	**8**
Reserved		INT1AB	INT1ENA	REG1AP		REG1ENA
R-0		R/WP-0	R/WP-0	R/WP-0		R/WP-0
7	**5**	**4**	**3**	**2**	**1**	**0**
Reserved		INT0AB	INT0ENA	REG0AP		REG0ENA
R-0		R/WP-0	R/WP-0	R/WP-0		R/WP-0

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privilege mode only; -*n* = value after reset

Table 20-71. DMA Memory Protection Control Register 1 (DMAMPCTRL1) Field Descriptions

Bit	Field	Value	Description
31-29	Reserved	0	Reads return 0. Writes have no effect.
28	INT3AB		Interrupt assignment of region 3 to Group A or Group B.
		0	The interrupt is routed to the VIM (Group A).
		1	The interrupt is routed to the second CPU (Group B).
27	INT3ENA		Interrupt enable of region 3.
		0	The interrupt is disabled.
		1	The interrupt is enabled.
26-25	REG3AP		Region 3 access permission. These bits determine the access permission for region 3.
		0	All accesses are allowed.
		1h	Read only accesses are allowed.
		2h	Write only accesses are allowed.
		3h	No accesses are allowed.
24	REG3ENA		Region 3 enable.
		0	The region is disabled (no address checking done).
		1	The region is enabled (address and access permission checking done).
23-21	Reserved	0	Reads return 0. Writes have no effect.
20	INT2AB		Interrupt assignment of region 2 to Group A or Group B.
		0	The interrupt is routed to the VIM (Group A).
		1	The interrupt is routed to the second CPU (Group B).
19	INT2ENA		Interrupt enable of region 2.
		0	The interrupt is disabled.
		1	The interrupt is enabled.
18-17	REG2AP		Region 2 access permission. These bits determine the access permission for region 2.
		0	All accesses are allowed.
		1h	Read only accesses are allowed.
		2h	Write only accesses are allowed.
		3h	No accesses are allowed.
16	REG2ENA		Region 2 enable.
		0	The region is disabled (no address checking done).
		1	The region is enabled (address and access permission checking done).

Table 20-71. DMA Memory Protection Control Register 1 (DMAMPCTRL1) Field Descriptions (continued)

Bit	Field	Value	Description
15-13	Reserved	0	Reads return 0. Writes have no effect.
12	INT1AB		Interrupt assignment of region 1 to Group A or Group B.
		0	The interrupt is routed to the VIM (Group A).
		1	The interrupt is routed to the second CPU (Group B).
11	INT1ENA		Interrupt enable of region 1.
		0	The interrupt is disabled.
		1	The interrupt is enabled.
10-9	REG1AP		Region 1 access permission. These bits determine the access permission for region 3.
		0	All accesses are allowed.
		1h	Read only accesses are allowed.
		2h	Write only accesses are allowed.
		3h	No accesses are allowed.
8	REG1ENA		Region 1 enable.
		0	The region is disabled (no address checking done).
		1	The region is enabled (address and access permission checking done).
7-5	Reserved	0	Reads return zeros and writes have no effect.
4	INT0AB		Interrupt assignment of region 0 to Group A or Group B.
		0	The interrupt is routed to the VIM (Group A).
		1	The interrupt is routed to the second CPU (Group B).
3	INT0ENA		Interrupt enable of region 0.
		0	The interrupt is disabled.
		1	The interrupt is enabled.
2-1	REG0AP		Region 0 access permission. These bits determine the access permission for region 0.
		0	All accesses are allowed.
		1h	Read only accesses are allowed.
		2h	Write only accesses are allowed.
		3h	No accesses are allowed.
0	REG0ENA		Region 0 enable.
		0	The region is disabled (no address checking done).
		1	The region is enabled (address and access permission checking done).

20.3.1.65 DMA Memory Protection Status Register 1 (DMAMPST1)

Figure 20-82. DMA Memory Protection Status Register 1 (DMAMPST1) [offset = 1B4h]

31-25	24	23-17	16
Reserved	REG3FT	Reserved	REG2FT
R-0	R/W1C-0	R-0	R/W1C-0

15-9	8	7-1	0
Reserved	REG1FT	Reserved	REG0FT
R-0	R/W1C-0	R-0	R/W1C-0

LEGEND: R/W = Read/Write; R = Read only; W1C = Write 1 to clear; -*n* = value after reset

Table 20-72. DMA Memory Protection Status Register 1 (DMAMPST1) Field Descriptions

Bit	Field	Value	Description
31-25	Reserved	0	Reads return 0. Writes have no effect.
24	REG3FT		Region 3 fault. This bit determines whether an access permission violation was detected in this region.
		0	Read: No fault was detected. Write: No effect.
		1	Read: A fault was detected. Write: The bit was cleared.
23-17	Reserved	0	Reads return 0. Writes have no effect.
16	REG2FT		Region 2 fault. This bit determines whether a access permission violation was detected in this region.
		0	Read: No fault was detected. Write: No effect.
		1	Read: A fault was detected. Write: The bit was cleared.
15-9	Reserved	0	Reads return 0. Writes have no effect.
8	REG1FT		Region 1 fault. This bit determines whether an access permission violation was detected in this region.
		0	Read: No fault was detected. Write: No effect.
		1	Read: A fault was detected. Write: The bit was cleared.
7-1	Reserved	0	Reads return 0. Writes have no effect.
0	REG0FT		Region 0 fault. This bit determines whether a access permission violation was detected in this region.
		0	Read: No fault was detected. Write: No effect.
		1	Read: A fault was detected. Write: The bit was cleared.

20.3.1.66 DMA Memory Protection Region 0 Start Address Register (DMAMPR0S)

Figure 20-83. DMA Memory Protection Region 0 Start Address Register (DMAMPR0S) [offset = 1B8h]

31 … 0
STARTADDRESS
R/WP-0

LEGEND: R/W = Read/Write; WP = Write in privilege mode only; -*n* = value after reset

Table 20-73. DMA Memory Protection Region 0 Start Address Register (DMAMPR0S) Field Descriptions

Bit	Field	Description
31-0	STARTADDRESS	Start Address defines the address at which the region begins. The effective start address is truncated to the nearest word address, that is, 0x103 = 0x100.

20.3.1.67 DMA Memory Protection Region 0 End Address Register (DMAMPR0E)

Figure 20-84. DMA Memory Protection Region 0 End Address Register (DMAMPR0E) [offset = 1BCh]

31 … 0
ENDADDRESS
R/WP-0

LEGEND: R/W = Read/Write; WP = Write in privilege mode only; -*n* = value after reset

Table 20-74. DMA Memory Protection Region 0 End Address Register (DMAMPR0E) Field Descriptions

Bit	Field	Description
31-0	ENDADDRESS	End Address defines the address at which the region ends. The end address usually is larger than the start address for this region; otherwise, the region will wrap around at the end of the address space. The end address is the start address plus the region length minus 1. The effective end address is rounded up to the nearest 32-bit word end address, that is, 0x200 = 0x203. **Note: When using 64-bit transfers, the address is rounded up to the nearest 64-bit word end address, that is, 0x200 = 0x207. All other transfers are rounded up to the nearest 32-bit word end address.**

20.3.1.68 DMA Memory Protection Region 1 Start Address Register (DMAMPR1S)

Figure 20-85. DMA Memory Protection Region 1 Start Address Register (DMAMPR1S) [offset = 1C0h]

31 ... 0
STARTADDRESS
R/WP-0

LEGEND: R/W = Read/Write; WP = Write in privilege mode only; -*n* = value after reset

Table 20-75. DMA Memory Protection Region 1 Start Address Register (DMAMPR1S) Field Descriptions

Bit	Field	Description
31-0	STARTADDRESS	Start Address defines the address at which the region begins. The effective start address is truncated to the nearest word address, that is, 0x103 = 0x100.

20.3.1.69 DMA Memory Protection Region 1 End Address Register (DMAMPR1E)

Figure 20-86. DMA Memory Protection Region 1 End Address Register (DMAMPR1E) [offset = 1C4h]

31 ... 0
ENDADDRESS
R/WP-0

LEGEND: R/W = Read/Write; WP = Write in privilege mode only; -*n* = value after reset

Table 20-76. DMA Memory Protection Region 1 End Address Register (DMAMPR1E) Field Descriptions

Bit	Field	Description
31-0	ENDADDRESS	End Address defines the address at which the region ends. The end address usually is larger than the start address for this region; otherwise, the region will wrap around at the end of the address space. The end address is the start address plus the region length minus 1. The effective end address is rounded up to the nearest 32-bit word end address, that is, 0x200 = 0x203. **Note: When using 64-bit transfers, the address is rounded up to the nearest 64-bit word end address, that is, 0x200 = 0x207. All other transfers are rounded up to the nearest 32-bit word end address.**

20.3.1.70 DMA Memory Protection Region 2 Start Address Register (DMAMPR2S)

Figure 20-87. DMA Memory Protection Region 2 Start Address Register (DMAMPR2S) [offset = 1C8h]

31	0
STARTADDRESS	
R/WP-0	

LEGEND: R/W = Read/Write; WP = Write in privilege mode only; -*n* = value after reset

Table 20-77. DMA Memory Protection Region 2 Start Address Register (DMAMPR2S) Field Descriptions

Bit	Field	Description
31-0	STARTADDRESS	Start Address defines the address at which the region begins. The effective start address is truncated to the nearest word address, that is, 0x103 = 0x100.

20.3.1.71 DMA Memory Protection Region 2 End Address Register (DMAMPR2E)

Figure 20-88. DMA Memory Protection Region 2 End Address Register (DMAMPR2E) [offset = 1CCh]

31	0
ENDADDRESS	
R/WP-0	

LEGEND: R/W = Read/Write; WP = Write in privilege mode only; -*n* = value after reset

Table 20-78. DMA Memory Protection Region 2 End Address Register (DMAMPR2E) Field Descriptions

Bit	Field	Description
31-0	ENDADDRESS	End Address defines the address at which the region ends. The end address usually is larger than the start address for this region; otherwise, the region will wrap around at the end of the address space. The end address is the start address plus the region length minus 1. The effective end address is rounded up to the nearest 32-bit word end address, that is, 0x200 = 0x203. **Note: When using 64-bit transfers, the address is rounded up to the nearest 64-bit word end address, that is, 0x200 = 0x207. All other transfers are rounded up to the nearest 32-bit word end address.**

20.3.1.72 DMA Memory Protection Region 3 Start Address Register (DMAMPR3S)

Figure 20-89. DMA Memory Protection Region 3 Start Address Register (DMAMPR3S) [offset = 1D0h]

31 ... 0
STARTADDRESS
R/WP-0

LEGEND: R/W = Read/Write; WP = Write in privilege mode only; -*n* = value after reset

Table 20-79. DMA Memory Protection Region 3 Start Address Register (DMAMPR3S) Field Descriptions

Bit	Field	Description
31-0	STARTADDRESS	Start Address defines the address at which the region begins. The effective start address is truncated to the nearest word address, that is, 0x103 = 0x100.

20.3.1.73 DMA Memory Protection Region 3 End Address Register (DMAMPR3E)

Figure 20-90. DMA Memory Protection Region 3 End Address Register (DMAMPR3E) [offset = 1D4h]

31 ... 0
ENDADDRESS
R/WP-0

LEGEND: R/W = Read/Write; WP = Write in privilege mode only; -*n* = value after reset

Table 20-80. DMA Memory Protection Region 3 End Address Register (DMAMPR3E) Field Descriptions

Bit	Field	Description
31-0	ENDADDRESS	End Address defines the address at which the region ends. The end address usually is larger than the start address for this region; otherwise, the region will wrap around at the end of the address space. The end address is the start address plus the region length minus 1. The effective end address is rounded up to the nearest 32-bit word end address, that is, 0x200 = 0x203. **Note: When using 64-bit transfers, the address is rounded up to the nearest 64-bit word end address, that is, 0x200 = 0x207. All other transfers are rounded up to the nearest 32-bit word end address.**

20.3.1.74 DMA Memory Protection Control Register 2 (DMAMPCTRL2)

Figure 20-91. DMA Memory Protection Control Register 2 (DMAMPCTRL2) [offset = 1D8h]

31	29	28	27	26	25	24
Reserved		INT7AB	INT7ENA	REG7AP		REG7ENA
R-0		R/WP-0	R/WP-0	R/WP-0		R/WP-0

23	21	20	19	18	17	16
Reserved		INT6AB	INT6ENA	REG6AP		REG6ENA
R-0		R/WP-0	R/WP-0	R/WP-0		R/WP-0

15	13	12	11	10	9	8
Reserved		INT5AB	INT5ENA	REG5AP		REG5ENA
R-0		R/WP-0	R/WP-0	R/WP-0		R/WP-0

7	5	4	3	2	1	0
Reserved		INT4AB	INT4ENA	REG4AP		REG4ENA
R-0		R/WP-0	R/WP-0	R/WP-0		R/WP-0

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privilege mode only; -*n* = value after reset

Table 20-81. DMA Memory Protection Control Register 2 (DMAMPCTRL2) Field Descriptions

Bit	Field	Value	Description
31-29	Reserved	0	Reads return 0. Writes have no effect.
28	INT7AB		Interrupt assignment of region 7 to Group A or Group B.
		0	The interrupt is routed to the VIM (Group A).
		1	The interrupt is routed to the second CPU (Group B).
27	INT7ENA		Interrupt enable of region 7.
		0	The interrupt is disabled.
		1	The interrupt is enabled.
26-25	REG7AP		Region 7 access permission. These bits determine the access permission for region 7.
		0	All accesses are allowed.
		1h	Read only accesses are allowed.
		2h	Write only accesses are allowed.
		3h	No accesses are allowed.
24	REG7ENA		Region 7 enable.
		0	The region is disabled (no address checking done).
		1	The region is enabled (address and access permission checking done).
23-21	Reserved	0	Reads return 0. Writes have no effect.
20	INT6AB		Interrupt assignment of region 6 to Group A or Group B.
		0	The interrupt is routed to the VIM (Group A).
		1	The interrupt is routed to the second CPU (Group B).
19	INT6ENA		Interrupt enable of region 6.
		0	The interrupt is disabled.
		1	The interrupt is enabled.
18-17	REG6AP		Region 6 access permission. These bits determine the access permission for region 6.
		0	All accesses are allowed.
		1h	Read only accesses are allowed.
		2h	Write only accesses are allowed.
		3h	No accesses are allowed.
16	REG6ENA		Region 6 enable.
		0	The region is disabled (no address checking done).
		1	The region is enabled (address and access permission checking done).

Table 20-81. DMA Memory Protection Control Register 2 (DMAMPCTRL2) Field Descriptions (continued)

Bit	Field	Value	Description
15-13	Reserved	0	Reads return 0. Writes have no effect.
12	INT5AB		Interrupt assignment of region 5 to Group A or Group B.
		0	The interrupt is routed to the VIM (Group A).
		1	The interrupt is routed to the second CPU (Group B).
11	INT5ENA		Interrupt enable of region 5.
		0	The interrupt is disabled.
		1	The interrupt is enabled.
10-9	REG5AP		Region 5 access permission. These bits determine the access permission for region 5.
		0	All accesses are allowed.
		1h	Read only accesses are allowed.
		2h	Write only accesses are allowed.
		3h	No accesses are allowed.
8	REG5ENA		Region 5 enable.
		0	The region is disabled (no address checking done).
		1	The region is enabled (address and access permission checking done).
7-5	Reserved	0	Reads return zeros and writes have no effect.
4	INT4AB		Interrupt assignment of region 4 to Group A or Group B.
		0	The interrupt is routed to the VIM (Group A).
		1	The interrupt is routed to the second CPU (Group B).
3	INT4ENA		Interrupt enable of region 4.
		0	The interrupt is disabled.
		1	The interrupt is enabled.
2-1	REG4AP		Region 4 access permission. These bits determine the access permission for region 4.
		0	All accesses are allowed.
		1h	Read only accesses are allowed.
		2h	Write only accesses are allowed.
		3h	No accesses are allowed.
0	REG4ENA		Region 4 enable.
		0	The region is disabled (no address checking done).
		1	The region is enabled (address and access permission checking done).

20.3.1.75 DMA Memory Protection Status Register 2 (DMAMPST2)

Figure 20-92. DMA Memory Protection Status Register 2 (DMAMPST2) [offset = 1DCh]

31	25	24	23	17	16
Reserved		REG7FT	Reserved		REG6FT
R-0		R/W1C-0	R-0		R/W1C-0

15	9	8	7	1	0
Reserved		REG5FT	Reserved		REG4FT
R-0		R/W1C-0	R-0		R/W1C-0

LEGEND: R/W = Read/Write; R = Read only; W1C = Write 1 to clear; -*n* = value after reset

Table 20-82. DMA Memory Protection Status Register 2 (DMAMPST2) Field Descriptions

Bit	Field	Value	Description
31-25	Reserved	0	Reads return 0. Writes have no effect.
24	REG7FT		Region 7 fault. This bit determines whether an access permission violation was detected in this region.
		0	Read: No fault was detected. Write: No effect.
		1	Read: A fault was detected. Write: Clears the bit.
23-17	Reserved	0	Reads return 0. Writes have no effect.
16	REG6FT		Region 6 fault. This bit determines whether a access permission violation was detected in this region.
		0	Read: No fault was detected. Write: No effect.
		1	Read: A fault was detected. Write: Clears the bit.
15-9	Reserved	0	Reads return 0. Writes have no effect.
8	REG5FT		Region 5 fault. This bit determines whether an access permission violation was detected in this region.
		0	Read: No fault was detected. Write: No effect.
		1	Read: A fault was detected. Write: Clears the bit.
7-1	Reserved	0	Reads return 0. Writes have no effect.
0	REG4FT		Region 4 fault. This bit determines whether a access permission violation was detected in this region.
		0	Read: No fault was detected. Write: No effect.
		1	Read: A fault was detected. Write: Clears the bit.

20.3.1.76 DMA Memory Protection Region 4 Start Address Register (DMAMPR4S)

Figure 20-93. DMA Memory Protection Region 4 Start Address Register (DMAMPR4S) [offset = 1E0h]

31	0
STARTADDRESS	
R/WP-0	

LEGEND: R/W = Read/Write; WP = Write in privilege mode only; -*n* = value after reset

Table 20-83. DMA Memory Protection Region 4 Start Address Register (DMAMPR4S) Field Descriptions

Bit	Field	Description
31-0	STARTADDRESS	Start Address defines the address at which the region begins. The effective start address is truncated to the nearest word address, that is, 0x103 = 0x100.

20.3.1.77 DMA Memory Protection Region 4 End Address Register (DMAMPR4E)

Figure 20-94. DMA Memory Protection Region 4 End Address Register (DMAMPR4E) [offset = 1E4h]

31	0
ENDADDRESS	
R/WP-0	

LEGEND: R/W = Read/Write; WP = Write in privilege mode only; -*n* = value after reset

Table 20-84. DMA Memory Protection Region 4 End Address Register (DMAMPR4E) Field Descriptions

Bit	Field	Description
31-0	ENDADDRESS	End Address defines the address at which the region ends. The end address usually is larger than the start address for this region; otherwise, the region will wrap around at the end of the address space. The end address is the start address plus the region length minus 1. The effective end address is rounded up to the nearest 32-bit word end address, that is, 0x200 = 0x203. **Note: When using 64-bit transfers, the address is rounded up to the nearest 64-bit word end address, that is, 0x200 = 0x207. All other transfers are rounded up to the nearest 32-bit word end address.**

20.3.1.78 DMA Memory Protection Region 5 Start Address Register (DMAMPR5S)

Figure 20-95. DMA Memory Protection Region 5 Start Address Register (DMAMPR5S) [offset = 1E8h]

31 … 0
STARTADDRESS
R/WP-0

LEGEND: R/W = Read/Write; WP = Write in privilege mode only; -*n* = value after reset

Table 20-85. DMA Memory Protection Region 5 Start Address Register (DMAMPR5S) Field Descriptions

Bit	Field	Description
31-0	STARTADDRESS	Start Address defines the address at which the region begins. The effective start address is truncated to the nearest word address, that is, 0x103 = 0x100.

20.3.1.79 DMA Memory Protection Region 5 End Address Register (DMAMPR5E)

Figure 20-96. DMA Memory Protection Region 5 End Address Register (DMAMPR5E) [offset = 1ECh]

31 … 0
ENDADDRESS
R/WP-0

LEGEND: R/W = Read/Write; WP = Write in privilege mode only; -*n* = value after reset

Table 20-86. DMA Memory Protection Region 5 End Address Register (DMAMPR5E) Field Descriptions

Bit	Field	Description
31-0	ENDADDRESS	End Address defines the address at which the region ends. The end address usually is larger than the start address for this region; otherwise, the region will wrap around at the end of the address space. The end address is the start address plus the region length minus 1. The effective end address is rounded up to the nearest 32-bit word end address, that is, 0x200 = 0x203. **Note: When using 64-bit transfers, the address is rounded up to the nearest 64-bit word end address, that is, 0x200 = 0x207. All other transfers are rounded up to the nearest 32-bit word end address.**

20.3.1.80 DMA Memory Protection Region 6 Start Address Register (DMAMPR6S)

Figure 20-97. DMA Memory Protection Region 6 Start Address Register (DMAMPR6S) [offset = 1F0h]

31 ... 0
STARTADDRESS
R/WP-0

LEGEND: R/W = Read/Write; WP = Write in privilege mode only; -*n* = value after reset

Table 20-87. DMA Memory Protection Region 6 Start Address Register (DMAMPR6S) Field Descriptions

Bit	Field	Description
31-0	STARTADDRESS	Start Address defines the address at which the region begins. The effective start address is truncated to the nearest word address, that is, 0x103 = 0x100.

20.3.1.81 DMA Memory Protection Region 6 End Address Register (DMAMPR6E)

Figure 20-98. DMA Memory Protection Region 6 End Address Register (DMAMPR6E) [offset = 1F4h]

31 ... 0
ENDADDRESS
R/WP-0

LEGEND: R/W = Read/Write; WP = Write in privilege mode only; -*n* = value after reset

Table 20-88. DMA Memory Protection Region 6 End Address Register (DMAMPR6E) Field Descriptions

Bit	Field	Description
31-0	ENDADDRESS	End Address defines the address at which the region ends. The end address usually is larger than the start address for this region; otherwise, the region will wrap around at the end of the address space. The end address is the start address plus the region length minus 1. The effective end address is rounded up to the nearest 32-bit word end address, that is, 0x200 = 0x203. **Note: When using 64-bit transfers, the address is rounded up to the nearest 64-bit word end address, that is, 0x200 = 0x207. All other transfers are rounded up to the nearest 32-bit word end address.**

20.3.1.82 DMA Memory Protection Region 7 Start Address Register (DMAMPR7S)

Figure 20-99. DMA Memory Protection Region 7 Start Address Register (DMAMPR7S) [offset = 1F8h]

31	0
STARTADDRESS	
R/WP-0	

LEGEND: R/W = Read/Write; WP = Write in privilege mode only; -*n* = value after reset

Table 20-89. DMA Memory Protection Region 7 Start Address Register (DMAMPR7S) Field Descriptions

Bit	Field	Description
31-0	STARTADDRESS	Start Address defines the address at which the region begins. The effective start address is truncated to the nearest word address, that is, 0x103 = 0x100.

20.3.1.83 DMA Memory Protection Region 7 End Address Register (DMAMPR7E)

Figure 20-100. DMA Memory Protection Region 7 End Address Register (DMAMPR7E) [offset = 1FCh]

31	0
ENDADDRESS	
R/WP-0	

LEGEND: R/W = Read/Write; WP = Write in privilege mode only; -*n* = value after reset

Table 20-90. DMA Memory Protection Region 7 End Address Register (DMAMPR7E) Field Descriptions

Bit	Field	Description
31-0	ENDADDRESS	End Address defines the address at which the region ends. The end address usually is larger than the start address for this region; otherwise, the region will wrap around at the end of the address space. The end address is the start address plus the region length minus 1. The effective end address is rounded up to the nearest 32-bit word end address, that is, 0x200 = 0x203. **Note: When using 64-bit transfers, the address is rounded up to the nearest 64-bit word end address, that is, 0x200 = 0x207. All other transfers are rounded up to the nearest 32-bit word end address.**

20.3.1.84 DMA Single-Bit ECC Control Register (DMASECCCTRL)

Figure 20-101. DMA Single-Bit ECC Control Register (DMASECCCTRL) [offset = 228h]

31 … 17	16
Reserved	SBERR
R-0	R/W1CP-0

15 … 12	11 … 8	7 … 4	3 … 0
Reserved	SBE_EVT_EN	Reserved	EDACMODE
R-0	R/WP-5h	R-0	R/WP-Ah

LEGEND: R/W = Read/Write; R = Read only; W1CP = Write 1 to clear in privilege mode only; WP = Write in privilege mode only; -*n* = value after reset

Table 20-91. DMA Single-Bit ECC Control Register (DMASECCCTRL) Field Description

Bit	Field	Value	Description
31-17	Reserved	0	Reads return 0. Writes have no effect.
16	SBERR		Error action.
		0	Read: No RAM check error has occurred. Write: No effect.
		1	Read: A single-bit error has occurred and was corrected by the SECDED logic. Write: Clears the bit.
15-12	Reserved	0	Reads return 0. Writes have no effect.
11-8	SBE_EVT_EN		Single-bit error enable.
		5h	Disable generation of single-bit error to ESM.
		Ah	Enable generation of single-bit error to ESM.
7-4	Reserved	0	Reads return 0. Writes have no effect.
3-0	EDACMODE	5h	Disable correction of SBE detected by the SECDED block.
		Ah	Enable correction of SBE detected by the SECDED block.

20.3.1.85 DMA ECC Single-Bit Error Address Register (DMAECCSBE)

Figure 20-102. DMA ECC Single-Bit Error Address Register (DMAECCSBE) [offset = 230h]

31	16
Reserved	
R-0	

15 12	11 0
Reserved	ERRORADDRESS
R-0	R-X

LEGEND: R = Read only; X= value is undefined; -*n* = value after reset

.

Table 20-92. DMA ECC Single-Bit Error Address Register (DMAECCSBE) Field Descriptions

Bit	Field	Value	Description
31-12	Reserved	0	Reads return 0. Writes have no effect.
11-0	ERRORADDRESS	0-FFFh	The DMA RAM address (offset from base address word aligned) of the ECC error location. This register gives the address of the first encountered single-bit ECC error since the SBERR flag has been clear. Subsequent single-bit ECC errors will not update this register until the SBERR flag has been cleared. This register is valid only when the SBERR flag is set. Read: This register clears to 0x0000 once it is read by the CPU. For a read issued by the debugger this address is frozen even when read. Write: No effect **Note: The error address register will not be reset by PORRST nor by any other reset source.**

20.3.1.86 FIFO A Status Register (FIFOASTAT)

Figure 20-103. FIFO A Status Register (FIFOASTAT) [offset = 240h]

31 ... 0
FFACH[31:0]
R-0

LEGEND: R = Read only; -*n* = value after reset

Table 20-93. FIFO A Status Register (FIFOASTAT) Field Descriptions

Bit	Field	Value	Description
31-0	FFACH[*n*]		Status of DMA channel running using FIFO A. Bit 0 corresponds to channel 0, bit 1 corresponds to channel 1, and so on.
		0	The channel is not being currently processed.
		1	The channel is currently being processed using FIFO A.
			Note: The status of a channel currently being processed remains active, even if emulation mode is entered or DMA is disabled by way of the DMA_EN bit. Up to 1 bit can be set in this register at any given time.

20.3.1.87 FIFO B Status Register (FIFOBSTAT)

Figure 20-104. FIFO B Status Register (FIFOBSTAT) [offset = 244h]

31 ... 0
FFBCH[31:0]
R-0

LEGEND: R = Read only; -*n* = value after reset

Table 20-94. FIFO B Status Register (FIFOBSTAT) Field Descriptions

Bit	Field	Value	Description
31-0	FFBCH[*n*]		Status of DMA channel running using FIFO B. Bit 0 corresponds to channel 0, bit 1 corresponds to channel 1, and so on.
		0	The channel is not being currently processed.
		1	The channel is currently being processed using FIFO B.
			Note: The status of a channel currently being processed remains active, even if emulation mode is entered or DMA is disabled by way of the DMA_EN bit. Up to 1 bit can be set in this register at any given time.

20.3.1.88 DMA Request Polarity Select Register 1 (DMAREQPS1)

Figure 20-105. DMA Request Polarity Select Register (DMAREQPS1) [offset = 330h]

31	0
DMAREQPS[63:32]	
R/WP-0	

LEGEND: R/W = Read/Write; WP = Write in privilege mode only; -*n* = value after reset

Table 20-95. DMA Request Polarity Select Register (DMAREQPS1) Field Descriptions

Bit	Field	Value	Description
31-0	DMAREQOS[*n*]		Polarity selection for DMA request lines for upper 32 requests, that is, request lines 63 to 32. Bit 0 corresponds to DMA Request line 32, bit 1 corresponds to DMA Request line 33, and so on.
		0	DMA Request polarity is active high.
		1	DMA Request polarity is active low.

20.3.1.89 DMA Request Polarity Select Register 0 (DMAREQPS0)

Figure 20-106. DMA Request Polarity Select Register (DMAREQPS0) [offset = 334h]

31	0
DMAREQPS[31:0]	
R/WP-0	

LEGEND: R/W = Read/Write; WP = Write in privilege mode only; -*n* = value after reset

Table 20-96. DMA Request Polarity Select Register (DMAREQPS1) Field Descriptions

Bit	Field	Value	Description
31-0	DMAREQOS[*n*]		Polarity selection for DMA request lines for lower 32 requests, that is, request lines 31 to 0. Bit 0 corresponds to DMA Request line 0, bit 1 corresponds to DMA Request line 1, and so on.
		0	DMA Request polarity is active high.
		1	DMA Request polarity is active low.

20.3.1.90 Transaction Parity Error Event Control Register (TERECTRL)

Figure 20-107. Transaction Parity Error Event Control Register (TERECTRL) [offset = 340h]

31 - 17	16
Reserved	TER_ERR
R-0	R/W1C-0

15 - 4	3 - 0
Reserved	TER_EN
R-0	R/WP-Ah

LEGEND: R/W = Read/Write; R = Read only; W1C = Write 1 to clear; WP = Write in privilege mode only; -*n* = value after reset

Table 20-97. Transaction Parity Error Event Control Register (TERECTRL) Field Descriptions

Bit	Field	Value	Description
31-17	Reserved	0	Reads return 0. Writes have no effect.
16	TER_ERR		Transactions parity error status.
		0	Read: No error occurred. Write: No effect.
		1	Read: A transaction error has occurred Write: Clears the bit.
15-4	Reserved	0	Reads return 0. Writes have no effect.
3-0	TER_EN		Transaction error event detection enable.
		5h	Write: Disable transaction error event detection by DMA . Read: Transaction error event will not be detected by DMA.
		Ah	Write: Enable transaction error event detection by DMA. Read: Transaction error event will be detected by DMA.

20.3.1.91 TER Event Flag Register (TERFLAG)

Figure 20-108. TER Event Flag Register (TERFLAG) [offset = 344h]

31 - 0
TERE[31:0]
R/W1CP-0

LEGEND: R/W = Read/Write; W1CP = Write 1 to clear in privilege mode only; -*n* = value after reset

Table 20-98. TER Event Flag Register (TERFLAG) Field Descriptions

Bit	Field	Value	Description
31-0	TERE[*n*]		If the bit is set, a TER event of the corresponding channel is pending. Bit 0 corresponds to channel 0, bit 1 corresponds to channel 1, and so on.
		0	Read: The associated TER Event of a channel is NOT pending. Write: No effect.
		1	Read: The associated TER Event of a channel is pending. Write: Clear this bit.

20.3.1.92 TER Event Channel Offset Register (TERROFFSET)

Figure 20-109. TER Event Channel Offset Register (TERROFFSET) [offset = 348h]

31 - 16
Reserved
R-0

15 - 8	7	6	5 - 0
Reserved	sbz	sbz	TER_OFF
R-0	R-0	R-0	R-x

LEGEND: R = Read only; -*n* = value after reset

Table 20-99. TER Event Channel Offset Register (TERROFFSET) Field Descriptions

Bit	Field	Value	Description
31-8	Reserved	0	Reads return 0. Writes have no effect.
7-6	sbz	0	These bits should always be programmed as zero.
5-0	TER_OFF		This register provides the offset of the first channel number that encountered bus parity failure on either port of DMA. Once this register is updated, it will not be changed by subsequent bus parity failures until TER_ERR flag is cleared. Writes have no effect.
		0	No interrupt is pending.
		1h	Channel 0 is causing the pending interrupt. (Read clears the register to 0 except when issued by a debugger).
		:	:
		20h	Channel 31 is causing the pending interrupt.
		21h-3Fh	Reserved **Note:** If both DMA ports encounter bus parity failure at the same time than lower channel number (assuming higher priority) will be stored and the other one will be ignored.

20.3.2 Channel Configuration

The channel configuration is defined by the channel control packet: channel control, transfer count, offset values, source/destination address.

- It is stored in local RAM, which is protected by parity.
- Each control packet contains a total of nine fields.
- The first six fields are programmable, while the last three fields are read only.
- The RAM is accessible by queue A and queue B state machines as well as CPU.
- When there are simultaneous accesses, the priority is resolved in a fixed priority scheme with the CPU having the highest priority.

All the control packets look the same. Following, there is the detailed layout of these registers shown for control packet 0.

20.3.2.1 Initial Source Address Register (ISADDR)

Figure 20-110. Initial Source Address Register (ISADDR) [offset = 00]

31	0
ISADDR	
R/WP-X	

LEGEND: R/W = Read/Write; WP = Write in privilege mode only; X = value is unknown; -*n* = value after reset

Table 20-100. Initial Source Address Register (ISADDR) Field Descriptions

Bit	Field	Description
31-0	ISADDR	Initial source address. These bits give the absolute 32-bit source address (physical).

20.3.2.2 Initial Destination Address Register (IDADDR)

Figure 20-111. Initial Destination Address Register (IDADDR) [offset = 04h]

31	0
IDADDR	
R/WP-X	

LEGEND: R/W = Read/Write; WP = Write in privilege mode only; X = value is unknown; -*n* = value after reset

Table 20-101. Initial Destination Address Register (IDADDR) Field Descriptions

Bit	Field	Description
31-0	IDADDR	Initial destination address. These bits give the absolute 32-bit destination address (physical).

20.3.2.3 Initial Transfer Count Register (ITCOUNT)

Figure 20-112. Initial Transfer Count Register (ITCOUNT) [offset = 08h]

31 29	28 16
Reserved	IFTCOUNT
R-X	R/WP-X

15 13	12 0
Reserved	IETCOUNT
R-X	R/WP-X

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privilege mode only; X = value is unknown; -*n* = value after reset

Table 20-102. Initial Transfer Count Register (ITCOUNT) Field Descriptions

Bit	Field	Value	Description
31-29	Reserved	0	Reads are undefined. Writes have no effect.
28-16	IFTCOUNT	0-1FFFh	Initial frame transfer count. These bits define the number of frame transfers.
15-13	Reserved	0	Reads are undefined. Writes have no effect.
12-0	IETCOUNT	0-1FFFh	Initial element transfer count. These bits define the number of element transfers. The block transfer size will be IETCOUNT x IFTCOUNT.

20.3.2.4 Channel Control Register (CHCTRL)

Figure 20-113. Channel Control Register (CHCTRL) [offset = 10h]

31 22	21 16
Reserved	CHAIN
R-X	R/WP-X

15 14	13 12	11 9	8	7 5	4 3	2 1	0
RES	WES	Reserved	TTYPE	Reserved	ADDMR	ADDMW	AIM
R/WP-X	R/WP-X	R-X	R/WP-X	R-X	R/WP-X	R/WP-X	R/WP-X

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privilege mode only; X = value is unknown; -*n* = value after reset

Table 20-103. Channel Control Register (CHCTRL) Field Descriptions

Bit	Field	Value	Description
31-22	Reserved	0	Reads are undefined. Writes have no effect.
21-16	CHAIN		Next channel to be triggered. At the end of the programmed number of frames, the specified channel will be triggered.
			Note: The programmer must program the CHAIN bits before initiating a DMA transfer.
		0	No channel is selected.
		1h	Channel 0 is selected.
		:	:
		20h	Channel 31 is selected.
		21h-3Fh	Reserved
15-14	RES		Read element size.
		0	The element is byte, 8-bit.
		1h	The element is half-word, 16-bit.
		2h	The element is word, 32-bit.
		3h	The element is double-word, 64-bit.
13-12	WES		Write element size.
		0	The element is byte, 8-bit.
		1h	The element is half-word, 16-bit.
		2h	The element is word, 32-bit.
		3h	The element is double-word, 64-bit.
11-9	Reserved	0	Reads are undefined. Writes have no effect.
8	TTYPE		Transfer type.
		0	A request triggers one frame transfer.
		1	A request triggers one block transfer.
7-5	Reserved	0	Reads are undefined. Writes have no effect.
4-3	ADDMR		Addressing mode read.
		0	Constant
		1h	Post-increment
		2h	Reserved
		3h	Indexed
2-1	ADDMW		Addressing mode write.
		0	Constant
		1h	Post-increment
		2h	Reserved
		3h	Indexed
0	AIM		Auto-initiation mode.
		0	Auto-initiation mode is disabled.
		1	Auto-initiation mode is enabled.

20.3.2.5 Element Index Offset Register (EIOFF)

Figure 20-114. Element Index Offset Register (EIOFF) [offset = 14h]

31 29	28 16
Reserved	EIDXD
R-X	R/WP-X

15 13	12 0
Reserved	EIDXS
R-X	R/WP-X

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privilege mode only; X = value is unknown; -*n* = value after reset

Table 20-104. Element Index Offset Register (EIOFF) Field Descriptions

Bit	Field	Value	Description
31-29	Reserved	0	Reads are undefined. Writes have no effect.
28-16	EIDXD	0-1FFFh	Destination address element index. These bits define the offset to be added to the destination address after each element transfer.
15-13	Reserved	0	Reads are undefined. Writes have no effect.
12-0	EIDXS	0-1FFFh	Source address element index. These bits define the offset to be added to the source address after each element transfer.

20.3.2.6 Frame Index Offset Register (FIOFF)

Figure 20-115. Frame Index Offset Register (FIOFF) [offset = 18h]

31 29	28 16
Reserved	FIDXD
R-X	R/WP-X

15 13	12 0
Reserved	FIDXS
R-X	R/WP-X

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privilege mode only; X = value is unknown; -*n* = value after reset

Table 20-105. Frame Index Offset Register (FIOFF) Field Descriptions

Bit	Field	Value	Description
31-29	Reserved	0	Reads are undefined. Writes have no effect.
28-16	FIDXD	0-1FFFh	Destination address frame index. These bits define the offset to be added to the destination address after element count reached 1.
15-13	Reserved	0	Reads are undefined. Writes have no effect.
12-0	FIDXS	0-1FFFh	Source address frame index. These bits define the offset to be added to the source address after element count reached 1.

20.3.2.7 Current Source Address Register (CSADDR)

Figure 20-116. Current Source Address Register (CSADDR) [offset = 800h]

31 ... 0
CSADDR
R-X

LEGEND: R = Read only; X = value is unknown; -*n* = value after reset

Table 20-106. Current Source Address Register (CSADDR) Field Descriptions

Bit	Field	Description
31-0	CSADDR	Current source address. These bits contain the current working absolute 32-bit source address (physical). These bits are only updated after a channel is arbitrated out from the priority queue.

20.3.2.8 Current Destination Address Register (CDADDR)

Figure 20-117. Current Destination Address Register (CDADDR) [offset = 804h]

31 ... 0
CDADDR
R-X

LEGEND: R = Read only; X = value is unknown; -*n* = value after reset

Table 20-107. Current Destination Address Register (CDADDR) Field Descriptions

Bit	Field	Description
31-0	CDADDR	Current destination address. These bits contain the current working absolute 32-bit destination address (physical). These bits are only updated after a channel is arbitrated out of the priority queue.

20.3.2.9 Current Transfer Count Register (CTCOUNT)

Figure 20-118. Current Transfer Count Register (CTCOUNT) [offset = 808h]

31 ... 29	28 ... 16
Reserved	CFTCOUNT
R-X	R-X
15 ... 13	**12 ... 0**
Reserved	CETCOUNT
R-X	R-X

LEGEND: R = Read only; X = value is unknown; -*n* = value after reset

Table 20-108. Current Transfer Count Register (CTCOUNT) Field Descriptions

Bit	Field	Value	Description
31-29	Reserved	0	Reads are undefined. Writes have no effect.
28-16	CFTCOUNT	0-1FFFh	Current frame transfer count. Returned the current remaining frame counts.
15-13	Reserved	0	Reads are undefined. Writes have no effect.
12-0	CETCOUNT	0-1FFFh	Current element transfer count. These bits return the current remaining element counts. CTCOUNT register is only updated after a channel is arbitrated out of the priority queue.

Chapter 21

SPNU563A–March 2018

External Memory Interface (EMIF)

This chapter describes the external memory Interface (EMIF).

Topic **Page**

21.1 Introduction

21.1.1 Purpose of the Peripheral

This EMIF memory controller is compliant with the JESD21-C SDR SDRAM memories utilizing a 16-bit data bus. The purpose of this EMIF is to provide a means for the CPU to connect to a variety of external devices including:

- Single data rate (SDR) SDRAM
- Asynchronous devices including NOR Flash and SRAM

The most common use for the EMIF is to interface with both a flash device and an SDRAM device simultaneously. Section 21.4 contains an example of operating the EMIF in this configuration.

21.1.2 Features

The EMIF includes many features to enhance the ease and flexibility of connecting to external SDR SDRAM and asynchronous devices.

21.1.2.1 Asynchronous Memory Support

EMIF supports asynchronous:

- SRAM memories
- NOR Flash memories

The EMIF data bus width is up to 16 bits and there are up to 22 address lines. There is an external wait input that allows slower asynchronous memories to extend the memory access. The EMIF module supports up to 3 chip selects (EMIF_nCS[4:2]). Each chip select has the following individually programmable attributes:

- Data Bus Width
- Read cycle timings: setup, hold, strobe
- Write cycle timings: setup, hold, strobe
- Bus turn-around time
- Extended Wait Option with Programmable Timeout
- Select Strobe option

21.1.2.2 Synchronous DRAM Memory Support

The EMIF module supports 16-bit SDRAM in addition to the asynchronous memories listed in Section 21.1.2.1. It has a single SDRAM chip select (EMIF_nCS[0]). SDRAM configurations that are supported are:

- One, Two and Four Bank SDRAM devices
- Devices with Eight, Nine, Ten, and Eleven Column Address
- CAS Latency of two or three clock cycles
- 16-bit Data Bus Width
- 3.3V LVCMOS Interface

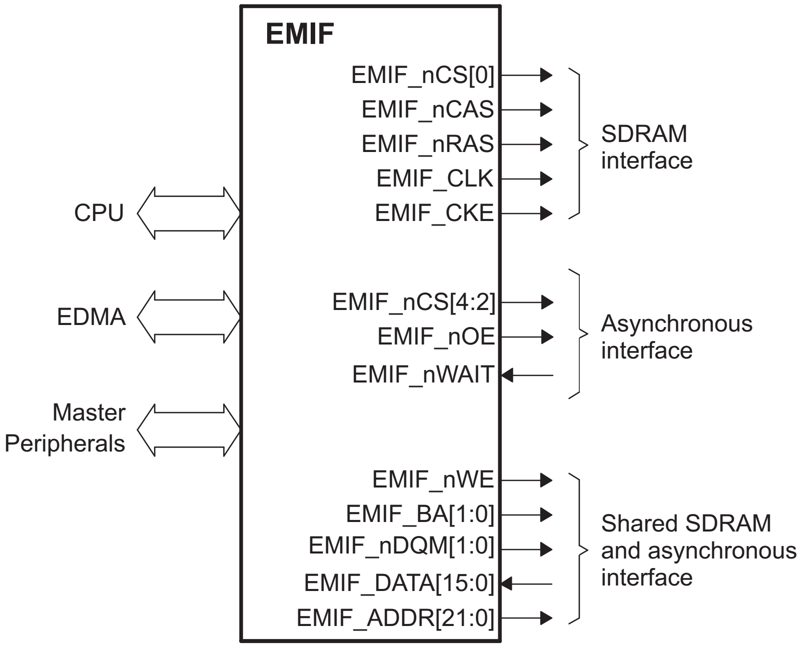
Additionally, the EMIF supports placing the SDRAM in Self-Refresh and Powerdown modes. Self-refresh mode allows the SDRAM to be put in a low-power state while still retaining memory contents; since the SDRAM will continue to refresh itself even without clocks from the microcontroller. Powerdown mode achieves even lower power, except the microcontroller must periodically wake up and issue refreshes if data retention is required.

Note that the EMIF module does not support Mobile SDRAM devices.

21.1.3 Functional Block Diagram

Figure 21-1 illustrates the connections between the EMIF and its internal requesters, along with the external EMIF pins. Section 21.2.2 contains a description of the entities internal to the SoC that can send requests to the EMIF, along with their prioritization. Section 21.2.3 describes the EMIF external pins and summarizes their purpose when interfacing with SDRAM and asynchronous devices.

Figure 21-1. EMIF Functional Block Diagram



21.2 EMIF Module Architecture

This section provides details about the architecture and operation of the EMIF. Both, SDRAM and asynchronous Interface are covered, along with other system-related issues such as clock control.

21.2.1 EMIF Clock Control

The EMIF clock is output on the EMIF_CLK pin and should be used when interfacing to external SDRAM devices. The EMIF module gets the VCLK3 clock domain as the input. This clock domain is running at half the frequency of the main oscillator by default, that is, between 2.5MHz to 10MHz. The VCLK3 frequency is divided down from the HCLK domain frequency by a programmable divider (/1 to /16). Refer the Architecture chapter of the device technical reference manual for more information on configuring the VCLK3 domain frequency.

21.2.2 EMIF Requests

Different sources within the SoC can make requests to the EMIF. These requests consist of accesses to SDRAM memory, asynchronous memory, and EMIF registers. The EMIF can process only one request at a time. Therefore a high performance crossbar switch exists within the SoC to provide prioritized requests from the different sources to the EMIF. The sources are:

1. CPU
2. DMA
3. Other master peripherals

If a request is submitted from two or more sources simultaneously, the crossbar switch will forward the highest priority request to the EMIF first. Upon completion of a request, the crossbar switch again evaluates the pending requests and forwards the highest priority pending request to the EMIF.

When the EMIF receives a request, it may or may not be immediately processed. In some cases, the EMIF will perform one or more auto refresh cycles before processing the request. For details on the EMIF's internal arbitration between performing requests and performing auto refresh cycles, see Section 21.2.13.

21.2.3 EMIF Signal Descriptions

This section describes the function of each of the EMIF signals.

Table 21-1. EMIF Pins Used to Access Both SDRAM and Asynchronous Memories

Pins(s)	I/O	Description
EMIF_DATA[15:0]	I/O	**EMIF data bus.**
EMIF_ADDR[21:0]	O	**EMIF address bus.** When interfacing to an SDRAM device, these pins are primarily used to provide the row and column address to the SDRAM. The mapping from the internal program address to the external values placed on these pins can be found in Table 21-13. EMIF_A[10] is also used during the PRE command to select which banks to deactivate. When interfacing to an asynchronous device, these pins are used in conjunction with the EMIF_BA pins to form the address that is sent to the device. The mapping from the internal program address to the external values placed on these pins can be found in Section 21.2.6.1.
EMIF_BA[1:0]	O	**EMIF bank address.** When interfacing to an SDRAM device, these pins are used to provide the bank address inputs to the SDRAM. The mapping from the internal program address to the external values placed on these pins can be found inTable 21-13. When interfacing to an asynchronous device, these pins are used in conjunction with the EMIF_A pins to form the address that is sent to the device. The mapping from the internal program address to the external values placed on these pins can be found in Section 21.2.6.1.
EMIF_nDQM[1:0]	O	**Active-low byte enables.** When interfacing to SDRAM, these pins are connected to the DQM pins of the SDRAM to individually enable/disable each of the bytes in a data access. When interfacing to an asynchronous device, these pins are connected to byte enables. See Section 21.2.6 for details.

Table 21-1. EMIF Pins Used to Access Both SDRAM and Asynchronous Memories (continued)

Pins(s)	I/O	Description
EMIF_nWE	O	**Active-low write enable.** When interfacing to SDRAM, this pin is connected to the nWE pin of the SDRAM and is used to send commands to the device. When interfacing to an asynchronous device, this pin provides a signal which is active-low during the strobe period of an asynchronous write access cycle.

Table 21-2. EMIF Pins Specific to SDRAM

Pin(s)	I/O	Description
EMIF_nCS[0]	O	**Active-low chip enable pin for SDRAM devices.** This pin is connected to the chip-select pin of the attached SDRAM device and is used for enabling/disabling commands. By default, the EMIF keeps this SDRAM chip select active, even if the EMIF is not interfaced with an SDRAM device. This pin is deactivated when accessing the asynchronous memory bank and is reactivated on completion of the asynchronous access.
EMIF_nRAS	O	**Active-low row address strobe pin.** This pin is connected to the nRAS pin of the attached SDRAM device and is used for sending commands to the device.
EMIF_nCAS	O	**Active-low column address strobe pin.** This pin is connected to the nCAS pin of the attached SDRAM device and is used for sending commands to the device.
EMIF_CKE	O	**Clock enable pin.** This pin is connected to the CKE pin of the attached SDRAM device and is used for issuing the SELF REFRESH command which places the device in self refresh mode. See Section 21.2.5.7 for details.
EMIF_CLK	O	**SDRAM clock pin.** This pin is connected to the CLK pin of the attached SDRAM device. See Section 21.2.1 for details on the clock signal.

Table 21-3. EMIF Pins Specific to Asynchronous Memory

Pin(s)	I/O	Description
EMIF_nCS[4:2]	O	**Active-low chip enable pins for asynchronous devices.** These pins are meant to be connected to the chip-select pins of the attached asynchronous device. These pins are active only during accesses to the asynchronous memory.
EMIF_nWAIT	I	**Wait input with programmable polarity.** A connected asynchronous device can extend the strobe period of an access cycle by asserting the EMIF_nWAIT input to the EMIF as described in Section 21.2.6.6. To enable this functionality, the EW bit in the asynchronous 1 configuration register (CE2CFG) must be set to 1. In addition, the WP0 bit in CE2CFG must be configured to define the polarity of the EMIF_nWAIT pin.
EMIF_nOE	O	**Active-low pin enable for asynchronous devices.** This pin provides a signal which is active-low during the strobe period of an asynchronous read access cycle.

21.2.4 EMIF Signal Multiplexing Control

Several EMIF signals are multiplexed with other functions on this microcontroller. Please refer to the I/O Multiplexing Module chapter of the technical reference manual for more information on how to enable the output of these EMIF signals.

21.2.5 SDRAM Controller and Interface

The EMIF can gluelessly interface to most standard SDR SDRAM devices and supports such features as self refresh mode and prioritized refresh. In addition, it provides flexibility through programmable parameters such as the refresh rate, CAS latency, and many SDRAM timing parameters. The following sections include details on how to Interface and properly configure the EMIF to perform read and write operations to externally connected SDR SDRAM devices. Also, Section 21.4 provides a detailed example of interfacing the EMIF to a common SDRAM device.

21.2.5.1 SDRAM Commands

The EMIF supports the SDRAM commands described in Table 21-4. Table 21-5 shows the truth table for the SDRAM commands, and an example timing waveform of the PRE command is shown in Figure 21-2. EMIF_A[10] is pulled low in this example to deactivate only the bank specified by the EMIF_BA pins.

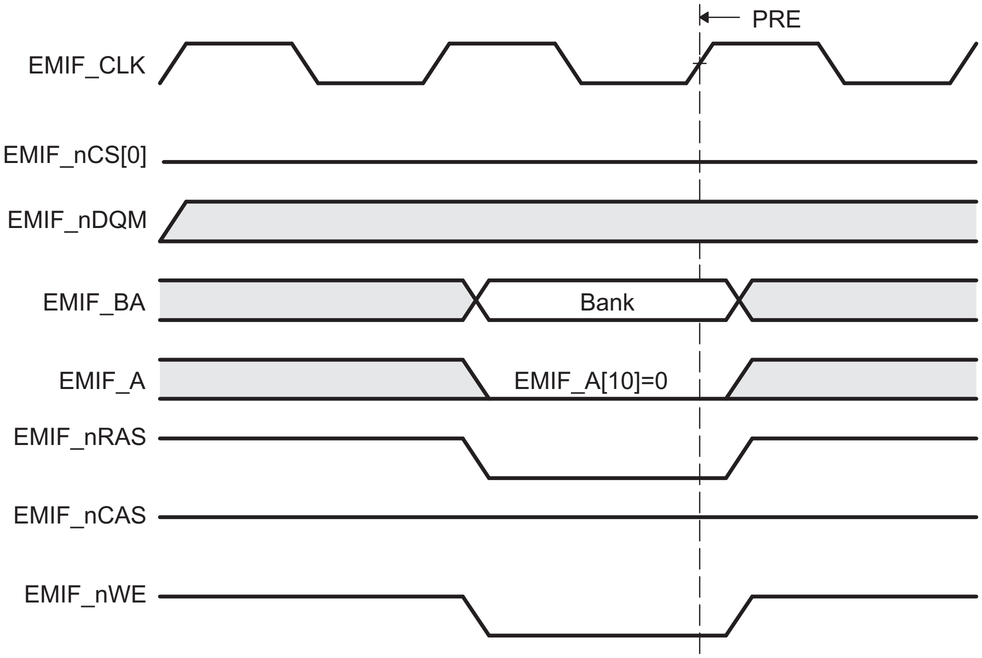
Table 21-4. EMIF SDRAM Commands

Command	Function
PRE	**Precharge.** Depending on the value of EMIF_A[10], the PRE command either deactivates the open row in all banks (EMIF_A[10] = 1) or only the bank specified by the EMIF_BA[1:0] pins (EMIF_A[10] = 0).
ACTV	**Activate**. The ACTV command activates the selected row in a particular bank for the current access.
READ	**Read.** The READ command outputs the starting column address and signals the SDRAM to begin the burst read operation. Address EMIF_A[10] is always pulled low to avoid auto precharge. This allows for better bank interleaving performance.
WRT	**Write.** The WRT command outputs the starting column address and signals the SDRAM to begin the burst write operation. Address EMIF_A[10] is always pulled low to avoid auto precharge. This allows for better bank interleaving performance.
BT	**Burst terminate**. The BT command is used to truncate the current read or write burst request.
LMR	**Load mode register**. The LMR command sets the mode register of the attached SDRAM devices and is only issued during the SDRAM initialization sequence described in Section 21.2.5.4.
REFR	**Auto refresh**. The REFR command signals the SDRAM to perform an auto refresh according to its internal address.
SLFR	**Self refresh**. The self refresh command places the SDRAM into self refresh mode, during which it provides its own clock signal and auto refresh cycles.
NOP	**No operation**. The NOP command is issued during all cycles in which one of the above commands is not issued.

Table 21-5. Truth Table for SDRAM Commands

SDRAM Pins:	CKE	nCS	nRAS	nCAS	nWE	BA[1:0]	A[12:11]	A[10]	A[9:0]
EMIF Pins:	**EMIF_CKE**	**EMIF_nCS[0]**	**EMIF_nRAS**	**EMIF_nCAS**	**EMIF_nWE**	**EMIF_BA[1:0]**	**EMIF_A[12:11]**	**EMIF_A[10]**	**EMIF_A[9:0]**
PRE	H	L	L	H	L	Bank/X	X	L/H	X
ACTV	H	L	L	H	H	Bank	Row	Row	Row
READ	H	L	H	L	H	Bank	Column	L	Column
WRT	H	L	H	L	L	Bank	Column	L	Column
BT	H	L	H	H	L	X	X	X	X
LMR	H	L	L	L	L	X	Mode	Mode	Mode
REFR	H	L	L	L	H	X	X	X	X
SLFR	L	L	L	L	H	X	X	X	X
NOP	H	L	H	H	H	X	X	X	X

Figure 21-2. Timing Waveform of SDRAM PRE Command



21.2.5.2 Interfacing to SDRAM

The EMIF supports a glueless interface to SDRAM devices with the following characteristics:

- Pre-charge bit is A[10]
- The number of column address bits is 8, 9, 10, or 11.
- The number of row address bits is 13, 14, 15, or 16.
- The number of internal banks is 1, 2, or 4.

Figure 21-3 shows an interface between the EMIF and a 2M × 16 × 4 bank SDRAM device, and Figure 21-4 shows an interface between the EMIF and a 512K × 16 × 2 bank SDRAM device. For devices supporting 16-bit interface, refer to Table 21-6 for list of commonly-supported SDRAM devices and the required connections for the address pins.

Figure 21-3. EMIF to 2M × 16 × 4 bank SDRAM Interface

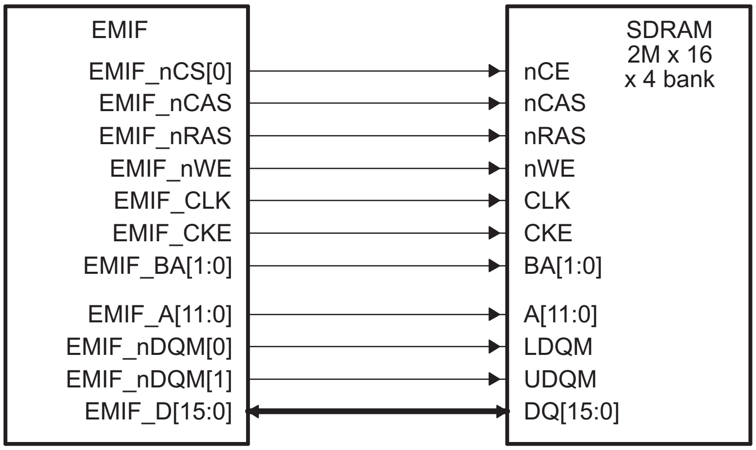


Figure 21-4. EMIF to 512K × 16 × 2 bank SDRAM Interface

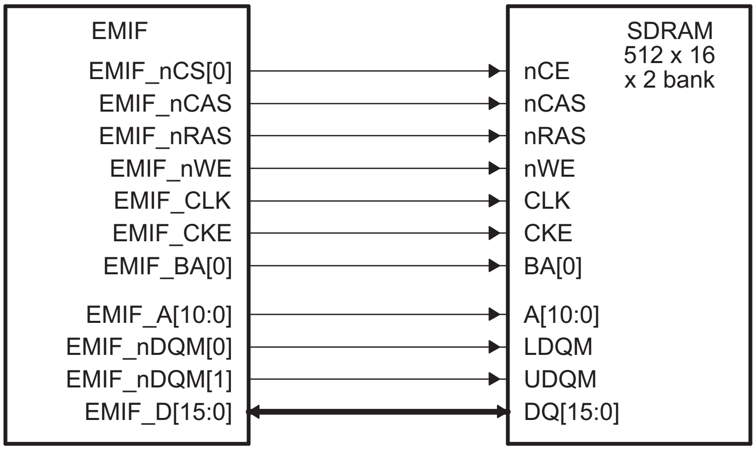


Table 21-6. 16-bit EMIF Address Pin Connections

SDRAM Size	Width	Banks	Device	Address Pins
16M bits	×16	2	SDRAM	A[10:0]
			EMIF	EMIF_A[10:0]
64M bits	×16	4	SDRAM	A[11:0]
			EMIF	EMIF_A[11:0]
128M bits	×16	4	SDRAM	A[11:0]
			EMIF	EMIF_A[11:0]
256M bits	x16	4	SDRAM	A[12:0]
			EMIF	EMIF_A[12:0]
512M bits	x16	4	SDRAM	A[12:0]
			EMIF	EMIF_A[12:0]

21.2.5.3 SDRAM Configuration Registers

The operation of the EMIF's SDRAM interface is controlled by programming the appropriate configuration registers. This section describes the purpose and function of each configuration register, but Section 21.3 should be referred for a more detailed description of each register, including the default registers values and bit-field positions. The following tables list the four such configuration registers, along with a description of each of their programmable fields.

NOTE: Writing to any of the fields: NM, CL, IBANK, and PAGESIZE in the SDRAM configuration register (SDCR) causes the EMIF to abandon whatever it is currently doing and trigger the SDRAM initialization procedure described in Section 21.2.5.4.

Table 21-7. Description of the SDRAM Configuration Register (SDCR)

Parameter	Description
SR	This bit controls entering and exiting of the Self-Refresh mode. The field should be written using a byte-write to the upper byte of SDCR to avoid triggering the SDRAM initialization sequence.
PD	This bit controls entering and exiting of the Power down mode. The field should be written using a byte-write to the upper byte of SDCR to avoid triggering the SDRAM initialization sequence. If both SR and PD bits are set, the EMIF will go into Self Refresh.
PDWR	Perform refreshes during Power Down. Writing a 1 to this bit will cause the EMIF to exit the power down state and issue an AUTO REFRESH command every time Refresh May level is set. The field should be written using a byte-write to the upper byte of SDCR to avoid triggering the SDRAM initialization sequence. This bit should be set along with PD when entering power-down mode.
NM	**Narrow Mode.** This bit defines the width of the data bus between the EMIF and the attached SDRAM device. When set to 1, the data bus is set to 16-bits. When set to 0, the data bus is set to 32-bits. This bit must always be set to 1.
CL	**CAS latency.** This field defines the number of clock cycles between when an SDRAM issues a READ command and when the first piece of data appears on the bus. The value in this field is sent to the attached SDRAM device via the LOAD MODE REGISTER command during the SDRAM initialization procedure as described in Section 21.2.5.4. Only, values of 2h (CAS latency = 2) and 3h (CAS latency = 3) are supported and should be written to this field. A 1 must be simultaneously written to the BIT11_9LOCK bit field of SDCR in order to write to the CL bit field.
IBANK	**Number of Internal SDRAM Banks.** This field defines the number of banks inside the attached SDRAM devices in the following way: • When IBANK = 0, 1 internal bank is used • When IBANK = 1h, 2 internal banks are used • When IBANK = 2h, 4 internal banks are used This field value affects the mapping of logical addresses to SDRAM row, column, and bank addresses. See Section 21.2.5.11 for details.
PAGESIZE	**Page Size.** This field defines the internal page size of the attached SDRAM devices in the following way: • When PAGESIZE = 0, 256-word pages are used • When PAGESIZE = 1h, 512-word pages are used • When PAGESIZE = 2h, 1024-word pages are used • When PAGESIZE = 3h, 2048-word pages are used This field value affects the mapping of logical addresses to SDRAM row, column, and bank addresses. See Section 21.2.5.11 for details.

Table 21-8. Description of the SDRAM Refresh Control Register (SDRCR)

Parameter	Description
RR	**Refresh Rate**. This field controls the rate at which attached SDRAM devices will be refreshed. The following equation can be used to determine the required value of RR for an SDRAM device: • RR = f_{EMIF_CLK} / (Required SDRAM Refresh Rate) More information about the operation of the SDRAM refresh controller can be found in Section 21.2.5.6.

Table 21-9. Description of the SDRAM Timing Register (SDTIMR)

Parameter	Description
T_RFC T_RP T_RCD T_WR T_RAS T_RC T_RRD	**SDRAM Timing Parameters.** These fields configure the EMIF to comply with the AC timing requirements of the attached SDRAM devices. This allows the EMIF to avoid violating SDRAM timing constraints and to more efficiently schedule its operations. More details about each of these parameters can be found in the register description in Section 21.3.6. These parameters should be set to satisfy the corresponding timing requirements found in the SDRAM's datasheet.

Table 21-10. Description of the SDRAM Self Refresh Exit Timing Register (SDSRETR)

Parameter	Description
T_XS	**Self Refresh Exit Parameter.** The T_XS field of this register informs the EMIF about the minimum number of EMIF_CLK cycles required between exiting Self Refresh and issuing any command. This parameter should be set to satisfy the t_{XSR} value for the attached SDRAM device.

21.2.5.4 SDRAM Auto-Initialization Sequence

The EMIF automatically performs an SDRAM initialization sequence, regardless of whether it is interfaced to an SDRAM device, when either of the following two events occur:

- The EMIF comes out of reset. No memory accesses to the SDRAM and Asynchronous interfaces are performed until this auto-initialization is complete.
- A write is performed to any of the three least significant bytes of the SDRAM configuration register (SDCR)

An SDRAM initialization sequence consists of the following steps:

1. If the initialization sequence is activated by a write to SDCR, and if any of the SDRAM banks are open, the EMIF issues a PRE command with EMIF_A[10] held high to indicate all banks. This is done so that the maximum ACTV to PRE timing for an SDRAM is not violated.
2. The EMIF drives EMIF_CKE high and begins continuously issuing NOP commands until eight SDRAM refresh intervals have elapsed. An SDRAM refresh interval is equal to the value of the RR field of SDRAM refresh control register (SDRCR), divided by the frequency of EMIF_CLK (RR/f_{EMIF_CLK}). This step is used to avoid violating the Power-up constraint of most SDRAM devices that requires 200 μs (sometimes 100 μs) between receiving stable Vdd and CLK and the issuing of a PRE command. Depending on the frequency of EMIF_CLK, this step may or may not be sufficient to avoid violating the SDRAM constraint. See Section 21.2.5.5 for more information.
3. After the refresh intervals have elapsed, the EMIF issues a PRE command with EMIF_A[10] held high to indicate all banks.
4. The EMIF issues eight AUTO REFRESH commands.
5. The EMIF issues the LMR command with the EMIF_A[9:0] pins set as described in Table 21-11.
6. Finally, the EMIF performs a refresh cycle, which consists of the following steps:
   a. Issuing a PRE command with EMIF_A[10] held high if any banks are open
   b. Issuing an REF command

Table 21-11. SDRAM LOAD MODE REGISTER Command

EMIF_A[9:7]	EMIF_A[6:4]	EMIF_A[3]	EMIF_A[2:0]
0 (Write bursts are of the programmed burst length in EMIF_A[2:0])	These bits control the CAS latency of the SDRAM and are set according to CL field in the SDRAM configuration register (SDCR) as follows: • If CL = 2, EMIF_A[6:4] = 2h (CAS latency = 2) • If CL = 3, EMIF_A[6:4] = 3h (CAS latency = 3)	0 (Sequential Burst Type. Interleaved Burst Type not supported)	These bits control the burst length of the SDRAM and are set according to the NM field in the SDRAM configuration register (SDCR) as follows: • If NM = 0, EMIF_A[2:0] = 2h (Burst Length = 4) • If NM = 1, EMIF_A[2:0] = 3h (Burst Length = 8)

21.2.5.5 SDRAM Configuration Procedure

There are two different SDRAM configuration procedures. Although EMIF automatically performs the SDRAM initialization sequence described in Section 21.2.5.4 when coming out of reset, it is recommended to follow one of the procedures listed below before performing any EMIF memory requests. Procedure A should be followed if it is determined that the SDRAM Power-up constraint was not violated during the SDRAM Auto-Initialization Sequence detailed in Section 21.2.5.4 on coming out of Reset. The SDRAM Power-up constraint specifies that 200 μs (sometimes 100 μs) should exist between receiving stable Vdd and CLK and the issuing of a PRE command. Procedure B should be followed if the SDRAM Power-up constraint was violated. The 200 μs (100 μs) SDRAM Power-up constraint will be violated if the frequency of EMIF_CLK is greater than 50 MHz (100 MHz for 100 μs SDRAM power-up constraint) during SDRAM Auto-Initialization Sequence. Procedure B should be followed if there is any doubt that the Power-up constraint was not met.

Procedure A — Following is the procedure to be followed if the SDRAM Power-up constraint was NOT violated:

1. Place the SDRAM into Self-Refresh Mode by setting the SR bit of SDCR to 1. A byte-write to the upper byte of SDCR should be used to avoid restarting the SDRAM Auto-Initialization Sequence described in Section 21.2.5.4. The SDRAM should be placed into Self-Refresh mode when changing the frequency of EMIF_CLK to avoid incurring the 200 μs Power-up constraint again.
2. Configure the desired EMIF_CLK clock frequency. The frequency of the memory clock must meet the timing requirements in the SDRAM manufacturer's documentation and the timing limitations shown in the electrical specifications of the device datasheet.
3. Remove the SDRAM from Self-Refresh Mode by clearing the SR bit of SDCR to 0. A byte-write to the upper byte of SDCR should be used to avoid restarting the SDRAM Auto-Initialization Sequence described in Section 21.2.5.4.
4. Program SDTIMR and SDSRETR to satisfy the timing requirements for the attached SDRAM device. The timing parameters should be taken from the SDRAM datasheet.
5. Program the RR field of SDRCR to match that of the attached device's refresh interval. See Section 21.2.5.6.1 details on determining the appropriate value.
6. Program SDCR to match the characteristics of the attached SDRAM device. This will cause the auto-initialization sequence in Section 21.2.5.4 to be re-run. This second initialization generally takes much less time due to the increased frequency of EMIF_CLK.

Procedure B — Following is the procedure to be followed if the SDRAM Power-up constraint was violated:

1. Configure the desired EMIF_CLK clock frequency. The frequency of the memory clock must meet the timing requirements in the SDRAM manufacturer's documentation and the timing limitations shown in the electrical specifications of the device datasheet.
2. Program SDTIMR and SDSRETR to satisfy the timing requirements for the attached SDRAM device. The timing parameters should be taken from the SDRAM datasheet.
3. Program the RR field of SDRCR such that the following equation is satisfied: $(RR \times 8)/(f_{EMIF\_CLK}) >$ 200 μs (sometimes 100 μs). For example, an EMIF_CLK frequency of 100 MHz would require setting RR to 2501 (9C5h) or higher to meet a 200 μs constraint.

4. Program SDCR to match the characteristics of the attached SDRAM device. This will cause the auto-initialization sequence in Section 21.2.5.4 to be re-run with the new value of RR.
5. Perform a read from the SDRAM to assure that step 5 of this procedure will occur after the initialization process has completed. Alternatively, wait for 200 µs instead of performing a read.
6. Finally, program the RR field to match that of the attached device's refresh interval. See Section 21.2.5.6.1 details on determining the appropriate value.

After following the above procedure, the EMIF is ready to perform accesses to the attached SDRAM device. See Section 21.4 for an example of configuring the SDRAM interface.

21.2.5.6 EMIF Refresh Controller

An SDRAM device requires that each of its rows be refreshed at a minimum required rate. The EMIF can meet this constraint by performing auto refresh cycles at or above this required rate. An auto refresh cycle consists of issuing a PRE command to all banks of the SDRAM device followed by issuing a REFR command. To inform the EMIF of the required rate for performing auto refresh cycles, the RR field of the SDRAM refresh control register (SDRCR) must be programmed. The EMIF will use this value along with two internal counters to automatically perform auto refresh cycles at the required rate. The auto refresh cycles cannot be disabled, even if the EMIF is not interfaced with an SDRAM. The remainder of this section details the EMIF's refresh scheme and provides an example for determining the appropriate value to place in the RR field of SDRCR.

The two counters used to perform auto-refresh cycles are a 13-bit refresh interval counter and a 4-bit refresh backlog counter. At reset and upon writing to the RR field, the refresh interval counter is loaded with the value from RR field and begins decrementing, by one, each EMIF clock cycle. When the refresh interval counter reaches zero, the following actions occur:

- The refresh interval counter is reloaded with the value from the RR field and restarts decrementing.
- The 4-bit refresh backlog counter increments unless it has already reached its maximum value.

The refresh backlog counter records the number of auto refresh cycles that the EMIF currently has outstanding. This counter is decremented by one each time an auto refresh cycle is performed and incremented by one each time the refresh interval counter expires. The refresh backlog counter saturates at the values of 0000b and 1111b. The EMIF uses the refresh backlog counter to determine the urgency with which an auto refresh cycle should be performed. The four levels of urgency are described in Table 21-12. This refresh scheme allows the required refreshes to be performed with minimal impact on access requests.

Table 21-12. Refresh Urgency Levels

Urgency Level	Refresh Backlog Counter Range	Action Taken
Refresh May	1-3	An auto-refresh cycle is performed only if the EMIF has no requests pending and none of the SDRAM banks are open.
Refresh Release	4-7	An auto-refresh cycle is performed if the EMIF has no requests pending, regardless of whether any SDRAM banks are open.
Refresh Need	8-11	An auto-refresh cycle is performed at the completion of the current access unless there are read requests pending.
Refresh Must	12-15	Multiple auto-refresh cycles are performed at the completion of the current access until the Refresh Release urgency level is reached. At that point, the EMIF can begin servicing any new read or write requests.

21.2.5.6.1 Determining the Appropriate Value for the RR Field

The value that should be programmed into the RR field of SDRCR can be calculated by using the frequency of the EMIF_CLK signal (f_{EMIF_CLK}) and the required refresh rate of the SDRAM ($f_{Refresh}$). The following formula can be used:

$RR = f_{EMIF\_CLK} / f_{Refresh}$

The SDRAM datasheet often communicates the required SDRAM Refresh Rate in terms of the number of REFR commands required in a given time interval. The required SDRAM Refresh Rate in the formula above can therefore be calculated by dividing the number of required cycles per time interval (n_{cycles}) by the time interval given in the datasheet ($t_{Refresh\ Period}$) :

$f_{Refresh} = n_{cycles} / t_{Refresh\ Period}$

Combining these formulas, the value that should be programmed into the RR field can be computed as:

$RR = f_{EMIF\_CLK} \times t_{Refresh\ Period} / n_{cycles}$

The following example illustrates calculating the value of RR. Given that:

- f_{EMIF_CLK} = 100 MHz (frequency of the EMIF clock)
- $t_{Refresh\ Period}$ = 64 ms (required refresh interval of the SDRAM)
- n_{cycles} = 8192 (number of cycles in a refresh interval for the SDRAM)

RR can be calculated as:

RR = 100 MHz × 64 ms/8192

RR = 781.25

RR = 782 cycles = 30Eh cycles

21.2.5.7 Self-Refresh Mode

The EMIF can be programmed to enter the self-refresh state by setting the SR bit of SDCR to 1. This will cause the EMIF to issue the SLFR command after completing any outstanding SDRAM access requests and clearing the refresh backlog counter by performing one or more auto refresh cycles. This places the attached SDRAM device into self-refresh mode in which it consumes a minimal amount of power while performing its own refresh cycles. The SR bit should be set and cleared using a byte-write to the upper byte of the SDRAM configuration register (SDCR) to avoid triggering the SDRAM initialization sequence.

While in the self-refresh state, the EMIF continues to service asynchronous bank requests and register accesses as normal, with one caveat. The EMIF will not park the data bus following a read to asynchronous memory while in the self-refresh state. Instead, the EMIF tri-states the data bus. Therefore, it is not recommended to perform asynchronous read operations while the EMIF is in the self-refresh state, in order to prevent floating inputs on the data bus. More information about data bus parking can be found in Section 21.2.7.

The EMIF will exit from the self-refresh state if either of the following events occur:

- The SR bit of SDCR is cleared to 0.
- An SDRAM accesses is requested.

The EMIF exits from the self-refresh state by driving EMIF_CKE high and performing an auto refresh cycle.

The attached SDRAM device should also be placed into Self-Refresh Mode when changing the frequency of EMIF_CLK. If the frequency of EMIF_CLK changes while the SDRAM is not in Self-Refresh Mode, Procedure B in Section 21.2.5.5 should be followed to reinitialize the device.

21.2.5.8 Power Down Mode

To support low-power modes, the EMIF can be requested to issue a POWER DOWN command to the SDRAM by setting the PD bit in the SDRAM configuration register (SDCR). When this bit is set, the EMIF will continue normal operation until all outstanding memory access requests have been serviced and the SDRAM refresh backlog (if there is one) has been cleared. At this point the EMIF will enter the power-down state. Upon entering this state, the EMIF will issue a POWER DOWN command (same as a NOP command but driving EMIF_CKE low on the same cycle). The EMIF then maintains EMIF_CKE low until it exits the power-down state.

Since the EMIF services the refresh backlog before it enters the power-down state, all internal banks of the SDRAM are closed (precharged) prior to issuing the POWER DOWN command. Therefore, the EMIF only supports Precharge Power Down. The EMIF does not support Active Power Down, where internal banks of the SDRAM are open (active) before the POWER DOWN command is issued.

During the power-down state, the EMIF services the SDRAM, asynchronous memory, and register accesses as normal, returning to the power-down state upon completion.

The PDWR bit in SDCR indicates whether the EMIF should perform refreshes in power-down state. If the PDWR bit is set, the EMIF exits the power-down state every time the Refresh Must level is set, performs AUTO REFRESH commands to the SDRAM, and returns back to the power-down state. This evenly distributes the refreshes to the SDRAM in power-down state. If the PDWR bit is not set, the EMIF does not perform any refreshes to the SDRAM. Therefore, the data integrity of the SDRAM is not assured upon power down exit if the PDWR bit is not set.

If the PD bit is cleared while in the power-down state, the EMIF will come out of the power-down state. The EMIF:

- Drives EMIF_CKE high.
- Enters its idle state.

21.2.5.9 SDRAM Read Operation

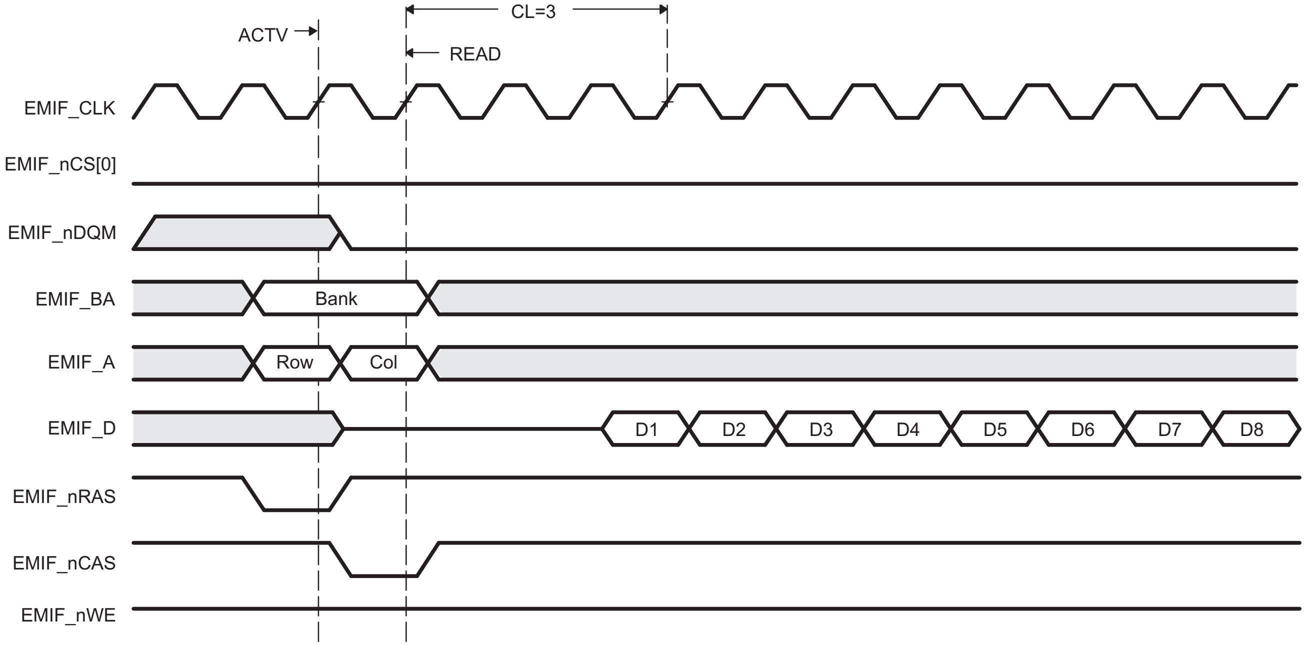
When the EMIF receives a read request to SDRAM from one of the requesters listed in Section 21.2.2, it performs one or more read access cycles. A read access cycle begins with the issuing of the ACTV command to select the desired bank and row of the SDRAM device. After the row has been opened, the EMIF proceeds to issue a READ command while specifying the desired bank and column address. EMIF_A[10] is held low during the READ command to avoid auto-precharging. The READ command signals the SDRAM device to start bursting data from the specified address while the EMIF issues NOP commands. Following a READ command, the CL field of the SDRAM configuration register (SDCR) defines how many delay cycles will be present before the read data appears on the data bus. This is referred to as the CAS latency.

Figure 21-5 shows the signal waveforms for a basic SDRAM read operation in which a burst of data is read from a single page. When the EMIF SDRAM interface is configured to 16 bit by setting the NM bit of the SDRAM configuration register (SDCR) to 1, a burst size of eight is used. Figure 21-5 shows a burst size of eight.

The EMIF will truncate a series of bursting data if the remaining addresses of the burst are not required to complete the request. The EMIF can truncate the burst in three ways:

- By issuing another READ to the same page in the same bank.
- By issuing a PRE command in order to prepare for accessing a different page of the same bank.
- By issuing a BT command in order to prepare for accessing a page in a different bank.

Figure 21-5. Timing Waveform for Basic SDRAM Read Operation



Several other pins are also active during a read access. The EMIF_nDQM[1:0] pins are driven low during the READ commands and are kept low during the NOP commands that correspond to the burst request. The state of the other EMIF pins during each command can be found in Table 21-5.

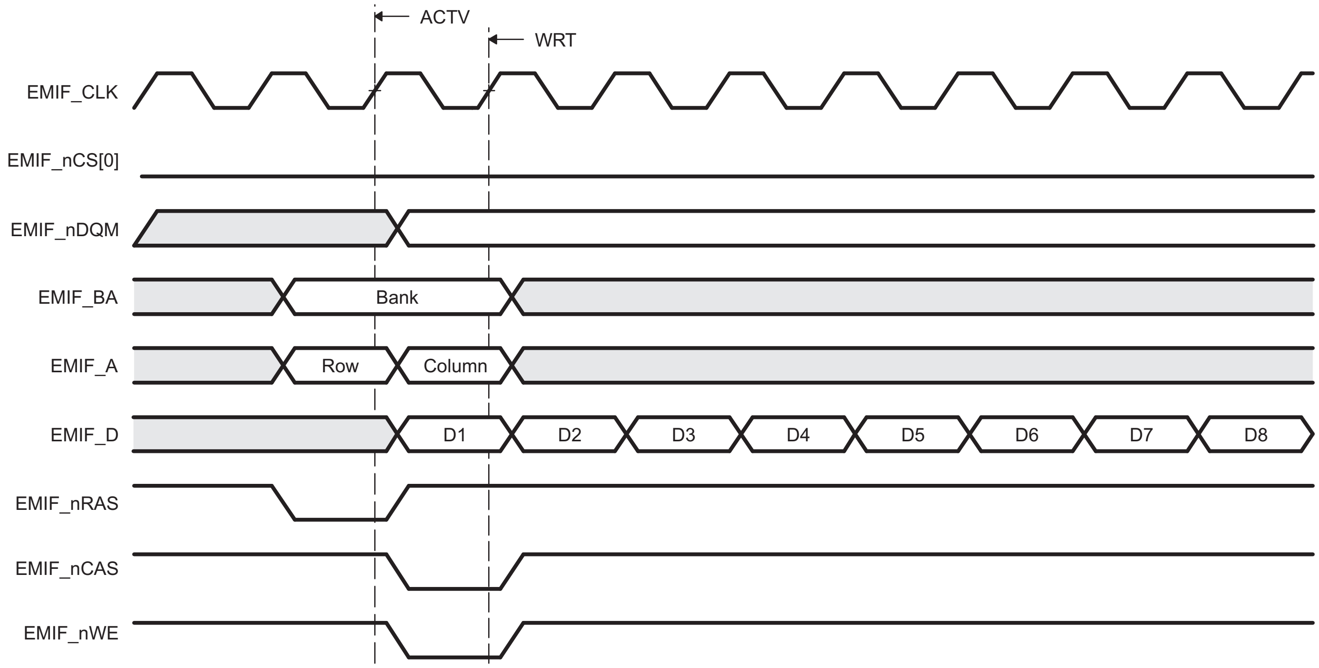
The EMIF schedules its commands based on the timing information that is provided to it in the SDRAM timing register (SDTIMR). The values for the timing parameters in this register should be chosen to satisfy the timing requirements listed in the SDRAM datasheet. The EMIF uses this timing information to avoid violating any timing constraints related to issuing commands. This is commonly accomplished by inserting NOP commands between various commands during an access. Refer to the register description of SDTIMR in Section 21.3.6 for more details on the various timing parameters.

21.2.5.10 SDRAM Write Operations

When the EMIF receives a write request to SDRAM from one of the requesters listed in Section 21.2.2, it performs one or more write-access cycles. A write-access cycle begins with the issuing of the ACTV command to select the desired bank and row of the SDRAM device. After the row has been opened, the EMIF proceeds to issue a WRT command while specifying the desired bank and column address. EMIF_A[10] is held low during the WRT command to avoid auto-precharging. The WRT command signals the SDRAM device to start writing a burst of data to the specified address while the EMIF issues NOP commands. The associated write data will be placed on the data bus in the cycle concurrent with the WRT command and with subsequent burst continuation NOP commands.

Figure 21-6 shows the signal waveforms for a basic SDRAM write operation in which a burst of data is read from a single page. When the EMIF SDRAM interface is configured to 16-bit by setting the NM bit of the SDRAM configuration register (SDCR) to 1, a burst size of eight is used. Figure 21-6 shows a burst size of eight.

Figure 21-6. Timing Waveform for Basic SDRAM Write Operation



The EMIF will truncate a series of bursting data if the remaining addresses of the burst are not part of the write request. The EMIF can truncate the burst in three ways:

- By issuing another WRT to the same page
- By issuing a PRE command in order to prepare for accessing a different page of the same bank
- By issuing a BT command in order to prepare for accessing a page in a different bank

Several other pins are also active during a write access. The EMIF_nDQM[1:0] pins are driven to select which bytes of the data word will be written to the SDRAM device. They are also used to mask out entire undesired data words during a burst access. The state of the other EMIF pins during each command can be found in Table 21-5.

The EMIF schedules its commands based on the timing information that is provided to it in the SDRAM timing register (SDTIMR). The values for the timing parameters in this register should be chosen to satisfy the timing requirements listed in the SDRAM datasheet. The EMIF uses this timing information to avoid violating any timing constraints related to issuing commands. This is commonly accomplished by inserting NOP commands during various cycles of an access. Refer to the register description of SDTIMR in Section 21.3.6 for more details on the various timing parameters.

21.2.5.11 Mapping from Logical Address to EMIF Pins

When the EMIF receives an SDRAM access request, it must convert the address of the access into the appropriate signals to send to the SDRAM device. The details of this address mapping are shown in Table 21-13 for 16-bit operation. Using the settings of the IBANK and PAGESIZE fields of the SDRAM configuration register (SDCR), the EMIF determines which bits of the logical address will be mapped to the SDRAM row, column, and bank addresses.

As the logical address is incremented by one halfword (16-bit operation), the column address is likewise incremented by one until a page boundary is reached. When the logical address increments across a page boundary, the EMIF moves into the same page in the next bank of the attached device by incrementing the bank address EMIF_BA and resetting the column address. The page in the previous bank is left open until it is necessary to close it. This method of traversal through the SDRAM banks helps maximize the number of open banks inside of the SDRAM and results in an efficient use of the device. There is no limitation on the number of banks that can be open at one time, but only one page within a bank can be open at a time.

The EMIF uses the EMIF_nDQM[1:0] pins during a WRT command to mask out selected bytes or entire words. The EMIF_nDQM[1:0] pins are always low during a READ command.

Table 21-13. Mapping from Logical Address to EMIF Pins for 16-bit SDRAM

<table>
<tr><th rowspan="2">IBANK</th><th rowspan="2">PAGESIZE</th><th colspan="14">Logical Address</th></tr>
<tr><th>31:27</th><th>26</th><th>25</th><th>24</th><th>23</th><th>22</th><th>21:14</th><th>13</th><th>12</th><th>11</th><th>10</th><th>9</th><th>8:1</th><th>0</th></tr>
<tr><td>0</td><td>0</td><td colspan="6">-</td><td colspan="6">Row Address</td><td>Col Address</td><td>EMIF_nDQM[0]</td></tr>
<tr><td>1</td><td>0</td><td colspan="5">-</td><td colspan="6">Row Address</td><td>EMIF_BA[0]</td><td>Col Address</td><td>EMIF_nDQM[0]</td></tr>
<tr><td>2</td><td>0</td><td colspan="4">-</td><td colspan="6">Row Address</td><td colspan="2">EMIF_BA[1:0]</td><td>Col Address</td><td>EMIF_nDQM[0]</td></tr>
<tr><td>0</td><td>1</td><td colspan="5">-</td><td colspan="6">Row Address</td><td colspan="2">Column Address</td><td>EMIF_nDQM[0]</td></tr>
<tr><td>1</td><td>1</td><td colspan="4">-</td><td colspan="6">Row Address</td><td>EMIF_BA[0]</td><td colspan="2">Column Address</td><td>EMIF_nDQM[0]</td></tr>
<tr><td>2</td><td>1</td><td colspan="3">-</td><td colspan="6">Row Address</td><td colspan="2">EMIF_BA[1:0]</td><td colspan="2">Column Address</td><td>EMIF_nDQM[0]</td></tr>
<tr><td>0</td><td>2</td><td colspan="4">-</td><td colspan="6">Row Address</td><td colspan="3">Column Address</td><td>EMIF_nDQM[0]</td></tr>
<tr><td>1</td><td>2</td><td colspan="3">-</td><td colspan="6">Row Address</td><td>EMIF_BA[0]</td><td colspan="3">Column Address</td><td>EMIF_nDQM[0]</td></tr>
<tr><td>2</td><td>2</td><td colspan="2">-</td><td colspan="6">Row Address</td><td colspan="2">EMIF_BA[1:0]</td><td colspan="3">Column Address</td><td>EMIF_nDQM[0]</td></tr>
<tr><td>0</td><td>3</td><td colspan="3">-</td><td colspan="6">Row Address</td><td colspan="4">Column Address</td><td>EMIF_nDQM[0]</td></tr>
<tr><td>1</td><td>3</td><td colspan="2">-</td><td colspan="6">Row Address</td><td>EMIF_BA[0]</td><td colspan="4">Column Address</td><td>EMIF_nDQM[0]</td></tr>
<tr><td>2</td><td>3</td><td>-</td><td colspan="6">Row Address</td><td colspan="2">EMIF_BA[1:0]</td><td colspan="4">Column Address</td><td>EMIF_nDQM[0]</td></tr>
</table>

NOTE: The upper bit of the Row Address is used only when addressing 256-Mbit and 512-Mbit SDRAM memories.

21.2.6 Asynchronous Controller and Interface

The EMIF easily interfaces to a variety of asynchronous devices including NOR Flash and SRAM. It can be operated in two major modes (see Table 21-14):

- Normal Mode
- Select Strobe Mode

Table 21-14. Normal Mode vs. Select Strobe Mode

Mode	Function of EMIF_nDQM pins	Operation of EMIF_nCS[4:2]
Normal Mode	Byte enables	Active during the entire asynchronous access cycle
Select Strobe Mode	Byte enables	Active only during the strobe period of an access cycle

The first mode of operation is Normal Mode, in which the EMIF_nDQM pins of the EMIF function as byte enables. In this mode, the EMIF_nCS[4:2] pins behaves as typical chip select signals, remaining active for the duration of the asynchronous access. See Section 21.2.6.1 for an example interface with multiple 8-bit devices.

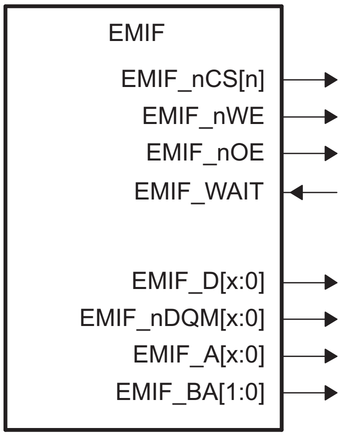
The second mode of operation is Select Strobe Mode, in which the EMIF_nCS[4:2] pins act as a strobe, active only during the strobe period of an access. In this mode, the EMIF_nDQM pins of the EMIF function as standard byte enables for reads and writes. A summary of the differences between the two modes of operation are shown in Table 21-14. Refer to Section 21.2.6.4 for the details of asynchronous operations in Normal Mode, and to Section 21.2.6.5 for the details of asynchronous operations in Select Strobe Mode. The EMIF hardware defaults to Normal Mode, but can be manually switched to Select Strobe Mode by setting the SS bit in the asynchronous *m* (m = 1, 2, 3, or 4) configuration register (CE*n*CFG) (*n* = 2, 3, or 4). Throughout the chapter, *m* can hold the values 1, 2, 3 or 4; and *n* can hold the values 2, 3, or 4.

The EMIF also provides configurable cycle timing parameters and an Extended Wait Mode that allows the connected device to extend the strobe period of an access cycle. The following sections describe the features related to interfacing with external asynchronous devices.

21.2.6.1 Interfacing to Asynchronous Memory

Figure 21-7 shows the EMIF's external pins used in interfacing with an asynchronous device. In EMIF_nCS[n], n = 2, 3, or 4.

Figure 21-7. EMIF Asynchronous Interface



Of special note is the connection between the EMIF and the external device's address bus. The EMIF address pin EMIF_A[0] always provides the least significant bit of a 32-bit word address. Therefore, when interfacing to a 16-bit or 8-bit asynchronous device, the EMIF_BA[1] and EMIF_BA[0] pins provide the least-significant bits of the halfword or byte address, respectively. Additionally, when the EMIF interfaces to a 16-bit asynchronous device, the EMIF_BA[0] pin can serve as the upper address line EMIF_A[22]. Figure 21-8 and Figure 21-9 show the mapping between the EMIF and the connected device's data and address pins for various programmed data bus widths. The data bus width may be configured in the asynchronous *n* configuration register (CE*n*CFG).

Figure 21-9 shows a common interface between the EMIF and external asynchronous memory. Figure 21-9 shows an interface between the EMIF and an external memory with byte enables. The EMIF should be operated in either Normal Mode or Select Strobe Mode when using this interface, so that the EMIF_nDQM signals operate as byte enables.

Figure 21-8. EMIF to 8-bit/16-bit Memory Interface

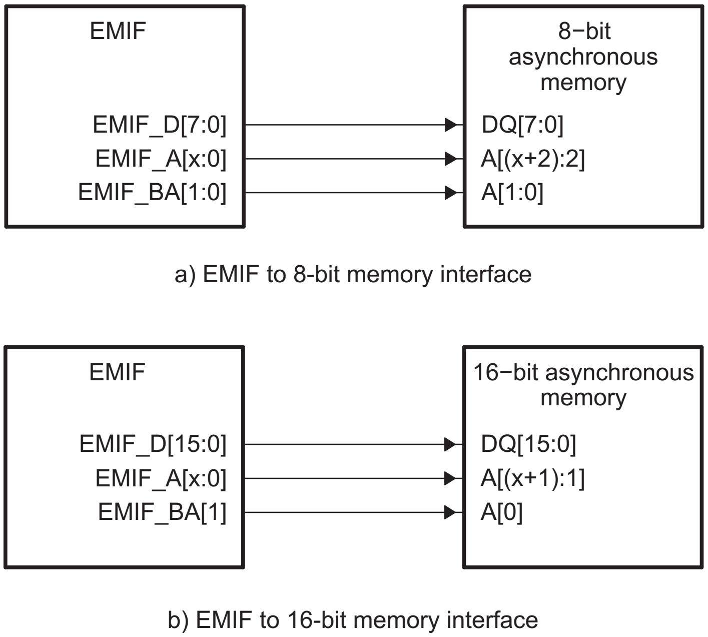
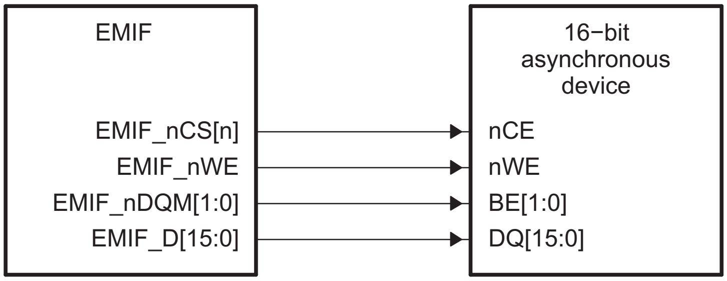


Figure 21-9. Common Asynchronous Interface



21.2.6.2 Accessing Larger Asynchronous Memories

The device has 22 dedicated EMIF address lines. If a device such as a large asynchronous flash needs to be attached to the EMIF, then GPIO pins may be used to control the flash device's upper address lines.

21.2.6.3 Configuring the EMIF for Asynchronous Accesses

The operation of the EMIF's asynchronous interface can be configured by programming the appropriate register fields. The reset value and bit position for each register field can be found in Section 21.3. The following tables list the register fields that can be programmed and describe the purpose of each field. These registers can be programmed prior to accessing the external memory, and the transfer following a write to these registers will use the new configuration.

Table 21-15. Description of the Asynchronous *m* Configuration Register (CE*n*CFG)

Parameter	Description
SS	**Select Strobe mode.** This bit selects the EMIF's mode of operation in the following way: • SS = 0 selects Normal Mode – EMIF_nDQM pins function as byte enables – EMIF_nCS[4:2] active for duration of access • SS = 1 selects Select Strobe Mode – EMIF_nDQM pins function as byte enables – EMIF_nCS[4:2] acts as a strobe.
EW	**Extended Wait Mode enable.** • EW = 0 disables Extended Wait Mode • EW = 1 enables Extended Wait Mode When set to 1, the EMIF enables its Extended Wait Mode in which the strobe width of an access cycle can be extended in response to the assertion of the EMIF_nWAIT pin. The WP*n* bit in the asynchronous wait cycle configuration register (AWCC) controls to polarity of EMIF_nWAIT pin. See Section 21.2.6.6 for more details on this mode of operation.
W_SETUP/R_SETUP	**Read/Write setup widths.** These fields define the number of EMIF clock cycles of setup time for the address pins (EMIF_A and EMIF_BA), byte enables (EMIF_nDQM), and asynchronous chip enable (EMIF_nCS[4:2]) before the read strobe pin (EMIF_nOE) or write strobe pin (EMIF_nWE) falls, minus one cycle. For writes, the W_SETUP field also defines the setup time for the data pins (EMIF_D). Refer to the datasheet of the external asynchronous device to determine the appropriate setting for this field.
W_STROBE/R_STROBE	**Read/Write strobe widths.** These fields define the number of EMIF clock cycles between the falling and rising of the read strobe pin (EMIF_nOE) or write strobe pin (EMIF_nWE), minus one cycle. If Extended Wait Mode is enabled by setting the EW field in the asynchronous *n* configuration register (CE*n*CFG), these fields must be set to a value greater than zero. Refer to the datasheet of the external asynchronous device to determine the appropriate setting for this field.
W_HOLD/R_HOLD	**Read/Write hold widths.** These fields define the number of EMIF clock cycles of hold time for the address pins (EMIF_A and EMIF_BA), byte enables (EMIF_nDQM), and asynchronous chip enable (EMIF_nCS[4:2]) after the read strobe pin (EMIF_nOE) or write strobe pin (EMIF_nWE) rises, minus one cycle. For writes, the W_HOLD field also defines the hold time for the data pins (EMIF_D). Refer to the datasheet of the external asynchronous device to determine the appropriate setting for this field.
TA	**Minimum turnaround time.** This field defines the minimum number of EMIF clock cycles between asynchronous reads and writes, minus one cycle. The purpose of this feature is to avoid contention on the bus. The value written to this field also determines the number of cycles that will be inserted between asynchronous accesses and SDRAM accesses. Refer to the datasheet of the external asynchronous device to determine the appropriate setting for this field.

Table 21-15. Description of the Asynchronous *m* Configuration Register (CE*n*CFG) (continued)

Parameter	Description
ASIZE	**Asynchronous Device Bus Width.** This field determines the data bus width of the asynchronous interface in the following way: • ASIZE = 0 selects an 8-bit bus • ASIZE = 1 selects a 16-bit bus The configuration of ASIZE determines the function of the EMIF_A and EMIF_BA pins as described in Section 21.2.6.1. This field also determines the number of external accesses required to fulfill a request generated by one of the sources mentioned in Section 21.2.2. For example, a request for a 32-bit word would require four external access when ASIZE = 0. Refer to the datasheet of the external asynchronous device to determine the appropriate setting for this field.

Table 21-16. Description of the Asynchronous Wait Cycle Configuration Register (AWCC)

Parameter	Description
WP*n*	**EM_WAIT Polarity.** • WP*n* = 0 selects active-low polarity • WP*n* = 1 selects active-high polarity When set to 1, the EMIF will wait if the EMIF_nWAIT pin is high. When cleared to 0, the EMIF will wait if the EMIF_nWAIT pin is low. The EMIF must have the Extended Wait Mode enabled for the EMIF_nWAIT pin to affect the width of the strobe period.
MAX_EXT_WAIT	**Maximum Extended Wait Cycles.** This field configures the number of EMIF clock cycles the EMIF will wait for the EMIF_nWAIT pin to be deactivated during the strobe period of an access cycle. The maximum number of EMIF clock cycles it will wait is determined by the following formula: Maximum Extended Wait Cycles = (MAX_EXT_WAIT + 1) × 16 If the EMIF_nWAIT pin is not deactivated within the time specified by this field, the EMIF resumes the access cycle, registering whatever data is on the bus and proceeding to the hold period of the access cycle. This situation is referred to as an Asynchronous Timeout. An Asynchronous Timeout generates an interrupt, if it has been enabled in the EMIF interrupt mask set register (INTMSKSET). Refer to Section 21.2.9.1 for more information about the EMIF interrupts.

Table 21-17. Description of the EMIF Interrupt Mask Set Register (INTMSKSET)

Parameter	Description
WR_MASK_SET	**Wait Rise Mask Set.** Writing a 1 enables an interrupt to be generated when a rising edge on EMIF_nWAIT occurs
AT_MASK_SET	**Asynchronous Timeout Mask Set.** Writing a 1 to this bit enables an interrupt to be generated when an Asynchronous Timeout occurs.

Table 21-18. Description of the EMIF Interrupt Mast Clear Register (INTMSKCLR)

Parameter	Description
WR_MASK_CLR	**Wait Rise Mask Clear.** Writing a 1 to this bit disables the interrupt, clearing the WR_MASK_SET bit in the EMIF interrupt mask set register (INTMSKSET).
AT_MASK_CLR	**Asynchronous Timeout Mask Clear.** Writing a 1 to this bit prevents an interrupt from being generated when an Asynchronous Timeout occurs.

21.2.6.4 Read and Write Operations in Normal Mode

Normal Mode is the asynchronous interface's default mode of operation. It is selected when the SS bit in the asynchronous *n* configuration register (CE*n*CFG) is cleared to 0. In this mode, the EMIF_nDQM pins operate as byte enables. Section 21.2.6.4.1 and Section 21.2.6.4.2 explain the details of read and write operations while in Normal Mode.

21.2.6.4.1 Asynchronous Read Operations (Normal Mode)

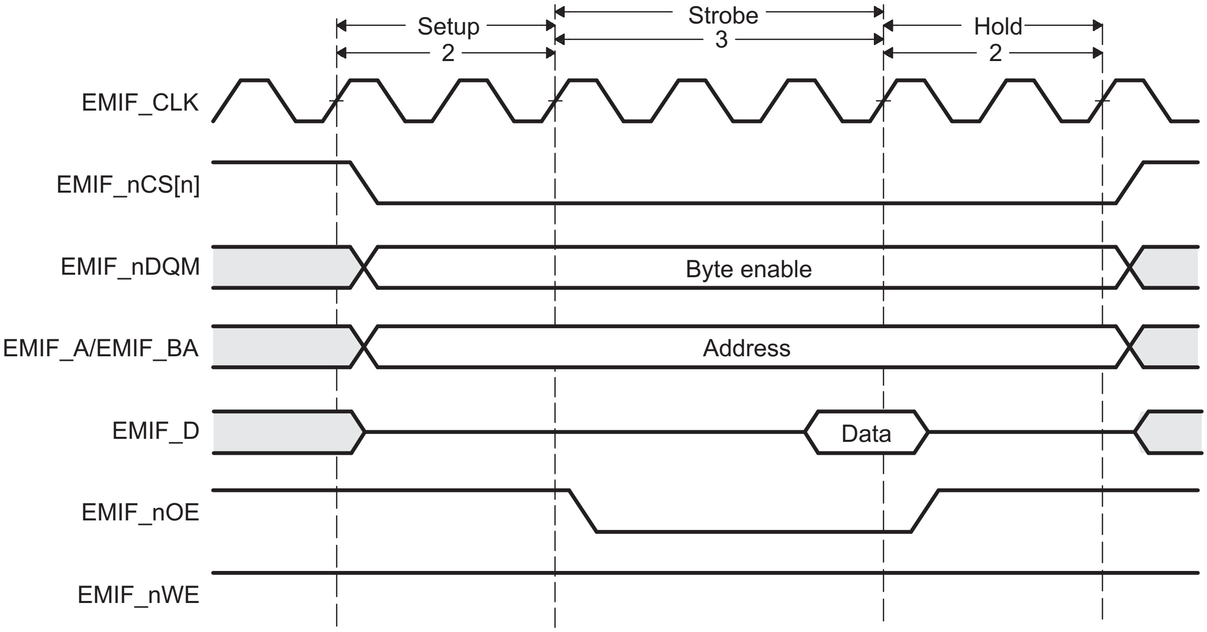
NOTE: During an entire asynchronous read operation, the EMIF_nWE pin is driven high.

An asynchronous read is performed when any of the requesters mentioned in Section 21.2.2 request a read from the attached asynchronous memory. After the request is received, a read operation is initiated once it becomes the EMIF's highest priority task, according to the priority scheme detailed in Section 21.2.13. In the event that the read request cannot be serviced by a single access cycle to the external device, multiple access cycles will be performed by the EMIF until the entire request is fulfilled. The details of an asynchronous read operation in Normal Mode are described in Table 21-19. Also, Figure 21-10 shows an example timing diagram of a basic read operation.

Table 21-19. Asynchronous Read Operation in Normal Mode

Time Interval	Pin Activity in Normal Mode
Turn-around period	Once the read operation becomes the highest priority task for the EMIF, the EMIF waits for the programmed number of turn-around cycles before proceeding to the setup period of the operation. The number of wait cycles is taken directly from the TA field of the asynchronous *n* configuration register (CE*n*CFG). There are two exceptions to this rule: • If the current read operation was directly proceeded by another read operation, no turnaround cycles are inserted. • If the current read operation was directly proceeded by a write operation and the TA field has been cleared to 0, one turn-around cycle will be inserted. After the EMIF has waited for the turnaround cycles to complete, it again checks to make sure that the read operation is still its highest priority task. If so, the EMIF proceeds to the setup period of the operation. If it is no longer the highest priority task, the EMIF terminates the operation.
Start of the setup period	The following actions occur at the start of the setup period: • The setup, strobe, and hold values are set according to the R_SETUP, R_STROBE, and R_HOLD values in CE*n*CFG. • The address pins EMIF_A and EMIF_BA become valid and carry the values described in Section 21.2.6.1. • EMIF_nCS[4:2] falls to enable the external device (if not already low from a previous operation)
Strobe period	The following actions occur during the strobe period of a read operation: 1. EMIF_nOE falls at the start of the strobe period 2. On the rising edge of the clock which is concurrent with the end of the strobe period: • EMIF_nOE rises • The data on the EMIF_D bus is sampled by the EMIF. In Figure 21-10, EMIF_nWAIT is inactive. If EMIF_nWAIT is instead activated, the strobe period can be extended by the external device to give it more time to provide the data. Section 21.2.6.6 contains more details on using the EMIF_nWAIT pin.
End of the hold period	At the end of the hold period: • The address pins EMIF_A and EMIF_BA become invalid • EMIF_nCS[4:2] rises (if no more operations are required to complete the current request) EMIF may be required to issue additional read operations to a device with a small data bus width in order to complete an entire word access. In this case, the EMIF immediately re-enters the setup period to begin another operation without incurring the turn-round cycle delay. The setup, strobe, and hold values are not updated in this case. If the entire word access has been completed, the EMIF returns to its previous state unless another asynchronous request has been submitted and is currently the highest priority task. If this is the case, the EMIF instead enters directly into the turnaround period for the pending read or write operation.

Figure 21-10. Timing Waveform of an Asynchronous Read Cycle in Normal Mode



21.2.6.4.2 Asynchronous Write Operations (Normal Mode)

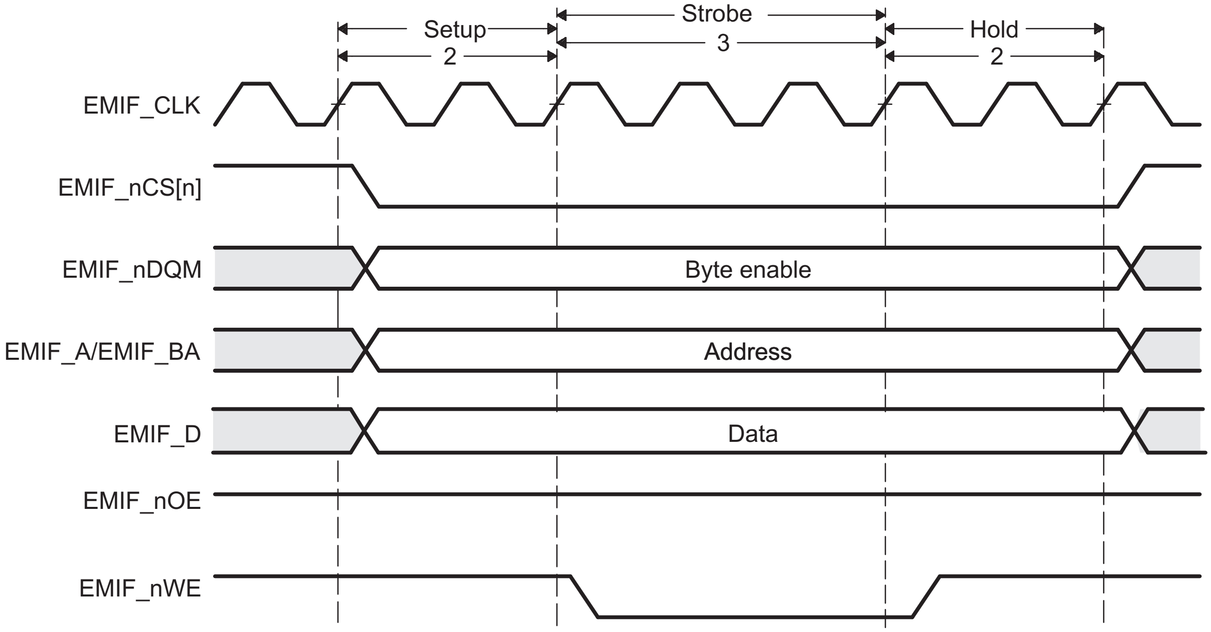
NOTE: During an entire asynchronous write operation, the EMIF_nOE pin is driven high.

An asynchronous write is performed when any of the requesters mentioned in Section 21.2.2 request a write to memory in the asynchronous bank of the EMIF. After the request is received, a write operation is initiated once it becomes the EMIF's highest priority task, according to the priority scheme detailed in Section 21.2.13. In the event that the write request cannot be serviced by a single access cycle to the external device, multiple access cycles will be performed by the EMIF until the entire request is fulfilled. The details of an asynchronous write operation in Normal Mode are described in Table 21-20. Also, Figure 21-11 shows an example timing diagram of a basic write operation.

Table 21-20. Asynchronous Write Operation in Normal Mode

Time Interval	Pin Activity in Normal Mode
Turnaround period	Once the write operation becomes the highest priority task for the EMIF, the EMIF waits for the programmed number of turn-around cycles before proceeding to the setup period of the operation. The number of wait cycles is taken directly from the TA field of the asynchronous *n* configuration register (CE*n*CFG). There are two exceptions to this rule: • If the current write operation was directly proceeded by another write operation, no turn-around cycles are inserted. • If the current write operation was directly proceeded by a read operation and the TA field has been cleared to 0, one turnaround cycle will be inserted. After the EMIF has waited for the turn-around cycles to complete, it again checks to make sure that the write operation is still its highest priority task. If so, the EMIF proceeds to the setup period of the operation. If it is no longer the highest priority task, the EMIF terminates the operation.
Start of the setup period	The following actions occur at the start of the setup period: • The setup, strobe, and hold values are set according to the W_SETUP, W_STROBE, and W_HOLD values in CE*n*CFG. • The address pins EMIF_A and EMIF_BA and the data pins EMIF_D become valid. The EMIF_A and EMIF_BA pins carry the values described in Section 21.2.6.1. • EMIF_nCS[4:2] falls to enable the external device (if not already low from a previous operation).
Strobe period	The following actions occur at the start of the strobe period of a write operation: 1. EMIF_nWE falls 2. The EMIF_nDQM pins become valid as byte enables. The following actions occur on the rising edge of the clock which is concurrent with the end of the strobe period: 1. EMIF_nWE rises 2. The EMIF_nDQM pins deactivate In Figure 21-11, EMIF_nWAIT is inactive. If EMIF_nWAIT is instead activated, the strobe period can be extended by the external device to give it more time to accept the data. Section 21.2.6.6 contains more details on using the EMIF_nWAIT pin.
End of the hold period	At the end of the hold period: • The address pins EMIF_A and EMIF_BA become invalid • The data pins become invalid • EMIF_nCS[n] (n = 2, 3, or 4) rises (if no more operations are required to complete the current request) The EMIF may be required to issue additional write operations to a device with a small data bus width in order to complete an entire word access. In this case, the EMIF immediately re-enters the setup period to begin another operation without incurring the turnaround cycle delay. The setup, strobe, and hold values are not updated in this case. If the entire word access has been completed, the EMIF returns to its previous state unless another asynchronous request has been submitted and is currently the highest priority task. If this is the case, the EMIF instead enters directly into the turnaround period for the pending read or write operation.

Figure 21-11. Timing Waveform of an Asynchronous Write Cycle in Normal Mode



21.2.6.5 Read and Write Operation in Select Strobe Mode

Select Strobe Mode is the EMIF's second mode of operation. It is selected when the SS bit of the asynchronous *n* configuration register (CE*n*CFG) is set to 1. In this mode, the EMIF_nDQM pins operate as byte enables and the EMIF_nCS[n] (n = 2, 3, or 4) pin is only active during the strobe period of an access cycle. Section 21.2.6.4.1 and Section 21.2.6.4.2 explain the details of read and write operations while in Select Strobe Mode.

21.2.6.5.1 Asynchronous Read Operations (Select Strobe Mode)

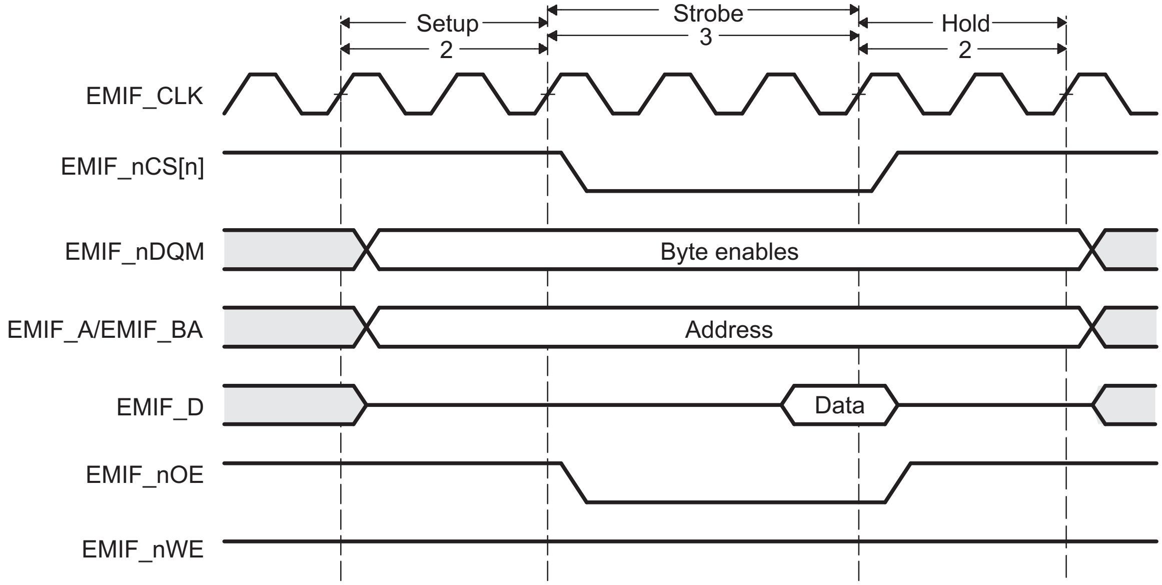
NOTE: During the entirety of an asynchronous read operation, the EMIF_nWE pin is driven high.

An asynchronous read is performed when any of the requesters mentioned in Section 21.2.2 request a read from the attached asynchronous memory. After the request is received, a read operation is initiated once it becomes the EMIF's highest priority task, according to the priority scheme detailed in Section 21.2.13. In the event that the read request cannot be serviced by a single access cycle to the external device, multiple access cycles will be performed by the EMIF until the entire request is fulfilled. The details of an asynchronous read operation in Select Strobe Mode are described in Table 21-21. Also, Figure 21-12 shows an example timing diagram of a basic read operation.

Table 21-21. Asynchronous Read Operation in Select Strobe Mode

Time Interval	Pin Activity in Select Strobe Mode
Turnaround period	Once the read operation becomes the highest priority task for the EMIF, the EMIF waits for the programmed number of turn-around cycles before proceeding to the setup period of the operation. The number of wait cycles is taken directly from the TA field of the asynchronous *n* configuration register (CE*n*CFG). There are two exceptions to this rule: • If the current read operation was directly proceeded by another read operation, no turn-around cycles are inserted. • If the current read operation was directly proceeded by a write operation and the TA field has been cleared to 0, one turn-around cycle will be inserted. After the EMIF has waited for the turn-around cycles to complete, it again checks to make sure that the read operation is still its highest priority task. If so, the EMIF proceeds to the setup period of the operation. If it is no longer the highest priority task, the EMIF terminates the operation.
Start of the setup period	The following actions occur at the start of the setup period: • The setup, strobe, and hold values are set according to the R_SETUP, R_STROBE, and R_HOLD values in CE*n*CFG. • The address pins EMIF_A and EMIF_BA become valid and carry the values described in Section 21.2.6.1. • The EMIF_nDQM pins become valid as byte enables.
Strobe period	The following actions occur during the strobe period of a read operation: 1. EMIF_nCS[n] (n = 2, 3, or 4) and EMIF_nOE fall at the start of the strobe period 2. On the rising edge of the clock which is concurrent with the end of the strobe period: • EMIF_nCS[n] (n = 2, 3, or 4) and EMIF_nOE rise • The data on the EMIF_D bus is sampled by the EMIF. In Figure 21-12, EMIF_nWAIT is inactive. If EMIF_nWAIT is instead activated, the strobe period can be extended by the external device to give it more time to provide the data. Section 21.2.6.6 contains more details on using the EMIF_nWAIT pin.
End of the hold period	At the end of the hold period: • The address pins EMIF_A and EMIF_BA become invalid • The EMIF_nDQM pins become invalid The EMIF may be required to issue additional read operations to a device with a small data bus width in order to complete an entire word access. In this case, the EMIF immediately re-enters the setup period to begin another operation without incurring the turnaround cycle delay. The setup, strobe, and hold values are not updated in this case. If the entire word access has been completed, the EMIF returns to its previous state unless another asynchronous request has been submitted and is currently the highest priority task. If this is the case, the EMIF instead enters directly into the turnaround period for the pending read or write operation.

Figure 21-12. Timing Waveform of an Asynchronous Read Cycle in Select Strobe Mode



21.2.6.5.2 Asynchronous Write Operations (Select Strobe Mode)

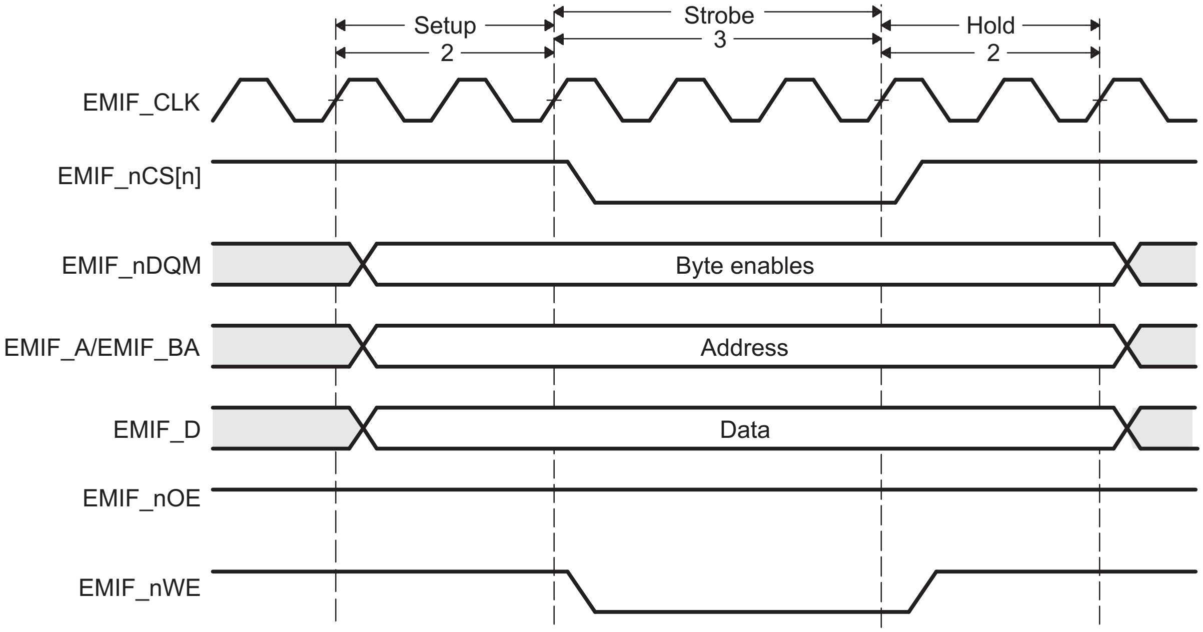
NOTE: During the entirety of an asynchronous write operation, the EMIF_nOE pin is driven high.

An asynchronous write is performed when any of the requesters mentioned in Section 21.2.2 request a write to memory in the asynchronous bank of the EMIF. After the request is received, a write operation is initiated once it becomes the EMIF's highest priority task, according to the priority scheme detailed in Section 21.2.13. In the event that the write request cannot be serviced by a single access cycle to the external device, multiple access cycles will be performed by the EMIF until the entire request is fulfilled. The details of an asynchronous write operation in Select Strobe Mode are described in Table 21-22. Also, Figure 21-13 shows an example timing diagram of a basic write operation.

Table 21-22. Asynchronous Write Operation in Select Strobe Mode

Time Interval	Pin Activity in Select Strobe Mode
Turnaround period	Once the write operation becomes the highest priority task for the EMIF, the EMIF waits for the programmed number of turnaround cycles before proceeding to the setup period of the operation. The number of wait cycles is taken directly from the TA field of the asynchronous *n* configuration register (CE*n*CFG). There are two exceptions to this rule: • If the current write operation was directly proceeded by another write operation, no turn-around cycles are inserted. • If the current write operation was directly proceeded by a read operation and the TA field has been cleared to 0, one turnaround cycle will be inserted. After the EMIF has waited for the turnaround cycles to complete, it again checks to make sure that the write operation is still its highest priority task. If so, the EMIF proceeds to the setup period of the operation. If it is no longer the highest priority task, the EMIF terminates the operation.
Start of the setup period	The following actions occur at the start of the setup period: • The setup, strobe, and hold values are set according to the W_SETUP, W_STROBE, and W_HOLD values in CE*n*CFG. • The address pins EMIF_A and EMIF_BA and the data pins EMIF_D become valid. The EMIF_A and EMIF_BA pins carry the values described in Section 21.2.6.1. • The EMIF_nDQM pins become active as byte enables.
Strobe period	The following actions occur at the start of the strobe period of a write operation: • EMIF_nCS[n] (n = 2, 3, or 4) and EMIF_nWE fall The following actions occur on the rising edge of the clock which is concurrent with the end of the strobe period: • EMIF_nCS[n] (n = 2, 3, or 4) and EMIF_nWE rise In Figure 21-13, EMIF_nWAIT is inactive. If EMIF_nWAIT is instead activated, the strobe period can be extended by the external device to give it more time to accept the data. Section 21.2.6.6 contains more details on using the EMIF_nWAIT pin.
End of the hold period	At the end of the hold period: • The address pins EMIF_A and EMIF_BA become invalid • The data pins become invalid • The EMIF_nDQM pins become invalid The EMIF may be required to issue additional write operations to a device with a small data bus width in order to complete an entire word access. In this case, the EMIF immediately re-enters the setup period to begin another operation without incurring the turnaround cycle delay. The setup, strobe, and hold values are not updated in this case. If the entire word access has been completed, the EMIF returns to its previous state unless another asynchronous request has been submitted and is currently the highest priority task. If this is the case, the EMIF instead enters directly into the turn-around period for the pending read or write operation.

Figure 21-13. Timing Waveform of an Asynchronous Write Cycle in Select Strobe Mode



21.2.6.6 Extended Wait Mode and the EMIF_nWAIT Pin

The EMIF supports the Extend Wait Mode. This is a mode in which the external asynchronous device may assert control over the length of the strobe period. The Extended Wait Mode can be entered by setting the EW bit in the asynchronous *n* configuration register (CE*n*CFG) (*n* = 2, 3, or 4). When this bit is set, the EMIF monitors the EMIF_nWAIT pin to determine if the attached device wishes to extend the strobe period of the current access cycle beyond the programmed number of clock cycles.

When the EMIF detects that the EMIF_nWAIT pin has been asserted, it will begin inserting extra strobe cycles into the operation until the EMIF_nWAIT pin is deactivated by the external device. The EMIF will then return to the last cycle of the programmed strobe period and the operation will proceed as usual from this point. Please refer to the device data manual for details on the timing requirements of the EMIF_nWAIT signal.

The EMIF_nWAIT pin cannot be used to extend the strobe period indefinitely. The programmable MAX_EXT_WAIT field in the asynchronous wait cycle configuration register (AWCC) determines the maximum number of EMIF_CLK cycles the strobe period may be extended beyond the programmed length. When the counter expires, the EMIF proceeds to the hold period of the operation regardless of the state of the EMIF_nWAIT pin. The EMIF can also generate an interrupt upon expiration of this counter. See Section 21.2.9.1 for details on enabling this interrupt.

For the EMIF to function properly in the Extended Wait mode, the WP*n* bit of AWCC must be programmed to match the polarity of the EMIF_nWAIT pin. In its reset state of 1, the EMIF will insert wait cycles when the EMIF_nWAIT pin is sampled high. When set to 0, the EMIF will insert wait cycles only when EMIF_nWAIT is sampled low. This programmability allows for a glueless connection to larger variety of asynchronous devices.

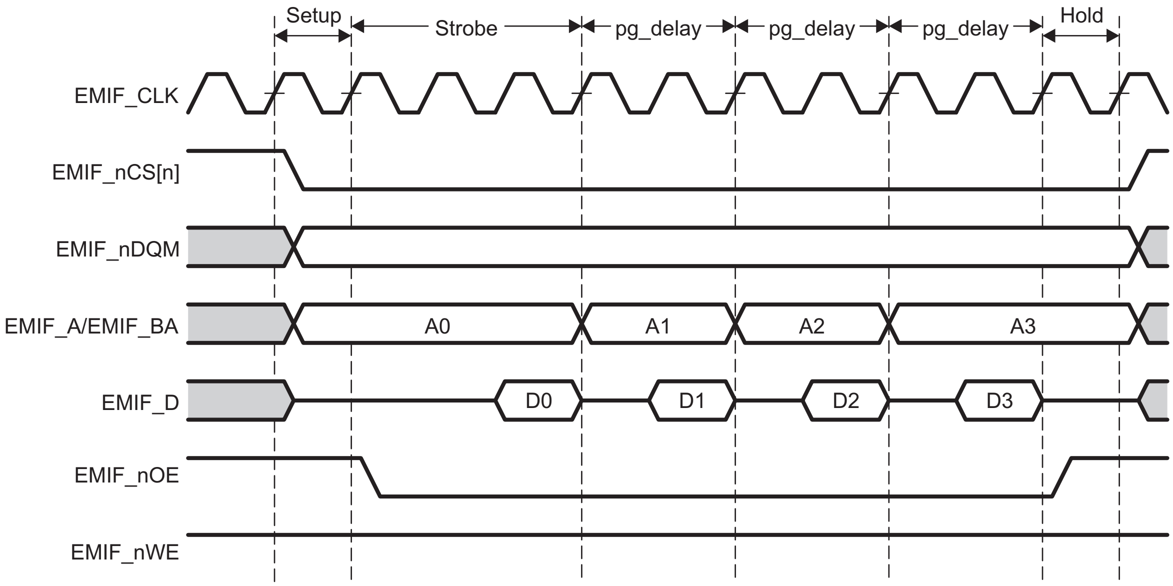
Finally, a restriction is placed on the strobe period timing parameters when operating in Extended Wait mode. Specifically, the sum of the W_SETUP and W_STROBE fields must be greater than 4, and the sum of the R_SETUP and R_STROBE fields must be greater than 4 for the EMIF to recognize the EMIF_nWAIT pin has been asserted. The W_SETUP, W_STROBE, R_SETUP, and R_STROBE fields are in CE*n*CFG.

21.2.6.7 NOR Flash Page Mode

EMIF supports Page mode reads for NOR Flash on its asynchronous memory chip selects. This mode can be enabled by writing a 1 to the CS*n*_PG_MD_EN (*n* = 2, 3, or 4) field in the Page Mode Control register for the chip select in consideration. Whenever Page Mode for reads is enabled for a particular chip select, the page size for the device connected must also be programmed in the CS*n*_PG_SIZE field of the Page Mode Control register. The address change to valid read data available timing must be programmed in the CS*n*_PG_DEL field of the Page Control register. All other asynchronous memory timings must be programmed in the asynchronous configuration register (CE*n*CFG). See Figure 21-14 for read in asynchronous page mode.

NOTE: The Extended Wait mode and the Select Strobe mode must be disabled when using the asynchronous interface in Page mode.

Figure 21-14. Asynchronous Read in Page Mode



21.2.7 Data Bus Parking

The EMIF always drives the data bus to the previous write data value when it is idle. This feature is called data bus parking. Only when the EMIF issues a read command to the external memory does it stop driving the data bus. After the EMIF latches the last read data, it immediately parks the data bus again.

The one exception to this behavior occurs after performing an asynchronous read operation while the EMIF is in the self-refresh state. In this situation, the read operation is not followed by the EMIF parking the data bus. Instead, the EMIF tri-states the data bus. Therefore, it is not recommended to perform asynchronous read operations while the EMIF is in the self-refresh state, in order to prevent floating inputs on the data bus. External pull-ups, such as 10kΩ resistors, should be placed on the 16 EMIF data bus pins (which do not have internal pull-ups) if it is required to perform reads in this situation. The precise resistor value should be chosen so that the worst case combined off-state leakage currents do not cause the voltage levels on the associated pins to drop below the high-level input voltage requirement.

For information about the self-refresh state, see Section 21.2.5.7.

21.2.8 Reset and Initialization Considerations

The EMIF memory controller has two active-low reset signals, CHIP_RST_n and MOD_G_RST_n. Both these reset signals are driven by the device system reset signal. This device does not offer the flexibility to reset just the EMIF state machine without also resetting the EMIF controller's memory-mapped registers. As soon as the device system reset is released (driven High), the EMIF memory controller immediately begins its initialization sequence. Command and data stored in the EMIF memory controller FIFOs are lost. Refer the Architecture chapter of the tecnical reference manual (TRM) for more information on conditions that can cause a device system reset to be asserted.

When system reset is released, the EMIF automatically begins running the SDRAM initialization sequence described in Section 21.2.5.4. Even though the initialization procedure is automatic, a special procedure, found in Section 21.2.5.5 must still be followed.

21.2.9 Interrupt Support

The EMIF supports a single interrupt to the CPU. Section 21.2.9.1 details the generation and internal masking of EMIF interrupts.

21.2.9.1 Interrupt Events

There are three conditions that may cause the EMIF to generate an interrupt to the CPU. These conditions are:

- A rising edge on the EMIF_nWAIT signal (wait rise interrupt)
- An asynchronous time out
- Usage of unsupported addressing mode (line trap interrupt)

The wait rise interrupt occurs when a rising edge is detected on EMIF_nWAIT signal. This interrupt generation is not affected by the WP*n* bit in the asynchronous wait cycle configuration register (AWCC). The asynchronous time out interrupt condition occurs when the attached asynchronous device fails to deassert the EMIF_nWAIT pin within the number of cycles defined by the MAX_EXT_WAIT bit in AWCC (this happens only in extended wait mode). EMIF supports only linear incrementing and cache line wrap addressing modes . If an access request for an unsupported addressing mode is received, the EMIF will set the LT bit in the EMIF interrupt raw register (INTRAW) and treat the request as a linear incrementing request.

Only when the interrupt is enabled by setting the appropriate bit (WR_MASK_SET/AT_MASK_SET/LT_MASK_SET) in the EMIF interrupt mask set register (INTMSKSET) to 1, will the interrupt be sent to the CPU. Once enabled, the interrupt may be disabled by writing a 1 to the corresponding bit in the EMIF interrupt mask clear register (INTMSKCLR). The bit fields in both the INTMSKSET and INTMSKCLR may be used to indicate whether the interrupt is enabled. When the interrupt is enabled, the corresponding bit field in both the INTMSKSET and INTMSKCLR will have a value of 1; when the interrupt is disabled, the corresponding bit field will have a value of 0.

The EMIF interrupt raw register (INTRAW) and the EMIF interrupt mask register (INTMSK) indicate the status of each interrupt. The appropriate bit (WR/AT/LT) in INTRAW is set when the interrupt condition occurs, whether or not the interrupt has been enabled. However, the appropriate bit (WR_MASKED/AT_MASKED/LT_MASKED) in INTMSK is set only when the interrupt condition occurs and the interrupt is enabled. Writing a 1 to the bit in INTRAW clears the INTRAW bit as well as the corresponding bit in INTMSK. Table 21-23 contains a brief summary of the interrupt status and control bit fields. See Section 21.3 for complete details on the register fields.

Table 21-23. Interrupt Monitor and Control Bit Fields

Register Name	Bit Name	Description
EMIF interrupt raw register (INTRAW)	WR	This bit is set when an rising edge on the EMIF_nWAIT signal occurs. Writing a 1 clears the WR bit as well as the WR_MASKED bit in INTMSK.
	AT	This bit is set when an asynchronous timeout occurs. Writing a 1 clears the AT bit as well as the AT_MASKED bit in INTMSK.
	LT	This bit is set when an unsupported addressing mode is used. Writing a 1 clears LT bit as well as the LT_MASKED bit in INTMSK.
EMIF interrupt mask register (INTMSK)	WR_MASKED	This bit is set only when a rising edge on the EMIF_nWAIT signal occurs and the interrupt has been enabled by writing a 1 to the WR_MASK_SET bit in INTMSKSET.
	AT_MASKED	This bit is set only when an asynchronous timeout occurs and the interrupt has been enabled by writing a 1 to the AT_MASK_SET bit in INTMSKSET.
	LT_MASKED	This bit is set only when line trap interrupt occurs and the interrupt has been enabled by writing a 1 to the LT_MASK_SET bit in INTMSKSET.
EMIF interrupt mask set register (INTMSKSET)	WR_MASK_SET	Writing a 1 to this bit enables the wait rise interrupt.
	AT_MASK_SET	Writing a 1 to this bit enables the asynchronous timeout interrupt.
	LT_MASK_SET	Writing a 1 to this bit enables the line trap interrupt.
EMIF interrupt mask clear register (INTMSKCLR)	WR_MASK_CLR	Writing a 1 to this bit disables the wait rise interrupt.
	AT_MASK_CLR	Writing a 1 to this bit disables the asynchronous timeout interrupt.
	LT_MASK_CLR	Writing a 1 to this bit disables the line trap interrupt.

21.2.10 DMA Event Support

EMIF memory controller is a DMA slave peripheral and therefore does not generate DMA events. Data read and write requests may be made directly, by masters and the DMA.

21.2.11 EMIF Signal Multiplexing

For details on EMIF signal multiplexing, see the I/O Multiplexing Module chapter of the technical reference manual.

21.2.12 Memory Map

For information describing the device memory-map, see your device-specific datasheet.

21.2.13 Priority and Arbitration

Section 21.2.2 describes the external prioritization and arbitration among requests from different sources within the microcontroller. The result of this external arbitration is that only one request is presented to the EMIF at a time. Once the EMIF completes a request, the external arbiter then provides the EMIF with the next pending request.

Internally, the EMIF undertakes memory device transactions according to a strict priority scheme. The highest priority events are:

- A device reset.
- A write to any of the three least significant bytes of the SDRAM configuration register (SDCR).

Either of these events will cause the EMIF to immediately commence its initialization sequence as described in Section 21.2.5.4.

Once the EMIF has completed its initialization sequence, it performs memory transactions according to the following priority scheme (highest priority listed first):

1. If the EMIF's backlog refresh counter is at the Refresh Must urgency level, the EMIF performs multiple SDRAM auto refresh cycles until the Refresh Release urgency level is reached.
2. If an SDRAM or asynchronous read has been requested, the EMIF performs a read operation.
3. If the EMIF's backlog refresh counter is at the Refresh Need urgency level, the EMIF performs an SDRAM auto refresh cycle.
4. If an SDRAM or asynchronous write has been requested, the EMIF performs a write operation.
5. If the EMIF's backlog refresh counter is at the Refresh May or Refresh Release urgency level, the EMIF performs an SDRAM auto refresh cycle.
6. If the value of the SR bit in SDCR has been set to 1, the EMIF will enter the self-refresh state as described in Section 21.2.5.7.

After taking one of the actions listed above, the EMIF then returns to the top of the priority list to determine its next action.

Because the EMIF does not issue auto-refresh cycles when in the self-refresh state, the above priority scheme does not apply when in this state. See Section 21.2.5.7 for details on the operation of the EMIF when in the self-refresh state.

21.2.14 System Considerations

This section describes various system considerations to keep in mind when operating the EMIF.

21.2.14.1 Asynchronous Request Times

In a system that interfaces to both SDRAM and asynchronous memory, the asynchronous requests must not take longer than the smaller of the following two values:

- t_{RAS} (typically 120 μs) - to avoid violating the maximum time allowed between issuing an ACTV and PRE command to the SDRAM.
- $t_{Refresh\ Rate}$ × 11 (typically 15.7 μs × 11 = 172.7 μs) - to avoid refresh violations on the SDRAM.

  The length of an asynchronous request is controlled by multiple factors, the primary factor being the number of access cycles required to complete the request. For example, an asynchronous request for 4 bytes will require four access cycles using an 8-bit data bus and only two access cycle using a 16-bit data bus. The maximum request size that the EMIF can be sent is 16 words, therefore the maximum number of access cycles per memory request is 64 when the EMIF is configured with an 8-bit data bus. The length of the individual access cycles that make up the asynchronous request is determined by the programmed setup, strobe, hold, and turnaround values, but can also be extended with the assertion of the EMIF_nWAIT input signal up to a programmed maximum limit. It is up to the user to make sure that an entire asynchronous request does not exceed the timing values listed above when also interfacing to an SDRAM device. This can be done by limiting the asynchronous timing parameters.

21.2.14.2 Interface to External Peripheral or FIFO Memory

If EMIF is used to interface to an external peripheral or FIFO logic (for example, UHPI), it is recommended to use the host CPU's Memory Protection Unit (MPU) to define this external memory range as a region that is either strongly-ordered or of device type.

21.2.14.3 Interface to External SDRAM

If EMIF is used to interface to an external SDRAM, it is recommended to burst as much as possible to normal memory to improve the interface bandwidth.

21.2.15 Power Management

Power dissipation from the EMIF memory controller may be managed by following methods:

- Self-refresh mode
- Power-down mode
- Gating input clocks to the module off

Gating input clocks off to the EMIF memory controller achieves higher power savings when compared to the power savings of self-refresh or power down mode. The input clock VCLK3 can be turned off through the use of the Global Clock Module (GCM). Before gating clocks off, the EMIF memory controller must place the SDR SDRAM memory in self-refresh mode. If the external memory requires a continuous clock, the VCLK3 clock domain must not be turned off because this may result in data corruption. See the following subsections for the proper procedures to follow when stopping the EMIF memory controller clocks.

21.2.15.1 Power Management Using Self-Refresh Mode

The EMIF can be placed into a self-refresh state in order to place the attached SDRAM devices into self-refresh mode, which consumes less power for most SDRAM devices. In this state, the attached SDRAM device uses an internal clock to perform its own auto refresh cycles. This maintains the validity of the data in the SDRAM without the need for any external commands. Refer to Section 21.2.5.7 for more details on placing the EMIF into the self-refresh state.

21.2.15.2 Power Management Using Power Down Mode

In the power down mode, EMIF drives EMIF_CKE low to lower the power consumption. EMIF_CKE goes high when there is a need to send refresh (REFR) commands, after which EMIF_CKE is again driven low. EMIF_CKE remains low until any request arrives. Refer to Section 21.2.5.8 for more details on placing EMIF in power down mode.

21.2.16 Emulation Considerations

EMIF memory controller remains fully functional during emulation halts in order to allow emulation access to external memory.

21.3 EMIF Registers

The external memory interface (EMIF) is controlled by programming its internal memory-mapped registers (MMRs). Table 21-24 lists the memory-mapped registers for the EMIF.

> **NOTE:** All EMIF MMRs, except SDCR, support only word (32-bit) accesses. Performing a byte (8-bit) or halfword (16-bit) write to these registers results in undefined behavior. The SDCR is byte writable to allow the setting of the SR, PD, and PDWR bits without triggering the SDRAM initialization sequence.

The EMIF registers must always be accessed using 32-bit accesses (unless otherwise specified in this chapter). The base address of the EMIF memory-mapped registers is FCFF E800h.

Table 21-24. External Memory Interface (EMIF) Registers

Offset	Acronym	Register Description	Section
00h	MIDR	Module ID Register	Section 21.3.1
04h	AWCC	Asynchronous Wait Cycle Configuration Register	Section 21.3.2
08h	SDCR	SDRAM Configuration Register	Section 21.3.3
0Ch	SDRCR	SDRAM Refresh Control Register	Section 21.3.4
10h	CE2CFG	Asynchronous 1 Configuration Register	Section 21.3.5
14h	CE3CFG	Asynchronous 2 Configuration Register	Section 21.3.5
18h	CE4CFG	Asynchronous 3 Configuration Register	Section 21.3.5
1Ch	CE5CFG	Asynchronous 4 Configuration Register	Section 21.3.5
20h	SDTIMR	SDRAM Timing Register	Section 21.3.6
3Ch	SDSRETR	SDRAM Self Refresh Exit Timing Register	Section 21.3.7
40h	INTRAW	EMIF Interrupt Raw Register	Section 21.3.8
44h	INTMSK	EMIF Interrupt Mask Register	Section 21.3.9
48h	INTMSKSET	EMIF Interrupt Mask Set Register	Section 21.3.10
4Ch	INTMSKCLR	EMIF Interrupt Mask Clear Register	Section 21.3.11
68h	PMCR	Page Mode Control Register	Section 21.3.12

21.3.1 Module ID Register (MIDR)

This is a read-only register indicating the module ID of the EMIF. The MIDR is shown in Figure 21-15 and described in Table 21-25.

Figure 21-15. Module ID Register (MIDR) [offset = 00]

31	0
REV	
R-x	

LEGEND: R = Read only; -*n* = value after reset

Table 21-25. Module ID Register (MIDR) Field Descriptions

Bit	Field	Value	Description
31-0	REV	x	Module ID of EMIF. See the device-specific data manual.

21.3.2 Asynchronous Wait Cycle Configuration Register (AWCC)

The asynchronous wait cycle configuration register (AWCC) is used to configure the parameters for extended wait cycles. Both the polarity of the EMIF_nWAIT pin(s) and the maximum allowable number of extended wait cycles can be configured. The AWCC is shown in Figure 21-16 and described in Table 21-26. Not all devices support both EMIF_nWAIT[1] and EMIF_nWAIT[0], see the device-specific data manual to determine support on each device.

NOTE: The EW bit in the asynchronous *n* configuration register (CE*n*CFG) must be set to allow for the insertion of extended wait cycles.

Figure 21-16. Asynchronous Wait Cycle Configuration Register (AWCCR) [offset = 04h]

31 30	29	28	27 24	23 22	21 20	19 18	17 16
Reserved	WP1	WP0	Reserved	CS5_WAIT	CS4_WAIT	CS3_WAIT	CS2_WAIT
R-3h	R/W-1	R/W-1	R-0	R/W-0	R/W-0	R/W-0	R/W-0

15 8	7 0
Reserved	MAX_EXT_WAIT
R-0	R/W-80h

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 21-26. Asynchronous Wait Cycle Configuration Register (AWCCR) Field Descriptions

Bit	Field	Value	Description
31-30	Reserved	3h	Reserved
29	WP1		EMIF_nWAIT[1] polarity bit. This bit defines the polarity of the EMIF_nWAIT[1] pin.
		0	Insert wait cycles if EMIF_nWAIT[1] pin is low.
		1	Insert wait cycles if EMIF_nWAIT[1] pin is high.
28	WP0		EMIF_nWAIT[0] polarity bit. This bit defines the polarity of the EMIF_nWAIT[0] pin.
		0	Insert wait cycles if EMIF_nWAIT[0] pin is low.
		1	Insert wait cycles if EMIF_nWAIT[0] pin is high.
27-24	Reserved	0	Reserved
23-22	CS5_WAIT	0-3h	Chip Select 5 WAIT signal selection. This signal determines which EMIF_nWAIT[*n*] signal will be used for memory accesses to chip select 5 memory space. This device does not support chip select 5, so any value written to this field has no effect.
21-20	CS4_WAIT		Chip Select 4 WAIT signal selection. This signal determines which EMIF_nWAIT[*n*] signal will be used for memory accesses to chip select 4 memory space.
		0	EMIF_nWAIT[0] pin is used to control external wait states.
		1h	EMIF_nWAIT[1] pin is used to control external wait states.
		2h-3h	Reserved
19-18	CS3_WAIT		Chip Select 3 WAIT signal selection. This signal determines which EMIF_nWAIT[*n*] signal will be used for memory accesses to chip select 3 memory space.
		0	EMIF_nWAIT[0] pin is used to control external wait states.
		1h	EMIF_nWAIT[1] pin is used to control external wait states.
		2h-3h	Reserved
17-16	CS2_WAIT		Chip Select 2 WAIT signal selection. This signal determines which EMIF_nWAIT[*n*] signal will be used for memory accesses to chip select 2 memory space.
		0	EMIF_nWAIT[0] pin is used to control external wait states..
		1h	EMIF_nWAIT[1] pin is used to control external wait states.
		2h-3h	Reserved
15-8	Reserved	0	Reserved
7-0	MAX_EXT_WAIT	0-FFh	Maximum extended wait cycles. The EMIF will wait for a maximum of (MAX_EXT_WAIT + 1) × 16 clock cycles before it stops inserting asynchronous wait cycles and proceeds to the hold period of the access.

21.3.3 SDRAM Configuration Register (SDCR)

The SDRAM configuration register (SDCR) is used to configure various parameters of the SDRAM controller such as the number of internal banks, the internal page size, and the CAS latency to match those of the attached SDRAM device. In addition, this register is used to put the attached SDRAM device into Self-Refresh mode. The SDCR is shown in Figure 21-17 and described in Table 21-27.

NOTE: Writing to the lower three bytes of this register will cause the EMIF to start the SDRAM initialization sequence described in Section 21.2.5.4.

Figure 21-17. SDRAM Configuration Register (SDCR) [offset = 08h]

31	30	29	28	24
SR	PD	PDWR	Reserved	
R/W-0	R/W-0	R/W-0	R-0	

23	16
Reserved	
R-0	

15	14	13	12	11	9	8
Reserved	NM[A]	Reserved		CL		BIT11_9LOCK
R-0	R/W-0	R-0		R/W-3h		R/W-0

7	6	4	3	2	0
Reserved	IBANK		Reserved	PAGESIZE	
R-0	R/W-2h		R-0	R/W-0	

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

A. The NM bit must be set to 1 if the EMIF on your device only has 16 data bus pins.

Table 21-27. SDRAM Configuration Register (SDCR) Field Descriptions

Bit	Field	Value	Description
31	SR		Self-Refresh mode bit. This bit controls entering and exiting of the Self-Refresh mode described in Section 21.2.5.7. The field should be written using a byte-write to the upper byte of SDCR to avoid triggering the SDRAM initialization sequence.
		0	Writing a 0 to this bit will cause connected SDRAM devices and the EMIF to exit the Self-Refresh mode.
		1	Writing a 1 to this bit will cause connected SDRAM devices and the EMIF to enter the Self-Refresh mode.
30	PD		Power Down bit. This bit controls entering and exiting of the power-down mode. The field should be written using a byte-write to the upper byte of SDCR to avoid triggering the SDRAM initialization sequence. If both SR and PD bits are set, the EMIF will go into Self Refresh.
		0	Writing a 0 to this bit will cause connected SDRAM devices and the EMIF to exit the power-down mode.
		1	Writing a 1 to this bit will cause connected SDRAM devices and the EMIF to enter the power-down mode.
29	PDWR		Perform refreshes during power down. Writing a 1 to this bit will cause EMIF to exit power-down state and issue and AUTO REFRESH command every time Refresh May level is set.
28-15	Reserved	0	Reserved. The reserved bit location is always read as 0. If writing to this field, always write the default value of 0.
14	NM		Narrow mode bit. This bit defines whether a 16- or 32-bit-wide SDRAM is connected to the EMIF. This bit field must always be set to 1. Writing to this field triggers the SDRAM initialization sequence.
		0	32-bit SDRAM data bus is used.
		1	16-bit SDRAM data bus is used.
13-12	Reserved	0	Reserved. The reserved bit location is always read as 0. If writing to this field, always write the default value of 0.

Table 21-27. SDRAM Configuration Register (SDCR) Field Descriptions (continued)

Bit	Field	Value	Description
11-9	CL		CAS Latency. This field defines the CAS latency to be used when accessing connected SDRAM devices. A 1 must be simultaneously written to the BIT11_9LOCK bit field of this register in order to write to the CL bit field. Writing to this field triggers the SDRAM initialization sequence.
		0-1h	Reserved
		2h	CAS latency = 2 EMIF_CLK cycles
		3h	CAS latency = 3 EMIF_CLK cycles
		4h-7h	Reserved
8	BIT11_9LOCK		Bits 11 to 9 lock. CL can only be written if BIT11_9LOCK is simultaneously written with a 1. BIT11_9LOCK is always read as 0. Writing to this field triggers the SDRAM initialization sequence.
		0	CL cannot be written.
		1	CL can be written.
7	Reserved	0	Reserved. The reserved bit location is always read as 0. If writing to this field, always write the default value of 0.
6-4	IBANK		Internal SDRAM Bank size. This field defines number of banks inside the connected SDRAM devices. Writing to this field triggers the SDRAM initialization sequence.
		0	1 bank SDRAM devices.
		1	2 bank SDRAM devices.
		2	4 bank SDRAM devices.
		3h-7h	Reserved.
3	Reserved	0	Reserved. The reserved bit location is always read as 0. If writing to this field, always write the default value of 0.
2-0	PAGESIZE		Page Size. This field defines the internal page size of connected SDRAM devices. Writing to this field triggers the SDRAM initialization sequence.
		0	8 column address bits (256 elements per row)
		1h	9 column address bits (512 elements per row)
		2h	10 column address bits (1024 elements per row)
		3h	11 column address bits (2048 elements per row)
		4h-7h	Reserved

21.3.4 SDRAM Refresh Control Register (SDRCR)

The SDRAM refresh control register (SDRCR) is used to configure the rate at which connected SDRAM devices will be automatically refreshed by the EMIF. Refer to Section 21.2.5.6 on the refresh controller for more details. The SDRCR is shown in Figure 21-18 and described in Table 21-28.

Figure 21-18. SDRAM Refresh Control Register (SDRCR) [offset = 0Ch]

31	16
Reserved	
R-0	

15 13	12 0
Reserved	RR
R-0	R/W-80h

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 21-28. SDRAM Refresh Control Register (SDRCR) Field Descriptions

Bit	Field	Value	Description
31-13	Reserved	0	Reserved. The reserved bit location is always read as 0. If writing to this field, always write the default value of 0.
12-0	RR	0-1FFFh	Refresh Rate. This field is used to define the SDRAM refresh period in terms of EMIF_CLK cycles. Writing a value < 0x0020 to this field will cause it to be loaded with (2 × T_RFC) + 1 value from the SDRAM timing register (SDTIMR).

21.3.5 Asynchronous *n* Configuration Registers (CE2CFG-CE5CFG)

The asynchronous *n* configuration registers (CE2CFG, CE3CFG, CE4CFG, and CE5CFG) are used to configure the shaping of the address and control signals during an access to asynchronous memory connected to CS2, CS3, CS4, and CS5, respectively. CS5 is not available on this device. It is also used to program the width of asynchronous interface and to select from various modes of operation. This register can be written prior to any transfer, and any asynchronous transfer following the write will use the new configuration. The CE*n*CFG is shown in Figure 21-19 and described in Table 21-29.

Figure 21-19. Asynchronous *n* Configuration Register (CE*n*CFG) [offset = 10h - 1Ch]

31	30	29 26	25 24
SS	EW[A]	W_SETUP	W_STROBE[B]
R/W-0	R/W-0	R/W-Fh	R/W-3Fh

23 20	19 17	16
W_STROBE[B]	W_HOLD	R_SETUP
R/W-3Fh	R/W-7h	R/W-Fh

15 13	12 7	6 4	3 2	1 0
R_SETUP	R_STROBE[B]	R_HOLD	TA	ASIZE
R/W-Fh	R/W-3Fh	R/W-7h	R/W-3h	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

A. The EW bit must be cleared to 0.

B. This bit field must be cleared to 0 if the EMIF on your device does not have an EMIF_nWAIT pin.

Table 21-29. Asynchronous *n* Configuration Register (CE*n*CFG) Field Descriptions

Bit	Field	Value	Description
31	SS		Select Strobe bit. This bit defines whether the asynchronous interface operates in Normal Mode or Select Strobe Mode. See Section 21.2.6 for details on the two modes of operation.
		0	Normal Mode enabled.
		1	Select Strobe Mode enabled.
30	EW		Extend Wait bit. This bit defines whether extended wait cycles will be enabled. See Section 21.2.6.6 on extended wait cycles for details. This bit field must be set to 0, if the EMIF on your device does not have an EMIF_nWAIT pin.
		0	Extended wait cycles disabled.
		1	Extended wait cycles enabled.
29-26	W_SETUP	0-Fh	Write setup width in EMIF_CLK cycles, minus one cycle. See Section 21.2.6.3 for details.
25-20	W_STROBE	0-3Fh	Write strobe width in EMIF_CLK cycles, minus one cycle. See Section 21.2.6.3 for details.
19-17	W_HOLD	0-7h	Write hold width in EMIF_CLK cycles, minus one cycle. See Section 21.2.6.3 for details.
16-13	R_SETUP	0-Fh	Read setup width in EMIF_CLK cycles, minus one cycle. See Section 21.2.6.3 for details.
12-7	R_STROBE	0-3Fh	Read strobe width in EMIF_CLK cycles, minus one cycle. See Section 21.2.6.3 for details.
6-4	R_HOLD	0-7h	Read hold width in EMIF_CLK cycles, minus one cycle. See Section 21.2.6.3 for details.
3-2	TA	0-3h	Minimum Turn-Around time. This field defines the minimum number of EMIF_CLK cycles between reads and writes, minus one cycle. See Section 21.2.6.3 for details.
1-0	ASIZE		Asynchronous Data Bus Width. This field defines the width of the asynchronous device's data bus.
		0	8-bit data bus
		1h	16-bit data bus
		2h-3h	Reserved

21.3.6 SDRAM Timing Register (SDTIMR)

The SDRAM timing register (SDTIMR) is used to program many of the SDRAM timing parameters. Consult the SDRAM datasheet for information on the appropriate values to program into each field. The SDTIMR is shown in Figure 21-20 and described in Table 21-30.

Figure 21-20. SDRAM Timing Register (SDTIMR) [offset = 20h]

31 27	26 24	23	22 20	19	18 16
T_RFC	T_RP	Rsvd	T_RCD	Rsvd	T_WR
R/W-8h	R/W-2h	R-0	R/W-2h	R-0	R/W-1h

15 12	11 8	7	6 4	3 0
T_RAS	T_RC	Rsvd	T_RRD	Reserved
R/W-5h	R/W-8h	R-0	R/W-1h	R-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 21-30. SDRAM Timing Register (SDTIMR) Field Descriptions

Bit	Field	Value	Description
31-27	T_RFC	0-1Fh	Specifies the Trfc value of the SDRAM. This defines the minimum number of EMIF_CLK cycles from Refresh (REFR) to Refresh (REFR), minus 1: T_RFC = (Trfc/t_{EMIF_CLK}) - 1
26-24	T_RP	0-7h	Specifies the Trp value of the SDRAM. This defines the minimum number of EMIF_CLK cycles from Precharge (PRE) to Activate (ACTV) or Refresh (REFR) command, minus 1: T_RP = (Trp/t_{EMIF_CLK}) - 1
23	Reserved	0	Reserved. The reserved bit location is always read as 0. If writing to this field, always write the default value of 0.
22-20	T_RCD	0-7h	Specifies the Trcd value of the SDRAM. This defines the minimum number of EMIF_CLK cycles from Active (ACTV) to Read (READ) or Write (WRT), minus 1: T_RCD = (Trcd/t_{EMIF_CLK}) - 1
19	Reserved	0	Reserved. The reserved bit location is always read as 0. If writing to this field, always write the default value of 0.
18-16	T_WR	0-7h	Specifies the Twr value of the SDRAM. This defines the minimum number of EMIF_CLK cycles from last Write (WRT) to Precharge (PRE), minus 1: T_WR = (Twr/t_{EMIF_CLK}) - 1
15-12	T_RAS	0-Fh	Specifies the Tras value of the SDRAM. This defines the minimum number of EMIF_CLK clock cycles from Activate (ACTV) to Precharge (PRE), minus 1: T_RAS = (Tras/t_{EMIF_CLK}) - 1
11-8	T_RC	0-Fh	Specifies the Trc value of the SDRAM. This defines the minimum number of EMIF_CLK clock cycles from Activate (ACTV) to Activate (ACTV), minus 1: T_RC = (Trc/t_{EMIF_CLK}) - 1
7	Reserved	0	Reserved. The reserved bit location is always read as 0. If writing to this field, always write the default value of 0.
6-4	T_RRD	0-7h	Specifies the Trrd value of the SDRAM. This defines the minimum number of EMIF_CLK clock cycles from Activate (ACTV) to Activate (ACTV) for a different bank, minus 1: T_RRD = (Trrd/t_{EMIF_CLK}) - 1
3-0	Reserved	0	Reserved. The reserved bit location is always read as 0. If writing to this field, always write the default value of 0.

21.3.7 SDRAM Self Refresh Exit Timing Register (SDSRETR)

The SDRAM self refresh exit timing register (SDSRETR) is used to program the amount of time between when the SDRAM exits Self-Refresh mode and when the EMIF issues another command. The SDSRETR is shown in Figure 21-21 and described in Table 21-31.

Figure 21-21. SDRAM Self Refresh Exit Timing Register (SDSRETR) [offset = 3Ch]

31	16
Reserved	
R-0	

15	5	4	0
Reserved		T_XS	
R-0		R/W-9h	

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 21-31. SDRAM Self Refresh Exit Timing Register (SDSRETR) Field Descriptions

Bit	Field	Value	Description
31-5	Reserved	0	Reserved. The reserved bit location is always read as 0.
4-0	T_XS	0-1Fh	This field specifies the minimum number of ECLKOUT cycles from Self-Refresh exit to any command, minus one. T_XS = Txsr / t_{EMIF_CLK} - 1

21.3.8 EMIF Interrupt Raw Register (INTRAW)

The EMIF interrupt raw register (INTRAW) is used to monitor and clear the EMIF's hardware-generated Asynchronous Timeout Interrupt. The AT bit in this register will be set when an Asynchronous Timeout occurs regardless of the status of the EMIF interrupt mask set register (INTMSKSET) and EMIF interrupt mask clear register (INTMSKCLR). Writing a 1 to this bit will clear it. The EMIF on some devices does not have the EMIF_nWAIT pin; therefore, these registers and fields are reserved on those devices. The INTRAW is shown in Figure 21-22 and described in Table 21-32.

Figure 21-22. EMIF Interrupt Raw Register (INTRAW) [offset = 40h]

31	8
Reserved	
R-0	

7	3	2	1	0
Reserved		WR	LT	AT
R-0		R/W1C-0	R/W1C-0	R/W1C-0

LEGEND: R/W = Read/Write; R = Read only; W1C = Write 1 to clear (writing 0 has no effect); -*n* = value after reset

Table 21-32. EMIF Interrupt Raw Register (INTRAW) Field Descriptions

Bit	Field	Value	Description
31-3	Reserved	0	Reserved. The reserved bit location is always read as 0. If writing to this field, always write the default value of 0.
2	WR		Wait Rise. This bit is set to 1 by hardware to indicate that a rising edge on the EMIF_nWAIT pin has occurred.
		0	Indicates that a rising edge has not occurred on the EMIF_nWAIT pin. Writing a 0 has no effect.
		1	Indicates that a rising edge has occurred on the EMIF_nWAIT pin. Writing a 1 will clear this bit and the WR_MASKED bit in the EMIF interrupt masked register (INTMSK).
1	LT		Line Trap. Set to 1 by hardware to indicate illegal memory access type or invalid cache line size.
		0	Writing a 0 has no effect.
		1	Indicates that a line trap has occurred. Writing a 1 will clear this bit as well as the LT_MASKED bit in the EMIF interrupt masked register(INTMSK).
0	AT		Asynchronous Timeout. This bit is set to 1 by hardware to indicate that during an extended asynchronous memory access cycle, the EMIF_nWAIT pin did not go inactive within the number of cycles defined by the MAX_EXT_WAIT field in the asynchronous wait cycle configuration register (AWCC).
		0	Indicates that an Asynchronous Timeout has not occurred. Writing a 0 has no effect.
		1	Indicates that an Asynchronous Timeout has occurred. Writing a 1 will clear this bit as well as the AT_MASKED bit in the EMIF interrupt masked register (INTMSK).

21.3.9 EMIF Interrupt Masked Register (INTMSK)

Like the EMIF interrupt raw register (INTRAW), the EMIF interrupt masked register (INTMSK) is used to monitor and clear the status of the EMIF's hardware-generated Asynchronous Timeout Interrupt. The main difference between the two registers is that when the AT_MASKED bit in this register is set, an active-high pulse will be sent to the CPU interrupt controller. Also, the AT_MASKED bit field in INTMSK is only set to 1 if the associated interrupt has been enabled in the EMIF interrupt mask set register (INTMSKSET). The EMIF on some devices does not have the EMIF_nWAIT pin, therefore, these registers and fields are reserved on those devices. The INTMSK is shown in Figure 21-23 and described in Table 21-33.

Figure 21-23. EMIF Interrupt Mask Register (INTMSK) [offset = 44h]

31	8
Reserved	
R-0	

7	3	2	1	0
Reserved		WR_MASKED	LT_MASKED	AT_MASKED
R-0		R/W1C-0	R/W1C-0	R/W1C-0

LEGEND: R/W = Read/Write; R = Read only; W1C = Write 1 to clear (writing 0 has no effect); -*n* = value after reset

Table 21-33. EMIF Interrupt Mask Register (INTMSK) Field Descriptions

Bit	Field	Value	Description
31-3	Reserved	0	Reserved. The reserved bit location is always read as 0. If writing to this field, always write the default value of 0.
2	WR_MASKED		Wait Rise Masked. This bit is set to 1 by hardware to indicate a rising edge has occurred on the EMIF_nWAIT pin, provided that the WR_MASK_SET bit is set to 1 in the EMIF interrupt mask set register (INTMSKSET).
		0	Indicates that a wait rise interrupt has not been generated. Writing a 0 has no effect.
		1	Indicates that a wait rise interrupt has been generated. Writing a 1 will clear this bit and the WR bit in the EMIF interrupt raw register (INTRAW).
1	LT_MASKED		Masked Line Trap. Set to 1 by hardware to indicate illegal memory access type or invalid cache line size, only if the LT_MASK_SET bit in the EMIF interrupt mask set register (INTMSKSET) is set to 1.
		0	Writing a 0 has no effect.
		1	Writing a 1 will clear this bit as well as the LT bit in the EMIF interrupt raw register(INTRAW).
0	AT_MASKED		Asynchronous Timeout Masked. This bit is set to 1 by hardware to indicate that during an extended asynchronous memory access cycle, the EMIF_nWAIT pin did not go inactive within the number of cycles defined by the MAX_EXT_WAIT field in the asynchronous wait cycle configuration register (AWCC), provided that the AT_MASK_SET bit is set to 1 in the EMIF interrupt mask set register (INTMSKSET).
		0	Indicates that an Asynchronous Timeout Interrupt has not been generated. Writing a 0 has no effect.
		1	Indicates that an Asynchronous Timeout Interrupt has been generated. Writing a 1 will clear this bit as well as the AT bit in the EMIF interrupt raw register (INTRAW).

21.3.10 EMIF Interrupt Mask Set Register (INTMSKSET)

The EMIF interrupt mask set register (INTMSKSET) is used to enable the Asynchronous Timeout Interrupt. If read as 1, the AT_MASKED bit in the EMIF interrupt masked register (INTMSK) will be set and an interrupt will be generated when an Asynchronous Timeout occurs. If read as 0, the AT_MASKED bit will always read 0 and no interrupt will be generated when an Asynchronous Timeout occurs. Writing a 1 to the AT_MASK_SET bit enables the Asynchronous Timeout Interrupt. The EMIF on some devices does not have the EMIF_nWAIT pin; therefore, these registers and fields are reserved on those devices. The INTMSKSET is shown in Figure 21-24 and described in Table 21-34.

Figure 21-24. EMIF Interrupt Mask Set Register (INTMSKSET) [offset = 48h]

31 — 16
Reserved
R-0

15 — 3	2	1	0
Reserved	WR_MASK_SET	LT_MASK_SET	AT_MASK_SET
R-0	R/W-0	R/W-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 21-34. EMIF Interrupt Mask Set Register (INTMSKSET) Field Descriptions

Bit	Field	Value	Description
31-3	Reserved	0	Reserved. The reserved bit location is always read as 0. If writing to this field, always write the default value of 0.
2	WR_MASK_SET		Wait Rise Mask Set. This bit determines whether or not the wait rise Interrupt is enabled. Writing a 1 to this bit sets this bit, sets the WR_MASK_CLR bit in the EMIF interrupt mask clear register (INTMSKCLR), and enables the wait rise interrupt. To clear this bit, a 1 must be written to the WR_MASK_CLR bit in INTMSKCLR.
		0	Indicates that the wait rise interrupt is disabled. Writing a 0 has no effect.
		1	Indicates that the wait rise interrupt is enabled. Writing a 1 sets this bit and the WR_MASK_CLR bit in the EMIF interrupt mask clear register (INTMSKCLR).
1	LT_MASK_SET		Mask set for LT_MASKED bit in the EMIF interrupt mask register (INTMSK).
		0	Indicates that the line trap interrupt is disabled. Writing a 0 has no effect.
		1	Indicates that the line trap interrupt is enabled. Writing a 1 sets this bit and the LT_MASK_CLR bit in the EMIF interrupt mask clear register (INTMSKCLR).
0	AT_MASK_SET		Asynchronous Timeout Mask Set. This bit determines whether or not the Asynchronous Timeout Interrupt is enabled. Writing a 1 to this bit sets this bit, sets the AT_MASK_CLR bit in the EMIF interrupt mask clear register (INTMSKCLR), and enables the Asynchronous Timeout Interrupt. To clear this bit, a 1 must be written to the AT_MASK_CLR bit of the EMIF interrupt mask clear register (INTMSKCLR).
		0	Indicates that the Asynchronous Timeout Interrupt is disabled. Writing a 0 has no effect.
		1	Indicates that the Asynchronous Timeout Interrupt is enabled. Writing a 1 sets this bit and the AT_MASK_CLR bit in the EMIF interrupt mask clear register (INTMSKCLR).

21.3.11 EMIF Interrupt Mask Clear Register (INTMSKCLR)

The EMIF interrupt mask clear register (INTMSKCLR) is used to disable the Asynchronous Timeout Interrupt. If read as 1, the AT_MASKED bit in the EMIF interrupt masked register (INTMSK) will be set and an interrupt will be generated when an Asynchronous Timeout occurs. If read as 0, the AT_MASKED bit will always read 0 and no interrupt will be generated when an Asynchronous Timeout occurs. Writing a 1 to the AT_MASK_CLR bit disables the Asynchronous Timeout Interrupt. The EMIF on some devices does not have the EMIF_nWAIT pin, therefore, these registers and fields are reserved on those devices. The INTMSKCLR is shown in Figure 21-25 and described in Table 21-35.

Figure 21-25. EMIF Interrupt Mask Clear Register (INTMSKCLR) [offset = 4Ch]

31-16
Reserved
R-0

15-3	2	1	0
Reserved	WR_MASK_CLR	LT_MASK_CLR	AT_MASK_CLR
R-0	R/W-0	R/W-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 21-35. EMIF Interrupt Mask Clear Register (INTMSKCLR) Field Descriptions

Bit	Field	Value	Description
31-3	Reserved	0	Reserved. The reserved bit location is always read as 0. If writing to this field, always write the default value of 0.
2	WR_MASK_CLR		Wait Rise Mask Clear. This bit determines whether or not the wait rise interrupt is enabled. Writing a 1 to this bit clears this bit, clears the WR_MASK_SET bit in the EMIF interrupt mask set register (INTMSKSET), and disables the wait rise interrupt. To set this bit, a 1 must be written to the WR_MASK_SET bit in INTMSKSET.
		0	Indicates that the wait rise interrupt is disabled. Writing a 0 has no effect.
		1	Indicates that the wait rise interrupt is enabled. Writing a 1 clears this bit and the WR_MASK_SET bit in the EMIF interrupt mask set register (INTMSKSET).
1	LT_MASK_CLR		Line trap Mask Clear. This bit determines whether or not the line trap interrupt is enabled. Writing a 1 to this bit clears this bit, clears the LT_MASK_SET bit in the EMIF interrupt mask set register (INTMSKSET), and disables the line trap interrupt. To set this bit, a 1 must be written to the LT_MASK_SET bit in INTMSKSET.
		0	Indicates that the line trap interrupt is disabled. Writing a 0 has no effect.
		1	Indicates that the line trap interrupt is enabled. Writing a 1 clears this bit and the LT_MASK_SET bit in the EMIF interrupt mask set register (INTMSKSET).
0	AT_MASK_CLR		Asynchronous Timeout Mask Clear. This bit determines whether or not the Asynchronous Timeout Interrupt is enabled. Writing a 1 to this bit clears this bit, clears the AT_MASK_SET bit in the EMIF interrupt mask set register (INTMSKSET), and disables the Asynchronous Timeout Interrupt. To set this bit, a 1 must be written to the AT_MASK_SET bit of the EMIF interrupt mask set register (INTMSKSET).
		0	Indicates that the Asynchronous Timeout Interrupt is disabled. Writing a 0 has no effect.
		1	Indicates that the Asynchronous Timeout Interrupt is enabled. Writing a 1 clears this bit and the AT_MASK_SET bit in the EMIF interrupt mask set register (INTMSKSET).

21.3.12 Page Mode Control Register (PMCR)

The page mode control register (PMCR) is shown in Figure 21-26 and described in Table 21-36. This register is configured when using NOR Flash page mode.

Figure 21-26. Page Mode Control Register (PMCR) [offset = 68h]

31 - 26	25	24
CS5_PG_DEL	CS5_PG_SIZE	CS5_PG_MD_EN
R/W-3Fh	R/W-0	R/W-0

23 - 18	17	16
CS4_PG_DEL	CS4_PG_SIZE	CS4_PG_MD_EN
R/W-3Fh	R/W-0	R/W-0

15 - 10	9	8
CS3_PG_DEL	CS3_PG_SIZE	CS3_PG_MD_EN
R/W-3Fh	R/W-0	R/W-0

7 - 2	1	0
CS2_PG_DEL	CS2_PG_SIZE	CS2_PG_MD_EN
R/W-3Fh	R/W-0	R/W-0

LEGEND: R/W = Read/Write; -*n* = value after reset

Table 21-36. Page Mode Control Register (PMCR) Field Descriptions

Bit	Field	Value	Description
31-26	CS5_PG_DEL	1-3Fh	Page access delay for NOR Flash connected on CS5. CS5 is not available on this device.
25	CS5_PG_SIZE		Page Size for NOR Flash connected on CS5. CS5 is not available on this device.
24	CS5_PG_MD_EN		Page Mode enable for NOR Flash connected on CS5. CS5 is not available on this device.
23-18	CS4_PG_DEL	1-3Fh	Page access delay for NOR Flash connected on CS4. Number of EMIF_CLK cycles required for the page read data to be valid, minus one cycle. This value must not be cleared to 0.
17	CS4_PG_SIZE		Page Size for NOR Flash connected on CS4.
		0	Page size is 4 words
		1	Page size is 8 words
16	CS4_PG_MD_EN		Page Mode enable for NOR Flash connected on CS4.
		0	Page mode disabled for this chip select
		1	Page mode enabled for this chip select
15-10	CS3_PG_DEL	1-3Fh	Page access delay for NOR Flash connected on CS3. Number of EMIF_CLK cycles required for the page read data to be valid, minus one cycle. This value must not be cleared to 0.
9	CS3_PG_SIZE		Page Size for NOR Flash connected on CS3.
		0	Page size is 4 words
		1	Page size is 8 words
8	CS3_PG_MD_EN		Page Mode enable for NOR Flash connected on CS3.
		0	Page mode disabled for this chip select
		1	Page mode enabled for this chip select
7-2	CS2_PG_DEL	1-3Fh	Page access delay for NOR Flash connected on CS2. Number of EMIF_CLK cycles required for the page read data to be valid, minus one cycle. This value must not be cleared to 0.
1	CS2_PG_SIZE		Page Size for NOR Flash connected on CS2.
		0	Page size is 4 words
		1	Page size is 8 words
0	CS2_PG_MD_EN		Page Mode enable for NOR Flash connected on CS2.
		0	Page mode disabled for this chip select
		1	Page mode enabled for this chip select

21.4 Example Configuration

This section presents an example of interfacing the EMIF to both an SDR SDRAM device and an asynchronous flash device.

21.4.1 Hardware Interface

Figure 21-27 shows the hardware interface between the EMIF, a Samsung K4S641632H-TC(L)70 64Mb SDRAM device, and two SHARP LH28F800BJE-PTTL90 8Mb Flash memory. The connection between the EMIF and the SDRAM is straightforward, but the connection between the EMIF and the flash deserves a detailed look.

The address inputs for the flash are provided by three sources. The A[18:0] address inputs are provided by a combination of the EMIF_A and EMIF_BA pins according to Section 21.2.6.1. RD/nBY signal from one flash is connected to EMIF_nWAIT pin of EMIF.

Finally, this example configuration connects the EMIF_nWE pin to the nWE input of the flash and operates the EMIF in Select Strobe Mode.

21.4.2 Software Configuration

The following sections describe how to configure the EMIF registers and bit fields to interface the EMIF with the Samsung K4S641632H-TC(L)70 SDRAM and the SHARP LH28F800BJE-PTTL90 8Mb Flash memory.

21.4.2.1 Configuring the SDRAM Interface

This section describes how to configure the EMIF to interface with the Samsung K4S641632H-TC(L)70 SDRAM with a clock frequency of f_{EMIF_CLK} = 100 MHz. Procedure A described in Section 21.2.5.5 is followed which assumes that the SDRAM power-up timing constraint were met during the SDRAM Auto-Initialization sequence after Reset.

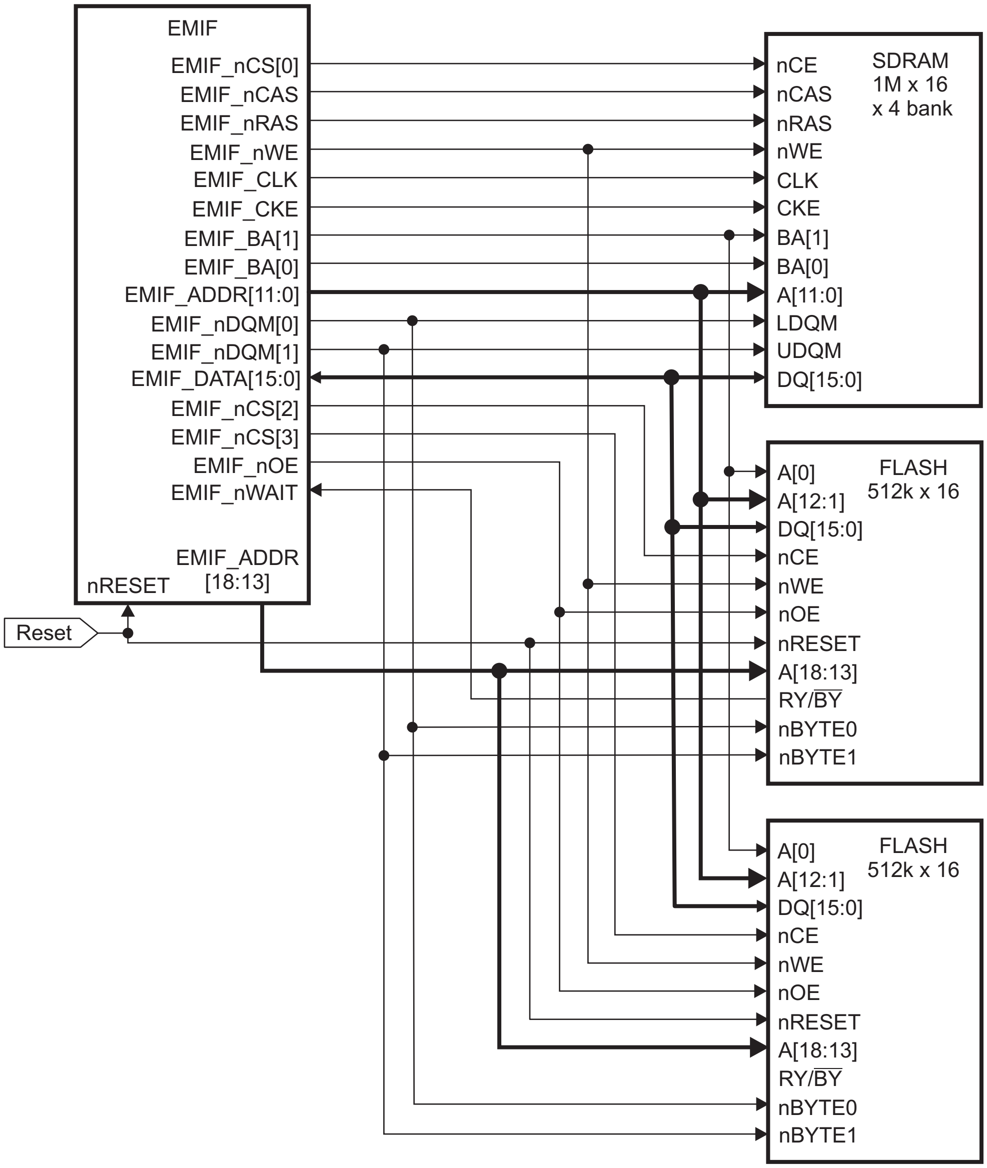
21.4.2.1.1 PLL Programming for the EMIF to K4S641632H-TC(L)70 Interface

The device global clock module (GCM) should first be programmed to select the desired EMIF_CLK frequency. Before doing this, the SDRAM should be placed in Self-Refresh Mode by setting the SR bit in the SDRAM configuration register (SDCR). The SR bit should be set using a byte-write to the upper byte of the SDCR to avoid triggering the SDRAM Initialization Sequence. The EMIF_CLK frequency can now be configured to the desired value by selecting the appropriate clock source for the VCLK3 domain. Once the VCLK3 domain frequency has been configured, remove the SDRAM from Self-Refresh by clearing the SR bit in SDCR, again with a byte-write.

Table 21-37. SR Field Value For the EMIF to K4S641632H-TC(L)70 Interface

Field	Value	Purpose
SR	1 then 0	To place the EMIF into the self refresh state

Figure 21-27. Example Configuration Interface



21.4.2.1.2 SDRAM Timing Register (SDTIMR) Settings for the EMIF to K4S641632H-TC(L)70 Interface

The fields of the SDRAM timing register (SDTIMR) should be programmed first as described in Table 21-38 to satisfy the required timing parameters for the K4S641632H-TC(L)70. Based on these calculations, a value of 6111 4610h should be written to SDTIMR. Figure 21-28 shows a graphical description of how SDTIMR should be programmed.

Table 21-38. SDTIMR Field Calculations for the EMIF to K4S641632H-TC(L)70 Interface

Field Name	Formula	Value from K4S641632H-TC(L)70 Datasheet	Value Calculated for Field
T_RFC	T_RFC >= (t_{RFC} × f_{EMIF_CLK}) - 1	t_{RC} = 68 ns (min)[1]	6
T_RP	T_RP >= (t_{RP} × f_{EMIF_CLK}) - 1	t_{RP} = 20 ns (min)	1
T_RCD	T_RCD >= (t_{RCD} × f_{EMIF_CLK}) - 1	t_{RCD} = 20 ns (min)	1
T_WR	T_WR >= (t_{WR} × f_{EMIF_CLK}) - 1	t_{RDL} = 2 CLK = 20 ns (min)[2]	1
T_RAS	T_RAS >= (t_{RAS} × f_{EMIF_CLK}) - 1	t_{RAS} = 49 ns (min)	4
T_RC	T_RC >= (t_{RC} × f_{EMIF_CLK}) - 1	t_{RC} = 68 ns (min)	6
T_RRD	T_RRD >= (t_{RRD} × f_{EMIF_CLK}) - 1	t_{RRD} = 14 ns (min)	1

[1] The Samsung datasheet does not specify a t_{RFC} value. Instead, Samsung specifies t_{RC} as the minimum auto refresh period.

[2] The Samsung datasheet does not specify a t_{WR} value. Instead, Samsung specifies t_{RDL} as last data in to row precharge minimum delay.

Figure 21-28. SDRAM Timing Register (SDTIMR)

31–27	26–24	23	22–20	19	18–16
0 0110	001	0	001	0	001
T_RFC	T_RP	Rsvd	T_RCD	Rsvd	T_WR

15–12	11–8	7	6–4	3–0
0100	0110	0	001	0000
T_RAS	T_RC	Rsvd	T_RRD	Reserved

21.4.2.1.3 SDRAM Self Refresh Exit Timing Register (SDSRETR) Settings for the EMIF to K4S641632H-TC(L)70 Interface

The SDRAM self refresh exit timing register (SDSRETR) should be programmed second to satisfy the t_{XSR} timing requirement from the K4S641632H-TC(L)70 datasheet. Table 21-39 shows the calculation of the proper value to program into the T_XS field of this register. Based on this calculation, a value of 6h should be written to SDSRETR. Figure 21-29 shows how SDSRETR should be programmed.

Table 21-39. RR Calculation for the EMIF to K4S641632H-TC(L)70 Interface

Field Name	Formula	Value from K4S641632H-TC(L)70 Datasheet	Value Calculated for Field
T_XS	T_XS >= (t_{XSR} × f_{EMIF_CLK}) - 1	t_{RC} = 68 ns (min)[1]	6

[1] The Samsung datasheet does not specify a t_{XSR} value. Instead, Samsung specifies t_{RC} as the minimum required time after CKE going high to complete self refresh exit.

Figure 21-29. SDRAM Self Refresh Exit Timing Register (SDSRETR)

31	16
0000 0000 0000 0000	
Reserved	

15 5	4 0
000 0000 0000	0 0110
Reserved	T_XS

21.4.2.1.4 SDRAM Refresh Control Register (SDRCR) Settings for the EMIF to K4S641632H-TC(L)70 Interface

The SDRAM refresh control register (SDRCR) should next be programmed to satisfy the required refresh rate of the K4S641632H-TC(L)70. Table 21-40 shows the calculation of the proper value to program into the RR field of this register. Based on this calculation, a value of 61Ah should be written to SDRCR. Figure 21-30 shows how SDRCR should be programmed.

Table 21-40. RR Calculation for the EMIF to K4S641632H-TC(L)70 Interface

Field Name	Formula	Values	Value Calculated for Field
RR	RR ≤ f_{EMIF_CLK} × $t_{Refresh\ Period}$ / n_{cycles}	From SDRAM datasheet: $t_{Refresh\ Period}$ = 64 ms; n_{cycles} = 4096 EMIF clock rate: f_{EMIF_CLK} = 100 MHz	RR = 1562 cycles = 61Ah cycles

Figure 21-30. SDRAM Refresh Control Register (SDRCR)

31 19	18 16
0 0000 0000 0000	000
Reserved	Reserved

15 13	12 0
000	0 0110 0001 1010 (61Ah)
Reserved	RR

21.4.2.1.5 SDRAM Configuration Register (SDCR) Settings for the EMIF to K4S641632H-TC(L)70 Interface

Finally, the fields of the SDRAM configuration register (SDCR) should be programmed as described in Table 21-37 to properly interface with the K4S641632H-TC(L)70 device. Based on these settings, a value of 4720h should be written to SDCR. Figure 21-31 shows how SDCR should be programmed. The EMIF is now ready to perform read and write accesses to the SDRAM.

Table 21-41. SDCR Field Values For the EMIF to K4S641632H-TC(L)70 Interface

Field	Value	Purpose
SR	0	To avoid placing the EMIF into the self refresh state
NM	1	To configure the EMIF for a 16-bit data bus
CL	011b	To select a CAS latency of 3
BIT11_9LOCK	1	To allow the CL field to be written
IBANK	010b	To select 4 internal SDRAM banks
PAGESIZE	0	To select a page size of 256 words

Figure 21-31. SDRAM Configuration Register (SDCR)

31	30	29	28 24
0	0	0	0 0000
SR	Reserved	Reserved	Reserved

23 18	17	16
00 0000	0	0
Reserved	Reserved	Reserved

15	14	13	12	11 9	8
0	1	0	0	011	1
Reserved	NM	Reserved	Reserved	CL	BIT11_9LOCK

7	6 4	3	2 0
0	010	0	000
Reserved	IBANK	Reserved	PAGESIZE

21.4.2.2 Configuring the Flash Interface

This section describes how to configure the EMIF to interface with the two of SHARP LH28F800BJE-PTTL90 8Mb Flash memory with a clock frequency of f_{EMIF_CLK} = 100 MHz. The example assumes that one flash is connected to EMIF_nCS2 and the other to EMIF_nCS3.

21.4.2.2.1 Asynchronous 1 Configuration Register (CE2CFG) Settings for the EMIF to LH28F800BJE-PTTL90 Interface

The asynchronous 1 configuration register (CE2CFG) and asynchronous 2 configuration register (CE3CFG) are the only registers that is necessary to program for this asynchronous interface (assuming that one Flash is connected to EMIF_nCS[2] and the other to EMIF_nCS[3]. The SS bit (in both registers) should be set to 1 to enable Select Strobe Mode and the ASIZE field (in both registers) should be set to 1 to select a 16-bit interface. The other fields in this register control the shaping of the EMIF signals, and the proper values can be determined by referring to the AC Characteristics in the Flash datasheet and the device datasheet. Based on the following calculations, a value of 8862 25BDh should be written to CE2CFG. Table 21-42 and Table 21-43 show the pertinent AC Characteristics for reads and writes to the Flash device, and Figure 21-32 and Figure 21-33 show the associated timing waveforms. Finally, Figure 21-34 shows programming the CE*n*CFG (*n* = 2, 3) with the calculated values.

Table 21-42. AC Characteristics for a Read Access

AC Characteristic	Device	Definition	Min	Max	Unit
t_{SU}	EMIF	Setup time, read EMIF_D before EMIF_CLK high	6.5		ns
t_H	EMIF	Data hold time, read EMIF_D after EMIF_CLK high	1		ns
t_D	EMIF	Output delay time, EMIF_CLK high to output signal valid		7	ns
t_{ELQV}	Flash	nCE to Output Delay		90	ns
t_{EHQZ}	Flash	nCE High to Output in High Impedance		55	ns

Table 21-43. AC Characteristics for a Write Access

AC Characteristic	Device	Definition	Min	Max	Unit
t_{AVAV}	Flash	Write Cycle Time	90		ns
t_{ELEH}	Flash	nCE Pulse Width Low	50		ns
t_{EHEL}	Flash	nCE Pulse Width High (not shown in Figure 21-33)	30		ns

Figure 21-32. LH28F800BJE-PTTL90 to EMIF Read Timing Waveforms

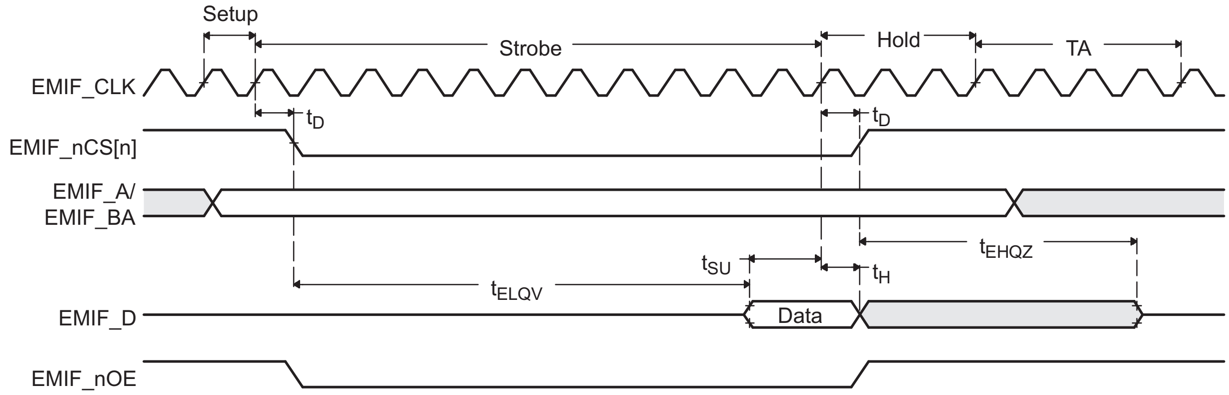
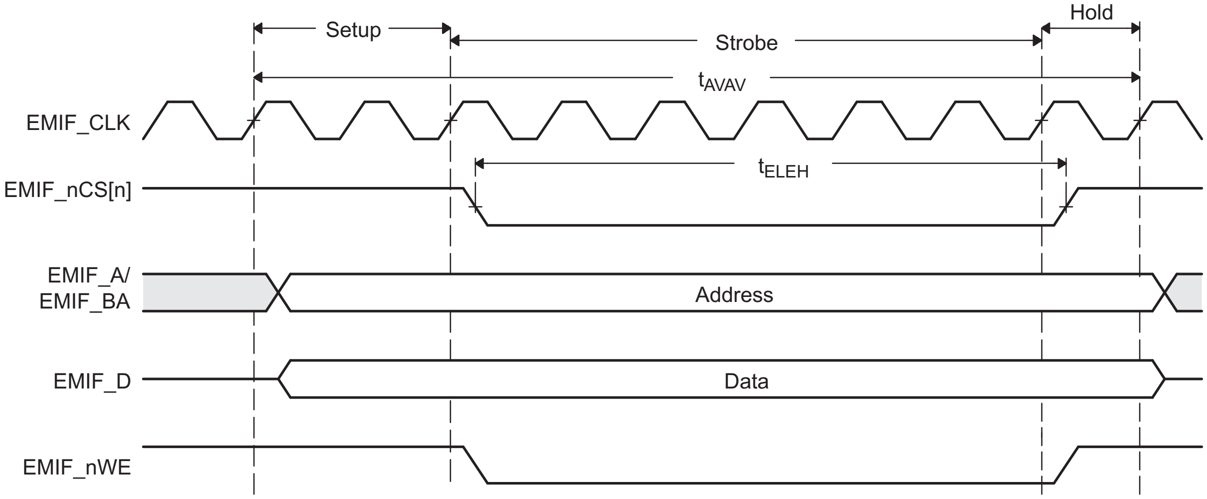


Figure 21-33. LH28F800BJE-PTTL90 to EMIF Write Timing Waveforms



The R_STROBE field should be set to meet the following equation:

R_STROBE >= $(t_D + t_{ELQV} + t_{SU}) \times f_{EMIF\_CLK}$ - 1

R_STROBE >= (7 ns + 90 ns + 6.5 ns) × 100 MHz - 1

R_STROBE >= 9.35

R_STROBE = 10

The R_HOLD field must be large enough to satisfy the EMIF Data hold time, t_H:

R_HOLD > = $t_H \times f_{EMIF\_CLK}$ - 1

R_HOLD >= 1 ns × 100 MHz - 1

R_HOLD >= -0.9

The R_HOLD field must also combine with the TA field to satisfy the Flash's nCE High to Output in High Impedance time, t_{EHQZ}:

R_HOLD + TA >= $(t_D + t_{EHQZ}) \times f_{EMIF\_CLK}$ - 2

R_HOLD + TA >= (7 ns + 55 ns) × 100 MHz - 2

R_HOLD + TA >= 4.2

The largest value that can be programmed into the TA field is 3h, therefore the following values can be used:

R_HOLD = 2

TA = 3

For Writes, the W_STROBE field should be set to satisfy the Flash's nCE Pulse Width constraint, t_{ELEH}:

W_STROBE >= $t_{ELEH} \times f_{EMIF\_CLK}$ - 1

W_STROBE >= 50 ns × 100 MHz - 1

W_STROBE >= 4

The W_SETUP and W_HOLD fields should combine to satisfy the Flash's nCE Pulse Width High constraint, t_{EHEL}, when performing back-to-back writes:

W_SETUP + W_HOLD > = t_{EHEL} × f_{EMIF_CLK} - 2

W_SETUP + W_HOLD > = 30 ns × 100 MHz - 2

W_SETUP + W_HOLD > = 1

In addition, the entire Write access length must satisfy the Flash's minimum Write Cycle Time, t_{AVAV}:

W_SETUP + W_STROBE + W_HOLD >= t_{AVAV} × f_{EMIF_CLK} - 3

W_SETUP + W_STROBE + W_HOLD >= 90 ns × 100 MHz - 3

W_SETUP + W_STROBE + W_HOLD >= 6

Solving the above equations for the Write fields results in the following possible solution:

W_SETUP = 1

W_STROBE = 5

W_HOLD = 0

Adding a 10 ns (1 cycle) margin to each of the periods (excluding TA which is already at its maximum) in this example produces the following recommended values:

W_SETUP = 2h

W_STROBE = 6h

W_HOLD = 1h

R_SETUP = 1h

R_STROBE = Bh

R_HOLD = 3h

TA = 3h

Figure 21-34. Asynchronous *m* Configuration Register (*m* = 1, 2) (CE*n*CFG (*n* = 2, 3))

31	30	29 26	25 24
1	0	0010	00
SS	EW	W_SETUP	W_STROBE

23 20	19 17	16
0110	001	0
W_STROBE	W_HOLD	R_SETUP

15 13	12 7	6 4	3 2	1 0
001	001011	011	11	01
R_SETUP	R_STROBE	R_HOLD	TA	ASIZE

Analog To Digital Converter (ADC) Module

This chapter describes the analog to digital converter (ADC) interface module.

Topic **Page**

22.1 Overview

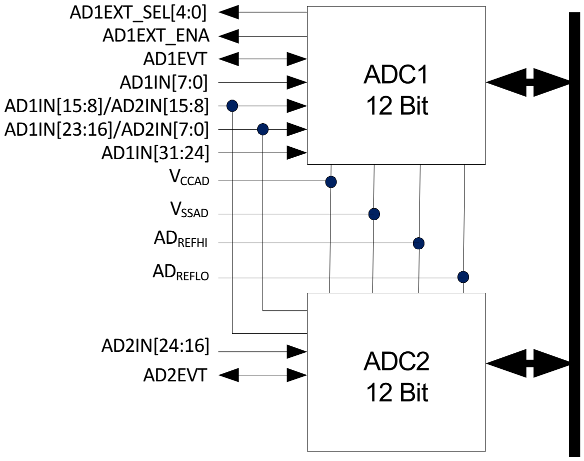
This microcontrollers implements up to two instances of the ADC module. The main features of the ADC module are:

- Selectable 10-bit or 12-bit resolution
- Successive-approximation-register architecture
- Three conversion groups – Group1, Group2, and Event Group
- All three conversion groups can be configured to be hardware-triggered; group1 and group2 can also be triggered by software
- Conversion results are stored in a 64-word memory (SRAM)
  - These 64 words are divided between the three conversion groups and are configurable by software
  - Accesses to the conversion result RAM are protected by parity
- Flexible options for generating DMA requests for transferring conversion results
- Selectable channel conversion order
  - Sequential conversions in ascending order of channel number, OR
  - User-defined channel conversion order with the Enhanced Channel Selection Mode
    - The Enhanced Channel Selection Mode is only available to ADC1.
- Single or continuous conversion modes
- Embedded self-test logic for input channel failure detection (open / short to power / short to ground)
- Embedded calibration logic for offset error correction
- Enhanced Power-down mode
- External event pin (ADEVT) to trigger conversions
  - ADEVT is also programmable as general-purpose I/O
- Eight hardware events to trigger conversions

The two instances of the 12-bit ADC modules on the microcontroller share 16 analog input channels. The connections are shown in Figure 22-1.

- ADC1 supports 32 channels.
- ADC2 supports 25 channels, of which 16 channels are shared with ADC1.
- When using both ADC1 and ADC2 on a shared channel, the sample windows must be identical such that the sample windows completely match each other or non-overlapping with a minimum of 2 ADC cycles buffer between the end of one ADC's sample window and the start of the other ADC's sample window.
- The reference voltages, as well as operating supply and ground, are shared between the two ADC cores.

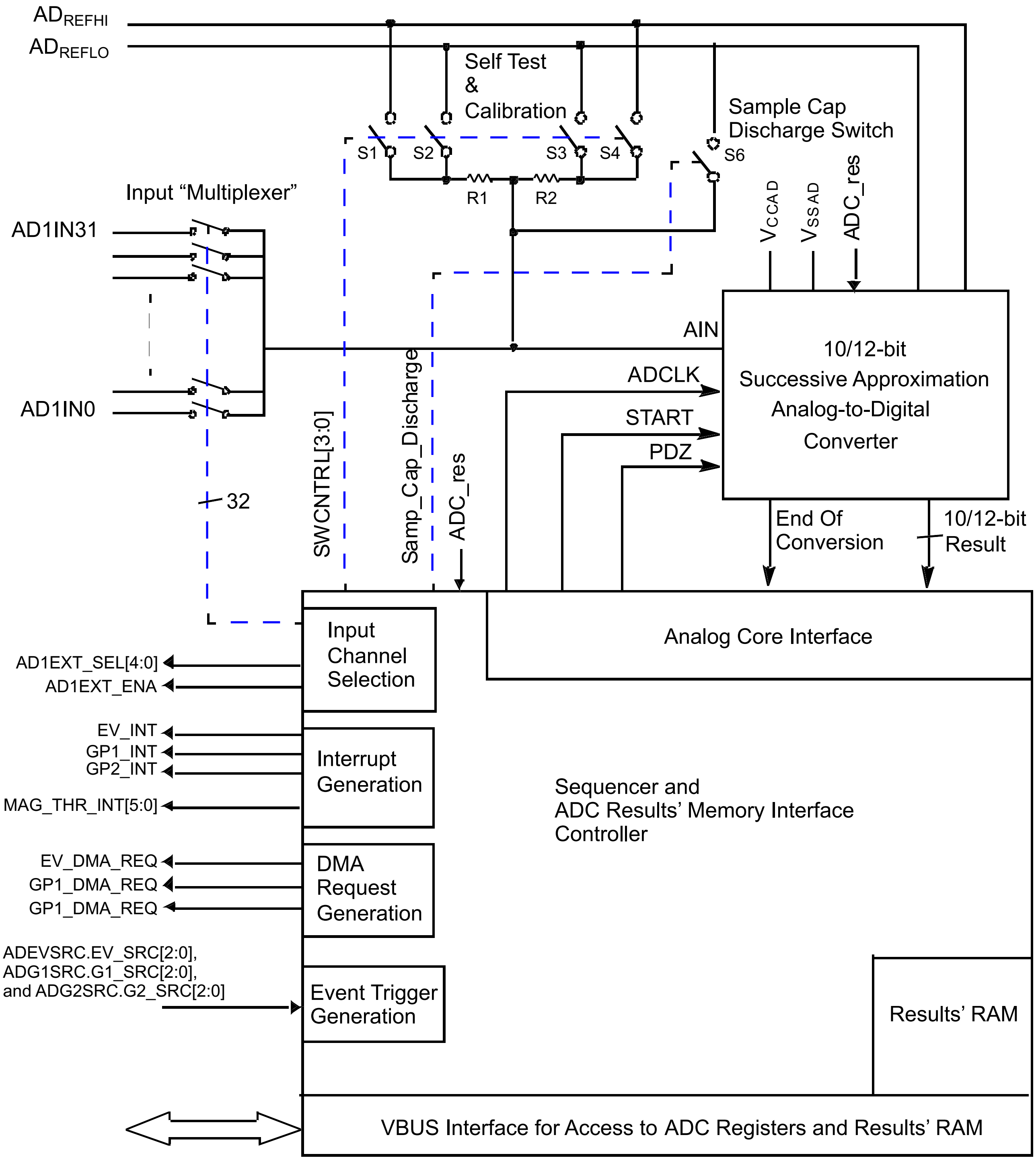
Figure 22-1. Channel Assignments of Two ADC Cores



22.1.1 Introduction

This section presents a brief functional description of the analog-to-digital converter (ADC) module. Figure 22-2 shows the components of the ADC module.

Figure 22-2. ADC Block Diagram



22.1.1.1 Input Multiplexor

The input multiplexor (MUX) connects the selected input channel to the AIN input of the ADC core. The ADC1 module supports up to 32 inputs as shown in Figure 22-2. The ADC2 module supports up to 25 inputs. The sequencer selects the channel to be converted. Enabling the enhanced channel selection mode also allows one or more of the analog input channels to be connected to the output of an external analog switch or multiplexor.

22.1.1.2 Self-Test and Calibration Cell

The ADC includes specific hardware that allows a software algorithm to detect open/short on an ADC analog input. It also allows the application program to calibrate the ADC. Also see Section 22.2.6.1 and Section 22.2.6.2.

22.1.1.3 Analog-to-Digital Converter Core

The ADC core is a combination voltage scaling, charge redistribution Successive Approximation Register (SAR) based analog-to-digital converter. The core can be configured for operation in 10-bit resolution (default) or 12-bit resolution. This is controlled by the sequencer logic. This selection applies to all conversions performed by the ADC module. It is not possible to convert some channels with a 12-bit resolution and some with a 10-bit resolution.

A single conversion from an analog input to a digital conversion result occurs in two distinct periods:

- Sampling Period:
  - The sequencer generates a START signal to the ADC core to signal the start of the sampling period.
  - The analog input signal is sampled directly on to the switched capacitor array during this period, providing an inherent sample-and-hold function.
  - The sampling period ends one full ADCLK after the falling edge of the START signal.
  - The sequencer can control the sampling period duration by configuring the conversion group's sample time control register (ADEVSAMP, ADG1SAMP, ADG2SAMP). This register controls the time for which the START signal stays high.
- Conversion Period:
  - The conversion period starts one full ADCLK after the falling edge of START.
  - One bit of the conversion result is output on each rising edge of ADCLK in the conversion period, starting with the most-significant bit first.
  - The conversion period is 12 ADCLK cycles in case of a 12-bit ADC, and is 10 ADCLK cycles in case of a 10-bit ADC.
  - The ADC core generates an End-Of-Conversion (EOC) signal to the sequencer at the end of the conversion period. At this time the complete 12-, or 10-bit conversion result is available.
  - The sequencer captures the ADC core conversion result output as soon as EOC is driven High.

The analog conversion range is determined by the reference voltages: AD_{REFHI} and AD_{REFLO}. AD_{REFHI} is the top reference voltage and is the maximum analog voltage that can be converted. An analog input voltage equal to AD_{REFHI} or higher results in an output code of 0x3FF for 10-bit resolution and 0xFFF for 12-bit resolution. AD_{REFLO} is the bottom reference voltage and is the minimum analog voltage that can be converted. Applying an input voltage equal to AD_{REFLO} or lower results in an output code of 0x000. Both AD_{REFHI} and AD_{REFLO} must be chosen not to exceed the analog power supplies: V_{CCAD} and V_{SSAD}, respectively. Input voltages between AD_{REFHI} and AD_{REFLO} produce a conversion result given by Equation 27 for 10-bit resolution and by Equation 28 for 12-bit resolution.

$$DigitalResult = \frac{1024 \ x \ (InputVoltage - AD_{REFLO})}{AD_{REFHI} - AD_{REFLO}} - 0.5 \tag{27}$$

$$DigitalResult = \frac{4096 \ x (InputVoltage - AD_{REFLO})}{(AD_{REFHI} - AD_{REFLO})} - 0.5 \tag{28}$$

22.1.1.4 Sequencer

The sequencer coordinates the operations of the ADC, including the input multiplexor, the ADC core, and the result memory. In addition, the logic of the sequencer sets the status register flags when the conversion is ongoing, stopped, or finished.

All the features of the sequencer are discussed in detail in the following sections of this document.

22.1.1.5 Conversion Groups

Several applications require groups of channels to be converted using a single trigger source for example. There could also be some groups of channels identified which require a specific setting of the acquisition time. The ADC module supports three conversion groups for this purpose – Group1, Group2 and the Event Group.

Any of the available analog input channels can be assigned to any of the conversion groups. This also allows a particular channel to be repeatedly sampled by selecting it in multiple groups. There is an inherent priority scheme used when multiple conversion groups are triggered at once. The Event Group is the highest-priority, followed by the Group1 and then the Group2.

The Event Group is always hardware event-triggered. Group1 and Group2 are software-triggered by default and can be configured to be hardware-, or event-triggered as well. The triggering of conversions in each group is discussed in Section 22.2.1.6.

Each conversion group has a separate set of control registers to:

- Select the input channels to be converted
- Configure the mode of conversion: single conversion sequence or continuous conversions
- Configure the input channel sampling time
- Configure the interrupt and/or DMA request generation conditions

22.2 Basic Operation

22.2.1 Basic Features and Usage of the ADC

This section describes the usage of the basic features of the ADC module.

22.2.1.1 How to Select Between 12-bit and 10-bit Resolutions

The 10_12_BIT field of the ADC Operating Mode Control Register (ADOPMODECR) configures the ADC to be in 10-bit or 12-bit resolution mode:

- If 10_12_BIT = 0, the module is in 10-bit resolution mode. This is the default mode of operation.
- If 10_12_BIT = 1, the module is in 12-bit resolution mode.

22.2.1.2 How to Set Up the ADCLK Speed

The ADC sequencer generates the clock for the ADC core, ADCLK. The ADC core uses the ADCLK signal for its timing. The ADCLK is generated by dividing down the input clock to the ADC module, which is the VBUSP interface clock, VCLK. A 5-bit field (PS) in the ADC Clock Control Register (ADCLOCKCR) is used to divide down the VCLK by 1 up to 32. The ADCLK valid frequency range is specified in the device datasheet.

$f_{ADCLK} = f_{VCLK} / (PS + 1)$

The maximum frequency for ADCLK is specified in the device datasheet.

22.2.1.3 How to Set Up the Input Channel Acquisition Time

The signal acquisition time for each group is separately configurable using the ADG1SAMP[11:0], ADG2SAMP[11:0], and ADEVSAMP[11:0] registers.

The acquisition time is specified in terms of ADCLK cycles and ranges from a minimum of 2 ADCLK cycles to a maximum of 4098 ADCLK cycles.

For example, Group1 acquisition time, t_{ACQG1} = G1SAMP[11:0] + 2, in ADCLK cycles.

The minimum acquisition time is specified in the device datasheet. This time also depends on the impedance of the circuit connected to the analog input channel being converted. See the *ADC Source Impedance for Hercules™ ARM® Safety MCUs Application Report* (SPNA118).

22.2.1.4 How to Select an Input Channel for Conversion

The ADC module needs to be enabled first before selecting an input channel for conversion. The ADC module can be enabled by setting the ADC_EN bit in the ADC Operating Mode Control Register (ADOPMODECR). Multiple input channels can be selected for conversion in each group. Only one input channel is converted at a time. The channels to be converted are configured in one or more of the three conversion groups' channel selection registers. Channels to be converted in Group1 are configured in the Group1 Channel-Select Register (ADG1SEL), those to be converted in Group2 are configured in the Group2 Channel-Select Register (ADG2SEL), and those to be converted in the Event Group are configured in the Event Group Channel-Select Register (ADEVSEL).

The description in this section only refers to the case when the enhanced channel selection mode is not enabled. Input channel selection in the enhanced channel selection mode is defined in Section 22.2.2.

22.2.1.5 How to Select Between Single Conversion Sequence or Continuous Conversions

Each group has its own mode control register. The MODE field of these control registers allow the application to select between a single conversion sequence or continuous conversion mode.

NOTE: Selecting continuous conversion mode for all three groups

All three conversion groups cannot be configured to be in a continuous conversion mode. If the application configures the group mode control registers to enable continuous conversion mode for all three groups, then the Group2 will be automatically be configured to be in a single conversion sequence mode.

With conversions ongoing in continuous conversion mode, if the MODE field of a group is cleared, then that group switches to the single conversion sequence mode. Conversions for this group will stop once all channels selected for that group have been converted.

22.2.1.6 How to Start a Conversion

The conversion groups Group1 and Group2 are software-triggered by default. A conversion in these groups can be started just by writing the desired channels to the respective Channel-Select Registers. For example, in order to convert channels 0, 1, 2, and 3 in Group1 and channels 8, 9, 10, and 11 in Group2, the application just has to write 0x0000000F to ADG1SEL and 0x00000F00 to ADG2SEL. The ADC module will start by servicing the group that was triggered first, Group1 in this example.

The conversions for all groups are performed in ascending order of the channel number. For the Group1 the conversions will be performed in the order: channel 0 first, followed by channel 1, then channel 2, and then channel 3. The Group2 conversions will be performed in the order: channels 8, 9, 10, and 11.

The Event Group is only hardware-triggered. There are up to eight hardware event trigger sources defined for the ADC module. Check the device datasheet for a complete listing of these eight hardware trigger options.

The trigger source to be used needs to be configured in the ADEVSRC register. Similar registers also exist for the Group1 and Group2 as these can also be configured to be event-triggered.

The polarity of the event trigger is also configurable, with a falling edge being the default.

An Event Group conversion starts when at least one channel is selected for conversion in this group, and when the defined event trigger occurs.

If any conversion group is configured to be in a continuous conversion mode, then it needs to only be triggered once. All the channels selected for conversion in that group will be converted repeatedly.

22.2.1.7 How to Know When the Group Conversion is Completed

Each conversion group has a status flag to indicate when its conversion has ended. See ADEVSR, ADG1SR, and ADG2SR. This bit is set when a conversion sequence for a group ends. This bit does is always set if a group is configured for continuous conversions.

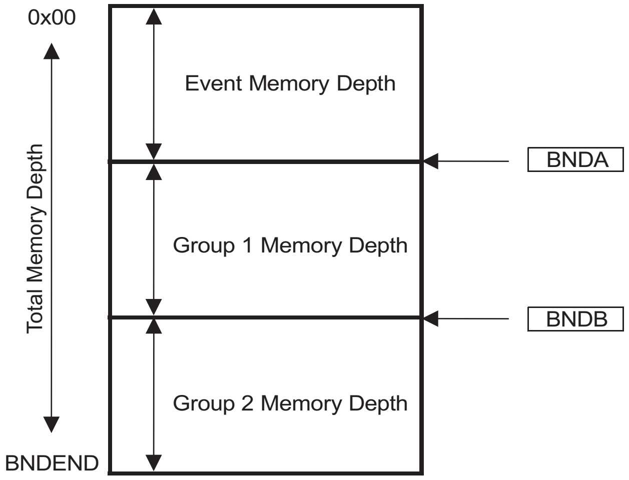
22.2.1.8 How Results are Stored in the Results' Memory

The ADC stores the conversion results in three separate memory regions in the ADC Results' RAM, one region for each group. Each memory region is a stack of buffers, with each buffer capable of holding one conversion result. The number of buffers allocated for each group is programmed by configuring the ADC module registers ADBNDCR and ADBNDEND.

ADBNDCR contains two 9-bit pointers BNDA and BNDB. BNDA, BNDB, and BNDEND are used to partition the total memory available into three memory regions as shown in Figure 22-3. Both BNDA and BNDB are pointers referenced from the start of the results' memory. BNDA specifies the number of buffers allocated for the Event Group conversion results in units of two buffers; BNDB specifies the number of buffers allocated for the Event Group plus Group1 in units of two buffers. Refer to Section 22.3.23 for more details on configuring the ADC results' memory.

ADBNDEND contains a 3-bit field called BNDEND that configures the total memory available. The ADC module can support up to 1024 buffers. The device supports a maximum of 64 buffers for both the ADC modules.

Figure 22-3. FIFO Implementation



- Number of buffers for Event Group = 2 × BNDA
- Number of buffers for Group1 = 2 × (BNDB – BNDA)
- Number of buffers for Group2 = Total number of buffers – 2 × BNDB

22.2.1.9 How to Read the Results from the Results' Memory

The CPU can read the conversion results in one of two ways:

1. By using the conversion results memory as a FIFO queue
2. By accessing the conversion results memory directly

22.2.1.9.1 Reading Conversion Results from a FIFO

The conversion results for each group can be accessed via a range of addresses provided to facilitate the use of the ARM Cortex-R4 CPU’s Load-Multiple (LDM) instruction. A single read performed using the LDR instruction can also be used to read out a single conversion result. The results are read out from the group’s memory region as a FIFO queue by reading from any location inside this address range. The conversion result that got stored first gets read first. A result that is read from the memory in this method is removed from the memory. For example, a read from any address in the range ADEVBUFFER (offset 90h to AFh) pulls out one conversion result from the Event Group memory.

Figure 22-4. Format of Conversion Result Read from FIFO, 12-bit ADC

<table>
<tr><th>Offset Address Register</th><th>31
15</th><th>30
14</th><th>29
13</th><th>28
12</th><th>27
11</th><th>26
10</th><th>25
9</th><th>24
8</th><th>23
7</th><th>22
6</th><th>21
5</th><th>20
4</th><th>19
3</th><th>18
2</th><th>17
1</th><th>16
0</th></tr>
<tr><td rowspan="2">0x90 to 0xAF
ADEVBUFFER</td><td>EV_
EMPTY</td><td colspan="10">Reserved</td><td colspan="5">EV_CHID</td></tr>
<tr><td colspan="4">Reserved</td><td colspan="12">EV_DR</td></tr>
<tr><td rowspan="2">0xB0 to 0xCF
ADG1BUFFER</td><td>G1_
EMPTY</td><td colspan="10">Reserved</td><td colspan="5">G1_CHID</td></tr>
<tr><td colspan="4">Reserved</td><td colspan="12">G1_DR</td></tr>
<tr><td rowspan="2">0xD0 to 0xEF
ADG2BUFFER</td><td>G2_
EMPTY</td><td colspan="10">Reserved</td><td colspan="5">G2_CHID</td></tr>
<tr><td colspan="4">Reserved</td><td colspan="12">G2_DR</td></tr>
</table>

Figure 22-5. Format of Conversion Result Read from FIFO, 10-bit ADC

<table>
<tr><th>Offset Address Register</th><th>31
15</th><th>30
14</th><th>29
13</th><th>28
12</th><th>27
11</th><th>26
10</th><th>25
9</th><th>24
8</th><th>23
7</th><th>22
6</th><th>21
5</th><th>20
4</th><th>19
3</th><th>18
2</th><th>17
1</th><th>16
0</th></tr>
<tr><td rowspan="2">0x90 to 0xAF
ADEVBUFFER</td><td colspan="16">Reserved</td></tr>
<tr><td>EV_
EMPTY</td><td colspan="5">EV_CHID</td><td colspan="10">EV_DR</td></tr>
<tr><td rowspan="2">0xB0 to 0xCF
ADG1BUFFER</td><td colspan="16">Reserved</td></tr>
<tr><td>G1_
EMPTY</td><td colspan="5">G1_CHID</td><td colspan="10">G1_DR</td></tr>
<tr><td rowspan="2">0xD0 to 0xEF
ADG2BUFFER</td><td colspan="16">Reserved</td></tr>
<tr><td>G2_
EMPTY</td><td colspan="5">G2_CHID</td><td colspan="10">G2_DR</td></tr>
</table>

Option to read channel id along with conversion result:

The application has an option to read the channel id along with the conversion result. This is controlled by the CHID field of the group’s mode control register. If the option to read the channel id is not selected, the channel id field of the conversion result reads as zeros.

Protection against reading from empty FIFO:

There is also a hardware mechanism to protect the application from reading past the number of new conversion results held in the FIFO. Once all available conversion results have been read out of the FIFO by the application, a subsequent read from the FIFO causes the mechanism to indicate that the FIFO is empty by setting the EMPTY field.

Debug / Emulation Support:

For debug purposes, each conversion group also provides an address that the application can read from for extracting the group’s conversion results. However, no status flags for a conversion group are affected by reading from these emulation buffer addresses. For example, reading from ADEVEMUBUFFER (offset F0h) returns the next result in the Event Group buffer but does not actually remove that result from the buffer or change the amount of data held in the buffer.

22.2.1.9.2 Reading Conversion Results Directly from the Conversion Results' Memory

The conversion result memory is part of the device's memory map. The base address for the ADC1 result memory is FF3E 0000h and for the ADC2 result memory is FF3A 0000h.

Figure 22-6. ADC Memory Mapping

ADC1	ADC2	
0xFF3E0000	0xFF3A0000	Conversion word 0
0xFF3E0004	0xFF3A0004	Conversion word 1
0xFF3E0008	0xFF3A0008	Conversion word 2
⋮	⋮	⋮
0xFF3E01F8	0xFF3A01F8	Conversion word 62
0xFF3E00FC	0xFF3A00FC	Conversion word 63

The application can identify the address ranges for each of the three memory regions for the three conversion groups after performing the segmentation as described in Section 22.2.1.8. It is up to the application to read the desired results from the three conversion groups. The formats of the conversion results when reading from RAM directly are shown in Figure 22-7 and Figure 22-8.

Figure 22-7. Format of Conversion Result Directly Read from ADC RAM, 12-bit ADC

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADC RAM address	Reserved															channel id [4]
	channel id [3–0]				12-bit conversion result											

Figure 22-8. Format of Conversion Result Directly Read from ADC RAM, 10-bit ADC

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADC RAM address	Reserved															
	Rsvd	channel id [4–0]					10-bit conversion result									

Note that there is no EMPTY field to protect the application from reading data that has been previously read.

Each group does have a separate register which holds the address in the group's result memory where the ADC will write the next conversion result. These are the ADEVRAMWRADDR, ADG1RAMWRADDR, and ADG2RAMWRADDR registers. The application can use this information to calculate how many valid conversion results are available to be read.

Benefit of reading conversion results directly from ADC RAM:

The application does not have to read out conversion results sequentially as in the case of reading from a FIFO. As a result, the application can selectively read the conversion results for any particular input channel of interest without having to read other channels' conversion results.

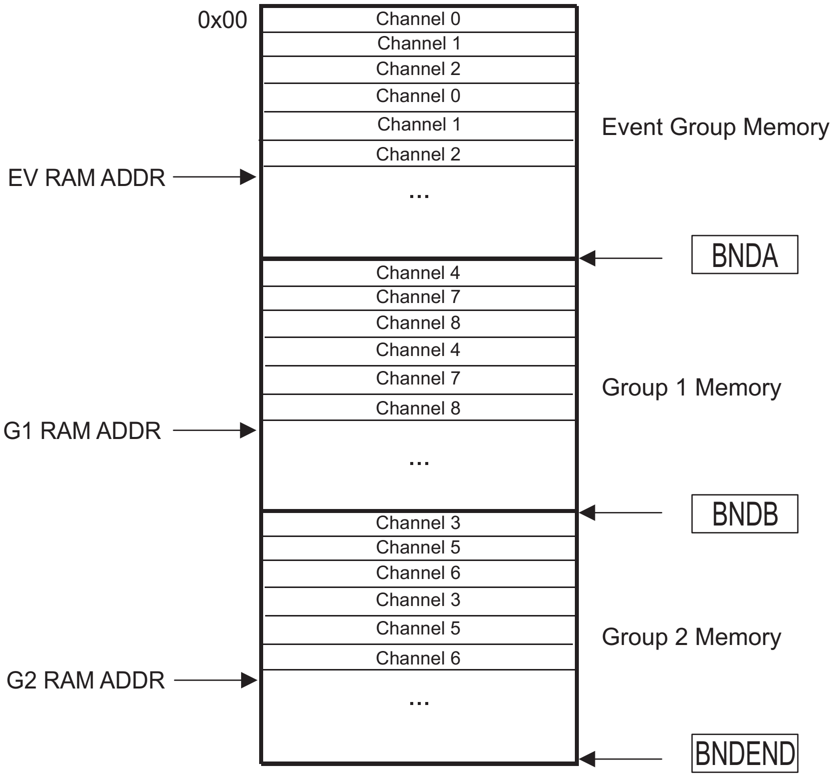
22.2.1.9.3 Example

Suppose that channels 0, 1, and 2 are selected for conversion in the Event Group, channels 4, 7, and 8 are selected for conversion in group 1, and channels 3, 5, and 6 are selected for conversion in group 2. The conversion results will get stored in the three memory regions as shown in Figure 22-9.

Suppose that the CPU wants to read out the results for the Event Group from a FIFO queue. The CPU needs to read from any address in the range ADEVBUFFER (offset 90h to AFh) multiple times, or do a “load multiple” from this range of addresses. This will cause the ADC to return the results for channel 0, then channel 1, then channel 2, then channel 0, and so on for each read access to this address range.

Now suppose that the application wants to read out the results for the group 1 from the RAM directly. The conversion results for the group 1 are accessible starting from address ADC RAM Base Address + BNDA. Also, it is known that the first result at this address is for the input channel 4, the next one is for input channel 7, and so on. So the application can selectively read the conversion results for only one channel if so desired.

Figure 22-9. Conversion Results Storage



22.2.1.10 How to Stop a Conversion

A group's conversion can be stopped by clearing the group's channel select register.

22.2.1.11 Example Sequence for Basic Configuration of ADC Module

The following sequence is necessary to configure the ADC to convert channels 0, 2, 4, and 8 in single-conversion mode using Group1:

1. Write 0 to the Reset Control Register (ADRSTCR) to release the module from the reset state
2. Write 1 to the ADC_EN bit of the Operating Mode Control Register (ADOPMODECR) to enable the ADC state machine
3. Configure the ADCLK frequency by programming the desired divider into the Clock Control Register (ADCLOCKCR)
4. Configure the acquisition time for the group that is to be used. For example, configure the Group1 Sampling Time Control Register (ADG1SAMP) to set the acquisition time for Group1.
5. Select the channels that need to be converted in Group1 by writing to the Group1 Channel Select Register (ADG1SEL). In this example, a value of 0x115 needs to be written to ADG1SEL in order to select channels 0, 2, 4, and 8 for conversion in Group1.
   - The ADC sequencer will start the Group1 conversions as soon as the write to the ADG1SEL register is completed.
6. Wait for the GP1_END bit to be set in the Group1 Conversion Status Register (ADG1SR). This bit gets set when all the channels selected for conversion in Group1 are converted and the results are stored in the Group1 memory.
7. Read the conversion results by reading from the Group1 FIFO access location (ADG1BUFFER) or by reading directly from the Group1 results' memory.

22.2.2 Advanced Conversion Group Configuration Options

Figure 22-10 shows the operating mode control registers and the status registers for each of the three conversion groups. The register addresses shown are offsets from the base address. The ADC1 register frame base address is FFF7 C000h and the ADC2 register frame base address is FFF7 C200h.

Figure 22-10. ADC Groups' Operating Mode Control and Status Registers

Offset Address Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x010 ADEVMODECR	Reserved															No Reset On ChnSel
	Reserved						EV_DATA_FMT		Reserved		EV_CHID	OVR_EV_RAM_IGN	Rsvd	EV_8BIT	EV_MODE	FRZ_EV
0x014 ADG1MODECR	Reserved															No Reset On ChnSel
	Reserved						G1_DATA_FMT		Reserved		G1_CHID	OVR_G1_RAM_IGN	Rsvd	G1_8BIT	G1_MODE	FRZ_G1
0x018 ADG2MODECR	Reserved															No Reset On ChnSel
	Reserved						G2_DATA_FMT		Reserved		G2_CHID	OVR_G2_RAM_IGN	Rsvd	G2_8BIT	G2_MODE	FRZ_G2
0x06C ADEVSR	Reserved															
	Reserved												EV_MEM_EMPTY	EV_BUSY	EV_STOP	EV_END
0x070 ADG1SR	Reserved															
	Reserved												G1_MEM_EMPTY	G1_BUSY	G1_STOP	G1_END
0x074 ADG2SR	Reserved															
	Reserved												G2_MEM_EMPTY	G2_BUSY	G2_STOP	G2_END
0x19C ADEVCURRCOUNT	Reserved															
	Reserved											EV_CURRENT_COUNT				
0x1A0 ADEVMAXCOUNT	Reserved															
	Reserved											EV_MAX_COUNT				
0x1A4 ADG1CURRCOUNT	Reserved															
	Reserved											G1_CURRENT_COUNT				
0x1A8 ADG1MAXCOUNT	Reserved															
	Reserved											G1_MAX_COUNT				
0x1AC ADG2CURRCOUNT	Reserved															
	Reserved											G2_CURRENT_COUNT				
0x1B0 ADG2MAXCOUNT	Reserved															
	Reserved											G2_MAX_COUNT				

22.2.2.1 Group Trigger Options

The Group1 and Group2 operating mode control registers have an extra control bit: HW_TRIG. This bit configures the group to be hardware event-triggered instead of software-triggered, which is the default.

When a group is configured to be event-triggered, the group conversion starts when at least one channel is selected for conversion in this group, and when the defined event trigger occurs. The event trigger source is defined for each group in the ADEVSRC, ADG1SRC, and the ADG2SRC registers. The actual connections used as the event trigger sources are defined in the device datasheet for both the ADC modules.

22.2.2.2 Analog Input Channel Selection Mode Options

The ADC1 module on this device supports two different modes for selecting the analog input channel to be converted:

- Sequential channel selection mode (default)
- Enhanced channel selection mode

NOTE: ADC2 module only supports the sequential channel selection mode (the default).

22.2.2.2.1 Sequential Channel Selection Mode

This is the default mode and allows the ADC module to be used in a backwards compatible mode to the ADC module on other Hercules™ ARM® Safety MCUs from Texas Instruments. As discussed in Section 22.2.1.4, an analog input channel can be selected for conversion in one or more conversion groups by setting the bit corresponding to that channel number in the group's channel select register.

22.2.2.2.2 Enhanced Channel Selection Mode

There are some important concepts related to the enhanced channel selection mode. These are defined first:

- Look-Up Table

  This is a 32-word deep memory-mapped region used to define the analog input channel number to be converted. The LUTs for the three groups are stacked together so that the entire LUT occupies 96 words. Each word is aligned on a 32-bit boundary. The LUTs for ADC1 start at FF3E 2000h and the LUTs for ADC2 start at FF3A 2000h.
- Conversion Group Sub-Sequence

  A group sub-sequence is defined as the conversion for a set of channels that is converted on each conversion trigger. The number of channels selected for conversion in a group sub-sequence is defined by the number of bits that are set in the group's channel select register. For example, setting bits 0, 1, 29 and 31 in ADG1SEL means that each Group1 conversion sub-sequence consists of 4 conversions.
- LUT Index

  A "CURRENT_COUNT" register for each group is maintained as an index into that group's LUT. This register increments each time a channel conversion is completed. Therefore, as its name suggests, a read from this register returns the number of conversions completed since the last write to the group's channel select register. The CURRENT_COUNT register resets to all zeros under any of the following conditions:

  1. The ADC peripheral is reset via a global peripheral reset
  2. The ADC peripheral is reset via the ADC Reset Control Register
  3. The CURRENT_COUNT becomes equal to the MAX_COUNT defined for that conversion group
  4. The application writes zeros to the CURRENT_COUNT register
  5. The conversion group's result RAM is reset

- Maximum Number of Conversions

  A MAX_COUNT register for each conversion group stores the maximum number of conversions to be performed before the index into a group's LUT is reset to 0. This register can be programmed to a value between 0 and 31. It is recommended to program the MAX_COUNT register with a value that is one less than a multiple of the number of channels in that group's conversion sub-sequence (number of bits that are set in the group's channel select register).

22.2.2.2.2.1 Look-Up Table Details

As described earlier, each conversion group has a look-up table (LUT) which is used when the enhanced channel selection mode is enabled. This look-up table starts at an offset of 8kB from the base of the ADC results RAM. The LUT holds 32 entries for each of the three conversion groups. The first 32 entries are for the event group, the next 32 entries are for Group1 and the last 32 entries are for Group2. Figure 22-11 shows an example LUT entry for the Event group.

Figure 22-11. Example Look-Up Table Entry

31 16
Reserved
R-0

15 13	12 8	7 5	4 0
Reserved	EV_EXT_CHN_MUX_SEL	Reserved	EV_INT_CHN_MUX_SEL
R-0	R/W-0	R-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 22-1. ADC Look-Up Table Field Descriptions

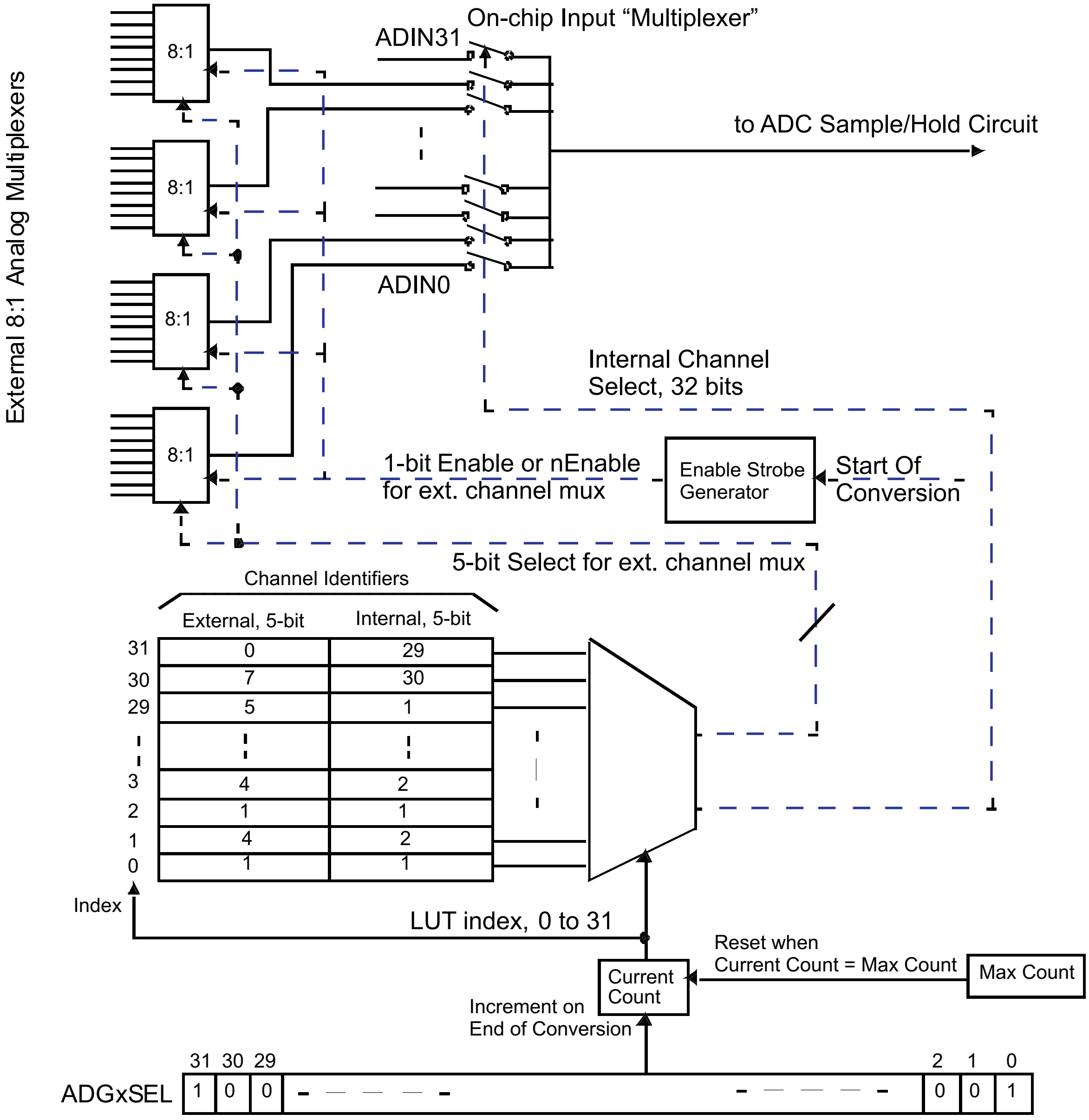
Bit	Field	Value	Description
31–13	Reserved	0	Reads return 0. Writes have no effect.
12–8	EV_EXT_CHN_MUX_SEL		This field defines the external analog mux select that is output from the ADC module when the Event group CURRENT_COUNT register points to this LUT entry, and when the Event group conversion is triggered with the enhanced channel selection mode enabled.
7–5	Reserved	0	Reads return 0. Writes have no effect.
4–0	EV_INT_CHN_MUX_SEL		This field defines the internal analog mux select that is output from the ADC module when the Event group CURRENT_COUNT register points to this LUT entry, and when the Event group conversion is triggered with the enhanced channel selection mode enabled. This can be a value between 0 and 31, which corresponds to the internal analog input channel number between 0 and 31. Note that this device only supports 24 input channels for ADC1 and 16 input channels for ADC2. If the application configures an unavailable channel number in the EV_INT_CHN_MUX_SEL field, the ADC will still perform the conversion and the result will be indeterminate.

22.2.2.2.2.2 Example ADC Conversion Sequence Using Enhanced Channel Selection Mode

Consider the example conversion Group1 configuration shown in Figure 22-12. Only bits 0 and 31 of ADG1SEL are set. Assume that all other bits in this register are zeros.

In case of the default sequential channel selection mode, the write to the ADG1SEL register would cause the Group1 conversions to start with channel 0 followed by channel 31. The conversions would then stop or repeat in this order depending on whether Group1 is in single or continuous conversion mode.

Figure 22-12. Group1 Enhanced Channel Selection Mode Example



Index	External, 5-bit	Internal, 5-bit
31	0	29
30	7	30
29	5	1
⋮	⋮	⋮
3	4	2
2	1	1
1	4	2
0	1	1

Now suppose that the application has enabled the enhanced channel selection mode for Group1 with the G1_MAX_COUNT register configured to be 3. Also suppose that the application has programmed the Group1 LUT as shown in Figure 22-12. Now suppose that the application triggers Group1 conversions by writing 0x80000001 to ADG1SEL, that is, bits 0 and 31 are set and all others are zeros. The ADC conversions will proceed in the following sequence:

- Input Channel Selection

  The initial value of G1_CURRENT_COUNT is 0, which is used as the index into the Group1 LUT. The row 0 of Group1's LUT has values of 1 for the G1_EXT_CHN_MUX_SEL and a value of 1 for the G1_INT_CHN_MUX_SEL. The 5-bit external channel id of 0b00001 is driven out on the AD1EXT_SEL terminals. This selects channel 1 for all the connected external analog multiplexors, as shown in Figure 22-12. The ADC module also outputs an enable signal to the external analog multiplexors via the AD1EXT_ENA terminal.

  Now consider the fact that the internal channel id is also configured to be 1 in row 0 of the Group1 LUT. This causes the switch for ADC's internal channel 1 (ADIN1) to be closed. All other internal ADC input switches (ADIN0, ADIN2, ADIN3, ..., ADIN31) will be open. Note that the ADIN1 input channel is actually connected to the output of an 8:1 analog multiplexor.

  In effect, the ADC will convert channel 1 of the 8:1 analog multiplexor connected to the ADIN1 terminal of the microcontroller.

- After Completion of Conversion

  Once the first conversion is completed, the CURRENT_COUNT value of 0 is stored in the "channel id" field of the conversion result RAM of Group1 along with the actual conversion result from the ADC core. Then the G1_CURRENT_COUNT value of 0 is compared against the G1_MAX_COUNT value of 3. The values do not match, so that G1_CURRENT_COUNT is incremented from 0 to 1.

- Next Channel Selection

  There are two bits set in the ADG1SEL register, so that the ADC module now uses the G1_CURRENT_COUNT value of 1 to index the Group1 LUT. As shown in Figure 22-12, this row in Group1 LUT contains 4 as the G1_EXT_CHN_MUX_SEL and 2 as the G1_INT_CHN_MUX_SEL.

  ADC input channel ADIN2 is not connected to any external analog multiplexor and is connected directly to the analog signal to be converted. Note that the ADC module still drives the AD1EXT_ENA and the AD1EXT_SEL (value of 4, that is, 0b00100) to all the external analog multiplexors connected to the microcontroller.

- End of Conversion Sub-Sequence

  Once the conversion of the internal channel ADIN2 is completed, the G1_CURRENT_COUNT of 1 is stored in the "channel id" field of the Group1 result RAM along with the actual conversion result. This value of 1 is compared against the G1_MAX_COUNT value of 3. The values do not match, so that G1_CURRENT_COUNT is incremented from 1 to 2.

  There are no more conversions required in this sub-sequence as only two bits are set in ADG1SEL.

- Continuation on Next Group1 Trigger

  When the ADC Group1 is triggered again or if Group1 is in continuous conversion mode, the G1_CURRENT_COUNT of 2 is again used to index the Group1 LUT. Following the same reasoning as before, this will cause the channel 1 of the 8:1 analog multiplexor connected to ADIN1 to be converted.

  Once this conversion is done, the G1_CURRENT_COUNT value of 2 is stored in the "channel id" field of the result RAM along with the conversion result. This still does not match the G1_MAX_COUNT of 3, so that G1_CURRENT_COUNT is now incremented from 2 to 3.

  This index value of 3 is used to again convert channel ADIN2, following the same reasoning as before.

  When this conversion is completed, the G1_CURRENT_COUNT of 3 is stored as the "channel id" field of the result RAM along with the conversion result.

  Also, now this G1_CURRENT_COUNT value of 3 matches the G1_MAX_COUNT. This resets the G1_CURRENT_COUNT to 0.

The sequence proceeds as described whenever Group1 is next triggered, or if Group1 is configured to be in a continuous conversion mode.

22.2.2.3 Single or Continuous Conversion Modes

The EV_MODE, G1_MODE, and G2_MODE bits are used to select between either single or continuous conversion mode for each of the three groups.

22.2.2.3.1 Single Conversion Mode

A conversion group configured to be in single-conversion mode gets serviced only once by the ADC for each group trigger. The trigger can be a software trigger as in the case of Group1 and Group2 by default, or it could be a hardware event trigger as in the case of the Event Group or Group1 or Group2.

The entire conversion sequence, from the acceptance of the group conversion request to the end of the last channel's conversion, is flagged for each group by the corresponding BUSY bit in that group's status register. After single-conversion mode is started, the BUSY bit is read as 1 until the conversion of the last channel is complete. The END bit for the group is set once all the channels in that group are converted.

For example, say channels 0, 2, 4, and 6 are selected for conversion in Group1 in single-conversion mode. When the Group1 gets serviced, the ADC will start conversion for channel 0, then channel 2, then channel 4, and then channel 6. It will then stop servicing the Group1, set the GP1_END status bit, and look to service the Event Group or the Group2, if required.

22.2.2.3.2 Continuous Conversion Mode

A conversion group configured to be in continuous-conversion mode gets serviced by the ADC continuously. The group still needs to be triggered appropriately for the first conversion to start. The conversions are performed continuously thereafter.

The entire conversion sequence, from the acceptance of the group conversion request to the end of the last channel's conversion, is flagged for each group by the corresponding BUSY bit in that group's status register. After continuous-conversion mode is started, the BUSY bit is read as 1 as long as the continuous-conversion mode for this group is selected.

As an example, say the channels 0, 2, 4, and 6 are selected for conversion in Group1, now in continuous-conversion mode. When the Group1 gets serviced, the ADC will complete conversions for channels 0, 2, 4 and 6, and then look to service the Event Group or the Group2. Once it is done servicing the Event Group or the Group2, it will return to service the Group1 again. The Group1 does not need to be triggered again for the repeated conversion.

NOTE: Configuring all conversion groups in continuous conversion mode

All the three groups cannot operate in continuous-conversion mode at the same time. If the application program configures all three groups to be in continuous-conversion mode, the Group2 is automatically reset to single-conversion mode, and the G2 MODE bit in the ADG2MODECR register is cleared to reflect the single-conversion mode of Group2.

22.2.2.4 Conversion Group Freeze Capability

The ADC module has an inherent priority order between the three conversion groups. This group priority determines the order of conversion in case multiple groups are triggered. The priority of conversions between the three groups in descending order is:

1. Event Group
2. Group1
3. Group2

Examples of conversion group priority:

- If an Event Group conversion is ongoing in single conversion sequence mode and Group2 and Group1 conversions are requested, then the ADC will finish conversion of channels selected in Event Group, then switch over to converting channels selected in Group1, and then convert channels selected in Group2.
- If Group1 conversions are ongoing in continuous conversion mode and Group2 conversion is requested, then the ADC will complete converting the current channel for Group1 and switch over to converting channels selected in Group2. The new conversion request for Group2 has a higher priority than the pending continuous conversion request for Group1.

The conversion group freeze capability allows the application to override this default priority between the conversion groups. Enabling the freeze capability allows the ADC to freeze a higher-priority conversion group's conversions whenever there is a request for conversion in another (lower-priority) group.

For example, setting the FRZ_EV bit in the ADEVMODECR register will allow the ADC to freeze ongoing Event Group conversions whenever there is a pending request, or a new request for a Group1 or Group2 conversion. The conversions for the Event Group will be frozen as long as the Group1 or Group2 conversions are active. Once the Group1 or Group2 conversions are completed, the Event Group conversions start from where they were frozen.

While a group's conversions are frozen, the group's STOP status bit is set. This bit is cleared once the group's conversions are restarted.

22.2.2.5 Conversion Group Memory Overrun Option

An overrun condition occurs when the ADC module tries to store more conversion results to a group's results' memory which is already full. In this case, the ADC allows two options.

If the OVR_RAM_IGN bit in the group's operating mode control register (ADEVMODECR, ADG1MODECR, ADG2MODECR) is set, then the ADC module ignores the contents of the group's results' memory and wraps around to overwrite the memory with the results of new conversions.

If the OVR_RAM_IGN bit is not set, then the application program has to read out the group's results' memory upon an overrun condition; only then can the ADC continue to write new results to the memory.

22.2.2.6 Response on Writing Non-Zero Value to Conversion Group's Channel Select Register

If the application writes a non-zero value to a group's channel select register while that group's conversions are already being serviced, then that group's conversions will be restarted with the new configuration programmed in the channel select registers.

The following rules apply in terms of the effect on the ADC conversion sequence:

- If the new conversion request comes from the same group as the ongoing conversion, then the ongoing conversion will be stopped in whichever stage it is in, and the new sequence of conversions will be started.
- If the new conversion request comes from a separate group, then the ongoing channel's conversion will be completed before starting the new sequence of conversions.

The following rules apply in terms of the effect on the group's results memory:

- If a group conversion is ongoing or is frozen, writing a non-zero value to the group's channel select register will also reset its results FIFO. This does not clear the contents of the results FIFO; only the ADC module is allowed to overwrite the FIFO's contents with new conversion results starting from the first location.
- If the group conversion is completed (<GRP>_END flag is set), or the group is not being used, then writing a non-zero value to the group's channel select register will either be reset or not depending on the value of the NoResetOnChnSel bit for that group (ADEVMODECR, ADG1MODECR, ADG2MODECR).
  - If the NoResetOnChnSel bit is 0, then the group's FIFO will be reset.
  - If the NoResetOnChnSel bit is 1, then the group's FIFO will not be reset.

22.2.2.7 Conversion Result Size on Reading: 8-bit, 10-bit, or 12-bit

Some applications do not need the full 12-bit resolution of the ADC modules on the device and can work with 8-bit or 10-bit conversion results.

22.2.2.7.1 ADC Configured in 12-bit Resolution

The mode control register for each conversion group contains a field called DATA_FMT, which defines the format of the conversion result read out of the result RAM, when accessed as a FIFO.

The DATA_FMT field is encoded as follows:

- If DATA_FMT = 0, the complete 12-bit conversion result is read out of the FIFO.
- If DATA_FMT = 1h, the 12-bit conversion result is right-shifted by 2 and the resulting 10-bit result is read out of the FIFO.
- If DATA_FMT = 2h, the 12-bit conversion result is right-shifted by 4 and the resulting 8-bit result is read out of the FIFO.

This control field is not effective when the application chooses to access the conversion result memory directly. In that case, the application can choose to mask off the number of bits as required.

22.2.2.7.2 ADC Configured in 10-bit Resolution

The DATA_FMT field is not effective in this mode and the application has the choice to read either the full 10-bit conversion result or an 8-bit conversion result. This is controlled by the 8BIT field of the group's operating mode control register.

- If 8BIT = 0, the complete 10-bit conversion result is read out of the FIFO.
- If 8BIT = 1, the 10-bit conversion result is right-shifted by 2 and the resulting 8-bit result is read out of the FIFO.

22.2.2.8 Option to Read Group Channel ID Along With Conversion Result

The ADC module allows the application program to also read out the analog input channel number along with its conversion result. This capability is enabled by setting the CHID bit in the group's operating mode control register.

- If CHID = 0, bits [14-10] are forced to 00000 when the conversion results are read out from the group's results' FIFO.
- If CHID = 1, bits [14-10] in the group's results' memory contain the input channel number to which the conversion result belongs.

NOTE: Actual Storage of Channel ID

Regardless of whether the CHID bit is set or not, the channel number is **always stored** in the memory along with the conversion result. The CHID bit only affects whether the channel number is available with the conversion result **when the group's memory is read**. Therefore, the CHID bit for a group can be changed dynamically without affecting that group's ongoing conversions.

22.2.3 ADC Module Basic Interrupts

This section describes the basic interrupts generated by the ADC module.

22.2.3.1 Group Conversion End Interrupt

The ADC module sets the group's conversion end flag (EV_END, G1_END, or G2_END) in that group's interrupt flag register (ADEVINTFLG, ADG1INTFLG, ADG2INTFLG) when all the channels selected for conversion in that group are converted. This causes a group conversion end interrupt to be generated if this interrupt is enabled by setting the group's END_INT_EN control bit (EV_END_INT_EN, G1_END_INT_EN, or G2_END_INT_EN).

This interrupt can be easily used for conversion groups configured to be in the single-conversion mode. The application program can read out the conversion results, change the group's configuration if necessary, and restart the conversions by triggering the group from within the interrupt service routine.

For groups configured to be in continuous conversion mode, this interrupt condition is not practical as the conversions are always in progress. In this case, the Group Memory Threshold Interrupt is more practical as the application can allow a programmable number of conversion results to accumulate before interrupting the CPU.

22.2.3.2 Group Memory Threshold Interrupt

The ADC module has the ability to generate an interrupt for a fixed number of conversions for each group. A group memory threshold register determines how many conversion results must be in a group's memory region before the CPU is interrupted. This feature can be used to significantly reduce the CPU load when using interrupts for reading the conversion results.

The group's threshold register needs to be configured before the group conversions are triggered. This threshold register value behaves like a down-counter, which decrements each time the ADC writes a conversion result to this group's memory. This counter is incremented each time the application program reads a conversion result from the results' memory by accessing the FIFO queue. Simultaneous read (by application program) and write (by ADC module) operations from the group's results' memory leave the threshold counter unchanged.

The threshold counter can decrement past 0 and become negative. It always increments back to its original value when the memory region is emptied. To determine how many samples are in the memory region at a given moment, the threshold counter can be subtracted from the originally configured threshold count.

Whenever the threshold counter transitions from +1 to 0, it sets the group's threshold interrupt flag, and the CPU is interrupted if the group's threshold interrupt is enabled. The CPU is expected to clear the interrupt flag after reading the conversion results from the memory.

The interrupt flag is not set when the threshold counter stays at 0 or transitions from -1 to 0.

22.2.3.3 Group Memory Overrun Interrupt

An interrupt can be generated for each group if the number of ADC conversions for that group exceed the number of buffers allocated for that conversion group. The application program can choose to read out all the conversion results using the CPU or the DMA. Alternatively, the application program can set the group's OVR_RAM_IGN bit and allow the ADC module to overwrite the group's results' memory contents with new conversion results.

22.2.4 ADC Module DMA Requests

This section describes the capabilities of the ADC module to take advantage of the Platform DMA controller module. The ADC module can generate a DMA request under two conditions:

22.2.4.1 DMA Request for Each Conversion Result Written to the Results' Memory

In this mode, the ADC module will generate the first DMA request as soon as a conversion result gets written to the group's results' memory. Subsequent writes to the results' memory will cause DMA requests to be generated. This mode allows a smaller amount of ADC results' memory to suffice for an application.

This DMA request generation is enabled by setting the group's DMA_EN bit in the group's DMA control register. The BLK_XFER bit in this register must be left cleared (default), if a DMA request is desired to be generated for new results getting written to the results' memory.

22.2.4.2 DMA Request for a Fixed Number of Conversion Results

This mode is enabled by setting both the group's DMA_EN and the group's BLK_XFER bits in the group's DMA control registers.

In this mode, a DMA request will be generated for a specified number of conversion results being available in the group's results' memory. The number of conversion results desired are configured using the group's BLOCKS field in the control registers.

For example, if the BLOCK count is configured for 10, then ADC module will generate a DMA request at the end of 10th conversion. DMA controller should complete reading out 10 data before next set of 10 conversions complete.

NOTE: Usage of Block DMA transfers with Threshold Interrupts

It is not recommended to enable the block DMA transfers for a group at the same time as the group threshold interrupt. The group's BLOCKS field is essentially the same as the group's THRESHOLD field in the group's interrupt control register described in Section 22.2.3.2.

22.2.5 ADC Magnitude Threshold Interrupts

The ADC allows up to three magnitude threshold interrupts to be generated. The comparison parameters are programmed via the Magnitude Threshold Control Register (ADMAGINTxCR).

22.2.5.1 Magnitude Threshold Interrupt Configuration

The following fields are configurable for each of the three available magnitude threshold interrupts:

1. CHN_THR_COMP: Specifies whether to compare two channels' conversion results, or to compare a channel's conversion result to a programmable threshold value. A value of 0 will select the programmable threshold to be compared, and a value of 1 will select the conversion result of the channel identified by the COMP_CHID field to be compared.
2. MAG_CHID: Specifies the channel number from 0 to 31 whose conversion result needs to be monitored.
3. COMP_CHID: Specifies the channel number from 0 to 31 whose last conversion result is used for the comparison with the conversion result of the channel being monitored.
4. MAG_THRESHOLD: Specifies the value for comparison with the conversion result of the channel identified by the MAG_CHID field.
5. CMP_GE_LT: Specifies whether the conversion result of the channel identified by MAG_CHID is compared to be “greater than or equal to”, or “less than” the reference value. The reference value can be the conversion result of another channel identified by the COMP CHID field, or it could be a threshold value specified in the MAG_THRESHOLD field. A value of 0 in the CMP_GE_LT field indicates a “less than” comparison and a value of 1 indicates a “greater than or equal to” comparison.

22.2.5.2 Magnitude Threshold Interrupt Comparison Mask Configuration

There is also a separate comparison mask register (ADMAGINTxMASK) for each of the three magnitude threshold interrupts. This register is used to specify the bits that are masked off for the sake of the comparison. For example, the lower 4 bits of the conversion result can be masked off by writing 0xf to the interrupt comparison mask register, allowing a gross comparison to be made. By default, the full 10/12-bit conversion results are compared.

22.2.5.3 Magnitude Threshold Interrupt Enable / Disable Control

Each of the three magnitude interrupts also have separate interrupt enable set (ADMAGINTENASET) and clear (ADMAGINTENACLR) registers. These are used to respectively enable and disable that particular magnitude threshold interrupt from being generated. To enable a magnitude threshold interrupt, write a 1 to the corresponding bit of the interrupt enable set register. Conversely, to disable a magnitude threshold interrupt, write a 1 to the corresponding bit of the interrupt enable clear register.

22.2.5.4 Magnitude Threshold Interrupt Flags

There is a separate Magnitude Interrupt Flag register (ADMAGINTFLG) that holds the flags for these three interrupts. This flag gets set whenever the comparison condition for the corresponding interrupt is met. A magnitude threshold interrupt is generated if the corresponding flag is set inside the flag register, and the interrupt generation is enabled. This flag can be cleared by writing a 1 to the flag or by reading from the interrupt offset register in case of this interrupt being the current highest-priority pending interrupt.

22.2.5.5 Magnitude Threshold Interrupt Offset Register

It is possible to have multiple magnitude threshold interrupts pending at the same time. The magnitude threshold interrupt offset register (ADMAGINTOFF) holds the index of the currently pending highest priority magnitude threshold interrupt. The magnitude threshold interrupt 1 has the highest priority while the magnitude threshold interrupt 3 has the lowest priority. This is a read-only register and returns zeros if none of the magnitude threshold interrupts are pending. Writes to this register have no effect.

A read from this register updates the register to the next highest-priority pending magnitude threshold interrupt. This read also clears the corresponding flag from the magnitude threshold interrupt flag register. However, a read from the magnitude threshold interrupt offset register in emulation mode does not affect the interrupt flag register or the interrupt offset register.

22.2.6 ADC Special Modes

The ADC module supports some special modes for diagnostics and power saving purposes.

22.2.6.1 ADC Error Calibration Mode

The application program can activate a calibration sequence any time self-test mode is disabled (SELF_TEST = 0). This calibration sequence includes the conversion of an embedded calibration reference voltage followed by the calculation of an offset error correction value.

NOTE: Disable Self-Test Mode Before Calibration

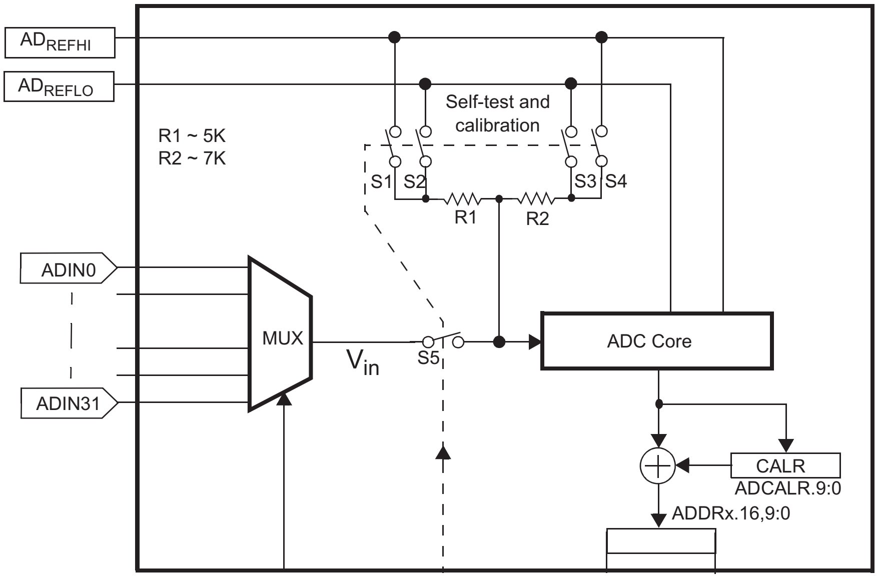
To avoid errors during the calibration operation, self-test mode must *not* be enabled during a calibration sequence. In addition, to ensure accurate results, calibrate the ADC in an environment with minimum noise.

Calibration mode is enabled by setting the CAL_EN bit (ADCALCR.0). The application needs to ensure that no conversion group is being serviced when the calibration mode is enabled.

The input multiplexor gets disabled and only the reference voltage is connected to the ADC core input. Switch S5 of Figure 22-13 is opened. In addition, the digital result issued from a conversion is output from the ADC core to the calibration and offset error correction register, ADCALR. The ADC results' memory is not affected by the calibration conversion.

When calibration mode is disabled, the ADC can be configured for normal conversions.

Figure 22-13. Self-Test and Calibration Logic



22.2.6.1.1 Calibration Conversion

The calibration conversion also needs to meet the minimum sampling time specification for the ADC. This value is typically 1 us. The Event Group sample time register (ADEVSAMP) is used to specify the number of ADCLK cycles for the calibration conversion.

The BRIDGE_EN and HILO bits (ADCALCR.9:8) control the voltage to the calibration reference device shown in Figure 22-15. The positions of the switches in calibration mode are listed in Table 22-2.

Table 22-2. Calibration Reference Voltages[1]

CAL_EN	BRIDGE_EN	HILO	S1	S2	S3	S4	S5	Reference Voltage
1	0	0	1	0	1	0	0	$(AD_{REFHI} \times R1 + AD_{REFLO} \times R2) / (R1 + R2)$
1	0	1	0	1	0	1	0	$(AD_{REFLO} \times R1 + AD_{REFHI} \times R2) / (R1 + R2)$
1	1	0	0	1	1	0	0	AD_{REFLO}
1	1	1	1	0	0	1	0	AD_{REFHI}
0	X	X	0	0	0	0	1	V_{in}

[1] The state of the switches in this table assumes that self-test mode is not enabled.

When CAL_ST (ADCALCR.16) is set, a calibration conversion is started. The voltage source selected via the bits BRIDGE_EN and HILO is converted once (single conversion mode) and the digital result is returned to the calibration and correction register, ADCALR, where it can be read by the CPU. The CAL_ST bit acts as a flag and must be polled by the CPU. It is held set during the conversion process and automatically clears to indicate the end of the reference voltage conversion.

NOTE: No Interrupt for end of calibration

The ADC does not generate an interrupt to signal the end of the calibration conversion. The application must poll the CAL_ST bit to determine the end of the calibration conversion.

After the CAL_ST bit is set by the application program, it can only be reset by the end of the ongoing conversion generated by the ADC core. If the calibration conversion is interrupted (CAL_EN bit is cleared), the CAL_ST bit is held at 1 until a new calibration conversion has been set and completed. Setting the CAL_ST bit while calibration is disabled (CAL_EN = 0) has no effect; however, in this situation, setting CAL_EN immediately starts a calibration conversion. When the calibration conversion is interrupted by an ADC_Enable (ADC_EN = 0, CAL_EN = 1, and CAL_ST = 1), a new conversion is automatically restarted as soon as the ADC_Enable bit is released (ADC_EN = 1).

22.2.6.1.2 Calibration and Offset Error Correction Sequences

The number of measurements and the source to measure for an ADC calibration are application dependent. The CAL_ST bit must be set for each calibration source to be measured. While calibration mode is enabled, any available calibration sources can be converted according to the BRIDGE_EN and HILO bits (see Table 22-2). The digital results of the calibration measurements should be read from ADCALR by the application after each reference conversion so that a correction value can be computed and written back into ADCALR.

When the application has the necessary calibration data, it should compute the offset error correction value and load it into the calibration and correction register, ADCALR. After the CAL_EN bit is cleared, normal conversion mode restarts, continuing from where it was frozen, but with the addition of self-correction data.

In normal mode, the self-correction system adds the correction value stored in ADCALR to each digital result before it is written to the respective group’s FIFO.

The basic calibration routine is as follows:

1. Enable calibration via CAL_EN (ADCALCR.0).
2. Select the voltage source via BRIDGE_EN and HILO (ADCALCR.9:8).
3. Start the conversion with CAL_ST (ADCALCR.16).
4. Wait for CAL_ST to go to 0.
5. Get the results from ADCALR and save to memory.
6. Loop to step 2 until the calibration conversion data is collected for the desired reference voltages.
7. Compute the error correction value using calibration data saved in memory.
8. Load the ADCALR register with the 2s complement of the computed error correction value.
9. Disable calibration mode.

At this point, the ADC can be configured for normal operation, and it corrects each digital result with the error correction value loaded in ADCALR.

NOTE: Prevent ADC Calibration Data From Being Overwritten

In calibration mode, the conversion result is written to ADCALR that overwrites any previous calibration data; therefore, the ADCALR register must be read before a new conversion is started.

For no correction, a value of 0x0000 must be written to ADCALR. In noncalibration mode, the ADCALR register can be read and written. Any value written to ADCALR in normal mode (CAL_EN = 0) is added to each digital result from the ADC core.

22.2.6.1.3 Mid-Point Calibration

Because of its connections to the ADC's reference voltage (VrefHi, VrefLo), the precision of the calibration reference is voltage independent. On the other hand, the accuracy of the switched bridge resistor (R1 & R2) relies on the manufacturing process deviation. Consequently, the mid-point voltage's accuracy can be affected due to the imperfections in the two resistors (expected mismatch error is around 1.5%).

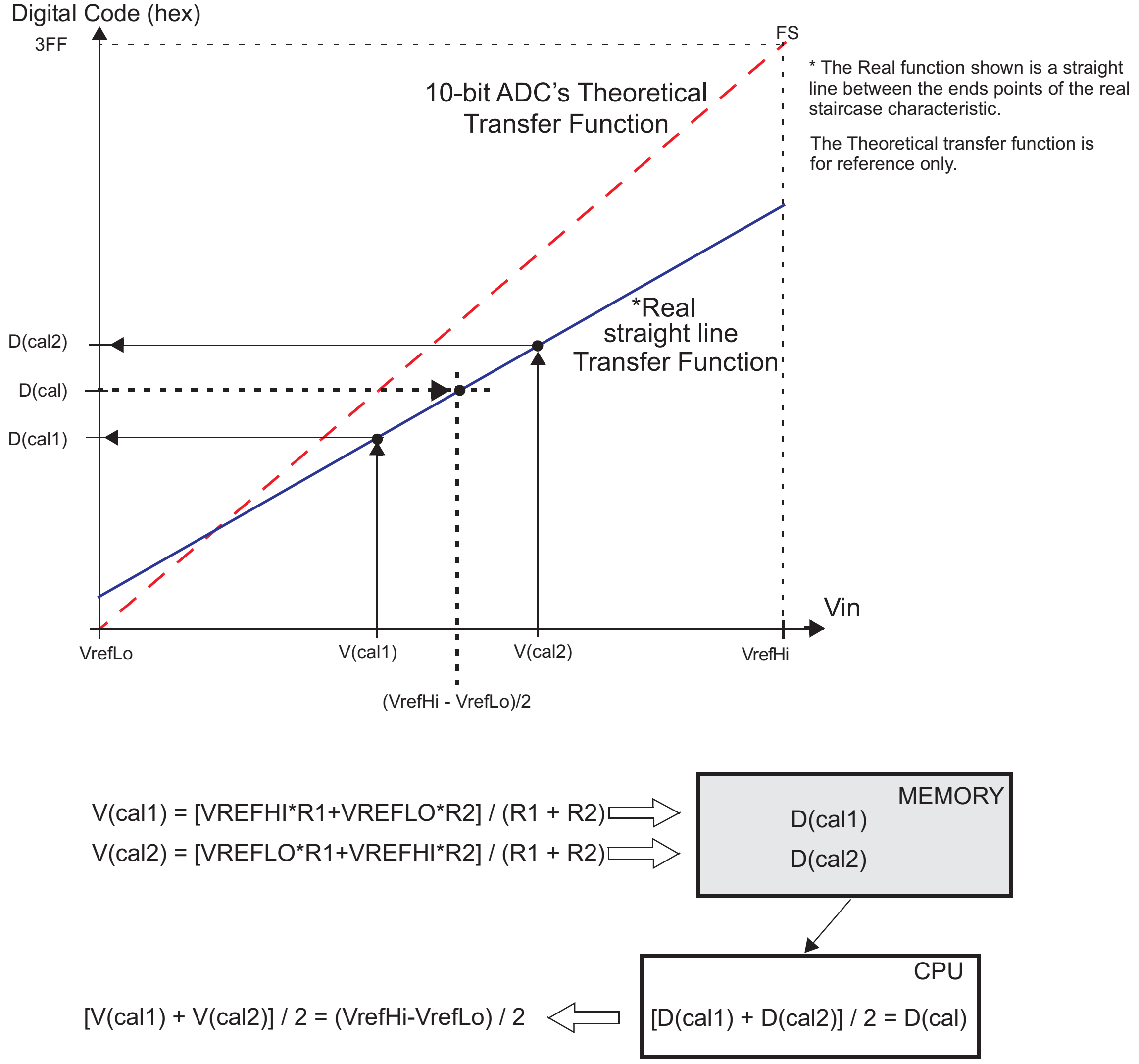
The switched reference voltage device has been specially designed to support a differential measurement of its mid-point voltage. This ensures the accuracy of the mid-point reference, and hence the efficiency of the calibration.

The differential mid-point calibration is software controlled; the algorithm (voltage source measurements and associated calculation) is inserted within the calibration software module included in the application program.

The basic differential mid-point calibration flow is illustrated here after:

1. The application program connects the voltage VrefHi to R1 and VrefLo to R2, (BRIDGE_EN = 0, HILO = 0), launches a conversion of the input voltage V(cal1), and stores the digital result D(cal1) into the memory.
2. Then the application program switches the voltage VrefHi to R2 and VrefLo to R1 (BRIDGE_EN = 0, HILO = 1), converts this new input voltage V(cal2) and again stores the issued digital result D(cal2) into the memory.
3. The actual value of the real middle point is obtained by computing the average of these two results. [D(cal1)+D(cal2)] /2; Figure 22-14 summarizes the mid-point calibration flow.

Figure 22-14. Mid-point Value Calculation

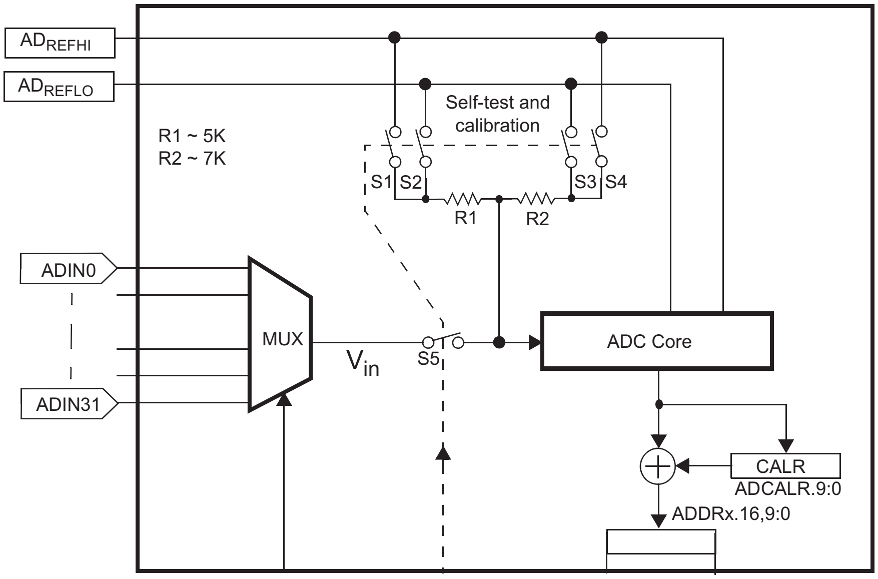

Digital Code (hex)
3FF
FS
10-bit ADC's Theoretical Transfer Function
* The Real function shown is a straight line between the ends points of the real staircase characteristic.
The Theoretical transfer function is for reference only.
*Real straight line Transfer Function
D(cal2)
D(cal)
D(cal1)
Vin
VrefLo
V(cal1)
V(cal2)
VrefHi
(VrefHi - VrefLo)/2
V(cal1) = [VREFHI*R1+VREFLO*R2] / (R1 + R2)
V(cal2) = [VREFLO*R1+VREFHI*R2] / (R1 + R2)
MEMORY
D(cal1)
D(cal2)
CPU
[D(cal1) + D(cal2)] / 2 = D(cal)
[V(cal1) + V(cal2)] / 2 = (VrefHi-VrefLo) / 2

22.2.6.2 ADC Self-Test Mode

The ADC module supports a self-test mode which can be used to detect an open or a short on the ADC input channels. Self-test mode is enabled by setting the SELF_TEST bit (ADCALCR.24). Any conversion type (continuous or single conversion, freeze enabled or non-freeze enabled, interrupts enabled or disabled) can be performed in this mode.

In normal mode, setting the self-test mode while a conversion sequence is in process can corrupt the current channel conversion results. However, the next channel in the sequence is converted correctly during the additional self-test cycle. The logic associated with both self-test and calibration is shown in Figure 22-15.

Figure 22-15. Self-Test and Calibration Logic



In self-test mode, a test voltage defined by the HILO bit (ADCALCR.8) is provided to the ADC core input through a resistor (see Table 22-3). To change the test source, this bit can be toggled before any single conversion mode request. Changing this bit while a conversion is in progress *can* corrupt the results if the source switches during the acquisition period.

Note that the switch S5 shown in Figure 22-15 is only for the purpose of explaining the self-test sequence. There is no physical switch.

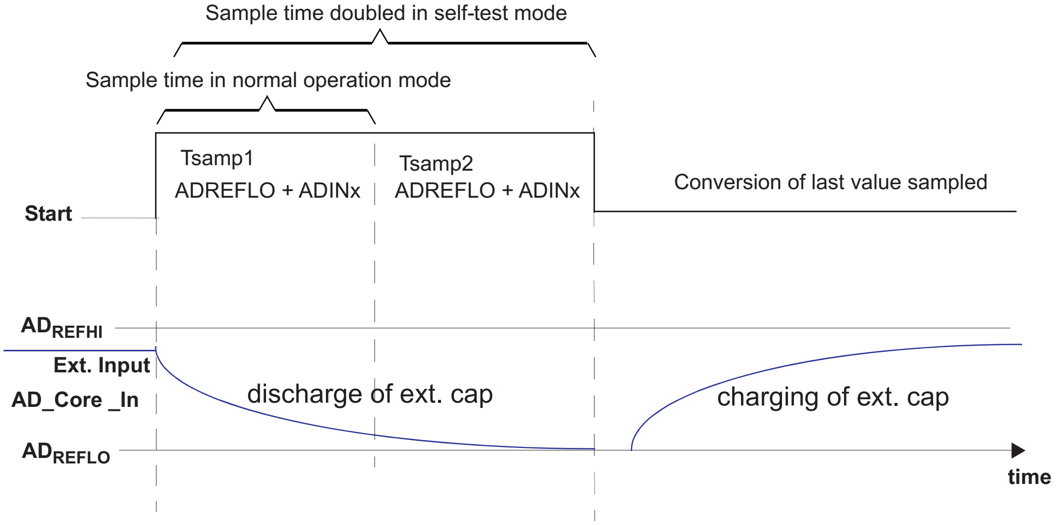
Table 22-3. Self-Test Reference Voltages[1]

SELF_TEST	HILO	S1	S2	S3	S4	S5	Reference Voltage
1	0	0	1	1	0	1	AD_{REFLO} via R1 || R2 connected to V_{in}
1	1	1	0	0	1	1	AD_{REFHI} via R1 || R2 connected to V_{in}
0	X	0	0	0	0	1	V_{in}

[1] Switches refer to Figure 22-15.

Conversions in self-test mode are started just as they are in the normal operating mode (see Section 22.2.1.6). The conversion starts according to the configuration set in the three mode control registers (ADEVMODECR, ADG1MODECR, ADG2MODECR) and the sampling time control registers (ADEVSAMP, ADG1SAMP, ADG2SAMP). The acquisition time for each conversion in self-test mode is extended to twice the normal configured acquisition time. The selected reference voltage and the input voltage from the ADINx input channel are both connected to the ADC internal sampling capacitor throughout this extended acquisition period. Figure 22-16 shows the self-test mode timing when the ADREFLO is chosen as the reference voltage for the self-test mode conversion. It also assumes an external capacitor connected to the ADC input channel.

Figure 22-16. Timing for Self-Test Mode



22.2.6.2.1 Use of Self-Test Mode to Determine Open/Short on ADC Input Channels

The following sequence needs to be used to deduce the ADC pin status:

- Convert the channel with self test enabled and with the reference voltage as Vreflo. Store the conversion result, say Vd.
- Convert the channel with self test enabled and with the reference voltage as Vrefhi. Store the conversion result, say Vu.
- Convert the channel with self test disabled. Store the conversion result, say Vn.

The results can be interpreted using the following table.

Table 22-4. Determination of ADC Input Channel Condition

Normal Conversion Result, Vn	Self-test Conversion Result, Vu	Self-test Conversion Result, Vd	Pin Condition
Vn	Vn < Vu < AD_{REFHI}	AD_{REFLO} < Vd < Vn	Good
AD_{REFHI}	AD_{REFHI}	approx. AD_{REFHI}	Shorted to AD_{REFHI}
AD_{REFLO}	approx. AD_{REFLO}	AD_{REFLO}	Shorted to AD_{REFLO}
Unknown	AD_{REFHI}	AD_{REFLO}	Open

22.2.6.3 ADC Power-Down Mode

This is an inactive mode in which the clocks to the ADC module are stopped leaving the module in a static state. The clock to the ADC core (ADCLK) is stopped whenever there are no ongoing conversions. This is the clock-gating implementation requirement. Also, the ADC module places the ADC core into the power down mode such that there is minimal current drawn from the ADC operating and reference supplies.

22.2.6.3.1 Powering Down Just The ADC Core

The ADC core can be individually powered down without stopping the clocks to the ADC module. This can be done by setting the POWERDOWN bit of the ADC Operating Mode Control Register (ADOPMODECR.3). Whenever a conversion is required the POWERDOWN bit must be cleared, and a minimum time $t_{d(PU\text{-}ADV)}$, (see the specific device data sheet for actual value) has to be allowed before starting a new conversion. This wait must be implemented in the application software.

22.2.6.3.2 Enhanced Power-Down Mode

A bit in the ADC operating mode control register, IDLE_PWRDN (ADOPMODECR.4) enables the enhanced power-down mode of the ADC.

Once this bit is set, the ADC module will power down the ADC core whenever there are no more ongoing or pending ADC conversions. The ADC core will be powered down regardless of the state of the POWERDOWN bit (ADOPMODECR.3).

The ADC module releases the ADC core from power down mode as soon as a new conversion is requested. The ADC logic state machine then has to wait for at least $t_{d(PU\text{-}ADV)}$ (see the device data sheet for actual value) before starting a new conversion. The IDLE_PWRDN bit will remain set at all times. The logic state machine can use this bit to determine that it needs to wait for a programmable number of VCLK cycles before it allows the input channel to be sampled. This time is configured by the ADC Power Up Delay Control register (ADPWRUPDLYCTRL).

If IDLE_PWRDN is not set, the ADC module does not wait for any additional delay before sampling the input channel and the application software has to take account of this required delay.

22.2.6.3.3 Managing Clocks to the ADC Module

The clock to the ADC module can be turned off via the appropriate Peripheral Central Resource (PCR) controller PSPWRDNSET register (check the specific device datasheet to identify the register and the bit to be set). If a conversion is ongoing when this bit is set, the ADC module will wait until the current conversion completes before allowing the ADC module clock to be stopped.

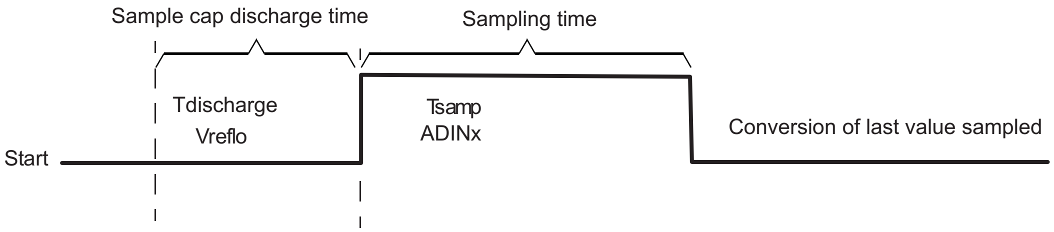
22.2.6.4 ADC Sample Capacitor Discharge Mode

This mode allows the charge on the ADC core's internal sampling capacitor to be discharged before starting the sampling phase of the next channel.

The ADC Sample Cap Discharge Mode is enabled by setting the SAMP_DIS_EN bit of the group's ADSAMPDISEN register. A discharge period for the sampling capacitor is added before the sampling period for each channel as shown in Figure 22-17. The duration of this discharge period is configurable via the corresponding group's_SAMP_DIS_CYC field in the ADSAMPDISEN register. The discharge time is specified in terms of number of ADCLK cycles.

During the sample capacitor discharge period, the V_{REFLO} reference voltage is connected to the input voltage terminal of the ADC core. This allows any charge collected on the sampling capacitor from the previous conversion to be discharged to ground. The V_{REFLO} reference voltage is usually connected to ground.

Figure 22-17. Timing for Sample Capacitor Discharge Mode



22.2.7 ADC Results' RAM Special Features

The following section describes some of the special features supported by the ADC module to enhance the results' RAM testability and integrity.

22.2.7.1 ADC Results' RAM Auto-Initialization

The ADC module allows the application to auto-initialize the ADC results' RAM to all zeros. The application must ensure that the ADC module is not in any of the conversion modes before triggering off the auto-initialization process.

The auto-initialization sequence is as follows:

1. Enable the global hardware memory initialization key by programming a value of 1010 to the bits [3-0] of the MINITGCR register of the System module.
2. Set the control bit for the ADC results' RAM in the MSINENA System module register. The bit 8 of the MSINENA register is used to control the initialization of the ADC1 results' RAM, while bit 14 controls the initialization of the ADC2 results' RAM. This starts the initialization process. The BUF_INIT_ACTIVE flag in the ADBNDEND register will get set to reflect that the initialization is ongoing.
3. When the memory initialization is completed, the corresponding status bit in the MINISTAT register will be set. Also, the BUF_INIT_ACTIVE flag will get cleared.

22.2.7.2 ADC Results' RAM Test Mode

In the defined conversion modes of the ADC, the application can only read from the ADC results' RAM. Only the ADC module is allowed to write to the results' RAM. A special test mode is defined to allow the application to also write into the ADC results' RAM - this mode is the ADC Results' RAM Test Mode. Only 32-bit reads and writes are allowed to the ADC results' RAM in this test mode.

NOTE: Contention on access to ADC Results' RAM

The ADC module cannot handle a contention between the application write to the results' RAM and the ADC writing a conversion result to the results' RAM. The application must ensure that the ADC is not likely to write a new conversion result to the results' RAM when the ADC Results' RAM Test Mode is enabled.

The ADC Results' RAM Test Mode is enabled by setting the RAM_TEST_EN bit in the ADOPMODECR.

22.2.7.3 ADC Results' RAM Parity

The following shows the ADC Results' RAM parity control registers.

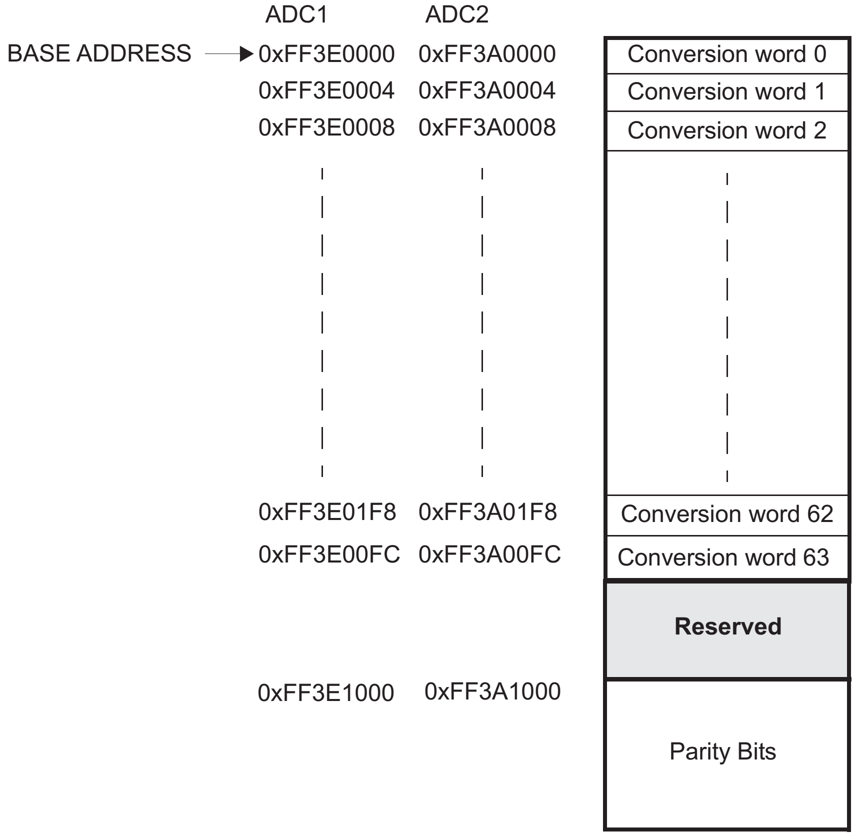
Parity checking is implemented using parity on a per-half word basis for the ADC RAM. That is, there is one parity bit for 16 bits of the ADC RAM. The polarity of the ADC RAM parity is controlled by the DEVCR1 register in the system module (address = 0xFFFFFFDC). The parity checking is enabled by the ADPARCR register. After reset, the parity checking is disabled and must be enabled if parity protection is required.

During a read access, the parity is calculated based on the data read from the ADC RAM and compared with the good parity value stored in the parity bits. If any word fails the parity check then the ADC generates an error signal hooked up to the Error Signaling Module (ESM). The ADC RAM address which generated the parity error is captured for host system debugging, and is frozen from being updated until it is read by the application.

Testing the Parity Checking Mechanism:

To test the parity checking mechanism itself, the parity RAM is made writable by the CPU in a special test mode. This is done by a control bit called TEST in the AD PAR CR register. Once this bit is set, the parity bits are mapped to an address starting at an address offset of 4KB from the base address of the ADC RAM. See Figure 22-18. The CPU can now manually insert parity errors. Note that the ADC RAM only supports 32-bit accesses.

Figure 22-18. ADC Memory Map in Parity Test Mode



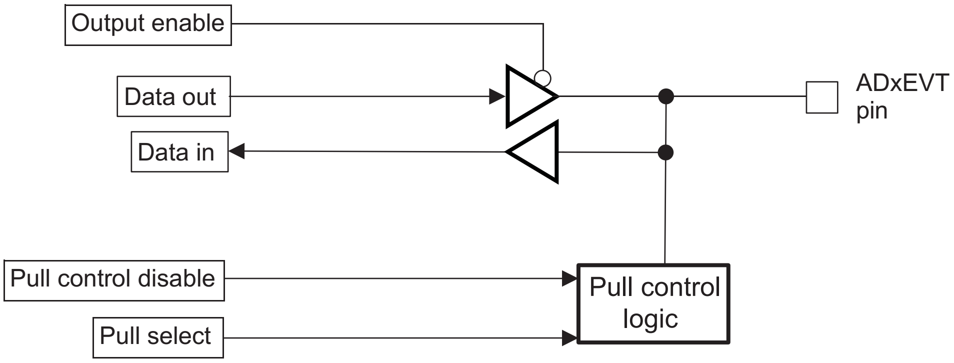
22.2.8 ADEVT Pin General Purpose I/O Functionality

The AD1EVT pin for ADC1 and AD2EVT pin for ADC2 can be configured as general-purpose I/O signals. The following sections describe the different ways in which the application can configure the ADxEVT pins.

22.2.8.1 GPIO Functionality

Figure 22-19 illustrates the GPIO functionality of the ADxEVT pin.

Figure 22-19. GPIO Functionality of ADxEVT



Once the device power-on reset is released, the ADC module controls the state of the ADxEVT pin.

- **Pull control:** The pull control can either be enabled or disabled by default (while system reset is active and after it is released). The actual default state of the pull control is specified in the device datasheet. The application can enable pull control by clearing the PDIS (pull control disable) bit in the ADEVTPDIS register. In this case, if the PSEL (pull select) bit in the ADEVTPSEL register is set, the pin will have a pull-up. If the PSEL bit is cleared, the pin will have a pull-down. If the PDIS bit is set in the control register, there is no pull-up or pull-down on the pin.

NOTE: **Pull Behavior when ADxEVT is configured as output**

If the ADxEVT pin is configured as output, then the pulls are disabled automatically. If the pin is configured as input, the pulls are enabled or disabled depending on bit PDIS in the pull disable register ADEVTPDIS.

- **Output buffer:** The ADxEVT pin can be driven as an output pin if the ADEVTDIR bit is set in the pin direction control register.
- **Open-Drain Feature:** The open drain output capability is enabled via the ADEVTPDR control register. The ADxEVT pin must be also configured to be an output pin for this mode.
  - The output buffer is enabled if a low signal is being driven on to the pin.
  - The output buffer is disabled if a high signal is being driven on to the pin.

22.2.8.2 Summary

The behavior of the output buffer, and the pull control is summarized in Table 22-5. The input buffer for the ADxEVT pins are enabled once the device power-on reset is released.

Table 22-5. Output Buffer and Pull Control Behavior for ADxEVT as GPIO Pins

System Reset Active?	Pin Direction (DIR) [1][2]	Pull Disable (PDIS) [1] [3]	Pull Select (PSEL) [1] [4]	Pull Control	Output Buffer
Yes	X	X	X	Enabled	Disabled
No	0	0	0	Pull down	Disabled
No	0	0	1	Pull up	Disabled
No	0	1	0	Disabled	Disabled
No	0	1	1	Disabled	Disabled
No	1	X	X	Disabled	Enabled

[1] X = Don't care

[2] DIR = 0 for input, 1 for output

[3] PULDIS = 0 for enabling pull control, 1 for disabling pull control

[4] PULSEL = 0 for pull-down functionality, 1 for pull-up functionality

22.3 ADC Registers

All registers in the ADC module are 32-bit, word-aligned; 8-bit, 16-bit and 32-bit accesses are allowed. The application must ensure that the reserved bits are always written as 0 to ensure software compatibility to future revisions of the module. Table 22-6 shows register address offsets from the base address of the ADC modules. The base address of ADC1 registers is FFF7 C000h and the base address of ADC2 registers is FFF7 C200h.

Table 22-6. ADC Registers

Offset	Acronym	Register Description	Section
00h	ADRSTCR	ADC Reset Control Register	Section 22.3.1
04h	ADOPMODECR	ADC Operating Mode Control Register	Section 22.3.2
08h	ADCLOCKCR	ADC Clock Control Register	Section 22.3.3
0Ch	ADCALCR	ADC Calibration Mode Control Register	Section 22.3.4
10h	ADEVMODECR	ADC Event Group Operating Mode Control Register	Section 22.3.5
14h	ADG1MODECR	ADC Group1 Operating Mode Control Register	Section 22.3.6
18h	ADG2MODECR	ADC Group2 Operating Mode Control Register	Section 22.3.7
1Ch	ADEVSRC	ADC Trigger Source Select Register	Section 22.3.8
20h	ADG1SRC	ADC Group1 Trigger Source Select Register	Section 22.3.9
24h	ADG2SRC	ADC Group2 Trigger Source Select Register	Section 22.3.10
28h	ADEVINTENA	ADC Event Interrupt Enable Control Register	Section 22.3.11
2Ch	ADG1INTENA	ADC Group1 Interrupt Enable Control Register	Section 22.3.12
30h	ADG2INTENA	ADC Group2 Interrupt Enable Control Register	Section 22.3.13
34h	ADEVINTFLG	ADC Event Group Interrupt Flag Register	Section 22.3.14
38h	ADG1INTFLG	ADC Group1 Interrupt Flag Register	Section 22.3.15
3Ch	ADG2INTFLG	ADC Group2 Interrupt Flag Register	Section 22.3.16
40h	ADEVTHRINTCR	ADC Event Group Threshold Interrupt Control Register	Section 22.3.17
44h	ADG1THRINTCR	ADC Group1 Threshold Interrupt Control Register	Section 22.3.18
48h	ADG2THRINTCR	ADC Group2 Threshold Interrupt Control Register	Section 22.3.19
4Ch	ADEVDMACR	ADC Event Group DMA Control Register	Section 22.3.20
50h	ADG1DMACR	ADC Group1 DMA Control Register	Section 22.3.21
54h	ADG2DMACR	ADC Group2 DMA Control Register	Section 22.3.22
58h	ADBNDCR	ADC Results Memory Configuration Register	Section 22.3.23
5Ch	ADBNDEND	ADC Results Memory Size Configuration Register	Section 22.3.24
60h	ADEVSAMP	ADC Event Group Sampling Time Configuration Register	Section 22.3.25
64h	ADG1SAMP	ADC Group1 Sampling Time Configuration Register()	Section 22.3.26
68h	ADG2SAMP	ADC Group2 Sampling Time Configuration Register	Section 22.3.27
6Ch	ADEVSR	ADC Event Group Status Register	Section 22.3.28
70h	ADG1SR	ADC Group1 Status Register	Section 22.3.29
74h	ADG2SR	ADC Group2 Status Register	Section 22.3.30
78h	ADEVSEL	ADC Event Group Channel Select Register	Section 22.3.31
7Ch	ADG1SEL	ADC Group1 Channel Select Register	Section 22.3.32
80h	ADG2SEL	ADC Group2 Channel Select Register	Section 22.3.33
84h	ADCALR	ADC Calibration and Error Offset Correction Register	Section 22.3.34
88h	ADSMSTATE	ADC State Machine Status Register	Section 22.3.35
8Ch	ADLASTCONV	ADC Channel Last Conversion Value Register	Section 22.3.36
90h-AFh	ADEVBUFFER	ADC Event Group Results FIFO Register	Section 22.3.37
B0h-CFh	ADG1BUFFER	ADC Group1 Results FIFO Register	Section 22.3.38
D0h-EFh	ADG2BUFFER	ADC Group2 Results FIFO Register	Section 22.3.39
F0h	ADEVEMUBUFFER	ADC Event Group Results Emulation FIFO Register	Section 22.3.40
F4h	ADG1EMUBUFFER	ADC Group1 Results Emulation FIFO Register	Section 22.3.41

Table 22-6. ADC Registers (continued)

Offset	Acronym	Register Description	Section
F8h	ADG2EMUBUFFER	ADC Group2 Results Emulation FIFO Register	Section 22.3.42
FCh	ADEVTDIR	ADC ADEVT Pin Direction Control Register	Section 22.3.43
100h	ADEVTOUT	ADC ADEVT Pin Output Value Control Register	Section 22.3.44
104h	ADEVTIN	ADC ADEVT Pin Input Value Register	Section 22.3.45
108h	ADEVTSET	ADC ADEVT Pin Set Register	Section 22.3.46
10Ch	ADEVTCLR	ADC ADEVT Pin Clear Register	Section 22.3.47
110h	ADEVTPDR	ADC ADEVT Pin Open Drain Enable Register	Section 22.3.48
114h	ADEVTPDIS	ADC ADEVT Pin Pull Control Disable Register	Section 22.3.49
118h	ADEVTPSEL	ADC ADEVT Pin Pull Control Select Register	Section 22.3.50
11Ch	ADEVSAMPDISEN	ADC Event Group Sample Cap Discharge Control Register	Section 22.3.51
120h	ADG1SAMPDISEN	ADC Group1 Sample Cap Discharge Control Register	Section 22.3.52
124h	ADG2SAMPDISEN	ADC Group2 Sample Cap Discharge Control Register	Section 22.3.53
128h-138h	ADMAGINTxCR	ADC Magnitude Compare Interrupt Control Register	Section 22.3.54
12Ch-13Ch	ADMAGxMASK	ADC Magnitude Compare Mask Register	Section 22.3.55
158h	ADMAGINTENASET	ADC Magnitude Compare Interrupt Enable Set Register	Section 22.3.56
15Ch	ADMAGINTENACLR	ADC Magnitude Compare Interrupt Enable Clear Register	Section 22.3.57
160h	ADMAGINTFLG	ADC Magnitude Compare Interrupt Flag Register	Section 22.3.58
164h	ADMAGINTOFF	ADC Magnitude Compare Interrupt Offset Register	Section 22.3.59
168h	ADEVFIFORESETCR	ADC Event Group FIFO Reset Control Register	Section 22.3.60
16Ch	ADG1FIFORESETCR	ADC Group1 FIFO Reset Control Register	Section 22.3.61
170h	ADG2FIFORESETCR	ADC Group2 FIFO Reset Control Register	Section 22.3.62
174h	ADEVRAMWRADDR	ADC Event Group RAM Write Address Register	Section 22.3.63
178h	ADG1RAMWRADDR	ADC Group1 RAM Write Address Register	Section 22.3.64
17Ch	ADG2RAMWRADDR	ADC Group2 RAM Write Address Register	Section 22.3.65
180h	ADPARCR	ADC Parity Control Register	Section 22.3.66
184h	ADPARADDR	ADC Parity Error Address Register	Section 22.3.67
188h	ADPWRUPDLYCTRL	ADC Power-Up Delay Control Register	Section 22.3.68
190h	ADEVCHNSELMODECTRL	ADC Event Group Channel Selection Mode Control Register	Section 22.3.69
194h	ADG1CHNSELMODECTRL	ADC Group1 Channel Selection Mode Control Register	Section 22.3.70
198h	ADG2CHNSELMODECTRL	ADC Group2 Channel Selection Mode Control Register	Section 22.3.71
19Ch	ADEVCURRCOUNT	ADC Event Group Current Count Register	Section 22.3.72
1A0h	ADEVMAXCOUNT	ADC Event Group Max Count Register	Section 22.3.73
1A4h	ADG1CURRCOUNT	ADC Group1 Current Count Register	Section 22.3.74
1A8h	ADG1MAXCOUNT	ADC Group1 Max Count Register	Section 22.3.75
1ACh	ADG2CURRCOUNT	ADC Group2 Current Count Register	Section 22.3.76
1B0h	ADG2MAXCOUNT	ADC Group2 Max Count Register	Section 22.3.77

22.3.1 ADC Reset Control Register (ADRSTCR)

Figure 22-20 and Table 22-7 describe the ADRSTCR register.

Figure 22-20. ADC Reset Control Register (ADRSTCR) [offset = 00]

31 … 1	0
Reserved	RESET
R-0	R/WP-0

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privileged mode only; -*n* = value after reset

Table 22-7. ADC Reset Control Register (ADRSTCR) Field Descriptions

Bit	Field	Value	Description
31-1	Reserved	0	Reads return 0. Writes have no effect.
0	RESET		This bit is used to reset the ADC internal state machines and control/status registers. This reset state is held until this bit is cleared. Read in all modes, write in privileged mode.
		0	Module is released from the reset state.
		1	All the module's internal state machines and the control/status registers are reset.

22.3.2 ADC Operating Mode Control Register (ADOPMODECR)

Figure 22-21 and Table 22-8 describe the ADOPMODECR register.

Figure 22-21. ADC Operating Mode Control Register (ADOPMODECR) [offset = 04]

31	30 … 25	24
10_12_BIT	Reserved	COS
R/W-0	R-0	R/W-0

23 … 21	20 … 17	16
Reserved	CHN_TEST_EN	RAM_TEST_ EN
R-0	R/W-Ah	R/W-0

15 … 9	8
Reserved	POWER DOWN
R-0	R/W-0

7 … 5	4	3 … 1	0
Reserved	IDLE_PWRDN	Reserved	ADC_EN
R-0	R/W-0	R-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 22-8. ADC Operating Mode Control Register (ADOPMODECR) Field Descriptions

Bit	Field	Value	Description
31	10_12_BIT		This bit controls the resolution of the ADC core. It also affects the size of the conversion results stored in the results' RAM.
			Any operation mode read/write:
		0	The ADC core and digital logic are configured to be in 10-bit resolution. This is the default mode of operation.
		1	The ADC core and digital logic are configured to be in 12-bit resolution.
30-25	Reserved	0	Reads return 0. Writes have no effect.

Table 22-8. ADC Operating Mode Control Register (ADOPMODECR) Field Descriptions (continued)

Bit	Field	Value	Description
24	COS		This bit affects *emulation operation only*. It defines whether the ADC core clock (ADCLK) is immediately halted when the emulation system enters suspend mode or if it should continue operating normally.
			Note: If COS = 0 when the ADC module enters the emulation mode, then the accuracy of the conversion results can be affected depending on how long the module stays in the emulation mode.
			Any operation mode read/write:
		0	ADC module halts all ongoing conversions immediately after emulation mode is entered.
		1	ADC module continues all ongoing conversions as per the configurations of the three conversion groups.
23-21	Reserved	0	Reads return 0. Writes have no effect.
20-17	CHN_TEST_EN		Enable the input channels' impedance measurement mode.
			This mode is reserved for use by TI.
			Any operation mode read/write:
		Ah	Input impedance measurement mode is disabled.
		5h	Input impedance measurement mode is enabled.
		other values	Input impedance measurement mode is disabled.
16	RAM_TEST_EN		Enable the ADC Results' RAM Test Mode.
			Refer to Section 22.2.7.2 for more details.
			Any operation mode read/write:
		0	ADC RAM Test Mode is disabled. The application cannot write to the ADC RAM by the CPU or the DMA.
		1	ADC RAM Test Mode is enabled. The application can directly write to the ADC RAM by the CPU or the DMA.
15-9	Reserved	0	Reads return 0. Writes have no effect.
8	POWERDOWN		ADC Power Down. This bit powers down only the ADC core; the digital logic in the sequencer stays active. To release the core from power down mode, this bit must be cleared. If a conversion is ongoing, the ADC module will wait until the current conversion is completed before powering down the ADC core.
			Also refer to Section 22.3.68, ADC Power-Up Delay Control Register (ADPWRUPDLYCTRL).
			Any operation mode read/write:
		0	The state of the ADC core is controlled by the IDLE_PWRDN bit, or by a global power down mode entry.
		1	ADC core is in the power-down state.
7-5	Reserved	0	Reads return 0. Writes have no effect.
4	IDLE_PWRDN		ADC Power Down When Idle. When this bit is set, the ADC module will automatically power down the ADC core whenever there are no conversions ongoing or pending. This is the enhanced power down mode.
			Also refer to Section 22.3.68, ADC Power-Up Delay Control Register (ADPWRUPDLYCTRL).
			Any operation mode read/write:
		0	The ADC stays in the normal operating mode even if no conversions are ongoing or pending. The power down state is entered only by configuring the POWER DOWN bit or via a global power down mode entry.
		1	Enhanced power down mode is enabled.
3-1	Reserved	0	Reads return 0. Writes have no effect.
0	ADC_EN		ADC Enable. This bit must be set to allow the ADC module to be configured to perform any conversions.
			Any operation mode read/write:
		0	No ADC conversions can occur. The input channel select registers: ADEVSEL, ADG1SEL, and ADG2SEL are held at their reset values.
		1	ADC conversions can now proceed as configured.

22.3.3 ADC Clock Control Register (ADCLOCKCR)

Figure 22-22 and Table 22-9 describe the ADCLOCKCR register.

Figure 22-22. ADC Clock Control Register (ADCLOCKCR) [offset = 08h]

31 5	4 0
Reserved	PS
R-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 22-9. ADC Clock Control Register (ADCLOCKCR) Field Descriptions

Bit	Field	Value	Description
31-5	Reserved	0	Reads return 0. Writes have no effect.
4-0	PS	0-1Fh	ADC Clock Prescaler. These bits define the prescaler value for the ADC core clock (ADCLK). The ADCLK is generated by dividing down the input bus clock (VCLK) to the ADC module. **Note:** The supported range for the ADC clock frequency is specified in the device datasheet. The ADC clock prescaler must be configured to meet this datasheet specification. Any operation mode read/write: $t_{C(ADCLK)} = t_{C(VCLK)} \times (PS[4:0] + 1)$, where $t_{C(ADCLK)}$ is the period of the ADCLK and $t_{C(VCLK)}$ is the period of the VCLK.

22.3.4 ADC Calibration Mode Control Register (ADCALCR)

Figure 22-23 and Table 22-10 describe the ADCALCR register.

Figure 22-23. ADC Calibration Mode Control Register (ADCALCR) [offset = 0Ch]

31 25	24
Reserved	SELF_TEST
R-0	R/W-0

23 17	16
Reserved	CAL_ST
R-0	R/S-0

15 10	9	8
Reserved	BRIDGE_EN	HILO
R-0	R/W-0	R/W-0

7 1	0
Reserved	CAL_EN
R-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; S = Set; -*n* = value after reset

Table 22-10. ADC Calibration Mode Control Register (ADCALCR) Field Descriptions

Bit	Field	Value	Description
31-25	Reserved	0	Reads return 0. Writes have no effect.
24	SELF_TEST		ADC Self Test Enable. When this bit is Set, either AD_{REFHI} or AD_{REFLO} is connected through a resistor to the selected input channel. The desired conversion mode is configured in the group mode control registers. For more details on the ADC Self Test Mode, refer to Section 22.2.6.2.
			Any operation mode read/write:
		0	ADC Self Test mode is disabled.
		1	ADC Self Test mode is enabled.
23-17	Reserved	0	Reads return 0. Writes have no effect.
16	CAL_ST		ADC Calibration Conversion Start. Setting the CAL_ST bit while the CAL_EN bit is set starts conversion of the selected reference voltage. The ADC module uses the sample time configured in the Event Group sample time configuration register (ADEVSAMP) for the calibration conversion.
			Any operation mode:
		0	Read: Calibration conversion has completed, or has not yet been started. Write: No effect.
		1	Read: Calibration conversion is in progress. Write: ADC module starts calibration conversion.
15-10	Reserved	0	Reads return 0. Writes have no effect.
9	BRIDGE_EN		Bridge Enable. When set with the HILO bit, BRIDGE_EN allows a reference voltage to be converted in calibration mode. Table 22-2 defines the four different reference voltages that can be selected.
8	HILO		ADC Self Test mode and Calibration Mode Reference Source Selection. In the ADC Self Test mode, this bit defines the test voltage to be combined through a resistor with the selected input pin voltage. Refer to Section 22.2.6.2 for details on the ADC Self Test Mode. In the ADC Calibration Mode, this bit defines the reference source polarity. Refer to Section 22.2.6.1 for details on the ADC Calibration Mode. In the ADC module’s normal operating mode, this bit has no effect.
7-1	Reserved	0	Reads return 0. Writes have no effect.
0	CAL_EN		ADC Calibration Enable. When this bit is set, the input channel multiplexor is disconnected and the calibration reference voltage is connected to the ADC core input. The calibration reference voltage is selected by the combination of the BRIDGE_EN and HILO. The actual conversion of this reference voltage starts when the CAL_ST bit is set. If the CAL_ST bit is already set when the CAL_EN bit is set, then the calibration conversion is immediately started. Refer to Section 22.2.6.1 for more details on the ADC calibration mode.
			Any operation mode read/write:
		0	Calibration mode is disabled.
		1	Calibration mode is enabled.

22.3.5 ADC Event Group Operating Mode Control Register (ADEVMODECR)

ADC Event Group Operating Mode Control Register (ADEVMODECR) is shown in Figure 22-24 and Figure 22-25, and described in Table 22-11. As shown, the format of the ADEVMODECR is different based on whether the ADC module is configured to be a 12-bit or a 10-bit ADC module.

Figure 22-24. 12-bit ADC Event Group Operating Mode Control Register (ADEVMODECR) [offset = 10h]

<table>
<tr><td>31</td><td></td><td></td><td></td><td></td><td></td><td></td><td>24</td></tr>
<tr><td colspan="8">Reserved</td></tr>
<tr><td colspan="8">R-0</td></tr>
<tr><td>23</td><td></td><td></td><td></td><td></td><td></td><td>17</td><td>16</td></tr>
<tr><td colspan="7">Reserved</td><td>No Reset on ChnSel</td></tr>
<tr><td colspan="7">R-0</td><td>R/W-0</td></tr>
<tr><td>15</td><td></td><td></td><td></td><td></td><td>10</td><td>9</td><td>8</td></tr>
<tr><td colspan="6">Reserved</td><td colspan="2">EV_DATA_FMT</td></tr>
<tr><td colspan="6">R-0</td><td colspan="2">R/W-0</td></tr>
<tr><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td></tr>
<tr><td colspan="2">Reserved</td><td>EV_CHID</td><td>OVR_EV_ RAM_IGN</td><td colspan="2">Reserved</td><td>EV_MODE</td><td>FRZ_EV</td></tr>
<tr><td colspan="2">R-0</td><td>R/W-0</td><td>R/W-0</td><td colspan="2">R-0</td><td>R/W-0</td><td>R/W-0</td></tr>
</table>

LEDGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Figure 22-25. 10-bit ADC Event Group Operating Mode Control Register (ADEVMODECR) [offset = 10h]

<table>
<tr><td>31</td><td></td><td></td><td></td><td></td><td></td><td></td><td>24</td></tr>
<tr><td colspan="8">Reserved</td></tr>
<tr><td colspan="8">R-0</td></tr>
<tr><td>23</td><td></td><td></td><td></td><td></td><td></td><td>17</td><td>16</td></tr>
<tr><td colspan="7">Reserved</td><td>No Reset on ChnSel</td></tr>
<tr><td colspan="7">R-0</td><td>R/W-0</td></tr>
<tr><td>15</td><td></td><td></td><td></td><td></td><td></td><td></td><td>8</td></tr>
<tr><td colspan="8">Reserved</td></tr>
<tr><td colspan="8">R-0</td></tr>
<tr><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td></tr>
<tr><td colspan="2">Reserved</td><td>EV_CHID</td><td>OVR_EV_ RAM_IGN</td><td>Reserved</td><td>EV_8BIT</td><td>EV_MODE</td><td>FRZ_EV</td></tr>
<tr><td colspan="2">R-0</td><td>R/W-0</td><td>R/W-0</td><td>R-0</td><td>R/W-0</td><td>R/W-0</td><td>R/W-0</td></tr>
</table>

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 22-11. ADC Event Group Operating Mode Control Register (ADEVMODECR) Field Descriptions

Field	Value	Description
Reserved	0	Reads return 0. Writes have no effect.
No Reset on ChnSel		No Event Group Results Memory Reset on New Channel Select.
		This bit determines whether the event group results' RAM is reset whenever a non-zero value is written to the event group channel select register.
		Any operation mode read/write:
	0	Event group results RAM is reset when a non-zero value is written to event group channel select register, even if event group conversions are completed.
	1	Event group results RAM is not reset when a non-zero value is written to event group channel select register, and event group conversions are completed.
		If the event group conversions are ongoing (active or frozen), then writing a non-zero value to the event group channel select register will always reset the event group results RAM.
EV_DATA_FMT		Event Group Read Data Format.
		This field is only applicable when the ADC module is configured to be in the 12-bit ADC module. This field is reserved when the module is configured as a 10-bit ADC module.
		This field determines the format in which the conversion results are read out of the Event group results RAM when using the FIFO interface, that is, when reading from the ADEVBUFFER or ADEVEMUBUFFER locations.
		Any operation mode read/write:
	0	Conversion results are read out in full 12-bit format. This is the default mode.
	1h	Conversion results are read out in 10-bit format. Bits 11-2 of the 12-bit conversion result are returned as the 10-bit conversion result.
	2h	Conversion results are read out in 8-bit format. Bits 11-4 of the 12-bit conversion result are returned as the 8-bit conversion result.
	3h	Reserved. The full 12-bit conversion result is returned if programmed.
EV_CHID		Enable Channel Id for the Event Group conversion results to be read. This bit only affects the "read from FIFO" mode. The ADC always stores the channel id in the results RAM. Any 16-bit read performed in the "read from RAM" mode will return the 5-bit channel id along with the 10-bit conversion result.
		Any operation mode read/write:
	0	Bits 14-10, the channel id field, of the data read from the Event Group results' FIFO is read as 00000b.
	1	Bits 14-10, the channel id field, of the data read from the Event Group results' FIFO contains the number of the ADC analog input to which the conversion result belongs.
OVR_EV_RAM_IGN		This bit allows the ADC module to overwrite the contents of the Event Group results memory under an overrun condition.
		Any operation mode read/write:
	0	The ADC cannot overwrite the contents of the Event Group results memory. When an overrun of this memory occurs, the software needs to read out all the contents of this memory before the ADC is able to write a new conversion result for the Event Group.
	1	When an overrun of the Event Group results memory occurs, the ADC proceeds to overwrite the contents with any new conversion results for the Event Group, starting with the first location in this memory.
EV_8BIT		Event Group 8-bit result mode.
		This bit is only applicable when the ADC module is configured to be a 10-bit ADC module. This field is reserved when the module is configured as a 12-bit ADC module.
		This bit allows the Event Group conversion results to be read out in an 8-bit format. This bit only applies to the "read from FIFO" mode. The lower 2 bits of the 10-bit conversion result are discarded and the upper 8 bits are shifted right two places to form the 8-bit conversion result.
		Any operation mode read/write:
	0	The Event Group conversion result is read out as a 10-bit value in the "read from Event Group FIFO" mode.
	1	The Event Group conversion result is read out as an 8-bit value in the "read from Event Group FIFO" mode.

Table 22-11. ADC Event Group Operating Mode Control Register (ADEVMODECR) Field Descriptions (continued)

Field	Value	Description
EV_MODE		Event Group Conversion Mode. This bit defines whether the input channels selected for conversion in the Event Group are converted only once per trigger, or are continuously converted.
		Any operation mode read/write:
	0	The channels selected for conversion in the Event Group are converted only once when the selected event trigger condition occurs.
	1	The channels selected for conversion in the Event Group are converted continuously when the selected event trigger condition occurs.
FRZ_EV		Event Group Freeze Enable. This bit allows an Event Group conversion sequence to be frozen if a Group1 or a Group2 conversion is requested. The Event Group conversion is kept frozen while the Group1 or Group2 conversion is active, and continues from where it was frozen once the Group1 or Group2 conversions are completed.
		While the Event Group conversion is frozen, the EV_STOP status flag in the ADEVSR register indicates that the Event Group conversions have stopped. This bit gets cleared when the Event Group conversions resume.
		Any operation mode read/write:
	0	Event Group conversions cannot be frozen. All the channels selected for conversion in the Event Group are converted before the ADC can switch over to servicing any other conversion group.
	1	Event Group conversions are frozen whenever there is a request for conversion from Group1 or Group2.

22.3.6 ADC Group1 Operating Mode Control Register (ADG1MODECR)

ADC Group1 Operating Mode Control Register (ADG1MODECR) is shown in Figure 22-26 and Figure 22-27, and described in Table 22-12. As shown, the format of the ADG1MODECR is different based on whether the ADC module is configured to be a 12-bit or a 10-bit ADC module.

Figure 22-26. 12-bit ADC Group1 Operating Mode Control Register (ADG1MODECR) [offset = 14h]

31	24
Reserved	
R-0	

23	17	16
Reserved		No Reset on ChnSel
R-0		R/W-0

15	10	9	8
Reserved		G1_DATA_FMT	
R-0		R/W-0	

7	6	5	4	3	2	1	0
Reserved		G1_CHID	OVR_G1_ RAM_IGN	G1_HW_TRIG	Reserved	G1_MODE	FRZ_G1
R-0		R/W-0	R/W-0	R/W-0	R-0	R/W-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Figure 22-27. 10-bit ADC Group1 Operating Mode Control Register (ADG1MODECR) [offset = 14h]

31	24
Reserved	
R-0	

23	17	16
Reserved		No Reset on ChnSel
R-0		R/W-0

15	8
Reserved	
R-0	

7	6	5	4	3	2	1	0
Reserved		G1_CHID	OVR_G1_ RAM_IGN	G1_HW_TRIG	G1_8BIT	G1_MODE	FRZ_G1
R-0		R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 22-12. ADC Group1 Operating Mode Control Register (ADG1MODECR) Field Descriptions

Field	Value	Description
Reserved	0	Reads return 0. Writes have no effect.
No Reset on ChnSel		No Group1 Results Memory Reset on New Channel Select. This bit determines whether the group1 results' RAM is reset whenever a non-zero value is written to the group1 channel select register. Any operation mode read/write:
	0	Group1 results RAM is reset when a non-zero value is written to group1 channel select register, even if group1 conversions are completed.
	1	Group1 results RAM is not reset when a non-zero value is written to group1 channel select register, and group1 conversions are completed. If the group1 conversions are ongoing (active or frozen), then writing a non-zero value to the group1 channel select register will always reset the group1 results RAM.
G1_DATA_FMT		Group1 Read Data Format. This field is only applicable when the ADC module is configured to be in the 12-bit ADC module. This field is reserved when the module is configured as a 10-bit ADC module. This field determines the format in which the conversion results are read out of the group1 results RAM when using the FIFO interface, that is, when reading from the ADG1BUFFER or ADG1EMUBUFFER locations. Any operation mode read/write:
	0	Conversion results are read out in full 12-bit format. This is the default mode.
	1h	Conversion results are read out in 10-bit format. Bits 11-2 of the 12-bit conversion result are returned as the 10-bit conversion result.
	2h	Conversion results are read out in 8-bit format. Bits 11-4 of the 12-bit conversion result are returned as the 8-bit conversion result.
	3h	Reserved. The full 12-bit conversion result is returned if programmed.
G1_CHID		Enable Channel Id for the Group1 conversion results to be read. This bit only affects the “read from FIFO” mode. The ADC always stores the channel id in the results RAM. Any 16-bit read performed in the “read from RAM” mode will return the 5-bit channel id along with the 10-bit conversion result. Any operation mode read/write:
	0	Bits 14-10, the channel id field, of the data read from the Group1 results' FIFO is read as 00000b.
	1	Bits 14-10, the channel id field, of the data read from the Group1 results' FIFO contains the number of the ADC analog input to which the conversion result belongs.
OVR_G1_RAM_IGN		This bit allows the ADC module to overwrite the contents of the Group1 results memory under an overrun condition. Any operation mode read/write:
	0	The ADC cannot overwrite the contents of the Group1 results memory. When an overrun of this memory occurs, the software needs to read out all the contents of this memory before the ADC is able to write a new conversion result for the Group1.
	1	When an overrun of the Group1 results memory occurs, the ADC proceeds to overwrite the contents with any new conversion results for the Group1, starting with the first location in this memory.
G1_HW_TRIG		Group1 Hardware Triggered. This bit allows the Group1 to be hardware triggered. The Group1 is software triggered by default. For more details on how to trigger a conversion group, refer to Section 22.2.1.6. Any operation mode read/write:
	0	The Group1 is software-triggered. A Group1 conversion starts whenever the Group1 channel select register (ADG1SEL) is written with a non-zero value.
	1	The Group1 is hardware-triggered. A Group1 conversion starts whenever the Group1 channel select register has a non-zero value, and the specified hardware trigger occurs. The hardware trigger for the Group1 is specified in the Group1 Trigger Source register (ADG1SRC).

Table 22-12. ADC Group1 Operating Mode Control Register (ADG1MODECR) Field Descriptions (continued)

Field	Value	Description
G1_8BIT		Group1 8-bit result mode. This field is only applicable when the ADC module is configured to be in the 10-bit ADC module. This field is reserved when the module is configured as a 12-bit ADC module. This bit allows the Group1 conversion results to be read out in an 8-bit format. This bit only applies to the “read from FIFO” mode. The lower 2 bits of the 10-bit conversion result are discarded and the upper 8 bits are shifted right two places to form the 8-bit conversion result. Any operation mode read/write:
	0	The Group1 conversion result is read out as a 10-bit value in the “read from Group1 FIFO” mode.
	1	The Group1 conversion result is read out as an 8-bit value in the “read from Group1 FIFO” mode.
G1_MODE		Group1 Conversion Mode. This bit defines whether the input channels selected for conversion in the Group1 are converted only once, or are continuously converted. Any operation mode read/write:
	0	The channels selected for conversion in the Group1 are converted only once.
	1	The channels selected for conversion in the Group1 are converted continuously.
FRZ_G1		Group1 Freeze Enable. This bit allows a Group1 conversion sequence to be frozen if an Event Group or a Group2 conversion is requested. The Group1 conversion is kept frozen while the Event Group or Group2 conversion is active, and continues from where it was frozen once the Event Group or Group2 conversions are completed. While the Group1 conversion is frozen, the G1_STOP status flag in the ADG1SR register indicates that the Group1 conversions have stopped. This bit gets cleared when the Group1 conversions resume. Any operation mode read/write:
	0	Group1 conversions cannot be frozen. All the channels selected for conversion in the Group1 are converted before the ADC can switch over to servicing any other conversion group.
	1	Group1 conversions are frozen whenever there is a request for conversion from Event Group or Group2.

22.3.7 ADC Group2 Operating Mode Control Register (ADG2MODECR)

ADC Group2 Operating Mode Control Register (ADG2MODECR) is shown in Figure 22-28 and Figure 22-29, described in Table 22-13. As shown, the format of the ADG2MODECR is different based on whether the ADC module is configured to be a 12-bit or a 10-bit ADC module.

Figure 22-28. 12-bit ADC Group2 Operating Mode Control Register (ADG2MODECR) [offset = 18h]

<table>
<tr><td>31</td><td></td><td></td><td></td><td></td><td></td><td></td><td>24</td></tr>
<tr><td colspan="8">Reserved</td></tr>
<tr><td colspan="8">R-0</td></tr>
<tr><td>23</td><td></td><td></td><td></td><td></td><td></td><td></td><td>16</td></tr>
<tr><td colspan="7">Reserved</td><td>No Reset on ChnSel</td></tr>
<tr><td colspan="7">R-0</td><td>R/W-0</td></tr>
<tr><td>15</td><td></td><td></td><td></td><td></td><td>10</td><td>9</td><td>8</td></tr>
<tr><td colspan="6">Reserved</td><td colspan="2">G2_DATA_FMT</td></tr>
<tr><td colspan="6">R-0</td><td colspan="2">R/W-0</td></tr>
<tr><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td></tr>
<tr><td colspan="2">Reserved</td><td>G2_CHID</td><td>OVR_G2_ RAM_IGN</td><td>G2_HW_TRIG</td><td>Reserved</td><td>G2_MODE</td><td>FRZ_G2</td></tr>
<tr><td colspan="2">R-0</td><td>R/W-0</td><td>R/W-0</td><td>R/W-0</td><td>R-0</td><td>R/W-0</td><td>R/W-0</td></tr>
</table>

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Figure 22-29. 10-bit ADC Group2 Operating Mode Control Register (ADG2MODECR) [offset = 18h]

<table>
<tr><td>31</td><td></td><td></td><td></td><td></td><td></td><td></td><td>24</td></tr>
<tr><td colspan="8">Reserved</td></tr>
<tr><td colspan="8">R-0</td></tr>
<tr><td>23</td><td></td><td></td><td></td><td></td><td></td><td></td><td>16</td></tr>
<tr><td colspan="7">Reserved</td><td>No Reset on ChnSel</td></tr>
<tr><td colspan="7">R-0</td><td>R/W-0</td></tr>
<tr><td>15</td><td></td><td></td><td></td><td></td><td></td><td></td><td>8</td></tr>
<tr><td colspan="8">Reserved</td></tr>
<tr><td colspan="8">R-0</td></tr>
<tr><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td></tr>
<tr><td colspan="2">Reserved</td><td>G2_CHID</td><td>OVR_G2_ RAM_IGN</td><td>G2_HW_TRIG</td><td>G2_8BIT</td><td>G2_MODE</td><td>FRZ_G2</td></tr>
<tr><td colspan="2">R-0</td><td>R/W-0</td><td>R/W-0</td><td>R/W-0</td><td>R/W-0</td><td>R/W-0</td><td>R/W-0</td></tr>
</table>

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 22-13. ADC Group 2 Operating Mode Control Register (ADG2MODECR) Field Descriptions

Field	Value	Description
Reserved	0	Reads return 0. Writes have no effect.
No Reset on ChnSel		No Group2 Results Memory Reset on New Channel Select.
		This bit determines whether the group2 results' RAM is reset whenever a non-zero value is written to the group2 channel select register.
		Any operation mode read/write:
	0	Group2 results RAM is reset when a non-zero value is written to group2 channel select register, even if group2 conversions are completed.
	1	Group2 results RAM is not reset when a non-zero value is written to group2 channel select register, and group2 conversions are completed.
		If the group2 conversions are ongoing (active or frozen), then writing a non-zero value to the group2 channel select register will always reset the group2 results RAM.
G2_DATA_FMT		Group2 Read Data Format.
		This field is only applicable when the ADC module is configured to be in the 12-bit ADC module. This field is reserved when the module is configured as a 10-bit ADC module.
		This field determines the format in which the conversion results are read out of the group1 results RAM when using the FIFO interface, that is, when reading from the ADG2BUFFER or ADG2EMUBUFFER locations.
		Any operation mode read/write:
	0	Conversion results are read out in full 12-bit format. This is the default mode.
	1h	Conversion results are read out in 10-bit format. Bits 11-2 of the 12-bit conversion result are returned as the 10-bit conversion result.
	2h	Conversion results are read out in 8-bit format. Bits 11-4 of the 12-bit conversion result are returned as the 8-bit conversion result.
	3h	Reserved. The full 12-bit conversion result is returned if programmed.
G2_CHID		Enable Channel Id for the Group2 conversion results to be read. This bit only affects the "read from FIFO" mode. The ADC always stores the channel id in the results RAM. Any 16-bit read performed in the "read from RAM" mode will return the 5-bit channel id along with the 10-bit conversion result.
		Any operation mode read/write:
	0	Bits 14-10, the channel id field, of the data read from the Group2 results' FIFO is read as 00000b.
	1	Bits 14-10, the channel id field, of the data read from the Group2 results' FIFO contains the number of the ADC analog input to which the conversion result belongs.
OVR_G2_RAM_IGN		This bit allows the ADC module to overwrite the contents of the Group2 results memory under an overrun condition.
		Any operation mode read/write:
	0	The ADC cannot overwrite the contents of the Group2 results memory. When an overrun of this memory occurs, the software needs to read out all the contents of this memory before the ADC is able to write a new conversion result for the Group2.
	1	When an overrun of the Group2 results memory occurs, the ADC proceeds to overwrite the contents with any new conversion results for the Group2, starting with the first location in this memory.
G2_HW_TRIG		Group2 Hardware Triggered. This bit allows the Group2 to be hardware triggered. The Group2 is software triggered by default. For more details on how to trigger a conversion group, refer to Section 22.2.1.6.
		Any operation mode read/write:
	0	The Group2 is software-triggered. A Group2 conversion starts whenever the Group2 channel select register (ADG2SEL) is written with a non-zero value.
	1	The Group2 is hardware-triggered. A Group2 conversion starts whenever the Group2 channel select register has a non-zero value, and the specified hardware trigger occurs. The hardware trigger for the Group2 is specified in the Group2 Trigger Source register (ADG2SRC).

Table 22-13. ADC Group 2 Operating Mode Control Register (ADG2MODECR) Field Descriptions (continued)

Field	Value	Description
G2_8BIT		Group2 8-bit result mode. This field is only applicable when the ADC module is configured to be in the 10-bit ADC module. This field is reserved when the module is configured as a 12-bit ADC module. This bit allows the Group2 conversion results to be read out in an 8-bit format. This bit only applies to the "read from FIFO" mode. The lower 2 bits of the 10-bit conversion result are discarded and the upper 8 bits are shifted right two places to form the 8-bit conversion result. Any operation mode read/write:
	0	The Group2 conversion result is read out as a 10-bit value in the "read from Group2 FIFO" mode.
	1	The Group2 conversion result is read out as an 8-bit value in the "read from Group2 FIFO" mode.
G2_MODE		Group2 Conversion Mode. This bit defines whether the input channels selected for conversion in the Group2 are converted only once, or are continuously converted. Any operation mode read/write:
	0	The channels selected for conversion in the Group2 are converted only once.
	1	The channels selected for conversion in the Group2 are converted continuously.
FRZ_G2		Group2 Freeze Enable. This bit allows a Group2 conversion sequence to be frozen if an Event Group or a Group1 conversion is requested. The Group2 conversion is kept frozen while the Event Group or Group1 conversion is active, and continues from where it was frozen once the Event Group or Group1 conversions are completed. While the Group2 conversion is frozen, the G2_STOP status flag in the ADG2SR register indicates that the Group2 conversions have stopped. This bit gets cleared when the Group2 conversions resume. Any operation mode read/write:
	0	Group2 conversions cannot be frozen. All the channels selected for conversion in the Group2 are converted before the ADC can switch over to servicing any other conversion group.
	1	Group2 conversions are frozen whenever there is a request for conversion from Event Group or Group1.

22.3.8 ADC Event Group Trigger Source Select Register (ADEVSRC)

ADC Event Group Trigger Source Select Register (ADEVSRC) is shown in Figure 22-30 and described in Table 22-14.

Figure 22-30. ADC Event Group Trigger Source Select Register (ADEVSRC) [offset = 1Ch]

31	8
Reserved	
R-0	

7	5	4	3	2	0
Reserved		EV_EDG_BOTH	EV_EDG_SEL	EV_SRC	
R-0		R/W-0	R/W-0	R/W-0	

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 22-14. ADC Event Group Trigger Source Select Register (ADEVSRC) Field Descriptions

Bit	Field	Value	Description
31-5	Reserved	0	Reads return 0. Writes have no effect.
4	EV_EDG_BOTH		EV Group Trigger Edge Polarity Select. This bit configures the event group to be triggered on both rising and falling edge detected on the selected trigger source.
			Any operation mode read/write:
		0	The conversion is triggered only upon detecting an edge defined by the EV_EDG_SEL bit.
		1	The conversion is triggered upon detecting either a rising or falling edge.
3	EV_EDG_SEL		Event Group Trigger Edge Polarity Select. This bit determines the polarity of the transition on the selected source that triggers the Event Group conversion.
			Any operation mode read/write:
		0	A high-to-low transition on the selected source will trigger the Event Group conversion.
		1	A low-to-high transition on the selected source will trigger the Event Group conversion.
2-0	EV_SRC		Event Group Trigger Source.
			Any operation mode read/write:
		0-7h	The ADC module allows a trigger source to be selected for the Event Group from up to eight options. These options are device-specific and the device specification must be referred to identify the actual trigger sources.

22.3.9 ADC Group1 Trigger Source Select Register (ADG1SRC)

ADC Group1 Trigger Source Select Register (ADG1SRC) is shown in Figure 22-31 and described in Table 22-15.

Figure 22-31. ADC Group1 Trigger Source Select Register (ADG1SRC) [offset = 20h]

31	8
Reserved	
R-0	

7	5	4	3	2	0
Reserved		G1_EDG_BOTH	G1_EDG_SEL	G1_SRC	
R-0		R/W-0	R/W-0	R/W-0	

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 22-15. ADC Group1 Trigger Source Select Register (ADG1SRC) Field Descriptions

Bit	Field	Value	Description
31-5	Reserved	0	Reads return 0. Writes have no effect.
4	GI_EDG_BOTH		Group1 Trigger Edge Polarity Select. This bit configures the group1 to be triggered on both rising and falling edge detected on the selected trigger source.
			Any operation mode read/write:
		0	The conversion is triggered only upon detecting an edge defined by the G1_EDG_SEL bit.
		1	The conversion is triggered upon detecting either a rising or falling edge.
3	G1_EDG_SEL		Group1 Trigger Edge Polarity Select. This bit determines the polarity of the transition on the selected source that triggers the Group1 conversion.
			Any operation mode read/write:
		0	A high-to-low transition on the selected source will trigger the Group1 conversion.
		1	A low-to-high transition on the selected source will trigger the Group1 conversion.
2-0	G1_SRC		Group1 Trigger Source.
			Any operation mode read/write:
		0-7h	The ADC module allows a trigger source to be selected for the Group1 from up to eight options. These options are device-specific and the device specification must be referred to identify the actual trigger sources.

22.3.10 ADC Group2 Trigger Source Select Register (ADG2SRC)

ADC Group2 Trigger Source Select Register (ADG2SRC) is shown in Figure 22-32 and described in Table 22-16.

Figure 22-32. ADC Group2 Trigger Source Select Register (ADG2SRC) [offset = 24h]

31 … 8
Reserved
R-0

7 … 5	4	3	2 … 0
Reserved	G2_EDG_BOTH	G2_EDG_SEL	G2_SRC
R-0	R/W-0	R/W-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 22-16. ADC Group2 Trigger Source Select Register (ADG2SRC) Field Descriptions

Bit	Field	Value	Description
31-5	Reserved	0	Reads return 0. Writes have no effect.
4	G2_EDG_BOTH		Group2 Trigger Edge Polarity Select. This bit configures the group2 to be triggered on both rising and falling edge detected on the selected trigger source.
			Any operation mode read/write:
		0	The conversion is triggered only upon detecting an edge defined by the G2_EDG_SEL bit.
		1	The conversion is triggered upon detecting either a rising or falling edge.
3	G2_EDG_SEL		Group2 Trigger Edge Polarity Select. This bit determines the polarity of the transition on the selected source that triggers the Group2 conversion.
			Any operation mode read/write:
		0	A high-to-low transition on the selected source will trigger the Group2 conversion.
		1	A low-to-high transition on the selected source will trigger the Group2 conversion.
2-0	G2_SRC		Group2 Trigger Source.
			Any operation mode read/write:
		0-7h	The ADC module allows a trigger source to be selected for the Group2 from up to eight options. These options are device-specific and the device specification must be referred to identify the actual trigger sources.

22.3.11 ADC Event Interrupt Enable Control Register (ADEVINTENA)

ADC Event Group Interrupt Enable Control Register (ADEVINTENA) is shown in Figure 22-33 and described in Table 22-17.

Figure 22-33. ADC Event Group Interrupt Enable Control Register (ADEVINTENA) [offset = 28h]

31	8
Reserved	
R-0	

7 4	3	2	1	0
Reserved	EV_END_ INT_EN	Reserved	EV_OVR_ INT_EN	EV_THR_ INT_EN
R-0	R/W-0	R-0	R/W-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 22-17. ADC Event Group Interrupt Enable Control Register (ADEVINTENA) Field Descriptions

Bit	Field	Value	Description
31-4	Reserved	0	Reads return 0. Writes have no effect.
3	EV_END_INT_EN		Event Group Conversion End Interrupt Enable. Refer to Section 22.2.3.1 for more details on the conversion end interrupts.
			Any operation mode read/write:
		0	No interrupt is generated when conversion of all the channels selected for conversion in the Event Group is done.
		1	An Event Group conversion end interrupt is generated when conversion of all the channels selected for conversion in the Event Group is done.
2	Reserved	0	Reads return 0. Writes have no effect.
1	EV_OVR_INT_EN		Event Group Memory Overrun Interrupt Enable. A memory overrun occurs when the ADC tries to write a new conversion result to the Event Group results memory which is already full. For more details on the overrun interrupts, refer to Section 22.2.3.3.
			Any operation mode read/write:
		0	No interrupt is generated if an Event Group memory overrun occurs.
		1	An Event Group memory overrun interrupt is generated if an Event Group memory overrun condition occurs.
0	EV_THR_INT_EN		Event Group Threshold Interrupt Enable. An Event Group threshold interrupt occurs when the programmed Event Group threshold counter counts down to 0. Refer to Section 22.2.3.2 for more details.
			Any operation mode read/write:
		0	No interrupt is generated if the Event Group threshold counter reaches 0.
		1	An Event Group threshold interrupt is generated if the Event Group threshold counter reaches 0.

22.3.12 ADC Group1 Interrupt Enable Control Register (ADG1INTENA)

ADC Group1 Interrupt Enable Control Register (ADG1INTENA) is shown in Figure 22-34 and described in Table 22-18.

Figure 22-34. ADC Group1 Interrupt Enable Control Register (ADG1INTENA) [offset = 2Ch]

31–8
Reserved
R-0

7–4	3	2	1	0
Reserved	G1_END_ INT_EN	Reserved	G1_OVR_ INT_EN	G1_THR_ INT_EN
R-0	R/W-0	R-0	R/W-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 22-18. ADC Group1 Interrupt Enable Control Register (ADG1INTENA) Field Descriptions

Bit	Field	Value	Description
31-4	Reserved	0	Reads return 0. Writes have no effect.
3	G1_END_INT_EN		Group1 Conversion End Interrupt Enable. Refer to Section 22.2.3.1 for more details on the conversion end interrupts. Any operation mode read/write:
		0	No interrupt is generated when conversion of all the channels selected for conversion in the Group1 is done.
		1	A Group1 conversion end interrupt is generated when conversion of all the channels selected for conversion in the Group1 is done.
2	Reserved	0	Reads return 0. Writes have no effect.
1	G1_OVR_INT_EN		Group1 Memory Overrun Interrupt Enable. A memory overrun occurs when the ADC tries to write a new conversion result to the Group1 results memory which is already full. For more details on the overrun interrupts Refer to Section 22.2.3.3. Any operation mode read/write:
		0	No interrupt is generated if a Group1 memory overrun occurs.
		1	A Group1 memory overrun interrupt is generated if a Group1 memory overrun condition occurs.
0	G1_THR_INT_EN		Group1 Threshold Interrupt Enable. A Group1 threshold interrupt occurs when the programmed Group1 threshold counter counts down to 0. Refer to Section 22.2.3.2 for more details. Any operation mode read/write:
		0	No interrupt is generated if the Group1 threshold counter reaches 0.
		1	A Group1 threshold interrupt is generated if the Group1 threshold counter reaches 0.

22.3.13 ADC Group2 Interrupt Enable Control Register (ADG2INTENA)

ADC Group2 Interrupt Enable Control Register (ADG2INTENA) is shown in Figure 22-35 and described in Table 22-19.

Figure 22-35. ADC Group2 Interrupt Enable Control Register (ADG2INTENA) [offset = 30h]

31–8
Reserved
R-0

7–4	3	2	1	0
Reserved	G2_END_INT_EN	Reserved	G2_OVR_INT_EN	G2_THR_INT_EN
R-0	R/W-0	R-0	R/W-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 22-19. ADC Group2 Interrupt Enable Control Register (ADG2INTENA) Field Descriptions

Bit	Field	Value	Description
31-4	Reserved	0	Reads return 0. Writes have no effect.
3	G2_END_INT_EN		Group2 Conversion End Interrupt Enable. Refer to Section 22.2.3.1 for more details on the conversion end interrupts.
			Any operation mode read/write:
		0	No interrupt is generated when conversion of all the channels selected for conversion in the Group2 is done.
		1	A Group2 conversion end interrupt is generated when conversion of all the channels selected for conversion in the Group2 is done.
2	Reserved	0	Reads return 0. Writes have no effect.
1	G2_OVR_INT_EN		Group2 Memory Overrun Interrupt Enable. A memory overrun occurs when the ADC tries to write a new conversion result to the Group2 results memory which is already full. For more details on the overrun interrupts, refer to Section 22.2.3.3.
			Any operation mode read/write:
		0	No interrupt is generated if a Group2 memory overrun occurs.
		1	A Group2 memory overrun interrupt is generated if a Group2 memory overrun condition occurs.
0	G2_THR_INT_EN		Group2 Threshold Interrupt Enable. A Group2 threshold interrupt occurs when the programmed Group2 threshold counter counts down to 0. Refer to Section 22.2.3.2 for more details.
			Any operation mode read/write:
		0	No interrupt is generated if the Group2 threshold counter reaches 0.
		1	A Group2 threshold interrupt is generated if the Group2 threshold counter reaches 0.

22.3.14 ADC Event Group Interrupt Flag Register (ADEVINTFLG)

ADC Event Group Interrupt Enable Control Register (ADEVINTENA) is shown in Figure 22-36 and described in Table 22-20.

Figure 22-36. ADC Event Group Interrupt Flag Register (ADEVINTFLG) [offset = 34h]

31-8
Reserved
R-0

7-4	3	2	1	0
Reserved	EV_END	EV_MEM_ EMPTY	EV_MEM_ OVERRUN	EV_THR_ INT_FLG
R-0	R/W1C-0	R-1	R-0	R/W1C-0

LEGEND: R/W = Read/Write; R = Read only; W1C = Write 1 to clear; -*n* = value after reset

Table 22-20. ADC Event Group Interrupt Flag Register (ADEVINTFLG) Field Descriptions

Bit	Field	Value	Description
31-4	Reserved	0	Reads return 0. Writes have no effect.
3	EV_END		Event Group Conversion End. This bit will be set only if the Event Group conversions are configured to be in the single-conversion mode.
			Any operation mode read:
		0	All the channels selected for conversion in the Event Group have not yet been converted.
		1	All the channels selected for conversion in the Event Group have been converted. An Event Group conversion end interrupt is generated, if enabled, when this bit gets set.
			This bit can be cleared by any one of the following ways: • By writing a 1 to this bit • By writing a 1 to the Event Group status register (ADEVSR) bit 0 (EV_END) • By reading one conversion result from the Event Group results' memory in the “read from FIFO” mode • By writing a new set of channels to the Event Group channel select register
2	EV_MEM_EMPTY		Event Group Results Memory Empty. This is a read-only bit; writes have no effect. It is not a source of an interrupt from the ADC module.
			Any operation mode read:
		0	The Event Group results memory is not empty.
		1	The Event Group results memory is empty.
1	EV_MEM_OVERRUN		Event Group Memory Overrun. This is a read-only bit; writes have no effect.
			Any operation mode read:
		0	Event Group results memory has not overrun.
		1	Event Group results memory has overrun.
0	EV_THR_INT_FLG		Event Group Threshold Interrupt Flag.
			Any operation mode read:
		0	The number of conversions completed for the Event Group is smaller than the threshold programmed in the Event Group interrupt threshold register.
		1	The number of conversions completed for the Event Group is equal to or greater than the threshold programmed in the Event Group interrupt threshold register.
			This bit can be cleared by writing a 1; writing a 0 has no effect.

22.3.15 ADC Group1 Interrupt Flag Register (ADG1INTFLG)

ADC Group1 Interrupt Flag Register (ADG1INTFLG) is shown in Figure 22-37 and described in Table 22-21.

Figure 22-37. ADC Group1 Interrupt Flag Register (ADG1INTFLG) [offset = 38h]

31–8
Reserved
R-0

7–4	3	2	1	0
Reserved	G1_END	G1_MEM_ EMPTY	G1_MEM_ OVERRUN	G1_THR_ INT_FLG
R-0	R/W1C-0	R-1	R-0	R/W1C-0

LEGEND: R/W = Read/Write; R = Read only; W1C = Write 1 to clear; -*n* = value after reset

Table 22-21. ADC Group1 Interrupt Flag Register (ADG1INTFLG) Field Descriptions

Bit	Field	Value	Description
31-4	Reserved	0	Reads return 0. Writes have no effect.
3	G1_END		Group1 Conversion End. This bit will be set only if the Group1 conversions are configured to be in the single-conversion mode.
			Any operation mode read:
		0	All the channels selected for conversion in the Group1 have not yet been converted.
		1	All the channels selected for conversion in the Group1 have been converted. A Group1 conversion end interrupt is generated, if enabled, when this bit gets set.
			This bit can be cleared by any one of the following ways: • By writing a 1 to this bit • By writing a 1 to the Group1 status register (ADG1SR) bit 0 (G1_END) • By reading one conversion result from the Group1 results' memory in the "read from FIFO" mode • By writing a new set of channels to the Group1 channel select register
2	G1_MEM_EMPTY		Group1 Results Memory Empty. This is a read-only bit; writes have no effect. It is not a source of an interrupt from the ADC module.
			Any operation mode read:
		0	The Group1 results memory is not empty.
		1	The Group1 results memory is empty.
1	G1_MEM_OVERRUN		Group1 Memory Overrun. This is a read-only bit; writes have no effect.
			Any operation mode read:
		0	Group1 results memory has not overrun.
		1	Group1 results memory has overrun.
0	G1_THR_INT_FLG		Group1 Threshold Interrupt Flag.
			Any operation mode read:
		0	The number of conversions completed for the Group1 is smaller than the threshold programmed in the Group1 interrupt threshold register.
		1	The number of conversions completed for the Group1 is equal to or greater than the threshold programmed in the Group1 interrupt threshold register.
			This bit can be cleared by writing a 1; writing a 0 has no effect.

22.3.16 ADC Group2 Interrupt Flag Register (ADG2INTFLG)

ADC Group2 Interrupt Flag Register (ADG2INTFLG) is shown in Figure 22-38 and described in Table 22-22.

Figure 22-38. ADC Group2 Interrupt Flag Register (ADG2INTFLG) [offset = 3Ch]

31 8
Reserved
R-0

7 4	3	2	1	0
Reserved	G2_END	G2_MEM_ EMPTY	G2_MEM_ OVERRUN	G2_THR_ INT_FLG
R-0	R/W1C-0	R-1	R-0	R/W1C-0

LEGEND: R/W = Read/Write; R = Read only; W1C = Write 1 to clear; -*n* = value after reset

Table 22-22. ADC Group2 Interrupt Flag Register (ADG2INTFLG) Field Descriptions

Bit	Field	Value	Description
31-4	Reserved	0	Reads return 0. Writes have no effect.
3	G2_END		Group2 Conversion End. This bit will be set only if the Group2 conversions are configured to be in the single-conversion mode.
			Any operation mode read:
		0	All the channels selected for conversion in the Group2 have not yet been converted.
		1	All the channels selected for conversion in the Group2 have been converted. A Group2 conversion end interrupt is generated, if enabled, when this bit gets set.
			This bit can be cleared by any one of the following ways: • By writing a 1 to this bit • By writing a 1 to the Group2 status register (ADG2SR) bit 0 (G2_END) • By reading one conversion result from the Group2 results' memory in the “read from FIFO” mode • By writing a new set of channels to the Group2 channel select register
2	G2_MEM_EMPTY		Group2 Results Memory Empty. This is a read-only bit; writes have no effect. It is not a source of an interrupt from the ADC module.
			Any operation mode read:
		0	The Group2 results memory is not empty.
		1	The Group2 results memory is empty.
1	G2_MEM_OVERRUN		Group2 Memory Overrun. This is a read-only bit; writes have no effect.
			Any operation mode read:
		0	Group2 results memory has not overrun.
		1	Group2 results memory has overrun.
0	G2_THR_INT_FLG		Group2 Threshold Interrupt Flag.
			Any operation mode read:
		0	The number of conversions completed for the Group2 is smaller than the threshold programmed in the Group2 interrupt threshold register.
		1	The number of conversions completed for the Group2 is equal to or greater than the threshold programmed in the Group2 interrupt threshold register.
			This bit can be cleared by writing a 1; writing a 0 has no effect.

22.3.17 ADC Event Group Threshold Interrupt Control Register (ADEVTHRINTCR)

ADC Event Group Threshold Interrupt Control Register (ADEVTHRINTCR) is shown in Figure 22-39 and described in Table 22-23.

Figure 22-39. ADC Event Group Threshold Interrupt Control Register (ADEVTHRINTCR) [offset = 40h]

31 16	15 9	8 0
Reserved	Sign Extension	EV_THR
R-0	R-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 22-23. ADC Event Group Threshold Interrupt Control Register (ADEVTHRINTCR) Field Descriptions

Bit	Field	Value	Description
31-16	Reserved	0	Reads return 0. Writes have no effect.
15-9	Sign Extension		These bits always read the same as EV_THR bit 8 of this register.
8-0	EV_THR		Event Group Threshold Counter. Before ADC conversions begin on the Event Group, this field is initialized to the number of conversion results that the Event Group memory should contain before interrupting the CPU. This counter decrements when the ADC module writes a new conversion result to the Event Group results' memory. The counter increments for each read of a conversion result from the Event Group results' memory in the "read from FIFO" mode. The threshold counter is not affected for a direct read from the Event Group results' memory. Also, a simultaneous ADC write and a CPU/DMA read from the Event Group FIFO will leave the threshold counter unchanged. In case of an Event Group Results' memory overrun condition, if new conversion results are not allowed to overwrite the existing memory contents, then the Event Group threshold counter is not decremented. Refer to Section 22.2.3.2 for more details on the threshold interrupts.

22.3.18 ADC Group1 Threshold Interrupt Control Register (ADG1THRINTCR)

ADC Group1 Threshold Interrupt Control Register (ADG1THRINTCR) is shown in Figure 22-40 and described in Table 22-24.

Figure 22-40. ADC Group1 Threshold Interrupt Control Register (ADG1THRINTCR) [offset = 44h]

31 16	15 9	8 0
Reserved	Sign Extension	G1_THR
R-0	R-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 22-24. ADC Group1 Threshold Interrupt Control Register (ADG1THRINTCR) Field Descriptions

Bit	Field	Value	Description
31-16	Reserved	0	Reads return 0. Writes have no effect.
15-9	Sign Extension		These bits always read the same as G1_THR bit 8 of this register.
8-0	G1_THR		Group1 Threshold Counter. Before ADC conversions begin on the Group1, this field is initialized to the number of conversion results that the Group1 memory should contain before interrupting the CPU. This counter decrements when the ADC module writes a new conversion result to the Group1 results' memory. The counter increments for each read of a conversion result from the Group1 results' memory in the "read from FIFO" mode. The threshold counter is not affected for a direct read from the group1 results' memory. Also, a simultaneous ADC write and a CPU/DMA read from the Group1 FIFO will leave the threshold counter unchanged. In case of an Group1 Results' memory overrun condition, if new conversion results are not allowed to overwrite the existing memory contents, then the Group1 threshold counter is not decremented. Refer to Section 22.2.3.2 for more details on the threshold interrupts.

22.3.19 ADC Group2 Threshold Interrupt Control Register (ADG2THRINTCR)

The ADC Group2 Threshold Interrupt Control Register (ADG2THRINTCR) is shown in Figure 22-41 and described in Table 22-25.

Figure 22-41. ADC Group2 Threshold Interrupt Control Register (ADG2THRINTCR) [offset = 48h]

31 16	15 9	8 0
Reserved	Sign Extension	G2_THR
R-0	R-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 22-25. ADC Group2 Threshold Interrupt Control Register (ADG2THRINTCR) Field Descriptions

Bit	Field	Value	Description
31-16	Reserved	0	Reads return 0. Writes have no effect.
15-9	Sign Extension		These bits always read the same as G2_THR bit 8 of this register.
8-0	G2_THR		Group2 Threshold Counter. Before ADC conversions begin on the Group2, this field is initialized to the number of conversion results that the Group2 memory should contain before interrupting the CPU. This counter decrements when the ADC module writes a new conversion result to the Group2 results' memory. The counter increments for each read of a conversion result from the Group2 results' memory in the "read from FIFO" mode. The threshold counter is not affected for a direct read from the group2 results' memory. Also, a simultaneous ADC write and a CPU/DMA read from the Group2 FIFO will leave the threshold counter unchanged. In case of an Group2 Results' memory overrun condition, if new conversion results are not allowed to overwrite the existing memory contents, then the Grou21 threshold counter is not decremented. Refer to Section 22.2.3.2 for more details on the threshold interrupts.

22.3.20 ADC Event Group DMA Control Register (ADEVDMACR)

ADC Event Group DMA Control Register (ADEVDMACR) is shown in Figure 22-42 and described in Table 22-26.

Figure 22-42. ADC Event Group DMA Control Register (ADEVDMACR) [offset = 4Ch]

31-25	24-16
Reserved	EV_BLOCKS
R-0	R/W-0

15-8
Reserved
R-0

7-4	3	2	1	0
Reserved	DMA_EV_END	EV_BLK_XFER	Reserved	EV_DMA_EN
R-0	R/W-0	R/W-0	R-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 22-26. ADC Event Group DMA Control Register (ADEVDMACR) Field Descriptions

Bit	Field	Value	Description
31-25	Reserved	0	Reads return 0. Writes have no effect.
24-16	EV_BLOCKS		Number of Event Group Result buffers to be transferred using DMA if the ADC module is configured to generate a DMA request. If the Event Group is configured to use the block transfer mode of the DMA module, then the ADC module generates a DMA request after the Event Group results' memory accumulates EV_BLOCKS number of conversion results. This feature is designed to be used in place of the threshold interrupt for the Event Group. As a result, the EV_THR field of the Event Group Interrupt Threshold Control Register and the EV_BLOCKS field of the Event Group DMA Control Register are the same. Any operation mode read/write:
		0	No DMA transfer occurs even if EV_BLK_XFER is set to 1.
		1h-1FFh	One DMA request is generated if the EV_BLK_XFER is set to 1 and the specified number of Event Group conversion results have been accumulated.
15-4	Reserved	0	Reads return 0. Writes have no effect.
3	DMA_EV_END		Event Group Conversion End DMA Transfer Enable. Any operation mode read:
		0	ADC module generates a DMA request for each write to the Event group results RAM if EV_DMA_EN is set.
		1	ADC module generates a DMA request when the ADC has completed the conversions for all channels selected for conversion in the event group. If DMA_EV_END bit is set to 1, EV_DMA_EN bit is ignored and DMA requests will be generated every time the DMA_EV_END flag in the event group status register is set. The DMA_EV_END bit must be set before enabling conversions for the event group.
2	EV_BLK_XFER		Event Group Block DMA Transfer Enable. Any operation mode read:
		0	ADC module generates a DMA request for each write to the Event Group memory if EV_DMA_EN is set.
		1	ADC module generates a DMA request when the ADC has written EV_BLOCKS number of buffers into the Event Group memory. If EV_BLK_XFER bit is set to 1, EV_DMA_EN bit is ignored and DMA requests will be generated every time the Threshold Counter reaches 0 from a count value of 1.
1	Reserved	0	Reads return 0. Writes have no effect.

Table 22-26. ADC Event Group DMA Control Register (ADEVDMACR) Field Descriptions (continued)

Bit	Field	Value	Description
0	EV_DMA_EN		Event Group DMA Transfer Enable.
			Any operation mode read:
		0	ADC module does not generate a DMA request when it writes the conversion result to the Event Group memory.
		1	ADC module generates a DMA transfer when the ADC has written to the Event Group memory. The EV_BLK_XFER bit must be cleared to 0 for this DMA request to be generated.

22.3.21 ADC Group1 DMA Control Register (ADG1DMACR)

ADC Group1 DMA Control Register (ADG1DMACR) is shown in Figure 22-43 and described in Table 22-27.

Figure 22-43. ADC Group1 DMA Control Register (ADG1DMACR) [offset = 50h]

31 25	24 16
Reserved	G1_BLOCKS
R-0	R/W-0

15 8
Reserved
R-0

7 4	3	2	1	0
Reserved	DMA_G1_END	G1_BLK_XFER	Reserved	G1_DMA_EN
R-0	R/W-0	R/W-0	R-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 22-27. ADC Group1 DMA Control Register (ADG1DMACR) Field Descriptions

Bit	Field	Value	Description
31-25	Reserved	0	Reads return 0. Writes have no effect.
24-16	G1_BLOCKS		Number of Group1 Result buffers to be transferred using DMA if the ADC module is configured to generate a DMA request. If the Group1 is configured to use the block transfer mode of the DMA module, then the ADC module generates a DMA request after the Group1 results' memory accumulates G1_BLOCKS number of conversion results. This feature is designed to be used in place of the threshold interrupt for the Group1. As a result, the G1_THR field of the Group1 Interrupt Threshold Control Register and the G1_BLOCKS field of the Group1 DMA Control Register are the same. Any operation mode read/write:
		0	No DMA transfer occurs even if G1_BLK_XFER is set to 1.
		1h-1FFh	One DMA request is generated if the G1_BLK_XFER is set to 1 and the specified number of Group1 conversion results have been accumulated.
15-4	Reserved	0	Reads return 0. Writes have no effect.
3	DMA_G1_END		Group1 Conversion End DMA Transfer Enable. Any operation mode read:
		0	ADC module generates a DMA request for each write to the group1 results RAM if G1_DMA_EN is set.
		1	ADC module generates a DMA request when the ADC has completed the conversions for all channels selected for conversion in the group1.
			If DMA_G1_END bit is set to 1, G1_DMA_EN bit is ignored and DMA requests will be generated every time the DMA_G1_END flag in the group 1 status register is set. The DMA_G1_END bit must be set before enabling conversions for the group 1.
2	G1_BLK_XFER		Group1 Block DMA Transfer Enable. Any operation mode read:
		0	ADC module generates a DMA request for each write to the Group1 memory if G1_DMA_EN is set.
		1	ADC module generates a DMA request when the ADC has written G1_BLOCKS number of buffers into the Group1 memory.
			If G1_BLK_XFER bit is set to 1, G1_DMA_EN bit is ignored and DMA requests will be generated every time the Threshold Counter reaches 0 from a count value of 1.
1	Reserved	0	Reads return 0. Writes have no effect.

Table 22-27. ADC Group1 DMA Control Register (ADG1DMACR) Field Descriptions (continued)

Bit	Field	Value	Description
0	G1_DMA_EN		Group1 DMA Transfer Enable.
			Any operation mode read:
		0	ADC module does not generate a DMA request when it writes the conversion result to the Group1 memory.
		1	ADC module generates a DMA transfer when the ADC has written to the Group1 memory. The G1_BLK_XFER bit must be cleared to 0 for this DMA request to be generated.

22.3.22 ADC Group2 DMA Control Register (ADG2DMACR)

ADC Group2 DMA Control Register (ADG2DMACR) is shown in Figure 22-44 and described in Table 22-28.

Figure 22-44. ADC Group2 DMA Control Register (ADG2DMACR) [offset = 54h]

31 25	24 16
Reserved	G2_BLOCKS
R-0	R/W-0

15 8
Reserved
R-0

7 4	3	2	1	0
Reserved	DMA_G2_END	G2_BLK_XFER	Reserved	G2_DMA_EN
R-0	R/W-0	R/W-0	R-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 22-28. ADC Group2 DMA Control Register (ADG2DMACR) Field Descriptions

Bit	Field	Value	Description
31-25	Reserved	0	Reads return 0. Writes have no effect.
24-16	G2_BLOCKS		Number of Group2 Result buffers to be transferred using DMA if the ADC module is configured to generate a DMA request. If the Group2 is configured to use the block transfer mode of the DMA module, then the ADC module generates a DMA request after the Group2 results' memory accumulates G2_BLOCKS number of conversion results.
			This feature is designed to be used in place of the threshold interrupt for the Group2. As a result, the G2_THR field of the Group2 Interrupt Threshold Control Register and the G2 BLOCKS field of the Group2 DMA Control Register are the same.
			Any operation mode read/write:
		0	No DMA transfer occurs even if G2_BLK_XFER is set to 1.
		1h-1FFh	One DMA request is generated if the G2_BLK_XFER is set to 1 and the specified number of Group2 conversion results have been accumulated.
15-4	Reserved	0	Reads return 0. Writes have no effect.
3	DMA_G2_END		Group2 Conversion End DMA Transfer Enable.
			Any operation mode read:
		0	ADC module generates a DMA request for each write to the group2 results RAM if G2_DMA_EN is set.
		1	ADC module generates a DMA request when the ADC has completed the conversions for all channels selected for conversion in the group2.
			If DMA_G2_END bit is set to 1, G2_DMA_EN bit is ignored and DMA requests will be generated every time the DMA_G2_END flag in the group 2 status register is set. The DMA_G2_END bit must be set before enabling conversions for the group 2.
2	G2_BLK_XFER		Group2 Block DMA Transfer Enable.
			Any operation mode read:
		0	ADC module generates a DMA request for each write to the Group2 memory if G2_DMA_EN is set.
		1	ADC module generates a DMA request when the ADC has written G2_BLOCKS number of buffers into the Group2 memory.
			If G2_BLK_XFER bit is set to 1, G2_DMA_EN bit is ignored and DMA requests will be generated every time the Threshold Counter reaches 0 from a count value of 1.
1	Reserved	0	Reads return 0. Writes have no effect.

Table 22-28. ADC Group2 DMA Control Register (ADG2DMACR) Field Descriptions (continued)

Bit	Field	Value	Description
0	G2_DMA_EN		Group2 DMA Transfer Enable.
			Any operation mode read:
		0	ADC module does not generate a DMA request when it writes the conversion result to the Group2 memory.
		1	ADC module generates a DMA transfer when the ADC has written to the Group2 memory. The G2_BLK_XFER bit must be cleared to 0 for this DMA request to be generated.

22.3.23 ADC Results Memory Configuration Register (ADBNDCR)

ADC Results Memory Configuration Register (ADBNDCR) [offset = 0x58] is shown in Figure 22-45 and described in Table 22-29.

Refer to Section 22.2.7 for further details on how the conversion results are stored in the ADC results' RAM.

Figure 22-45. ADC Results Memory Configuration Register (ADBNDCR) [offset = 58h]

31 – 25	24 – 16
Reserved	BNDA
R-0	R/W-0

15 – 9	8 – 0
Reserved	BNDB
R-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 22-29. ADC Results Memory Configuration Register (ADBNDCR) Field Descriptions

Bit	Field	Value	Description
31-25	Reserved	0	Reads return 0. Writes have no effect.
24-16	BNDA		Buffer Boundary A. These bits determine the memory available for the Event Group conversion results. The memory available is specified in terms of pairs of result buffers. Any operation mode read/write:
		0	Event Group conversions are not required. If Event Group conversions are performed with the BNDA value of 0, then the Event Group memory size will default to 1024 words. For proper usage of the ADC results memory, configure the BNDA value to be non-zero and lower than the BNDB value.
		0-1FFh	A total of (2 × BNDA) buffers are available in the ADC results memory for storing Event Group conversion results.
15-9	Reserved	0	Reads return 0. Writes have no effect.
8-0	BNDB		Buffer Boundary B. These bits specify the number of buffers allocated for the Event Group plus the number of buffers allocated for the Group1. The number of buffer pairs allocated for storing Group1 conversion results can be determined by subtracting BNDA from BNDB. As a result, BNDB must always be specified as greater than or equal to BNDA. Any operation mode read/write:
		0	Event Group as well as Group1 conversions are not required.
		0-1FFh	A total of 2 × (BNDB - BNDA) buffers are available in the ADC results memory for storing Group1 conversion results.

22.3.24 ADC Results Memory Size Configuration Register (ADBNDEND)

ADC Results Memory Size Configuration Register (ADBNDEND) is shown in Figure 22-46 and described in Table 22-30.

Figure 22-46. ADC Results Memory Size Configuration Register (ADBNDEND) [offset = 5Ch]

31-17	16
Reserved	BUF_INIT_ACTIVE
R-0	R-0

15-3	2-0
Reserved	BNDEND
R-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 22-30. ADC Results Memory Size Configuration Register (ADBNDEND) Field Descriptions

Bit	Field	Value	Description
31-17	Reserved	0	Reads return 0. Writes have no effect.
16	BUF_INIT_ACTIVE		ADC Results Memory Auto-initialization Status.
			Any operation mode read/write:
		0	ADC Results Memory is currently not being initialized, and the ADC is available. If this bit is read as '0' after triggering an auto-initialization of the ADC results memory, then the ADC results memory has been completely initialized to zeros. For devices requiring parity checking on the ADC results memory, the parity bit in the results memory will also be initialized according to the parity polarity. The parity polarity as well as the auto-initialization process is controlled by the System module. Please refer to Chapter 2 for more details.
		1	ADC results memory is being initialized, and the ADC is not available for conversion.
15-3	Reserved	0	Reads return 0. Writes have no effect.
2-0	BNDEND		Buffer Boundary End. These bits specify the total number of memory buffers available for storing the ADC conversion results. These bits should be programmed to match the number of ADC conversion result buffers required to be used for the application.
			Any operation mode read/write:
		0	16 words available for storing ADC conversion results.
		1h	32 words available for storing ADC conversion results.
		2h	64 words available for storing ADC conversion results. This is the maximum configuration allowed since the device supports 64 buffers each for ADC1 as well as ADC2.
		4h-7h	Reserved. These combinations must not be used.

22.3.25 ADC Event Group Sampling Time Configuration Register (ADEVSAMP)

ADC Event Group Sampling Time Configuration Register (ADEVSAMP) is shown in Figure 22-47 and described in Table 22-31.

Figure 22-47. ADC Event Group Sampling Time Configuration Register (ADEVSAMP) [offset = 60h]

31	12	11	0
Reserved		EV_ACQ	
R-0		R/W-0	

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 22-31. ADC Event Group Sampling Time Configuration Register (ADEVSAMP) Field Descriptions

Bit	Field	Value	Description
31-12	Reserved	0	Reads return 0. Writes have no effect.
11-0	EV_ACQ		Event Group Acquisition Time. These bits define the sampling window (SW) for the Event Group conversions. SW = EV_ACQ + 2 in terms of ADCLK cycles. There are two factors that determine the minimum sampling window value required: First, the ADC module design requires that SW >= 3 ADCLK cycles. Second, the ADC input impedance necessitates a certain minimum sampling time. This needs to be assured by configuring the EV_ACQ value properly considering the frequency of the ADCLK signal. Refer to the device datasheet to determine the minimum sampling time for this device.

22.3.26 ADC Group1 Sampling Time Configuration Register (ADG1SAMP)

ADC Group1 Sampling Time Configuration Register (ADG1SAMP) is shown in Figure 22-48 and described in Table 22-32.

Figure 22-48. ADC Group1 Sampling Time Configuration Register (ADG1SAMP) [offset = 64h]

31	12	11	0
Reserved		G1_ACQ	
R-0		R/W-0	

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 22-32. ADC Group1 Sampling Time Configuration Register (ADG1SAMP) Field Descriptions

Bit	Field	Value	Description
31-12	Reserved	0	Reads return 0. Writes have no effect.
11-0	G1_ACQ		Group1 Acquisition Time. These bits define the sampling window (SW) for the Group1 conversions. SW = G1_ACQ + 2 in terms of ADCLK cycles. There are two factors that determine the minimum sampling window value required: First, the ADC module design requires that SW >= 3 ADCLK cycles. Second, the ADC input impedance necessitates a certain minimum sampling time. This needs to be assured by configuring the G1_ACQ value properly considering the frequency of the ADCLK signal. Refer to the device datasheet to determine the minimum sampling time for this device.

22.3.27 ADC Group2 Sampling Time Configuration Register (ADG2SAMP)

ADC Group2 Sampling Time Configuration Register (ADG2SAMP) is shown in Figure 22-49 and described in Table 22-33.

Figure 22-49. ADC Group2 Sampling Time Configuration Register (ADG2SAMP) [offset = 68h]

31 12	11 0
Reserved	G2_ACQ
R-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 22-33. ADC Group2 Sampling Time Configuration Register (ADG2SAMP) Field Descriptions

Bit	Field	Value	Description
31-12	Reserved	0	Reads return 0. Writes have no effect.
11-0	G2_ACQ		Group2 Acquisition Time. These bits define the sampling window (SW) for the Group2 conversions. SW = G2_ACQ + 2 in terms of ADCLK cycles. There are two factors that determine the minimum sampling window value required: First, the ADC module design requires that SW >= 3 ADCLK cycles. Second, the ADC input impedance necessitates a certain minimum sampling time. This needs to be assured by configuring the G2_ACQ value properly considering the frequency of the ADCLK signal. Refer to the device datasheet to determine the minimum sampling time for this device.

22.3.28 ADC Event Group Status Register (ADEVSR)

ADC Event Group Status Register (ADEVSR) is shown in Figure 22-50 and described in Table 22-34.

Figure 22-50. ADC Event Group Status Register (ADEVSR) [offset = 6Ch]

31	8
Reserved	
R-0	

7	4	3	2	1	0
Reserved		EV_MEM_ EMPTY	EV_BUSY	EV_STOP	EV_END
R-0		R-1	R-0	R-0	R/W1C-0

LEGEND: R/W = Read/Write; R = Read only; W1C = Write 1 to clear; -*n* = value after reset

Table 22-34. ADC Event Group Status Register (ADEVSR) Field Descriptions

Bit	Field	Value	Description
31-4	Reserved	0	Reads return 0. Writes have no effect.
3	EV_MEM_EMPTY		Event Group Results Memory Empty. This bit can be effectively used only when the conversion results are read out of the Event Group results memory in the "read from FIFO" mode.
			Any operation mode read:
		0	The Event Group results memory has valid conversion results.
		1	The Event Group results memory is empty, or does not contain any unread conversion results.
2	EV_BUSY		Event Group Conversion Busy.
			Any operation mode read:
		0	Event Group conversions are neither in progress nor frozen.
		1	Event Group conversions are either in progress, or are frozen for servicing some other group. This bit will always be set when the Event Group is configured to be in the continuous conversion mode.
1	EV_STOP		Event Group Conversion Stopped.
			Any operation mode read:
		0	Event Group conversions are not currently frozen.
		1	Event Group conversions are currently frozen.
0	EV_END		Event Group Conversions Ended.
			Any operation mode read:
		0	Event Group conversions have either not been started or have not yet completed since the last time this status bit was cleared.
		1	The conversion for all the channels selected in the Event Group has completed. This bit can be cleared under the following conditions: • By reading a conversion result from the Event Group results memory in the "read from FIFO" mode. • By writing a new value to the Event Group channel select register (ADEVSEL). • By writing a 1 to this bit. • By disabling the ADC module by clearing the ADC_EN bit in the ADC operating mode control register (ADOPMODECR).

22.3.29 ADC Group1 Status Register (ADG1SR)

ADC Group1 Status Register (ADG1SR) is shown in Figure 22-51 and described in Table 22-35.

Figure 22-51. ADC Group1 Status Register (ADG1SR) [offset = 70h]

31–8
Reserved
R-0

7–4	3	2	1	0
Reserved	G1_MEM_ EMPTY	G1_BUSY	G1_STOP	G1_END
R-0	R-1	R-0	R-0	R/W1C-0

LEGEND: R/W = Read/Write; R = Read only; W1C = Write 1 to clear; -*n* = value after reset

Table 22-35. ADC Group1 Status Register (ADG1SR) Field Descriptions

Bit	Field	Value	Description
31-4	Reserved	0	Reads return 0. Writes have no effect.
3	G1_MEM_EMPTY		Group1 Results Memory Empty. This bit can be effectively used only when the conversion results are read out of the Group1 results memory in the "read from FIFO" mode. Any operation mode read:
		0	The Group1 results memory has valid conversion results.
		1	The Group1 results memory is empty, or does not contain any unread conversion results.
2	G1_BUSY		Group1 Conversion Busy. Any operation mode read:
		0	Group1 conversions are neither in progress nor frozen.
		1	Group1 conversions are either in progress, or are frozen for servicing some other group. This bit will always be set when the Group1 is configured to be in the continuous conversion mode.
1	G1_STOP		Group1 Conversion Stopped. Any operation mode read:
		0	Group1 conversions are not currently frozen.
		1	Group1 conversions are currently frozen.
0	G1_END		Group1 Conversions Ended. Any operation mode read:
		0	Group1 conversions have either not been started or have not yet completed since the last time this status bit was cleared.
		1	The conversion for all the channels selected in the Group1 has completed. This bit can be cleared under the following conditions: • By reading a conversion result from the Group1 results memory in the "read from FIFO" mode. • By writing a new value to the Group1 channel select register (ADG1SEL). • By writing a 1 to this bit. • By disabling the ADC module by clearing the ADC_EN bit in the ADC operating mode control register (ADOPMODECR).

22.3.30 ADC Group2 Status Register (ADG2SR)

ADC Group2 Status Register (ADG2SR) is shown in Figure 22-52 and described in Table 22-36.

Figure 22-52. ADC Group2 Status Register (ADG2SR) [offset = 74h]

31	8
Reserved	
R-0	

7	4	3	2	1	0
Reserved		G2_MEM_ EMPTY	G2_BUSY	G2_STOP	G2_END
R-0		R-1	R-0	R-0	R/W1C-0

LEGEND: R/W = Read/Write; R = Read only; W1C = Write 1 to clear; -*n* = value after reset

Table 22-36. ADC Group2 Status Register (ADG2SR) Field Descriptions

Bit	Field	Value	Description
31-4	Reserved	0	Reads return 0. Writes have no effect.
3	G2_MEM_EMPTY		Group2 Results Memory Empty. This bit can be effectively used only when the conversion results are read out of the Group2 results memory in the "read from FIFO" mode. Any operation mode read:
		0	The Group2 results memory has valid conversion results.
		1	The Group2 results memory is empty, or does not contain any unread conversion results.
2	G2_BUSY		Group2 Conversion Busy. Any operation mode read:
		0	Group2 conversions are neither in progress nor frozen.
		1	Group2 conversions are either in progress, or are frozen for servicing some other group. This bit will always be set when the Group2 is configured to be in the continuous conversion mode.
1	G2_STOP		Group2 Conversion Stopped. Any operation mode read:
		0	Group2 conversions are not currently frozen.
		1	Group2 conversions are currently frozen.
0	G2_END		Group2 Conversions Ended. Any operation mode read:
		0	Group2 conversions have either not been started or have not yet completed since the last time this status bit was cleared.
		1	The conversion for all the channels selected in the Group2 has completed. This bit can be cleared under the following conditions: • By reading a conversion result from the Group2 results memory in the "read from FIFO" mode. • By writing a new value to the Group2 channel select register (ADG2SEL). • By writing a 1 to this bit. • By disabling the ADC module by clearing the ADC_EN bit in the ADC operating mode control register (ADOPMODECR).

22.3.31 ADC Event Group Channel Select Register (ADEVSEL)

ADC Event Group Channel Select Register (ADEVSEL) is shown in Figure 22-53 and described in Table 22-37.

NOTE: Clearing ADEVSEL During a Conversion

Writing 0x0000 to ADEVSEL stops the Event Group conversions. This does not cause the ADC Event Group Results Memory pointer or the Event Group Threshold Register to be reset.

NOTE: Writing A Non-Zero Value To ADEVSEL During a Conversion

Writing a new value to ADEVSEL while a Channel in Event Group is being converted results in a new conversion sequence starting immediately with the highest priority channel in the new ADEVSEL selection. This also causes the ADC Event Group Results Memory pointer to be reset so that the memory allocated for storing the Event Group conversion results gets overwritten. Care should be taken to re-program the corresponding Interrupt Threshold Counter or DMA Threshold Counter again so that correct number of conversions happen before a Threshold interrupt or Block DMA request is generated.

ADC1 supports up to 32 channels and ADC2 supports up to 25 channels on the microcontroller.

Figure 22-53. ADC Event Group Channel Select Register (ADEVSEL) [offset = 78h]

31	0
EV_SEL	
R/W-0	

LEGEND: R/W = Read/Write; -*n* = value after reset

Table 22-37. ADC Event Group Channel Select Register (ADEVSEL) Field Descriptions

Bit	Field	Value	Description
31-0	EV_SEL		Event Group channels selected.
			Any operation mode read/write:
		0	No ADC input channel is selected for conversion in the Event Group.
		Non-zero	The channels marked by the bit positions that are set to 1 will be converted in ascending order when the Event Group is triggered.

22.3.32 ADC Group1 Channel Select Register (ADG1SEL)

ADC Group1 Channel Select Register (ADG1SEL) is shown in Figure 22-54 and described in Table 22-38.

NOTE: Clearing ADG1SEL During a Conversion

Writing 0x0000 to ADG1SEL stops the Group1 conversions. This does not cause the ADC Group1 Results Memory pointer or the Group1 Threshold Register to be reset.

NOTE: Writing A Non-Zero Value To ADG1SEL During a Conversion

Writing a new value to ADG1SEL while a Channel in Group1 is being converted results in a new conversion sequence starting immediately with the highest priority channel in the new ADG1SEL selection. This also causes the ADC Group1 Results Memory pointer to be reset so that the memory allocated for storing the Group1 conversion results gets overwritten. Care should be taken to re-program the corresponding Interrupt Threshold Counter or DMA Threshold Counter again so that correct number of conversions happen before a Threshold interrupt or Block DMA request is generated.

ADC1 supports up to 32 channels and ADC2 supports up to 25 channels on the microcontroller.

Figure 22-54. ADC Group1 Channel Select Register (ADG1SEL) [offset = 7Ch]

31	0
G1_SEL	
R/W-0	

LEGEND: R/W = Read/Write; -*n* = value after reset

Table 22-38. ADC Group1 Channel Select Register (ADG1SEL) Field Descriptions

Bit	Field	Value	Description
31-0	G1_SEL		Group1 channels selected. Any operation mode read/write:
		0	No ADC input channel is selected for conversion in the Group1.
		Non-zero	The channels marked by the bit positions that are set to 1 will be converted in ascending order when the Group1 is triggered.

22.3.33 ADC Group2 Channel Select Register (ADG2SEL)

ADC Group2 Channel Select Register (ADG2SEL) is shown in Figure 22-55 and described in Table 22-39.

NOTE: Clearing ADG2SEL During a Conversion

Writing 0x0000 to ADG2SEL stops the Group2 conversions. This does not cause the ADC Group2 Results Memory pointer or the Group2 Threshold Register to be reset.

NOTE: Writing A Non-Zero Value To ADG2SEL During a Conversion

Writing a new value to ADG2SEL while a Channel in Group2 is being converted results in a new conversion sequence starting immediately with the highest priority channel in the new ADG2SEL selection. This also causes the ADC Group2 Results Memory pointer to be reset so that the memory allocated for storing the Group2 conversion results gets overwritten. Care should be taken to re-program the corresponding Interrupt Threshold Counter or DMA Threshold Counter again so that correct number of conversions happen before a Threshold interrupt or Block DMA request is generated.

ADC1 supports up to 32 channels and ADC2 supports up to 25 channels on the microcontroller.

Figure 22-55. ADC Group2 Channel Select Register (ADG2SEL) [offset = 80h]

31	0
G2_SEL	
R/W-0	

LEGEND: R/W = Read/Write; -*n* = value after reset

Table 22-39. ADC Group2 Channel Select Register (ADG2SEL) Field Descriptions

Bit	Field	Value	Description
31-0	G2_SEL		Group2 channels selected.
			Any operation mode read/write:
		0	No ADC input channel is selected for conversion in the Group2.
		Non-zero	The channels marked by the bit positions that are set to 1 will be converted in ascending order when the Group2 is triggered.

22.3.34 ADC Calibration and Error Offset Correction Register (ADCALR)

ADC Calibration and Error Offset Correction Register (ADCALR) is shown in Figure 22-56 and Figure 22-57, and described in Table 22-40. As shown, the format of the ADCALR is different based on whether the ADC module is configured to be a 12-bit or a 10-bit ADC module.

Figure 22-56. 12-bit ADC Calibration and Error Offset Correction Register (ADCALR) [offset = 84h]

31 12	11 0
Reserved	ADCALR
R-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Figure 22-57. 10-bit ADC Calibration and Error Offset Correction Register (ADCALR) [offset = 84h]

31 10	9 0
Reserved	ADCALR
R-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 22-40. ADC Calibration and Error Offset Correction Register (ADCALR) Field Descriptions

Field	Value	Description
Reserved	0	Reads return 0. Writes have no effect.
ADCALR		ADC Calibration Result and Offset Error Correction Value. The ADC module writes the results of the calibration conversions to this register. The application is required to use these conversion results and determine the ADC offset error. The application can then compute the correction for the offset error and this correction value needs to be written back to the ADCALR register in the 2's complement form. During normal conversion (when calibration is disabled), the ADCALR register contents are automatically added to each digital output from the ADC core before it is stored in the ADC results memory. For more details on error calibration, refer to Section 22.2.6.1.

22.3.35 ADC State Machine Status Register (ADSMSTATE)

Figure 22-58 and Table 22-41 describe the ADSMSTATE register.

Figure 22-58. ADC State Machine Status Register (ADSMSTATE) [offset = 88h]

31 4	3 0
Reserved	SMSTATE
R-0	R-0

LEGEND: R = Read only; -*n* = value after reset

Table 22-41. ADC State Machine Status Register (ADSMSTATE) Field Descriptions

Bit	Field	Value	Description
31-4	Reserved	0	Reads return 0. Writes have no effect.
3-0	SMSTATE		ADC State Machine Current State. These bits reflect the current state of the state machine and are reserved for use by TI for debug purposes.

22.3.36 ADC Channel Last Conversion Value Register (ADLASTCONV)

ADC Channel Last Conversion Value Register (ADLASTCONV) is shown in Figure 22-59 and described in Table 22-42.

Figure 22-59. ADC Channel Last Conversion Value Register (ADLASTCONV) [offset = 8Ch]

31 - 24	23 - 0
Reserved	LAST_CONV
R-0	R-U

LEGEND: R = Read only; -*n* = value after reset; U = value after reset is unknown

Table 22-42. ADC Channel Last Conversion Value Register (ADLASTCONV) Field Descriptions

Bit	Field	Value	Description
31-24	Reserved	0	Reads return 0. Writes have no effect.
23-0	LAST_CONV		ADC Input Channel's Last Converted Value. This register indicates whether the last converted value for a particular input channel was lower or higher than the mid-point of the reference voltage. In other words, this register acts as a digital input register and can be read by the application to determine the digital level at the input pins. This data is only valid for an input channel if it has been converted at least once. Any operation mode read for each bit of this register:
		0	A level lower than the midpoint reference voltage was measured at the last conversion for this channel.
		1	A level higher than or equal to the midpoint reference voltage was measured at the last conversion for this channel.

22.3.37 ADC Event Group Results' FIFO Register (ADEVBUFFER)

ADC Event Group Results' FIFO Register (ADEVBUFFER) is shown in Figure 22-60 and Figure 22-61, and described in Table 22-43. As shown, the format of the data read from the ADEVBUFFER locations is different based on whether the ADC module is configured to be a 12-bit or a 10-bit ADC module.

Figure 22-60. 12-bit ADC Event Group Results' FIFO Register (ADEVBUFFER) [offset = 90h-AFh]

31	30–21	20–16
EV_EMPTY	Reserved	EV_CHID
R-1	R-0	R-0

15–12	11–0
Reserved	EV_DR
R-0	R-U

LEGEND: R = Read only; -*n* = value after reset; U = value after reset is unknown

Figure 22-61. 10-bit ADC Event Group Results' FIFO Register (ADEVBUFFER) [offset = 90h-AFh]

31–16
Reserved
R-0

15	14–10	9–0
EV_EMPTY	EV_CHID	EV_DR
R-1	R-0	R-U

LEGEND: R = Read only; -*n* = value after reset; U = value after reset is unknown

Table 22-43. ADC Event Group Results' FIFO Register (ADEVBUFFER) Field Descriptions

Field	Value	Description
Reserved	0	Reads return 0. Writes have no effect.
EV_EMPTY		Event Group FIFO Empty. This bit is applicable only when the "read from FIFO" mode is used for reading the Event Group conversion results. Any operation mode read:
	0	The data in the EV_DR field of this buffer is valid.
	1	The data in the EV_DR field of this buffer is not valid and there are no valid data in the Event Group results memory.
EV_CHID		Event Group Channel Id. These bits are also applicable only when the "read from FIFO" mode is used for reading the Event Group conversion results. Any operation mode read:
	0	The conversion result in the EV_DR field of this buffer is from the ADC input channel 0, or the channel id mode is disabled in the Event Group mode control register (ADEVMODECR).
	1h-1Fh	The conversion result in the EV_DR field of this buffer is from the ADC input channel number denoted by the EV_CHID field.
EV_DR		Event Group Digital Conversion Result. The Event Group results’ FIFO location is aliased eight times, so that any word-aligned read from the address range 90h to AFh results in one conversion result to be read from the Event Group results’ memory. This allows the ARM LDMIA instruction to read out up to 8 conversion results from the Event Group results’ memory with just one instruction.

22.3.38 ADC Group1 Results FIFO Register (ADG1BUFFER)

ADC Group1 Results FIFO Register (ADG1BUFFER) is shown in Figure 22-62 and Figure 22-63, described in Table 22-44. As shown, the format of the data read from the ADG1BUFFER locations is different based on whether the ADC module is configured to be a 12-bit or a 10-bit ADC module.

Figure 22-62. 12-bit ADC Group1 Results FIFO Register (ADG1BUFFER) [offset = B0h-CFh]

31	30	21	20	16
G1_EMPTY	Reserved		G1_CHID	
R-1	R-0		R-0	

15	12	11	0
Reserved		G1_DR	
R-0		R-U	

LEGEND: R = Read only; -*n* = value after reset; U = value after reset is unknown

Figure 22-63. 10-bit ADC Group1 Results' FIFO Register (ADG1BUFFER) [offset = B0h-CFh]

31	16
Reserved	
R-0	

15	14	10	9	0
G1_EMPTY	G1_CHID		G1_DR	
R-1	R-0		R-U	

LEGEND: R = Read only; -*n* = value after reset; U = value after reset is unknown

Table 22-44. ADC Group1 Results FIFO Register (ADG1BUFFER) Field Descriptions

Field	Value	Description
Reserved	0	Reads return 0. Writes have no effect.
G1_EMPTY		Group1 FIFO Empty. This bit is applicable only when the "read from FIFO" mode is used for reading the Group1 conversion results.
		Any operation mode read:
	0	The data in the G1_DR field of this buffer is valid.
	1	The data in the G1_DR field of this buffer is not valid and there are no valid data in the Group1 results memory.
G1_CHID		Group1 Channel Id. These bits are also applicable only when the "read from FIFO" mode is used for reading the Group1 conversion results.
		Any operation mode read:
	0	The conversion result in the G1_DR field of this buffer is from the ADC input channel 0, or the channel id mode is disabled in the Group1 mode control register (ADG1MODECR).
	1h-1Fh	The conversion result in the G1_DR field of this buffer is from the ADC input channel number denoted by the G1_CHID field.
G1_DR		Group1 Digital Conversion Result. The Group1 results' FIFO location is aliased eight times, so that any word-aligned read from the address range B0h to CFh results in one conversion result to be read from the Group1 results' memory. This allows the ARM LDMIA instruction to read out up to 8 conversion results from the Group1 results' memory with just one instruction.

22.3.39 ADC Group2 Results FIFO Register (ADG2BUFFER)

ADC Group2 Results FIFO Register (ADG2BUFFER) is shown in Figure 22-64 and Figure 22-65, described in Table 22-45. As shown, the format of the data read from the ADG2BUFFER locations is different based on whether the ADC module is configured to be a 12-bit or a 10-bit ADC module.

Figure 22-64. 12-bit ADC Group2 Results FIFO Register (ADG2BUFFER) [offset = D0h-EFh]

31	30-21	20-16
G2_EMPTY	Reserved	G2_CHID
R-1	R-0	R-0

15-12	11-0
Reserved	G2_DR
R-0	R-U

LEGEND: R = Read only; -*n* = value after reset; U = value after reset is unknown

Figure 22-65. 10-bit ADC Group2 Results' FIFO Register (ADG2BUFFER) [offset = D0h-EFh]

31-16
Reserved
R-0

15	14-10	9-0
G2_EMPTY	G2_CHID	G2_DR
R-1	R-0	R-U

LEGEND: R = Read only; -*n* = value after reset; U = value after reset is unknown

Table 22-45. ADC Group2 Results FIFO Register (ADG2BUFFER) Field Descriptions

Field	Value	Description
Reserved	0	Reads return 0. Writes have no effect.
G2_EMPTY		Group2 FIFO Empty. This bit is applicable only when the "read from FIFO" mode is used for reading the Group2 conversion results.
		Any operation mode read:
	0	The data in the G2_DR field of this buffer is valid.
	1	The data in the G2_DR field of this buffer is not valid and there are no valid data in the Group2 results memory.
G2_CHID		Group2 Channel Id. These bits are also applicable only when the "read from FIFO" mode is used for reading the Group2 conversion results.
		Any operation mode read:
	0	The conversion result in the G2_DR field of this buffer is from the ADC input channel 0, or the channel id mode is disabled in the Group2 mode control register (ADG2MODECR).
	1h-1Fh	The conversion result in the G2_DR field of this buffer is from the ADC input channel number denoted by the G2_CHID field.
G2_DR		Group2 Digital Conversion Result. The Group2 results' FIFO location is aliased eight times, so that any word-aligned read from the address range D0h to EFh results in one conversion result to be read from the Group2 results' memory. This allows the ARM LDMIA instruction to read out up to 8 conversion results from the Group2 results' memory with just one instruction.

22.3.40 ADC Event Group Results Emulation FIFO Register (ADEVEMUBUFFER)

ADC Event Group Results Emulation FIFO Register (ADEVEMUBUFFER) is shown in Figure 22-66 and Figure 22-67, and described in Table 22-46. As shown, the format of the data read from the ADEVEMUBUFFER locations is different based on whether the ADC module is configured to be a 12-bit or a 10-bit ADC module.

A read from this location also gives out one conversion result from the Event Group results' memory along with the EV_EMPTY status bit and the optional channel id. However, this read will not affect any of the status flags in the Event Group interrupt flag register or the Event Group status register. This register is useful for debuggers.

Figure 22-66. 12-bit ADC Event Group Results Emulation FIFO Register (ADEVEMUBUFFER) [offset = F0h]

31	30 21	20 16
EV_EMPTY	Reserved	EV_CHID
R-1	R-0	R-0

15 12	11 0
Reserved	EV_DR
R-0	R-U

LEGEND: R = Read only; -*n* = value after reset; U = value after reset is unknown

Figure 22-67. 10-bit ADC Event Group Results Emulation FIFO Register (ADEVEMUBUFFER) [offset = F0h]

31 16
Reserved
R-0

15	14 10	9 0
EV_EMPTY	EV_CHID	EV_DR
R-1	R-0	R-U

LEGEND: R = Read only; -*n* = value after reset; U = value after reset is unknown

Table 22-46. ADC Event Group Results Emulation FIFO Register (ADEVEMUBUFFER) Field Descriptions

Field	Value	Description
Reserved	0	Reads return 0. Writes have no effect.
EV_EMPTY		Event Group FIFO Empty. This bit is applicable only when the "read from FIFO" mode is used for reading the Event Group conversion results.
		Any operation mode read:
	0	The data in the EV_DR field of this buffer is valid.
	1	The data in the EV_DR field of this buffer is not valid and there are no valid data in the Event Group results memory.
EV_CHID		Event Group Channel Id. These bits are also applicable only when the "read from FIFO" mode is used for reading the Event Group conversion results.
		Any operation mode read:
	0	The conversion result in the EV_DR field of this buffer is from the ADC input channel 0, or the channel id mode is disabled in the Event Group operating mode control register (ADEVMODECR).
	1h-1Fh	The conversion result in the EV_DR field of this buffer is from the ADC input channel number denoted by the EV_CHID field.
EV_DR		Event Group Digital Conversion Result. These bits contain the digital result output from the Event Group FIFO buffer. The result can be presented in an 8-bit, 10-bit, or 12-bit format for a 12-bit ADC module, or in an 8-bit or 10-bit format for a 10-bit ADC module. The conversion result data is automatically shifted right by the appropriate number of bits when using a reduced-size data format with the upper bits reading as zeros.

22.3.41 ADC Group1 Results Emulation FIFO Register (ADG1EMUBUFFER)

ADC Group1 Results Emulation FIFO Register (ADG1EMUBUFFER) is shown in Figure 22-68 and Figure 22-69, described in Table 22-47. As shown, the format of the data read from the ADG1EMUBUFFER locations is different based on whether the ADC module is configured to be a 12-bit or a 10-bit ADC module.

A read from this location also gives out one conversion result from the Group1 results' memory along with the G1_EMPTY status bit and the optional channel id. However, this read will not affect any of the status flags in the Group1 interrupt flag register or the Group1 status register. This register is useful for debuggers.

Figure 22-68. 12-bit ADC Group1 Results Emulation FIFO Register (ADG1EMUBUFFER) [offset = F4h]

31	30 - 21	20 - 16
G1_EMPTY	Reserved	G1_CHID
R-1	R-0	R-0

15 - 12	11 - 0
Reserved	G1_DR
R-0	R-U

LEGEND: R = Read only; -*n* = value after reset; U = value after reset is unknown

Figure 22-69. 10-bit ADC Group1 Results Emulation FIFO Register (ADG1EMUBUFFER) [offset = F4h]

31 - 16
Reserved
R-0

15	14 - 10	9 - 0
G1_EMPTY	G1_CHID	G1_DR
R-1	R-0	R-U

LEGEND: R = Read only; -*n* = value after reset; U = value after reset is unknown

Table 22-47. ADC Group1 Results Emulation FIFO Register (ADG1EMUBUFFER) Field Descriptions

Field	Value	Description
Reserved	0	Reads return 0. Writes have no effect.
G1_EMPTY		Group1 FIFO Empty. This bit is applicable only when the "read from FIFO" mode is used for reading the Group1 conversion results.
		Any operation mode read:
	0	The data in the G1_DR field of this buffer is valid.
	1	The data in the G1_DR field of this buffer is not valid and there are no valid data in the Group1 results memory.
G1_CHID		Group1 Channel Id. These bits are also applicable only when the "read from FIFO" mode is used for reading the Group1 conversion results.
		Any operation mode read:
	0	The conversion result in the G1_DR field of this buffer is from the ADC input channel 0, or the channel id mode is disabled in the Group1 operating mode control register (ADG1MODECR).
	1h-1Fh	The conversion result in the G1_DR field of this buffer is from the ADC input channel number denoted by the G1_CHID field.
G1_DR		Group1 Digital Conversion Result. These bits contain the digital result output from the Group 1 FIFO buffer. The result can be presented in an 8-bit, 10-bit, or 12-bit format for a 12-bit ADC module, or in an 8-bit or 10-bit format for a 10-bit ADC module. The conversion result data is automatically shifted right by the appropriate number of bits when using a reduced-size data format with the upper bits reading as zeros.

22.3.42 ADC Group2 Results Emulation FIFO Register (ADG2EMUBUFFER)

ADC Group2 Results Emulation FIFO Register (ADG2EMUBUFFER) is shown in Figure 22-70 and Figure 22-71, described in Table 22-48. As shown, the format of the data read from the ADG2EMUBUFFER locations is different based on whether the ADC module is configured to be a 12-bit or a 10-bit ADC module.

A read from this location also gives out one conversion result from the Group2 results' memory along with the G2_EMPTY status bit and the optional channel id. However, this read will not affect any of the status flags in the Group2 interrupt flag register or the Group2 status register. This register is useful for debuggers.

Figure 22-70. 12-bit ADC Group2 Results Emulation FIFO Register (ADG2EMUBUFFER) [offset = F8h]

31	30 21	20 16
G2_EMPTY	Reserved	G2_CHID
R-1	R-0	R-0

15 12	11 0
Reserved	G2_DR
R-0	R-U

LEGEND: R = Read only; -*n* = value after reset; U = value after reset is unknown

Figure 22-71. 10-bit ADC Group2 Results Emulation FIFO Register (ADG2EMUBUFFER) [offset = F8h]

31 16
Reserved
R-0

15	14 10	9 0
G2_EMPTY	G2_CHID	G2_DR
R-1	R-0	R-U

LEGEND: R = Read only; -*n* = value after reset; U = value after reset is unknown

Table 22-48. ADC Group2 Results Emulation FIFO Register (ADG2EMUBUFFER) Field Descriptions

Field	Value	Description
Reserved	0	Reads return 0. Writes have no effect.
G2_EMPTY		Group2 FIFO Empty. This bit is applicable only when the "read from FIFO" mode is used for reading the Group2 conversion results.
		Any operation mode read:
	0	The data in the G2_DR field of this buffer is valid.
	1	The data in the G2_DR field of this buffer is not valid and there are no valid data in the Group2 results memory.
G2_CHID		Group2 Channel Id. These bits are also applicable only when the "read from FIFO" mode is used for reading the Group2 conversion results.
		Any operation mode read:
	0	The conversion result in the G2_DR field of this buffer is from the ADC input channel 0, or the channel id mode is disabled in the Group2 operating mode control register (ADG2MODECR).
	1h-1Fh	The conversion result in the G2_DR field of this buffer is from the ADC input channel number denoted by the G2_CHID field.
G2_DR		Group2 Digital Conversion Result. These bits contain the digital result output from the Group 2 FIFO buffer. The result can be presented in an 8-bit, 10-bit, or 12-bit format for a 12-bit ADC module, or in an 8-bit or 10-bit format for a 10-bit ADC module. The conversion result data is automatically shifted right by the appropriate number of bits when using a reduced-size data format with the upper bits reading as zeros.

22.3.43 ADC ADEVT Pin Direction Control Register (ADEVTDIR)

ADC ADEVT Pin Direction Control Register (ADEVTDIR) is shown in Figure 22-72 and described in Table 22-49.

Figure 22-72. ADC ADEVT Pin Direction Control Register (ADEVTDIR) [offset = FCh]

31 … 1	0
Reserved	ADEVT_DIR
R-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 22-49. ADC ADEVT Pin Direction Control Register (ADEVTDIR) Field Descriptions

Bit	Field	Value	Description
31-1	Reserved	0	Reads return 0. Writes have no effect.
0	ADEVT_DIR		ADEVT Pin Direction. Any operating mode read/write:
		0	ADEVT is an input pin; the output buffer is disabled.
		1	ADEVT is an output pin; the output buffer is enabled.

22.3.44 ADC ADEVT Pin Output Value Control Register (ADEVTOUT)

ADC ADEVT Pin Output Value Control Register (ADEVTOUT) is shown in Figure 22-73 and described in Table 22-50.

Figure 22-73. ADC ADEVT Pin Output Value Control Register (ADEVTOUT) [offset = 100h]

31 1	0
Reserved	ADEVT_OUT
R-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 22-50. ADC ADEVT Pin Output Value Control Register (ADEVTOUT) Field Descriptions

Bit	Field	Value	Description
31-1	Reserved	0	Reads return 0. Writes have no effect.
0	ADEVT_OUT		ADEVT Pin Output Value. This bit determines the logic level to be output to the ADEVT pin when the pin is configured to be an output pin.
			Any operating mode read/write:
		0	Output logic LOW on the ADEVT pin.
		1	Output logic HIGH on the ADEVT pin.

22.3.45 ADC ADEVT Pin Input Value Register (ADEVTIN)

ADC ADEVT Pin Input Value Register (ADEVTIN) is shown in Figure 22-74 and described in Table 22-51.

Figure 22-74. ADC ADEVT Pin Input Value Register (ADEVTIN) [offset = 104h]

31 1	0
Reserved	ADEVT_IN
R-0	R-U

LEGEND: R = Read only; -*n* = value after reset; U = value after reset is unknown

Table 22-51. ADC ADEVT Pin Input Value Register (ADEVTIN) Field Descriptions

Bit	Field	Value	Description
31-1	Reserved	0	Reads return 0. Writes have no effect.
0	ADEVT_IN		ADEVT Pin Input Value. This is a read-only bit that reflects the logic level on the ADEVT pin.
			Any operating mode read:
		0	Logic LOW present on the ADEVT pin.
		1	Logic HIGH present on the ADEVT pin.

22.3.46 ADC ADEVT Pin Set Register (ADEVTSET)

ADC ADEVT Pin Set Register (ADEVTSET) is shown in Figure 22-75 and described in Table 22-52.

Figure 22-75. ADC ADEVT Pin Set Register (ADEVTSET) [offset = 108h]

31 1	0
Reserved	ADEVT_SET
R-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 22-52. ADC ADEVT Pin Set Register (ADEVTSET) Field Descriptions

Bit	Field	Value	Description
31-1	Reserved	0	Reads return 0. Writes have no effect.
0	ADEVT_SET		ADEVT Pin Set. This bit drives the output of the ADEVT pin high. A read from this bit always returns the current state of the ADEVT pin.
			Any operating mode read/write:
		0	Output value on the ADEVT pin is unchanged.
		1	Output logic HIGH on the ADEVT pin, if the pin is configured to be an output pin.

22.3.47 ADC ADEVT Pin Clear Register (ADEVTCLR)

ADC ADEVT Pin Clear Register (ADEVTCLR) is shown in Figure 22-76 and described in Table 22-53.

Figure 22-76. ADC ADEVT Pin Clear Register (ADEVTCLR) [offset = 10Ch]

31 1	0
Reserved	ADEVT_CLR
R-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 22-53. ADC ADEVT Pin Clear Register (ADEVTCLR) Field Descriptions

Bit	Field	Value	Description
31-1	Reserved	0	Reads return 0. Writes have no effect.
0	ADEVT_CLR		ADEVT Pin Clear. A read from this bit always returns the current state of the ADEVT pin.
			Any operating mode read/write:
		0	Output value on the ADEVT pin is unchanged.
		1	Output logic LOW on the ADEVT pin, if the pin is configured to be an output pin.

22.3.48 ADC ADEVT Pin Open Drain Enable Register (ADEVTPDR)

ADC ADEVT Pin Open Drain Enable Register (ADEVTPDR) is shown in Figure 22-77 and described in Table 22-54.

Figure 22-77. ADC ADEVT Pin Open Drain Enable Register (ADEVTPDR) [offset = 110h]

31	1	0
Reserved		ADEVT_PDR
R-0		R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 22-54. ADC ADEVT Pin Open Drain Enable Register (ADEVTPDR) Field Descriptions

Bit	Field	Value	Description
31-1	Reserved	0	Reads return 0. Writes have no effect.
0	ADEVT_PDR		ADEVT Pin Open Drain Enable. This bit enables the open-drain capability for the ADEVT pin if it is configured to be an output and a logic HIGH is being driven on to the pin.
			Any operating mode read/write:
		0	Output value on the ADEVT pin is logic HIGH.
		1	The ADEVT pin is tristated.

22.3.49 ADC ADEVT Pin Pull Control Disable Register (ADEVTPDIS)

ADC ADEVT Pin Pull Control Disable Register (ADEVTPDIS) is shown in Figure 22-78 and described in Table 22-55.

Figure 22-78. ADC ADEVT Pin Pull Control Disable Register (ADEVTPDIS) [offset = 114h]

31	1	0
Reserved		ADEVT_PDIS
R-0		R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 22-55. ADC ADEVT Pin Pull Control Disable Register (ADEVTPDIS) Field Descriptions

Bit	Field	Value	Description
31-1	Reserved	0	Reads return 0. Writes have no effect.
0	ADEVT_PDIS		ADEVT Pin Pull Control Disable. This bit enables or disables the pull control on the ADEVT pin if it is configured to be an input pin.
			Any operating mode read/write:
		0	Pull on ADEVT pin is enabled.
		1	Pull on ADEVT pin is disabled.

22.3.50 ADC ADEVT Pin Pull Control Select Register (ADEVTPSEL)

ADC ADEVT Pin Pull Control Select Register (ADEVTPSEL) is shown in Figure 22-79 and described in Table 22-56.

Figure 22-79. ADC ADEVT Pin Pull Control Select Register (ADEVTPSEL) [offset = 118h]

31 … 1	0
Reserved	ADEVT_PSEL
R-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 22-56. ADC ADEVT Pin Pull Control Select Register (ADEVTPSEL) Field Descriptions

Bit	Field	Value	Description
31-1	Reserved	0	Reads return 0. Writes have no effect.
0	ADEVT_PSEL		ADEVT Pin Pull Control Select. This bit selects a pull-down or pull-up on the ADEVT pin if it is configured to be an input pin. Any operating mode read/write:
		0	Pull-down is selected on ADEVT pin.
		1	Pull-up is selected on ADEVT pin.

22.3.51 ADC Event Group Sample Cap Discharge Control Register (ADEVSAMPDISEN)

ADC Event Group Sample Cap Discharge Control Register (ADEVSAMPDISEN) is shown in Figure 22-80 and described in Table 22-57.

Figure 22-80. ADC Event Group Sample Cap Discharge Control Register (ADEVSAMPDISEN) [offset = 11Ch]

31 … 16		
Reserved		
R-0		
15 … 8	7 … 1	0
EV_SAMP_DIS_CYC	Reserved	EV_SAMP_ DIS_EN
R/W-0	R-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 22-57. ADC Event Group Sample Cap Discharge Control Register (ADEVSAMPDISEN) Field Descriptions

Bit	Field	Value	Description
31-16	Reserved	0	Reads return 0. Writes have no effect.
15-8	EV_SAMP_DIS_CYC		Event Group sample cap discharge cycles. These bits specify the duration in terms of ADCLK cycles for which the ADC internal sampling capacitor is allowed to discharge before sampling the input channel voltage.
7-1	Reserved	0	Reads return 0. Writes have no effect.
0	EV_SAMP_DIS_EN		Event Group sample cap discharge enable. Any operation mode read/write:
		0	Event Group sample cap discharge mode is disabled.
		1	Event Group sample cap discharge mode is enabled. The ADC internal sampling capacitor is connected to the V_{REFLO} reference voltage for a duration specified by the EV_SAMP_DIS_CYC field. After this discharge time has expired the selected ADC input channel is sampled and converted normally based on the Event Group settings.

22.3.52 ADC Group1 Sample Cap Discharge Control Register (ADG1SAMPDISEN)

ADC Group1 Sample Cap Discharge Control Register (ADG1SAMPDISEN) is shown in Figure 22-81 and described in Table 22-58.

Figure 22-81. ADC Group1 Sample Cap Discharge Control Register (ADG1SAMPDISEN) [offset = 120h]

31	16
Reserved	
R-0	

15	8	7	1	0
G1_SAMP_DIS_CYC		Reserved		G1_SAMP_ DIS_EN
R/W-0		R-0		R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 22-58. ADC Group1 Sample Cap Discharge Control Register (ADG1SAMPDISEN) Field Descriptions

Bit	Field	Value	Description
31-16	Reserved	0	Reads return 0. Writes have no effect.
15-8	G1_SAMP_DIS_CYC		Group1 sample cap discharge cycles. These bits specify the duration in terms of ADCLK cycles for which the ADC internal sampling capacitor is allowed to discharge before sampling the input channel voltage.
7-1	Reserved	0	Reads return 0. Writes have no effect.
0	G1_SAMP_DIS_EN		Group1 sample cap discharge enable.
			Any operation mode read/write:
		0	Group1 sample cap discharge mode is disabled.
		1	Group1 sample cap discharge mode is enabled. The ADC internal sampling capacitor is connected to the V_{REFLO} reference voltage for a duration specified by the G1_SAMP_DIS_CYC field. After this discharge time has expired the selected ADC input channel is sampled and converted normally based on the Group1 settings.

22.3.53 ADC Group2 Sample Cap Discharge Control Register (ADG2SAMPDISEN)

ADC Group2 Sample Cap Discharge Control Register (ADG2SAMPDISEN) is shown in Figure 22-82 and described in Table 22-59.

Figure 22-82. ADC Group2 Sample Cap Discharge Control Register (ADG2SAMPDISEN) [offset = 124h]

31	16
Reserved	
R-0	

15 8	7 1	0
G2_SAMP_DIS_CYC	Reserved	G2_SAMP_ DIS_EN
R/W-0	R-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 22-59. ADC Group2 Sample Cap Discharge Control Register (ADG2SAMPDISEN) Field Descriptions

Bit	Field	Value	Description
31-16	Reserved	0	Reads return 0. Writes have no effect.
15-8	G2_SAMP_DIS_CYC		Group2 sample cap discharge cycles. These bits specify the duration in terms of ADCLK cycles for which the ADC internal sampling capacitor is allowed to discharge before sampling the input channel voltage.
7-1	Reserved	0	Reads return 0. Writes have no effect.
0	G2_SAMP_DIS_EN		Group2 sample cap discharge enable.
			Any operation mode read/write:
		0	Group2 sample cap discharge mode is disabled.
		1	Group2 sample cap discharge mode is enabled. The ADC internal sampling capacitor is connected to the V_{REFLO} reference voltage for a duration specified by the G2_SAMP_DIS_CYC field. After this discharge time has expired the selected ADC input channel is sampled and converted normally based on the Group2 settings.

22.3.54 ADC Magnitude Compare Interrupt Control Registers (ADMAGINTxCR)

ADC Magnitude Compare Interrupt Control Registers (ADMAGINTxCR) are shown in Figure 22-83 and Figure 22-84, and described in Table 22-60. As shown, the format of the ADMAGINTxCR is different based on whether the ADC module is configured to be a 12-bit or a 10-bit ADC module. The ADC module supports up to three magnitude compare interrupts. These registers are at offset addresses 128h, 130h, and 138h.

Figure 22-83. 12-bit ADC Magnitude Compare Interrupt Control Registers (ADMAGINTxCR) [offset = 128h-138h]

31 28	27 16
Reserved	MAG_THRx
R-0	R/W-0

15	14	13	12 8
CHN_THR_ COMPx	CMP_GE_LTx	Reserved	COMP_CHIDx
R/W-0	R/W-0	R-0	R/W-0

7 5	4 0
Reserved	MAG_CHIDx
R-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Figure 22-84. 10-bit ADC Magnitude Compare Interrupt Control Registers (ADMAGINTxCR) [offset = 128h-138h]

31	30 26	25 16
Rsvd	MAG_CHIDx	MAG_THRx
R-0	R/W-0	R/W-0

15 13	12 8
Reserved	COMP_CHIDx
R-0	R/W-0

7 2	1	0
Reserved	CHN_THR_ COMPx	CMP_GE_LTx
R-0	R/W-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 22-60. ADC Magnitude Compare Interrupt Control Registers (ADMAGINTxCR) Field Descriptions

Field	Value	Description
Reserved	0	Reads return 0. Writes have no effect.
MAG_CHIDx		These bits specify the channel number from 0 to 31 for which the conversion result needs to be monitored by the ADC.
MAG_THRx		These bits specify the 12-bit or 10-bit compare value that the ADC will use for the comparison with the MAG_CHIDx channel's conversion result.
COMP_CHIDx		These bits specify the channel number from 0 to 31 whose last conversion result is compared with the MAG_CHIDx channel's conversion result.
CHN_THR_COMPx		Channel OR Threshold comparison.
		Any operation mode read/write:
	0	The ADC module will compare the MAG_CHIDx channel's conversion result with the fixed threshold value specified by the MAG_THRx field
	1	The ADC module will compare the MAG_CHIDx channel's conversion result with the last conversion result for the COMP_CHIDx channel.
		Both the MAG_CHIDx and the COMP_CHIDx channel must have been converted at least once for the ADC to perform the comparison.
CMP_GE_LTx		"Greater than or equal to" OR "Less than" comparison operator.
		Any operation mode read/write:
	0	The ADC module will check if the conversion result is lower than the reference value (fixed threshold or COMP_CHIDx conversion result).
	1	The ADC module will check if the conversion result is greater than or equal to the reference value (fixed threshold or COMP_CHIDx conversion result).

22.3.55 ADC Magnitude Compare Interruptx Mask Register (ADMAGINTxMASK)

ADC Magnitude Compare Interruptx Mask Register (ADMAGINTxMASK) is shown in Figure 22-85and Figure 22-86, and described in Table 22-61. As shown, the format of the ADMAGINTxMASK is different based on whether the ADC module is configured to be a 12-bit or a 10-bit ADC module. There are three mask registers for the three magnitude compare interrupts. These registers are at offset addresses 12Ch, 134h, and 13Ch.

Figure 22-85. 12-bit ADC Magnitude Compare Mask Register (ADMAGINTxMASK) [offset = 12Ch-13Ch]

31 12	11 0
Reserved	MAG_INTx_MASK
R-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Figure 22-86. 10-bit ADC Magnitude Compare Mask Register (ADMAGINTxMASK) [offset = 12Ch-13Ch]

31 10	9 0
Reserved	MAG_INTx_MASK
R-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 22-61. ADC Magnitude Compare Interruptx Mask Register (ADMAGINTxMASK) Field Descriptions

Field	Value	Description
Reserved	0	Reads return 0. Writes have no effect.
MAG_INTx_MASK		These bits specify the mask for the comparison in order to generate the magnitude compare interrupt # x.
		Any operation mode read/write:
	0	The ADC module will not mask the corresponding bit for the comparison.
	1	The ADC module will mask the corresponding bit for the comparison.

22.3.56 ADC Magnitude Compare Interrupt Enable Set Register (ADMAGINTENASET)

ADC Magnitude Compare Interrupt Enable Set Register (ADMAGINTENASET) is shown in Figure 22-87 and described in Table 22-62.

Figure 22-87. ADC Magnitude Compare Interrupt Enable Set Register (ADMAGINTENASET) [offset = 158h]

31	3	2	0
Reserved		MAG_INT_ENA_SET	
R-0		R/W-0	

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 22-62. ADC Magnitude Compare Interrupt Enable Set Register (ADMAGINTENASET) Field Descriptions

Bit	Field	Value	Description
31-3	Reserved	0	Reads return 0. Writes have no effect.
2-0	MAG_INT_ENA_SET		Each of these three bits, when set, enable the corresponding magnitude compare interrupt.
			Any operation mode read/write for each bit:
		0	The enable status of the corresponding magnitude compare interrupt is left unchanged.
		1	The corresponding magnitude compare interrupt is enabled.

22.3.57 ADC Magnitude Compare Interrupt Enable Clear Register (ADMAGINTENACLR)

ADC Magnitude Compare Interrupt Enable Clear Register (ADMAGINTENACLR) is shown in Figure 22-88 and described in Table 22-63.

Figure 22-88. ADC Magnitude Compare Interrupt Enable Clear Register (ADMAGINTENACLR) [offset = 15Ch]

31	3	2	0
Reserved		MAG_INT_ENA_CLR	
R-0		R/W-0	

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 22-63. ADC Magnitude Compare Interrupt Enable Clear Register (ADMAGINTENACLR) Field Descriptions

Bit	Field	Value	Description
31-3	Reserved	0	Reads return 0. Writes have no effect.
2-0	MAG_INT_ENA_CLR		Each of these three bits, when set, enable the corresponding magnitude compare interrupt.
			Any operation mode read/write for each bit:
		0	The enable status of the corresponding magnitude compare interrupt is left unchanged.
		1	The corresponding magnitude compare interrupt is disabled.

22.3.58 ADC Magnitude Compare Interrupt Flag Register (ADMAGINTFLG)

ADC Magnitude Compare Interrupt Flag Register (ADMAGINTFLG) is shown in Figure 22-89 and described in Table 22-64.

Figure 22-89. ADC Magnitude Compare Interrupt Flag Register (ADMAGINTFLG) [offset = 160h]

31 3	2 0
Reserved	MAG_INT_FLG
R-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 22-64. ADC Magnitude Compare Interrupt Flag Register (ADMAGINTFLG) Field Descriptions

Bit	Field	Value	Description
31-3	Reserved	0	Reads return 0. Writes have no effect.
2-0	MAG_INT_FLG		Magnitude Compare Interrupt Flags. These bits can be polled by the application to determine if the magnitude compares have been evaluated as true. When a magnitude compare interrupt flag is set, the corresponding magnitude compare interrupt will be generated if enabled. Any operation mode, for each bit:
		0	Read: The condition for the corresponding magnitude threshold interrupt was false. Write: The corresponding flag is left unchanged.
		1	Read: The condition for the corresponding magnitude threshold interrupt was true. Write: The corresponding flag is cleared. The flag can also be cleared by reading from the magnitude compare interrupt offset register.

22.3.59 ADC Magnitude Compare Interrupt Offset Register (ADMAGINTOFF)

ADC Magnitude Compare Interrupt Offset Register (ADMAGINTOFF) is shown in Figure 22-90 and described in Table 22-65.

Figure 22-90. ADC Magnitude Compare Interrupt Offset Register (ADMAGINTOFF) [offset = 164h]

31 4	3 0
Reserved	MAG_INT_OFF
R-0	RC-0

LEGEND: R = Read only; RC = Clear field after read; -*n* = value after reset

Table 22-65. ADC Magnitude Compare Interrupt Offset Register (ADMAGINTOFF) Field Descriptions

Bit	Field	Value	Description
31-4	Reserved	0	Reads return 0. Writes have no effect.
3-0	MAG_INT_OFF		Magnitude Compare Interrupt Offset. This field indexes the currently highest-priority magnitude compare interrupt. Interrupt 1 has the highest priority and interrupt 3 has the lowest priority among the magnitude compare interrupts. Writes to these bits have no effect. A read from this register clears this register as well as the corresponding magnitude compare interrupt flag in the ADMAGINTFLG register. However, a read from this register in emulation mode does not affect this register or the interrupt status flags. Any operation mode read:
		0	No magnitude compare interrupt is pending.
		1h	Magnitude compare interrupt # 1 is pending.
		2h	Magnitude compare interrupt # 2 is pending.
		3h	Magnitude compare interrupt # 3 is pending.
		4h-Fh	Reserved.

22.3.60 ADC Event Group FIFO Reset Control Register (ADEVFIFORESETCR)

ADC Event Group FIFO Reset Control Register (ADEVFIFORESETCR) is shown in Figure 22-91 and described in Table 22-66.

Figure 22-91. ADC Event Group FIFO Reset Control Register (ADEVFIFORESETCR) [offset = 168h]

31 1	0
Reserved	EV_FIFO_RESET
R-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 22-66. ADC Event Group FIFO Reset Control Register (ADEVFIFORESETCR) Field Descriptions

Bit	Field	Value	Description
31-1	Reserved	0	Reads return 0. Writes have no effect.
0	EV_FIFO_RESET		ADC Event Group FIFO Reset. The application can set this bit in case of an overrun condition. This allows the ADC module to overwrite the contents of the Event Group results memory starting from the first location. When this bit is set to 1, the ADC module resets its internal Event Group results memory pointers. Then this bit automatically gets cleared, so that the ADC module allows the Event Group results memory to be overwritten only once each time this bit is set to 1. As a result, the EV_FIFO_RESET bit will always be read as a 0. The EV_FIFO_RESET bit will only have the desired effect when the Event Group results memory is in an overrun condition. It must be used when the data already available in the results memory can be discarded. If the application needs the Event Group memory to always be overwritten with the latest available conversion results, then the OVR_EV_RAM_IGN bit in the Event Group operating mode control register (ADEVMODECR) needs to be set to 1.

22.3.61 ADC Group1 FIFO Reset Control Register (ADG1FIFORESETCR)

ADC Group1 FIFO Reset Control Register (ADG1FIFORESETCR) is shown in Figure 22-92 and described in Table 22-67.

Figure 22-92. ADC Group1 FIFO Reset Control Register (ADG1FIFORESETCR) [offset = 16Ch]

31 1	0
Reserved	G1_FIFO_RESET
R-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 22-67. ADC Group1 FIFO Reset Control Register (ADG1FIFORESETCR) Field Descriptions

Bit	Field	Value	Description
31-1	Reserved	0	Reads return 0. Writes have no effect.
0	G1_FIFO_RESET		ADC Group1 FIFO Reset. The application can set this bit in case of an overrun condition. This allows the ADC module to overwrite the contents of the Group1 results memory starting from the first location. When this bit is set to 1, the ADC module resets its internal Group1 results memory pointers. Then this bit automatically gets cleared, so that the ADC module allows the Group1 results memory to be overwritten only once each time this bit is set to 1. As a result, the G1_FIFO_RESET bit will always be read as a 0. The G1_FIFO_RESETbit will only have the desired effect when the Group1 results memory is in an overrun condition. It must be used when the data already available in the results memory can be discarded. If the application needs the Group1 memory to always be overwritten with the latest available conversion results, then the OVR_G1_RAM_IGN bit in the Group1 operating mode control register (ADG1MODECR) needs to be set to 1.

22.3.62 ADC Group2 FIFO Reset Control Register (ADG2FIFORESETCR)

ADC Group2 FIFO Reset Control Register (ADG2FIFORESETCR) is shown in Figure 22-93 and described in Table 22-68.

Figure 22-93. ADC Group2 FIFO Reset Control Register (ADG2FIFORESETCR) [offset = 170h]

31 1	0
Reserved	G2_FIFO_RESET
R-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 22-68. ADC Group2 FIFO Reset Control Register (ADG2FIFORESETCR) Field Descriptions

Bit	Field	Value	Description
31-1	Reserved	0	Reads return 0. Writes have no effect.
0	G2_FIFO_RESET		ADC Group2 FIFO Reset. The application can set this bit in case of an overrun condition. This allows the ADC module to overwrite the contents of the Group2 results memory starting from the first location. When this bit is set to 1, the ADC module resets its internal Group2 results memory pointers. Then this bit automatically gets cleared, so that the ADC module allows the Group2 results memory to be overwritten only once each time this bit is set to 1. As a result, the G2_FIFO_RESET bit will always be read as a 0. The G2_FIFO_RESET bit will only have the desired effect when the Group2 results memory is in an overrun condition. It must be used when the data already available in the results memory can be discarded. If the application needs the Group2 memory to always be overwritten with the latest available conversion results, then the OVR_G2_RAM_IGN bit in the Group2 operating mode control register (ADG2MODECR) needs to be set to 1.

22.3.63 ADC Event Group RAM Write Address Register (ADEVRAMWRADDR)

ADC Event Group RAM Write Address Register (ADEVRAMWRADDR) is shown in Figure 22-94 and described in Table 22-69.

Figure 22-94. ADC Event Group RAM Write Address Register (ADEVRAMWRADDR) [offset = 174h]

31 9	8 0
Reserved	EV_RAM_ADDR
R-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 22-69. ADC Event Group RAM Write Address Register (ADEVRAMWRADDR) Field Descriptions

Bit	Field	Value	Description
31-9	Reserved	0	Reads return 0. Writes have no effect.
8-0	EV_RAM_ADDR		Event Group results memory write pointer. This field shows the address of the location where the next Event Group conversion result will be stored. This is specified in terms of the buffer number. The application can read this register to determine the number of valid Event Group conversion results available until that time.

22.3.64 ADC Group1 RAM Write Address Register (ADG1RAMWRADDR)

ADC Group1 RAM Write Address Register (ADG1RAMWRADDR) is shown in Figure 22-95 and described in Table 22-70.

Figure 22-95. ADC Group1 RAM Write Address Register (ADG1RAMWRADDR) [offset = 178h]

31 9	8 0
Reserved	G1_RAM_ADDR
R-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 22-70. ADC Group1 RAM Write Address Register (ADG1RAMWRADDR) Field Descriptions

Bit	Field	Value	Description
31-9	Reserved	0	Reads return 0. Writes have no effect.
8-0	G1_RAM_ADDR		Group1 results memory write pointer. This field shows the address of the location where the next Group1 conversion result will be stored. This is specified in terms of the buffer number. The application can read this register to determine the number of valid Group1 conversion results available until that time.

22.3.65 ADC Group2 RAM Write Address Register (ADG2RAMWRADDR)

ADC Group2 RAM Write Address Register (ADG2RAMWRADDR) is shown in Figure 22-96 and described in Table 22-71.

Figure 22-96. ADC Group2 RAM Write Address Register (ADG2RAMWRADDR) [offset = 17Ch]

31 9	8 0
Reserved	G2_RAM_ADDR
R-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 22-71. ADC Group2 RAM Write Address Register (ADG2RAMWRADDR) Field Descriptions

Bit	Field	Value	Description
31-9	Reserved	0	Reads return 0. Writes have no effect.
8-0	G2_RAM_ADDR		Group2 results memory write pointer. This field shows the address of the location where the next Group2 conversion result will be stored. This is specified in terms of the buffer number. The application can read this register to determine the number of valid Group2 conversion results available until that time.

22.3.66 ADC Parity Control Register (ADPARCR)

ADC Parity Control Register (ADPARCR) is shown in Figure 22-97 and described in Table 22-72.

Figure 22-97. ADC Parity Control Register (ADPARCR) [offset = 180h]

31	16
Reserved	
R-0	

15	9	8	7	4	3	0
Reserved		TEST	Reserved		PARITY_ENA	
R-0		R/WP-0	R-0		R/WP-5h	

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privileged mode only; -*n* = value after reset

Table 22-72. ADC Parity Control Register (ADPARCR) Field Descriptions

Bit	Field	Value	Description
31-9	Reserved	0	Reads return 0. Writes have no effect.
8	TEST		This bit maps the parity bits into the ADC results' RAM frame so that the application can access them.
			Any operation mode read, privileged mode write:
		0	The parity bits are not memory-mapped.
		1	The parity bits are memory-mapped.
7-4	Reserved	0	Reads return 0. Writes have no effect.
3-0	PARITY_ENA		Enable parity checking. These bits enable the parity check on read operations and the parity calculation on write operations to the ADC results memory.
			If parity checking is enabled and a parity error is detected the ADC module sends a parity error signal to the System module.
			Any operation mode read, privileged mode write:
		5h	Parity check is disabled.
		All other values	Parity check is enabled.

22.3.67 ADC Parity Error Address Register (ADPARADDR)

ADC Parity Error Address Register (ADPARADDR) is shown inFigure 22-98 and described in Table 22-73.

Figure 22-98. ADC Parity Error Address Register (ADPARADDR) [offset = 184h]

31 16
Reserved
R-0

15 12	11 2	1 0
Reserved	ERROR_ADDRESS	Reserved
R-0	R-U	R-0

LEGEND: R = Read only; -*n* = value after reset; U = value after reset is unknown

Table 22-73. ADC Parity Error Address Register (ADPARADDR) Field Descriptions

Bit	Field	Value	Description
31-12	Reserved	0	Reads return 0. Writes have no effect.
11-2	ERROR_ADDRESS		These bits hold the address of the first parity error generated in the ADC results' RAM. This error address is frozen from being updated until it is read by the application. In emulation mode, this address is maintained frozen even when read.
1-0	Reserved	0	Reads return 0. Writes have no effect. Reading [11:0] provides the 32-bit aligned address.

22.3.68 ADC Power-Up Delay Control Register (ADPWRUPDLYCTRL)

Figure 22-99 and Table 22-74 describe the ADPWRDLYCTRL register.

Figure 22-99. ADC Power-Up Delay Control Register (ADPWRUPDLYCTRL) [offset = 188h]

31 10	9 0
Reserved	PWRUP_DLY
R-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 22-74. ADC Power-Up Delay Control Register (ADPWRUPDLYCTRL) Field Descriptions

Bit	Field	Value	Description
31-10	Reserved	0	Reads return 0. Writes have no effect.
9-0	PWRUP_DLY		This register defines the number of VCLK cycles that the ADC state machine has to wait after releasing the ADC core from power down before starting a new conversion. Refer to Section 22.2.6.3 for more details.

22.3.69 ADC Event Group Channel Selection Mode Control Register (ADEVCHNSELMODECTRL)

Figure 22-100 and Table 22-75 describe the ADEVCHNSELMODECTRL register.

Figure 22-100. ADC Event Group Channel Selection Mode Control Register (ADEVCHNSELMODECTRL) (offset = 190h)

31	4	3	0
Reserved		EV_ENH_CHNSEL_MODE_ENABLE	
R-0		R/W-5h	

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 22-75. ADC Event Group Channel Selection Mode Control Register (ADEVCHNSELMODECTRL) Field Descriptions

Bit	Field	Value	Description
31-4	Reserved	0	Reads return 0. Writes have no effect.
3-0	EV_ENH_CHNSEL_MODE_ENABLE		Enable enhanced channel selection mode for Event group. Refer to Section 22.2.2.2.2 for a description of the enhanced channel selection mode.
		5h	Read: Indicates that the enhanced channel selection mode for Event group is not enabled. The default sequential channel selection mode is used for Event group conversions. Write: Disables the enhanced channel selection mode for Event group and enables the sequential channel selection mode.
		Ah	Read: Indicates that the enhanced channel selection mode for Event group is enabled. Write: Enables the enhanced channel selection mode for Event group.
		All other values	Writing any value other than 5h or Ah to this field has no effect on the selected channel selection mode for the Event group, and the ADC module continues to use the channel selection mode that was previously programmed channel selection mode.

22.3.70 ADC Group1 Channel Selection Mode Control Register (ADG1CHNSELMODECTRL)

Figure 22-101 and Table 22-76 describe the ADG1CHNSELMODECTRL register.

Figure 22-101. ADC Group1 Channel Selection Mode Control Register (ADG1CHNSELMODECTRL) (offset = 194h)

31	4	3	0
Reserved		G1_ENH_CHNSEL_MODE_ENABLE	
R-0		R/W-5h	

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 22-76. ADC Group1 Channel Selection Mode Control Register (ADG1CHNSELMODECTRL) Field Descriptions

Bit	Field	Value	Description
31-4	Reserved	0	Reads return 0. Writes have no effect.
3-0	G1_ENH_CHNSEL_MODE_ENABLE		Enable enhanced channel selection mode for Group1. Refer to Section 22.2.2.2.2 for a description of the enhanced channel selection mode.
		5h	Read: Indicates that the enhanced channel selection mode for Group1 is not enabled. The default sequential channel selection mode is used for Group1 conversions. Write: Disables the enhanced channel selection mode for Group1 and enables the sequential channel selection mode.
		Ah	Read: Indicates that the enhanced channel selection mode for Group1 is enabled. Write: Enables the enhanced channel selection mode for Group1.
		All other values	Writing any value other than 5h or Ah to this field has no effect on the selected channel selection mode for the Group1, and the ADC module continues to use the channel selection mode that was previously programmed channel selection mode.

22.3.71 ADC Group2 Channel Selection Mode Control Register (ADG2CHNSELMODECTRL)

Figure 22-102 and Table 22-77 describe the ADG2CHNSELMODECTRL register.

Figure 22-102. ADC Group2 Channel Selection Mode Control Register (ADG1CHNSELMODECTRL) (offset = 198h)

31 4	3 0
Reserved	G2_ENH_CHNSEL_MODE_ENABLE
R-0	R//W-5h

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 22-77. ADC Group2 Channel Selection Mode Control Register (ADG2CHNSELMODECTRL) Field Descriptions

Bit	Field	Value	Description
31-4	Reserved	0	Reads return 0. Writes have no effect.
3-0	G2_ENH_CHNSEL_MODE_ENABLE		Enable enhanced channel selection mode for Group2. Refer to Section 22.2.2.2.2 for a description of the enhanced channel selection mode.
		5h	Read: Indicates that the enhanced channel selection mode for Group2 is not enabled. The default sequential channel selection mode is used for Group2 conversions. Write: Disables the enhanced channel selection mode for Group2 and enables the sequential channel selection mode.
		Ah	Read: Indicates that the enhanced channel selection mode for Group2 is enabled. Write: Enables the enhanced channel selection mode for Group2.
		All other values	Writing any value other than 5h or Ah to this field has no effect on the selected channel selection mode for the Group2, and the ADC module continues to use the channel selection mode that was previously programmed channel selection mode.

22.3.72 ADC Event Group Current Count Register (ADEVCURRCOUNT)

Figure 22-103 and Table 22-78 describe the ADEVCURRCOUNT register.

Figure 22-103. ADC Event Group Current Count Register (ADEVCURRCOUNT) (offset = 19Ch)

31 5	4 0
Reserved	EV_CURRENT_COUNT
R-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 22-78. ADC Event Group Current Count Register (ADEVCURRCOUNT) Field Descriptions

Bit	Field	Value	Description
31-5	Reserved	0	Reads return 0. Writes have no effect.
4-0	EV_CURRENT_COUNT		CURRENT_COUNT value for the Event group conversions when enhanced channel selection mode is enabled. Refer to Section 22.2.2.2.2 for a description of the enhanced channel selection mode. This register resets to 0 on any of the following conditions: • A peripheral reset occurs • An ADC software reset occurs via the ADC Reset Control Register (ADRSTCR) • EV_CURRENT_COUNT becomes equal to EV_MAX_COUNT • Application writes zeros to ADEVCURRCOUNT register • Event group's result RAM is reset A read from the ADEVCURRCOUNT register returns the value of the current index into the Event group's look-up table.

22.3.73 ADC Event Group Maximum Count Register (ADEVMAXCOUNT)

Figure 22-104 and Table 22-79 describe the ADEVMAXCOUNT register.

Figure 22-104. ADC Event Group Maximum Count Register (ADEVMAXCOUNT) (offset = 1A0h)

31 5	4 0
Reserved	EV_MAX_COUNT
R-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 22-79. ADC Event Group Maximum Count Register (ADEVMAXCOUNT) Field Descriptions

Bit	Field	Value	Description
31-5	Reserved	0	Reads return 0. Writes have no effect.
4-0	EV_MAX_COUNT		MAX_COUNT value for the Event group conversions when enhanced channel selection mode is enabled. Refer to Section 22.2.2.2.2 for a description of the enhanced channel selection mode. It is recommended to clear the Event group's CURRENT_COUNT register (ADEVCURRCOUNT) whenever the EV_MAX_COUNT is changed.

22.3.74 ADC Group1 Current Count Register (ADG1CURRCOUNT)

Figure 22-105 and Table 22-80 describe the ADG1CURRCOUNT register.

Figure 22-105. ADC Group1 Current Count Register (ADG1CURRCOUNT) (offset = 1A4h)

31 5	4 0
Reserved	G1_CURRENT_COUNT
R-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 22-80. ADC Group1 Current Count Register (ADG1CURRCOUNT) Field Descriptions

Bit	Field	Value	Description
31-5	Reserved	0	Reads return 0. Writes have no effect.
4-0	G1_CURRENT_ COUNT		CURRENT_COUNT value for the Group1 conversions when enhanced channel selection mode is enabled. Refer to Section 22.2.2.2.2 for a description of the enhanced channel selection mode. This register resets to 0 on any of the following conditions: • A peripheral reset occurs • An ADC software reset occurs via the ADC Reset Control Register (ADRSTCR) • G1_CURRENT_COUNT becomes equal to G1_MAX_COUNT • Application writes zeros to ADG1CURRCOUNT register • Group1's result RAM is reset A read from the ADG1CURRCOUNT register returns the value of the current index into the Group1's look-up table.

22.3.75 ADC Group1 Maximum Count Register (ADG1MAXCOUNT)

Figure 22-106 and Table 22-81 describe the ADG1MAXCOUNT register.

Figure 22-106. ADC Group1 Maximum Count Register (ADG1MAXCOUNT) (offset = 1A8h)

31 5	4 0
Reserved	G1_MAX_COUNT
R-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 22-81. ADC Group1 Maximum Count Register (ADG1MAXCOUNT) Field Descriptions

Bit	Field	Value	Description
31-5	Reserved	0	Reads return 0. Writes have no effect.
4-0	G1_MAX_ COUNT		MAX_COUNT value for the Group1 conversions when enhanced channel selection mode is enabled. Refer to Section 22.2.2.2.2 for a description of the enhanced channel selection mode. It is recommended to clear the Group1's CURRENT_COUNT register (ADG1CURRCOUNT) whenever the G1_MAX_COUNT is changed.

22.3.76 ADC Group2 Current Count Register (ADG2CURRCOUNT)

Figure 22-107 and Table 22-82 describe the ADG2CURRCOUNT register.

Figure 22-107. ADC Group2 Current Count Register (ADG2CURRCOUNT) (offset = 1ACh)

31 5	4 0
Reserved	G2_CURRENT_COUNT
R-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 22-82. ADC Group2 Current Count Register (ADG2CURRCOUNT) Field Descriptions

Bit	Field	Value	Description
31-5	Reserved	0	Reads return 0. Writes have no effect.
4-0	G2_CURRENT_COUNT		CURRENT_COUNT value for the Group2 conversions when enhanced channel selection mode is enabled. Refer to Section 22.2.2.2.2 for a description of the enhanced channel selection mode. This register resets to 0 on any of the following conditions: • A peripheral reset occurs • An ADC software reset occurs via the ADC Reset Control Register (ADRSTCR) • G2_CURRENT_COUNT becomes equal to G2_MAX_COUNT • Application writes zeros to ADG2CURRCOUNT register • Group2's result RAM is reset A read from the ADG2CURRCOUNT register returns the value of the current index into the Group2's look-up table.

22.3.77 ADC Group2 Maximum Count Register (ADG2MAXCOUNT)

Figure 22-108 and Table 22-83 describe the ADG2MAXCOUNT register.

Figure 22-108. ADC Group2 Maximum Count Register (ADG2MAXCOUNT) (offset = 1B0h)

31 5	4 0
Reserved	G2_MAX_COUNT
R-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 22-83. ADC Group2 Maximum Count Register (ADG2MAXCOUNT) Field Descriptions

Bit	Field	Value	Description
31-5	Reserved	0	Reads return 0. Writes have no effect.
4-0	G2_MAX_COUNT		MAX_COUNT value for the Group2 conversions when enhanced channel selection mode is enabled. Refer to Section 22.2.2.2.2 for a description of the enhanced channel selection mode. It is recommended to clear the Group2's CURRENT_COUNT register (ADG2CURRCOUNT) whenever the G2_MAX_COUNT is changed.

High-End Timer (N2HET) Module

This chapter provides a general description of the High-End Timer (N2HET). The N2HET is a software-controlled timer with a dedicated specialized timer micromachine and a set of 30 instructions. The N2HET micromachine is connected to a port of up to 32 input/output (I/O) pins.

NOTE: This chapter describes a superset implementation of the N2HET module that includes features and functionality that require DMA. Since not all devices have DMA capability, consult your device-specific datasheet to determine the applicability of these features and functions to your device being used.

Topic **Page**

23.1 Overview

The N2HET is a fifth-generation Texas Instruments (TI) advanced intelligent timer module. It provides an enhanced feature set compared to previous generations.

This timer module provides sophisticated timing functions for real-time applications such as engine management or motor control. The high resolution hardware channels allow greater accuracy for widely used timing functions such as period and pulse measurements, output compare, and PWMs.

The reduced instruction set, based mostly on very simple, but comprehensive instructions, improves the definition and development cycle time of an application and its derivatives. The N2HET breakpoint feature, combined with various stop capabilities, makes the N2HET software application easy to debug.

23.1.1 Features

- Programmable timer for input and output timing functions
- Reduced instruction set (30 instructions) for dedicated time and angle functions
- Up to maximum of 128 96-bit words of instruction RAM protected by parity. Check your datasheet for the actual number of words implemented.
- User defined configuration of 25-bit virtual counters for timer, event counters and angle counters
- 7-bit hardware counters for each pin allow up to 32-bit resolution in conjunction with the 25-bit virtual counters
- Up to 32 pins usable for input signal measurements or output signal generation
- Programmable suppression filter for each input pin with adjustable suppression window
- Low CPU overhead and interrupt load
- Efficient data transfer to or from the CPU memory with dedicated High-End-Timer Transfer Unit (HTU) or DMA
- Diagnostic capabilities with different loopback mechanisms and pin status readback functionality
- Hardware Angle Generator (HWAG)

23.1.2 Major Advantages

In addition to classic time functions such as input capture or multiple PWMs, higher-level time functions can be easily implemented in the timer program main loop. Higher-level time functions include angle driven wave forms, angle- and time-driven pulses, and input pulse width modulation (PWM) duty cycle measurement.

Because of these high-level functions, data exchanges with the CPU are limited to the fundamental parameters of the application (periods, pulse widths, angle values, etc.); and the real-time constraints for parameter communication are dramatically minimized; for example, few interrupts are required and asynchronous parameter updates are allowed.

The reduced instruction set and simple execution flow control make it simple and easy to develop and modify programs. Simple algorithms can embed the entire flow control inside the N2HET program itself. More complex algorithms can take advantage of the CPU access to the N2HET RAM. With this, the CPU program can make calculations and can modify the timer program flow by changing the data and control fields of the N2HET RAM. CPU access to the N2HET RAM also improves the debug and development of timer programs. The CPU program can stop the N2HET and view the contents of the program, control, and data fields that reside in the N2HET RAM.

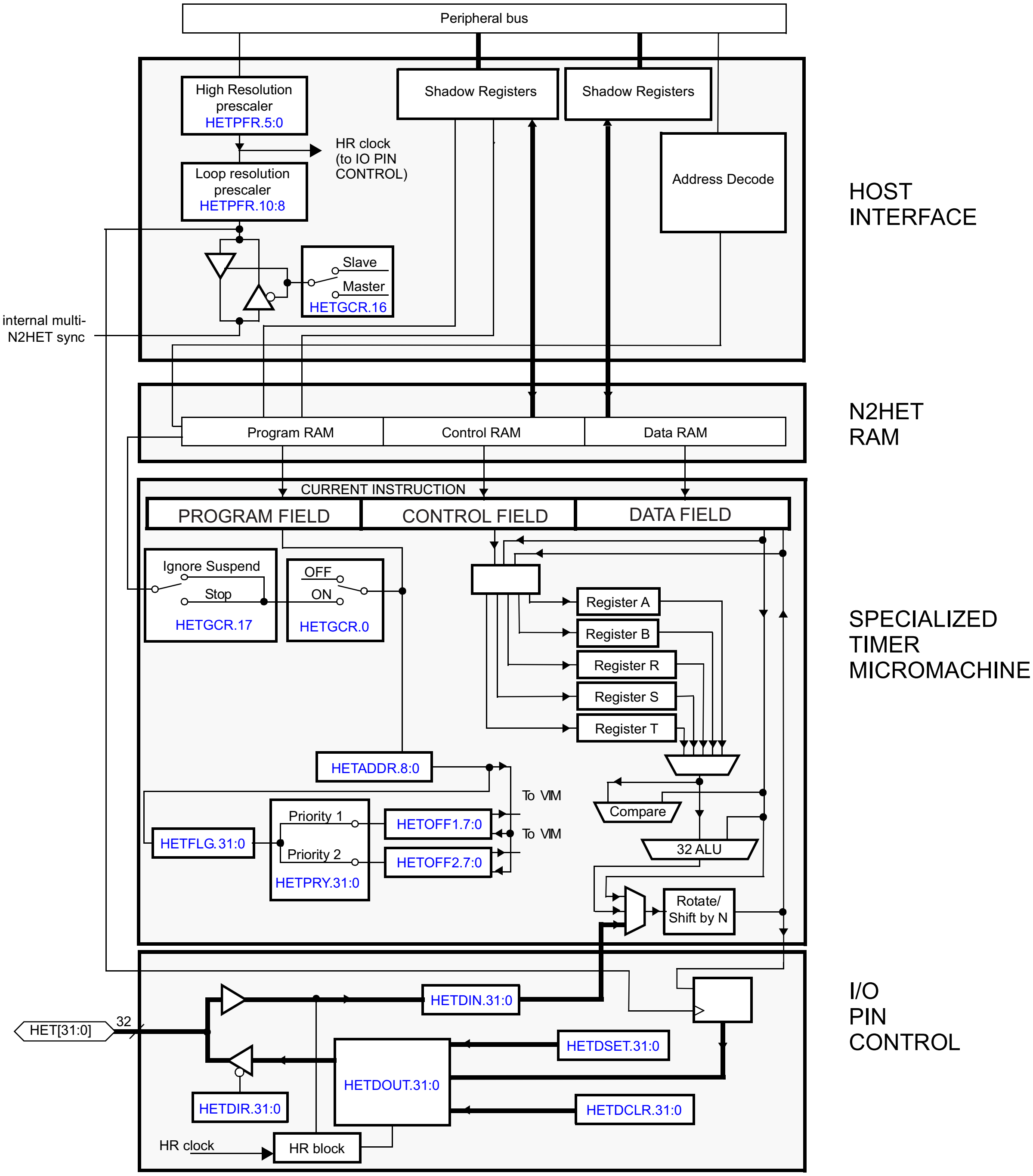
Finally, the modular structure provides maximum flexibility to address a wide range of applications. The timer resolution can be selected from two cascaded prescalers to adjust the loop resolution and HR clocks. The 32 I/O pins can provide any combination of input, period or pulse capture, and output compare, including high resolution for each channel.

23.1.3 Block Diagram

The N2HET module (see Figure 23-1) comprises four separate components:

- Host interface
- N2HET RAM
- Specialized timer micromachine
- I/O control (the N2HET is attached to an I/O port of up to 32 pins)

Figure 23-1. N2HET Block Diagram



23.1.4 Timer Module Structure and Execution

The timer consists of a specialized micromachine that operates a reduced instruction set. Two 25-bit registers and three 32-bit registers are available to manipulate information such as time, event counts, and angle values. System performance is improved by a wide instruction format (96 bits) that allows the N2HET to fetch the instructional operation code and data in one system cycle, thus increasing the speed at which data can be processed. The typical operations performed in the ALU are additions (count), compares, and magnitude compares (higher or same).

Each instruction is made up of a 32-bit program field, a 32-bit control field and a 32-bit data field. The N2HET execution unit fetches the complete 96-bit instruction in one cycle and executes it. All instructions include a 9-bit field for specifying the address of the next instruction to be executed. Some instructions also include a 9-bit conditional address, which is used as the next address whenever a particular condition is true. This makes controlling the flow of an N2HET program inexpensive; in many cases a separate branch instruction is not required.

The interface to the host CPU is based on both communication memory and control registers. The communication memory includes timer instructions (program and data). This memory is typically initialized by the CPU or DMA after reset before the timer starts execution. Once the timer program is loaded into the memory, the CPU starts the timer execution, and typically data parameters are then read or written into the timer memory. The control registers include bits for selecting timer clock, configuring I/O pins, and controlling the timer module.

The programmer implements timer functions by combining instructions in specific sequences. For instance, a single count (CNT) instruction implements a timer. A simple PWM generator can be implemented with a two instruction sequence: CNT and compare (ECMP or MCMP). A complex time function may include many instructions in the sequence. The total timer program is a set of instructions executed sequentially, one after the other. Reaching the end, the program must roll to the first instruction so that it behaves as a loop. The time for a loop to execute is referred to as a *loop resolution clock cycle* or *loop resolution period (LRP).* When the N2HET rolls over to the first instruction, the timer waits for the loop resolution clock to restart the execution of the loop to ensure that only one loop is executed for each loop resolution clock.

The longest path through an N2HET program must be completed within the loop resolution clock (LRP). Otherwise, the program will execute unpredictably because some instructions will not be executed each time through the loop. This effect creates a strong link between the accuracy of the timer functions and the number of functions (the number of instructions) the timer can perform. High resolution (HR) hardware timer extensions are available for each of the N2HET pins to help overcome this limitation.

The high resolution hardware timers operate from the *high resolution clock*, which may be configured for frequency multiples between 2 and 128 times the loop resolution clock frequency. This extending the resolution of timer events and measurements well beyond what is possible with only loop resolution instructions.

Most of the commonly used N2HET instructions can operate either at loop resolution or high resolution; with the restriction that for each pin at most one high resolution instruction can be executed per loop resolution period.

Certain instructions (MOV32, ADM32, ...) can modify the data fields of other instructions. This feature enables the N2HET program to implement double buffering on capture and compare functions. For example, an ECMP compare instruction can be followed by a MOV32 instruction that is conditionally executed when the ECMP instruction matches. The host CPU can update the next compare value by writing asynchronously to the data field of the MOV32 instruction instead of writing directly to the data field of the ECMP instruction. The copy from the buffer (MOV32 data field) to the compare register (ECMP data field) will occur when the MOV32 instruction is actually executed which occurs after the ECMP instruction matches its current compare value. This is the same behavior as one would expect from a double buffered hardware compare register.

Other instructions (MOV64, RADM64) can modify both the control and data fields of other instructions. This allows the N2HET to implement toggle functionality. For example, an ECMP instruction can be followed by a pair of MOV64 instructions. The MOV64 instruction updates the data field of the ECMP instruction to implement the double buffering behavior. But it also updates the control field of the ECMP instruction which allows it to change things like pin action and the conditional address. If one MOV64 instruction configures the ECMP pin action to SET while the second changes it to CLEAR, and the two MOV64 instructions update the conditional address to point to each other, then a single ECMP instruction can be used to toggle a pin each time the compare match occurs.

23.1.5 Performance

Most instructions execute in one cycle, but a few take two or three cycles.

The N2HET can generate many complex output waveforms without CPU interrupts. Where special algorithms are needed following a specific event (for example, missing teeth or a short/long input signal), a minimal number of interrupts to the CPU are needed freeing the CPU for other tasks.

23.1.6 N2HET Compared to NHET

N2HET enhancements from NHET include:

- Eight new instructions: ADD, ADC, SUB, SBB, AND, OR, XOR, RCNT
- Full set of ALU flags Carry (C), Negative (N), Zero (Z), Overflow (V)
- Branch instruction (BR) extended to support signed and unsigned arithmetic comparison conditions
- Two additional 32-bit temporary working registers R, S.
- New HETAND register for AND-Sharing of High Resolution structure between pairs of pins
- Improved high resolution PCNT instruction

23.1.7 NHET and N2HET Compared to HET

Compared to the HET module, the N2HET contains all of the enhancements described in Section 23.1.6 plus the following additional enhancements:

- New Interrupt Enable Set and Clear registers
- Capability to generate requests to the DMA module or the HET Transfer Unit (HTU) including new Request Enable Set and Clear registers
- N2HET RAM parity error detection
- Suppression filters for each of the 32 I/O channel and control register to configure the limiting frequency and counter clock
- Enhanced edge detection hardware that does not rely on the previous bit field in the control word of the N2HET instruction.
- The next, conditional and remote addresses are extended from 8 to 9 bits
- The loop resolution data fields are extended from 20 to 25 bits
- The high resolution data fields are extended from 5 to 7 bits
- Instructions with an adequate condition are able to specify the number of the request line, which triggers either the HET Transfer Unit (HTU) or the DMA module
- The CNT instruction provides a bit, which allows to configure either an equal comparison or a greater or equal comparison when comparing the selected register value with the Max-value
- The MOV32 instruction provides a new bit. If set to one the MOV32 will only perform the move, when the Z-flag is set. If set to zero the MOV32 will perform the move whenever it is executed (independent on the state of the Z-flag)
- There is a new instruction WCAPE, which is a combination of a time stamp and an edge counter
- New Open Drain, Pull Disable, and Pull Select registers

23.1.8 Instructions Features

The N2HET has the following instructions features:

- N2HET uses a RISC-based specialized timer micromachine to carry out a set of 30 instructions
- Instructions are implemented in a Very Long Instruction Word (VLIW) format (96-bits wide)
- The N2HET program execution is self-driven by external or internal events, branching to special routines based on input edges or output compares
- Instructions point to the next instruction executed, eliminating the need for a program counter
- Several instructions can change the program flow based on internal or external conditions

23.1.9 Program Usage

The N2HET instructions/program can be assembled with the N2HET assembler. The assembler generates a C-structure which can be included into the main application program. The application has to copy the content of the structure into the N2HET RAM, set up necessary registers and start the N2HET program execution. In addition to the C-structure, the assembler generates also a header file which makes it easy for the main application to access the different instructions and change for example the duty cycle of a PWM or read out the captured value of a specific signal edge.

23.2 N2HET Functional Description

The N2HET contains RAM into which N2HET code is loaded. The N2HET code is run by the specialized timer micromachine. The host interface and I/O control provide an interface to the CPU and external pins respectively.

23.2.1 Specialized Timer Micromachine

The N2HET has its own instruction set, detailed in Section 23.6.1. The timer micromachine reads each instruction from the N2HET RAM. The program and control fields contain the instructions for how the specialized timer micromachine executes the command. For most instructions, the data field stores the information that needs to be manipulated.

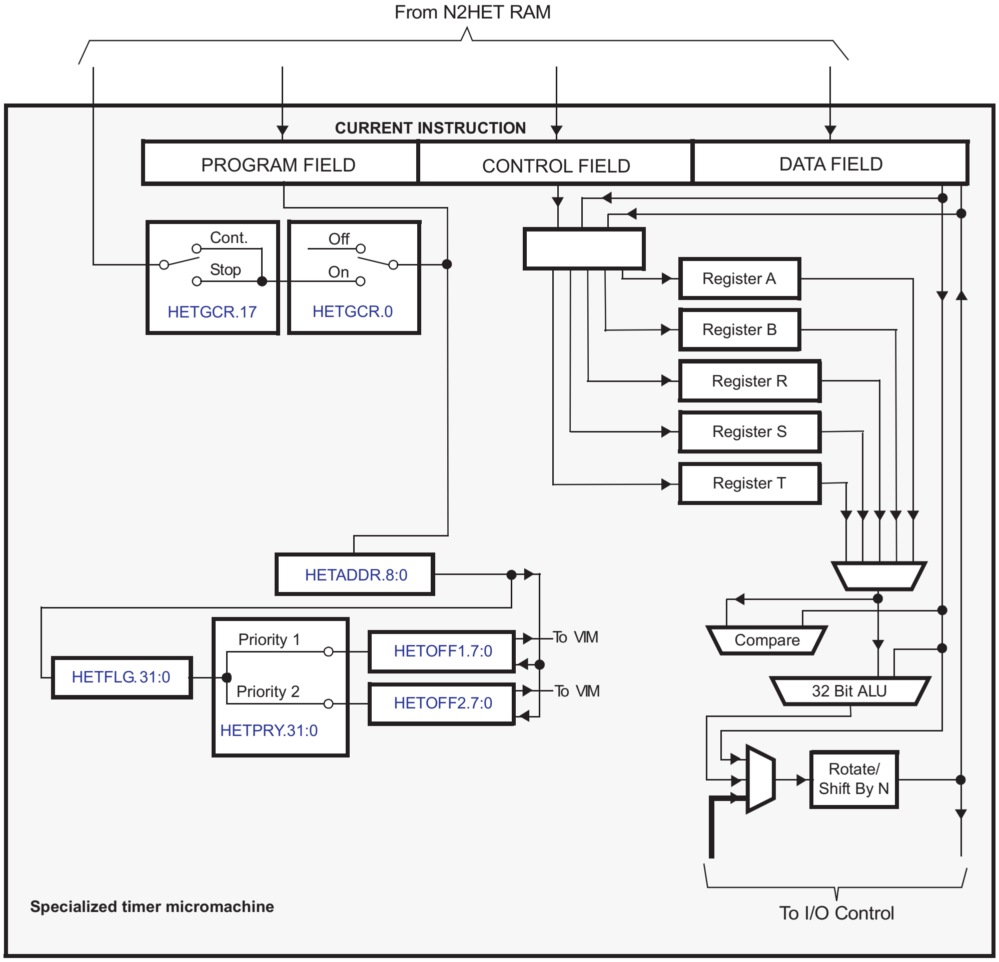
The specialized timer micromachine executes the instructions stored in the N2HET RAM sequentially. The N2HET program execution is self-driven by external or internal events. This means that input edges or output compares may force the program to branch to special routines using a conditional address.

Figure 23-2 shows some of the major operations that the N2HET can carry out, namely compares, captures, angle functions, additions, and shifts. The N2HET contains five registers (A, B, R, S, and T) used to hold compare or counter values and are used by the N2HET instructions. Data may be taken from the registers or the data field for manipulation; likewise, the data may be returned to the registers or the data field.

23.2.1.1 Time Slots and Resolution Loop

Each instruction requires a specific number of cycles or time slots to execute. The resolution specified in the prescaler bitfields determines the timer accuracy. All input captures, event counts, and output compares are executed once in each resolution loop. HR captures and compares are possible (up to N2HET clock accuracy) on the HR I/O pins. For more information about the HR I/O structure, see Section 23.2.5.

Figure 23-2. Specialized Timer Micromachine



23.2.1.2 Program Loop Time

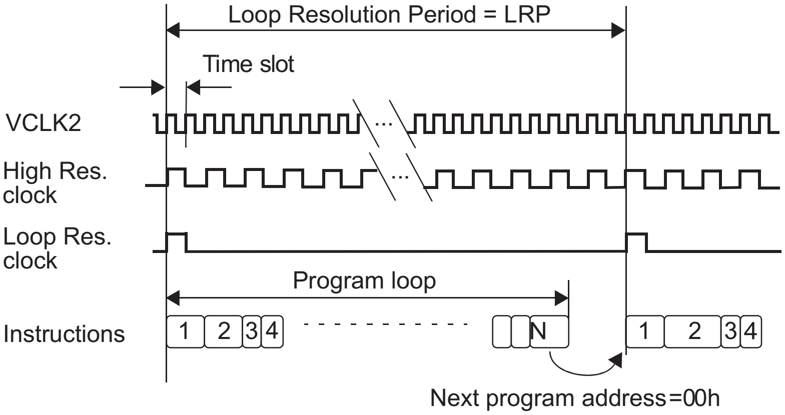
The program loop time is the sum of all cycles used for instruction execution. This time may vary from one loop to another if the N2HET program includes conditionally executed instructions.

The timer program restarts on every resolution loop. The start address is fixed at N2HET RAM address 00h. The longest path through a program must fit within one loop resolution period to guarantee complete accuracy.

The last instruction of a program must branch back to the fixed start address (next program address = 00h). When an N2HET program branches back to address 00h before the end of a loop resolution period, the N2HET detects this and pauses instruction execution until the beginning of the next loop resolution period.

The timing diagram in Figure 23-3 illustrates the program flow execution.

Figure 23-3. Program Flow Timings



23.2.1.3 Instruction Execution Sequence

The execution of a N2HET program begins with the first occurrence of the loop resolution clock, after the N2HET is turned on. At the first and subsequent occurrences of the loop resolution, the instruction at location address 00h is prefetched. The program execution begins at the occurrence of the loop resolution clock and continues executing the instructions until the program branches to 00h location. The instruction is prefetched at location 00h and execution flag is reset. The N2HET pauses instruction execution until the occurrence of the loop resolution clock and resumes normal execution.

N2HET programs must be written so that they complete execution and return to address 00h before the occurrence of the next loop resolution clock. If the N2HET program exceeds this execution time limit, then a program overflow condition occurs as described in Section 23.2.1.4.

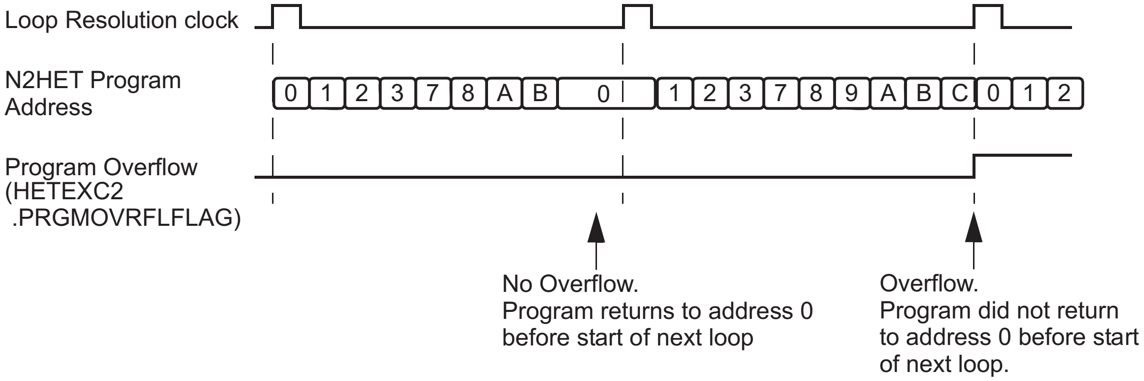
23.2.1.4 Program Overflow Condition

If the number of time slots used in a program loop exceeds the number available time slots in one loop resolution, the timer sets the program overflow interrupt flag located in the HETEXC2 register. To maintain synchronization of the I/Os, this condition should never be allowed to occur in a normal operation. The HETEXC2.PRGMOVRFLFLAG flag provides a mechanism for checking that the condition does not occur during the debug and validation phases.

As Figure 23-4 illustrates, when a program overflow occurs, the currently executing N2HET program sequence is interrupted and restarted at N2HETaddress 0 for the beginning of the next loop resolution clock period. Also, HETEXC2.PRGMOVRFLFLAG is set.

If the instruction that caused the overflow (instruction at address 0xC in Figure 23-4) has any pin actions selected, these pin actions will not be performed. However other actions of the instruction including register and RAM updates will still be performed.

Figure 23-4. Use of the Overflow Interrupt Flag (HETEXC2)


Loop Resolution clock
N2HET Program Address
0 1 2 3 7 8 A B 0 1 2 3 7 8 9 A B C 0 1 2
Program Overflow (HETEXC2 .PRGMOVRFLFLAG)
No Overflow. Program returns to address 0 before start of next loop
Overflow. Program did not return to address 0 before start of next loop.

23.2.1.5 Architectural Restrictions on N2HET Programs

Certain architectural restrictions apply to N2HET programs:

1. The size of an N2HET program must be greater than one instruction.
2. An extra wait state is incurred by any instruction that modifies a field in the next instruction to be executed.
3. Only one instruction (using high resolution) is allowed per high resolution pin.
4. Consecutive break points are not supported. Instructions with break points must have at least a distance of two instructions (for example, at addresses 1, 3, 5, 7, and so on, assuming the program executes linearly)

NOTE: While it would be unusual to code an N2HET program that is only one instruction long, it is trivial to modify such a program to meet the requirement of restriction 1. Simply add a second instruction to the program, which may be a simple branch to zero.

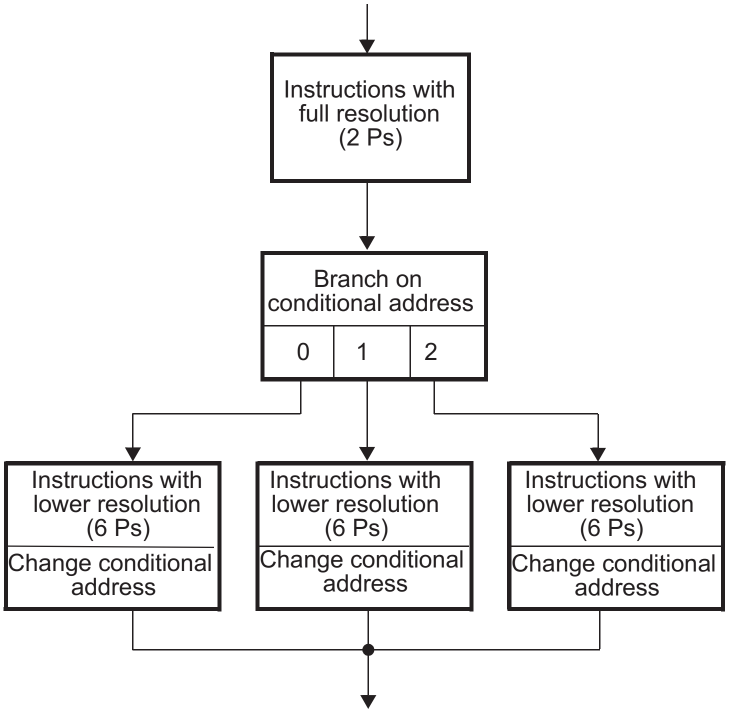
To enforce restriction 3, the high resolution pin structures respond only to the first instruction that is executed matching their pin number with hr_lr=HIGH, regardless of whether or not the en_pin_action field is ON. Subsequent instructions are ignored by the high resolution pin structure for the remainder of the loop resolution period.

23.2.1.6 Multi-Resolution Scheme

The N2HET has the capability to virtually extend the counter width by executing instructions only once every N loop resolution periods. This decreases the timer resolution, but extends the counter range which may be useful when generating or measuring slow signals. Figure 23-5 illustrates how a multi-resolution scheme may be implemented in an N2HET program. An unconditional Branch instruction and an index sequence, using a MOV64 instruction in each low resolution loop, is required to control this particular program flow.

NOTE: HR instructions must be placed in the main (full resolution) loop to ensure proper operation.

Figure 23-5. Multi-Resolution Operation Flow Example


Instructions with full resolution (2 Ps)
Branch on conditional address
0
1
2
Instructions with lower resolution (6 Ps)
Change conditional address
Instructions with lower resolution (6 Ps)
Change conditional address
Instructions with lower resolution (6 Ps)
Change conditional address

23.2.1.7 Debug Capability

The N2HET supports breakpoints to allow you to more easily debug your N2HET program. Figure 23-6 provides an illustration of the breakpoint mechanism.

The steps to enable an N2HET breakpoint are:

1. Make sure the device nTRST pin is high, since N2HET breakpoints are disabled whenever this pin is low. (Normally this is handled automatically when a JTAG debugger is attached).
2. Attach a JTAG debugger and connect to the device that has been already programmed with the N2HET code that needs to debugged. (downloading to on-chip flash is outside the scope of this section).
3. Execute the CPU program at least until the point where the N2HET program RAM has been initialized by the CPU.
4. Open a memory window in the N2HET registers.
5. Make sure HETEXC2.DEBUGSTATUSFLAG bit is cleared.
6. Open a memory window on the N2HET RAM
7. Set bit 22 in the program field of the instruction(s) on which you wish to break. Note that this instruction will be executed **before** the N2HET is halted - slightly different from how CPU breakpoints behave.
8. Make sure the CPU and N2HET are running, if they are halted then restart the CPU through the JTAG emulator (N2HET will start when the CPU starts).
9. Both the CPU and N2HET will halt when breakpoint is reached.

When the N2HET is halted, its state machines are frozen but all of the N2HET control registers can be accessed through the JTAG emulator interface.

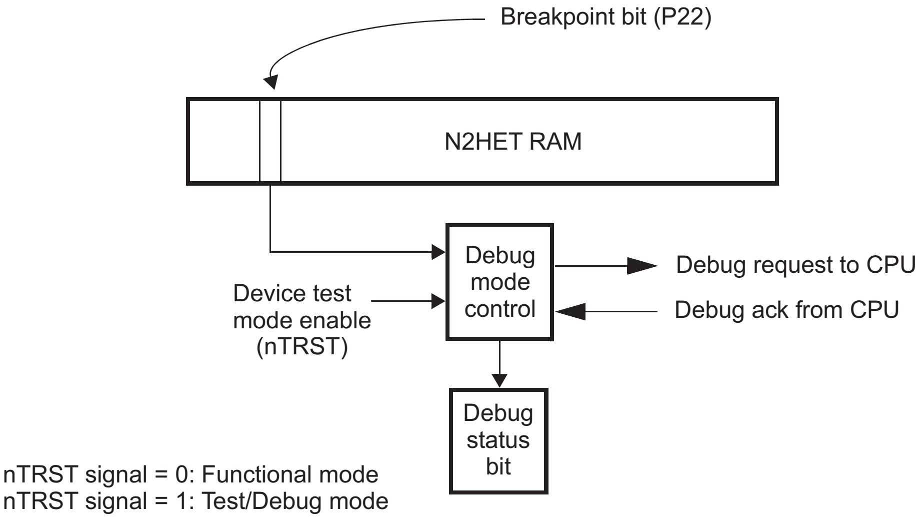
The current N2HET instruction address can be inspected by reading the HETADDR register; this should be pointing to the instruction that caused the breakpoint.

The N2HET internal working registers (A,B,R,S,T) are not directly visible through the JTAG emulator interface. If the content of these registers needs to be inspected, it is best to add an instruction like MOV32 which copies the register value to the N2HET RAM. This RAM location can be inspected when the N2HET halts.

To restart execution of both the CPU and the N2HET from the halted state:

1. Clear HETEXC2.DEBUGSTATUSFLAG bit.
2. Clear bit 22 in the program field of the instruction on which the breakpoint was reached.
3. Restart the CPU through the normal JTAG emulator procedure (‘Run’ or ‘Go’). The N2HET will automatically start executing when it sees that the CPU has exited the debug state.

Figure 23-6. Debug Control Configuration



NOTE: Consecutive break points are not supported. Instructions with break points must have at least a distance of two instructions (for example, at N2HET addresses 1, 3, 5, 7, and so on)

23.2.2 N2HET RAM Organization

The N2HET RAM is organized into two sections. The first contains the N2HET program itself. The second contains parity protection bits for the N2HET program.

Each N2HET instruction is 96-bits wide but aligned to a 128-bit boundary. Instructions consist of three 32-bit fields: Program, Control, and Data. Instructions are separated by a fourth unimplemented address to force alignment to 128-bit boundaries.

The integrity of the N2HET program can be protected by Parity. Parity protection is enabled through the N2HET Parity Control Register (HETPCR).

Table 23-1 shows the base addresses for N2HET RAM and N2HET Parity RAM.

Table 23-1. N2HET RAM Base Addresses

N2HET1 Base Address	N2HET2 Base Address	Memory
0xFF46_0000	0xFF44_0000	N2HET Instruction RAM (Program/Control/Data)
0xFF46_2000	0xFF44_2000	N2HET Parity RAM

23.2.2.1 N2HET RAM Banking

Because the CPU must make updates to the N2HET RAM while the N2HET is executing, for example to update the duty cycle value of a PWM, it is important to understand how the N2HET RAM organization facilitates simultaneous accesses by both the HOST CPU and the N2HET.

The N2HET RAM is implemented as 4 banks of 96-bit wide two port RAM. This means that there a total of 8 ports available; four read and four write. Normally the N2HET will use up to two of these ports at a time. One read port is used to allow the N2HET to prefetch the next N2HET instruction while a write port may be used to update the data or control fields that have changed as a result of executing the current instruction.

N2HET accesses to its own internal RAM are given priority over accesses from an external host (CPU or DMA), this makes N2HET program execution deterministic which is a critical requirement for a timer.

Most N2HET instructions execute in a single cycle. Cases where a wait state impacts the N2HET program execution time are:

- The current N2HET instruction writes data back to the next N2HET in the execution sequence.
- The external host reads from an N2HET instruction where the automatic read-clear option is set, while the N2HET is executing from/on the same address (See Section 23.2.4.3).

Except for the case of automatic read-clear, the external host is stalled when the host and N2HET have a bank conflict. However this will typically only result in a stall of one cycle, due to the N2HET bank ordering which is organized on the N2HET Address least significant bit boundaries (See Table 23-2).

Assuming most of the N2HET program executes linearly through the N2HET Address space; if a bank conflict does exist it is usually resolved in the next cycle as the N2HET program moves to the next bank. N2HET programmers should avoid writing a program that accesses the same bank of N2HET RAM on every cycle, as this could lock the external host out of the N2HET memory completely.

Table 23-2 describes the N2HET memory map, as viewed by the N2HET as well as from the memory space of the host CPU and DMA.

Table 23-2. N2HET RAM Bank Structure

N2HET Address	Host CPU or DMA Address Space				
Instruction	Program Field Address	Control Field Address	Data Field Address	Reserved Address	N2HET RAM Bank
000h	XX0000h	XX0004h	XX0008h	XX000Ch	A
001h	XX0010h	XX0014h	XX0018h	XX001Ch	B
002h	XX0020h	XX0024h	XX0028h	XX002Ch	C
003h	XX0030h	XX0034h	XX0038h	XX003Ch	D
004h	XX0040h	XX0044h	XX0048h	XX004Ch	A
:	:	:	:	:	:
03Fh	XX03F0h	XX03F4h	XX03F8h	XX03FCh	D
040h	XX0400h	XX0404h	XX0408h	XX040Ch	A
:	:	:	:	:	:
1FFh	XX1FF0h	XX1FF4h	XX1FF8h	XX1FFCh	D

NOTE: The external host interface supports any access size for reads, but only 32-bit writes to the N2HET RAM are supported. Reserved addresses should not be accessed, the result of doing so is indeterminate.

23.2.2.2 Parity Checking

The N2HET module can detect parity errors in N2HET RAM. As described in Section 23.2.2 the N2HET allows 32-bit writes only. Therefore N2HET RAM parity checking is implemented using one parity bit per 32-bit field in N2HET RAM.

Even or odd parity selection for N2HET parity detection can be configured in the system module. Parity calculation and checking can be enabled/disabled by a 4-bit key in HETPCR.

During a read access to the N2HET RAM, the parity is calculated based on the data read from the RAM and compared with the good parity value stored in the parity bits. The parity check is performed when the N2HET execution unit makes a read access to N2HET RAM, but also when a different master (for example, CPU, HTU, DMA) performs the read access. If any 32-bit-word fails the parity check then an error is signaled to the ESM module. The N2HET address, which generated the error is detected and is captured in HETPAR for host system debugging. The address is frozen from being updated until it is read by the bus master.

The N2HET execution unit reads the instructions, which are 96-bit wide. They contain the program-, control- and data-field whereby each is 32-bit wide. So when fetching N2HET instructions parity checking is performed on three words in parallel.

If a parity error is detected in two or more words in the same cycle then only one address (word at the lower address) is captured. The captured N2HET address is always aligned to a 32-bit word boundary.

During debug, parity checking is still performed on accesses originating from the on-chip host CPU and DMA. However, parity errors that are detected during an access initiated by the debugger itself are ignored.

23.2.2.3 Parity Error Detection Actions

Detection of a N2HET parity error causes the following actions:

1. An error is signaled to the ESM module.
2. The Parity Address Register (HETPAR) is loaded with the address of the faulty N2HET field.
3. N2HET execution immediately stops. (The instruction that triggered the parity error is not executed.)
4. The Turn-On/Off-Bit in the N2HET Global Configuration Register (HETGCR) is automatically cleared.
5. All N2HET internal flags are cleared.
6. All N2HET pins selected by N2HET Parity Pin Register (HETPPR) enter a predefined safe state.
7. Register HETDOUT is also updated to reflect changes in pin state due to HETPPR.

The safe state for N2HET pins selected through the HETPPR register depends on how the pin is configured in the HETDIR, HETPDR, and HETPSL registers. Table 23-3 explains how the safe state is determined.

Table 23-3. Pin Safe State Upon Parity Error Detection

Safe State	HETDIR	HETPDR	HETPSL
Drive Low	1	0	0
Drive High	1	0	1
High Impedance	1	1	x

23.2.2.4 Testing Parity Detection Logic

To test the parity detection logic, the parity RAM has to be made accessible to the CPU in order to allow a diagnostic program to insert parity errors. The control register bit HETPCR.TEST must be set in order to make the parity RAM accessible. Once HETPCR.TEST is set, the parity bits are accessible as described in Table 23-4.

Each 32-bit N2HET field has its own parity bit in the N2HET Parity RAM as shown in Table 23-4. There are no parity bits for the reserved fields, since there is no physical N2HET RAM for these fields.

Table 23-4. N2HET Parity Bit Mapping

Address N2HET1	Address N2HET2	Bits	
		[31:1]	[0]
0xFF46_2000	0xFF44_2000	Reads 0, Writes have no effect	Instruction 0 Program Field Parity Bit
0xFF46_2004	0xFF44_2004	Reads 0, Writes have no effect	Instruction 0 Control Field Parity Bit
0xFF46_2008	0xFF44_2008	Reads 0, Writes have no effect	Instruction 0 Data Field Parity Bit
0xFF46_200C	0xFF44_200C	Reads 0, Writes have no effect	Read 0
0xFF46_2010	0xFF44_2010	Reads 0, Writes have no effect	Instruction 1 Program Field Parity Bit
....	....	...	...

23.2.2.5 Initialization of Parity RAM

After device power up, the N2HET RAM contents including the parity bits cannot be guaranteed. In order to avoid false parity failures due to the random state in which RAM powers up, the RAM has to be initialized.

Before initializing the N2HET RAM, enable the N2HET parity logic by writing to HETPCR. Then the N2HET Instruction RAM should be initialized. With parity enabled, the N2HET parity RAM will be initialized automatically by N2HET at the same time that the N2HET instruction RAM is initialized by the CPU. Note that loading the N2HET program with parity enabled is also effective.

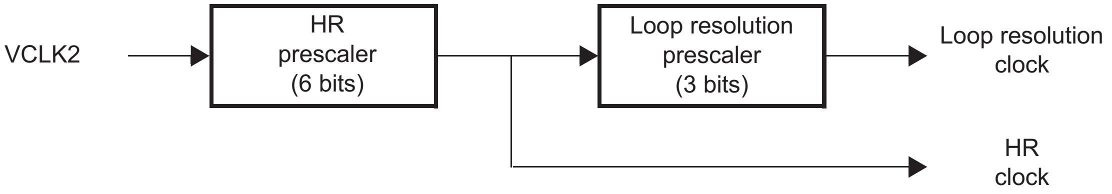
Another possibility to initialize the N2HET memory and its parity bits is, to use the system module to start the automatic initialization of all RAMs on the microcontroller. The RAMs will be initialized to ‘0’. Depending on the even/odd parity selection, the parity bit will be calculated accordingly.

23.2.3 Time Base

All N2HET timings are derived from VCLK2 (see Figure 23-7). Internally N2HET instructions execute at the VCLK2 rate; but the timer loop clock and the high-resolution hardware timer clock can be scaled down from VCLK2. Two prescalers are available to adjust the timer loop resolution clock for the program loop, and the high resolution (HR) clock for the HR I/O counters.

- **Time Slots:** The number of cycles available for instruction execution per loop. Time Slots is the number of VCLK2 cycles in a Loop Resolution Clock.
- **High Resolution Clock:** The high resolution clock is the smallest time increment with which a pin can change it’s state or can be measured in the case of input signals. A 6-bit prescaler dividing VCLK2 by a user-defined HR prescale divide rate (hr) stored in the 6-bit HR prescale factor code (HETPFR). See Table 23-5.
- **Loop Resolution Clock:** The loop resolution clock defines the timebase for executing all instructions in a N2HET program. Since instructions can be conditionally executed, the longest path through the N2HET program must fit into one loop resolution clock period (LRP).A 3-bit prescaler dividing the HR clock by a user-defined loop-resolution prescale divide rate (lr) stored in the 3-bit loop-resolution prescale factor code (HETPFR). See Table 23-5.

Figure 23-7. Prescaler Configuration



The following abbreviations and relations are used in this document:

1. hr: high resolution prescale factor (1, 2, 3, 4,..., 63, 64)
2. lr: loop resolution prescale factor (1, 2, 4, 8, 16, 32, 64,128)
3. ts: Time slots (cycles) available for instruction execution per loop. ts = hr x lr
4. HRP = high resolution clock period HRP = hr × T_{VCLK2} (ns)
5. LRP = loop resolution clock period LRP = lr × HRP (ns)

The loop resolution period (LRP) must be selected to be larger than the number of Time slots (VCLK2 cycles) required to complete the worst-case execution path through the N2HET program. Otherwise a program overflow condition may occur (see Section 23.2.1.4). Because of the relationship of time slots to the hr and lr prescalers as described in item 3 above, increasing either hr or lr increases the number of time slots available for program execution. However, lr would typically be increased first, since increasing hr results in a decrease in timer resolution since it reduces the clock to the High Resolution IO structures.

The divide rates hr and lr can be defined in the HETPFR register. Table 23-5 lists the bit field encodings for the prescale options.

Table 23-5. Prescale Factor Register Encoding

LRPFC - Loop Resolution		HRPFC - High Resolution	
HETPFR[10:8]	Prescale Factor lr	HETPFR[5:0]	Prescale Factor hr
000	/1	000000	/1
001	/2	000001	/2
010	/4	000010	/3
011	/8	000011	/4
100	/16	:	:
101	/32	111101	/62
110	/64	111110	/63
111	/128	111111	/64

23.2.3.1 Determining Loop Resolution

As an example, consider an application that requires high resolution of HRP = 62.5 ns, and loop resolution of LRP = 8 µs, and needs at least 250 time slots for the N2HET application program.

Assuming VCLK2 = 32 MHz, the following shows which divide-by rates and which value in the Prescale Factor Register (HETPFR) is required for the above requirements:

$$hr = 2 \rightarrow HRP = \frac{hr}{VCLK2} = \frac{2}{32MHz} = 62.5ns$$

$$lr = 128 \rightarrow lr \times HRP = 128 \times 62.5ns = 8\ \mu s$$

$$ts = hr \times lr = 2 \times 128 = 256$$

$$hr = 2,\ lr = 128 \rightarrow HETPFR[31:0] = 0x00000701 \qquad (29)$$

In the example above, if the loop resolution period needs to decrease from 8 µs to 4 µs, then only 128 time slots will be available for program execution. The program may need to be restructured as suggested in Section 23.2.1.6.

23.2.3.2 The 7-Bit HR Data Field

The instruction execution examples of ECMP (Section 23.2.5.9), MCMP (Section 23.2.5.10), PCNT (Section 23.2.5.12), PWCNT (Section 23.2.5.11), and WCAP (Section 23.2.5.13) show that the 7-bit HR data field can generate or measure high resolution delays (HR delay) relative to the start of an LRP within one N2HET loop LRP. The last section showed that:

LRP = lr × HRP

There are lr high resolution clock periods (HRP) within the N2HET loop resolution clock period (LRP). If lr = 128 then the HR delay can range from 0 to127 HRP clocks within LRP and all 7 bits of the HR data field are needed. Instead of being limited to measuring and triggering events based on the loop resolution clock period (LRP) the HR extension allows measurements and events to be described in terms fractions of an LRP (down to 1/128 of an LRP). The only limitation is that a maximum of one HR delay can be specified per pin during each loop resolution period.

Table 23-6 shows which bits of the HR data field are not used by the high resolution IO structures if lr is less than 128. In this case the non-relevant bits (LSBs) of the HR data fields will be one of the following:

- Written as 0 for HR capture (for PCNT, WCAP)
- Or interpreted as 0 for HR compare (for ECMP, MCMP. PWCNT)

Table 23-6. Interpretation of the 7-Bit HR Data Field

Loop Resolution Prescale divide rate (lr)	Bits of the HR data field [1]							HRP Cycles delay range
	D[6]	D[5]	D[4]	D[3]	D[2]	D[1]	D[0]	
1	X X X X X X X							0
2	1/2	X X X X X X						0 to 1
4	1/2	1/4	X X X X X					0 to 3
8	1/2	1/4	1/8	X X X X				0 to 7
16	1/2	1/4	1/8	1/16	X X X			0 to 15
32	1/2	1/4	1/8	1/16	1/32	X X		0 to 31
64	1/2	1/4	1/8	1/16	1/32	1/64	X	0 to 63
128	1/2	1/4	1/8	1/16	1/32	1/64	1/128	0 to 127

[1] X = Non-relevant bit (treated as '0')

23.2.3.2.1 Example:

Prescale Factor Register (HETPFR) = 0x0300

—> lr = 8 —> LRP = 8 × HRP

Assumption: HR data field = 0x50 = 1010000b

lr = 8 —> Bits D[3:0] are ignored —> HR delay = 101b = 5 HRPs

or by using the calculation with weight factors:

HR Delay
= lr · (D[6] · 1/2 + D[5] · 1/4 + D[4] · 1/8 + D[3] · 1/16 + D[2] · 1/32 + D[1] · 1/64 + D[0] · 1/128)
= 8 · (1 · 1/2 + 0 · 1/4 + 1 · 1/8 + 0 · 1/16 + 0 · 1/32 + 0 · 1/64 + 0 · 1/128)
= 5 HRPs

23.2.4 Host Interface

The host interface controls all communications between timer-RAM and masters accessing the N2HET RAM. It includes following components:

23.2.4.1 Host Accesses to N2HET RAM

The host interface supports the following types of accesses to N2HET RAM:

- Read accesses of 8, 16, or 32 bits
- Read accesses of 64-bits that follow the shadow register sequence described in Section 23.2.4.2.
- Write accesses of 32 bits

Writes of 8 or 16 bits to N2HET RAM by an external host are not supported.

23.2.4.2 64-bit Read Access

The consecutive read of a control field CF(n) and a data field DF(n) of the same instruction (n) performed by the same master (for example, CPU, DMA, or any other master) is always done as a simultaneous 64-bit read access. This means that at the same time CF(n) is read, DF(n) is loaded in a shadow register. So the second access will read DF(n) from the shadow register instead of the N2HET RAM.

In general a 64-bit read access of one master could be interrupted by a 64-bit read access of another master. A total of three shadow registers are available. Therefore up to three masters can perform 64-bit reads in an interleaved manner (Master1 CF, Master2 CF, Master3 CF, Master1 DF, Master2 DF, Master3 DF).

If all three shadow registers are activated and a 4th master performs a CF or DF read it will result in an address error and the RAM access will not happen. Other access types by a fourth master (reads from the PF field or writes to any of the fields) will occur because these access types do not require an available shadow register resource to complete.

23.2.4.3 Automatic Read Clear Feature

The N2HET provides a feature allowing to automatically clear the data field immediately after the data field is read by the external host CPU (or DMA). This feature is implemented via the *control bit*, which is located in the control field (bit C26). This is a static bit that can be used by any instruction, and specified in the N2HET program by adding the option (control = ON) to the N2HET instruction. The automatic read clear feature works for both 32 and 64 bit reads that follow the sequence described in Section 23.2.4.2.

When the host CPU reads the data field of that instruction, the current data value is returned to the host CPU but the field is cleared automatically as a side effect of the read. In case the master reads data from an instruction currently executing, any new capture result is stored and this takes priority over the automatic read clear feature, so that the new capture result is not lost.

As an example of where the automatic read clear feature is useful, consider the PCNT instruction. If this instruction is configured for automatic read clear, then when the host CPU reads the PCNT data field it will be cleared automatically. The host CPU can then poll the PCNT data field again, and as long as the field returns a value of zero the host CPU program knows a new capture event has not occurred. If the data field were not cleared, it would be impossible for the host CPU to determine whether the data field holds data from the previous capture event, or if it happens to be data from a new capture event with the same value.

23.2.4.4 Emulation Mode

Emulation mode, used by the software debugger, is specified in the global configuration register. When the host CPU debugger hits a breakpoint, the CPU sends a suspend signal to the modules. Two modes of operation are provided: suspend and ignore suspend.

- Suspend

When a suspend is issued, the timer operation stops at the end of the current timer instruction. However, the CPU accesses to the timer RAM or control registers are freely executed.

- Ignore suspend

The timer RAM ignores the suspend signal and operates real time as normal.

23.2.4.5 Power-Down

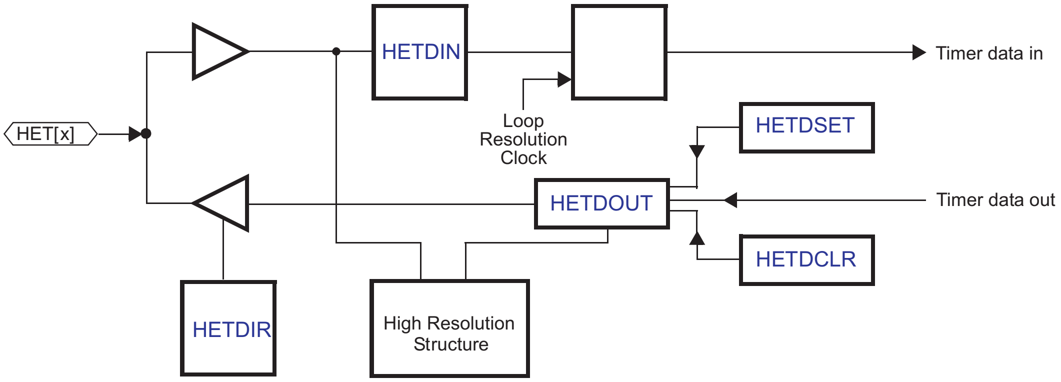
After setting the turn-off bit in the Global Configuration Register (HETGCR), it is required to delay until the end of the timer program loop before putting the N2HET in power-down mode. This can be done by waiting until the N2HET Current Address (HETADDR) becomes zero, before disabling the N2HET clock source in the device’s Global Clock Module (GCM).

23.2.5 I/O Control

The N2HET has up to 32 pins. Refer to device specific data sheets for information concerning the number of N2HETIO available. All of the N2HET pins available are programmable as either inputs or outputs.

These 32 I/Os have an identical structure connected to pins HET[31] to HET[0]. See Figure 23-8 for an illustration of the I/O control. In addition all 32 I/Os have a special HR structure based on the HR clock. This structure allows any N2HET instruction to use any of these I/Os with an accuracy of either loop resolution or high resolution accuracy.

Figure 23-8. I/O Control



Pins N2HET [31] to N2HET [0] can be used by the CPU as general-purpose inputs or outputs using the N2HET Data Input Register (HETDIN) for reading and N2HET Data Output Register (HETDOUT), N2HET Data Set Register (HETDSET) or N2HET Data Clear Register (HETDCLR) for writing, depending on the type of action to perform. The N2HET pins used as general-purpose inputs are sampled on each VCLK2 period.

23.2.5.1 Using General-Purpose I/O Data Set and Clear Registers

The N2HET Data Clear Register (HETDCLR) and N2HET Data Set Register (HETDSET) can be used to minimize the number of accesses to the peripheral to modify the output register and output pins. When the application needs to set or to reset some N2HET pins without changing the value of the others pins, the first possibility is to read N2HET Data Output Register (HETDOUT), modify the content (AND, OR, and so on), and write the result into N2HET Data Output Register (HETDOUT). However, this read-modify-write sequence could be interrupted by a different function modifying the same register which will result in a data coherency problem.

Using the N2HET Data Set Register (HETDSET) or N2HET Data Clear Register (HETDCLR), the application program must write the mask value (same mask value for the first option) to the register to set or reset the desired pins. Any bits written as 0 to HETDSET and HETDCLR are left unchanged, which avoids the possible coherency problem of the read-modify-write approach.

```
Coding Example (C program): Set pins using the 2 methods.

unsigned int MASK;                          /* Variable that content the bit mask  */
volatile unsigned int *HETDOUT,*HETDSET;    /* Pointer to HET registers            */
...
*HETDOUT = *HETDOUT | MASK;                 /* Read-modify-write of HETDOUT        */
*HETDSET = MASK;                            /* Set the pin without reading HETDOUT */
```

23.2.5.2 Loop Resolution Structure

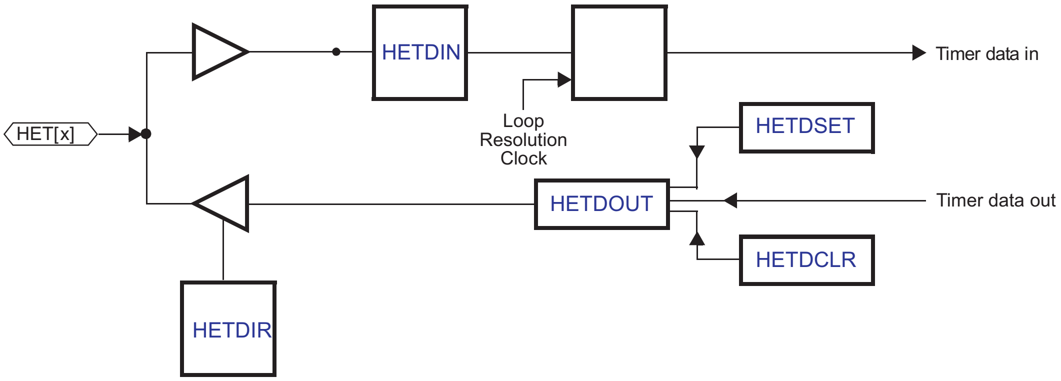
The N2HET uses the pins N2HET [31:0] as input and/or output by the way of the instruction set. Actually, each pin could monitor the N2HET program or could be monitored by the N2HET program. By using the I/O register of the N2HET, the CPU is able to interact with the N2HET program flow.

When an action (set or reset) is taken on a pin by the N2HET program, the N2HET will modify the pin at the rising edge of the next loop resolution clock.

When an event occurs on a N2HET I/O pin, it is taken into account at the next rising edge of the loop resolution clock.

The structure of each pin is shown in Figure 23-9.

Figure 23-9. N2HET Loop Resolution Structure for Each Bit

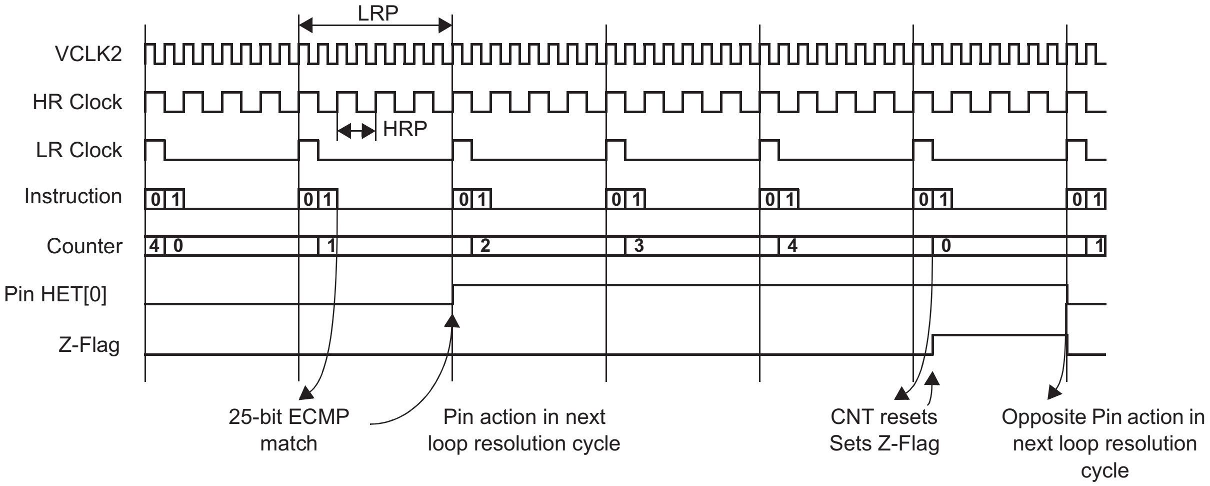


The example in Figure 23-10 shows a simple PWM generation with loop resolution accuracy. The corresponding program is:

```
HETPFR[31:0] register = 0x201 --> lr=4 and hr=2 --> ts = 8

N2HET Program:

L00  CNT   { next= L01, reg=A, irq=OFF, max = 4 }
L01  ECMP  { next= L00, cond_addr= L00, hr_lr=LOW, en_pin_action=ON, pin=0,
             action=PULSEHI, reg=A, irq=OFF, data= 1, hr_data = 0x0 }

; 25 bit compare value is 1 and the 7-bit HR compare value is 0
```

The CNT and ECMP instructions are executed once each loop resolution cycle. When the CNT instruction is executed, the specified register (A) and the CNT instruction data field are both incremented by one. Next the ECMP is executed and the data field of the ECMP is compared with the specified register (A). If both values match, then the pin action (PULSEHI in this case) will be performed in the next loop resolution cycle. The CNT continues incrementing each loop resolution cycle. When the data field overflows (max + 1), then the Z-flag is set by the CNT instruction. In the next loop resolution cycle, the Z-flag is evaluated and the opposite pin action is performed if it is set. The Z-flag will only be active for one loop resolution cycle.

Figure 23-10. Loop Resolution Instruction Execution Example



23.2.5.3 High Resolution Structure

All 32 I/Os provide the HR structure based on the HR clock. The HR clock frequency is programmed through the Prescale Factor Register (HETPFR). In addition to the standard I/O structure, all pins have HR hardware so that these pins can be used as HR input captures (using the HR instructions PCNT or WCAP) or HR output compares (using the HR instructions ECMP, MCMP, or PWCNT).

All five HR instructions (PCNT, WCAP, ECMP, MCMP, and PWCNT) have a dedicated hr_lr bit (high resolution/low resolution; program field bit 8) allowing operation either in HR mode or in standard resolution mode by ignoring the HR field. By default, the hr_lr bit value is 0 which implies HR operation mode. However, setting this bit to one allows the use of several HR instructions on a single HR pin. Only one instruction is allowed to operate in HR mode (bit cleared to 0), but the other instructions can be used in standard resolution mode (bit set to 1).

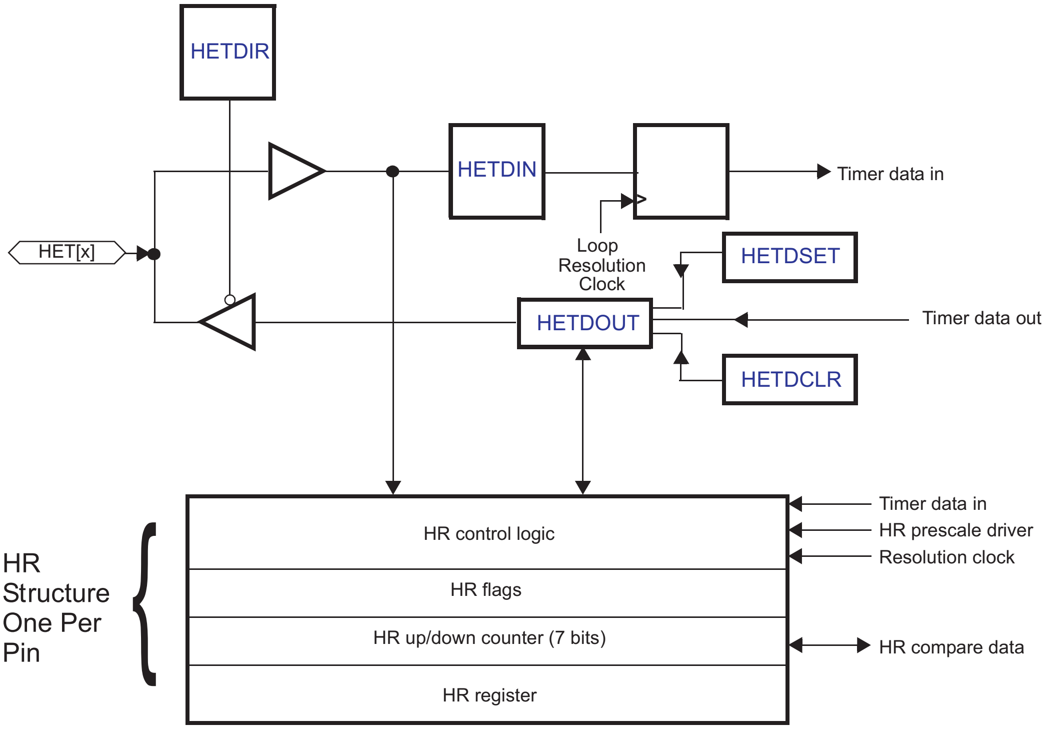
23.2.5.4 HR Block Diagram

Each time an HR instruction is executed on a given pin, the HR structure for that pin is programmed and synchronized to the next loop-resolution cycle (which HR function to perform and on which edges it should take an action) with the information given by the instruction. The HR structure for each pin decodes the pin select field of the instruction and programs its HR structure if it matches.

NOTE: For each N2HET pin, only one instruction specifying a high resolution operation (hr_lr = HIGH) is allowed to execute per loop resolution period. This includes any instructions where (hr_lr = HIGH) but (en_pin_action = OFF).

The first high resolution instruction that executes and specifies a particular pin locks out subsequent high resolution instructions from operating on the same pin until the end fo the current loop resolution period.

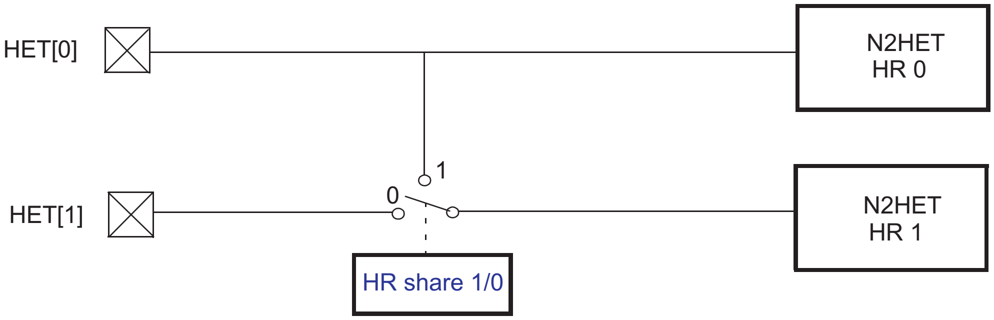
Figure 23-11. HR I/O Architecture



23.2.5.5 HR Structures Sharing (Input)

The HR Share Control Register (HETHRSH) allows two HR structures to share the same pin **for input capture only**. If these bits are set, the HR structures N and N+1 are connected to pin N. In this structure, pin N+1 remains available for general-purpose input/output. See Figure 23-12.

Figure 23-12. Example of HR Structure Sharing for N2HET Pins 0/1



The following program gives an example how the HR share feature (HET[0] HR structure and HET[1] HR structure shared) can be used for the PCNT instruction:

```
L00 PCNT { next=L01, type=rise2fall, pin=0 }
L01 PCNT { next=L00, type=fall2rise, pin=1 }
```

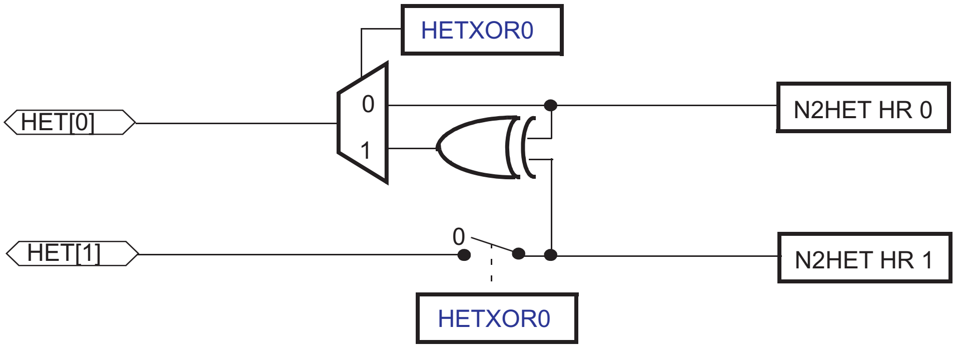
The HET[1] HR structure is also connected to the HET[0] pin. The L00_PCNT data field is able to capture a high pulse and the L01_PCNT captures a low pulse on the **same** pin (N2HET [0] pin).

23.2.5.6 AND / XOR-shared HR Structure (Output)

Usually the N2HET design allows only one HR structure to generate HR edges on a pin configured as output pin. The HETXOR register allows a logical XOR of the output signals of two consecutive HR structures N (even) and N+1 (odd). See Figure 23-13. In this way, it is possible to generate pulses smaller than the loop resolution clock since both edges can be generated by two independent HR structures. This is especially required for symmetrical PWM. See Figure 23-14.

The hardware provides a XOR gate that is connected to the outputs of the HR structure of two consecutive pins. In this structure, pin N+1 remains available for general-purpose input/output.

Figure 23-13. XOR-shared HR I/O



The following N2HET program gives an example for **one** channel of the symmetrical PWM. The generated timing is given in Figure 23-14.

```
MAXC .equ 22
A_   .equ 0  ; HR structure HR0
B_   .equ 1  ; HR structure HR1

CN CNT   { next=EA, reg=A, max=MAXC }

EA ECMP  { next=EB, cond_addr=MA, hr_lr=HIGH, en_pin_action=ON, pin=A_,
          action=PULSELO, reg=A, data=17, hr_data=115 }

MA MOV32 { next=EB, remote=EA, type=IMTOREG&REM, reg=NONE, data=17, hr_data=19 }

EB ECMP  { next=CN, cond_addr=MB, hr_lr=HIGH, en_pin_action=ON, pin=B_,
          action=PULSELO, reg=A, data=5, hr_data=13 }

MB MOV32 { next=CN, remote=EB, type=IMTOREG&REM, reg=NONE, data=5, hr_data=13 }
```

N2HET Settings and output signal calculation for this example program:

- Pin HET[0] and HET[1] are XOR-shared.
- HETPFR[31:0] register = 0x700: lr=128, hr=1, time slots ts = 128
- PWM period (determined by CNT_max field) = (22+1) · LRP = 2944 HRP
- Length of high pulse of (HET[0] XOR HET[1]) =

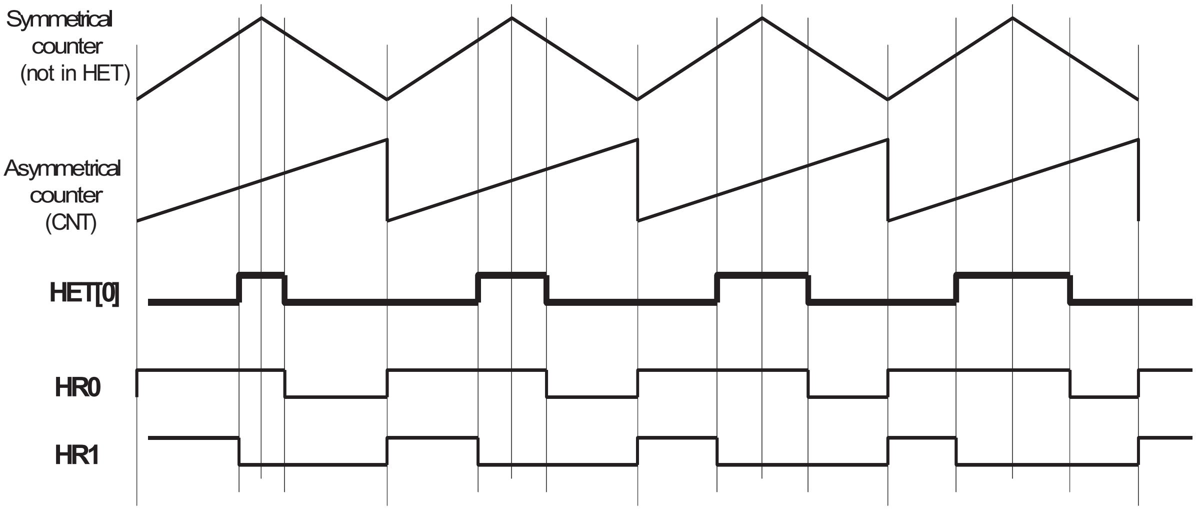
  LH = (17·LRP+115·HRP) - (5·LRP+13·HRP)

  With lr=128 there is LRP = 128 · HRP, so

  LH = (2291 - 653) · HRP = 1638 HRP
- Duty cycle = DC = LH / PWM_period = 1638 HRP / (2944·HRP) = 55.6 %

Figure 23-14 graphically shows the implementation of the XOR-shared feature. The first 2 waveforms (symmetrical counter and CNT) show a symmetric counter and asymmetric counter. The symmetric counter is shown only to highlight the axis of symmetry and is not implemented in the N2HET. The asymmetric counter, which is implemented with a CNT instruction, needs to be set to the period of the symmetric counter. The next two waveforms (HR [0] and HR [1]) show the output of the HR structures, which are the inputs for the XOR gate to create the PWM output on pin HET[0]. Notice that the pulses of signal HET[0] are centered about the axis of symmetry.

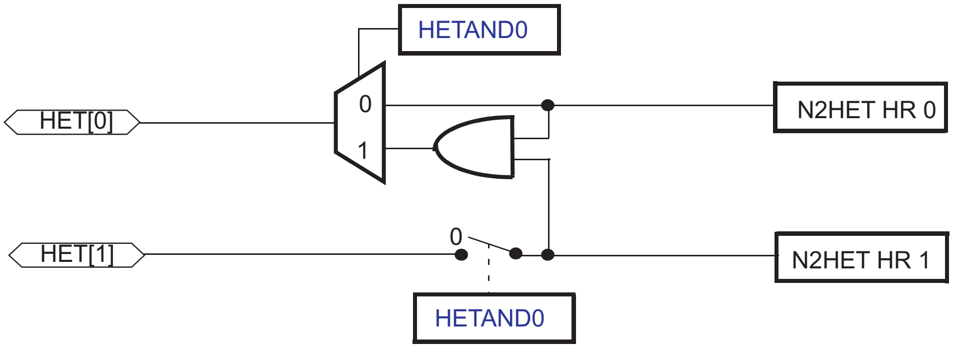
Figure 23-14. Symmetrical PWM with XOR-sharing Output



As an alternative, HR structures may be shared using a logical AND function to combine the effects of the pin structures. The HETAND allows sharing two consecutive HR structures N (even) and N+1 (odd). See Figure 23-15. In this structure, pin N+1 remains available for general-purpose input/output.

NOTE: Setting both the HETAND bit and HETXOR bits at the same time for a given pair of N2HET pins is not supported, must be avoided by the application program.

Figure 23-15. AND-shared HR I/O



23.2.5.7 Loop Back Mode

The loop back feature can be used by the application to monitor an N2HET output signal. For example, if a PWM is generated by HR structure 0, then a PCNT instruction assigned to HR structure 1 can measure back the pulse length or periods of the PWM output signal.

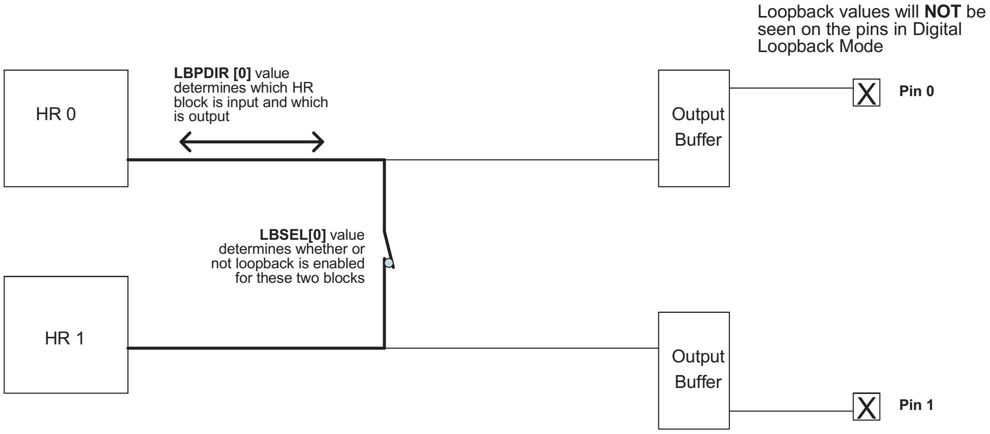
Loopback mode is activated between two high resolution structures by setting LBPSEL[x] to 1 in the HETLBPSEL register for the corresponding structure pair. The **direction** of the loopback between the two structures in the structure pair is determined by the value of LBPDIR[x] in the HETLBPDIR Register.

For example, if bit LBPSEL[0] is set to 1, then HR structures 0 and 1 will be internally connected in loop back mode. If bit LBPDIR[0] is set to 0, then structure 0 will be the input and structure 1 will be the output.

Digital Loopback

Digital loopback mode is enabled by setting LBPTYPE[x] to 0 in the HETLBPSEL register for the corresponding structure pairs. In digital loopback mode, the structure pairs are connected directly and the output buffers are bypassed. Therefore, the loopback values will NOT be seen on the corresponding pins. Figure 23-16 shows an example of digital loopback between structures HR0 and HR1. LBSEL[0] has been set to 1 to enable loopback between the two structures. LBTYPE[0] has been set to 0 to select digital mode for the loopback pair. The LPBDIR[0] value will determine the direction of the loopback by selecting which of the HR blocks is output, and which is input. The bold lines show the digital loopback path.

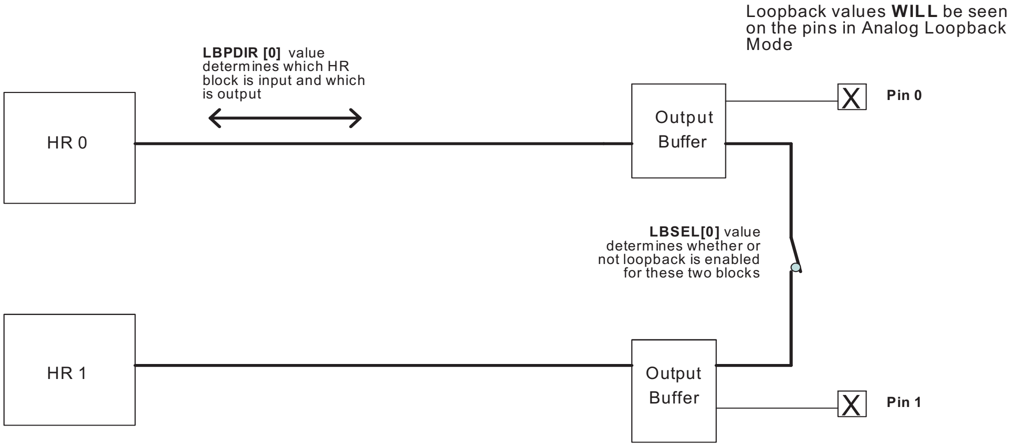
Figure 23-16. HR0 to HR1 Digital Loopback Logic: LBTYPE[0] = 0


Loopback values will **NOT** be seen on the pins in Digital Loopback Mode
HR 0
LBPDIR [0] value determines which HR block is input and which is output
Output Buffer
Pin 0
LBSEL[0] value determines whether or not loopback is enabled for these two blocks
HR 1
Output Buffer
Pin 1

Analog Loopback

Analog loopback mode is enabled by setting LBPTYPE[x] to 1 in the HETLBPSEL register for the corresponding structure pairs. In analog loopback mode, the structure pairs are connected outside of the output buffers. Therefore, the loopback values WILL be seen on the corresponding pins. Figure 23-17 shows an example of analog loopback between structures HR0 and HR1. LBSEL[0] has been set to 1 to enable loopback between the two structures. LBTYPE[0] has been set to 1 to select analog mode for the loopback pair. The LPBDIR[0] value will determine the direction of the loopback by selecting which of the HR blocks is output, and which is input. The bold lines show the analog loopback path.

Figure 23-17. HR0 to HR1 Analog Loop Back Logic: LBTYPE[0] = 1


Loopback values **WILL** be seen on the pins in Analog Loopback Mode
LBPDIR [0] value determines which HR block is input and which is output
HR 0
Output Buffer
Pin 0
LBSEL[0] value determines whether or not loopback is enabled for these two blocks
HR 1
Output Buffer
Pin 1

Note:

- The loop back direction can be selected independent of the HETDIR register setting.
- The pin that is not driven by the N2HET output pin actions can still be used as normal GIO pin.

23.2.5.8 Edge Detection Input Timing

There are several timing requirements for input signals in order to be captured correctly by N2HET. Figure 23-18 illustrates these requirements, with min and max values described in Table 23-7 (Loop Resolution) and Table 23-8 (High Resolution).

Figure 23-18. N2HET Input Edge Detection

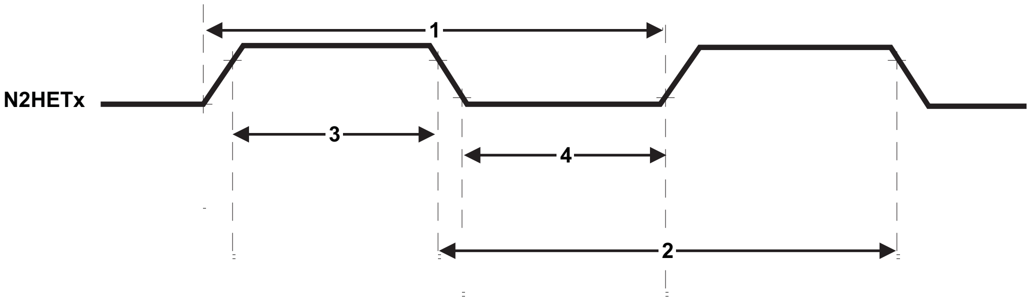


Table 23-7. Edge Detection Input Timing for Loop Resolution Instructions

Parameter #	Description	min	max
1	Input Signal Period, rising edge to rising edge	> 2 (hr) (lr) $t_{c(VCLK2)}$	< 2^{25} (hr) (lr) $t_{c(VCLK2)}$
2	Input Signal Period, falling edge to falling edge		
3	Input Signal, high phase	> (hr) (lr) $t_{c(VCLK2)}$	
4	Input Signal, high phase		

Table 23-8. Edge Detection Input Timing for High Resolution Instructions

Parameter #	Description	min	max
1	Input Signal Period, rising edge to rising edge	> (hr) (lr) $t_{c(VCLK2)}$	< 2^{25} (hr) (lr) $t_{c(VCLK2)}$
2	Input Signal Period, falling edge to falling edge		
3	Input Signal, high phase	> 2 (hr) $t_{c(VCLK2)}$	
4	Input Signal, high phase		

These are the N2HET architectural limitations. Actual limitations will be slightly different due to on chip routing and IO buffer delays, usually by several nanoseconds. Be sure to consult the device datasheet for actual timings that apply to that device. Also, certain devices place additional restrictions on which pins support the high resolution timings of Table 23-8, if present these additional limitations will also be called out in the device datasheet.

Note that the max limit in Table 23-7 and Table 23-8 is based on the counter range of a single N2HET instruction. The max value could be extended by employing an additional N2HET instruction to keep track of counter overflows of the input counter / capture instruction.

23.2.5.9 PWM Generation Example 1 (in HR Mode)

The following example shows how an ECMP instruction works in high resolution mode. The example assumes a VCLK2 of 32 MHz and the following values for the prescale divide rates (hr and lr), number of time slots (ts), high and loop resolution period (HRP and LRP):

hr = 2, lr = 4, ts = hr × lr = 8

HRP = hr / VCLK2 = 2 / 32 MHz = 62.5 ns

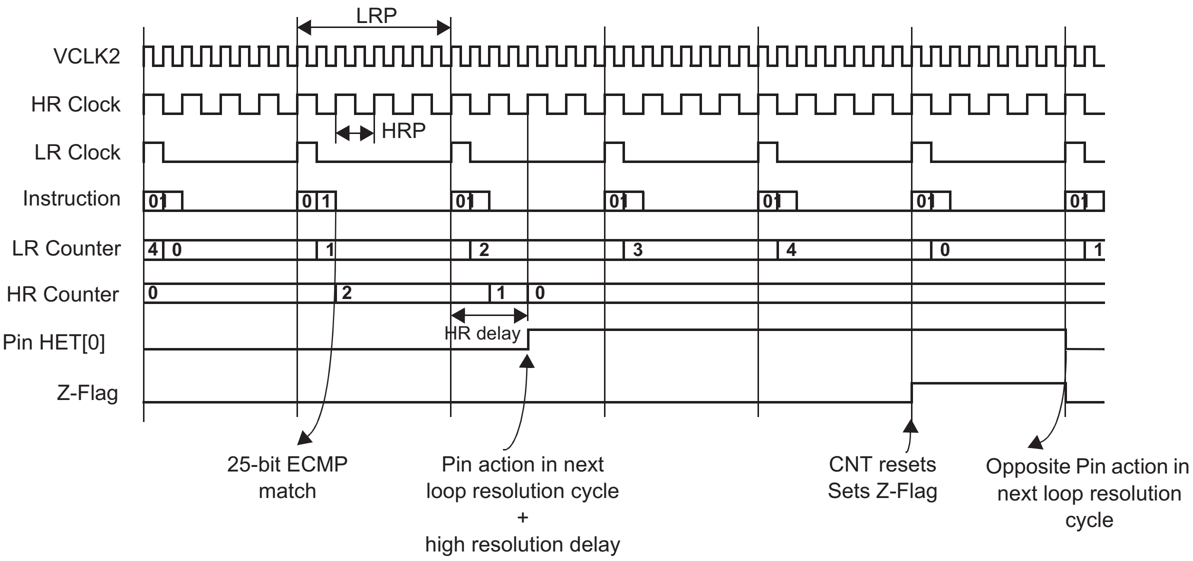
LRP = (hr × lr) / VCLK2 = 8 / 32 MHz = 250 ns

With ts = 8, there are eight time slots available for the program execution, which in this case will consist of one CNT and one ECMP instruction as shown below. The data field of the ECMP instruction is the 32-bit compare value, whereby the lower 7 bits represent the high resolution compare field.

When the 25-bit (loop resolution) compare matches, the HR compare value will be loaded from the 7 lower bits of the instruction data field to the HR counter. At the next loop resolution clock, the HR counter will count down at the HR clock frequency and perform the pin action when it reaches zero.

In the example illustrated by Figure 23-19, the 25-bit compare value is 1 and the 7-bit HR compare value is 2. According to Section 23.2.3.2, depending on the loop resolution divide rate (lr), only certain bits of the 7-bit HR compare value are valid. In this example only the upper 2 bits (D[6:5]) are taken into account. The example program below has a setting of hr_data = 100000b. Shifting this value right by 5 bits, results in 10b which equals the two HR clock cycles delay mentioned above.

Figure 23-19. ECMP Execution Timings



```
HETPFR[31:0] register = 0x201 --> lr=4 and hr=2 --> ts = 8
```

N2HET Program:

```
L00   CNT   { next= L01, reg=A, irq=OFF, max = 4 }
L01   ECMP  { next= L00, cond_addr= L00, hr_lr=HIGH, en_pin_action=ON, pin=0,
              action=PULSEHI, reg=A, irq=OFF, data= 1, hr_data = 0x40 }

; 25 bit compare value is 1 and the 7-bit HR compare value is 2
; (Because of lr=4 the D[4:0] of the 7-bit HR field are ignored )
```

NOTE: ECMP Opposite Actions

ECMP opposite pin actions are always synchronized to the loop resolution clock.

Changing the duty cycle of a PWM generated by an ECMP instruction, can lead to a missing pulse if the data field of the instruction is updated directly. This can happen when it is changed from a high value to a lower value while the CNT instruction has already passed the new updated lower value. To avoid this a synchronous duty cycle update can be performed with the use of an additional instruction (MOV32). This instruction is only executed when the compare of the ECMP matches. For this the cond_addr of the ECMP needs to point to the MOV32. On execution of the MOV32, it moves its data field into the data field of the ECMP. The update of the duty cycle has to be made to the MOV32 data field instead of the ECMP data field.

23.2.5.10 PWM Generation Example 2 (in HR Mode)

The MCMP instruction can also be used in HR mode. In this case operation is exactly the same as for the ECMP instruction except that the 25-bit low resolution is now the result of a magnitude compare (greater or equal) rather than an equality compare. When the 25-bit (loop resolution) magnitude compare matches, the HR compare value will be loaded from the 7 lower bits of the instruction data field to the HR counter. At the next loop resolution clock, the HR counter will count down at the HR clock frequency and perform the pin action when it reaches zero.

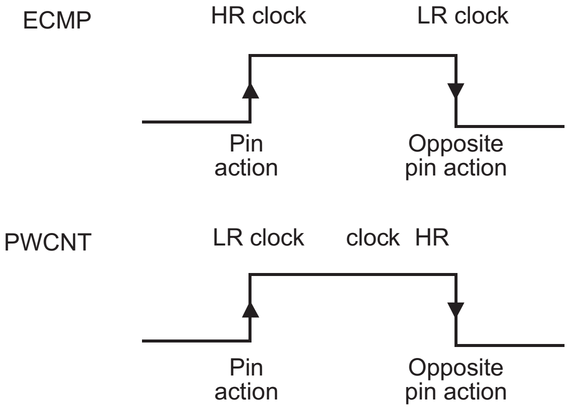
The MCMP instruction avoids the missing pulse problem of the ECMP instruction (see previous example), however the duty cycle of the signal might not be exact for one PWM period. The benefit of the MCMP is that it avoids adding another instruction to do the duty cycle update synchronously.

23.2.5.11 Pulse Generation Example (in HR Mode)

The PWCNT instruction may also be used in HR mode to generate pulse outputs with HR width. It generates a single pulse when the data field of the instruction is non-zero. It remains at the opposite pin action when the data field is zero.

The PWCNT instruction operates conversely to the ECMP instruction. See Figure 23-20. For PWCNT, the opposite pin action is synchronous with the HR clock and for ECMP the pin action is synchronous with the HR clock. The PWCNT pin action is synchronous with the loop resolution clock.

Figure 23-20. High/Low Resolution Modes for ECMP and PWCNT



23.2.5.12 Pulse Measurement Example (in HR Mode)

The PCNT instruction captures HR measurement of the high/low pulse time or periods of the input. As shown in Figure 23-21, at marker (1) the input goes HIGH and the HR counter immediately begins to count. The counter increments and rolls over until the falling edge on the input pin, where it captures the counter value into the HR capture register (marker (2)). The PCNT instruction begins counting when the synchronized input signal goes HIGH and captures both the 25-bit data field and the HR capture register into RAM when the synchronized input falls (marker (3)).

NOTE: The HR capture value written into RAM is shifted appropriately depending on the loop resolution prescale divide rate (lr). (See also Section 23.2.3.2).

Figure 23-21 shows what happens when the capture edge arrives *after* the HR counter overflows. This causes the incremented value to be captured by the PCNT instruction.

Figure 23-21. PCNT Instruction Timing (With Capture Edge After HR Counter Overflow)

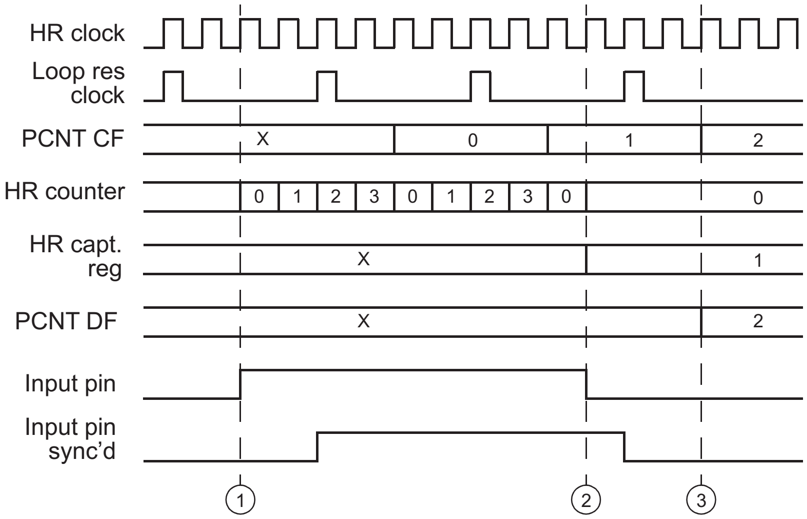
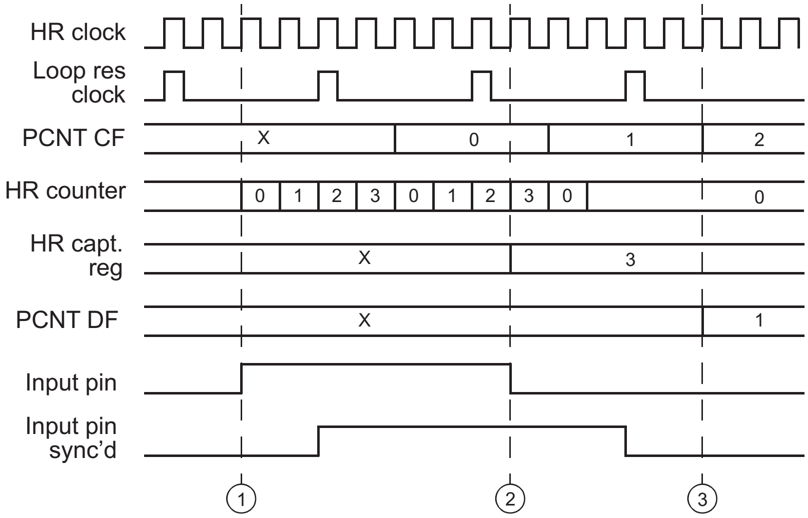


Figure 23-22 shows what happens when the capture edge arrives *before* the HR counter overflows. This causes the non-incremented value to be captured by the PCNT instruction.

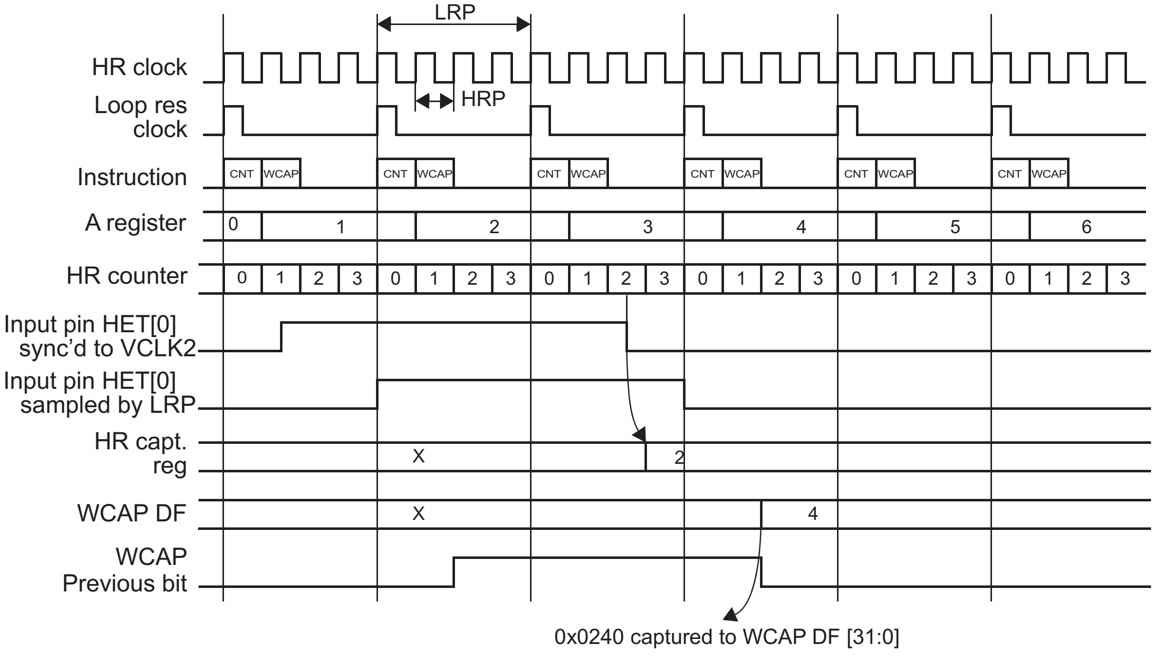
Figure 23-22. PCNT Instruction Timing (With Capture Edge Before HR Counter Overflow)



23.2.5.13 WCAP Execution Example (in HR Mode)

The HR capability is enabled for WCAP, if its hr_lr bit is zero. In this case the HR counter is always enabled and is synchronized with the resolution loop. When the specified edge is detected, the current value of the HR counter is captured in the HR capture register and written into the RAM after the next WCAP execution. The WCAP instruction effectively time stamps the free running timer saved in a register (for example, register A shown in Figure 23-23).

Figure 23-23. WCAP Instruction Timing

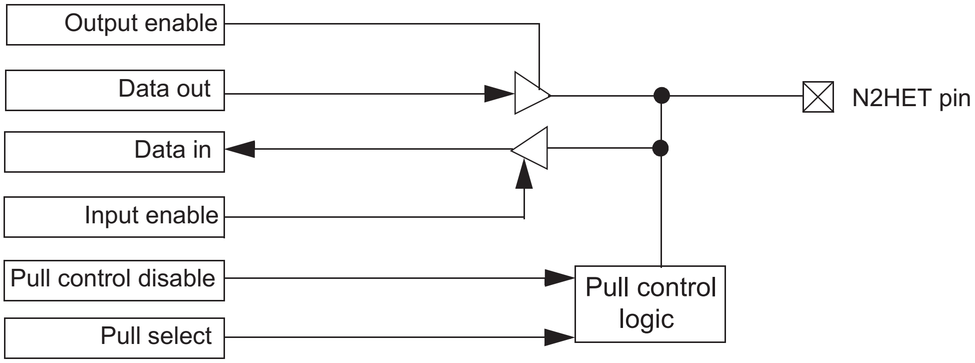


```
HETPFR_register = 0x0200 --> lr = 4, hr = 1, ts = 4

N2HET Program:

L00 CNT  {reg=A, max=01ffffffh}
L01 WCAP {next=L00, cond_addr=L00, hr_lr=high, reg=A, event= FALL, pin=0,
          data=0}
```

In the example, the WCAP is configured to capture the counter when a **falling** edge occurs. The WCAP data field (WCAP_DF) is updated in the loop succeeding the loop in which the edge occurred. The WCAP instruction evaluates an edge by comparing its Previous bit with the sync'd input signal. In Figure 23-23, the current value of the counter (4) is captured to WCAP_DF[31:7] and the value of the HR capture register (2) is transferred to the valid bits (according the lr prescaler) of WCAP_DF[6:0]. Therefore, in the example 0x0240 is captured in WCAP_DF[31:0].

23.2.5.14 I/O Pull Control Feature

Figure 23-24. I/O Block Diagram Including Pull Control Logic



The following apply if the device is under reset:

- Pull control: The reset pull control on the pins is enabled and a pulldown is configured.
- Input buffer: The input buffer is enabled.
- Output buffer: The output buffer is disabled.

The following apply if the device is out of reset:

- Pull control: The pull control is enabled by clearing the corresponding bit in the N2HET Pull Disable Register (HETPULDIS). In this case, if the corresponding bit in the N2HET Pull Select Register (HETPSL) is set, the pin will have a pull-up; if the bit in the N2HET Pull Select Register (HETPSL) is cleared, the pin will have a pull-down. If the bit in the N2HET Pull Disable Register (HETPULDIS) is set, there is no pull-up or pull-down on the pin.
- Input buffer: The input buffer is disabled only if the pin direction is set to input AND the pull control is disabled AND pull down is selected as the pull bias. In all other cases, the input buffer is enabled.

NOTE: The pull-disable logic depends on the pin direction. If the pin is configured as output, then the pulls are disabled automatically. If the pin is configured as input, the pulls are enabled or disabled depending on the pull disable register bit.

- Output buffer: A pin can be driven as an output pin if the corresponding bit in the N2HET Direction Register (HETDIR) is set AND the open-drain feature (N2HET Open Drain Register (HETPDR)) is not enabled. See Section 23.2.5.15 for more details.

The behavior of the input buffer, output buffer, and the pull control is summarized in Table 23-9. When an input buffer is disabled, it appears as a logic low to on-chip logic.

Table 23-9. Input Buffer, Output Buffer, and Pull Control Behavior

Device under Reset?	Pin Direction (DIR)[1]	Pull Disable (PULDIS)[1]	Pull Select (PULSEL)[1]	Pull Control	Output Buffer	Input Buffer
Yes	X	X	X	Enabled	Disabled	Enabled
No	0	0	0	Pull down	Disabled	Enabled
No	0	0	1	Pull up	Disabled	Enabled
No	0	1	0	Disabled	Disabled	Disabled
No	0	1	1	Disabled	Disabled	Enabled
No	1	X	X	Disabled	Enabled	Enabled

[1] X = Don't care

23.2.5.15 Open-Drain Feature

The following apply if the open-drain feature is enabled on a pin, that is, the corresponding bit in the N2HET Open Drain Register (HETPDR) is set:

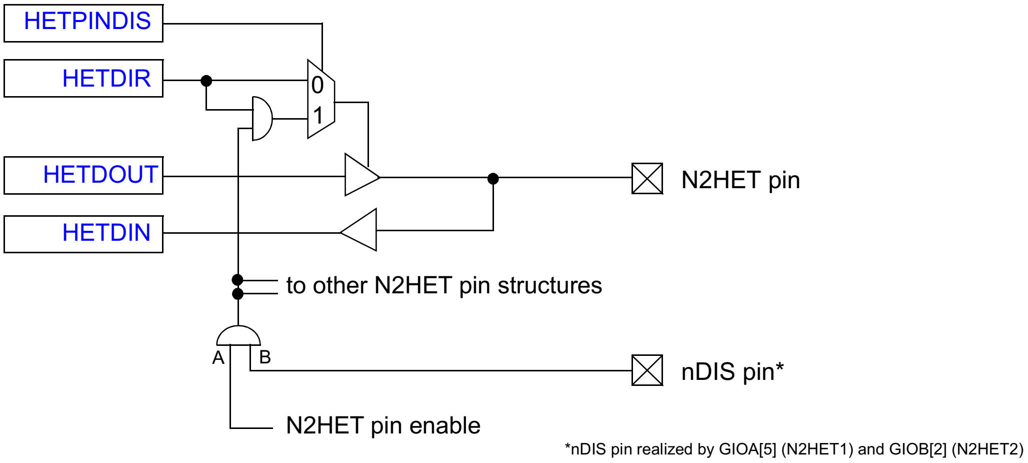
- Output buffer is enabled if a low signal is being driven internally to the pin.
- The output buffer is disabled if a high signal is being driven internally to the pin.

23.2.5.16 N2HET Pin Disable Feature

This feature is provided for the safe operation of systems such as power converters and motor drives. It can be used to inform the monitoring software of motor drive abnormalities such as over-voltage, over-current, and excessive temperature rise.

Table 23-10 shows the conditions for the output buffer to be enabled/disabled.

Figure 23-25. N2HET Pin Disable Feature Diagram



*nDIS pin realized by GIOA[5] (N2HET1) and GIOB[2] (N2HET2)

Table 23-10. N2HET Pin Disable Feature

HETPINDIS.x	nDIS Pin (Input)	HET_PIN_ENA (HETGCR.24)	HETDIR.x	Output Buffer
0	X	X	0	Disabled
0	X	X	1	Enabled
1	0	X	0	Disabled
1	0	X	1	Disabled
1	1	X	0	Disabled
1	1	0	1	Disabled
1	1	1	1	Enabled

An interrupt capable device I/O pin can share the same pin as the N2HET nDIS signal. Normally GIOA[5] serves as nDIS for N2HET1 and GIOB[2] as nDIS for N2HET2. Check the device datasheet for the actual implementation. Sharing a pin with a GIO pin that is Interrupt capable allows the N2HET nDIS input to also generate an interrupt to the CPU. An active low level on nDIS is intended to signal an abnormal situation as described above. All N2HET pins, which are selected with the N2HET Pin Disable Register (HETPINDIS), will be put in the high-impedance state by hardware immediately after the nDIS signal is pulled low. At this time a CPU interrupt is issued, if it is enabled in the GIO pin logic.

The bit HET_PIN_ENA is automatically cleared in the failure condition and this state remains as long as the software explicitly sets the bit again. The steps to do this are:

- Software detects, by reading the HETDIN register of the GIO pin, that the level on nDIS is inactive (high).
- Software sets bit HET_PIN_ENA to deactivate the high impedance state of the pins.

23.2.6 Suppression Filters

Each N2HET pin is equipped with a suppression filter. If the pin is configured as an input it enables to filter out pulses shorter than a programmable duration. Each filter consists of a 10-bit down counter, which starts counting at a programmable preloaded value and is decremented using the VCLK2 clock.

- The counter starts counting when the filter input signal has the opposite state of the filter output signal. The output signal is preset to the same input signal state after reset, in order to ensure proper operation after device reset.
- Once the counter reaches zero without detecting an opposite pin state on the filter input signal, the output signal is set to the opposite state.
- When the counter detects an opposite pin action on the filter input signal before reaching zero, the counter is loaded with it's preload value and the opposite pin action on the filter output signal does not take place. The counter resumes at the preload value until it detects an opposite pin action on the input signal again.
- Therefore the filter output signal is delayed compared to the filter input signal. The amount of delay depends on the counter clock frequency (VCLK2) and the programmed preload value.
- The accuracy of the output signal is +/- the counter clock frequency.

Table 23-11 gives examples for a 100 MHz VCLK2 frequency.

Figure 23-26. Suppression Filter Counter Operation

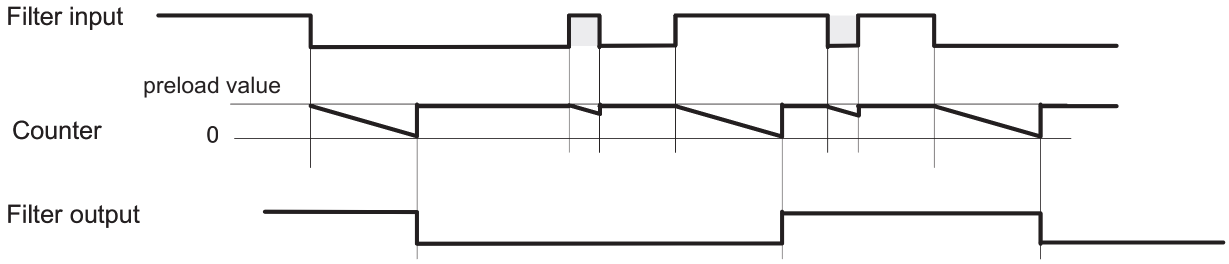


Table 23-11. Pulse Length Examples for Suppression Filter

Divider CCDIV	VCLK2	Possible values for the suppressed pulse length / frequency resulting from the programmable 10 bit preload value (0,1,..,1023)	
1	100.0 MHz	10 ns, 20 ns, …, 10.22 µs, 10.23 µs	50 MHz, 25 MHz, …, 48.924 kHz, 48.876 kHz
2	50.0 MHz	20 ns, 40 ns, …, 20.44 µs, 20.48 µs	25 MHz, 12.5 MHz, …, 24.462 kHz, 24.414 kHz
3	33.3 MHz	30 ns, 60 ns, …, 30.66 µs, 30.69 µs	16.7 MHz, 8.3 MHz, …, 16.308 kHz, 16.292 kHz

23.2.7 Interrupts and Exceptions

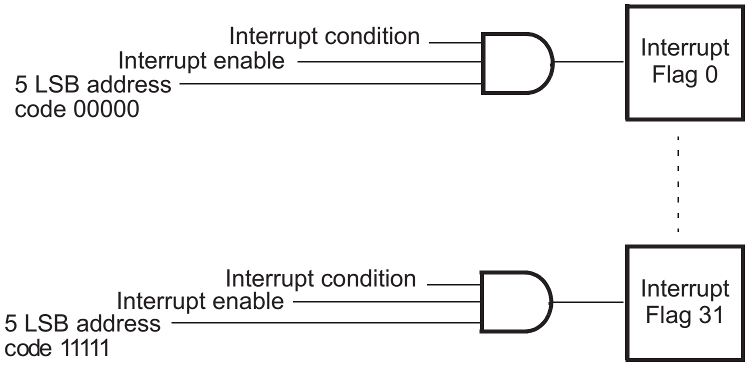
N2HET interrupts can be generated by any instruction that has an interrupt enable bit in its instruction format. When the interrupt condition in an instruction is true and the interrupt enable bit of that instruction is set, an interrupt flag is then set in the N2HET Interrupt Flag Register (HETFLG). The address code for this flag is determined by the five LSBs of the current timer program address. The flag in the N2HET Interrupt Flag Register (HETFLG) is set even if the corresponding bit in the N2HET Interrupt Enable Set Register (HETINTENAS) is 0. To generate an interrupt, the corresponding bit in the N2HET Interrupt Enable Set Register (HETINTENAS) must be 1. In the N2HET interrupt service routine, the main CPU must first determine which source inside the N2HET created the interrupt request. This operation is accelerated by the N2HET Offset Index Priority Level 1 Register (HETOFF1) or N2HET Offset Index Priority Level 2 Register (HETOFF2) that automatically provides the number of the highest priority source within each priority level. Reading the offset register will automatically clear the corresponding N2HET interrupt flag that created the request. However, if the offset registers are not used by the N2HET interrupt service routine, the flag should be cleared explicitly by the CPU once the interrupt has been serviced.

Table 23-12. Interrupt Sources and Corresponding Offset Values in Registers HETOFFx

Source No.	Offset Value
no interrupt	0
Instruction 0, 32, 64...	1
Instruction 1, 33, 65...	2
:	:
Instruction 31, 63, 95...	32
Program Overflow	33
APCNT underflow:	34
APCNT overflow	35

The instructions capable of generating interrupts are listed in Table 23-75.

Figure 23-27. Interrupt Functionality on Instruction Level



Each interrupt source is associated with a priority level (level 1 or level 2). When multiple interrupts with the same priority level occur during the same loop resolution the lowest flag bit is serviced first.

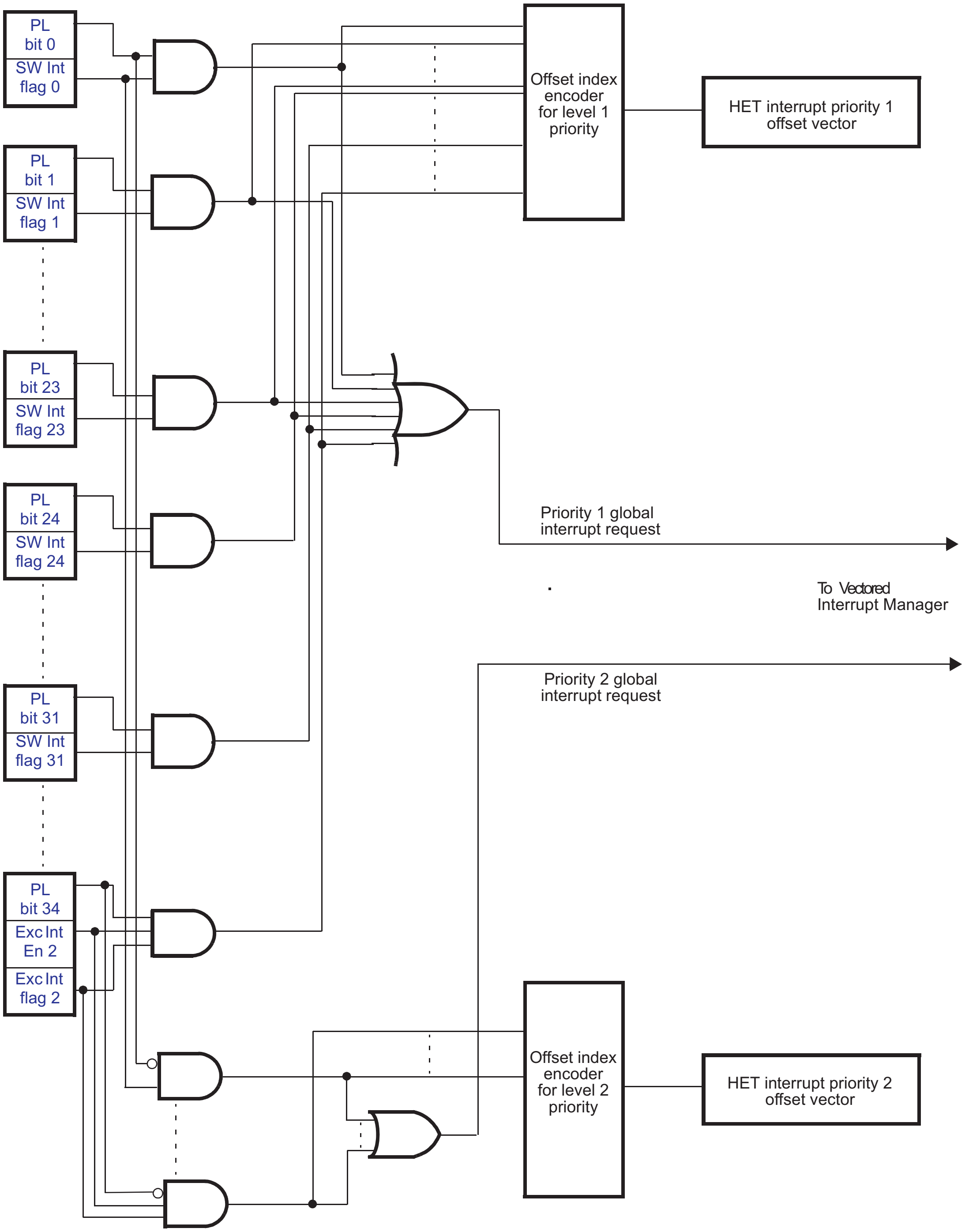
In addition to the interrupts generated by the instructions the N2HET can generate three additional exceptions:

- Program overflow
- APCNT underflow (see Section 23.3.1.2)
- APCNT overflow (see Section 23.3.1.3)

23.2.8 Hardware Priority Scheme

If two or more software interrupts are pending on the same priority level, the offset value will show the one with the highest priority. The interrupt with the highest priority is the one with the lower offset value. This scheme is hard-wired in the offset encoder. See Figure 23-28.

Figure 23-28. Interrupt Flag/Priority Level Architecture

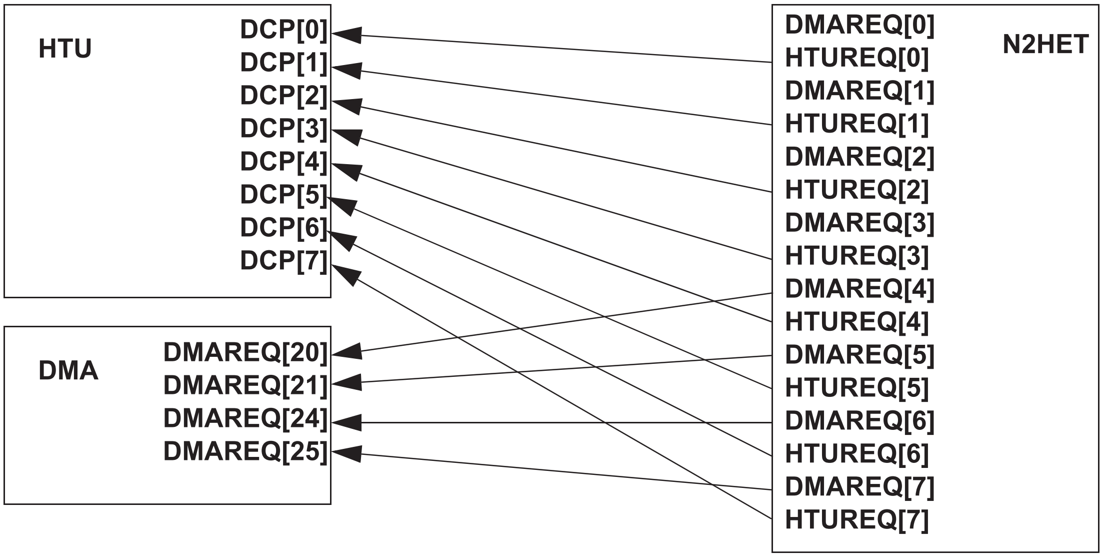

PL
bit 0
SW Int
flag 0
PL
bit 1
SW Int
flag 1
PL
bit 23
SW Int
flag 23
PL
bit 24
SW Int
flag 24
PL
bit 31
SW Int
flag 31
PL
bit 34
Exc Int
En 2
Exc Int
flag 2
Offset index encoder for level 1 priority
HET interrupt priority 1 offset vector
Priority 1 global interrupt request
To Vectored Interrupt Manager
Priority 2 global interrupt request
Offset index encoder for level 2 priority
HET interrupt priority 2 offset vector

23.2.9 N2HET Requests to DMA and HTU

As described in Section 23.6.3, the majority of the N2HET instructions are able to generate a transfer request to the High-End Timer Transfer Unit (HTU) and/or to the DMA module when an instruction-specific condition is true. One N2HET instruction can select one of 8 request lines by programming the "reqnum" parameter. The "request" field in an instruction is used to enable, disable, or to generate a quiet request (see Section 23.6.2) on the selected request line. Quiet requests can be used by the HTU, but not by the DMA. For quiet request, refer to the *High-End Timer Transfer Unit (HTU) Module* chapter (see Section 24.2.4.1).

The configuration of the N2HET Request Destination Select Register (HETREQDS) bits determines if a request line triggers an HTU-DCP, a DMA channel or both. This means the register bits will determine whether an N2HET instruction triggers DMAREQ[x], HTUREQ[x] or both signals (shown in Figure 23-29). The request line number x corresponds to the "reqnum" parameter used in the instruction.

Figure 23-29. Request Line Assignment Example



23.3 Angle Functions

Engine management systems require an angle-referenced time base to synchronize signals to the engine toothed wheel. The N2HET has a method to provide such a time base for low-end engine systems. The reference is created by the N2HET using three dedicated instructions with fractional angle steps equal to /8, /16, /32, /64.

23.3.1 Software Angle Generator

The N2HET provides three specialized count instructions to generate an angle referenced time base synchronized to an external reference signal (the toothed wheel signal) that defines angular reference points.

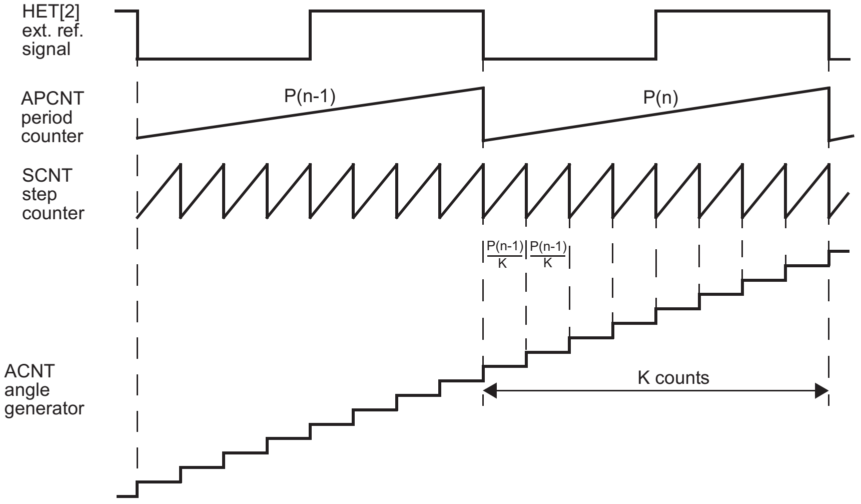
The time base is used to generate fractional angle steps between the reference points. The step width K (= 8, 16, 32, or 64) programmed by the user defines the angle accuracy of the time base. These fractional steps are then accumulated in an angle counter to form the absolute angle value.

The first counter, APCNT, incremented on each loop resolution clock measures the periods P(n) of the external signal. The second counter SCNT counts by step K up to the previous period value P(n-1), measured by APCNT, and then recycles. The resulting period of SCNT is the fraction P(n-1) / K. The third counter ACNT accumulates the fractions generated by SCNT.

Figure 23-30 illustrates the basic operation of APCNT, SCNT, and ACNT.

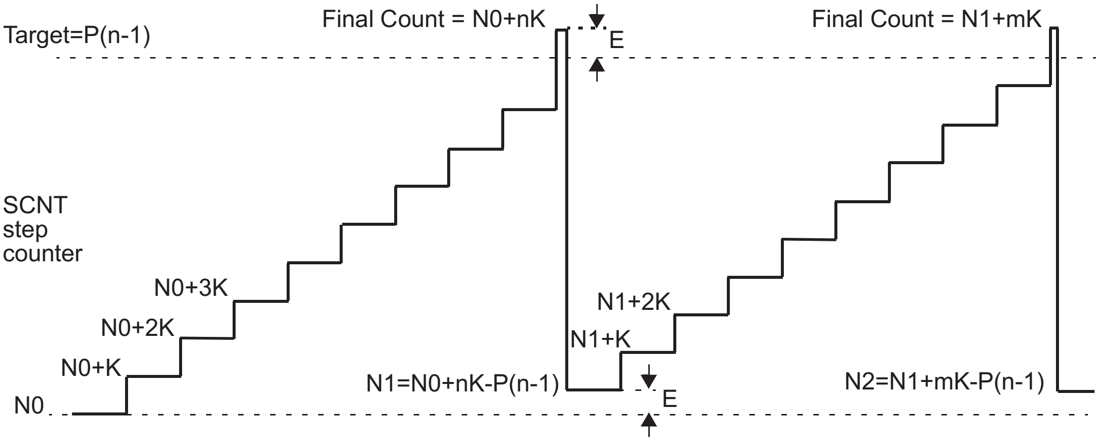
A N2HET timer program can only have one angle generator.

Figure 23-30. Operation of N2HET Count Instructions



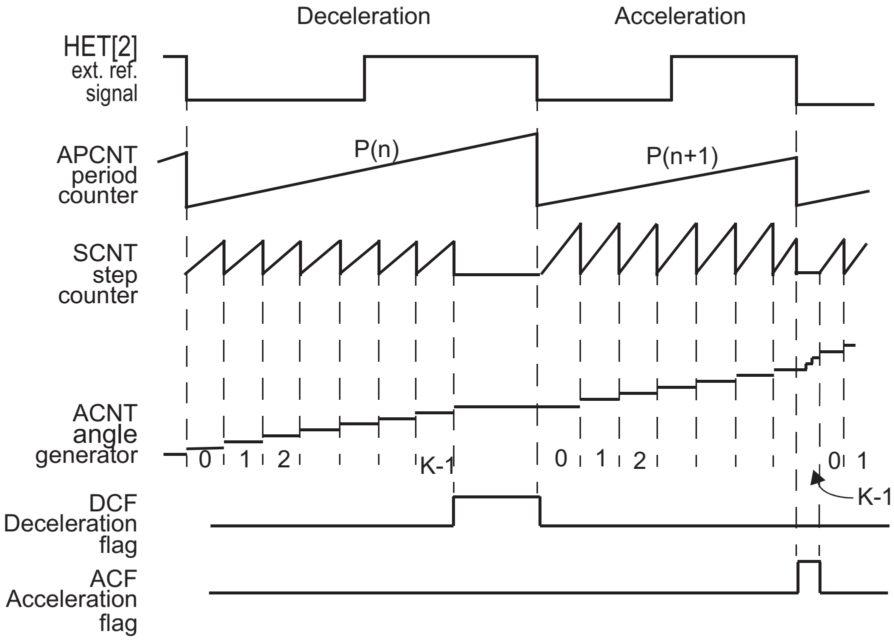
Due to stepping, the final count of SCNT does not usually exactly match the target value P(n-1). Figure 23-31 illustrates how SCNT compensates for this feature by starting each cycle with the remainder (final count - target) of the previous cycle.

Figure 23-31. SCNT Count Operation



ACNT detects period variations of the external signal measured by APCNT and compensates related counting errors. A period increase is flagged in the deceleration flag. A period decrease is flagged in the acceleration flag. If no variation is flagged, ACNT increments the counter value each time SCNT reaches its target. If acceleration is detected, ACNT increments the counter value on each timer resolution (fast mode). If deceleration is detected, ACNT is stopped. Figure 23-32 illustrates how the compensations for acceleration and deceleration operate.

Figure 23-32. ACNT Period Variation Compensations



23.3.1.1 Singularities

Singularities (gaps, in this case, from missing teeth in a toothed wheel) in the external reference signal can be masked. The start and end of singularities are defined by gap start and gap end values specified in SCNT and ACNT. When ACNT reaches gap start or gap end, it sets/resets the gap flag.

While the gap flag is set, new periods of the external reference signal are ignored for angle computation. SCNT uses the last period measured by APCNT just before gap start.

Figure 23-33 and Figure 23-34 illustrate the behavior of the angle generator during a gap after a deceleration or acceleration of the N2HET.

Figure 23-33. N2HET Timings Associated with the Gap Flag (ACNT Deceleration)

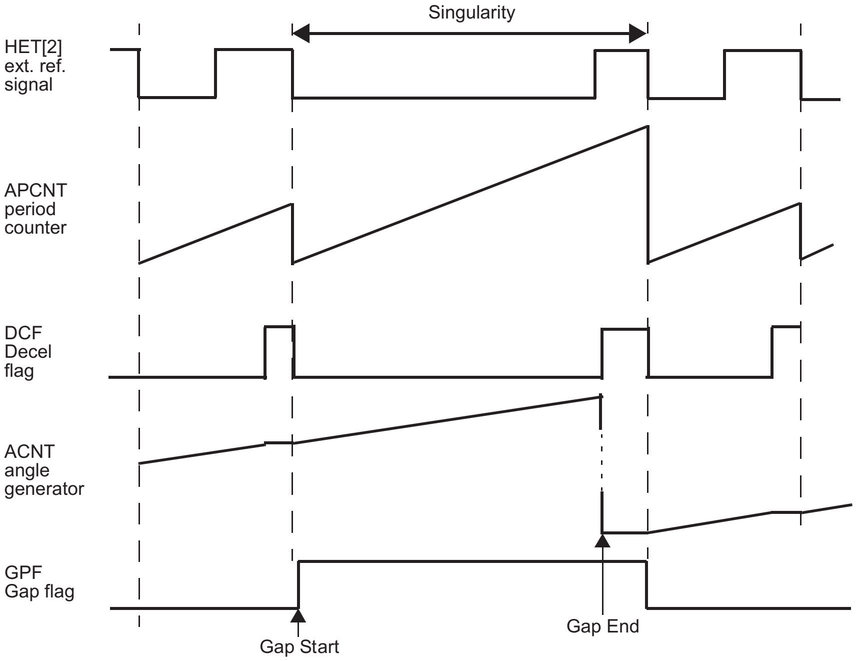
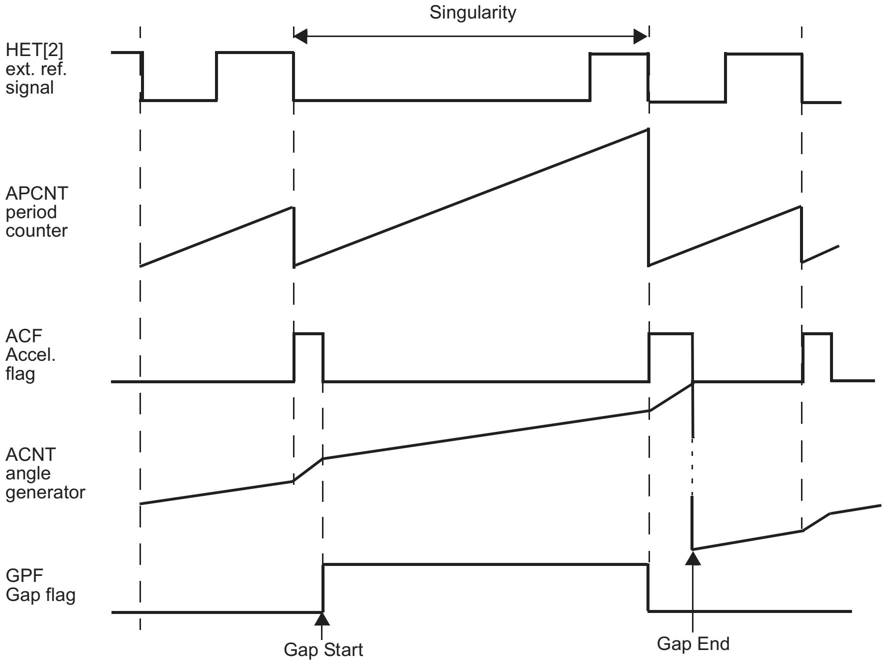


Figure 23-34. N2HET Timings Associated with the Gap Flag (ACNT Acceleration)



23.3.1.2 APCNT Underflow

The fastest valid external signal APCNT can accept must satisfy the following condition:

Step Width K < Period Min. Resolution (LRP)

This condition fixes the maximum possible step width once the minimum period and the resolution of an application are specified.

If a period value accidentally falls below the minimum allowed, APCNT stops the capture of these periods and sets the APCNT underflow interrupt flag located in the exceptions interrupt control register. In such a situation, SCNT and ACNT continue to be executed using the last valid period captured by APCNT.

23.3.1.3 APCNT Overflow

The slowest valid external signal APCNT can measure must satisfy the following condition:

Period Max Resolution < 33554431

When this limit is reached (APCNT Count equals all 1’s), APCNT stays at a maximum count (stops counting). APCNT remains in this position until the next specified capture edge is detected on the selected pin and sets the APCNT overflow interrupt flag located in the exceptions interrupt control register. In this situation, SCNT and ACNT continue to be executed using the maximum APCNT period count.

23.3.2 Hardware Angle Generator (HWAG)

23.3.2.1 Overview

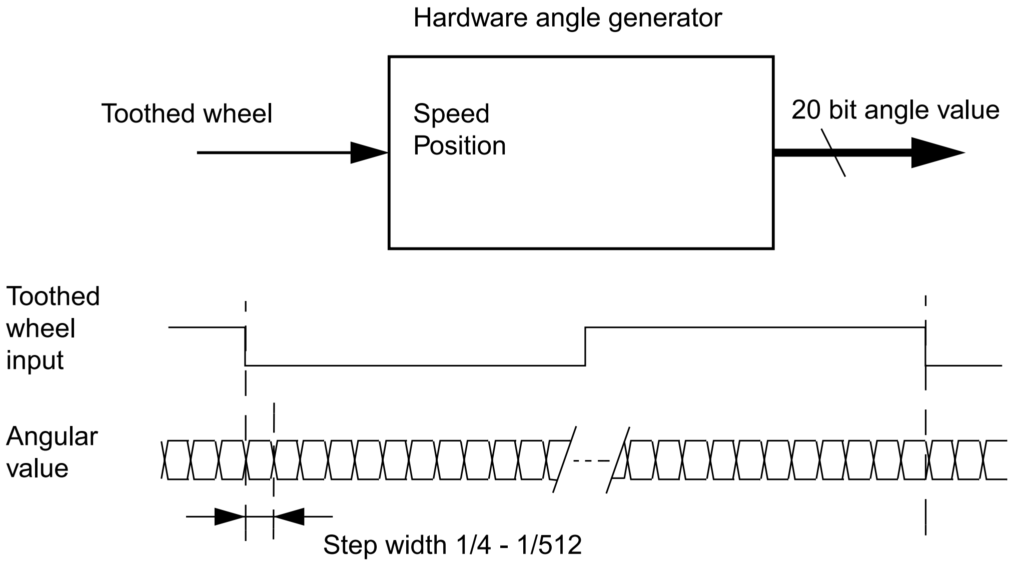
More engine control functions require powerful microcontrollers to process the timing. These controllers must generate signals such as dwell time, spark time, and fuel injection, at precise engine angles. These signals must be synchronized with the engine cycle.

The hardware angle generator (HWAG) generates angle value from toothed wheels. Because the toothed wheels are inaccurate (the most widely wheel used has 60 teeth with 6°/tooth), the period between two tooth edges (\) interpolates the angle value and the step width gives the number of interpolated angles. For an example of the angle generator principle, see Figure 23-35.

The HWAG can complement the high-end timer (NHET) to generate complex angle-angle or angle-time wave forms.

To work with the majority of toothed wheels, the HWAG provides registers to allow the CPU to configure step width, singularity, and filtering when initializing.

Figure 23-35. Angle Generator Principle



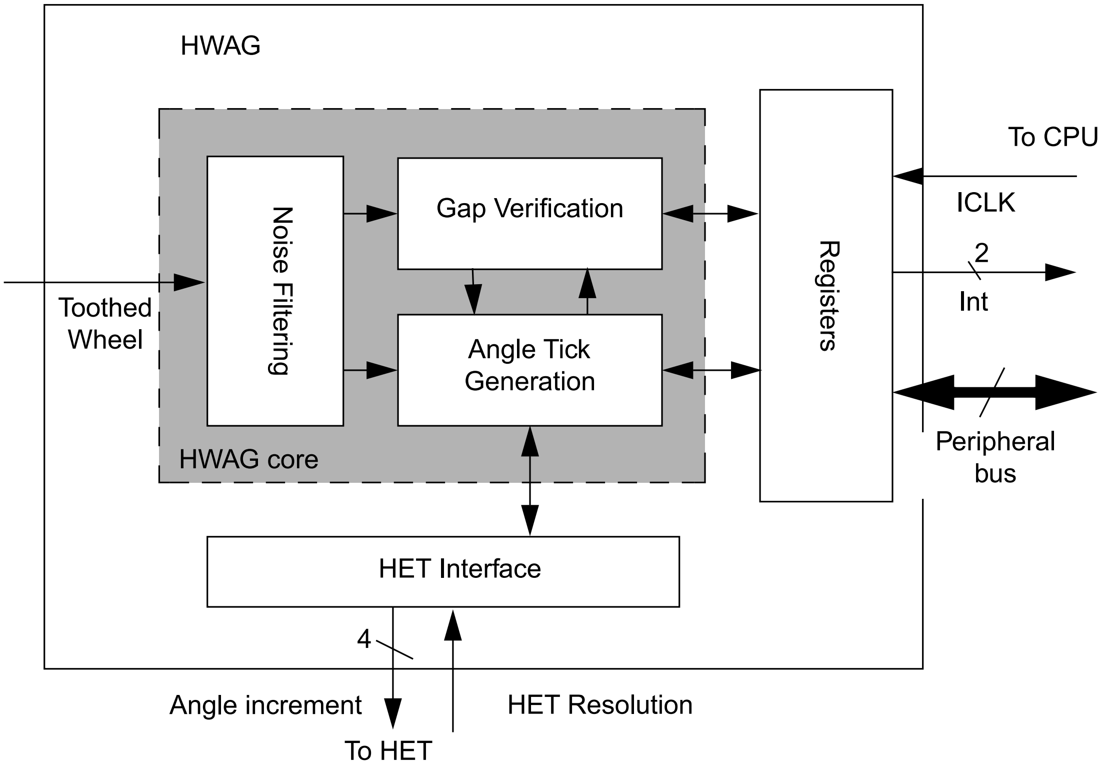
23.3.2.1.1 HWAG Features

The HWAG provides the following features:

- Programmable step width from 1/4 to 1/512
- Automatic synchronization check after first singularity synchronization
- Direct interface with the high-end timer
- 15 to 10,000 RPM range
- Programmable toothed-wheel input filter
- Programmable active edge on toothed-wheel
- Start bit synchronized to the tooth edge
- Pin selection capability for toothed-wheel input

23.3.2.1.2 Block Diagram

Figure 23-36. Hardware Angle Generator Block Diagram



23.3.2.2 HWAG Operation

23.3.2.2.1 Angle Tick Generation Algorithm

23.3.2.2.1.1 Angle Tick Generation Principle

The angle tick generator is the core kernel of this module. It uses the time-interpolation algorithm to generate angle ticks based on the last toothed wheel period. The angle counter is incremented at each new angle tick.

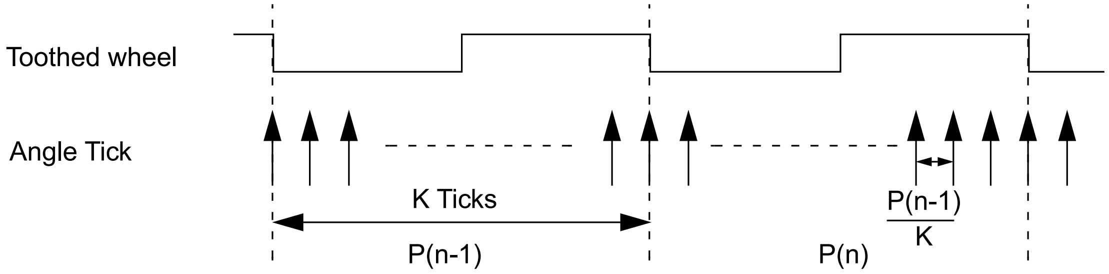
Because the toothed wheel is too inaccurate to fit with actual power-train applications, the algorithm is based on dividing the previous tooth period by K angle steps. The tooth period is the period between two active edges, which the HWAG global control register 2 (HWAGCR2) defines as the falling or the rising edge of the input signal. For an example of the angle tick generation principle, see Figure 23-37.

The speed of the toothed wheel varies. This variance in speed creates some discontinuities in the angle counter behavior.

When the toothed wheel accelerates, the current period becomes shorter than the previous one and the tooth edge arrives before the last tick has been generated. To compensate for any missed ticks, the HWAG adds them to the angle counter when the active edge of the tooth arrives. The angle value is updated and resynchronized at each new active tooth edge.

When the toothed wheel decelerates, the period becomes longer than the previous period and K ticks are already counted before the active edge tooth arrives. After the last tick has been generated, the HWAG generates a tick only after the active tooth edge arrives.

Figure 23-37. Angle Tick Generation Principle



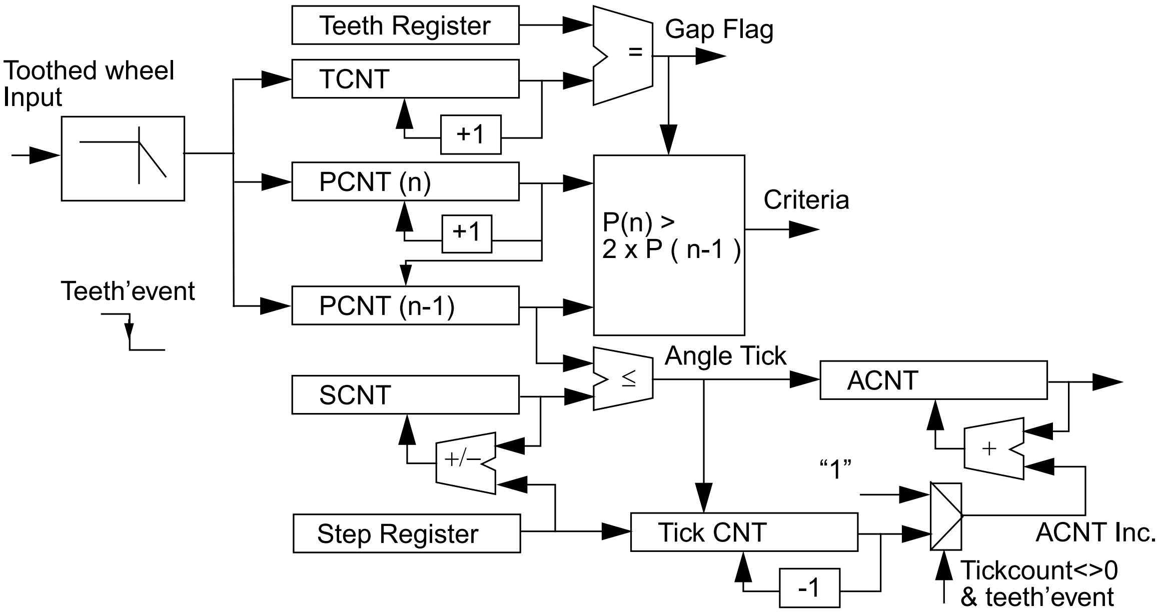
23.3.2.2.1.2 Angle Tick Generation Implementation

The time-interpolation algorithm, which generates ticks based on the toothed wheel tooth period, consists of the following five main counters linked together:

- Tooth counter (TCNT): Current tooth
- Period counter (PCNT): Period between two teeth
- Step counter (SCNT): Angle step
- Tick counter (TCKC): Angle ticks
- Angle counter (ACNT): Angle value

The algorithm also includes differences comparison, adder, and working registers as shown in Figure 23-38.

Figure 23-38. New Angle Tick Generation Architecture



The TCNT is an 8-bit counter. It counts teeth until it reaches the teeth register value then generates a gap flag signal. The gap flag signal which changes the behavior of the HWAG during the singularity and resets the TCNT on the next active edge of the toothed wheel input.

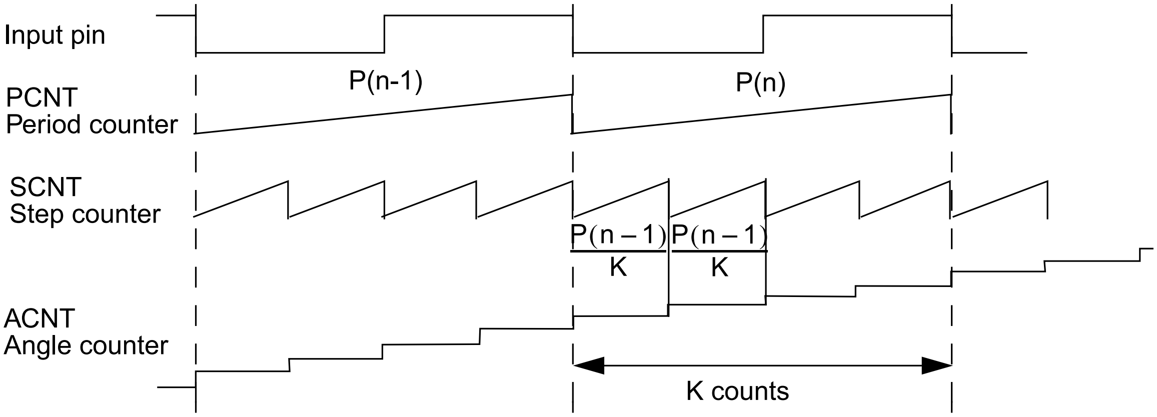
The PCNT calculates the period P(n) between two teeth (two active edges on the toothed wheel input). The active edge (falling or rising) is selected by setting the TED bit in the HWAG global control register 2 (HWAGCR2). On an active edge from the toothed wheel input, the PCNT is saved in the HWAG previous tooth period value register (HWAPCNT1).

The SCNT counts by K steps up to the previous period value, which is contained in the HWAPCNT1 register. When the SCNT overflows PCNT(n-1), an angle tick is generated and SCNT is reset to the remainder between the SCNT and PCNT(n-1). The resulting period of the SCNT is the fraction PCNT(n-1)/K.

The TCKC counts every angle tick until it reaches K and then stops the SCNT. If an active edge occurs before the TCKC has reached K, the remainder is added directly to the ACNT.

When encountering an earlier active edge, the ACNT accumulates the fractions (angle ticks) generated by the SCNT and the remainder of the TCKC. For an example of angle generation using the time-based algorithm, see Figure 23-39.

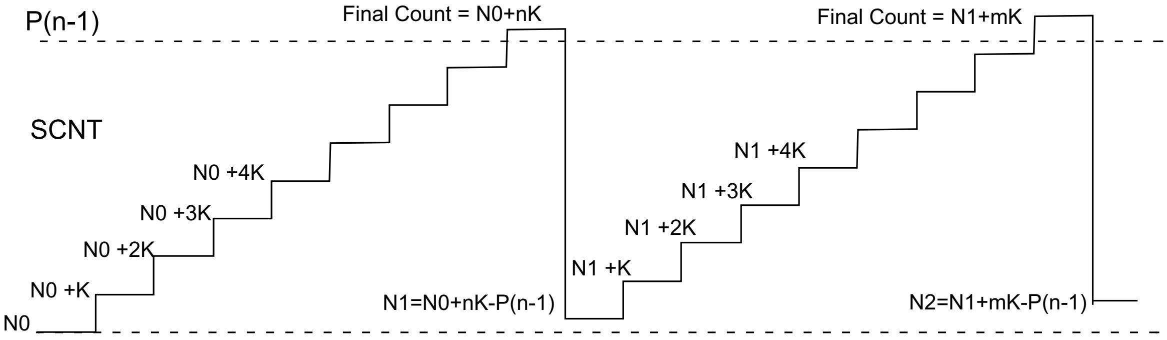
Figure 23-39. Angle Generation Using Time Based Algorithm



Because of stepping, the final count of the SCNT will usually be unequal to the target value PCNT(n-1) and then will overflow. To compensate for this error generated by the algorithm, reset the SCNT to the remainder of the difference between (SCNT - PCNT(n-1)).

To see how the SCNT and PCNT(n-1) generate angle ticks and compensate for the error due to the integer fractions, see Figure 23-40.

Figure 23-40. SCNT Stepping Compensation



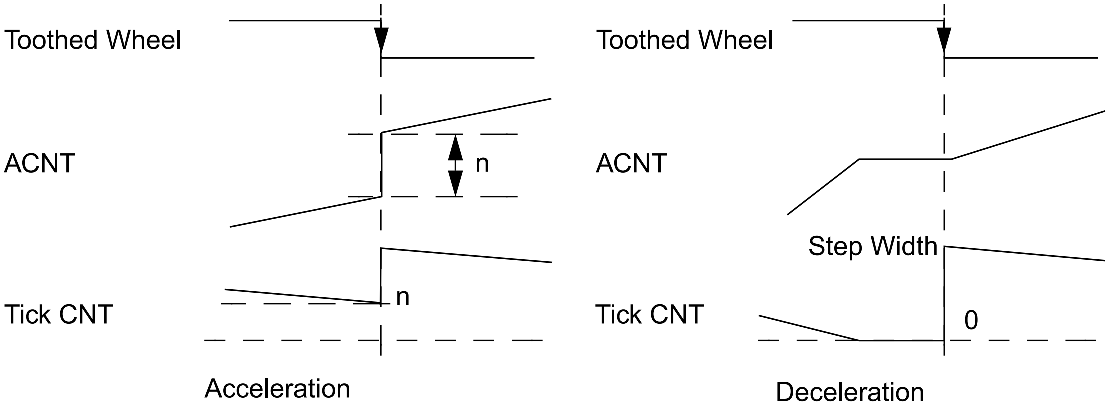
23.3.2.2.1.3 Acceleration and Deceleration

Because the toothed wheel speed is inconstant, it creates discontinuities in the angle counter behavior.

If the TCKC reaches zero before a new active tooth edge during a deceleration, the angle tick signal is no longer generated by the SCNT and PCNT(n-1). This halts the ACNT until the new active tooth arrives.

If the TCKC is unequal to zero when the new active tooth edge arrives during an acceleration (that is, the falling edge on the toothed wheel input in the example below), the rest of the tick counter increments the ACNT. For an example of the ACNT during acceleration and deceleration, see Figure 23-41.

Figure 23-41. ACNT During Acceleration and Deceleration



23.3.2.2.1.4 End of Cycle

The HWAG behaves differently during the singularity tooth period of the toothed wheel. During the singularity period, the HWAG counts three virtual teeth (that is, three times the step width is added to the ACNT) to ensure that the ACNT reaches the maximum value (that is, every angle step has been counted) before resetting it.

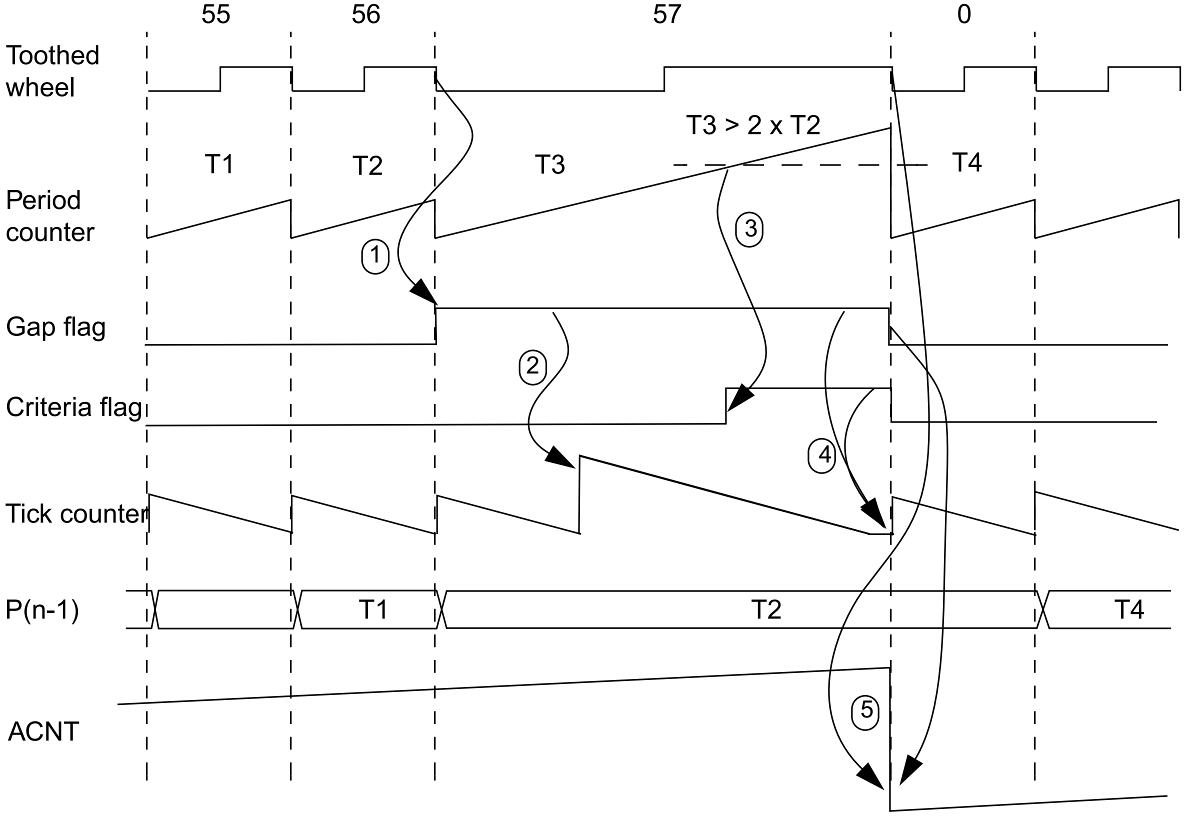
During the singularity period, the HWAG generates angle ticks like for a normal tooth but with three times the value. To generate these angle ticks, the HWAG uses a constant period based on the previous tooth period. Because the period is based on the previous tooth period, the HWAG must recover from a deceleration or acceleration of three teeth when realizing the active edge tooth at the end of the singularity tooth.

The HWAG must ensure that the singularity occurs where expected and must verify it. When the singularity tooth arrives, TCNT reaches the teeth register, sets the signal gap flag, and then keeps PCNT(n-1) until the first tooth of the next round has passed. Because of these conditions, angle ticks before the second tooth will be based on the previous singularity tooth period.

The tick counter is first loaded with a normal value. When the counter reaches zero, it is reloaded once with twice the step width value if the criteria flag is not set. PCNT(n) continues to be incremented and to check the criteria with PCNT(n-1). For more information on gap verification, see Section 23.3.2.2.4. The SCNT continues to generate angle ticks until the tick counter reaches zero the second time. The criteria flag validates the tooth in order to reset the counters. For an example of how the criteria flag validates the tooth to reset the counters, see Figure 23-42.

When the tooth active edge occurs, the ACNT is incremented with the remainder value if the tick counter is not equal to zero. When the ACNT contains a value equals to K times the teeth register, the PCNT, the TCNT and the ACNT are reset to begin a new revolution.

Figure 23-42. Singularity Check, ACNT Reset and Timing Associated



1. When TCNT = teeth register, the Gap flag is raised
2. Tick CNT reloads automatically with 2x the step-width because the Gap flag = ’1’
3. If PCNT (n) > 2 x PCNT (n-1) and the Gap flag = ’1’ then the Criteria flag is raised
4. The tick counter is not reloaded because the Criteria flag is raised
5. The Gap flag and tooth active edge reset, followed by ACNT

23.3.2.2.2 Angle Zero Initialization

Before any angle operation, initialize the HWAG and then initialize the angle zero as the singularity tooth. To initialize the angle zero as the singularity tooth, the HWAG can send an interrupt at each new tooth to help the software detect the first tooth if the interrupt is set. This allows you to decide which algorithm to apply to detect the zero degree tooth (by enabling the corresponding interrupt, you can also use the wired criteria).

When researching which algorithm to apply, the counters ACNT and TCNT are frozen and must be initialized to their start values. The ACNT value is equal to T times the step value (T is the tooth where the start will take effect and the initial value of the tooth counter). The counters PCNT(n) and PCNT(n-1) contain the current period and the previous period respectively. These counters allow you to set a detection criteria. When the application software sets the start bit, the software unfreezes the ACNT and TCNT counters. The counters count from the preloaded values at the next tooth active edge. The ACNT is preloaded with the value of 2 teeth and started synchronously with the next active edge of the toothed wheel. For an example of the HWAG start sequence, see Figure 23-43.

Figure 23-43. Example of HWAG Start Sequence

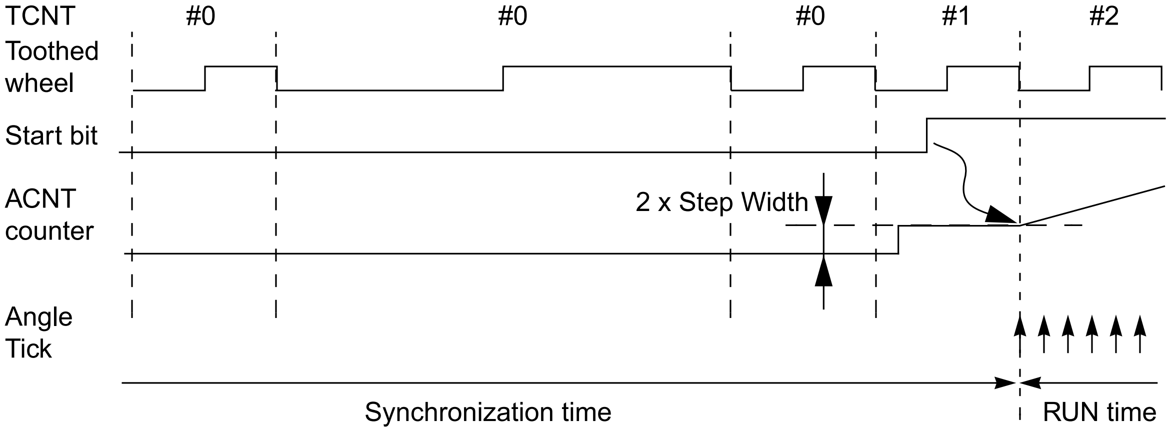
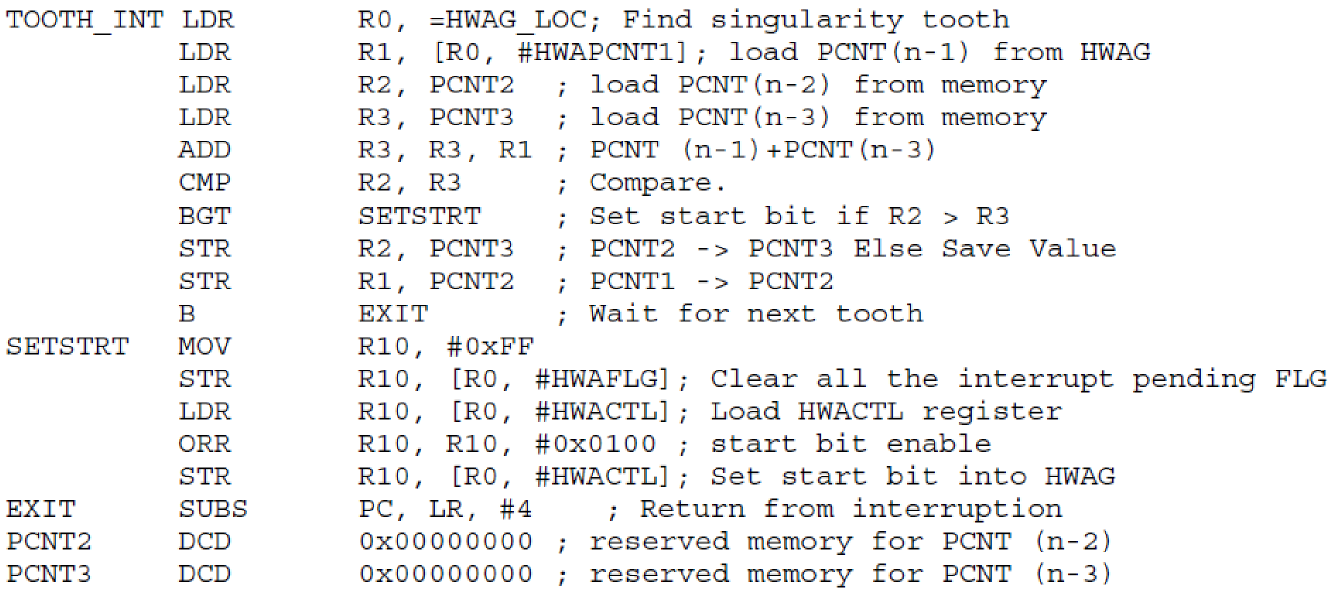


Figure 23-44 is an example of a singularity research initializing the HWAG at the second tooth to start synchronously with the third tooth. The HWAG angle value register (HWAACNT) contains 1024 (2 × 512) and the HWAG current teeth number register (HWATHVL) contains 2.

The code is executed in a tooth interrupt subroutine in code using the PCNT(n-2) > PCNT (n-3) + PCNT (n-1) algorithm.

Figure 23-44. Code



```
TOOTH_INT LDR      R0, =HWAG_LOC; Find singularity tooth
          LDR      R1, [R0, #HWAPCNT1]; load PCNT(n-1) from HWAG
          LDR      R2, PCNT2   ; load PCNT(n-2) from memory
          LDR      R3, PCNT3   ; load PCNT(n-3) from memory
          ADD      R3, R3, R1  ; PCNT (n-1)+PCNT(n-3)
          CMP      R2, R3      ; Compare.
          BGT      SETSTRT     ; Set start bit if R2 > R3
          STR      R2, PCNT3   ; PCNT2 -> PCNT3 Else Save Value
          STR      R1, PCNT2   ; PCNT1 -> PCNT2
          B        EXIT        ; Wait for next tooth
SETSTRT   MOV      R10, #0xFF
          STR      R10, [R0, #HWAFLG]; Clear all the interrupt pending FLG
          LDR      R10, [R0, #HWACTL]; Load HWACTL register
          ORR      R10, R10, #0x0100 ; start bit enable
          STR      R10, [R0, #HWACTL]; Set start bit into HWAG
EXIT      SUBS     PC, LR, #4    ; Return from interruption
PCNT2     DCD      0x00000000 ; reserved memory for PCNT (n-2)
PCNT3     DCD      0x00000000 ; reserved memory for PCNT (n-3)
```

23.3.2.2.3 *Stopping the HWAG*

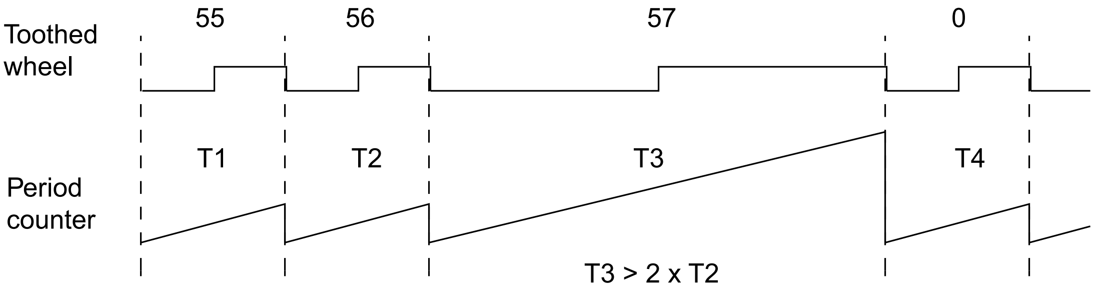
The HWAG starts synchronously with the active edge of the toothed wheel, but stops when the start (STRT) bit in the HWAG global control register 2 (HWAGCR2) is reset. Within a tooth, the HWAG can be stopped and parameters can be changed (that is, step width, angle counter, and so on) If this happens, the restart will take effect on the next active tooth edge.

NOTE: When stopping the HWAG, stop the angle increment delivered to the NHET and set it to zero. Reload the NHET counter with the same value of the angle counter (± corrections), if restarting the HWAG.

23.3.2.2.4 Gap Verification

After the CPU sets the synchronization and puts the HWAG into RUN time (that is, the start bit is set), the tooth counter counts until reaching the teeth register (the number of real teeth of a full wheel revolution). When the tooth counter reaches the teeth register, the gap flag signal is set. For more information on the end of the cycle, see Section 23.3.2.2.1.4. When the gap flag signal is set, it allows the HWAG to verify if the singularity is in the correct position (last tooth). The module then applies the PCNT(n) > 2 x PCNT(n-1) criteria by comparing PCNT(n) and PCNT(n-1) with one bit left shifted. If the criteria does not match when the tooth arrives, then the HWAG sends an interrupt to the CPU and does not reset the ACNT counter. The application software must recover from such an interrupt to keep the HWAG operating optimally. For an example of gap verification criteria for a 60-2 toothed wheel, see Figure 23-45.

Figure 23-45. Gap Verification Criteria For a 60-2 Toothed Wheel



If the hardware criteria is not enabled, you must set the angle reset (ARST) bit in the HWAG global control register 2 (HWAGCR2) to validate the singularity. The HWAGCR2 register must validate the singularity before the active edge of the singularity tooth. If the HWAGCR2 register fails to validate the singularity, the HWAG generates an interrupt and does not clear the ACNT counter when the tooth edge occurs.

NOTE: For a 60-2 toothed wheel, set the ARST flag after the reload of the tick counter(when PCNT(n) = PCNT(n-1)). By verifying the criteria, the application software can set the ARST bit after this point.

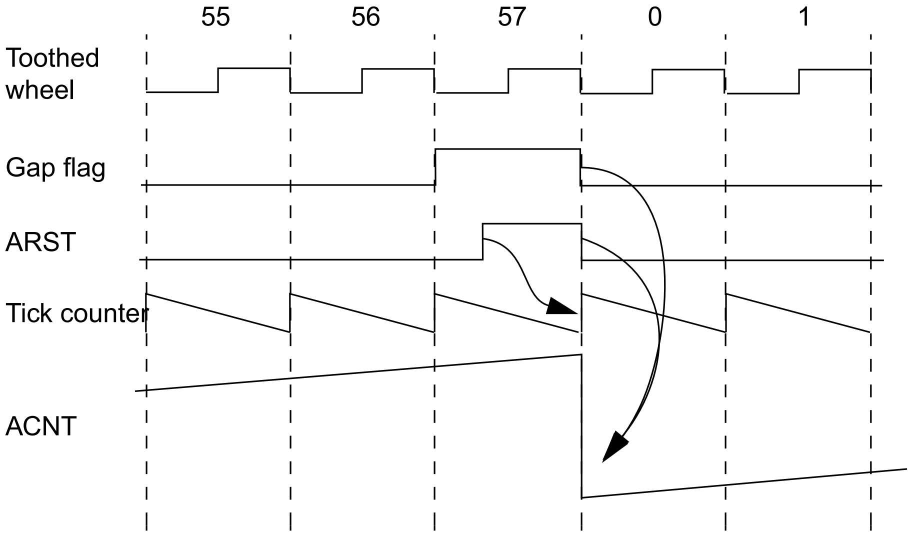
The CPU can read the PCNT counter and make a custom criteria set the ARST bit on time for the HWAG. The application software can use the gap flag interrupt to find the singularity tooth. Alternately, the CPU can verify the validity of the singularity in the second tooth with a more accurate criteria by using the HWAG previous tooth period value register (HWAPCNT1).

23.3.2.2.4.1 Use of the ARST Bit In Case of a Toothed Wheel Without Singularity

If a toothed wheel has no singularity (that is, no missing teeth), the ACNT must be reset when it reaches the angle zero point. To reset the ACNT when it reaches the angle zero point, set the ARST bit to 1.

Setting the ARST bit before the reload of the tick counter will cause the HWAG to fail to reload the tick counter. The HWAG will act like a normal tooth but the next active edge on the toothed wheel input will reset the ACNT and TCNT and clear the ARST bit. For an example of using the ARST bit in a toothed wheel without singularity, see Figure 23-46.

Figure 23-46. Using the ARST Bit in a Toothed Wheel Without Singularity



23.3.2.2.5 Input Noise Filtering

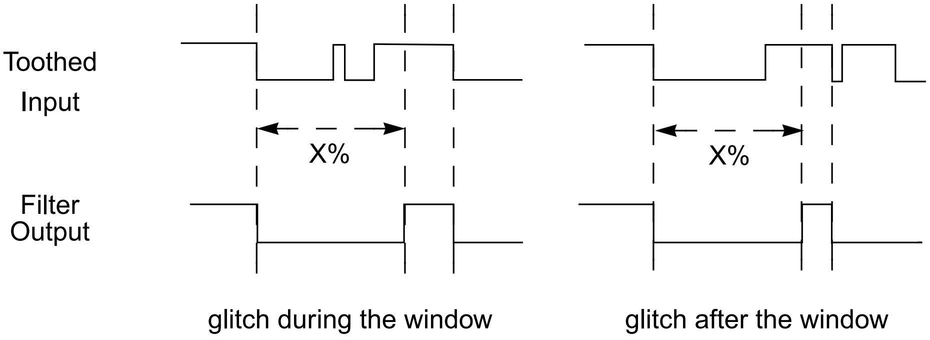
The toothed wheel input comes from an analog part and is sensitive to external noise. Due to this sensitivity, the input needs to be filtered because of glitches in the signal.

The HWAG digitally filters the toothed wheel input signal before it is used inside the core. The filter blocks the signal which negates the effect inside the HWAG. The HWAG provides two filter registers that filter the same way.

The filters validate the input signal after n angle ticks. The n angle ticks are like X% of the tick counter. The value of the remaining percentage of the tick counter (1- X%) need to be set because the tick counter is a down counter. Calculate the value to put into the filter registers from the step width value (or angle ticks value per tooth). The toothed wheel input is like a low pass filter with a cut-off frequency that functions like a toothed-wheel speed, but without acceleration and decelerations side effects. For an example of a windowing filter for a toothed wheel input on a falling active edge, see Figure 23-47.

NOTE: At any time, the CPU can modify the filter values to fine tune with the application.

Figure 23-47. Windowing Filter for Toothed Wheel Input on Falling Active Edge



To calculate this number:

Step Width × (1 – X%) = Filter Register Value

If the step width value is equal to 512 and you want to filter 75% of the tooth, calculate the filter register as follows:

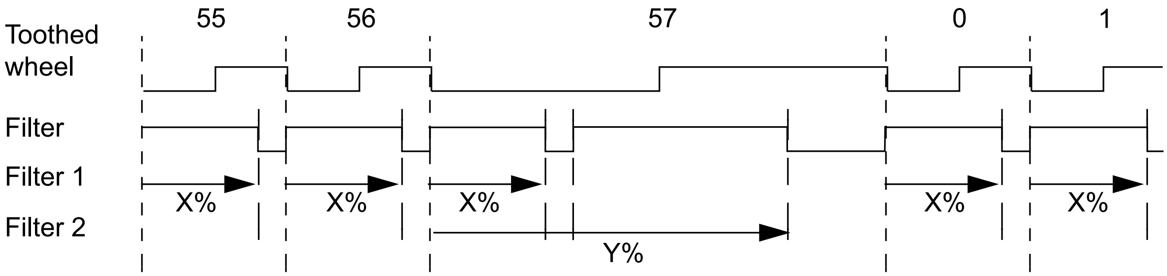
512 × (1 – 0.75) = 128

When the tick counter reaches the filter register value, the toothed wheel input is unblocked.

23.3.2.2.5.1 Filter During Singularity Tooth

During the singularity tooth, the filter acts differently than during a normal tooth. The filter releases the input for a normal tooth. When the tick counter is reloaded, a second filter value is applied to the toothed wheel input. For an example of filtering during a singularity tooth, see Figure 23-48.

Figure 23-48. Filtering During Singularity Tooth



The second filter value is set using the same equation as the first filter with the step width multiplied by 3.

To calculate this number:

$$(3 \times \text{Step Width}) \times (1 - Y\%) = \text{Second Filter Register Value} \quad (30)$$

If the step width value is equal to 512 and you want 70% of singularity tooth period to be filtered, calculate the filter register value as follows:

$$3 \times 512 \times (1 - 0.70) = 460$$

23.3.2.2.6 HWAG Interrupts

When conditions are set, the HWAG interrupts are generated.

When the interrupt condition is true, the corresponding flag is set in the HWAG interrupt flag register (HWAFLG). If the corresponding enable bit in the HWAG interrupt enable set register (HWAENASET) is also set, an interrupt request is sent to the CPU through one of the interrupt lines, depending on the priority of the interrupt (HWAG interrupt level set register (HWALVLSET)).

Because the HWAG can set interruptions, the CPU must determine which source created the interrupt request and then execute the interrupt service routine. The CPU reads the offset register (HWAOFFx) that gives the number of the source. If the CPU reads the offset register, it will automatically clear the source flag that created the request.

NOTE: If the corresponding enable bit is not set, a read in the offset register will not clear a flag. To set the bit, write a 1 in the corresponding bit within the HWAG interrupt flag register (HWAFLG).

The HWAG generates eight different interrupts:

- 0 = Overflow period
- 1 = Singularity not found
- 2 = Tooth interrupt
- 3 = ACNT overflow
- 4 = PCNT(n) > 2 × PCNT (n-1) during normal tooth
- 5 = Bad active edge tooth
- 6 = Gap flag
- 7 = Angle increment overflow

For more information on these interrupts, see Table 23-14. Each interrupt source is associated with a low or high priority. When one or more interrupts with the same priority occur, a fixed priority determines the offset vector if the corresponding enable bits are set.

The HWAG generates two interrupt request signals for the central interrupt module (CIM). For information on servicing interrupts, see Figure 23-49. For a list offset values, see Table 23-13.

Table 23-13. HWAG Interrupt Sources and Offset Values

Source Number	Offset Value
0	1
1	2
:	:
7	8

Figure 23-49. HWAG Interrupt Block Diagram

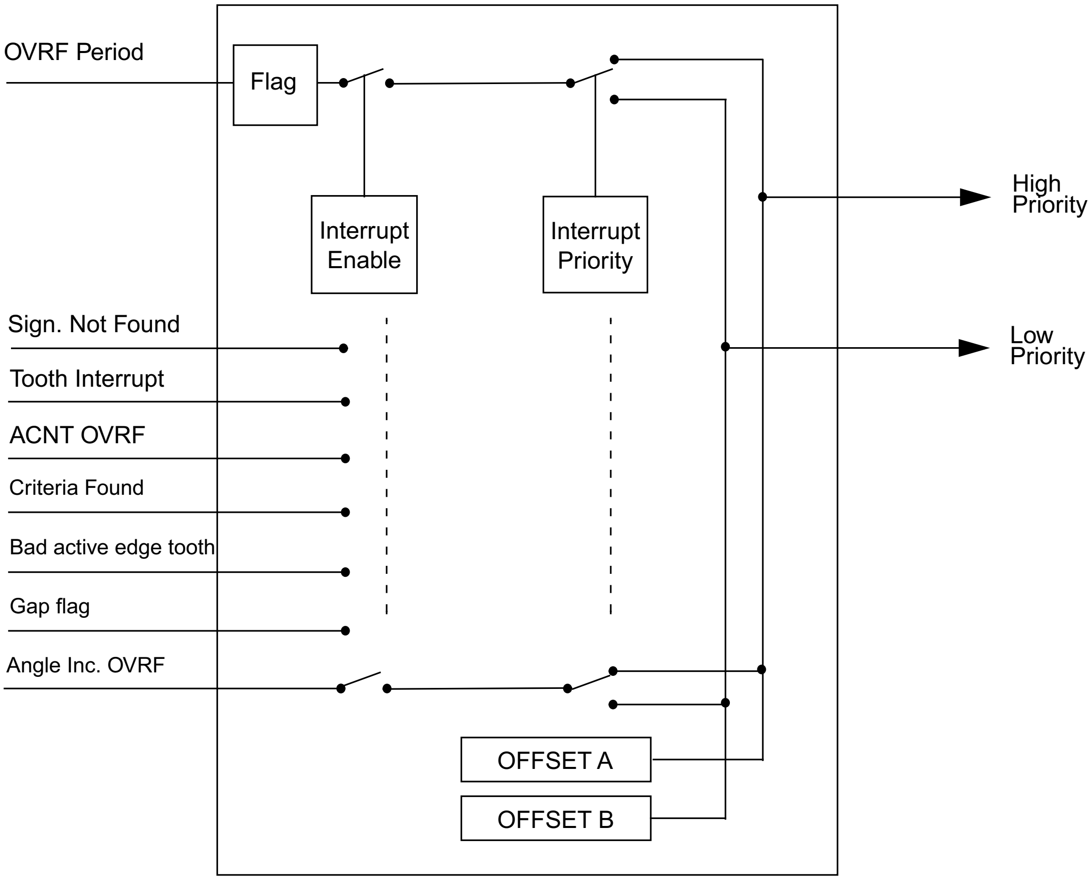


Table 23-14. HWAG Interrupt Descriptions

Interrupt Names	Interrupt Descriptions
Overflow period	Occurs when the PCNT (n) counter reaches the maximum value. Can occur if the toothed wheel input remains stable. May indicate failure of an engine stall or a toothed wheel sensor.
Singularity not found	When the TCNT counter sets the gap flag, the HWAG waits for the criteria flag to raise before the toothed wheel active edge. If the toothed wheel active edge occurs before the criteria flag, the HWAG raises the singularity not found interrupt flag.
New edge tooth	This interrupt can sync or let you control the tick generation. This interrupt indicates the new active edge tooth. This interrupt could be filtered or unfiltered (Bit FIL in control register).
Angle counter (ACNT) overflow	This interrupt occurs when the singularity is unable to be found. The angle counter (ACNT) continues until overflow.
Singularity found during normal tooth	This interrupt indicates that the period of the current tooth is at least two times longer than the previous one when the HWAG expects a normal tooth. This interrupt can detect the singularity without bit manipulation by the CPU.
Bad active edge tooth	This interrupt indicates that an active edge has occurred before the end of the filtering (toothed wheel input blocked) but the HWAG remains inactive internally. This interrupt can detect glitches on the toothed wheel input.
Gap flag	When TCNT reaches the teeth register and the HWAG raises the gap flag , This interrupt is set when the gap flag is raised by the HWAG,
Angle increment overflow	This interrupt indicates that the number of the angle increment is more than 15 since the last resolution tick. This interrupt can prevent any discrepancies between the NHET and the HWAG.

NOTE: Before enabling any interruption, clear the HWAG interrupt flag register (HWAFLG) to ensure that any interrupts have finished. If interrupts are pending, the HWAG could generate an interrupt based on an unrealistic event.

23.3.2.3 Emulation

Because the HWAG is designed to synchronize with a real-time environment, the HWAG counters continue during emulation.

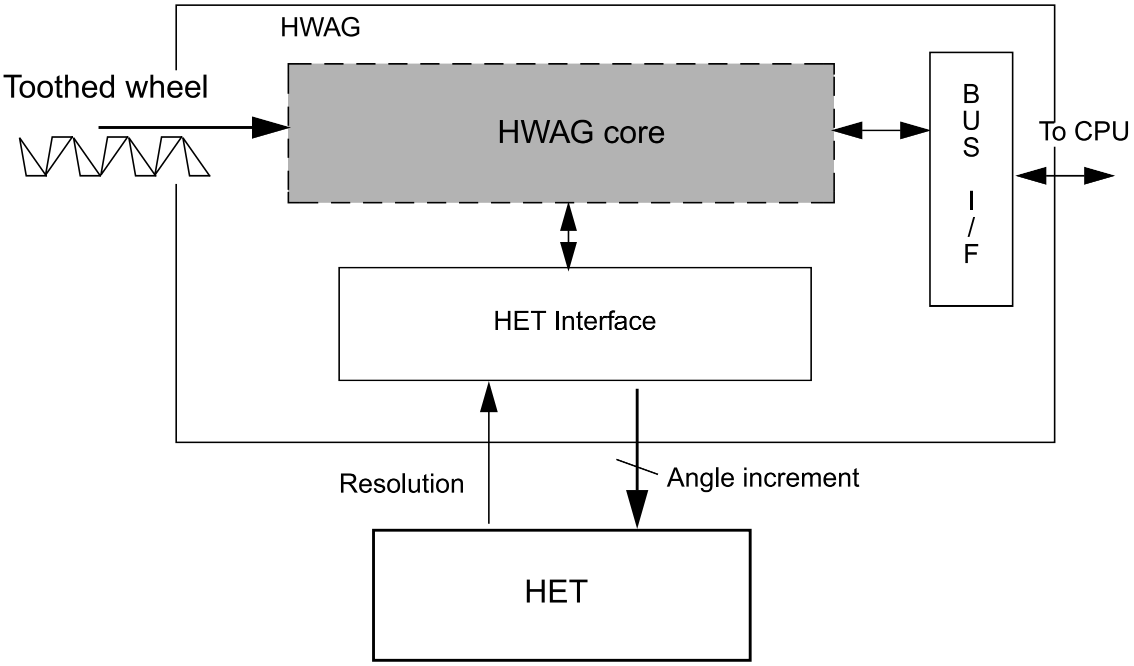
When the CPU is frozen, the HWAG continues to run and update registers. Only the offset registers remain uncleared when entering debug mode.

During debug mode, interrupts can occur and will wait until the CPU enters run mode again. If interrupts occur, they could affect synchronization with the toothed wheel

23.3.2.4 Hardware Angle Generator and High-End Timer

In the engine management application, the HWAG is usually connected to one or more high-end timers. This connection allows you to perform angle compare and angle/time compare. For an example of the hardware angle generator/high-end timer interface, see Figure 23-50.

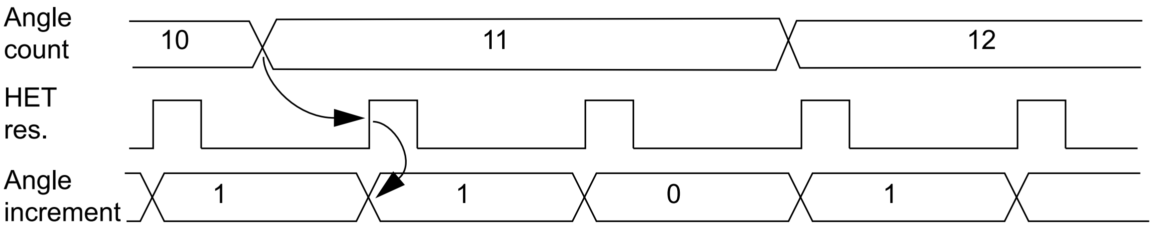
Figure 23-50. Hardware Angle Generator/High End Timer Interface



23.3.2.4.1 Signal Description

To perform a resynchronization, the HWAG interface provides to the NHET at every resolution clock an angle increment value that represents how much the angle counter of the HWAG has been incremented since the last NHET resolution clock. For an example of the angle count within the HWAG, see Figure 23-51.

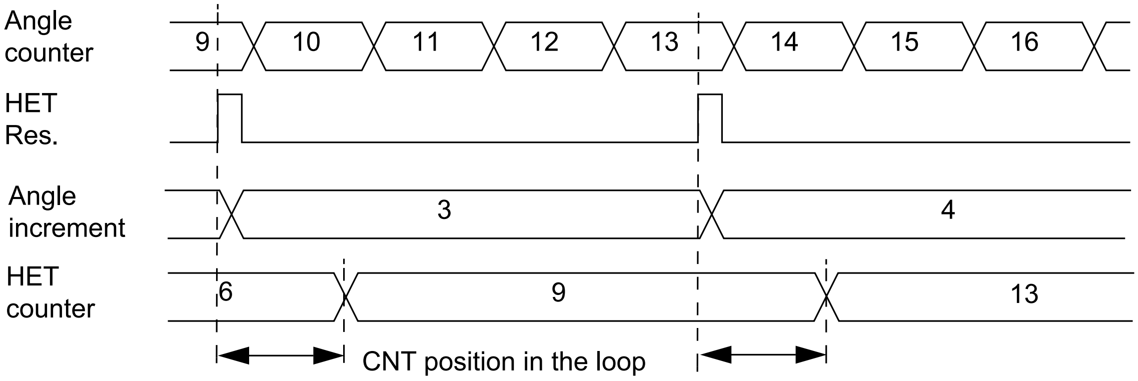
Figure 23-51. Angle Count Within the HWAG at Resolution Clock



When the engine speed increases, the angle count can increment by more than one in a NHET resolution but the HWAG will continue to provide the angle increment value at every resolution..

The NHET can then implement its own angle counter (using a CNT instruction in angle mode) which will be incremented once per resolution by the value given by the angle increment. For an example of an angle count within the NHET with increments, see Figure 23-52.

Figure 23-52. Angle Count Within the NHET With Increments

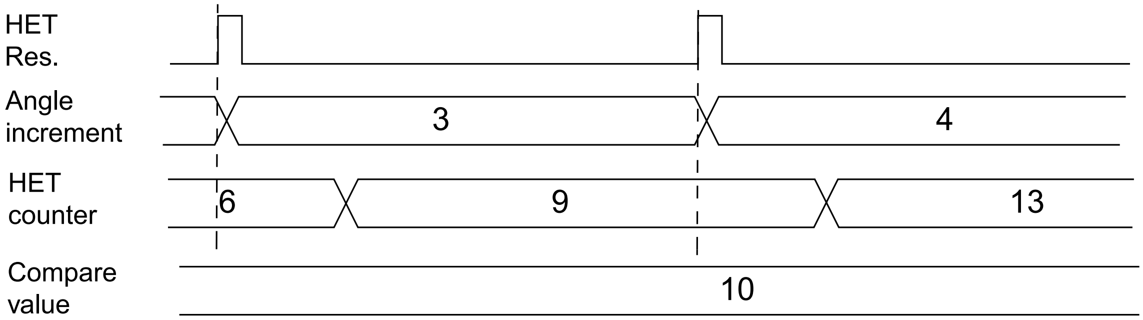


23.3.2.4.2 NHET Operation on Angle Functions (ACMP, CNT)

23.3.2.4.2.1 State of the Art

Because the angle value can be increased by more than one, the compare value could be in-between the old angle value and the new angle value of the NHET angle counter (where new angle value = old angle value + angle increment). To perform an angle compare that ensures not to miss a compare value, the NHET provides the ACMP instruction. For an example of a compare without ACMP instruction, see Figure 23-53.

Figure 23-53. Compare Without ACMP Instruction

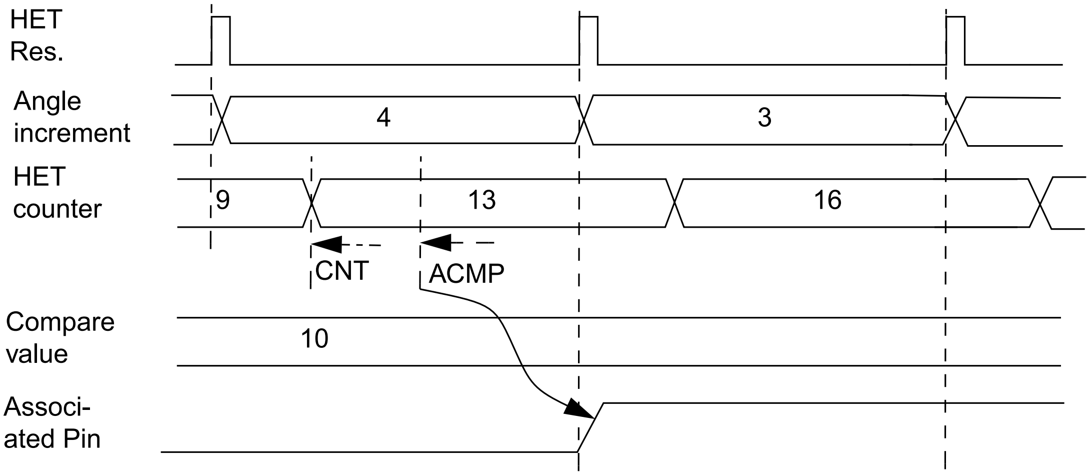


When the HET counter passes from 9 to 13, the equality compare can not match the compare value 10. Consequently, the angle position is missed!

23.3.2.4.2.2 ACMP Instruction Advantage

The ACMP instruction is more than an equality compare. ACMP instruction performs an in-between comparison (old angle value < compare value ≤ new angle value) to match the position of the toothed wheel. This instruction, where an equality compare executes every resolution, may miss a compare match. For an example of ACMP compare within the NHET, see Figure 23-54.

Figure 23-54. Example of ACMP Compare Within the NHET



With the ACMP instruction, the compare that is performed will be: **9** < **10** ≤ **13**

With the ACMP instruction, the compare is: 9 < 10 ≤ 13

NOTE: To avoid multiple matches, the ACMP only matches during a single resolution.

Performing the following equations at the same time implements this compare:

CMP > NHET angle counter – Angle increment

CMP ≤ NHET angle counter

23.3.2.4.3 NHET Interface

23.3.2.4.3.1 Input Signal Selection

The input pin of the toothed-wheel signal is software selectable. In previous generations of NHET/HWAG, this was fixed to HET[2]. On this device, the input pin is programmable to provide more flexibility for the system implementation. However, the implementation is done in a way to be backward compatible.

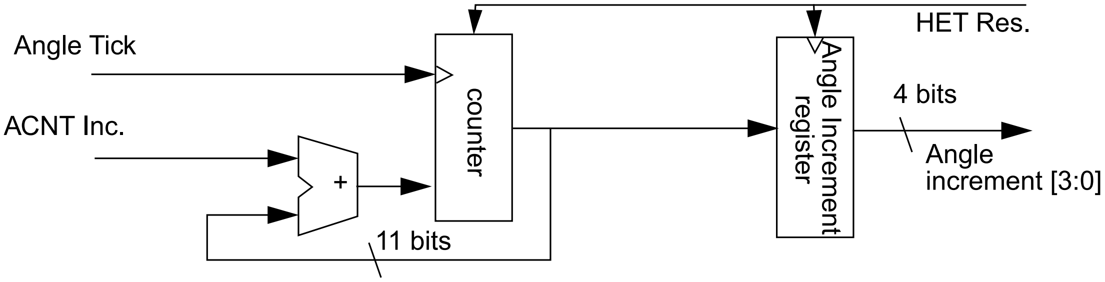
A separate register, HWAG pin select register (HWAPINSEL), is implemented to allow this selection functionality. The HWAPINSEL register should be programmed before the HWAG is turned on. The default selection will be HET[2] (PINSEL = 2h) after reset. The signals will be derived from the input buffer of each pin. This will allow configuring the pin as an output and measure back the output signal with the HWAG.

You can change the HWAPINSEL register at any time, but the proper functionality of the HWAG is not assured if the selection is changed while the HWAG is already operational. It is recommended that the input selection is done before the STRT bit in the HWAG global control register 2 (HWAGCR2)) is programmed to 1.

23.3.2.4.3.2 HWAG to NHET Interface

The NHET interface is a 11-bit counter sampled by the NHET and reset by the NHET resolution. The counter contains the value of ACNT incremented during the last resolution (see Section 23.3.2.4.1). For the NHET interface block diagram, see Figure 23-55.

Figure 23-55. NHET Interface Block Diagram



When the ACNT register is reset to zero, the angle increment register is not reset. The NHET software checks if its own angle register is higher than 360° and either clears it or continues to 720°. If ACNT is reset within the HWAG, the angle increment register gives the NHET the number of angle ticks from the last resolution.

During a strong acceleration after a tooth active edge, the number of angle ticks can exceed 15. If the number of ticks exceeds 15, the HWAG delivers to the NHET several angle increments at 15. This allow the NHET to follow without missing any angle positions from the HWAG. When the counter is below 15, the angle increment reflects the counter. When the angle increment overflows, sets to 15, and if the enable bit (bit 7 in the control register) is set, the HWAG can send an interrupt to the CPU.

During a strong deceleration, the angle increment can stay null for one or more NHET resolution clocks.

To minimize the error between the fly-wheel and NHET angle counter, the step width and the NHET resolution must be set to avoid any overflow of the 11-bit counter of the NHET interface. This can happen if the number of angle ticks always exceeds 15 during one resolution.

23.3.2.5 Range of Operations

23.3.2.5.1 Intrinsic HWAG Limitation

The following factors limit the HWAG:

- SYSCLK
- PCNT counter (overflow)
- Number of teeth
- Angle step

These factors will influence the engine speed range (RPM limitation) and the maximum accuracy of the angle steps (wheel limitation).

- RPM limitation

  The toothed wheel speed is limited by the period counter (PCNT) and the angle step for a given SYSCLK.

  RPM minimum is related to PCNT overflow and SYSCLK.

  Maximum PCNT value × SYSCLK = Maximum tooth period

  $$\mathrm{RPM} = \frac{60}{\mathrm{TeethNumber} \times \mathrm{ToothPeriod}}$$

  PCNT is a 24-bit counter based on SYSCLK.

  RPM maximum is related to the angle step and SYSCLK.

  Minimum tooth period > Step Width × SYSCLK

  The angle ticks period could not be inferior to the SYSCLK.

  Example: The toothed wheel is a 60-2, SYSCLK is 50 Mhz (20 ns), and step width is 512:

  RPM minimum ≥ 16 777 215 × 20 ns = 335.5443 ms ≥ ~3 RPM

  RPM maximum ≥ 512 × 20 ns = 10.24 µs ≥ 97 656 RPM

  NOTE: With a 60-2 toothed wheel, the tooth period is the reverse of the RPM number.

- Wheel Limitation

  The HWAG is limited by the number of teeth and the increments in a revolution.

  The maximum number of teeth is 256. This limits the number of increments per revolution to 512 steps × 256 teeth = 131 072 angle increments.

23.3.2.5.2 HWAG-NHET Limitation

The maximum angle accuracy is a function of the angle step and the NHET loop resolution.

The increment per resolution limits the interface between the HWAG and the NHET. The maximum angle increment per NHET resolution is 15 increments/NHET_res, which is an angular speed. If the angle increment overflows 15 during a constant speed, the system is diverging.

In the HWAG, the angular speed is given by the relation:

$$\text{Angular Speed} = \frac{\text{Step Width}}{\text{Minimum Tooth Period}}$$

To ensure that the values are correct, they must satisfy the following equation:

$$\frac{\text{Max HET resolution} \times \text{Step Width}}{\text{Minimum Tooth Period}} < 15$$

Then,

$$\text{Max HET resolution} = \frac{15 \times \text{Min Tooth Period}}{\text{Step Width}}$$

Example: For a 60-2 at 10000 RPM, the tooth period is 100 µs and the step width is 512:

$$\text{Max HET resolution} = \frac{15 \times 100}{512} = 2.93\,\mu\text{s}$$

23.3.2.6 Tricks

23.3.2.6.1 Using HWAG Previous Tooth Period Value Register (HWAPCNT1)

The HWAG previous tooth period value register (HWAPCNT1) can compensate for errors because of acceleration or deceleration.

If there is a variation of the toothed wheel, the ACNT register will have a discontinuity . For an explanation of acceleration and deceleration, see Section 23.3.2.2.1.3. Avoid this discontinuity by giving the HWAPCNT1 register a smaller or larger value, depending of the variation. When HWAPCNT1 is modified, the angle tick period is also be modified which causes faster or slower tick generation and decreases the discontinuity on the next falling edge.

Because of this compensation, the NHET interface will not overflow and fewer errors will occur on the NHET angle counter in case of strong acceleration.

NOTE: Reading the angle increment will give the application the amount of the acceleration. However, adding the value directly to the NHET counter will result in a discontinuity in the compare sequence. Particularly angle based compare could be missed.

23.3.2.6.2 Using the Singularity During Normal Tooth Interrupt

This interrupt detects if the HWAG is desynchronized with the toothed wheel and resynchronizes the HWAG.

Because the criteria was set during a tooth other than the singularity tooth, the interrupt occurs. Because the criteria is based on PCNT > 2 × PCNT (n-1), this interrupt is likely due to the singularity.

The following steps explain how to resynchronize the HWAG with this interrupt:

1. Stop the HWAG
2. Reset ACNT
3. Reset tooth counter
4. Reset interrupt
5. Set start bit.

The HWAG will restart on the tooth zero.

23.4 N2HET Control Registers

Table 23-15 summarizes all the N2HET registers. The base address for the control registers is FFF7 B800h for N2HET1 and FFF7 B900h for N2HET2.

Table 23-15. N2HET Registers

Offset	Acronym	Register Description	Section
00h	HETGCR	Global Configuration Register	Section 23.4.1
04h	HETPFR	Prescale Factor Register	Section 23.4.2
08h	HETADDR	NHET Current Address Register	Section 23.4.3
0Ch	HETOFF1	Offset Index Priority Level 1 Register	Section 23.4.4
10h	HETOFF2	Offset Index Priority Level 2 Register	Section 23.4.5
14h	HETINTENAS	Interrupt Enable Set Register	Section 23.4.6
18h	HETINTENAC	Interrupt Enable Clear Register	Section 23.4.7
1Ch	HETEXC1	Exception Control Register 1	Section 23.4.8
20h	HETEXC2	Exception Control Register 2	Section 23.4.9
24h	HETPRY	Interrupt Priority Register	Section 23.4.10
28h	HETFLG	Interrupt Flag Register	Section 23.4.11
2Ch	HETAND	AND Share Control Register	Section 23.4.12
34h	HETHRSH	HR Share Control Register	Section 23.4.13
38h	HETXOR	HR XOR-Share Control Register	Section 23.4.14
3Ch	HETREQENS	Request Enable Set Register	Section 23.4.15
40h	HETREQENC	Request Enable Clear Register	Section 23.4.16
44h	HETREQDS	Request Destination Select Register	Section 23.4.17
4Ch	HETDIR	NHET Direction Register	Section 23.4.18
50h	HETDIN	NHET Data Input Register	Section 23.4.19
54h	HETDOUT	NHET Data Output Register	Section 23.4.20
58h	HETDSET	NHET Data Set Register	Section 23.4.21
5Ch	HETDCLR	NHET Data Clear Register	Section 23.4.22
60h	HETPDR	NHET Open Drain Register	Section 23.4.23
64h	HETPULDIS	NHET Pull Disable Register	Section 23.4.24
68h	HETPSL	NHET Pull Select Register	Section 23.4.25
74h	HETPCR	Parity Control Register	Section 23.4.26
78h	HETPAR	Parity Address Register	Section 23.4.27
7Ch	HETPPR	Parity Pin Register	Section 23.4.28
80h	HETSFPRLD	Suppression Filter Preload Register	Section 23.4.29
84h	HETSFENA	Suppression Filter Enable Register	Section 23.4.30
8Ch	HETLBPSEL	Loop Back Pair Select Register	Section 23.4.31
90h	HETLBPDIR	Loop Back Pair Direction Register	Section 23.4.32
94h	HETPINDIS	NHET Pin Disable Register	Section 23.4.33

23.4.1 Global Configuration Register (HETGCR)

N2HET1: offset = FFF7 B800h; **N2HET2:** offset = FFF7 B900h

Figure 23-56. Global Configuration Register (HETGCR) [offset = 00h]

31	25	24
Reserved		HET_PIN_ENA
R-0		R/W-1

23	22	21	20	19	18	17	16
Reserved	MP		Reserved		PPF	IS	CMS
R-0	R/W-0		R-0		R/W-0	R/W-0	R/W-0

15	1	0
Reserved		TO
R-0		R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 23-16. Global Configuration Register (HETGCR) Field Descriptions

Bit	Field	Value	Description
31-25	Reserved	0	Reads return 0. Writes have no effect.
24	HET_PIN_ENA		Enables the output buffers of the pin structures depending on the value of nDIS and DIR.x when PINDIS.x is set.
			Note: This bit will automatically get cleared when nDIS pin (input port) value is 0.
		0	No affect on the pin output buffer structure.
		1	Enables the pin output buffer structure when DIR = output, PINDIS.x is set and nDIS = 1.
23	Reserved	0	Reads return 0. Writes have no effect.
22-21	MP		Master Priority
			The NHET can prioritize master accesses to N2HET RAM between the HET Transfer Unit and another arbiter, which outputs the access of one of the remaining masters. The MP bits allow the following selections:
		0	The HTU has lower priority to access the N2HET RAM than the arbiter output.
		1h	The HTU has higher priority to access the N2HET RAM than the arbiter output.
		2h	The HTU and the arbiter output use a round robin scheme to access the N2HET RAM.
		3h	Reserved
20-19	Reserved	0	Reads return 0. Writes have no effect.
18	PPF		Protect Program Fields
			The PPF bit together with the Turn On/Off bit (TO) allows to protect the program fields of all instructions in N2HET RAM.
			When TO = 0:
		0	All masters can read and write the program fields.
		1	All masters can read and write the program fields.
			When TO = 1:
		0	All masters can read and write the program fields.
		1	The program fields are readable but not writable for all masters, which could access the N2HET RAM. Possible masters are the CPU, HTU, DMA and a secondary CPU (if available). Writes initiated by these masters are discarded.
17	IS		Ignore Suspend
			When Ignore Suspend = 0, the timer operation is stopped on suspend (the current timer instruction is completed). Timer RAM can be freely accessed during suspend. When set to 1, the suspend is ignored and the N2HET continues operating.
		0	N2HET stops when in suspend mode.
		1	N2HET ignores suspend mode and continues operation.

Table 23-16. Global Configuration Register (HETGCR) Field Descriptions (continued)

Bit	Field	Value	Description
16	CMS		Clk_master/slave This bit is used to synchronize multi-N2HETs. If set (N2HET is master), the N2HET outputs a signal to synchronize the prescalers of the slave N2HET. By default, this bit is reset, which means a slave configuration. **Note:** This bit must be set to one (1) for single-N2HET configuration.
		0	N2HET is configured as a slave.
		1	N2HET is configured as a master.
15-1	Reserved	0	Reads return 0. Writes have no effect.
0	TO		Turn On/Off TO does not affect the state of the pins. You must set/reset the timer pins when they are turned off, or re-initialize the timer RAM and control registers before a reset. After a device reset, the timer is turned off by default.
		0	N2HET is OFF. The timer program stops executing. Turn-off is automatically delayed until the current timer program loop is completed. Turn-off does not affect the content of the timer RAM, ALU registers, or control registers. Turn-off resets all flags.
		1	N2HET is ON. The timer program execution starts synchronously to the Loop clock. In case of multiple N2HETs configuration, the slave N2HETs are waiting for the loop clock to come from the master before starting execution. Then, the timer address points automatically address 00h (corresponding to program start).

23.4.2 Prescale Factor Register (HETPFR)

N2HET1: offset = FFF7 B804h; **N2HET2:** offset = FFF7 B904h

Figure 23-57. Prescale Factor Register (HETPFR)

31 … 17 16
Reserved
R-0

15 … 11	10 … 8	7 … 6	5 … 0
Reserved	LRPFC	Reserved	HRPFC
R-0	R/WP-0	R-0	R/WP-0

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privileged mode only; -*n* = value after reset

Table 23-17. Prescale Factor Register (HETPFR) Field Descriptions

Bit	Field	Value	Description
31-11	Reserved	0	Reads return 0. Writes have no effect.
10-8	LRPFC		Loop-Resolution Pre-scale Factor Code. LRPFC determines the loop-resolution prescale divide rate (lr).
		0	/1
		1h	/2
		2h	/4
		3h	/8
		4h	/16
		5h	/32
		6h	/64
		7h	/128
7-6	Reserved	0	Reads return 0. Writes have no effect.
5-0	HRPFC		High-Resolution Pre-scale Factor Code. HRPFC determines the high-resolution prescale divide rate (hr).
		0	/1
		1h	/2
		2h	/3
		3h	/4
		:	:
		3Dh	/62
		3Eh	/63
		3Fh	/64

23.4.3 N2HET Current Address Register (HETADDR)

N2HET1: offset = FFF7 B808h; **N2HET2:** offset = FFF7 B908h

Figure 23-58. N2HET Current Address (HETADDR)

31	16
Reserved	
R-0	

15	9	8	0
Reserved		HETADDR	
R-0		R-0	

LEGEND: R = Read only; -*n* = value after reset

Table 23-18. N2HET Current Address (HETADDR) Field Descriptions

Bit	Field	Value	Description
31-9	Reserved	0	Reads return 0. Writes have no effect.
8-0	HETADDR		N2HET Current Address Read: Returns the current N2HET program address. Write: Writes have no effect.

23.4.4 Offset Index Priority Level 1 Register (HETOFF1)

N2HET1: offset = FFF7 B80Ch; **N2HET2:** offset = FFF7 B90Ch

Figure 23-59. Offset Index Priority Level 1 Register (HETOFF1)

31	16
Reserved	
R-0	

15	6	5	0
Reserved		OFFSET1	
R-0		R-0	

LEGEND: R = Read only; -*n* = value after reset

Table 23-19. Offset Index Priority Level 1 Register (HETOFF1) Field Descriptions

Bit	Field	Value	Description
31-6	Reserved	0	Reads return 0. Writes have no effect.
5-0	OFFSET1		OFFSET1 indexes the currently pending high-priority interrupt. Offset values and sources are listed in Table 23-20. Read: Read of these bits determines the pending N2HET interrupt. Write: Writes have no effect. **Note:** In any read operation mode, the corresponding flag (in the HETFLG) is also cleared. In Emulation mode the corresponding flag is not cleared.

Table 23-20. Interrupt Offset Encoding Format

Offset Value	Source No.
0	No interrupt
1	Instruction 0, 32, 64...
2	Instruction 1, 33, 65...
:	:
32	Instruction 31, 63, 95...
33	Program Overflow
34	APCNT Underflow
35	APCNT Overflow

23.4.5 Offset Index Priority Level 2 Register (HETOFF2)

N2HET1: offset = FFF7 B810h; **N2HET2:** offset = FFF7 B910h

Figure 23-60. Offset Index Priority Level 2 Register (HETOFF2)

31-16
Reserved
R-0

15-6	5-0
Reserved	OFFSET2
R-0	R-0

LEGEND: R = Read only; -*n* = value after reset

Table 23-21. Offset Index Priority Level 2 Register (HETOFF2) Field Descriptions

Bit	Field	Value	Description
31-6	Reserved	0	Reads return 0. Writes have no effect.
5-0	OFFSET2		OFFSET2 indexes the currently pending low-priority interrupt. Offset values and sources are listed in Table 23-20. Read: Read of these bits determines the pending N2HET interrupt. Write: Writes have no effect. **Note:** In any read operation mode, the corresponding flag (in the HETFLG) is also cleared. In Emulation mode, the corresponding flag is not cleared.

23.4.6 Interrupt Enable Set Register (HETINTENAS)

N2HET1: offset = FFF7 B814h; **N2HET2:** offset = FFF7 B914h

Figure 23-61. Interrupt Enable Set Register (HETINTENAS)

31	16
HETINTENAS	
R/W-0	

15	0
HETINTENAS	
R/W-0	

LEGEND: R/W = Read/Write; -*n* = value after reset

Table 23-22. Interrupt Enable Set Register (HETINTENAS) Field Descriptions

Bit	Field	Value	Description
31-0	HETINTENAS[n]		Interrupt Enable Set bits. HETINTENAS is readable and writable in any operation mode. Writing a 1 to bit x enables the interrupts of the N2HET instructions at N2HET addresses x+0, x+32, x+64, and so on. Generating an interrupt requires to set bit x in HETINTENAS and to enable the interrupt bit in one of the instructions at addresses x+0, x+32, x+64, and so on. To avoid ambiguity, only one of the instructions x+0, x+32, x+64, and so on, should have the interrupt enable bit (inside the instruction) set. Writing a 0 to HETINTENAS has no effect. When reading from HETINTENAS bit x gives the information, if N2HET instructions x+0, x+32, x+64, and so on, have the interrupt enabled or disabled.
		0	Read: Interrupt is disabled. Write: Writes have no effect.
		1	Read: Interrupt is enabled. Write: Interrupt is enabled.

23.4.7 Interrupt Enable Clear Register (HETINTENAC)

N2HET1: offset = FFF7 B818h; **N2HET2:** offset = FFF7 B918h

Figure 23-62. Interrupt Enable Clear (HETINTENAC)

31	16
HETINTENAC	
R/W-0	

15	0
HETINTENAC	
R/W-0	

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 23-23. NHET Interrupt Enable Clear (HETINTENAC) Field Descriptions

Bit	Field	Value	Description
31-0	HETINTENAC[n]		Interrupt Enable Clear bits. HETINTENAC is readable and writable in any operation mode. Writing a 1 to bit x disables the interrupts of the N2HET instructions at N2HET addresses x+0, x+32, x+64, and so on. (See also description in Table 23-22). Writing a 0 to HETINTENAC has no effect. When reading from HETINTENAC bit x gives the information, if N2HET instructions x+0, x+32, x+64, and so on, have the interrupt enabled or disabled.
		0	Read: Interrupt is disabled. Write: Writes have no effect.
		1	Read: Interrupt is enabled. Write: Interrupt is disabled.

23.4.8 Exception Control Register 1 (HETEXC1)

N2HET1: offset = FFF7 B81Ch; **N2HET2:** offset = FFF7 B91Ch

Figure 23-63. Exception Control Register (HETEXC1)

31–25	24
Reserved	APCNT_OVRFL_ENA
R-0	R/W-0

23–17	16
Reserved	APCNT_UNRFL_ENA
R-0	R/W-0

15–9	8
Reserved	PRGM_OVRFL_ENA
R-0	R/W-0

7–3	2	1	0
Reserved	APCNT_OVRFL_PRY	APCNT_UNRFL_PRY	PRGM_OVRFL_PRY
R-0	R/W-0	R/W-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 23-24. Exception Control Register 1 (HETEXC1) Field Descriptions

Bit	Field	Value	Description
31-17	Reserved	0	Reads return 0. Writes have no effect.
24	APCNT_OVRFL_ENA		APCNT Overflow Enable
		0	APCNT overflow exception is not enabled.
		1	Enables the APCNT overflow exception.
23-17	Reserved	0	Reads return 0. Writes have no effect.
16	APCNT_UNRFL_ENA		APCNT Underflow Enable
		0	APCNT underflow exception is not enabled.
		1	Enables the APCNT underflow exception.
15-9	Reserved	0	Reads return 0. Writes have no effect.
8	PRGM_OVRFL_ENA		Program Overflow Enable
		0	The program overflow exception is not enabled.
		1	Enables the program overflow exception.
7-3	Reserved	0	Reads return 0. Writes have no effect.
2	APCNT_OVRFL_PRY		APCNT Overflow Exception Interrupt Priority
		0	Exception priority level 2.
		1	Exception priority level 1.
1	APCNT_UNRFL_PRY		APCNT Underflow Exception Interrupt Priority
		0	Exception priority level 2.
		1	Exception priority level 1.
0	PRGM_OVRFL_PRY		ProgramOverflow Exception Interrupt Priority
		0	Exception priority level 2.
		1	Exception priority level 1.

23.4.9 Exception Control Register 2 (HETEXC2)

N2HET1: offset = FFF7 B820h; **N2HET2:** offset = FFF7 B920h

Figure 23-64. Exception Control Register 2 (HETEXC2)

31 - 16
Reserved
R-0

15 - 9	8
Reserved	DEBUG_STATUS_FLAG
R-0	R/WC-0

7 - 3	2	1	0
Reserved	APCNT_OVRFL_FLAG	APCNT_UNRFL_FLAG	PRGM_OVRFL_FLAG
R-0	R/W1C-0	R/W1C-0	R/W1C-0

LEGEND: R/W = Read/Write; R = Read only; W1C = Write 1 to clear; -*n* = value after reset

Table 23-25. Exception Control Register 2 (HETEXC2) Field Descriptions

Bit	Field	Value	Description
31-9	Reserved	0	Reads return 0. Writes have no effect.
8	DEBUG_STATUS_FLAG		Debug Status Flag. This flag is set when N2HET has stopped at a breakpoint. Also generates a debug request to halt the ARM CPU.
		0	Read: N2HET is either running, or stopped, flag cleared but not yet restarted. Write: No effect.
		1	Read: N2HET is stopped at a breakpoint. Write: Clears the bit. To restart N2HET clear this bit and then restart the ARM CPU. The N2HET and ARM CPU will start synchronously.
7-3	Reserved	0	Reads return 0. Writes have no effect.
2	APCNT_OVRFL_FLAG		APCNT Overflow Flag
		0	Read: Exception has not occurred since the flag was cleared. Write: No effect.
		1	Read: Exception has occurred since the flag was cleared. Write: Clears the bit.
1	APCNT_UNDFL_FLAG		APCNT Underflow Flag
		0	Read: Exception has not occurred since the flag was cleared. Write: No effect.
		1	Read: Exception has occurred since the flag was cleared. Write: Clears the bit.
0	PRGM_OVERFL_FLAG		Program Overflow Flag
		0	Read: Exception has not occurred since the flag was cleared. Write: No effect.
		1	Read: Exception has occurred since the flag was cleared Write: Clears the bit.

23.4.10 Interrupt Priority Register (HETPRY)

N2HET1: offset = FFF7 B824h; **N2HET2:** offset = FFF7 B924h

Figure 23-65. Interrupt Priority Register (HETPRY)

31	16
HETPRY	
R/WP-0	

15	0
HETPRY	
R/WP-0	

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privileged mode only; -*n* = value after reset

Table 23-26. Interrupt Priority Register (HETPRY) Field Descriptions

Bit	Field	Value	Description
31-0	HETPRY[n]		HET Interrupt Priority Level Bits Used to select the priority of any of the 32 potential interrupt sources coming from N2HET instructions.
		0	Interrupt priority level 2 (low level).
		1	Interrupt priority level 1 (high level).

23.4.11 Interrupt Flag Register (HETFLG)

N2HET1: offset = FFF7 B828h; **N2HET2:** offset = FFF7 B928h

Figure 23-66. Interrupt Flag Register (HETFLG)

31	16
HETFLAG	
R/W1C-0	

15	0
HETFLAG	
R/W1C-0	

LEGEND: R/W = Read/Write; R = Read only; W1C = Write 1 to clear; -*n* = value after reset; X = Unknown

Table 23-27. Interrupt Flag Register (HETFLG) Field Descriptions

Bit	Field	Value	Description
31-0	HETFLAG[n]		Interrupt Flag Register Bits Bit x is set when an interrupt condition has occurred on one of the instructions x+0, x+32, x+64, and so on. The flag position x (in the register) is decoded from the five LSBs of the instruction address that generated the interrupt. The hardware will set the flag only if the interrupt enable bit (in the corresponding instruction) is set. The flag will be set even if bit x in the Interrupt Enable Set Register (HETINTENAS) is not enabled. Enabling bit x in HETINTENAS is required if an interrupt should be generated. Clearing the flag can be done by writing a one to the flag. Alternatively reading the corresponding Offset Index Priority Level 1 Register (HETOFF1) or Offset Index Priority Level 2 Register (HETOFF2) will automatically clear the flag.
		0	Read: No N2HET instruction with an interrupt has been reached since the flag was cleared. Write: No effect.
		1	Read: A N2HET instruction with an interrupt has been reached since the flag was cleared. Write: Clears the bit.

23.4.12 AND Share Control Register (HETAND)

N2HET1: offset = FFF7 B82Ch; **N2HET2:** offset = FFF7 B92Ch

Figure 23-67. AND Share Control Register (HETAND)

31–16
Reserved
R-0

15	14	13	12	11	10	9	8
AND SHARE31/30	AND SHARE29/28	AND SHARE27/26	AND SHARE25/24	AND SHARE23/22	AND SHARE21/20	AND SHARE19/18	AND SHARE17/16
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

7	6	5	4	3	2	1	0
AND SHARE15/14	AND SHARE13/12	AND SHARE11/10	AND SHARE9/8	AND SHARE7/6	AND SHARE5/4	AND SHARE3/2	AND SHARE1/0
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 23-28. AND Share Control Register (HETAND) Field Descriptions

Bit	Field	Value	Description
31-16	Reserved	0	Reads return 0. Writes have no effect.
15-0	ANDSHARE n+1 / n		AND Share Enable Enable the AND sharing of the same pin for two HR structures. For example, if bit ANDSHARE1/0 is set, the pin HET[0] will then be commanded by a logical AND of both HR structures 0 and 1. **Note:** If HR AND SHARE bits are used, pins not connected to HR structures (the odd number pin in each pair) can be accessed as general inputs/outputs.
		0	HR Output of HET[n+1] and HET[n] are not AND shared.
		1	HR Output of HET[n+1] and HET[n] are AND shared onto pin HET[n].

23.4.13 HR Share Control Register (HETHRSH)

N2HET1: offset = FFF7 B834h; **N2HET2:** offset = FFF7 B934h

Figure 23-68. HR Share Control Register (HETHRSH)

31–16
Reserved
R-0

15	14	13	12	11	10	9	8
HR SHARE31/30	HR SHARE29/28	HR SHARE27/26	HR SHARE25/24	HR SHARE23/22	HR SHARE21/20	HR SHARE19/18	HR SHARE17/16
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

7	6	5	4	3	2	1	0
HR SHARE15/14	HR SHARE13/12	HR SHARE11/10	HR SHARE9/8	HR SHARE7/6	HR SHARE5/4	HR SHARE3/2	HR SHARE1/0
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 23-29. HR Share Control Register (HETHRSH) Field Descriptions

Bit	Field	Value	Description
31-16	Reserved	0	Reads return 0. Writes have no effect.
15-0	HRSHARE n+1 / n		HR Share Bits Enables the share of the same pin for two HR structures. For example, if bit HRSHARE1/0 is set, the pin HET[0] will then be connected to both HR input structures 0 and 1. **Note:** If HR share bits are used, pins not connected to HR structures (the odd number pin in each pair) can be accessed as general inputs/outputs.
		0	HR Input of HET[n+1] and HET[n] are not shared.
		1	HR Input of HET[n+1] and HET[n] are shared; both measure pin HET[n].

23.4.14 XOR Share Control Register (HETXOR)

N2HET1: offset = FFF7 B838h; **N2HET2:** offset = FFF7 B938h

Figure 23-69. XOR Share Control Register (HETXOR)

31–16
Reserved
R-0

15	14	13	12	11	10	9	8
XOR SHARE31/30	XOR SHARE29/28	XOR SHARE27/26	XOR SHARE25/24	XOR SHARE23/22	XOR SHARE21/20	XOR SHARE19/18	XOR SHARE17/16
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

7	6	5	4	3	2	1	0
XOR SHARE15/14	XOR SHARE13/12	XOR SHARE11/10	XOR SHARE9/8	XOR SHARE7/6	XOR SHARE5/4	XOR SHARE3/2	XOR SHARE1/0
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 23-30. XOR Share Control Register (HETXOR) Field Descriptions

Bit	Field	Value	Description
31-16	Reserved	0	Reads return 0. Writes have no effect.
15-0	XORSHARE n+1 / n		XOR Share Enable Enable the XOR-share of the same pin for two output HR structures. For example, if bit XORSHARE1/0 is set, the pin HET[0] will then be commanded by a logical XOR of both HR structures 0 and 1. **Note:** If XOR share bits are used, pins not connected to HR structures (the odd number pin in each pair) can be accessed as general inputs/outputs.
		0	HR Output of HET[n+1] and HET[n] are not XOR shared.
		1	HR Output of HET[n+1] and HET[n] are XOR shared onto pin HET[n].

23.4.15 Request Enable Set Register (HETREQENS)

N2HET1: offset = FFF7 B83Ch; **N2HET2:** offset = FFF7 B93Ch

Figure 23-70. Request Enable Set Register (HETREQENS)

31–8
Reserved
R-0

7	6	5	4	3	2	1	0
REQENA7	REQENA6	REQENA5	REQENA4	REQENA3	REQENA2	REQENA1	REQENA0
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 23-31. Request Enable Set Register (HETREQENS) Field Descriptions

Bit	Field	Value	Description
31-8	Reserved	0	Reads return 0. Writes have no effect.
7-0	REQENAn		Request Enable Bits
		0	Read: Returns the information that request line n is disabled. Write: Writing a 0 has no effect.
		1	Read: Returns the information that request line n is enabled. Write: Writing a 1 to bit n enables the N2HET request line n. **Note:** The request line can trigger a DMA control packet (DMA channel), an HTU double control packet (DCP) or both simultaneously. The HETREQDS register determines to which module(s) the N2HET request line n is assigned. **Note:** A disabled request line does not memorize old requests. So there are no pending requests to service after enabling request line n.

23.4.16 Request Enable Clear Register (HETREQENC)

N2HET1: offset = FFF7 B840h; **N2HET2:** offset = FFF7 B940h

Figure 23-71. Request Enable Clear Register (HETREQENC)

31–8
Reserved
R-0

7	6	5	4	3	2	1	0
REQDIS7	REQDIS6	REQDIS5	REQDIS4	REQDIS3	REQDIS2	REQDIS1	REQDIS0
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 23-32. Request Enable Clear Register (HETREQENC) Field Descriptions

Bit	Field	Value	Description
31-8	Reserved	0	Reads return 0. Writes have no effect.
7-0	REQDISn		Request Disable Bits
		0	Read: Returns the information that request line n is disabled. Write: Writing a 0 has no effect.
		1	Read: Returns the information that request line n is enabled. Write: Writing a 1 to bit n disables the N2HET request line n.

23.4.17 Request Destination Select Register (HETREQDS)

N2HET1: offset = FFF7 B844h; **N2HET2:** offset = FFF7 B944h

Figure 23-72. Request Destination Select Register (HETREQDS) [offset = FFF7 B844h]

31	24	23	22	21	20	19	18	17	16
Reserved		TDBS7	TDBS6	TDBS5	TDBS4	TDBS3	TDBS2	TDBS1	TDBS0
R-0		R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

15	8	7	6	5	4	3	2	1	0
Reserved		TDS7	TDS6	TDS5	TDS4	TDS3	TDS2	TDS1	TDS0
R-0		R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 23-33. Request Destination Select Register (HETREQDS) Field Descriptions

Bit	Field	Value	Description
31-24	Reserved	0	Reads return 0. Writes have no effect.
23-16	TDBSn		HTU, DMA or Both Select Bits
		0	N2HET request line n is assigned to the module specified by TDS bit n.
		1	N2HET request line n is assigned to both DMA and HTU. TDS bit n is ignored in this case.
15-8	Reserved	0	Reads return 0. Writes have no effect.
7-0	TDSn		HTU or DMA Select Bits **Note:** It must be ensured in the N2HET program, that one request line is triggered by only one N2HET instruction.
		0	N2HET request line n is assigned to HTU (TDBS bit n is zero).
		1	N2HET request line n is assigned to DMA (TDBS bit n is zero).

NOTE: Please refer to the device data sheet how each of the 8 N2HET request lines are connected to these modules. See also Section 23.2.9.

23.4.18 NHET Direction Register (HETDIR)

N2HET1: offset = FFF7 B84Ch; **N2HET2:** offset = FFF7 B94Ch

Figure 23-73. N2HET Direction Register (HETDIR)

31	16
HETDIR	
R/W-0	

15	0
HETDIR	
R/W-0	

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 23-34. N2HET Direction Register (HETDIR) Field Descriptions

Bit	Field	Value	Description
31-0	HETDIR[n]		Data direction of NHET pins
		0	Pin HET[n] is an input (and its output buffer is tristated).
		1	Pin HET[n] is an output.

NOTE: Table 23-9 shows how the register bits of DIR, PULDIS and PULSEL are affecting the N2HET pins.

23.4.19 N2HET Data Input Register (HETDIN)

N2HET1: offset = FFF7 B850h; **N2HET2:** offset = FFF7 B950h

Figure 23-74. N2HET Data Input Register (HETDIN)

31 — 16
HETDIN
R-x

15 — 0
HETDIN
R-x

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset;

Table 23-35. N2HET Data Input Register (HETDIN) Field Descriptions

Bit	Field	Value	Description
31-0	HETDIN[n]		Data input. This bit displays the logic state of the pin.
		0	Pin HET[n] is at logic low (0).
		1	Pin HET[n] is at logic high (1).

23.4.20 N2HET Data Output Register (HETDOUT)

N2HET1: offset = FFF7 B854h; **N2HET2:** offset = FFF7 B954h

Figure 23-75. N2HET Data Output Register (HETDOUT)

31 — 16
HETDOUT
R/W-0

15 — 0
HETDOUT
R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 23-36. N2HET Data Output Register (HETDOUT) Field Descriptions

Bit	Field	Value	Description
31-0	HETDOUT[n]		Data out write. Writes to this bit will only take effect when the pin is configured as an output. The current logic state of the pin will be displayed by this bit even when the pin state is changed by writing to HETDSET or HETDCLR.
		0	Pin HET[n] is at logic low (0).
		1	Pin HET[n] is at logic high (1) if the HETPDR[n] bit = 0 or the output is in high-impedance state if the HETPDR[n] bit = 1.

23.4.21 NHET Data Set Register (HETDSET)

N2HET1: offset = FFF7 B858h; **N2HET2:** offset = FFF7 B958h

Figure 23-76. N2HET Data Set Register (HETDSET)

31	16
HETDSET	
R/WS-0	

15	0
HETDSET	
R/WS-0	

LEGEND: R/W = Read/Write; R = Read only; S = Set; -*n* = value after reset

Table 23-37. N2HET Data Set Register (HETDSET) Field Descriptions

Bit	Field	Value	Description
31-0	HETDSET[n]		This register allows bits of HETDOUT to be set while avoiding the pitfalls of a read-modify-write sequence in a multitasking environment. Bits written as a logic 1 set the same bit in the HETDOUT register; while bits written as logic 0 leave the same bit in HETDOUT unchanged. Reads from this address return the value of the HETDOUT register.
		0	Write: HETDOUT[n] is unchanged.
		1	Write: HETDOUT[n] is set.

23.4.22 N2HET Data Clear Register (HETDCLR)

N2HET1: offset = FFF7 B85Ch; **N2HET2:** offset = FFF7 B95Ch

Figure 23-77. N2HET Data Clear Register (HETDCLR)

31	16
HETDCLR	
R/WC-0	

15	0
HETDCLR	
R/WC-0	

LEGEND: R/W = Read/Write; R = Read only; C = Clear; -*n* = value after reset

Table 23-38. N2HET Data Clear Register (HETDCLR) Field Descriptions

Bit	Field	Value	Description
31-0	HETDCLR[n]		This register allows bits of HETDOUT to be cleared while avoiding the pitfalls of a read-modify-write sequence in a multitasking environment. Bits written as a logic 1 clear the same bit in the HETDOUT register; while bits written as logic 0 leave the same bit in HETDOUT unchanged. Reads from this address return the value of the HETDOUT register.
		0	Write: HETDOUT[n] is unchanged.
		1	Write: HETDOUT[n] is cleared.

23.4.23 N2HET Open Drain Register (HETPDR)

Values in this register enable or disable the open drain capability of the data pins.
N2HET1: offset = FFF7 B860h; **N2HET2:** offset = FFF7 B960h

Figure 23-78. N2HET Open Drain Register (HETPDR)

31	16
HETPDR	
R/W-0	

15	0
HETPDR	
R/W-0	

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 23-39. N2HET Open Drain Register (HETPDR) Field Descriptions

Bit	Field	Value	Description
31-0	HETPDR[n]		Open drain control for HET[n] pins
		0	The pin is configured in push/pull mode.
		1	The pin is configured in open drain mode. The HETDOUT register controls the state of the output buffer: HETDOUT[n] = 0 The output buffer of pin HET[n] is driven low. HETDOUT[n] = 1 The output buffer of pin HET[n] is tristated.

23.4.24 N2HET Pull Disable Register (HETPULDIS)

Values in this register enable or disable the pull-up/-down functionality of the pins.
N2HET1: offset = FFF7 B864h; **N2HET2:** offset = FFF7 B964h

Figure 23-79. N2HET Pull Disable Register (HETPULDIS)

31	16
HETPULDIS	
R/W-n	

15	0
HETPULDIS	
R/W-n	

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset; n is device dependent, see device specific data sheet

Table 23-40. N2HET Pull Disable Register (HETPULDIS) Field Descriptions

Bit	Field	Value	Description
31-0	HETPULDIS[n]		Pull disable for N2HET pins
		0	The pull functionality is enabled on pin HET[n].
		1	The pull functionality is disabled on pin HET[n].

NOTE: See device data sheet for which pins provide programmable pullups/pulldowns.

Table 23-9 shows how the register bits of HETDIR, HETPULDIS, and HETPSL are affecting the N2HET pins.

23.4.25 N2HET Pull Select Register (HETPSL)

Values in this register select the pull-up or pull-down functionality of the pins.
N2HET1: offset = FFF7 B868h; **N2HET2:** offset = FFF7 B968h

Figure 23-80. N2HET Pull Select Register (HETPSL)

31	16
HETPSL	
R/W-0	

15	0
HETPSL	
R/W-0	

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 23-41. N2HET Pull Select Register (HETPSL) Field Descriptions

Bit	Field	Value	Description
31-0	HETPSL[n]		Pull select for NHET pins
		0	The pull down functionality is enabled if corresponding bit in HETPULDIS is 0.
		1	The pull up functionality is enabled if corresponding bit in HETPULDIS is 0.

NOTE: See device data sheet for which pins provide programmable pullups/pulldowns.

Table 23-9 shows how the register bits of HETDIR, HETPULDIS and HETPSL are affecting the N2HET pins.

The information of this register is also used to define the pin states after a parity error:

After a parity error all N2HET pins, which are

1. Defined as output pins in the HETDIR register
2. Not defined as open drain pins (with the HETPDR register)
3. Selected with the HETPPR register, will remain outputs, but automatically change their levels in the following way:
   - If the HETPSL register specifies 0 for the pin, it will switch to low level.
   - If the HETPSL register specifies 1 for the pin, it will switch to high level.

This behavior is independent of the value, which register HETPULDIS specifies for the corresponding pin.

23.4.26 Parity Control Register (HETPCR)

N2HET1: offset = FFF7 B874h; **N2HET2:** offset = FFF7 B974h

Figure 23-81. Parity Control Register (HETPCR)

31	16
Reserved	
R-0	

15 9	8	7 4	3 0
Reserved	TEST	Reserved	PARITY_ENA
R-0	R/WP-0	R-0	R/WP-5h

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privileged mode only; -*n* = value after reset

Table 23-42. Parity Control Register (HETPCR) Field Descriptions

Bit	Field	Value	Description
31-9	Reserved	0	Reads return 0. Writes have no effect.
8	TEST		Test Bit. When this bit is set, the parity bits are mapped into the peripheral RAM frame to make them accessible by the CPU.
		0	Read: Parity bits are not memory mapped. Write: Disable mapping.
		1	Read: Parity bits are memory mapped. Write: Enable mapping.
7-4	Reserved	0	Reads return 0. Writes have no effect.
3-0	PARITY_ENA		Enable/disable parity checking. This bit field enables or disables the parity check on read operations and the parity calculation on write operations. If parity checking is enabled and a parity error is detected the N2HET_UERR signal is activated.
		5h	Read: Parity check is disabled. Write: Disable checking.
		Others	Read: Parity check is enabled. Write: Enable checking.

NOTE: It is recommended to write Ah to enable error detection, to guard against soft errors flipping PARITY_ENA to a disable state.

23.4.27 Parity Address Register (HETPAR)

N2HET1: offset = FFF7 B878h; **N2HET2:** offset = FFF7 B978h

Figure 23-82. Parity Address Register (HETPAR)

31–16
Reserved
R-0

15–13	12–2	1–0
Reserved	PAOFF	Reserved
R-0	R-X	R-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset; X = Value unchanged after reset

Table 23-43. Parity Address Register (HETPAR) Field Descriptions

Bit	Field	Value	Description
31-13	Reserved	0	Reads return 0. Writes have no effect.
12-2	PAOFF		Parity Error Address Offset. This register holds the offset address of the first parity error, which is detected in N2HET RAM. This error address is frozen from being updated until it is read by the CPU. During emulation mode, this address is frozen even when read. In case of a N2HET RAM parity error, PAOFF will contain the offset address of the erroneous 32-bit N2HET RAM field counted from the beginning of the N2HET RAM. Examples: The 32-bit program field of instruction 0 will return 0, the 32-bit control field of instruction 0 will return 1, ..., the 32-bit control field of instruction 1 will return 5, and so on. Read: Returns the offset address of the erroneous 32-bit word in bytes from the beginning of the N2HET RAM. Write: Writes have no effect.
1-0	Reserved	0	Reads return 0. Writes have no effect.

NOTE: The Parity Error Address Register will not be reset, neither by PORRST nor by any other reset source.

23.4.28 Parity Pin Register (HETPPR)

N2HET1: offset = FFF7 B87Ch; **N2HET2:** offset = FFF7 B97Ch

Figure 23-83. Parity Pin Register (HETPPR)

31	16
HETPPR	
R/W-0	

15	0
HETPPR	
R/W-0	

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 23-44. Parity Pin Register (HETPPR) Field Descriptions

Bit	Field	Value	Description
31-0	HETPPR[n]		NHET Parity Pin Select Bits. Allows HET[n] pins to be configured to drive to a known state when an N2HET parity error is detected.
		0	Pin HET[n] is not affected by the detection of an N2HET parity error.
		1	Pin HET[n] is driven to a known state when an N2HET parity error is detected. The known state is a function of bits HETDIR[n], HETPSL[n], HETPDR[n] as described in Table 23-45 (this state is also independent of HETPULDIS[n]).

Table 23-45. Known State on Parity Error

HETDIR[n]	HETPDR[n]	HETPSL[n]	Known State on Parity Error
0	x	x	High Impedance
1	0	0	Drive Logic 0
1	0	1	Drive Logic 1
1	1	x	High Impedance

23.4.29 Suppression Filter Preload Register (HETSFPRLD)

N2HET1: offset = FFF7 B880h; **N2HET2:** offset = FFF7 B980h

Figure 23-84. Suppression Filter Preload Register (HETSFPRLD)

31–18	17–16
Reserved	CCDIV
R-0	R/W-0

15–10	9–0
Reserved	CPRLD
R-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 23-46. Suppression Filter Preload Register (HETSFPRLD) Field Descriptions

Bit	Field	Value	Description
31-18	Reserved	0	Reads return 0. Writes have no effect.
17-16	CCDIV		Counter Clock Divider
			CCDIV determines the ratio between the counter clock and VCLK2.
		0	CCLK = VCLK2
		1h	CCLK = VCLK2 / 2
		2h	CCLK = VCLK2 / 3
		3h	CCLK = VCLK2 / 4
15-10	Reserved	0	Reads return 0. Writes have no effect.
9-0	CPRLD		Counter Preload Value
			CPRLD contains the preload value for the counter clock.

23.4.30 Suppression Filter Enable Register (HETSFENA)

N2HET1: offset = FFF7 B884h; **N2HET2:** offset = FFF7 B984h

Figure 23-85. Suppression Filter Enable Register (HETSFENA)

31–16
HETSFENA
R/W-0

15–0
HETSFENA
R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 23-47. Suppression Filter Enable Register (HETSFENA) Field Descriptions

Bit	Field	Value	Description
31-0	HETSFENA[n]		Suppression Filter Enable Bits
			Note: If the pin is configured as an output by the N2HET Direction Register (HETDIR), the filter is automatically disabled independent on the bit in HETSFENA.
		0	The input noise suppression filter for pin HET[n] is disabled.
		1	The input noise suppression filter for pin HET[n] is enabled.

23.4.31 Loop Back Pair Select Register (HETLBPSEL)

Refer to Section 23.2.5.7 for a description of loopback test functions.

N2HET1: offset = FFF7 B88Ch; **N2HET2:** offset = FFF7 B98Ch

Figure 23-86. Loop Back Pair Select Register (HETLBPSEL)

31	30	29	28	27	26	25	24
LBPTYPE31/30	LBPTYPE29/28	LBPTYPE27/26	LBPTYPE25/24	LBPTYPE23/22	LBPTYPE21/20	LBPTYPE19/18	LBPTYPE17/16
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

23	22	21	20	19	18	17	16
LBPTYPE15/14	LBPTYPE13/12	LBPTYPE11/10	LBPTYPE9/8	LBPTYPE7/6	LBPTYPE5/4	LBPTYPE3/2	LBPTYPE1/0
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

15	14	13	12	11	10	9	8
LBPSEL31/30	LBPSEL29/28	LBPSEL27/26	LBPSEL25/24	LBPSEL23/22	LBPSEL21/20	LBPSEL19/18	LBPSEL17/16
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

7	6	5	4	3	2	1	0
LBPSEL15/14	LBPSEL13/12	LBPSEL11/10	LBPSEL9/8	LBPSEL7/6	LBPSEL5/4	LBPSEL3/2	LBPSEL1/0
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 23-48. Loop Back Pair Select Register (HETLBPSEL) Field Descriptions

Bit	Field	Value	Description
31-16	LBPTYPE n+1 / n		Loop Back Pair Type Select Bits These bits are valid only when Loopback mode is enabled (HETLBPDIR[19:16] = 1010).
		0	Digital loopback is selected for HR structures on pins HET[n+1] and HET[n].
		1	Analog loopback is selected for HR structures on pins HET[n+1] and HET[n].
15-0	LBPSEL n+1 / n		Loop Back Pair Select Bits These bits are valid only when Loopback mode is enabled (HETLBPDIR[19:16] = 1010). If bit x is set, the HR structures on pins HET[n+1] and HET[n] are connected in a loop back mode. The direction is given by LBPDIR n+1/n and type is selected by LBPTYPE n+1/n. The pin which is not driven by the N2HET pin actions can still be used as normal GIO pin.

23.4.32 Loop Back Pair Direction Register (HETLBPDIR)

Refer to Section 23.2.5.7 for a description of loopback test functions.

N2HET1: offset = FFF7 B890h; **N2HET2:** offset = FFF7 B990h

Figure 23-87. Loop Back Pair Direction Register (HETLBPDIR)

31-20	19-16
Reserved	LBPTSTENA
R-0	R/WP-5h

15	14	13	12	11	10	9	8
LBPDIR31/30	LBPDIR29/28	LBPDIR27/26	LBPDIR25/24	LBPDIR23/22	LBPDIR21/20	LBPDIR19/18	LBPDIR17/16
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

7	6	5	4	3	2	1	0
LBPDIR15/14	LBPDIR13/12	LBPDIR11/10	LBPDIR9/8	LBPDIR7/6	LBPDIR5/4	LBPDIR3/2	LBPDIR1/0
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privileged mode only; -*n* = value after reset

Table 23-49. Loop Back Pair Direction Register (HETLBPDIR) Field Descriptions

Bit	Field	Value	Description
31-20	Reserved	0	Reads return 0. Writes have no effect.
19-16	LBPTSTENA		Loopback Test Enable Key
		5h	Loopback Test is disabled.
		Ah	Loopback Test is enabled.
		Others	Loopback Test is disabled.
15-0	LBPDIR n+1 / n		Loop Back Pair Direction Bits
		0	The HR structures on pins HET[n+1] and HET[n] are internally connected with HET[n] as input and HET[n+1] as output.
		1	The HR structures on pins HET[n+1] and HET[n] connected with HET[n] as output and HET[n+1] as input.

NOTE: The loop back direction can be selected independent on the HETDIR register setting.

23.4.33 N2HET Pin Disable Register (HETPINDIS)

N2HET1: offset = FFF7 B894h; **N2HET2:** offset = FFF7 B994h

Figure 23-88. N2HET Pin Disable Register (HETPINDIS)

31 – 16
HETPINDIS
R/W-0

15 – 0
HETPINDIS
R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 23-50. NHET Pin Disable Register (HETPINDIS) Field Descriptions

Bit	Field	Value	Description
31-0	HETPINDIS[n]		N2HET Pin Disable Bits
		0	Logic low: No affect on the output buffer enable of the pin (is controlled by the value of the HETDIR[n] bit).
		1	Logic high: Output buffer of the pin is enabled if pin nDIS = 1, HET_PIN_ENA = 1, and HETDIR = 1; or disabled if nDIS = 0, HETDIR = 0, or HET_PIN_ENA = 0.

23.5 HWAG Registers

Table 23-51 lists the HWAG registers.

Table 23-51. HWAG Registers

Offset	Acronym	Register Description	Section
9Ch	HWAPINSEL	HWAG Pin Select Register	Section 23.5.1
A0h	HWAGCR0	HWAG Global Control Register 0	Section 23.5.2
A4h	HWAGCR1	HWAG Global Control Register 1	Section 23.5.3
A8h	HWAGCR2	HWAG Global Control Register 2	Section 23.5.4
ACh	HWAENASET	HWAG Interrupt Enable Set Register	Section 23.5.5
B0h	HWAENACLR	HWAG Interrupt Enable Clear Register	Section 23.5.6
B4h	HWALVLSET	HWAG Interrupt Level Set Register	Section 23.5.7
B8h	HWALVLCLR	HWAG Interrupt Level Clear Register	Section 23.5.8
BCh	HWAFLG	HWAG Interrupt Flag Register	Section 23.5.9
C0h	HWAOFF0	HWAG Interrupt Offset Register 1	Section 23.5.10
C4h	HWAOFF1	HWAG Interrupt Offset Register 2	Section 23.5.11
C8h	HWAACNT	HWAG Angle Value Register	Section 23.5.12
CCh	HWAPCNT1	HWAG Previous Tooth Period Value Register	Section 23.5.13
D0h	HWAPCNT	HWAG Current Tooth Period Value Register	Section 23.5.14
D4h	HWASTWD	HWAG Step Width Register	Section 23.5.15
D8h	HWATHNB	HWAG Teeth Number Register	Section 23.5.16
DCh	HWATHVL	HWAG Current Teeth Number Register	Section 23.5.17
E0h	HWAFIL	HWAG Filter Register	Section 23.5.18
E8h	HWAFIL2	HWAG Filter Register 2	Section 23.5.19
F0h	HWAANGI	HWAG Angle Increment Register	Section 23.5.20

23.5.1 HWAG Pin Select Register (HWAPINSEL)

Figure 23-89. HWAG Pin Select Register (HWAPINSEL)

31	16
Reserved	
R-0	

15	5	4	0
Reserved		PINSEL	
R-0		R/W-2h	

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 23-52. HWAG Pin Select Register (HWAPINSEL) Field Descriptions

Bit	Field	Value	Description
31-5	Reserved	0	Reads return 0. Writes have no effect.
4-0	PINSEL		HWAG Pin Select. Selects from which NHET pin input buffer the HWAG toothed-wheel signal is derived.
		0	Read: Pin HET[0] is selected. Write: Selects pin HET[0].
		1h	Read: Pin HET[1] is selected Write: Selects pin HET[1].
		2h	Read: Pin HET[2] is selected Write: Selects pin HET[2]. Default after reset for backwards compatibility
		:	:
		1Fh	Read: Pin HET[31] selected Write: Selects pin HET[31].

23.5.2 HWAG Global Control Register 0 (HWAGCR0)

Figure 23-90. HWAG Global Control Register 0 (HWAGCR0)

31	16
Reserved	
R-0	

15	1	0
Reserved		RESET
R-0		R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 23-53. HWAG Global Control Register 0 (HWAGCR0) Field Descriptions

Bit	Field	Value	Description
31-1	Reserved	0	Reads return 0. Writes have no effect.
0	RESET		HWAG Module Reset.
		0	HWAG module is reset.
		1	HWAG module is not in reset.

23.5.3 HWAG Global Control Register 1 (HWAGCR1)

Figure 23-91. HWAG Global Control Register 1 (HWAGCR1)

31	16
Reserved	
R-0	

15	1	0
Reserved		PPWN
R-0		R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 23-54. HWAG Global Control Register 1 (HWAGCR1) Field Descriptions

Bit	Field	Value	Description
31-1	Reserved	0	Reads return 0. Writes have no effect.
0	PPWN	0	HWAG Module Power Down. This bit is implemented for legacy purposes, but has no functionality, however the HWAG module power down is controlled by the NHET power down. The HWAG cannot be powered down separately.

23.5.4 HWAG Global Control Register 2 (HWAGCR2)

Figure 23-92. HWAG Global Control Register 2 (HWAGCR2)

31 25	24	23 18	17	16
Reserved	ARST	Reserved	TED	CRI
R-0	R/W-0	R-0	R/W-0	R/W-0

15 9	8	7 1	0
Reserved	FIL	Reserved	STRT
R-0	R/W-0	R-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 23-55. HWAG Global Control Register 2 (HWAGCR2) Field Descriptions

Bit	Field	Value	Description
31-25	Reserved	0	Reads return 0. Writes have no effect.
24	ARST		Angle Reset. This bit is used by the HWAG to validate the singularity when the hardware criteria is not used. The bit is cleared when the HWAG angle value register (HWAACNT) is cleared by the HWAG, when the last tooth edge occurs. If this bit is not set before the tooth edge during an singularity tooth, the HWAG generates an interruption “singularity not found”, if the interrupt is enabled.
		0	Do not reset ACNT once it reaches the angle zero point.
		1	Reset ACNT once it reaches the angle zero point.
23-18	Reserved	0	Reads return 0. Writes have no effect.
17	TED		Tooth Edge. This bit is used to select which edge of the tooth wheel must be considered as active.
		0	Falling edge
		1	Rising edge
16	CRI		Criteria enable. This bits is used to control whether the criteria are applied. You could set your own criteria filter by disabling the hardwired criteria.
		0	Criteria is disabled.
		1	Criteria is enabled.
15-9	Reserved	0	Reads return 0. Writes have no effect.
8	FIL		Input Filter Enable. This bit is used to enable the toothed wheel input filter.
		0	Filter is disabled.
		1	Filter is enabled.
7-1	Reserved	0	Reads return 0. Writes have no effect.
0	STRT		Start bit. Put the HWAG into run time. Allows the HWAG to start counting ACNT, TCNT and criteria mechanism (if set). The HWAG starts at the next active edge from the toothed wheel, once set. If the start bit is cleared to 0, the HWAG is stopped immediately.
		0	Do not start counting.
		1	Start counting.

23.5.5 HWAG Interrupt Enable Set Register (HWAENASET)

Figure 23-93. HWAG Interrupt Enable Set Register (HWAENASET)

31 — 8
Reserved
R-0

7	6	5	4	3	2	1	0
SETINTENA7	SETINTENA6	SETINTENA5	SETINTENA4	SETINTENA3	SETINTENA2	SETINTENA1	SETINTENA0
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 23-56. HWAG Interrupt Enable Set Register (HWAENASET) Field Descriptions

Bit	Field	Value	Description
31-8	Reserved	0	Reads return 0. Writes have no effect.
7-0	SETINTENA[n]		Enable interrupt. See Table 23-57.
		0	Read: Corresponding interrupt is not enabled. Write: No effect.
		1	Read: Corresponding interrupt is enabled. Write: Enable corresponding interrupt.

Table 23-57. HWAG Interrupts

Bit	Interrupt
0	Overflow period
1	Singularity not found
2	Tooth interrupt
3	ACNT overflow
4	PCNT(n) > 2 x PCNT (n-1) during normal tooth
5	Bad active edge tooth
6	Gap flag
7	Angle increment overflow

23.5.6 HWAG Interrupt Enable Clear Register (HWAENACLR)

Figure 23-94. HWAG Interrupt Enable Clear Register (HWAENACLR)

31-8
Reserved
R-0

7	6	5	4	3	2	1	0
CLRINTENA7	CLRINTENA6	CLRINTENA5	CLRINTENA4	CLRINTENA3	CLRINTENA2	CLRINTENA1	CLRINTENA0
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 23-58. HWAG Interrupt Enable Clear Register (HWAENACLR) Field Descriptions

Bit	Field	Value	Description
31-8	Reserved	0	Reads return 0. Writes have no effect.
7-0	CLRINTENA[n]		Disable interrupt. See Table 23-57.
		0	Read: Corresponding interrupt is not enabled. Write: No effect.
		1	Read: Corresponding interrupt is enabled. Write: Disable corresponding interrupt.

23.5.7 HWAG Interrupt Level Set Register (HWALVLSET)

Figure 23-95. HWAG Interrupt Level Set Register (HWALVLSET)

31	8
Reserved	
R-0	

7	6	5	4	3	2	1	0
SETINTLVL7	SETINTLVL6	SETINTLVL5	SETINTLVL4	SETINTLVL3	SETINTLVL2	SETINTLVL1	SETINTLVL0
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 23-59. HWAG Interrupt Level Set Register (HWALVLSET) Field Descriptions

Bit	Field	Value	Description
31-8	Reserved	0	Reads return 0. Writes have no effect.
7-0	SETINTLVL[n]		Set Interrupt Level. See Table 23-57.
		0	Read: Low-priority interrupt. Write: No effect.
		1	Read: High-priority interrupt. Write: Set interrupt priority to high.

23.5.8 HWAG Interrupt Level Clear Register (HWALVLCLR)

Figure 23-96. HWAG Interrupt Level Clear Register (HWALVLCLR)

31	8
Reserved	
R-0	

7	6	5	4	3	2	1	0
CLRINTLVL7	CLRINTLVL6	CLRINTLVL5	CLRINTLVL4	CLRINTLVL3	CLRINTLVL2	CLRINTLVL1	CLRINTLVL0
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 23-60. HWAG Interrupt Level Clear Register (HWALVLCLR) Field Descriptions

Bit	Field	Value	Description
31-8	Reserved	0	Reads return 0. Writes have no effect.
7-0	CLRINTLVL[n]		Clear Interrupt Level. See Table 23-57.
		0	Read: Low-priority interrupt. Write: No effect.
		1	Read: High-priority interrupt. Write: Set interrupt priority to low.

23.5.9 HWAG Interrupt Flag Register (HWAFLG)

Figure 23-97. HWAG Interrupt Flag Register (HWAFLG)

31-8
Reserved
R-0

7	6	5	4	3	2	1	0
INTFLG7	INTFLG6	INTFLG5	INTFLG4	INTFLG3	INTFLG2	INTFLG1	INTFLG0
R/W1C-0	R/W1C-0	R/W1C-0	R/W1C-0	R/W1C-0	R/W1C-0	R/W1C-0	R/W1C-0

LEGEND: R/W = Read/Write; R = Read only; W1C = Write 1 to clear; -*n* = value after reset

Table 23-61. HWAG Interrupt Flag Register (HWAFLG) Field Descriptions

Bit	Field	Value	Description
31-8	Reserved	0	Reads return 0. Writes have no effect.
7-0	INTFLG[n]		Interrupt Flag. These bit are set when an interrupt condition has occurred inside the HWAG. The interrupt is sent to the CPU if, and only if, the corresponding enable bit is set. HWAFLG is cleared by either reading the HWAOFF0 or HWAOFF1 register (if the corresponding bit is set) or by writing 1 to the bit. If HWAFLG is 1 but the corresponding interrupt is not enabled then it will not generate an interrupt, also the OFFSET index will not be generated for that particular HWAFLG bit. So, a read of HWAOFF registers will not clear a HWAFLG bit that is not enabled. See Table 23-57.
		0	Read: No interrupt is pending. Write: No effect.
		1	Read: Interrupt is pending. Write: Clear the corresponding interrupt flag.

23.5.10 HWAG Interrupt Offset Register 0 (HWAOFF0)

This register is a read-only register and provides a numerical value that represents the pending interrupt with a high priority. The index can be used to locate the interrupt routine position in the vector table. A read to this register clears the corresponding interrupt pending bit in the HWAG interrupt flag register (HWAFLG). An interrupt pending bit in the HWAFLG register is the bit for which the corresponding interrupt enable bit is set.

During suspend mode, a read to this register does not clear the corresponding interrupt bit.

Figure 23-98. HWAG Interrupt Offset Register 0 (HWAOFF0)

31	16
Reserved	
R-0	

15	8	7	0
Reserved		OFFSET1	
R-0		R-0	

LEGEND: R = Read only; -*n* = value after reset

Table 23-62. HWAG Interrupt Offset Register 0 (HWAOFF0) Field Descriptions

Bit	Field	Value	Description
31-8	Reserved	0	Reads return 0. Writes have no effect.
7-0	OFFSET1		High-Priority Interrupt Offset. These bits give the offset for the corresponding interrupts.
		0	Phantom interrupt
		1	Overflow period
		2	Singularity not found
		3	Tooth interrupt
		4	ACNT overflow
		5	PCNT(n) > 2 × PCNT (n-1) during normal tooth
		6	Bad active edge tooth
		7	Gap flag
		8	Angle increment overflow

23.5.11 HWAG Interrupt Offset Register 1 (HWAOFF1)

This register is a read-only register and provides a numerical value that represents the pending interrupt with a low priority. The index can be used to locate the interrupt routine position in the vector table. A read to this register clears the corresponding interrupt pending bit in the HWAG interrupt flag register (HWAFLG). An interrupt pending bit in the HWAFLG register is the bit for which the corresponding interrupt enable bit is set.

During suspend mode, a read to this register does not clear the corresponding interrupt bit.

Figure 23-99. HWAG Interrupt Offset Register 1 (HWAOFF1)

31	16
Reserved	
R-0	

15	8	7	0
Reserved		OFFSET2	
R-0		R-0	

LEGEND: R = Read only; -*n* = value after reset

Table 23-63. HWAG Interrupt Offset Register 1 (HWAOFF1) Field Descriptions

Bit	Field	Value	Description
31-8	Reserved	0	Reads return 0. Writes have no effect.
7-0	OFFSET2		Low-Priority Interrupt Offset.. These bits give the offset for the corresponding interrupts.
		0	Phantom interrupt
		1	Overflow period
		2	Singularity not found
		3	Tooth interrupt
		4	ACNT overflow
		5	PCNT(n) > 2 × PCNT (n-1) during normal tooth
		6	Bad active edge tooth
		7	Gap flag
		8	Angle increment overflow

23.5.12 HWAG Angle Value Register (HWAACNT)

Figure 23-100. HWAG Angle Value Register (HWAACNT)

31 24	23 16
Reserved	ACNT
R-0	R/W-0

15 0
ACNT
R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 23-64. HWAG Angle Value Register (HWAACNT) Field Descriptions

Bit	Field	Value	Description
31-24	Reserved	0	Reads return 0. Writes have no effect.
23-0	ACNT	0-FF FFFFh	Angle Value. Provides the current angle value from the toothed wheel. This is equal to step width × teeth value.

23.5.13 HWAG Previous Tooth Period Value Register (HWAPCNT1)

Figure 23-101. HWAG Previous Tooth Period Value Register (HWAPCNT1)

31 24	23 16
Reserved	PCNT(n-1)
R-0	R/W-0

15 0
PCNT(n-1)
R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 23-65. HWAG Previous Tooth Period Value Register (HWAPCNT1) Field Descriptions

Bit	Field	Value	Description
31-24	Reserved	0	Reads return 0. Writes have no effect.
23-0	PCNT(n-1)	0-FF FFFFh	Period (n-1) Value. Gives the period value of the previous tooth.

23.5.14 HWAG Current Tooth Period Value Register (HWAPCNT)

Figure 23-102. HWAG Current Tooth Period Value Register (HWAPCNT)

31 24	23 16
Reserved	PCNT(n)
R-0	R/W-0

15 0
PCNT(n)
R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 23-66. HWAG Current Tooth Period Value Register (HWAPCNT) Field Descriptions

Bit	Field	Value	Description
31-24	Reserved	0	Reads return 0. Writes have no effect.
23-0	PCNT(n)	0-FF FFFFh	Period (n) Value. Provides the current period since the beginning of the last tooth active edge seen by the HWAG (PCNT (n)). This period would not be accurate due to the fact that the PCNT counter is running at VCLK2 and that the peripheral bus is running at VCLK. Then, the value will have changed when used.

23.5.15 HWAG Step Width Register (HWASTWD)

Figure 23-103. HWAG Step Width Register (HWASTWD)

31	16
Reserved	
R-0	

15	4	3	0
Reserved		STWD	
R-0		R/W-0	

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 23-67. HWAG Step Width Register (HWASTWD) Field Descriptions

Bit	Field	Value	Description
31-4	Reserved		Reads return 0. Writes have no effect.
3-0	STWD		Step Width. Sets the step width for the tick generation, dividing the period into K steps. (131072, 65536, ..., 8, 4). The step count is decoded from the three LSBs using the following encoding:
		0h	4 ticks per period
		1h	8 ticks per period
		2h	16 ticks per period
		:	:
		Eh	65536 ticks per period
		Fh	131072 ticks per period

23.5.16 HWAG Teeth Number Register (HWATHNB)

Figure 23-104. HWAG Teeth Number Register (HWATHNB)

31 … 16
Reserved
R-0

15 … 8	7 … 0
Reserved	THNB
R-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 23-68. HWAG Teeth Number Register (HWATHNB) Field Descriptions

Bit	Field	Value	Description
31-8	Reserved	0	Reads return 0. Writes have no effect.
7-0	THNB	0-FFh	Teeth Number. Sets the teeth number with the maximum value of the toothed wheel. This must be equal to N-1 real teeth (that is, 57 for a 60-2 toothed wheel).

23.5.17 HWAG Current Teeth Number Register (HWATHVL)

Figure 23-105. HWAG Current Teeth Number Register (HWATHVL)

31 … 16
Reserved
R-0

15 … 8	7 … 0
Reserved	THVL
R-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 23-69. HWAG Current Teeth Number Register (HWATHVL) Field Descriptions

Bit	Field	Value	Description
31-8	Reserved	0	Reads return 0. Writes have no effect.
7-0	THVL	0-FFh	Teeth Value. Provides the current teeth number.

23.5.18 HWAG Filter Register (HWAFIL)

Figure 23-106. HWAG Filter Register (HWAFIL)

31-16
Reserved
R-0

15-10	9-0
Reserved	FIL1
R-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 23-70. HWAG Filter Register (HWAFIL) Field Descriptions

Bit	Field	Value	Description
31-10	Reserved	0	Reads return 0. Writes have no effect.
9-0	FIL1	0-3FFh	Filter Value. Contains the value to be compared to the tick counter. It allows the tooth signal to be taken into account by the HWAG. This function works only if the mode filtering is set. The value is calculated as shown in Section 23.3.2.2.5.

23.5.19 HWAG Filter Register 2 (HWAFIL2)

Figure 23-107. HWAG Filter Register 2 (HWAFIL2)

31-16
Reserved
R-0

15-12	11-0
Reserved	FIL2
R-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 23-71. HWAG Filter Register 2 (HWAFIL2) Field Descriptions

Bit	Field	Value	Description
31-12	Reserved	0	Reads return 0. Writes have no effect.
11-0	FIL2	0-FFFh	Filter Value 2. Contains the value to be compared to the tick counter during the singularity tooth. It allows the tooth signal to be taken into account by the HWAG. This function works only if the mode filtering is set. The value is calculated as shown in Section 23.3.2.2.5.1.

23.5.20 HWAG Angle Increment Register (HWAANGI)

Figure 23-108. HWAG Angle Increment Register (HWAANGI)

31	16
Reserved	
R-0	

15	10	9	0
Reserved		ANGI	
R-0		R-0	

LEGEND: R = Read only; -*n* = value after reset

Table 23-72. HWAG Angle Increment Register (HWAANGI) Field Descriptions

Bit	Field	Value	Description
31-10	Reserved	0	Reads return 0. Writes have no effect.
9-0	ANGI	0-3FFh	Angle Increment Value. Provides the current angle increment value. The value is incremented by the tick counter and is decremented by the NHET resolution clock.

23.6 Instruction Set

23.6.1 Instruction Summary

Table 23-73 presents a list of the instructions in the N2HET instruction set. The pages following describe each instruction in detail.

Table 23-73. Instruction Summary

Abbreviation	Instruction Name	Opcode	Sub-Opcode	Cycles[1]
ACMP	Angle Compare	Ch	-	1
ACNT	Angle Count	9h	-	2
ADCNST	Add Constant	5h	-	2
ADC	Add with Carry and Shift	4h	C[25:23] = 011, C5 = 1	1-3
ADD	Add and Shift	4h	C[25:23] = 001, C5 = 1	1-3
ADM32	Add Move 32	4h	C[25:23] = 000, C5 = 1	1-2
AND	Bitwise AND and Shift	4h	C[25:23] = 010, C5 = 1	1-3
APCNT	Angle Period Count	Eh	-	1-2
BR	Branch	Dh	-	1
CNT	Count	6h	-	1-2
DADM64	Data Add Move 64	2h	-	2
DJZ	Decrement and Jump if -zero	Ah	P[7:6] = 10	1
ECMP	Equality Compare	0h	C[6:5] = 00	1
ECNT	Event Count	Ah	P[7:6] = 01	1
MCMP	Magnitude Compare	0h	C[6] = 1	1
MOV32	Move 32	4h	C[5] = 0	1-2
MOV64	Move 64	1h	-	1
OR	Bitwise OR	4h	C[25:23] = 100, C5 = 1	1-3
PCNT	Period/Pulse Count	7h	-	1
PWCNT	Pulse Width Count	Ah	P[7:6] = 11	1
RADM64	Register Add Move 64	3h	-	1
RCNT	Ratio Count	Ah	P[7:6] = 00, P[0] = 1	3
SBB	Subtract with Borrow and Shift	4h	C[25:23] =110, C[5] = 1	1-3
SCMP	Sequence Compare	0h	C[6:5] = 01	1
SCNT	Step Count	Ah	P[7:6] = 00, P[0] = 0	3
SHFT	Shift	Fh	C[3] = 0	1
SUB	Subtract and Shift	4h	C[25:23] = 101, C[5] = 1	1-3
WCAP	Software Capture Word	Bh	-	1
WCAPE	Software Capture Word and Event Count	8h	-	1
XOR	Bitwise Exclusive-Or and Shift	4h	C[25:23] = 111, C[5] = 1	1-3

[1] Cycles refers to the clock cycle of the N2HET module; which on most devices is VCLK2. (Check the device datasheet description of clock domains to confirm). If the high-resolution prescale value is set to /1, then this is also the same as the number of HR clock cycles.

Table 23-74. FLAGS Generated by Instruction

Abbreviation	Flag Name	Set/Reset by	Used by
C	Carry Flag	ADC, ADD, AND, OR, RCNT, SBB, SUB, XOR	ADC, BR, SBB
N	Negative Flag	ADC, ADD, AND, OR, SBB, SUB, XOR	BR
V	Overflow Flag	ADC, ADD, AND, OR, SBB, SUB, XOR	BR
Z	Zero flag	ACNT, ADC, ADD, AND, APCNT, CNT, OR, PCNT, SBB, SCNT, SHFT, SUB, XOR	ACMP, ACNT, BR, ECMP, MCMP, MOV32, RCNT, SCMP, SHFT
X	Angle Compare Match Flag	ACMP	SCMP
SWF 0-1	Step Width flags	SCNT	ACNT
NAF	New Angle Flag	ACNT	NAF_global
NAF_global	New Angle Flag (global)	HWAG or NAF	ACMP, BR, CNT, ECMP, ECNT
ACF	Acceleration Flag	ACNT	,ACNT, SCNT
DCF	Deceleration Flag	ACNT	,ACNT, SCNT
GPF	Gap Flag	ACNT	ACNT, APCNT

The instructions capable of generating software interrupts are listed in Table 23-75.

Table 23-75. Interrupt Capable Instructions

Interrupt Capable Instructions	Non Interrupt Capable Instructions
ACMP	ADC
ACNT	ADCNST
APCNT	ADD
BR	ADM32
CNT	AND
DJZ	DADM32
ECMP	MOV32
ECNT	MOV64
MCMP	OR
PCNT	RADM64
PWCNT	RCNT
SCMP	SBB
SHFT	SCNT
WCAP	SUB
WCAPE	XOR

23.6.2 Abbreviations, Encoding Formats and Bits

Abbreviations marked with a star (*) are available only on specific instructions.

U Reading a bit marked with U will return an indeterminate value.

BRK Defines the software breakpoint for the device software debugger.

Default: OFF

Location: Program field [22]

next Defines the program address of the next instruction in the program flow. This value may be a label or an 9-bit unsigned integer.

Default: Current instruction + 1

Location: Program field [21:13]

reqnum* Defines the number of the request line (0,1,..,7) to trigger either the HTU or the DMA.

Default: 0

Location: Program field [25:23]

request* Allows to select between no request (NOREQ), request (GENREQ) and quiet request (QUIET). See Section 23.2.9.

Default: No request

Location: Control Field [28:27]

Request	C[28]	C[27]	To HTU	To DMA
NOREQ	0	0	no request	no request
	1	0		
GENREQ	0	1	request	request
QUIET	1	1	quiet request	no request

remote* Determines the 9-bit address of the remote address for the instruction.

Default: Current instruction + 1

Location: Program field [8:0]

control Determines whether the immediate data field [31:0] is cleared when it is read. When the bit is not set, reads do not clear the immediate data field.

Default: OFF

Location: Control field [26]

en_pin_action* Determines whether the selected pin is ON so that the action occurs on the chosen pin

Default: OFF

Location: Control field [22]

Cond_addr* Conditional address (optional): Defines the address of the next instruction when the condition occurs.

Default: Current address + 1

Location: Control field [21:13]

Pin* Pin Select: Selects the pin on which the action occurs. Enter the pin number.

Default: pin 0

Location: Control field [12:8] except PCNT

The format CC{pin number} is also supported.

MSB				LSB	Description
0	0	0	0	0	Select HET[0]
0	0	0	0	1	Select HET[1]
(Each pin may be selected by writing its number in binary)					
1	1	1	1	0	Select HET[30]
1	1	1	1	1	Select HET[31]

Reg* Register select: Selects the register for data comparison and storage

Default: No register (None)

Location: Control field [2:1] except for CNT instruction.

Extended Register Select C[7] is available for ACMP, ADC, ADD, ADM32, AND, DADM64, ECMP, ECNT, MCMP, MOV32, MOV64, OR, RADM64, SBB, SHFT, SUB, WCAP, WCAPE instructions.

Register	Ext Reg. C[7]	C[2]	C[1]
A	0	0	0
B	0	0	1
T	0	1	0
None	0	1	1
R	1	0	0
S	1	0	1
Reserved	1	1	0
Reserved	1	1	1

Action* (2 Action Option) Either sets or clears the pin

Default: Clear

Location: Control Field [4]

Action	C[4]
Clear	0
Set	1

Action* (4 Action Option) Either sets, clears, pulse high or pulse low on the pin. Set/clear are single pin actions, pulse high/low include the opposite pin action.

Default: Clear

Location: Control Field [4:3]

Action	Action Type	C[4]	C[3]
Clear	Set low on match	0	0
Set	Set high on match	1	0
Pulse Low	Set low on match + reset to high on Z=1 (opposite action)	0	1
Pulse High	Set high on match + reset to low on Z=1 (opposite action)	1	1

hr_lr* Specifies HIGH/LOW data resolution. If the hr_lr field is HIGH, the instruction uses the hr_data field. If the hr_lr field is LOW, the hr_data field is ignored.

Default: HIGH

Location: Program Field [8]

hr_lr	Prog. field [8]
LOW	1
HIGH	0

prv* Specifies the initial value defining the previous bit (see Section 23.2.5.8). A value of ON sets the previous pin-level bit to 1. A value of OFF sets the initial value of the previous (prv) bit to 0. The prv bit is overwritten (set or reset) by the N2HET the first time the instruction is executed.

Default: OFF

Location: Control Field [25]

cntl_val* Available for DADM64, MOV64, and RADM64, this bit field allows the user to specify the replacement value for the remote control field.

comp_mode* Specifies the compare mode. This field is used with the 64-bit move instructions. This field ensures that the sub-opcodes are moved correctly.

Default: ECMP

Location: Control Field [6:5]

Action	C[6]	C[5]	Order
ECMP	0	0	
SCMP	0	1	
MCMP1	1	0	REG_GE_DATA
MCMP2	1	1	DATA_GE_REG

23.6.3 Instruction Description

The following sections provide information for individual instructions.

Parameters in [] are optional. Refer to the N2HET assembler user guide for the default values when parameters are omitted.

23.6.3.1 ACMP (Angle Compare)

Syntax

ACMP {
[brk={OFF | ON}]
[next={label | 9-bit unsigned integer}]
[reqnum={3-bit unsigned integer}]
[request={NOREQ | GENREQ | QUIET}]
[control={OFF | ON}]
[en_pin_action={OFF | ON}]
[cond_addr={label | 9-bit unsigned integer}]
pin={pin number}
[action={CLEAR | SET}]
reg={A | B | R | S | T | NONE}
[irq ={OFF | ON}]
data={25-bit unsigned integer}
}

Figure 23-109. ACMP Program Field (P31:P0)

31 26	25 23	22	21 13	12 9	8 0
0	Request Number	BRK	Next program address	1100	Reserved
6	3	1	9	4	9

Figure 23-110. ACMP Control Field (C31:C0)

31 29	28 27	26	25	24 23	22	21 16
Reserved	Request type	Control	Cout prv	Reserved	En. pin action	Conditional address
3	2	1	1	2	1	9

15 13	12 8	7	6 5	4	3	2 1	0
Conditional address	Pin select	Ext. Reg	Reserved	Pin action	Res.	Register select	Int. ena
9	5	1	2	1	1	2	1

Figure 23-111. ACMP Data Field (D31:D0)

31 7	6 0
Data	Reserved
25	7

Cycles One

Register modified Selected register (A, B, R, S, or T)

The purpose of the comparison is to assert pin action when the angle compare value lies between the old counter value and the new counter value (held in the selected register). Since the angle increment varies from one loop resolution clock to another, an exact equality test cannot be applied. Instead, the following inequality is used to determine the occurrence of a match:

Old counter value < Angle compare value ≤ New counter value

This is done by performing following comparisons:

Selected register value minus angle increment < angle compare value
Angle compare value ≤ Selected register value

register Register B is recommended for typical applications with ACMP.

irq Specifies whether or not an interrupt is generated. Specifying ON generates an interrupt when the edge state is satisfied and the gap flag is set. Specifying OFF prevents an interrupt from being generated.

Default: OFF.

data Specifies the 25-bit angle compare value.

Execution

```
X = 0;
If (Data <= Selected Register)
      Cout = 0;
else
      Cout = 1;
If (Z == 0 AND (Selected Register - Angle Inc. < Data ) AND Cout == 0) OR

 (Z == 1 AND (Cout_prv == 1 OR Cout == 0)))
{
      X = 1;
      If (Enable Pin Action == 1)
          Selected Pin = Pin Action AT next loop resolution clock;

      If (Interrupt Enable == 1)
          HETFLG[n] = 1;   /* n depends on address */
      If ([C28:C27] == 01)
          Generate request on request line [P25:P23];
      If ([C28:C27] == 11)
          Generate quiet request on request line [P25:P23];

      Jump to Conditional Address;
 }

 else
      Jump to Next Program Address;

Cout_prv = Cout (always executed)
```

NOTE: Carry-Out Signal (Cout)

Cout is the carry-out signal of the adder. Even if it is not a flag, it is valid all along ACMP instruction execution.

Angle inc. = NAF_global or hardware angle generator 11-bit input.

The specific interrupt flag that is triggered depends on the address from which the instruction is executed, see Section 23.2.7.

23.6.3.2 ACNT (Angle Count)

Syntax

ACNT {
[brk={OFF | ON}]
[next={label | 9-bit unsigned integer}]
[reqnum={3-bit unsigned integer}]
[request={NOREQ | GENREQ | QUIET}]
edge={RISING | FALLING}
[irq ={OFF | ON}]
[control={OFF | ON}]
[prv={OFF | ON}]
gapend ={25-bit unsigned integer}
data={25-bit unsigned integer}
}

Figure 23-112. ACNT Program Field (P31:P0)

31 26	25 23	22	21 13	12 9	8	7 1	0
0	Request Number	BRK	Next program address	1001	Edge select	Reserved	Int. ena
6	3	1	9	4	1	7	1

Figure 23-113. ACNT Control Field (C31:C0)

31 29	28 27	26	25	24 0
Res.	Request type	Control	Prv.	Gap End
3	2	1	1	25

Figure 23-114. ACNT Data Field (D31:D0)

31 7	6 0
Data	Reserved
25	7

Cycles

Two, as follows:

- First cycle: Angle increment condition and gap end comparison.
- Second cycle: Gap start comparison.

Register modified

Register B (angle value)

Description

This instruction defines a specialized virtual timer used after SCNT and APCNT to generate an angle-referenced time base that is synchronized to an external signal (that is, a toothed wheel signal). ACNT uses pin HET[2] exclusively. The edge select must be the same as the HET[2] edge which was selected in the previous APCNT.

ACNT refers to the same step width selection that the previous SCNT saved in flags SWF0 and SWF1 (see information on SCNT).

ACNT detects period variations of the external signal measured by APCNT and compensates related count errors.

A period increase is flagged in the deceleration flag (DCF). A period decrease is flagged in the acceleration flag (ACF). If no variation is detected, ACNT increments the counter value each time SCNT reaches its target.

If acceleration is detected, ACNT increments the counter value on each timer resolution. If deceleration is detected ACNT does not increment and is thus saturated.

ACNT also specifies the gap end angle value defining the end value of a gap range in ACNT where period measurements in APCNT are temporarily stopped to mask singularities in the external signal. ACNT uses register A containing gap start and register B to store the counter value.

Edge Specifies the edge for the input capture pin (HET[2]).

Action	P8	Edge Select
Rising	1	Detects a rising edge of HET[2]
Falling	0	Detects a falling edge of HET[2]

irq ON generates an interrupt when the edge state is satisfied and the gap flag is set. OFF prevents an interrupt from being generated.

Default: OFF.

gapend Defines the 25-bit end value of a gap range. The start value is defined in the SCNT instruction.

GAPEND = (Step Value * (# of teeth on the toothed wheel + # of missing teeth)) - 1

data Specifies the 25-bit initial count value for the data field.

Default: 0.

NOTE: Target Edge Field

The target edge field represents the three LSBs of data field register in case of step width = 8, four LSBs for step width = 16, five LSBs for step width = 32 and six LSBs for step width = 64.

Execution

```
Increment Condition: ((Z = 1 AND DCF = 0) OR ACF = 1)
Pin Edge Condition: Specified edge detected on HET[2]
Target Edge Condition: (Target Edge field in data field = 0) AND (Angle
Increment condition is true) AND (GPF = 0)

If (Angle Increment Condition) is false
{
      NAF = 0;
      Register B = Data field register;
}
else
{
      NAF = 1;
      If (Counter value != GapEnd)
      {
           Register B = Data field register + 1;
           Data Field Register = Counter value + 1;
      }
      else
      {
           Register B = 0;
           Data Field Register = 0;
           If (ACF == 0) DCF = 1;
      }
}

Z = 0;

If (Data field register == GapStart)
{
       GPF = 1;
       If (Target Edge condition is true)
       {
           ACF = 0;
           If ((specified edge is not detected on pin HET[2]) AND (data
           field register != 0) AND (ACF == 0) AND (angle increment condition
           is true))
             DCF = 1;
       }
       If (specified edge is detected on pin HET[2])
       {
           DCF = 0;
           If ((target_edge_field != 0) AND (DCF == 0)) ACF = 1;
           If (GPF == 1)
           {
             GPF = 0;
             Z = 1;
             If (Interrupt Enable == 1)
                HETFLG[n] = 1;      /* n depends on address */
             If ([C28:C27] == 01)
                Generate request on request line [P25:P23];;
             If ([C28:C27] == 11)
                Generate quiet request on request line
                [P25:P23];
           }
        }
}

If ((target_edge_field != 0) and (pin_edge_cond == 1))
{
   pin_update = 0;
}
else if (target_edge_field == 0)
{
   pin_update = 1;
}

If (pin_update is true in next loop clock cycle)
{
   Prv bit = Current Lx value of HET[2] pin;
}

Jump to next program address;
```

The specific interrupt flag that is triggered depends on the address from which the instruction is executed, see Section 23.2.7.

23.6.3.3 ADCNST (Add Constant)

Syntax

ADCNST {
[brk={OFF | ON}]
[next={label | 9-bit unsigned integer}]
[control={OFF | ON}]
remote={label | 9-bit unsigned integer}
min_off={25-bit unsigned integer}
data={25-bit unsigned integer}
[hr_data={7-bit unsigned integer}]
}

Figure 23-115. ADCNST Program Field (P31:P0)

31 26	25 23	22	21 13	12 9	8 0
0	Reserved	BRK	Next program address	0101	Remote address
6	3	1	9	4	9

Figure 23-116. ADCNST Control Field (C31:C0)

31 27	26	25	24 0
Reserved	Control	Res.	Minimum offset
5	1	1	25

Figure 23-117. ADCNST Data Field (D31:D0)

31 7	6 0
Data	HR Data
25	7

Cycles Two

Register modified Register T (implicity)

Description ADCNST is an extension of ADM32. ADCNST first checks whether the data field value at the remote address is zero; it then performs different adds and moves on the result. ADCNST is typically used to extend the counter value of PWCNT.

min_off A 25-bit constant value that is added to the data field value if the remote data field is null.

data A 25-bit value that is always added to the remote data field.

Default: 0.

hr_data Seven least significant bits of the data addition to the remote data field.

Default: 0.

Figure 23-118 and Figure 23-119 illustrate the behavior of ADCNST if the remote data field is zero or is not zero.

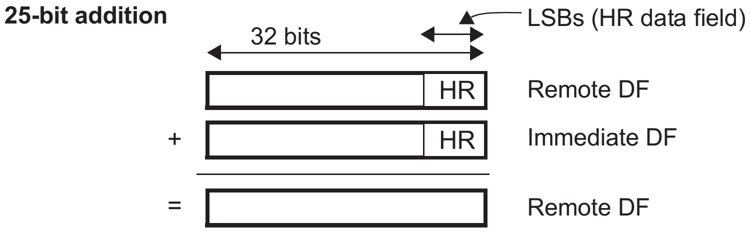


Figure 23-118. ADCNST Operation If Remote Data Field[31:7] Is Not Zero

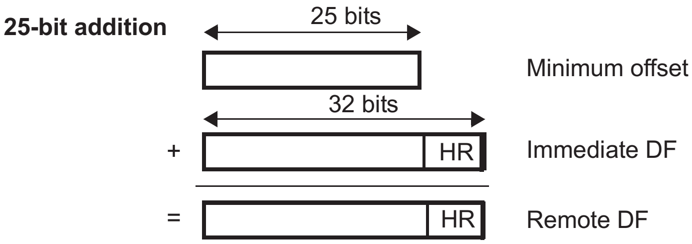


Figure 23-119. ADCNST Operation if Remote Data Field [31:7] Is Zero

Execution

```
If (Remote Data Field Value [31:7] != 0)
      Remote Data Field = Immediate Data Field + Remote Data Field;
else
      Remote Data Field = Immediate Data Field + min. offset(bits C24:C0);

Jump to Next Program Address;
```

23.6.3.4 ADC, ADD, AND, OR, SBB, SUB, XOR

Syntax

```
ADC | ADD | AND | OR | SBB | SUB | XOR {
src1 = { ZERO | IMM | A | B | R | S | T | ONES | REM | REMP }
src2 = { ZERO | IMM | A | B | R | S | T | ONES }
dest = { NONE | IMM | A | B | R | S | T }
[rdest = { NONE | REM | REMP }]
[brk={OFF | ON}]
[next={label | 9-bit unsigned integer}]
[remote={label | 9-bit unsigned integer}]
[control={OFF | ON}]
[init={OFF | ON}]
[smode = {LSL | CSL | LSR | CSR | RR | CRR | ASR }]
[scount = {5 bit unsigned integer}]
[data={25-bit unsigned integer}]
[hr_data={7-bit unsigned integer}]
}
```

Figure 23-120. ADC, ADD, AND, OR, SBB, SUB, XOR Program Field (P31:P0)

31 26	25 23	22	21 13	12 9	8 0
0	Reserved	BRK	Next program address	0100	Remote address
6	3	1	9	4	9

Figure 23-121. ADC, ADD, AND, OR, SBB, SUB, XOR Control Field (C31:C0)

31 27	26	25 23	22 19	18 16
Reserved	Control	Sub Opcode	Src1	Src2
5	1	3	4	3

15 13	12 8	7	6	5	4 3	2 1	0
Smode	Scount	Ext. Reg	Init flag	1	Rdest	Register select	Res.
3	5	1	1	1	2	2	1

Figure 23-122. ADC, ADD, AND, OR, SBB, SUB, XOR Data Field (D31:D0)

31 7	6 0
Data	HR Data
25	7

Cycles One to three cycles, depending on operands selected. (See Table 23-80)

Register modified Selected register (A, B, R, S, T, or NONE)

Description This instruction performs the specified 32-bit arithmetic or logical operation on operands src1 and src2, followed by an optional shift/rotate step. The result of this operation is then stored to either an N2HET register or the immediate data field of the instruction. In addition, the same result may be stored in a remote data field or the least signficant bits of a remote instruction program field (P[8:0]). Bits P[8:0] of the program field are used by most instructions formats to hold the remote address that the instruction operates on, so the ability to update this field programatically makes it easier to write subroutines that operate on different data sets.

The Sub-Opcode field C[25:3] determines which type of operation (ADD, ADC, AND, OR, SBB, SUB, XOR) is executed by the instruction. A list of these operations and the corresponding Sub-Opcode encoding can be found in Table 23-76.

All arithmetic is performed using 32-bit integer math. However, source and destination operands vary in width and can be 9 bits (REMP), 25 bits (A, B) or 32 bits (R,S,T, IMM, REM). Source operands REMP, A,B are extended to 32-bits before being operated on. Also the result of the computation needs to be truncated before being written back to REMP, A, or B when these are selected as destination operands. Table 23-77 provides a list of source operand options, how they are expanded to 32-bit integers (if applicable) and the control field encoding to select the option for src1 and src2 operands.

Table 23-78 provides a similar list of destination operands and their encodings. Up to two destination operands may be selected for each instruction, a register/immediate destination and a remote destination may be selected simultaneously. Truncation is performed independently for each destination operand as appropriate to its size.

An optional shift step following the arithmetic or logical operation may be selected through the smode and scount operands. The shift or rotate type is selected by the smode field; Table 23-79 illustrates the options that are available for smode. The number of bits shifted is determined by the scount operand.

Table 23-76. Arithmetic / Bitwise Logic Sub-Opcodes

Instruction	Description	Operation	Sub-Opcode
ADC	Add with Carry	result = src1 + src2 + C	C[25:23] = 011
ADD	Add	result = src1 + src2	C[25:23] = 001
AND	Bitwise Logic And	result = src1 & src2	C[25:23] = 010
OR	Bitwise Logic Or	result = src1 | src2	C[25:23] = 100
SBB	Subtract with Borrow	result = src1 - src2 - C	C[25:23] = 110
SUB	Subtract	result = src1 - src2	C[25:23] = 101
XOR	Bitwise Logic Exclusive Or	result = src1 ^ src2	C[25:23] = 111

Table 23-77. Source Operand Choices

Source Operand	32-bit value	Address	src1	src2
A	{A[24:0], 0x00}	n/a	C[22:19] = 0010	C[18:16] = 010
B	{B[24:0], 0x00}	n/a	C[22:19] = 0011	C[18:16] = 011
R	R[31:0]	n/a	C[22:19] = 0100	C[18:16] = 100
S	S[31:0]	n/a	C[22:19] = 0101	C[18:16] = 101
T	T[31:0]	n/a	C[22:19] = 0110	C[18:16] = 110
IMM	D[31:0]	current instruction address	C[22:19] = 0001	C[18:16] = 001
ZERO	0x00000000	n/a	C[22:19] = 0000	C[18:16] = 000
ONES	0xFFFFFFFF	n/a	C[22:19] = 0111	C[18:16] = 111
REM	D[31:0]	specified by remote[8:0]	C[22:19] = 1000	n/a
REMP	{0x000000, P[8:0]}	specified by remote[8:0]	C[22:19] = 1001	n/a

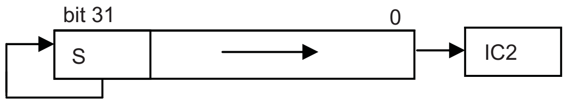
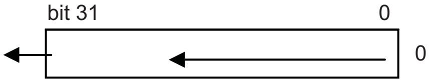
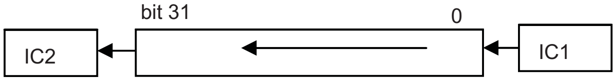
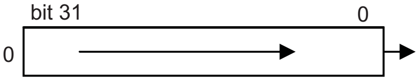
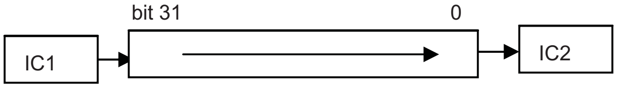
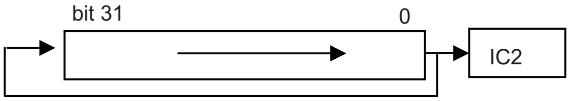
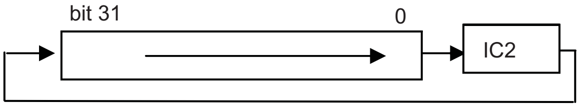
Table 23-78. Destination Operand Choices

Destination Operand	Stored Value	Address	dest	rdest
A	A[24:0] = result [31:8]	n/a	C[7] = 0, C[2:1] = 00	n/a
B	B[24:0] = result [31:8]	n/a	C[7] = 0, C[2:1] = 01	n/a
R	R[24:0] = result [31:0]	n/a	C[7] = 1, C[2:1] = 00	n/a
S	S[24:0] = result [31:0]	n/a	C[7] = 1, C[2:1] = 01	n/a
T	T[24:0] = result [31:0]	n/a	C[7] = 0, C[2:1] = 10	n/a

Table 23-78. Destination Operand Choices (continued)

Destination Operand	Stored Value	Address	dest	rdest
IMM	D[31:0] = result [31:0]	current instruction address	C[7] = 1, C[2:1] = 10	n/a
NONE	n/a	n/a	C[7] = 0, C[2:1] = 11	C[4:3] = 00
REM	D[31:0] = result [31:0]	specified by remote[8:0]	n/a	C[4:3] = 01
REMP	P[8:0] = result [8:0]	specified by remote[8:0]	n/a	C[4:3] = 10

Table 23-79. Shift Encoding

Shift Type	C[15:13] smode	Operation Illustrated [1]
No Shift Applied	0 0 0	n/a - no shift
ASR-Arithmetic Shift Right	0 0 1	
LSL-Logical Shift Left	0 1 0	
CSL-Carry Shift Left	0 1 1	
LSR-Logical Shift Right	1 0 0	
CSR-Carry Shift Right	1 0 1	
RR - Rotate Right	1 1 0	
CRR – Carry Rotate Right	1 1 1	

[1] IC1 is the carry flag after the arithmetic / logical operation is performed. Ic2 is the updated carry flag after the shift operation is performed. s is the sign bit.

Table 23-80. Execution Time for ADC, ADD, AND, OR, SBB, SUB, XOR Instructions

src1	dest	rdest	remote[8:0]	Cycles
ZERO, IMM, A, B, R, S, T, or ONES	A,B,R,S,T, or NONE	NONE	! = next[8:0]	1
REM or REMP	A,B,R,S,T, or NONE	NONE	!= next[8:0]	2
ZERO, IMM, A, B, R, S, T, or ONES	IMM	REM	!= next[8:0]	2
ZERO, IMM, A, B, R, S, T, or ONES	A,B,R,S,T, or NONE	REMP	!= next[8:0]	2
ZERO, IMM, A, B, R, S, T, or ONES	A,B,R,S,T, or NONE	NONE	== next[8:0]	2
REM or REMP	IMM	REM	x	3
x	IMM	REMP	x	3
REM or REMP	x	REM	== next[8:0]	3
x	x	REMP	== next[8:0]	3

Execution

```
         / Notes: IR1, IR2 are 32-bit intermediate results
   // SRC1, SRC2 are 32-bit sources selected
   //            by fields src1, src2
   // IC1, IC2 are intermediate values of the carry flag
   // IZ1, IZ2 are intermediate values of the zero flag
   // IN1, IN2 are intermediate values of the negative flag
   // IV1, IV2 are intermediate values of the overflow flag
   // scount is the shift count (0 to 31) specified by C12:C8

   /********** SOURCE OPERAND DECODING STAGE **********/
   switch (C22:C19)
{
     case 0000:SRC1[31:0] = 0x00000000
     case 0001:SRC1[31:0] = Immediate Data Field D[31:0]
     case 0010:SRC1[31:8] = A[24:0]; SRC1[6:0] = 0
     case 0011:SRC1[31:8] = B[24:0]; SRC1[6:0] = 0
     case 0100:SRC1[31:0] = R[31:0]
     case 0101:SRC1[31:0] = S[31:0]
     case 0110:SRC1[31:0] = T[31:0]
     case 0111:SRC1[31:0] = 0xFFFFFFFF
     case 1000:SRC1[31:0] = Remote Data Field D[31:0]
     case 1001:SRC1[31:9] = 0; SRC1[8:0] = Remote Program Field P[8:0]
   }

switch (C18:C16)
{
     case 000:SRC2[31:0] = 0x00000000
     case 001:SRC2[31:0] = Immediate Data Field[31:0]
     case 010:SRC2[31:8] = A[24:0]; SRC2[6:0] = 0
     case 011:SRC2[31:8] = B[24:0]; SRC2[6:0] = 0
     case 100:SRC2[31:0] = R[31:0]
     case 101:SRC2[31:0] = S[31:0]
     case 110:SRC2[31:0] = T[31:0]
     case 111:SRC2[31:0] = 0xFFFFFFFF
   }
   /******** ARITHMETIC / LOGICAL OPERATION STAGE *******/
   switch (C[25:23])
   {
     case 011:IR1 = src1 + src2 + C // ADC
     case 001:IR1 = src1 + src2 // ADD
     case 010:IR1 = src1 & src2 // AND
     case 100:IR1 = src1 | src2 // OR
     case 110:IR1 = src1 - src2 - C // SBB
     case 101:IR1 = src1 - src2 // SUB
     case 111:IR1 = src1 ^ src2 // XOR
   }

   IC1 = Carry Out if Operation is ADD, ADC, SUB, SBB
           = 0 if Operation is AND, OR, XOR
   IZ1 = Set if IR1 is zero, Clear if IR1 is non-zero
   IN1 = IR[31]
   IV1 = (IC1 XOR IR1[31]) AND NOT(SRC1[31] XOR SRC2[31])

   /******************** SHIFT STAGE ********************/
   switch (C15:C13)
{
     case 000: // smode = No Shift
           IR2 = IR1
           IC2 = IC1; IZ2 = IZ1; IN2 = IN1; IV2 = IV1;

     case 001: // smode = Arithmetic Shift Right
           IR2[31 - scount : 0] = IR1[31:scount]

           if (scount>0) {
              IR2[31 : 31 - scount + 1] = IR1[31]
```

```
                     IC2 = IR1[scount-1]
                  }
                  else {
                     IC2 = IC1
                  }

                  IN2 = IR2[31];
                  if (IR2 == 0) { IZ2 = 1 } else {IZ2 = 0};
                  IV2 = (IR2[31] XOR IR1[31]) OR IV1

            case 010: // smode = Logical Shift Left
                  IR2[31 : scount] = IR1[31 - scount: 0]

                  if (scount > 0) {
                     IR2[scount - 1 : 0] = 0
                  }

                  IC2 = IC1
                  IN2 = IR2[31];
                  if (IR2 == 0) { IZ2 = 1 } else {IZ2 = 0};
                  IV2 = (IR2[31] XOR IR1[31]) OR IV1

            case 011: // smode = Carry Shift Left
                  IR2[31 : scount] = IR1[31 - scount: 0]

               if (scount>0) {
                     IR2[scount - 1 : 0] = [IC1,...IC1]
                     IC2 = IR1[31 - scount + 1]
                  }
                  else
               {
                     IC2 = IC1
                  }

               IN2 = IR2[31];
                  if (IR2 == 0) { IZ2 = 1 } else {IZ2 = 0};
                  IV2 = (IR2[31] XOR IR1[31]) OR IV1

            case 100: // smode = Logical Shift Right
                  IR2[31 - scount : 0] = IR1[31:scount]

                  if (scount>0) {
                     IR2[31 : 31 - scount + 1] = 0
                  }

                  IC2 = IC1
                  IN2 = IR2[31];
                  if (IR2 == 0) { IZ2 = 1 } else {IZ2 = 0};
                  IV2 = (IR2[31] XOR IR1[31]) OR IV1

            case 101: // smode = Carry Shift Right
                  IR2[31 - scount : 0] = IR1[31:scount]

                  if(scount>0) {
                     IR2[31:31-scount + 1] = [IC1,...IC1]
                     IC2 = IR1[scount-1]
                  }
                  else {
                     IC2 = IC1
                  }

                  IN2 = IR2[31];
                  IZ2 = Set if IR2 == 0;
                  IV2 = (IR2[31] XOR IR1[31]) OR IV1

            case 110: // smode = Rotate Right
```

```
         IR2[31 - scount : 0] = IR1[31:scount]

         if(scount>0) {
            IR2[31:31-scount+1] = IR1[scount-1:0]
            IC2 = IR1[scount-1]
         }
         else {
            IC2 = IC1
         }

         IN2 = IR2[31];
         if (IR2 == 0) { IZ2 = 1 } else {IZ2 = 0};
         IV2 = (IR2[31] XOR IR1[31]) OR IV1

   case 111: // smode = Carry Rotate Right
         IR2[31 - scount : 0] = IR1[31:scount]

         if (scount == 0) {
            IC2 = IC1
         }
         else if (scount == 1) {
            IR2[31] = IC1
            IC2 = IR1[0]
         }
         else {
            IR2[31:31-scount+1] = {IR1[scount-2:0],IC1}
            IC2 = IR1[scount - 1]
         }

         IN2 = IR2[31];
         if (IR2 == 0) { IZ2 = 1 } else {IZ2 = 0};
         IV2 = (IR2[31] XOR IR1[31]) OR IV1
   }
   /********** WRITE REGISTER DESTINATION STAGE ***********/
   switch (C7, C2:C1)
   {
         case 000:A[24:0] = IR2[31:8]
         case 001:B[24:0] = IR2[31:8]
         case 010:T[31:0] = IR2[31:0]
         case 011:IR2 is not stored in register, immediate
         case 100:R[31:0] = IR2[31:0]
         case 101:S[31:0] = IR2[31:0]
         case 110:Immediate Data Field[31:0] = IR2
         case 111:IR2 is not stored in register, immediate
   }
   /*********** WRITE REMOTE DESTINATION STAGE ***********/
   switch (C4:3)
{
         case 00:IR2 is not stored in remote field
         case 01:Remote Data Field D[31:0] = IR2
         case 10:Remote Program Field P[8:0] = IR2[8:0]
         case 11:IR2 is not stored in remote field
   }
   /***************** UPDATE FLAGS STAGE *****************/
   C FLAG = IC2
   N FLAG = IN2
   Z FLAG = IZ2
   V FLAG = IV2
   If (Init Flag == 1)
   {
         ACF = 0;
         DCF = 1;
         GPF = 0;
         NAF = 0;
   }
   else ACF, DCF, GPF, NAF remain unchanged;
```

23.6.3.5 ADM32 (Add Move 32)

Syntax

ADM32 {
[brk={OFF | ON}]
[next={label | 9-bit unsigned integer}]
remote={label | 9-bit unsigned integer}
[control={OFF | ON}]
[init={OFF | ON}]
type={IM&REGTOREG | REM&REGTOREG | IM&REMTOREG | IM&REGTOREM}
reg={A | B | R | S | T }
data={25-bit unsigned integer}
[hr_data={7-bit unsigned integer}]
}

Figure 23-123. ADM32 Program Field (P31:P0)

31 26	25 23	22	21 13	12 9	8 0
0	Reserved	BRK	Next program address	0100	Remote address
6	3	1	9	4	9

Figure 23-124. ADM32 Control Field (C31:C0)

31 27	26	25 23	22 16
Reserved	Control	000	Reserved
5	1	3	15

15 8	7	6	5	4 3	2 1	0
Reserved	Ext Reg	Init flag	1	Move type	Register select	Res.
15	1	1	1	2	2	1

Figure 23-125. ADM32 Data Field (D31:D0)

31 7	6 0
Data	HR Data
25	7

Cycles One or two cycles (see Table 23-81)

Register modified Selected register (A, B, R, S, or T)

Description This instruction modifies the selected ALU register or data field values at the remote address depending on the move type. The modified value results from adding the immediate or remote data field to the ALU register or the remote data field, depending on the move type. Table description shows the C2 and C1 bit encoding for determining which register is selected.

init (Optional) Determines whether or not system flags are initialized. A value of ON reinitializes the following system flags to these states:

Acceleration flag (ACF) = 0

Deceleration flag (DCF) = 1

Gap flag (GPF) = 0

New angle flag (NAF) = 0

A value of OFF results in no change to the system flags.

Default: OFF

type Specifies the move type to be executed.

Table 23-81. Move Types for ADM32

Type	C4	C3	Add	Destination(s)	Cycles
IM&REGTOREG	0	0	Imm. data field + Reg. A, B, R, S, or T	Register A, B, R, S, or T	1
REM&REGTOREG	0	1	Remote data field + Reg. A, B, R, S, or T	Register A, B, R, S, or T	2
IM&REMTOREG	1	0	Imm. data field + Remote data field	Register A, B, R, S, or T	2
IM&REGTOREM	1	1	Imm. data field + Reg. A, B, R, S, or T	Remote data field	1

If selected register is R, S, or T, the operation is a 32-bit Addition/move. If A or B register is selected, it is limited to 25-bit operation since A and B only support 25-bit.

data Specifies the 25-bit integer value for the immediate data field.

hr_data Specifies the 7 least significant bits of the immediate data field.

Default: 0.

Execution

```
switch (C4:C3)
{
      case 00:
           Selected register = Selected register + Immediate Data Field;
      case 01:
           Selected register = Selected register + Remote Data Field;
      case 10:
           Selected register = Immediate Data Field + Remote Data Field;
      case 11:
           Remote Data Field = Selected register + Immediate Data Field;
}

If (Init Flag == 1)
{
      ACF = 0;
      DCF = 1;
      GPF = 0;
      NAF = 0;
}
else
      All flags remain unchanged;

Jump to Next Program Address;
```

Figure 23-126 and Figure 23-127 illustrate the ADM32 operation for various cases.

Figure 23-126. ADM32 Add and Move Operation for IM&REGTOREG (Case 00)

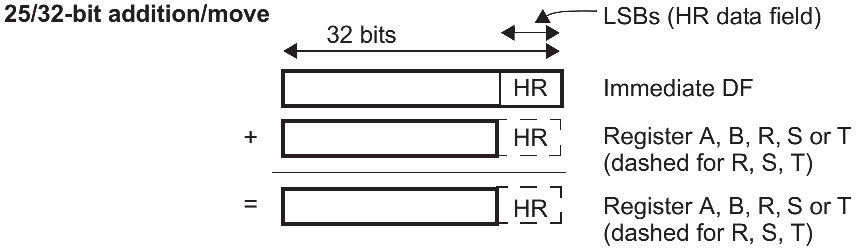
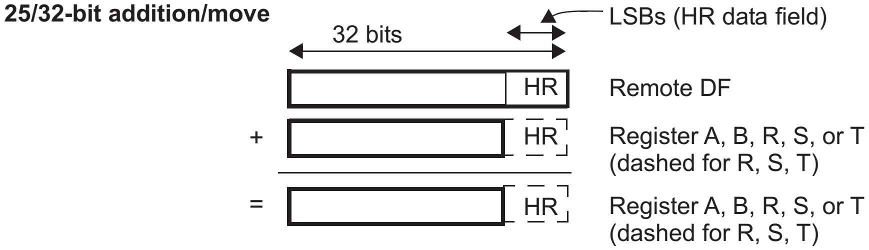


Figure 23-127. ADM32 Add and Move Operation for REM&REGTOREG (Case 01)



23.6.3.6 APCNT (Angle Period Count)

Syntax

```
APCNT {
[brk={OFF | ON}]
[next={label | 9-bit unsigned integer}]
[reqnum={3-bit unsigned integer}]
[request={NOREQ | GENREQ | QUIET}]
[irq={OFF | ON}]
type={FALL2FALL | RISE2RISE}
[control={OFF | ON}]
prv={OFF | ON}}]
period={25-bit unsigned integer}
data={25-bit unsigned integer}
}
```

Figure 23-128. APCNT Program Field (P31:P0)

31 26	25 23	22	21 13	12 9	8	7 6	5 0
0	Request Number	BRK	Next program address	1110	Int. ena	Edge select	Reserved
6	3	1	9	4	1	2	6

Figure 23-129. APCNT Control Field (C31:C0)

31 29	28 27	26	25	24 0
Res.	Request type	Control	Prv.	Period Count
3	2	1	1	25

Figure 23-130. APCNT Data Field (D31:D0)

31 7	6 0
Data	Reserved
25	7

Cycles

One or two cycles

- Cycle 1: edge detected (normal operation)
- Cycle 2: edge detected and GPF = 1 and underflow condition is true

One cycle (normal operation) two cycles (edge detected)

Register modified

Register A and T (implicitly)

Description

This instruction is used before SCNT and ACNT to generate an angle-referenced time base synchronized to an external signal (that is, a toothed wheel signal). It is assumed that the pin and edge selections are the same for APCNT and ACNT.

APCNT is restricted to pin HET[2]. The toothed wheel must then be connected to pin HET[2].

APCNT uses the gap flag (GPF) defined by ACNT to start or stop captures in the period count field [C24:C0]. When GPF = 1, the previous period value is held in the control field and in register T. When GPF = 0, the current period value is captured in the control field and in register T.

APCNT uses the step width flags (SWF0 and SWF1) defined by SCNT to detect period durations shorter than one step, and then disables capture.

The edge select encoding is shown in Table 23-82.

irq ON generates an interrupt when the edge state is satisfied. OFF prevents an interrupt from being generated.

Default: OFF.

type Specifies the edge type that triggers the instruction.

Default: Fall2Fall.

Table 23-82. Edge Select Encoding for APCNT

type	P7	P6	Selected Condition
Fall2Fall	1	0	Falling edge
Rise2Rise	1	1	Rising edge

period Contains the 25-bit count value from the previous APCNT period.

data 25-bit value serving as a counter.

Default: 0.

Execution

```
Z = 0;

If (Data field register != 1FFFFFFh)
{
   Register A = Data field register + 1;
   Data field register = Data field register + 1;
}
elseIf (specified edge not detected on HET[2])
{
   Register A = 1FFFFFFh;
   APCNT Ovflw flag = 1;
}

If (specified edge detected on HET[2])
{
   Z = 1;

   If (Data field register == 1FFFFFFh)
   {
      Register A = 1FFFFFFh;
      Register T = 1FFFFFFh;Period count = 1FFFFFFh;
      Period count = 1FFFFFFh;
   }
   elseIf (GPF == 0 AND Data Field register >= Step width)
   {
      Register A = Data field register + 1;
      Register T = Register A;
      Period count = Register T;

      If (Interrupt Enable == 1)
         HETFLG[n] = 1;      /* n depends on address */
      If ([C28:C27] == 01)
         Generate request on request line [P25:P23];
      If ([C28:C27] == 11)
         Generate quiet request on request line [P25:P23];
   }

   If (GPF == 1)
          Register T = Period count;
   If (Data Field register < Step width)
   {
      Register T = Period count;
      APCNT Undflw flag = 1;
      Period Count = 000000h;
   }

   Data field register = 000000h;
}
else
{
      Register T = Period count;
}

Prv bit = Current Lx value of HET[2] pin;

Jump to Next Program Address;
```

The specific interrupt flag that is triggered depends on the address from which the instruction is executed, see Section 23.2.7.

23.6.3.7 BR (Branch)

Syntax

BR {
[brk={OFF | ON}]
[next={label | 9-bit unsigned integer}]
[reqnum={3-bit unsigned integer}]
[request={NOREQ | GENREQ | QUIET}]
[control={OFF | ON}]
[prv={OFF | ON}]
cond_addr={label | 9-bit unsigned integer}
[pin= {pin number}]
event={NOCOND | FALL | RISE | BOTH | ZERO | NAF | LOW | HIGH | C | NC | EQ | Z | NE | NZ | N | PZ | V | NV | ZN | P | GE | LT | GT | LE | LO | HS }
[irq={OFF | ON}]
}

Figure 23-131. BR Program Field (P31:P0)

31 26	25 23	22	21 13	12 9	8 0
0	Request Number	BRK	Next program address	1101	Reserved
6	3	1	9	4	9

Figure 23-132. BR Control Field (C31:C0)

31 29	28 27	26	25	24 22	21 16
Reserved	Request type	Control	Prv	Reserved	Conditional address
3	2	1	1	3	9

15 13	12 8	7 3	2 1	0
Conditional address	Pin select	Branch cond.	Reserved	Int. ena
9	5	5	2	1

Figure 23-133. BR Data Field (D31:D0)

31 0
Reserved
32

Cycles One

Register modified None

Description This instruction executes a jump to the conditional address [C21:C13] on a pin or a flag condition, and can be used with all pins.

Table 23-83 provides the branch condition encoding.

event Specifies the event that triggers a jump to the indexed program address.

Default: FALL

irq ON generates an interrupt when the event occurs that triggers the jump. If irq is set to OFF, no interrupt is generated.

Default: OFF.

Table 23-83. Branch Condition Encoding for BR

Event	C7	C6	C5	C4	C3	Branch Condition
NOCOND	0	0	0	0	0	Always
FALL	0	0	1	0	0	On falling edge on the selected pin
RISE	0	1	0	0	0	On rising edge on selected pin
BOTH	0	1	1	0	0	On rising or falling edge on selected pin
ZERO	1	0	0	0	0	If Zero flag is set
NAF	1	0	1	0	0	If NAF_global flag is set
LOW	1	1	0	0	0	On LOW level on selected pin
HIGH	1	1	1	0	0	On HIGH level on selected pin
C	0	0	0	0	1	Carry Set: C==1
NC	0	0	0	1	1	Carry Not Set: C==0
EQ, Z	0	0	1	0	1	Equal or Zero: Z==1
NE, NZ	0	0	1	1	1	Not Equal or Not Zero: Z==0
N	0	1	0	0	1	Negative: N==1
PZ	0	0	1	1	1	Positive or Zero: N==0
V	0	1	1	0	1	Overflow: V==1
NV	0	1	1	1	1	No Overflow: V==0
ZN	1	0	0	0	1	Zero or Negative: (Z OR N) == 1
P	1	0	0	1	1	Positive: (Z OR N) == 0
GE	1	0	1	1	1	Signed Greater Than or Equal: (N XOR V) == 0
L	1	0	1	0	1	Signed Less Than (N XOR V) == 1
G	1	1	0	1	1	Signed Greater Than (Z OR (N XOR V)) == 0
LE	1	1	0	0	1	Signed Less Than (Z OR (N XOR V)) == 1
LO	1	1	1	1	1	Unsigned Less Than: (C OR Z) == 0
HS	1	1	1	0	1	Unsigned Higher or Same (C OR Z) == 1

Execution

```
If (Condition is true)
{
   If (Interrupt Enable == 1) HETFLG[n] = 1;      /* n depends on address */
   If ([C28:C27] == 01) Generate request on request line [P25:P23];
   If ([C28:C27] == 11) Generate quiet request on request line [P25:P23];
   Jump to Conditional Address;
}
else
{
   Jump to Next Program Address;
}

Prv bit = Current Lx value of selected pin; (Always Executed)
```

The specific interrupt flag that is triggered depends on the address from which the instruction is executed, see Section 23.2.7.

23.6.3.8 CNT (Count)

Syntax

CNT {
[brk={OFF | ON}]
[next={label | 9-bit unsigned integer}]
[reqnum={3-bit unsigned integer}]
[request={NOREQ | GENREQ | QUIET}]
[angle_count={OFF | ON}]
[reg={A | B | T | NONE}]
[comp ={EQ | GE}]
[irq={OFF | ON}]
[control={OFF | ON}]
max={25-bit unsigned integer}
[data={25-bit unsigned integer]
}

Figure 23-134. CNT Program Field (P31:P0)

31 26	25 23	22	21 13	12 9	8	7 6	5	4 1	0
0	Request Number	BRK	Next program address	0110	Angle count	Register	Comp. select	Res.	Int. ena
6	3	1	9	4	1	2	1	4	1

Figure 23-135. CNT Control Field (C31:C0)

31 29	28 27	26	25	24 0
Res.	Request type	Control	Res.	Max Count
3	2	1	1	25

Figure 23-136. CNT Data Field (D31:D0)

31 7	6 0
Data	Reserved
25	7

Cycles

One or two

One cycle (time mode), two cycles (angle mode)

Register modified

Selected register (A, B or T)

Description

This instruction defines a virtual timer. The counter value stored in the data field [D31:7] is incremented unconditionally on each execution of the instruction when in time mode (angle count bit [P8] = 0). When the count reaches the maximum count specified in the control field, the counter is reset. It takes one cycle in this mode.

In angle mode (angle count bit [P8] = 1), CNT needs data from the software angle generator (SWAG). When in angle count mode the angle increment value will be 0 or 1. It takes two cycles in this mode.

angle_count

Specifies when the counter is incremented. A value of ON causes the counter value to be incremented only if the new angle flag is set (NAF_global = 1). A value of OFF increments the counter each time the CNT instruction is executed.

Default value for this field is OFF.

comp

When set to EQ the counter is reset, when it is equal to the maximum count.

When set to GE the counter is reset, when it is greater or equal to the maximum count.

Default: GE.

irq

ON generates an interrupt when the counter overflows to zero. The interrupt is not generated until the data field is reset to zero. If irq is set to OFF, no interrupt is generated.

Default: OFF.

max

Specifies the 25-bit integer value that defines the maximum count value allowed in the data field. When the count in the data field is equal to max, the data field is reset to 0 and the Z system flag is set to 1.

data

Specifies the 25-bit integer value serving as a counter.

Default: 0.

Execution

```
Z = 0;

If (Angle Count (bit P8 == 1))
{
   If (NAF_global == 0)
   {
      Selected register = immediate data field;
      Jump to Next Program Address;
   }
   else
   {
      If ((Immediate Data Field + Angle Increment) >= Max count)
      {
         Z = 1;
         Selected register = ((Immediate Data Field + Angle Inc.) - Max count);
         Immediate Data Field = ((Immediate Data Field + Angle Inc.) - Max count);

         If (Interrupt Enable == 1) HETFLG[n] = 1;      /* n depends on address */
         If ([C28:C27] == 01) Generate request on request line [P25:P23];
         If ([C28:C27] == 11) Generate quiet request on request line [P25:P23];
      }
      else
      {
         Selected register = Immediate Data Field + Angle Increment;
         Immediate Data Field = Immediate Data Field + Angle Increment;
      }
   }
}

else if(Time mode (bit P8 == 0))
{
   If [(P5==0) AND (Immediate Data Field == Max count)]
   OR [(P5==1) AND (Immediate Data Field >= Max count)]
   {
       Z = 1;
       Selected register = 00000;
       Immediate Data Field = 00000;

       If (Interrupt Enable == 1) HETFLG[n] = 1;      /* n depends on address */
       If ([C28:C27] == 01) Generate request on request line [P25:P23];
       If ([C28:C27] == 11) Generate quiet request on request line [P25:P23];
   }
   else
   {
       Selected register = Immediate Data Field + 1;
       Immediate Data Field = Immediate Data Field + 1;
   }
}

Jump to Next Program Address;
```

The specific interrupt flag that is triggered depends on the address from which the instruction is executed, see Section 23.2.7.

23.6.3.9 DADM64 (Data Add Move 64)

Syntax

```
DADM64 {
[brk={OFF | ON}]
[next={label | 9-bit unsigned integer}]
remote={label | 9-bit unsigned integer}
[request={NOREQ | GENREQ | QUIET}]
[control={OFF | ON}]
[en_pin_action={OFF | ON}]
[cond_addr={label | 9-bit unsigned integer}]
[pin={pin number}]
comp_mode={ECMP | SCMP | MCMP1 | MCMP2}
[action={CLEAR | SET | PULSELO | PULSEHI}]
[reg={A | B | R | S | T | NONE}]
[irq={OFF | ON}]
[data={25-bit unsigned integer]
[hr_data= {7-bit unsigned integer}]
}
```

-or-

Syntax

```
DADM64 {
[brk={OFF | ON}]
[next={label | 9-bit unsigned integer}]
remote={label | 9-bit unsigned integer}
cntl_val={29-bit unsigned integer}
data={25-bit unsigned integer}
[hr_data= {7-bit unsigned integer}]
}
```

Figure 23-137. DADM64 Program Field (P31:P0)

31 26	25 23	22	21 13	12 9	8 0
0	Reserved	BRK	Next program address	0010	Remote Address
6	3	1	9	4	9

Figure 23-138. DADM64 Control Field (C31:C0)

31 29	28 27	26	25 23	22	21 16
Reserved	Request type	Control	Reserved	En. pin action	Conditional address
3	2	1	3	1	9

15 13	12 8	7	6 5	4 3	2 1	0
Conditional address	Pin select	Ext Reg	Comp. mode	Action	Register select	Int. ena
9	5	1	2	2	2	1

Figure 23-139. DADM64 Data Field (D31:D0)

31 7	6 0
Data	HR Data
25	7

Cycles Two

Register modified Register T (implicitly)

Description This instruction modifies the data field and the control field at the remote address. The remote data field value is not just replaced, but is added with the DADM64 data field.

DADM64 has two distinct syntaxes. In the first syntax, bit values may be set by assigning a value to each of the control fields. This syntax is convenient for modifying control fields that are arranged similarly to the format of the DADM64 control field. A second syntax, in which the entire 29-bit control field is specified by the cntl_val field, is convenient when the remote control field is dissimilar to the DADM64 control field. Either syntax may be used, but you must use one or the other but not a combination of syntaxes.

Figure 23-140 shows the DADM64 add and move operation.

Figure 23-140. DADM64 Add and Move Operation

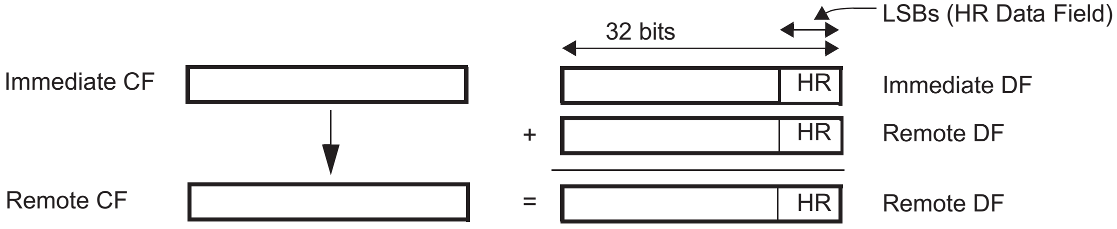


Table 23-84. DADM64 Control Field Description

request	maintains the control field for the remote instruction
control	maintains the control field for the remote instruction
en_pin_action	maintains the control field for the remote instruction
cond_addr	maintains the control field for the remote instruction

Table 23-84. DADM64 Control Field Description (continued)

pin	maintains the control field for the remote instruction
register	maintains the control field for the remote instruction
action	maintains the control field for the remote instruction
irq	maintains the control field for the remote instruction
data	Specifies the 25-bit initial value for the data field.
hr_data	Seven least significant bits of the 32 bit data field. Default: 0
cntl_val	Specifies the 29 least significant bits of the Control field.

Execution

```
Remote Data Field = Remote Data Field + Immediate Data Field;
Register T = Immediate Data Field;
Remote Control Field = Immediate Control Field;
Jump to Next Program Address;
```

23.6.3.10 DJZ (Decrement and Jump if Zero)

DJNZ is also a supported syntax. The functionality of the two instruction names is identical.

Syntax

```
DJZ {
[brk={OFF | ON}]
[next={label | 9-bit unsigned integer}]
[reqnum={3-bit unsigned integer}
[request={NOREQ | GENREQ | QUIET}]
[control={OFF | ON}]
[cond_addr={label | 9-bit unsigned integer}]
[reg={A | B | T | NONE}]
[irq={OFF | ON}]
[data={25-bit unsigned integer]
}
```

Figure 23-141. DJZ Program Field (P31:P0)

31 26	25 23	22	21 13	12 9	8	7 6	5 0
0	Request Number	BRK	Next program address	1010	Res.	10	Reserved
6	3	1	9	4	1	2	6

Figure 23-142. DJZ Control Field (C31:C0)

31 29	28 27	26	25 22	21 16
Reserved	Request type	Control	Reserved	Conditional address
3	2	1	4	9

15 13	12 8 7 3	2 1	0
Conditional address	Reserved	Register select	Int. ena
9	10	2	1

Figure 23-143. DJZ Data Field (D31:D0)

31 7	6 0
Data	Reserved
25	7

Cycles One

Register modified Selected register (A, B, or T)

Description This instruction defines a virtual down counter used for delayed execution of certain instructions (to generate minimum on/off times). When DJZ is executed with counter value not zero, the counter value is decremented. If the counter value is zero, the counter remains zero until it is reloaded with a non-zero value. The program flow can be modified when down counter value is zero by using the conditional address.

cond_addr This field is not optional for the DJZ instruction.

irq ON generates an interrupt when the data field reaches zero. No interrupt is generated when the bit is OFF.

Default: OFF.

data Specifies the 25-bit integer value used as a counter. This counter is decremented each time the DJZ instruction is executed until the counter reaches 0.

Default: 0.

Execution

```
If (Data != 0)
{
   Data = Selected register = Data - 1;
   Jump to Next Program Address;
}
else
{
   Selected register = 000000h;

   If (Interrupt Enable == 1) HETFLG[n] = 1;      /* n depends on address */
   If ([C28:C27] == 01) Generate request on request line [P25:P23];
   If ([C28:C27] == 11) Generate quiet request on request line [P25:P23];

   Jump to conditional Address;
}
```

The specific interrupt flag that is triggered depends on the address from which the instruction is executed, see Section 23.2.7.

23.6.3.11 ECMP (Equality Compare)

Syntax

```
ECMP {
[brk={OFF | ON}]
[next={label | 9-bit unsigned integer}]
[reqnum={3-bit unsigned integer}
[request={NOREQ | GENREQ | QUIET}]
[hr_lr={HIGH | LOW}]
[angle_comp={OFF | ON}]
[control={OFF | ON}]
[en_pin_action={OFF | ON}]
[cond_addr={label | 9-bit unsigned integer}]
pin={pin number}
[action={CLEAR | SET | PULSELO | PULSEHI}]
[reg={A | B | R | S| T | NONE}]
[irq={OFF | ON}]
[data={25-bit unsigned integer]
[hr_data={7-bit unsigned integer}]
}
```

Figure 23-144. ECMP Program Field (P31:P0)

31 26	25 23	22	21 13	12 9	8	7	6 0
0	Request Number	BRK	Next program address	0000	hr_lr	Angle comp.	Reserved
6	3	1	9	4	1	1	7

Figure 23-145. ECMP Control Field (C31:C0)

31 29	28 27	26	25 23	22	21 16
Reserved	Request type	Control	Reserved	En. pin action	Conditional address
3	2	1	3	1	9

15 13	12 8	7	6 5	4 3	2 1	0
Conditional address	Pin select	Ext Reg	00	Action	Register select	Int. ena
9	5	1	2	2	2	1

Figure 23-146. ECMP Data Field (D31:D0)

31 7	6 0
Data	HR Data
25	7

Cycles One

Register modified Register A, B, R, S or T if selected

Description ECMP can use all pins. This instruction compares a 25-bit data value stored in the data field (D31–D7) to the value stored in the selected ALU register (A, B, R, S, or T). Register select encoding can be found in Section 23.6.2.

If R, S, or T registers are selected, and if the 25-bit data field matches, ECMP updates the register with the 32-bit value (D31-D0).

If the hr_lr bit is cleared, the pin action will occur after a high resolution delay from the next loop resolution clock. If the hr_lr bit is set, the delay is ignored. This delay is programmed in the data field (D6–D0).

The behavior of the pins is governed by the four action options in bits C4:C3. ECMP uses the zero flag to generate opposite pin action (synchronized to the loop resolution clock).

angle_comp Determines if an angle compare is performed. A value of ON causes the comparison to be performed only if the new angle flag is set (NAF = 1). If OFF is specified, the compare is then performed regardless of the state of the new angle flag.

Default: OFF.

irq Specifies whether or not an interrupt is generated. A value of ON sends an interrupt if register and data field values are equivalent. If OFF is selected, no interrupt is generated.

Default: OFF.

data Specifies the value for the data field. This value is compared with the selected register.

hr_data Specifies the HR delay.

Default: 0.

Execution

```
If (Angle Comp. bit == 0 OR (Angle Comp. bit == 1 AND NAF_global == 1))
{

   If (Selected register value == Immediate data field value)
   {

       If (hr_lr bit == 0)
       {
          If (Enable Pin action == 1)
          {
             Selected Pin = Pin Action AT next loop resolution clock + HR delay;
          }
       }
       else
       {
          If (Enable Pin action == 1)
          {
             Selected Pin = Pin Action AT next loop resolution clock;
          }
       }

       If (Z == 1 AND Opposite action == 1)
       {
          If (Enable Pin action == 1)
          {
             Selected Pin = opposite Pin Action AT next loop resolution clock;
          }

          If (Interrupt Enable == 1) HETFLG[n] = 1;      /* n depends on address */
          If ([C28:C27] == 01) Generate request on request line [P25:P23];
          If ([C28:C27] == 11)Generate quiet request on request line [P25:P23];

          If (register R is selected) R register = Compare value (32 bit);
          If (register S is selected) S register = Compare value (32 bit);
          If (register T is selected) T register = Compare value (32 bit);

          Jump to Conditional Address;
       }
   }
   elseIf (Z == 1 AND Opposite action == 1)
   {
      If (Enable Pin action == 1)
      {
         Selected Pin = opposite Pin Action AT next loop resolution clock;
      }
      Jump to Next Program Address;
}
else  // Angle Comp. bit == 1 AND NAF_global == 0
{
   Jump to Next Program Address;
}
```

The specific interrupt flag that is triggered depends on the address from which the instruction is executed, see Section 23.2.7.

23.6.3.12 ECNT (Event Count)

Syntax

ECNT {
[brk={OFF | ON}]
[next={label | 9-bit unsigned integer}]
[reqnum={3-bit unsigned integer}
[request={NOREQ | GENREQ | QUIET}]
[control={OFF | ON}]
[prv={OFF | ON}]
[cond_addr={label | 9-bit unsigned integer}]
pin={pin number}
event={NAF | FALL | RISE | BOTH | ACCUHIGH | ACCULOW}
[reg={A | B | R| S | T | NONE}]
[irq={OFF | ON}]
[data={25-bit unsigned integer]
}

Figure 23-147. ECNT Program Field (P31:P0)

31 26	25 23	22	21 13	12 9	8	7 6	5 0
0	Request Number	BRK	Next program address	1010	Res.	01	Reserved
6	3	1	9	4	1	2	6

Figure 23-148. ECNT Control Field (C31:C0)

31 29	28 27	26	25	24 22	21 16
Reserved	Request type	Control	Prv.	Reserved	Conditional address
3	2	1	1	3	9

15 13	12 8	7	6 4	3	2 1	0
Conditional address	Pin select	Ext Reg	Event	Res.	Register select	Int. ena
9	5	1	3	1	2	1

Figure 23-149. ECNT Data Field (D31:D0)

31 7	6 0
Data	Reserved
25	7

Cycles One cycle

Register modified Selected Register (A, B, R, S, T or none)

Description This instruction defines a specialized 25-bit virtual counter used as an event counter or pulse accumulator (see Table 23-85). The counter value is stored in the data field [D31:D7] and the selected register. If one of the 32-bit registers (R,S,T) is selected, the 25 bit count value is stored left justified in the register with zeros in the seven least significant bits.

When an event count condition is specified, the counter value is incremented on a pin edge condition or on the NAF condition (NAF is defined in ACNT). This instruction can be used with all pins.

event The event that triggers the counter.

Table 23-85. Event Encoding Format for ECNT

Event	C6	C5	C4	Count Conditions	Mode	Int. Available
NAF	0	0	0	NAF flag is Set	Angle counter	Y
FALL	0	0	1	Falling edge on selected pin	Event counter	Y
RISE	0	1	0	Rising edge on selected pin	Event counter	Y
BOTH	0	1	1	Rising and Falling edge on selected pin	Event counter	Y
ACCUHIGH	1	0	-	while pin is high level	Pulse accumulation	N
ACCULOW	1	1	-	while pin is low level	Pulse accumulation	N

irq ON generates an interrupt when event in counter mode occurs. No interrupt is generated with OFF.

Default: OFF.

data 25-bit integer value serving as a counter.

Default: 0.

Execution

```
If (event occurs)
{
   If (Register A or B Selected) {
      Selected register = Immediate Data Field + 1;
   }

   If (Register R, S or T Selected)
   {
      Selected register[31:7] = Immediate Data Field + 1;
      Selected register[6:0] = 0;
   }

   Immediate Data Field = Immediate Data Field + 1;

   If (Interrupt Enable == 1) HETFLG[n] = 1;      /* n depends on address */
   If ([C28:C27] == 01) Generate request on line [P25:P23];
   If ([C28:C27] == 11) Generate quiet request on line [P25:P23];

   Jump to Conditional Address;
}
else
{
    Jump to Next Program Address;
}

Prv bit = Current Logic (Lx) value of selected pin; (Always executed)
```

The specific interrupt flag that is triggered depends on the address from which the instruction is executed, see Section 23.2.7.

23.6.3.13 MCMP (Magnitude Compare)

Syntax

```
MCMP {
[brk={OFF | ON}]
[next={label | 9-bit unsigned integer}]
[reqnum={3-bit unsigned integer}
[request={NOREQ | GENREQ | QUIET}]
[hr_lr={LOW |HIGH}]
[angle_comp={OFF | ON}]
[savesub={OFF | ON}]
[control={OFF | ON}]
[en_pin_action={OFF | ON}]
[cond_addr={label | 9-bit unsigned integer}]
pin={pin number}
order={REG_GE_DATA | DATA_GE_REG}
[action={CLEAR | SET | PULSELO | PULSEHI}]
reg={A | B | R | S | T | NONE}
[irq={OFF | ON}]
[data={25-bit unsigned integer]
[hr_data={7-bit unsigned integer}]
}
```

Figure 23-150. MCMP Program Field (P31:P0)

31 26	25 23	22	21 13	12 9	8	7	6	5	4 0
0	Request Number	BRK	Next program address	0000	hr_lr	Angle comp.	Res.	Save sub.	Res.
6	3	1	9	4	1	1	1	1	5

Figure 23-151. MCMP Control Field (C31:C0)

31 29	28 27	26	25 23	22	21 16
Reserved	Request type	Control	Reserved	En. pin action	Conditional address
3	2	1	3	1	9

15 13	12 8	7	6	5	4 3	2 1	0
Conditional address	Pin select	Ext Reg	1	Order	Action	Register select	Int. ena
9	5	1	1	1	2	2	1

Figure 23-152. MCMP Data Field (D31:D0)

31 7	6 0
Data	HR Data
25	7

Cycles One

Register modified T (if save sub bit P[5] is set)

Description

This instruction compares the magnitude of the 25-bit data value stored in the data field (D31-D7) and the 25-bit value stored in the selected ALU register (A, B, R, S, or T).

If the hr_lr bit is reset, pin action will occur after a delay from the next loop resolution clock. If the hr_lr bit is set, the delay is ignored. This delay is programmed in the data field (D6-D0).

When the data value matches, an output pin can be set or reset according to the pin action bit (C[4]). The pin will not change states if the enable pin action bit (C[22]) is reset.

MCMP uses the zero flag set to generate opposite pin action (synchronized to the loop resolution clock). The save sub bit (P[5]) provides the option to save the result of a subtraction into register T.

NOTE: The Difference Between Compare Values

The difference between the two data values must not exceed (2^{24}) - 1.

angle_comp

Determines whether or not an angle compare is performed. A value of ON causes the comparison to be performed only if the new angle flag is set (NAF = 1). If OFF is specified, the compare is then performed regardless of the state of the new angle flag.

Default: OFF.

savesub

When set, the comparison result is saved into the T register (upper 25 bits).

Default: OFF.

order

Specifies the order of the operands for the comparison.

Table 23-86. Magnitude Compare Order for MCMP

Order	C5	Description
REG_GE_DATA	0	Evaluates to true if the register value is greater than or equal to the data field value.
DATA_GE_REG	1	Evaluates to true if the data field value is greater than or equal to the register value.

irq

Specifies whether or not an interrupt is generated. A value of ON sends an interrupt if the compare match occurs according to the order selected. If OFF is selected, no interrupt is generated.

data

Specifies the value for the data field. This value is compared with the selected register.

hr_data

HR delay. The default value for an unspecified bit is 0.

Execution

```
If (Angle Compare P[7] == 0 OR (P[7] == 1 AND NAF_global == 1))
{
    If(  (Order C[5] == 1) AND (Data[31:7]- Selected register[31:7]) >= 0))
      OR ( (Order C[5] == 0) AND Selected register[31:7] - Data[31:7]) >= 0))
    {
       If (Order C[5] == 1 AND Save subtract P[5] == 1)
       {
          Register T[31:7] = Data[31:7] - Selected register[31:7];
          Register T[6:0] = 0;
       }

       If (Order C[5] == 0 AND Save subtract P[5] == 1)
       {
          Register T[31:7] = Selected register[31:7] - Data[31:7];
          Register T[6:0] = 0;
       }

       If (Enable Pin Action C[22] == 1)
       {
          If (hr_lr P[8] = 0) {
             Schedule Action on Selected Pin C[12:8] at start of next loop
             + HR Delay D[6:0];
          }
          else
          {
             Schedule Pin Action on Selected Pin C[12:8] at start of next loop;
          }
       }

       If (Interrupt Enable == 1) HETFLG[n] = 1;      /* n depends on address */
       If ([C28:C27] == 01) Generate request on request line [P25:P23];
       If ([C28:C27] == 11) Generate quiet request on request line [P25:P23];

       Jump to Conditional Address;
    }
    else if (Z == 1 AND Opposite Action C[3] == 1 )
    {
       If (Enable Pin Action C[22] == 1)
       {
          Schedule Opposite Pin Action on Selected Pin C[12:8] at start of next loop;
       }

       Jump to Next Program Address;
    }
    else
       Jump to Next Program Address;
}
else // Angle Comp. bit == 1 AND NAF_global == 0
   Jump to Next Program Address;
```

The specific interrupt flag that is triggered depends on the address from which the instruction is executed, see Section 23.2.7.

23.6.3.14 MOV32 (Data Move 32)

Syntax

```
MOV32 {
[brk={OFF | ON}]
[next={label | 9-bit unsigned integer}]
remote={label | 9-bit unsigned integer}
[control={OFF | ON}]
[z_cond={OFF | ON}]
[init={OFF | ON}]| ON}]
type={IMTOREG | IMTOREG&REM | REGTOREM | REMTOREG}
[reg={A | B | R | S | T | NONE}]
[data={25-bit unsigned integer]
[hr_data={7-bit unsigned integer}]
}
```

Figure 23-153. MOV32 Program Field (P31:P0)

31 26	25 23	22	21 13	12 9	8 0
0	Reserved	BRK	Next program address	0100	Remote Address
6	3	1	9	4	9

Figure 23-154. MOV32 Control Field (C31:C0)

31 27	26	25 23	22	21 16
Reserved	Control	Reserved	Z Fl. Cond.	Reserved
5	1	3	1	14

15 8	7	6	5	4 3	2 1	0
Reserved	Ext Reg	Init flag	0	Move type	Register select	Res.
14	1	1	1	2	2	1

Figure 23-155. MOV32 Data Field (D31:D0)

31 7	6 0
Data	HR Data
25	7

Cycles One or two cycles

Register modified Selected register (A, B, R, S, or T)

Description MOV32 replaces the selected ALU register and/or the data field values at the remote address location depending on the move type.

Figure 23-156 through Figure 23-159 illustrate these operations. If *no register* is selected, the move is not executed, except for configuration C4:C3 = 01, where the remote data field is written with the immediate data field value.

remote Determines the location of the remote address.

Default: Current instruction + 1.

z_cond When set to OFF the MOV32 performs the move operation specified by the move type whenever it is executed (independent on the state of the Z-Flag).

When set to ON the MOV32 performs the move operation specified by the move type only when the Z-Flag is set.

init (Optional) Determines whether or not system flags are initialized. A value of ON reinitializes the following system flags to these states:

Acceleration flag (ACF) = 0

Deceleration flag (DCF) = 1

Gap flag (GPF) = 0

New angle flag (NAF) = 0

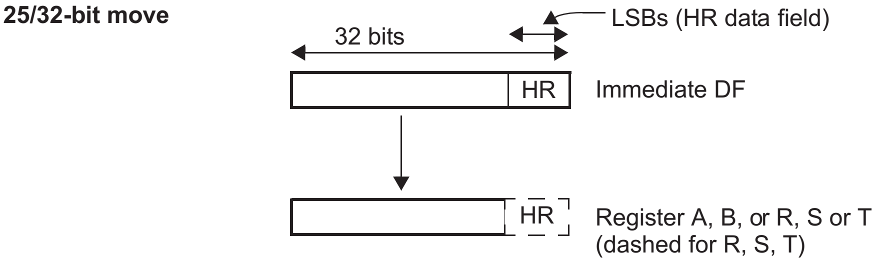
A value of OFF results in no change to the system flags.

type Specifies the move type to be executed.

Table 23-87. Move Type Encoding Selection

Move Type	C4	C3	Source	Destination(s)	Cycles
IMTOREG	0	0	Immediate data field	Register A, B, R, S, or T	1
IMTOREG&REM	0	1	Immediate data field	Remote data field and register A, B, R, S, or T	1
REGTOREM	1	0	Register A, B, R, S, or T	Remote data field	1
REMTOREG	1	1	Remote data field	Register A, B, R, S, or T	2

Figure 23-156. MOV32 Move Operation for IMTOREG (Case 00)



reg Specifies which register (A, B, T, or NONE) is involved in the move. A register (A, B, or T) must be specified for every move type except IMTOREG&REM. If *NONE* is used with move type IMTOREG&REM, the MOV32 executes a move from the immediate data field to the remote data field. If *NONE* is used with any other move type, no move is executed.

data Specifies a 25-bit integer value to be written to the remote data field or selected register.

hr_data (Optional) HR delay. The default value for an unspecified bit is 0.

Figure 23-157. MOV32 Move Operation for IMTOREG&REM (Case 01)

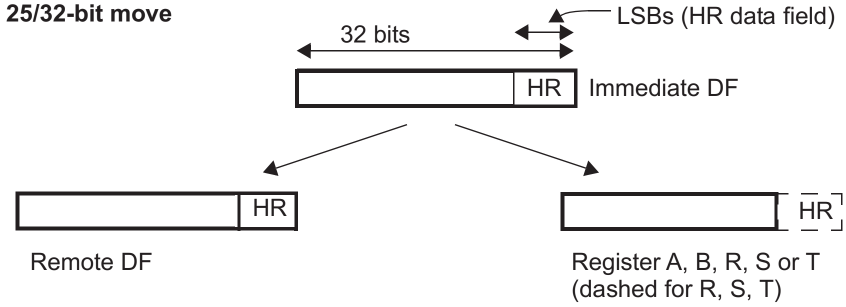


Figure 23-158. MOV32 Move Operation for REGTOREM (Case 10)

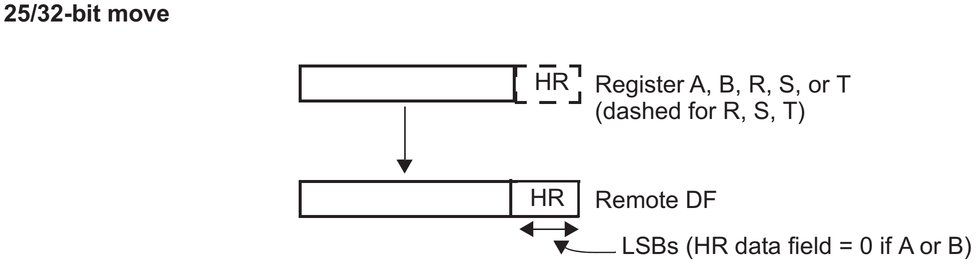
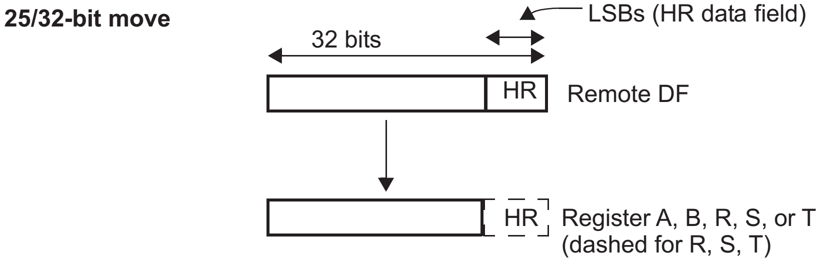


Figure 23-159. MOV32 Move Operation for REMTOREG (Case 11)



Execution

```
If [(z_cond C[22] ==0) OR ((z_cond C[22] == 1) AND (Z Flag == 1))]
{
   switch (type C[4:3])
   {
      case 00: // IMTOREG
          Selected register = Immediate Data Field;
      case 01: // IMTOREG&REM
          Selected register = Immediate Data Field;
          Remote Data Field = Immediate Data Field;
      case 10: // REGTOREM
          Remote Data Field = Selected register;
      case 11: // REMTOREG
          Selected register = Remote Data Field;
   }
}

If (Init Flag == 1)
{
   ACF = 0;
   DCF = 1;
   GPF = 0;
   NAF = 0;
}
else
  All flags remain unchanged;

Jump to Next Program Address;
```

23.6.3.15 MOV64 (Data Move 64)

Syntax

```
MOV64 {
[brk={OFF | ON}]
[next={label | 9-bit unsigned integer}]
remote={label | 9-bit unsigned integer}
[request={NOREQ | GENREQ | QUIET}]
[control={OFF | ON}]
[en_pin_action={OFF | ON}]
[cond_addr={label | 9-bit unsigned integer}]
[pin={pin number}]
comp_mode={ECMP | SCMP | MCMP1 | MCMP2}
[action={CLEAR | SET | PULSELO | PULSEHI}]
[reg={A | B | R | S | T | NONE}]
[irq={OFF | ON}]
[data={25-bit unsigned integer]
[hr_data= {7-bit unsigned integer}
}
```

-or-

Syntax

```
MOV64 {
[brk={OFF | ON}]
[next={label | 9-bit unsigned integer}]
remote={label | 9-bit unsigned integer}
cntl_val={29-bit unsigned integer}
[data={25-bit unsigned integer]
[hr_data= {7-bit unsigned integer}
}
```

Figure 23-160. MOV64 Program Field (P31:P0)

31 26	25 23	22	21 13	12 9	8 0
0	Reserved	BRK	Next program address	0001	Remote Address
6	3	1	9	4	9

Figure 23-161. MOV64 Control Field (C31:C0)

31 29	28 27	26	25 23	22	21 16
Reserved	Request type	Control	Reserved	En. pin action	Conditional address
3	2	1	3	1	9

15 13	12 8	7	6 5	4 3	2 1	0
Conditional address	Pin select	Ext Reg	Comp. mode	Action	Register select	Int. ena
9	5	1	2	2	2	1

Figure 23-162. MOV64 Data Field (D31:D0)

31 7	6 0
Data	HR Data
25	7

Cycles One

Register modified None

Description This instruction modifies the data field and the control field at the remote address.

MOV64 has two distinct syntaxes. In the first syntax, bit values may be set by assigning a value to each of the control fields. This syntax is convenient for modifying control fields that are arranged similarly to the format of the MOV64 control field. A second syntax, in which the entire 29-bit control field is specified by the cntl_val field, is convenient when the remote control field is dissimilar to the MOV64 control field. Either syntax may be used, but you must use one or the either but not a combination of syntaxes. See Figure 23-163.

Figure 23-163. MOV64 Move Operation

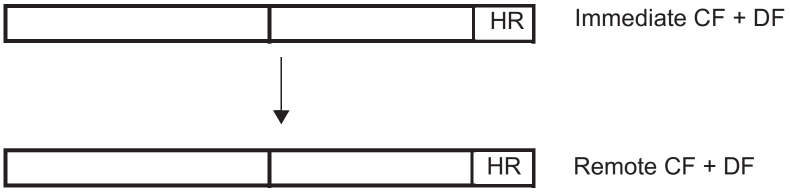


Table 23-88. MOV64 Control Field Descriptions

request	Maintains the control field for the remote instruction.
control	Maintains the control field for the remote instruction.
en_pin_action	Maintains the control field for the remote instruction.
cond_addr	Maintains the control field for the remote instruction.
pin	Maintains the control field for the remote instruction.
register, ext reg	Maintains the control field for the remote instruction.
comp_mode	Selects the comparison mode type to be used by the remote instruction.

Table 23-88. MOV64 Control Field Descriptions (continued)

action	Maintains the control field for the remote instruction.
irq	Maintains the control field for the remote instruction.
data	Specifies the 25-bit initial count value for the data field. If omitted, the field defaults to 0.
hr_data	(Optional) HR delay. The default value for an unspecified bit is 0.

Table 23-89. Comparison Type Encoding Format

comp_mode	C[6]	C[5]	MCMP Order
ECMP	0	0	
SCMP	0	1	
MCMP1	1	0	REG_GE_DATA
MCMP2	1	1	DATA_GE_REG

Execution

```
Remote Data Field = Immediate Data Field;
Remote Control Field = Immediate control Field;
Jump to Next Program Address;
```

23.6.3.16 PCNT (Period/Pulse Count)

Syntax

```
PCNT {
[hr_lr={HIGH | LOW}]
[brk={OFF | ON}]
[next={label | 9-bit unsigned integer}]
[reqnum={3-bit unsigned integer}
[request={NOREQ | GENREQ | QUIET}]
[irq={OFF | ON}]
type={FALL2RISE | RISE2FALL | FALL2FALL | RISE2RISE}
pin={pin number}
[control={OFF | ON}]
[prv={OFF | ON}]
[period={25-bit unsigned integer}]
[data={25-bit unsigned integer]
[hr_data= {7-bit unsigned integer}
}
```

Figure 23-164. PCNT Program Field (P31:P0)

31 26	25 23	22	21 13	12 9	8	7 6	5	4 0
0	Request Number	BRK	Next program address	0111	Int. ena	Type select	hr_lr	Pin select
6	3	1	9	4	1	2	1	5

Figure 23-165. PCNT Control Field (C31:C0)

31 29	28 27	26	25	24 0
Res.	Request type	Control	Prv.	Period Count
3	2	1	1	25

Figure 23-166. PCNT Data Field (D31:D0)

31 7	6 0
Data	HR Data
25	7

Cycles One

Register modified Register A

Description This instruction detects the edges of the external signal at loop start and measures its period or pulse duration. The counter value stored in the control field C[24:0] and in the register A is incremented each N2HET loop. PCNT uses the HR structure on the pin to measure an HR period/pulse count value.

hr_lr (Optional) Specifies whether the PCNT instruction captures the HR delay into the HR data field on the selected edge condition. If hr_lr is 0 (HIGH) then PCNT captures the HR delay. if hr_lr is 1 (LOW) then PCNT only captures at loop resolution.

irq (Optional) Specifies whether or not an interrupt is generated. A value of ON sends an interrupt when a new value is captured. If OFF is selected, no interrupt is generated.

type (Optional) Determines the type of counter that is implemented.

Table 23-90. Counter Type Encoding Format

	P7	P6	Period/Pulse Select	Reset On	Capture On
FALL2RISE	0	0	Count low-pulse duration on selected pin	Falling edge	Rising edge
RISE2FALL	0	1	Count high-pulse duration on selected pin	Rising edge	Falling edge
FALL2FALL	1	0	Count period between falling edges on selected pin	Falling edge	Falling edge
RISE2RISE	1	1	Count period between rising edges on selected pin	Rising edge	Rising edge

period Specifies the 25-bit integer value that holds the counter value. The counter value is also stored in register A.

Default: 0.

data 25-bit integer representing the last captured counter value.

Default: 0.

hr_data HR delay.

Default: 0.

If *period-measure* is selected, PCNT captures the counter value into the period/pulse data field [D31:D7] on the selected edge. The HR structure provides HR capture field [D6:D0]. The counter value [C24:C0] is reset on the same edge. The captured period value is a 32-bit value.

If *pulse-measure* is selected, PCNT captures the counter value into the period/pulse count field [D31:D7] on the selected edge. The HR structure provides HR capture field [D6:D0]. The counter value [C24:C0] is reset on the next opposite edge. The captured pulse value is a 32-bit value.

When the overflow count (all 1’s in the counter value) is reached, PCNT stops counting until the next reset edge is detected.

Note: For FALL2FALL/RISE2RISE, the user should always discard the first interrupt/HTU request if interrupt/request are enabled before HET_ON. For both the types, reset edge and capture edge are the same and the interrupt or HTU request is triggered on capture edge (which is nothing but the reset edge). Once the execution unit is enabled, the first edge generates an interrupt but the value of the counter is of no use as this is not the period between 2 edges. So first edge after turning on N2HET is used mainly for resetting the counter and start the period count.

Execution

```
Z = 0;

If (Period C[24:0] != 1FF_FFFFh) {
    Period C[24:0] = Period C[24:0] + 1;
}

Register A = Period C[24:0];

If (specified capture edge detected on selected pin)
{
    Z = 1;

    If (Period value != 1FF_FFFFh)
    {
       HR Capture Value = selected HR counter;
    }
    else
    {
       HR Capture Value = 7Fh;
    }

    If (Interrupt Enable == 1) HETFLG[n] = 1;      /* n depends on address */
    If ([C28:C27] == 01) Generate request on request line [P25:P23];
    If ([C28:C27] == 11) Generate quiet request on request line [P25:P23];
}

If (specified reset edge detected on selected pin)
{
    Period value = 0000000h;
}

Prv bit = Current Logic (Lx) value of selected pin;

Jump to Next Program Address;
```

The specific interrupt flag that is triggered depends on the address from which the instruction is executed, see Section 23.2.7.

23.6.3.17 PWCNT (Pulse Width Count)

Syntax

```
PWCNT {
[brk={OFF | ON}]
[next={label | 9-bit unsigned integer}]
[reqnum={3-bit unsigned integer}
[request={NOREQ | GENREQ | QUIET}]
[hr_lr={HIGH | LOW}]
[control={OFF | ON}]
[cond_addr={label | 9-bit unsigned integer}
[en_pin_action={OFF | ON}]
pin ={pin number}
[action={CLEAR | SET | PULSELO | PULSEHI}]
[reg={A | B | T | NONE}]
[irq={OFF | ON}]
[data={25-bit unsigned integer]
[hr_data={7-bit unsigned integer}]
}
```

Figure 23-167. PWCNT Program Field (P31:P0)

31 26	25 23	22	21 13	12 9	8	7 6	5 0
0	Request Number	BRK	Next program address	1010	hr_lr	11	Reserved
6	3	1	9	4	1	2	6

Figure 23-168. PWCNT Control Field (C31:C0)

31 29	28 27	26	25 23	22	21 16
Reserved	Request type	Control	Reserved	En. pin action	Conditional address
3	2	1	3	1	9

15 13	12 8	7 5	4 3	2 1	0
Conditional address	Pin select	Reserved	Action	Register select	Int. ena
9	5	3	2	2	1

Figure 23-169. PWCNT Data Field (D31:D0)

31 7	6 0
Data	HR Data
25	7

Cycles One

Register modified Selected register (A, B or T)

Description This instruction defines a virtual timer used to generate variable length pulses. The counter value stored in the data field is decremented unconditionally on each timer resolution until it reaches zero, and it then stays at zero until it is reloaded with a non-zero value.

The specified pin action is performed as long as the count after count value is decremented is greater than 0. The opposite pin action is performed when the count after decrement just reaches 0.

If the hr_lr bit is reset, the opposite pin action will be taken after a HR delay from the next loop resolution clock. If the hr_lr bit is set, the delay is ignored. This delay is programmed in bits [D6:D0].

irq ON generates an interrupt when the data field value reaches 0. No interrupt is generated for OFF.

Default: OFF.

data 25-bit integer value serving as a counter.

hr_data HR delay.

Default: 0.

Execution

```
If (Data field value == 0)
{
   Selected register = 0;
   Jump to Next Program Address;
}

If (Data field value > 1)
{
   Selected register = Data field value - 1;
   Data field value = Counter value - 1;

   If (Enable Pin action == 1)
   {
      Selected Pin = Pin Action AT next loop resolution clock;
   }

   Jump to Next Program Address;
}

If (Data field value == 1)
{
   Selected register = 0000000h;
   Data field value = 0000000h;

   If (Opposite action == 1)
   {
      If (hr_lr bit == 0)
      {
         If (Enable Pin action == 1)
         {
              Selected Pin = Opposite level of Pin Action AT next loop resolution clock
              + HR delay;
         }
      }
      else
      {
         If (Enable Pin action == 1)
         {
            Selected Pin = Opposite level of Pin Action AT next loop
            resolution clock;
         }
      }

      If (Interrupt Enable == 1) HETFLG[n] = 1;      /* n depends on address */
      If ([C28:C27] == 01) Generate request on request line [P25:P23];
      If ([C28:C27] == 11) Generate quiet request on request line [P25:P23];

   }

   Jump to Conditional Address
}
```

The specific interrupt flag that is triggered depends on the address from which the instruction is executed, see Section 23.2.7.

23.6.3.18 RADM64 (Register Add Move 64)

Syntax

```
RADM64 {
[brk={OFF | ON}]
[next={label | 9-bit unsigned integer}]
remote={label | 9-bit unsigned integer}
[request={NOREQ | GENREQ | QUIET}]
[control={OFF | ON}]
[en_pin_action={OFF | ON}]
[cond_addr={label | 9-bit unsigned integer}]
[pin={pin number}]
comp_mode={ECMP | SCMP | MCMP1 | MCMP2}
[action={CLEAR | SET | PULSELO | PULSEHI}]
[reg={A | B | R | S | T | NONE}]
[irq={OFF | ON}]
[data={25-bit unsigned integer]
[hr_data= {7-bit unsigned integer}
}
```

-or-

Syntax

```
RADM64 {
[brk={OFF | ON}]
[next={label | 9-bit unsigned integer}]
remote={label | 9-bit unsigned integer}
cntl_val={29-bit unsigned integer}
[data={25-bit unsigned integer]
[hr_data= {7-bit unsigned integer}
}
```

Figure 23-170. RADM64 Program Field (P31:P0)

31 26	25 23	22	21 13	12 9	8 0
0	Reserved	BRK	Next program address	0011	Remote Address
6	3	1	9	4	9

Figure 23-171. RADM64 Control Field (C31:C0)

31 29	28 27	26	25 23	22	21 16
Reserved	Request type	Control	Reserved	En. pin action	Conditional address
3	2	1	3	1	9

15 13	12 8	7	6 5	4 3	2 1	0
Conditional address	Pin select	Ext Reg	Comp. mode	Action	Register select	Int. ena
9	5	1	2	2	2	1

Figure 23-172. RADM64 Data Field (D31:D0)

31 7	6 0
Data	HR Data
25	7

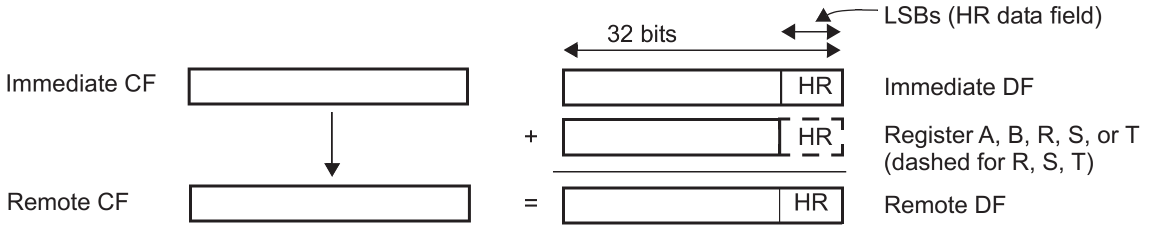
Cycles Normally One Cycle. Two cycles if writing to remote address that is also the next address.

Register modified None

Description This instruction modifies the data field, the HR data field and the control field at the remote address. The advantage over DADM64 is that It executes one cycle faster. In case the R, S, or T register is selected, the addition is a 32-bit addition. The table description shows the bit encoding for determining which ALU register is selected.

RADM64 has two distinct syntaxes. In the first syntax, bit values may be set by assigning a value to each of the control fields. This syntax is convenient for modifying control fields that are arranged similar to the format of the RADM64 control field. A second syntax, in which the entire 29-bit control field is specified by the cntl_val field, is convenient when the remote control field is dissimilar from the RADM64 control field. Either syntax may be used, but you must use one or the either but not a combination of syntaxes. See Figure 23-173.

Figure 23-173. RADM64 Add and Move Operation



comp_mode Selects the comparison mode type to be used.

Table 23-91. Comparison Type Encoding Format

comp_mode	C[6]	C[5]	MCMP Order
ECMP	0	0	
SCMP	0	1	
MCMP1	1	0	REG_GE_DATA
MCMP2	1	1	DATA_GE_REG

Table 23-92. RADM64 Control Field Descriptions

request	Maintains the control field for the remote instruction.
Control	Maintains the control field for the remote instruction.
en_pin_action	Maintains the control field for the remote instruction.
cond_addr	Maintains the control field for the remote instruction.
pin	Maintains the control field for the remote instruction.
register	Maintains the control field for the remote instruction.
action	Maintains the control field for the remote instruction.
irq	Maintains the control field for the remote instruction.
data	Specifies the 25-bit initial value for the data field. If omitted, the field defaults to 0.
hr_data	Seven least significant bits of the 32-bit data field. Default: 0.
cntl_val	Specifies the 29 least significant bits of the Control field.

Execution

```
Remote Data Field = Selected register + Immediate Data Field (including HR field);
Remote Control Field = Immediate Control Field;
Jump to Next Program Address;
```

23.6.3.19 RCNT (Ratio Count)

Syntax

RCNT {
[brk={OFF | ON}]
[next={label | 9-bit unsigned integer}]
[control={OFF | ON}]
divisor={25-bit unsigned integer}
[data={25-bit unsigned integer]
}

Figure 23-174. RCNT Program Field (P31:P0)

31 26	25 23	22	21 13	12 9	8	7 6	5 4	3 1	0
0	Reserved	BRK	Next program address	1010	Res.	00	Step width	Res.	1
6	3	1	9	4	1	2	2	3	1

Figure 23-175. RCNT Control Field (C31:C0)

31 27	26	25	24 0
Reserved	Control	Res.	Divisor
5	1	1	25

Figure 23-176. RCNT Data Field (D31:D0)

31 7	6 0
Data	Reserved
25	7

Cycles Two Cycles (One Cycle if T=0)

Register modified None

Description RCNT is used with other instructions to convert an input period measurement T_{Input} to the form of (Equation 31) where the input period is expressed as a fraction of a reference period $T_{Reference}$.

$$T_{Input} = T_{Reference} \bullet \left(\frac{N}{M}\right) \tag{31}$$

RCNT computes the numerator N of (Equation 31). The denominator M of (Equation 31) is a constant that is of interest. For example, choosing M = 100 allows the input period to be expressed as a percentage (%) of the reference period. Note that if $T_{Input} > T_{Reference}$, then RCNT will return N > M ; which would be correct if, for example, the input pulse period is 110% of the reference pulse period.

RCNT expects that register T is loaded with the value of $T_{Reference}$. The input period T_{Input} is determined by counting the number of loop resolution periods between edges on the input pin. This information is conveyed through the Z flag from a PCNT instruction that precedes the RCNT instruction.

The divisor field of the RCNT instruction should be chosen as: `Divisor = M ·        lr` , where M is the desired denominator from (Equation 31) and lr is the loop resolution prescale value.

An example N2HET program that makes use of the RCNT instruction is:

```
L0: MOV32 { remote=dummy,type=IMTOREG,reg=T,data=0x8,hr_data=0};

L1: PCNT { hr_lr=HIGH,brk=OFF,type=FALL2FALL,pin=0};

L2: RCNT { divisor=320,data=0x4};

L3: BR {cond_addr=L5, event = Z}

L4: ADC { src1=ZERO,src2=IMM,dest=IMM,next=L0,data=0,hr_data=0};

L5: ADD { src1=REM,src2=ZERO,dest=IMM,remote=L4,data=0,hr_data=0};

L6: ADD { src1=ZERO,src2=ZERO,dest=NONE,rdest=REM,
           next=L0,remote=L4,data=0,hr_data=0};

dummy
```

In this small program an input signal on pin 0 is measured both in terms of absolute cycles by the PCNT instruction at L1 and as in 1/10ths of the reference period by the RCNT instruction at L2. In this example the reference period is a constant 0x400 cycles; this value is loaded into register T by the MOV32 instruction at L0. (0x400 is data=8, hr_data=0)

RCNT follows PCNT and is initialized to a working count of T/2 (0x200) whenever the PCNT instruction detects a falling edge on pin 0. Between falling edges on pin0, RCNT accumulates counts 10x faster than PCNT; so that the working data field of RCNT will reach the reference value of 0x400 in 1/10th the time that a PCNT instruction would. Each time the RCNT instruction passes the reference value, it sets the carry out flag and subtracts the reference value from the working count. By accumulating carry-outs from RCNT, the add with carry instruction at L4 effectively counts in increments of 1/10th of the reference period. Note that the divisor value 320 is 10 times 32; this assumes lr=32.

When the next falling edge is detected on pin 0, PCNT sets the Z flag and the RCNT instruction resets again to the initial data field of T/2. RCNT does not modify the Z flag, so that the branch instruction at L3 can execute instructions at L5, L6 instead of L4. The instructions at L5 and L6 capture the final result from L4 and reset the ADC instruction at L4 to zero for the start of the next period measurement.

Execution

```
If (register T[31:0] != 00000000h)
{
   C = 0;

   If (Z == 0)
   {
       Data Field[31:0] = Data Field[31:0] + Divisor[24:0];

       If (Data Field[31:0] >= Reg T[31:0])
       {
          Data Field[31:0]=Data Field[31:0] - Reg T[31:0];
          C = 1;
       }
   }
   else
   {
      Data Field[31:0] = T[31:0] >> 1; /* T/2 */
   }
}
Jump to Next Program Address;
```

23.6.3.20 SCMP (Sequence Compare)

Syntax

```
SCMP {
[brk={OFF | ON}]
[next={label | 9-bit unsigned integer}]
[reqnum={3-bit unsigned integer}
[request={NOREQ | GENREQ | QUIET}]
[control={OFF | ON}]
[en_pin_action={OFF | ON}]
cond_addr={label | 9-bit unsigned integer}
pin ={pin number}
[action={CLEAR | SET}]
[restart={OFF | ON}]
[irq={OFF | ON}]
[data={25-bit unsigned integer]
}
```

Figure 23-177. SCMP Program Field (P31:P0)

31 26	25 23	22	21 13	12 9	8 0
0	Request Number	BRK	Next program address	0000	Reserved
6	3	1	9	4	9

Figure 23-178. SCMP Control Field (C31:C0)

31 29	28 27	26	25	24 23	22	21 16
Reserved	Request type	Control	Cout prv	Reserved	En. pin action	Conditional address
3	2	1	1	2	1	9

15 13	12 8	7	6 5	4	3 2	1	0
Conditional address	Pin select	Res.	01	Action	Reserved	Restart enable	Int. ena
9	5	1	2	1	2	1	1

Figure 23-179. SCMP Data Field (D31:D0)

31 7	6 0
Data	Reserved
25	7

Cycles One

Register modified Register T (implicitly)

Description This instruction alternately performs angle- and time-based operations to generate pulse sequences, using the angle referenced time base. These pulse sequences last for a relative duration using a free running time base. Generally, register B holds the angle values and register A holds the time values. Bit 0 of the conditional address field (C13) specifies whether the instruction is operating in angle or time operation mode.

When the compared values match in angle mode, a pin can be set or reset according to the pin action bit (C4). The pin does not change states if the enable pin action bit (C22) is reset.

The restart enable bit (C1) provides the option to unconditionally restart a sequence using the X-flag bit of ACMP.

restart If restart is set to ON and the X flag = 1, the assembler writes a value of 1 into the immediate index field, writes the value in register A into the immediate data field, and jumps to the next program address. The X flag is set or cleared by the ACMP instruction. If restart is set to OFF, the X flag is ignored; no special action is performed.

Default: OFF.

irq ON generates an interrupt if the compare match occurs in angle mode. No interrupt is generated when the field is OFF.

Default: OFF.

data Specifies the 25-bit compare value.

cond_addr Since the LSB of the conditional address is used to select between time mode and angle mode, and since the conditional address is taken only in time mode, the destination for the conditional address must be odd.

Execution

```
If (Data field value <= Selected register value) Cout = 0; else Cout = 1;

If (Restart Enable == 1 AND X == 1)
{
   C13 = 1;
   Immediate Data Field = Register A;
   Cout = 0;
   Jump to Next Program Address;
}

If (Angle Mode (C13 == 0) AND ((Restart En. == 1 AND X == 0) OR Restart En. == 0))
{
   If (Z == 0 AND (Register B value - Angle Inc. < Data field value) AND Cout == 0) OR
      (Z == 1 AND (Cout_prv == 1 OR Cout == 0)))
   {
      If (Enable Pin Action == 1) Selected Pin = Pin Action;
      If (Interrupt Enable == 1) HETFLG[n] = 1;      /* n depends on address */
      If ([C28:C27] == 01) Generate request on request line [P25:P23];
      If ([C28:C27] == 11) Generate quiet request on request line [P25:P23];

      Immediate Data Field = Register A;
      C13 = 1; /*** switch to Time Mode ***/

   }
   Jump to Next Program Address;
}
Else If (Time Mode (C13 == 1)) AND ((Restart En. == 1 AND X == 0) OR Restart En. == 0)
{
   /* Result of subtract must not exceed 2^24 - 1 */
   Register T = Register A - Immediate Data Field;
   Jump to Conditional Program Address;
}
Cout_prv = Cout; (always executed)
```

The specific interrupt flag that is triggered depends on the address from which the instruction is executed, see Section 23.2.7.

23.6.3.21 SCNT (Step Count)

Syntax

SCNT {
[brk={OFF | ON}]
[next={label | 9-bit unsigned integer}]
step={8 | 16 | 32 | 64}
[control={OFF | ON}]
gapstart={25-bit unsigned integer}
[data={25-bit unsigned integer]
}

Figure 23-180. SCNT Program Field (P31:P0)

31 26	25 23	22	21 13	12 9	8	7 6	5 4	3 1	0
0	Reserved	BRK	Next program address	1010	Res.	00	Step width	Res.	1
6	3	1	9	4	1	2	2	3	1

Figure 23-181. SCNT Control Field (C31:C0)

31 27	26	25	24 0
Reserved	Control	Res.	Gap start
5	1	1	25

Figure 23-182. SCNT Data Field (D31:D0)

31 7	6 0
Data	Reserved
25	7

Cycles One or two cycles (two cycles when DF is involved in the calculations)

Register modified Register A

Description This instruction can be used only once in a program and defines a specialized virtual timer used after APCNT and before ACNT to generate an angle-referenced time base synchronized to an external signal (that is, a toothed wheel signal) as defined in APCNT and ACNT. Step width selection bits are saved in two flags, SWF0, and SWF1, to be re-used in ACNT.

SCNT multiplies the frequency of the external signal by a constant *K* defined in the step width field, [P5:P4]. The bit encoding for this field is defined in Table 23-93.

step Specifies the step increment to be added to the counter value each program resolution. These two bits provide the values for the SWF0 and SWF1 flags. The valid values are listed in Table 23-93.

Table 23-93. Step Width Encoding for SCNT

P5	P4	Step Width (K)
0	0	8
0	1	16
1	0	32
1	1	64

gapstart Defines the gap start angle, which SCNT writes to register A. The gap start value has no effect on the SCNT instruction, but if the ACNT instruction is being used, register A must contain the correct gap start value. For a typical toothed wheel gear:

GAPSTART = (*stepwidth* × (*actual teeth on gear* - 1)) + 1.

data Specifies the 25-bit integer value serving as a counter.

Default: 0.

This instruction is incremented by the step value K on each timer resolution up to the previous period value P(n-1) measured by APCNT (stored in register T). The resulting period of SCNT is: P(n - 1)/K

Due to stepping, the final count of SCNT will not usually exactly match the target p(n-1). SCNT compensates for this error by starting each cycle with the remainder of the previous cycle.

When SCNT reaches the target p(n-1), the zero flag is set as an increment condition for ACNT.SCNT also specifies a gap start angle, defining the start of a range in ACNT where period measurements in APCNT are temporarily stopped to mask singularities in the external signal.

SCNT uses register A to store the gap start value. Gap start has no effect for SCNT.

Execution

```
SWF1 = P5;
SWF0 = P4;
Z = 0;

If (register T != 0000000h)
{
   If (DCF == 1 OR ACF == 1)
   {
      Data Field register = 0000000h;
      Counter value = 0000000h;
   }

   If (DCF == 0 AND ACF == 0)
   {
      Data Field register = Data field register + Step Width;
   }

   If ((Data Field register - register T) >= 0)
   {
      Data field register = Data Field register - register T;
      Z = 1;
   }

   Register A = Gap start value;

}

Jump to Next Program Address;
```

23.6.3.22 SHFT (Shift)

Syntax

SHFT {
[brk={OFF | ON}]
[next={label | 9-bit unsigned integer}]
[reqnum={3-bit unsigned integer}
[request={NOREQ | GENREQ | QUIET}]
smode={OR0 | OL0 | OR1 | OL1 | ORZ | OLZ | IRM | ILL | IRZ | ILZ}
[control={OFF | ON}]
[prv={OFF | ON}]
[cond_addr={label | 9-bit unsigned integer}
cond={UNC | FALL | RISE}
pin ={pin number}
[reg={A | B | R | S | T | NONE}]
[irq={OFF | ON}]
[data={25-bit unsigned integer]
}

Figure 23-183. SHFT Program Field (P31:P0)

31 26	25 23	22	21 13	12 9	8 4	3 0
0	Request Number	BRK	Next program address	1111	Reserved	Smode
6	3	1	9	4	5	4

Figure 23-184. SHFT Control Field (C31:C0)

31 29	28 27	26	25	24 22	21 16
Reserved	Request type	Control	Prv.	Reserved	Conditional address
3	2	1	1	3	9

15 13	12 8	7	6 5	4	3	2 1	0
Conditional address	Pin select	Ext Reg	Shift condition	Res.	0	Register select	Int. ena
9	5	1	2	1	1	2	1

Figure 23-185. SHFT Data Field (D31:D0)

31 7	6 0
Data	Reserved
25	7

Cycles One

Register modified Selected register (A, B, R, S or T)

Description This instruction shifts the data field of the Instruction. N2HET pins can be used for data in or data out. SHFT includes parameters to select the shift direction (in, out, left, right), shift condition (shift on a defined clock edge on HET[0] or shift always), register for data storage (A, B, R, S or T), and the data pin.

smode Shift mode

Table 23-94. SHIFT MODE Encoding Format

smode	P3	P2	P1	P0	Operation	
OR0	0	0	0	0	Shift Out / Right	LSB 1st on HETx / 0 into MSB
OL0	0	0	0	1	Shift Out / Left	MSB 1st on HETx / 0 into LSB
OR1	0	0	1	0	Shift Out / Right	LSB 1st on HETx / 1 into MSB
OL1	0	0	1	1	Shift Out / Left	MSB 1st on HETx / 1 into LSB
ORZ	0	1	0	0	Shift Out / Right	LSB 1st on HETx / Z into MSB
OLZ	0	1	0	1	Shift Out / Left	MSB 1st on HETx / Z into LSB
IRM	1	0	0	0	Shift In / Right	HETx into MSB
ILL	1	0	0	1	Shift In / Left	HETx into LSB
IRZ	1	0	1	0	Shift In / Right	HETx in MSB / LSB into Z
ILZ	1	0	1	1	Shift In / Left	HETx in LSB / MSB into Z

cond Specifies the shift condition.

Table 23-95. SHIFT Condition Encoding

C6	C5	Shift Condition
0	X	Always
1	0	Rising edge of HET[0]
1	1	Falling edge of HET[0]

irq ON generates an interrupt if the Z flag is set. A value of OFF does not generate an interrupt.

Default: OFF.

data Specifies the 25-bit value for the data field.

Execution

```
If (SHIFT condition == 0X)
OR (SHIFT condition == 10 AND HET[0] rising edge)
OR (SHIFT condition == 11 AND HET[0] falling edge)
{
   If ([P3:P2] == 00)
   {
       If ((Immediate Data Field == all 0’s AND [P3:P0] == 000X)
           OR (Immediate Data Field == all 1’s AND [P3:P0] == 001X))
       {
          Z = 1;
       }
       else
       {
          Z = 0;
       }
   }
   else If ([P3:P0] == 1010)
   {
      Z = LSB of the Immediate Data Field;
   }
   else if ([P3:P0] == 1011)
   {
      Z = MSB of the Immediate Data Field;
   }
}

If( (Immediate Data Field == all 0’s) OR
    (Immediate Data Field == all 1’s))
{
   if (Interrupt Enable == 1)
                                          {
      HETFLG[n] = 1;      /* n depends on address */
   }
   Jump to Conditional Address;
}
else
{
   Jump to Next Program Address;
}

Prv. bit = HET[0] Pin level; (Always executed)

Shift Immediate Data Field once according to P[3:0];

Immediate Data Field = Result of the shift;

Selected register = Result of the shift;

Jump to Next Program Address;
```

NOTE: The immediate data field evaluates all 0s or all 1s and is performed before the shift operation.

The specific interrupt flag that is triggered depends on the address from which the instruction is executed, see Section 23.2.7.

23.6.3.23 WCAP (Software Capture Word)

Syntax

WCAP {
[brk={OFF | ON}]
[next={label | 9-bit unsigned integer}]
[reqnum={3-bit unsigned integer}
[request={NOREQ | GENREQ | QUIET}]
[hr_lr={HIGH | LOW}]
[control={OFF | ON}]
[prv={OFF | ON}]
[cond_addr={label | 9-bit unsigned integer}]
pin ={pin number}
event={NOCOND | FALL | RISE | BOTH}
reg={A | B | R | S | T | NONE}
[irq={OFF | ON}]
[data={25-bit unsigned integer]
[hr_data={7-bit unsigned integer}]
}

Figure 23-186. WCAP Program Field (P31:P0)

31 26	25 23	22	21 13	12 9	8	7 0
0	Request Number	BRK	Next program address	1011	hr_lr	Reserved
6	3	1	9	4	1	8

Figure 23-187. WCAP Control Field (C31:C0)

31 29	28 27	26	25	24 22	21 16
Reserved	Request type	Control	Prv.	Reserved	Conditional address
3	2	1	1	3	9

15 13	12 8	7	6 5	4 3	2 1	0
Conditional address	Pin select	Ext Reg	Capture condition	Reserved	Register select	Int. ena
9	5	1	2	2	2	1

Figure 23-188. WCAP Data Field (D31:D0)

31 7	6 0
Data	HR Data
25	7

Cycles One

Register modified None

Description This instruction captures the selected register into the data field if the specified capture condition is true on the selected pin. This instruction can be used with all pins.

If the hr_lr bit is reset, the WCAP instruction will capture an HR time stamp into the data field on the selected edge condition. If the hr_lr bit is set, the HR capture is ignored.

event Specifies the event that triggers the capture.

Table 23-96. Event Encoding Format for WCAP

C6	C5	Capture Condition
0	0	Always
0	1	Capture on falling edge
1	0	Capture on rising edge
1	1	Capture on rising and falling edge

irq ON generates an interrupt when the capture condition is met. No interrupt is generated for OFF.

Default: OFF.

data Specifies the 25-bit integer value to be written to the data field or selected register.

hr_data HR capture value.

Default: 0.

NOTE: WCAP in HR Mode: The HR Counter starts on a WCAP instruction execution (in the first loop clock) and will synchronize to the next loop clock. When N2HET is turned on and a capture edge occurs in the first loop clock (where the HR counter hasn't been synchronized to the loop clock), then the captured HR counter value is wrong and is of no use. So the captured HR data in the first loop clock should be ignored.

Execution

```
If (Specified Capture Condition is true on Selected Pin
OR Unconditional capture is selected)
{
   Immediate Data Field = Selected register value;

   If (hr_lr bit == 0) Capture the HR value in Immediate HR Data Field;

   If (Interrupt Enable == 1) HETFLG[n] = 1;      /* n depends on address */
   If ([C28:C27] == 01) Generate request on request line [P25:P23];
   If ([C28:C27] == 11) Generate quiet request on request line [P25:P23];

   Jump to Conditional Address;
}

Jump to Next Program Address;

Prv bit = Current Logic (Lx) value of selected pin; (always executed)
```

The specific interrupt flag that is triggered depends on the address from which the instruction is executed, see Section 23.2.7.

23.6.3.24 WCAPE (Software Capture Word and Event Count)

Syntax

WCAPE {
[brk={OFF | ON}]
[next={label | 9-bit unsigned integer}]
[reqnum={3-bit unsigned integer}
[request={NOREQ | GENREQ | QUIET}]
[control={OFF | ON}]
[prv={OFF | ON}]
[cond_addr={label | 9-bit unsigned integer}
pin ={pin number}
event={NOCOND | FALL | RISE | BOTH}
[reg={A | B | R | S | T | NONE}]
[irq={OFF | ON}]
[ts_data={25-bit unsigned integer]
[ec_data={7-bit unsigned integer}]
}

Figure 23-189. WCAPE Program Field (P31:P0)

31 26	25 23	22	21 13	12 9	8 0
0	Request Number	BRK	Next program address	1000	Reserved
6	3	1	9	4	9

Figure 23-190. WCAPE Control Field (C31:C0)

31 29	28 27	26	25	24 23 22	21 16
Reserved	Request type	Control	Prv.	Reserved	Conditional address
3	2	1	1	3	9

15 13	12 8	7	6 5	4 3	2 1	0
Conditional address	Pin select	Ext Reg	Capture condition	Reserved	Register select	Int. ena
9	5	1	2	2	2	1

Figure 23-191. WCAPE Data Field (D31:D0)

31 7	6 0
Time Stamp	Edge Counter
25	7

Cycles One

Register modified None

Description This instruction captures the selected register into the data field [D31:D7] and increments an event counter [D6:D0] if the specified capture condition is true on the selected pin. This instruction can be used with all pins, but the time stamp [D31:D7] has loop resolution only.

event Specifies the event that triggers the capture.

Table 23-97. Event Encoding Format for WCAPE

C6	C5	Capture Condition
0	0	Always
0	1	Capture on falling edge
1	0	Capture on rising edge
1	1	Capture on rising and falling edge

irq ON generates an interrupt when the capture condition is met. No interrupt is generated for OFF.

Default: OFF.

ts_data Specifies the 25-bit integer value for [D31:D7]

Default: 0.

ec_data Specifies the initial 7-bit integer value for [D6:D0].

Default: 0.

Execution

```
If (Specified Capture Condition is true on Selected Pin
OR Unconditional capture is selected)
{
   Immediate Data Field[31:7] = Selected register value;
   Immediate Data Field [6:0] = Immediate Data Field [6:0] + 1;

   If (Interrupt Enable == 1) HETFLG[n] = 1;      /* n depends on address */
   If ([C28:C27] == 01) Generate request on request line [P25:P23];
   If ([C28:C27] == 11) Generate quiet request on request line [P25:P23];

   Jump to Conditional Address;
}

Jump to Next Program Address;

Prv bit = Current Logic (Lx) value of selected pin; (always executed)
```

The specific interrupt flag that is triggered depends on the address from which the instruction is executed, see Section 23.2.7.

High-End Timer Transfer Unit (HTU) Module

This chapter describes the high-end timer transfer unit (HTU) module. The HTU is similar to the DMA (Direct Memory Access) module, but it is specialized to transfer N2HET (High-End Timer) data to or from the microcontroller RAM.

NOTE: This chapter describes a superset implementation of the HTU module that includes features and functionality that require DMA. Since not all devices have DMA capability, consult your device-specific datasheet to determine applicability of these features and functions to your device being used.

Topic **Page**

24.1 Overview

The HET transfer unit is a dedicated direct memory access controller that transfers data between the N2HET RAM and RAM buffers located in the main memory address range. This eliminates time consuming CPU accesses to the N2HET RAM to gather measurement data or creating output waveforms and thus freeing up the CPU to perform other tasks.

24.1.1 Features

- Independently transfers data between the N2HET and the main memory
- 8 double control packets supporting dual buffer configuration
- Transfer requests generated by N2HET instructions/events
- One shot, circular and auto switch buffer transfer modes for each double control packet for flexible buffer handling
- Constant and post-increment addressing modes
- 32- or 64-bit transactions
- Programmable memory protection region
- Parity protect control packet RAM
- Extensive diagnostic functionality

24.2 Module Operation

The HTU is tightly coupled to the N2HET and is not intended to transfer data from other peripheral modules. It initiates transfers with the help of requests generated by the N2HET program and configurable control packets. Figure 24-1 shows a system block diagram of the HTU and the main path for the data transfer. The tight coupling and the dedicated bus into the SCR (Switched Central Resource) reduces the amount of data transferred on the peripheral bus, which increases the overall system performance. However if the application decides to use the direct CPU access method to the N2HET RAM, it is free to do so.

Figure 24-2 shows a more detailed block diagram of the HTU module.

Figure 24-1. System Block Diagram

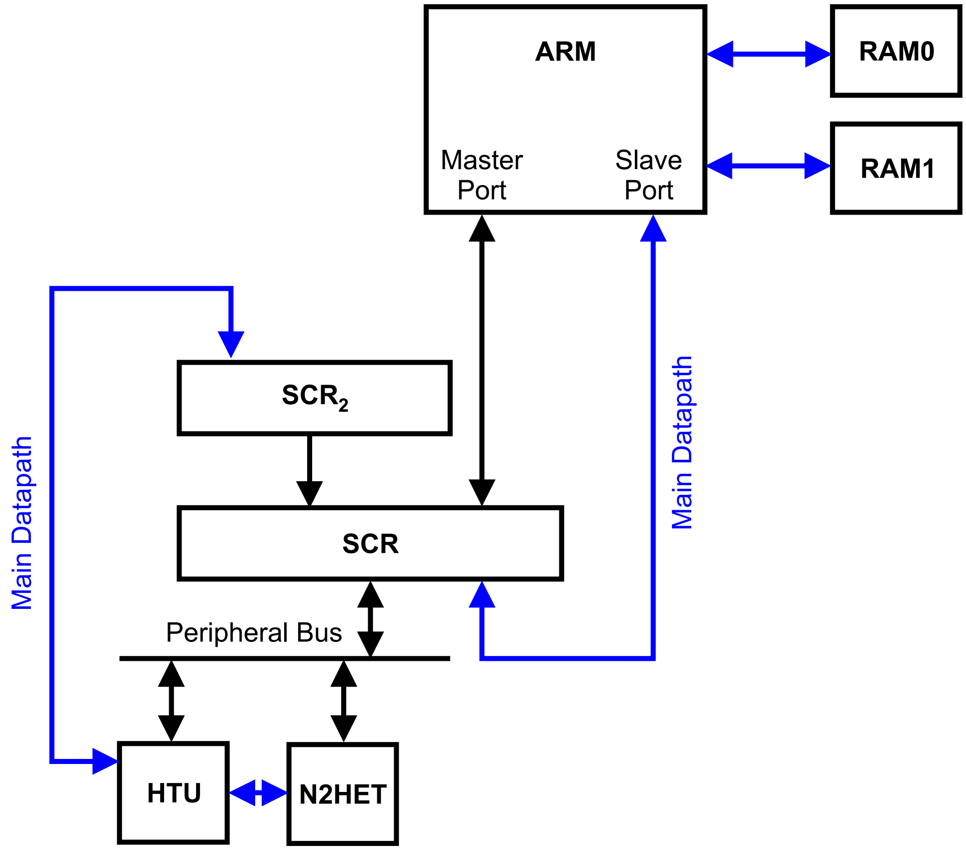
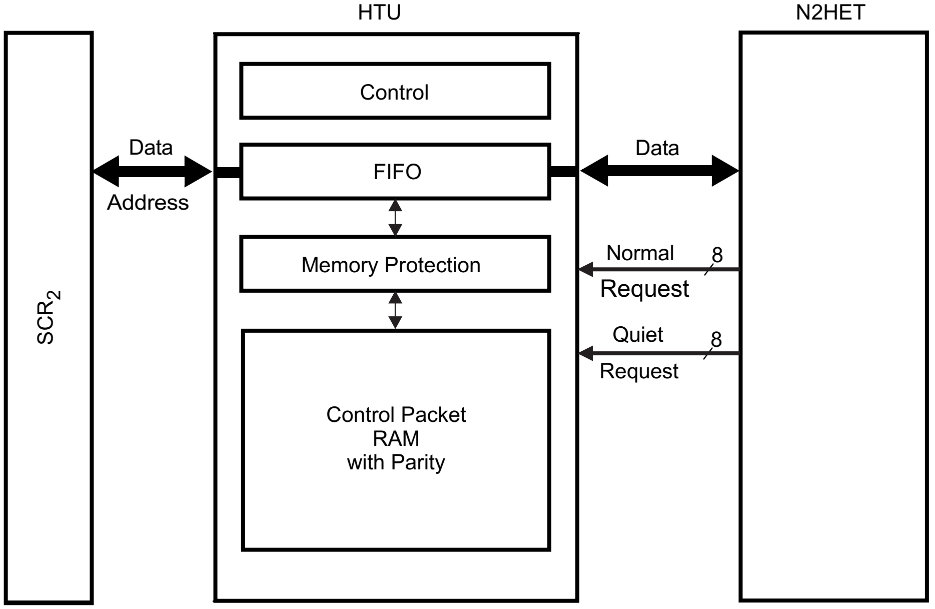


Figure 24-2. HTU Block Diagram



Transfers between N2HET RAM and the main memory are triggered by 8 different normal N2HET requests. Quiet requests are used for specific cases and are discussed in Section 24.2.4.1. Control packets, which store the source and destination addresses, the transfer count and other information (see Section 24.5), are associated with the requests. A FIFO decouples the read- and write-path and allows to do data-packing in the case of different read- and write-data sizes. The application can specify a section of memory into or from which the data is transferred. This serves as memory protection in the case that information in the control packet RAM was unintentionally altered and avoids that the HTU can overwrite important application data.

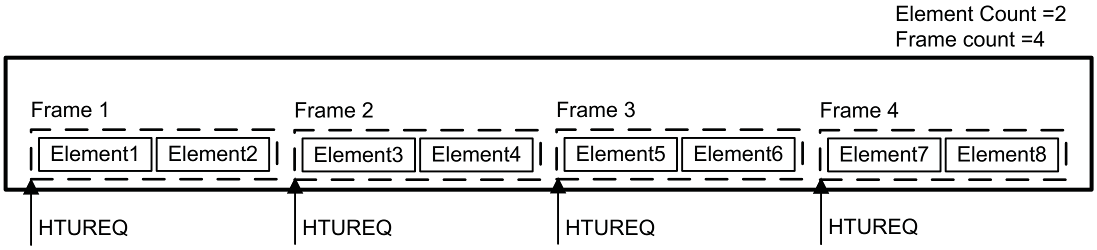
Control packets are implemented as double control packets (DCP) which allow to specify two buffers for the data transfer. This enables the CPU to work with one buffer, while new data is transferred to/from the other buffer.

The control packet defines:

- the start address of the source/destination buffers
- the N2HET instruction address location
- how many elements need to be transferred per request
- the buffer size as the number of elements times the number of frames
- the buffer handling

A transfer is triggered when a certain condition (for example, capture, compare condition) is detected by a N2HET instruction. The N2HET instruction specifies which request line to the HTU will be triggered at the event. The DCPs have a fixed assignment to the request lines and the corresponding assignment can be found in the device datasheet. Once a request is triggered, it starts a frame transfer. A frame can contain one or more elements. Elements are defined as 32-bit or 64-bit words of data.

Figure 24-3. Example of a HTU Transfer



24.2.1 Data Transfers between Main RAM and N2HET RAM

24.2.1.1 Addressing Modes

The addressing modes of a control packet need to be distinguished between the main RAM of the CPU and the N2HET RAM.

Main RAM

For each double control packet (see Section 24.2.1.3), the addressing mode for the main RAM (RAM0/1) can be configured to constant or post-increment mode in register IHADDRCT.

- **Constant Addressing:** In constant mode, the HTU writes/reads the data to/from the same address in the main RAM.
- **Post-increment Addressing:** In post-increment mode, the HTU writes/reads the data to/from the main RAM by incrementing through the addresses after each transfer. If 32-bit transfers are selected it will automatically increment by 4 Byte, if 64-bit transfers are selected, it will increment by 8 Byte. The examples of Use Cases illustrate the post-increment mode, where the elements of consecutive frames are transferred to/from consecutive locations in the main RAM buffer.

N2HET RAM

How a DCP addresses the N2HET RAM is determined by the initial N2HET address, the initial element counter (IETCOUNT) and the N2HET addressing mode (ADDMH). The main difference to the main RAM addressing mode is that the HET address is reset to the initial HET address for every first element of a frame. To implement constant addressing, the initial element counter needs to be set to 1. Post-increment addressing is selected by programming the initial element counter to a value other than 1.

24.2.1.2 Single Buffer Implementation

In a single buffer implementation, the DCP is set up to transfer data to/from a single buffer in the main RAM. With each transfer request, the programmed number of elements is transferred and the buffer pointer is reset to its starting address after the programmed number of frame transfers have completed.

Figure 24-4 shows the request on one request line of the HTU and the frame running on the assigned control packet visualized by the element counter. In the diagram, the frame has 5 element transfers (element count = 5).

Before the application reads the buffer, it has to disable the control packet to avoid that new data overwrites the buffer while it's being accessed by the application. Regardless of the control packet being disabled at t1 or t2 the last frame will always be completed, since the trigger request has been received already. The application can determine any ongoing transfers by the TIPF flag and the NACP bits.

- **One Shot Buffer Mode:** If TMBA or TMBB is set to one shot buffer mode then the data stream will stop after all elements of buffer A or buffer B have been transferred. This means that the corresponding DCP will be disabled after the last frame was transferred to/from buffer A or B and CFTCTA or CFTCTB decrements to 0.
- **Circular Buffer Mode:** If TMBA or TMBB is set to circular buffer mode, then the data stream will continue back at the start of buffer A or B after all elements of buffer A or B have been transferred. The example of Timing Example for Circular Buffer Mode assumes IETCOUNT = 3 (Initial Element Transfer Count), IFTCOUNT = 3 (Initial Frame Transfer Count, SIZE = 0 (Size of Transfer = 32-bit) and ADDFM = 0 (Addressing Mode Main Memory = Post Increment). So there are in total 9 32-bit values in the buffer. It also assumes IFADDRx = 10h. "U" means uninitialized.

Figure 24-4. Single Buffer Timing and Memory Representation

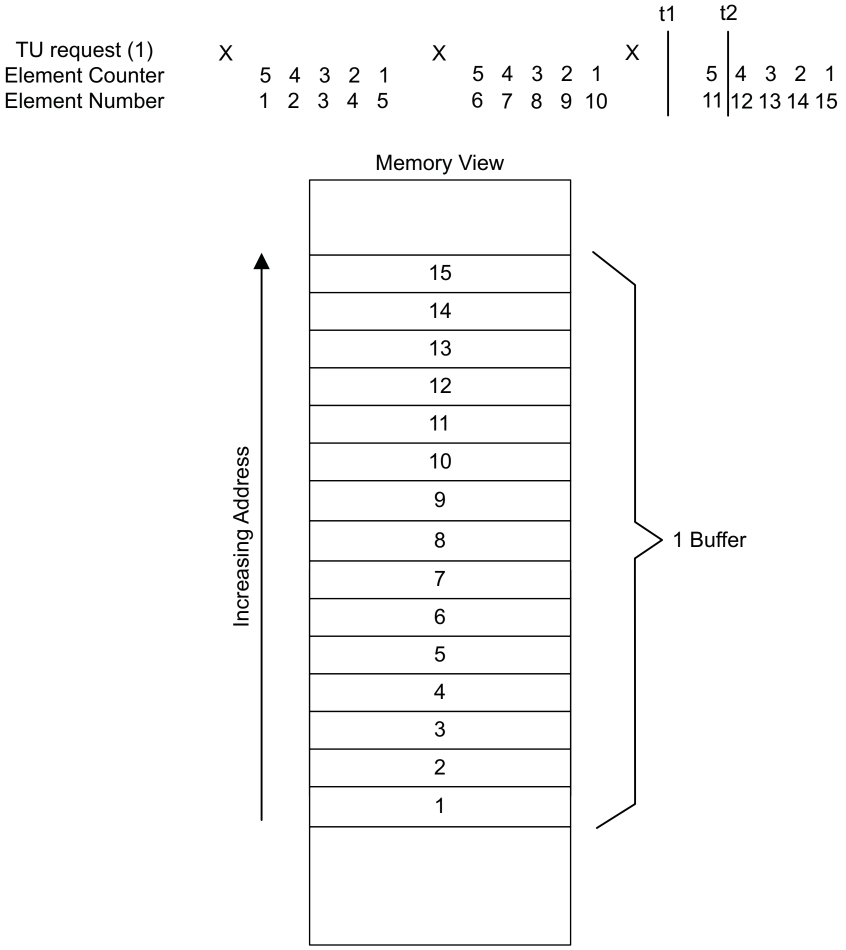
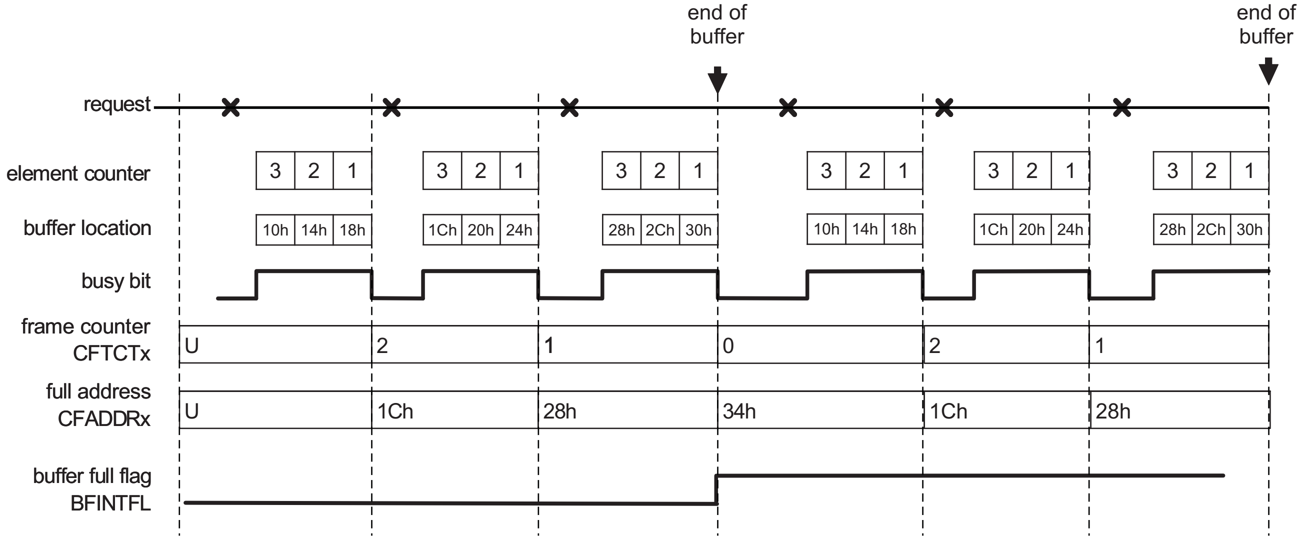


Figure 24-5. Timing Example for Circular Buffer Mode



24.2.1.3 Dual Buffer Implementation

The transfer unit provides **double control packets (DCPs)** supporting the use of two buffers per data stream (per HTU request source). If one buffer should be read by the CPU or DMA, the data stream is directed to the other buffer and the first buffer is frozen. Switching to the other buffer can be triggered with a write access to the CPENA register or with the DCP configured to automatically switch to the other buffer when the programmed number of frames has been transmitted. Freezing the buffer avoids this buffer to be overwritten with new HET data while the CPU or DMA reads this buffer.

Figure 24-6 shows a timing example of two HET instructions 1 and 2, which are the request sources for the HTU (and are controlled by DCP 1 and DCP 2). Each generated frame has 5 element transfers. Request source 1 has two RAM buffers, controlled by two control packets 1A and 1B. Request source 2 has two RAM buffers, controlled by two control packets 2A and 2B.

Figure 24-6. Dual Buffer Timing

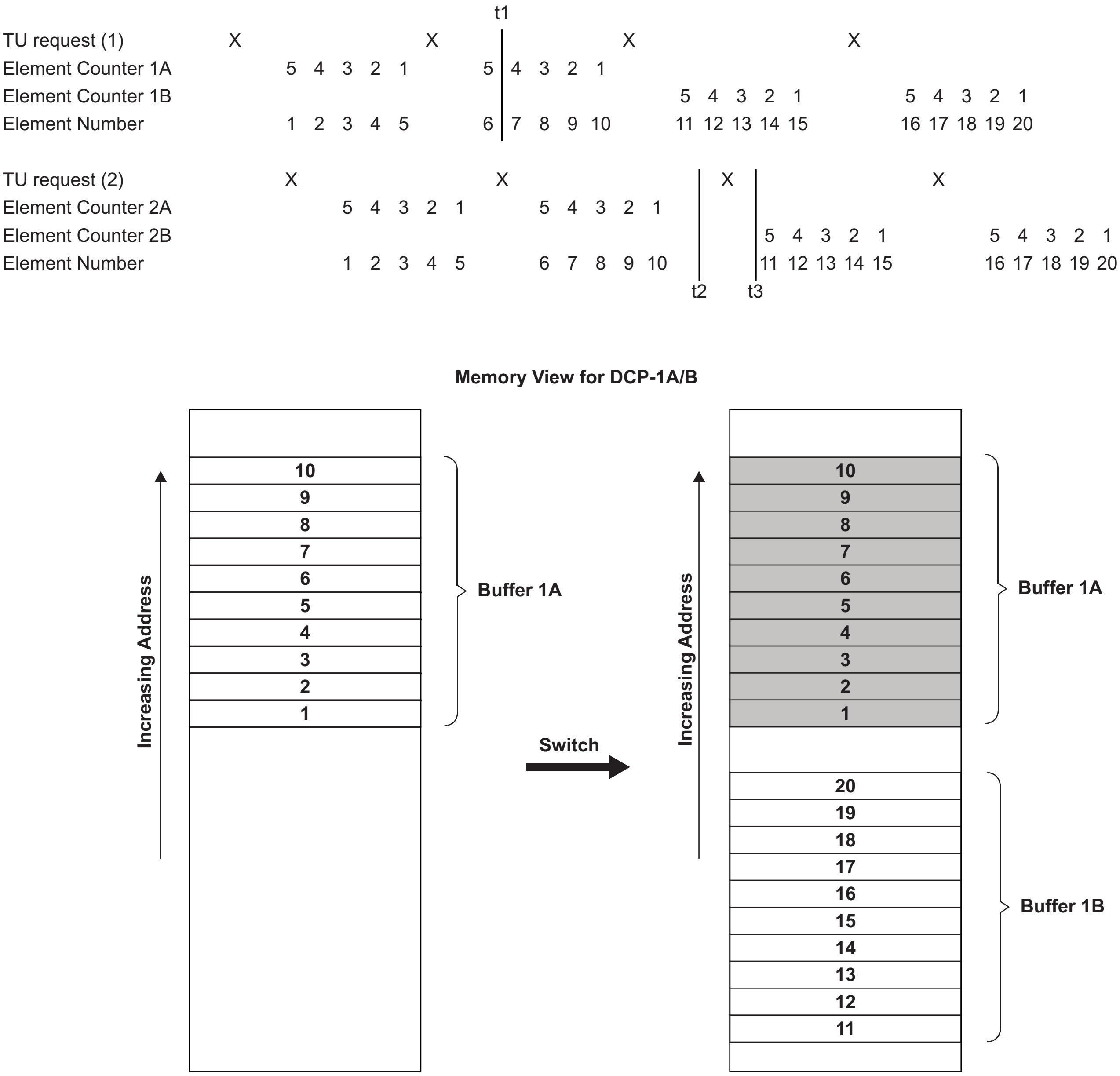

t1
TU request (1) X X X X
Element Counter 1A 5 4 3 2 1 5 4 3 2 1
Element Counter 1B 5 4 3 2 1 5 4 3 2 1
Element Number 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20
TU request (2) X X X X
Element Counter 2A 5 4 3 2 1 5 4 3 2 1
Element Counter 2B 5 4 3 2 1 5 4 3 2 1
Element Number 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20
t2
t3
Memory View for DCP-1A/B
Increasing Address
10
9
8
7
6
5
4
3
2
1
Buffer 1A
Switch
Increasing Address
10
9
8
7
6
5
4
3
2
1
Buffer 1A
20
19
18
17
16
15
14
13
12
11
Buffer 1B

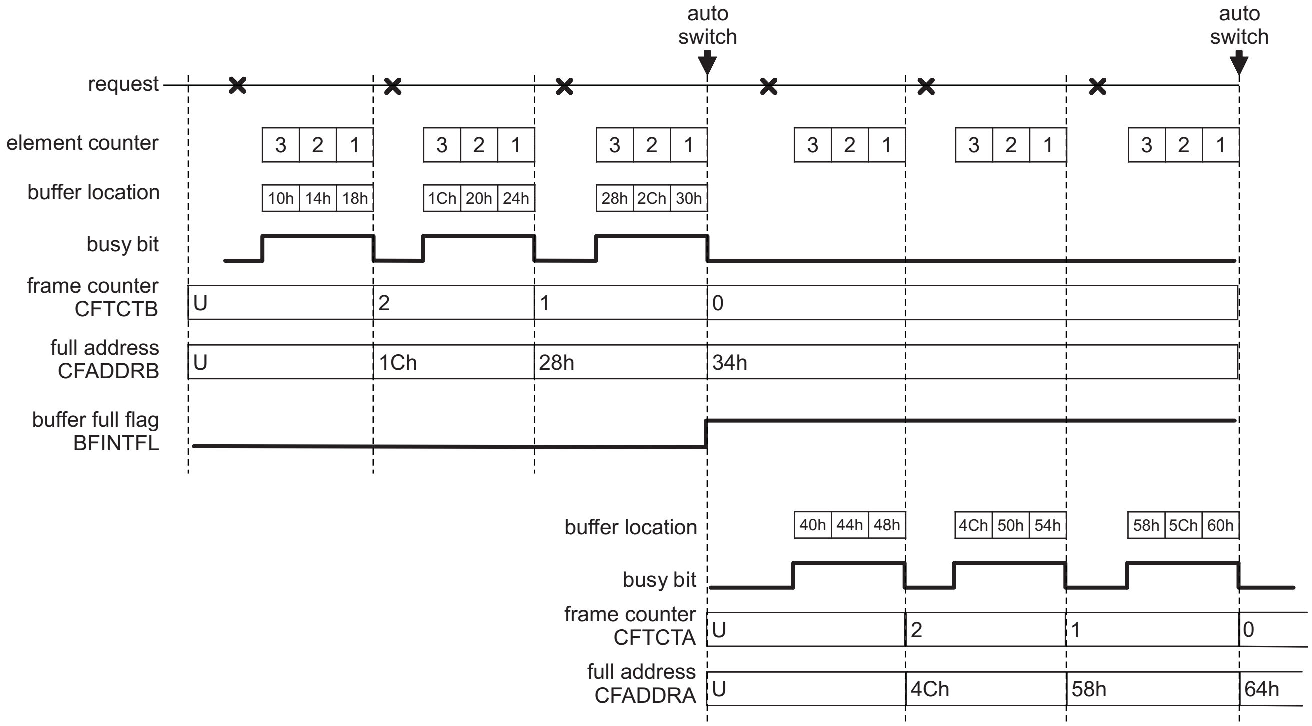
Figure 24-6 shows a switch at time t1, where buffer 1A is frozen and data stream 1 is directed to buffer 1B, but only after the frame has been completed. It also shows the time (t2 or t3) where 2A is frozen and data stream 2 is directed to buffer 2B. If the switch happens between the request and the start of the frame (for example, time t3), then the frame is processed by the new control packet (although the old control packet was active at the time of the request). The delays between the HTU requests and the start of the element transfers result from the fact that the HTU can process only one transfer at a time.

Auto Switch Buffer Mode

If TMBA is set to auto switch mode, then the data stream will continue at the start of buffer B after all elements of buffer A have been transferred. This means that in the CPENA register, CP A is disabled and CP B is enabled automatically and buffer B uses its initial main memory address and initial frame counter to start. The same principle is valid for TMBB and buffer B.

The examples of Figure 24-7 assumes IETCOUNT=3 (Initial Element Transfer Count), IFTCOUNT=3 (Initial Frame Transfer Count, SIZE=0 (Size of Transfer = 32-bit) and ADDFM=0 (Addressing Mode Main Memory = Post Increment). So there are in total 9 32-bit values in buffer A and B. It also assumes IFADDRB=10h and IFADDRA=40h. "U" means uninitialized.

Figure 24-7. Timing Example for Auto Switch Buffer Mode



24.2.1.4 General Control Packet Behavior

The action defined by the selected mode will be performed at the end of the last frame, which has the frame counter value of 1. The one shot and auto switch mode will automatically update the CPENA register at this time. Note, that for all three modes listed above, it is possible to switch to the other buffer by writing to CPENA before the end of the current buffer is reached.

If a write access to CPENA happens while the last frame of DCP x (with frame counter = 1) is transferred then the priority is defined by Table 24-1.

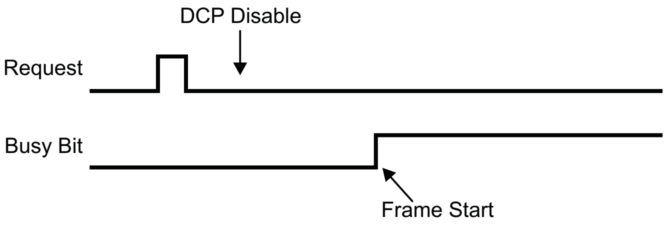
Table 24-1. CPENA / TMBx Priority Rules

Write access to CPENA bits (2 × x+1) and (2 × x) during the frame with frame counter = 1 [(1)]	Priority Rule
Disable: 01 --> 00 or 10 --> 00	Disabling the DCP by the write to CPENA has priority, TMBx is ignored.
Stay: 01 --> 01 or 10 --> 10	The write access to CPENA is ignored, TMBx has priority and defines the action.
Switch: 01 --> 10 or 10 --> 01	Switching the DCP by the write to CPENA has priority, TMBx is ignored.

[(1)] See read table of CPENA register (Table 24-14)

There could be a case where the CPU wants to do main memory operations, but does not want the HTU modifying the main memory. It could happen that a request was already active, but the frame transfer hasn't started yet when the application disabled the control packets. The timing diagram in Figure 24-8 shows this scenario.

Figure 24-8. Timing for Disabling Control Packets



Since the request for the transfer was already received before the DCPx is disabled, the HTU will still start the frame transfer. The application would poll the BUSYx bit during the time the DCPx was disabled and before the frame was started and would read a non-busy information. It then would start the main memory operations thinking all transfers have completed, however after some time the HTU will start the outstanding frame transfer and corrupt the main memory.

To avoid this, the application can set the VBUSHOLD bit to disable all transactions between the HTU and the main memory. It has to poll the BUSBUSY bit to ensure that no outstanding transactions on the bus are pending. The HTU will still receive all transfer requests from the N2HET, but it will not be able to transfer any data to or from the main memory, while the VBUSHOLD bit is set.

24.2.2 Arbitration of HTU Elements and Frames

- Frames do not interrupt each other. If a request occurs on DCP x while another frame runs on DCP y (and x ≠ y), then the current frame completes before the new frame starts.
- If two or more request lines are active, the request line with the lower number (specified in the request number field of the corresponding N2HET instruction) is serviced first.

24.2.3 Conditions for Frame Transfer Interruption

If a frame is currently transferred on DCP x and one of the events listed below happens, then the event will (1.) clear the element counter of DCP x, (2.) stop new element transfers on DCP x (3.) clear the active busy bit of DCP x and (4.) disable DCP x in the CPENA register. The DCPs other than DCP x will not be affected.

- Request Lost Error of DCP x (with CORL bit set to 0).
- Parity Error of DCP x (with parity check enabled and COPE bit set to 0). See also Section 24.2.6.
- Bus Error of DCP x.
- Memory Protection Error of DCP x (with memory protection enabled). See also Section 24.2.5.
- Writing a 1 to a BUSY bit (belonging to DCP x) if that bit is 1. There is no effect if the BUSY bit is 0.
- Writing a 1 to the HTURES bit.

When a memory protection error occurs, the access to the protected address is blocked. The frame is stopped before the element, which caused the violation transfer, starts. All other errors will let the current element transfer finish.

In case of the Request Lost and Bus Error, one more element transfer goes on the bus, before the frame is actually stopped. Accordingly, the busy bit is cleared after the element, which follows the element that caused the error.

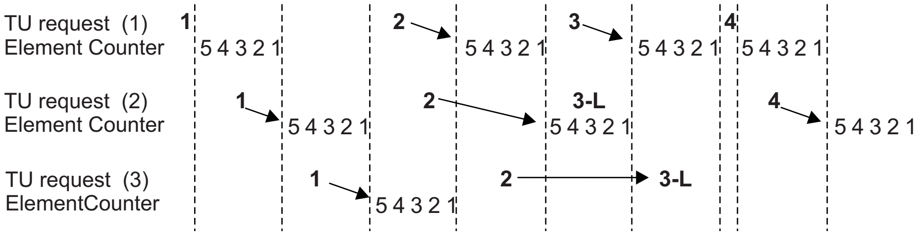
In case of the Bus Error, the counter for the element, which follows the element that caused the error, is captured to the ERRETC register field.

NOTE: If the HTUEN bit is cleared during a frame is transferred, then the frame will be completed before the HTU is disabled.

24.2.4 HTU Overload and Request Lost Detection

If the number of different HTU request sources is "high", the period between the requests is "short" and/or the initial element counter values are "big", then the HTU could get into a overload situation. In Figure 24-9, all requests marked with "L" are lost, since their frame is not completed at the time the next request occurs. Each number in the rows "TU request (x)" represents a time, where the associated N2HET instruction generates a request on DCP x. The arrows in Figure 24-9 point to the associated frame, which could be delayed compared to the request. The delays are caused by different frames, which are currently processed. The figure assumes that the CORL bit in the RLBECTRL register is set, which causes the DCP to stay enabled and let the data stream continue after a request lost error occurred on the DCP (see 3-L for TU request (2)).

Figure 24-9. Timing Example Including Lost Requests



Lost requests are signaled with the RLOSTFL register, and if enabled, can generate request lost interrupts.

If the CORL bit is set, a frame will be completed and the corresponding DCP stays enabled even if a request lost was generated during this frame.

In dual buffer mode, the request lost detection works continuously, independent of the CP switches.

24.2.4.1 Requests and Quiet Requests

In addition to generating too many transfer requests and thus overloading the HTU and not being able to transfer data at all, it can happen that inconsistent data is transferred. The following examples illustrate such scenarios.

In the examples below, the HTU reads a frame of three elements from the datafield of three different instructions. In Figure 24-10, the L3-Instruction generates the HTU request at time t2, t7, and so on. and the according frame (at t3). The frame is delayed because of the HTU load. However, as shown in Figure 24-10, the delay still allows the frame to complete before the datafield of instruction L1 is updated again. However, when the delay is longer (as shown in Figure 24-11), then the frame could fall into the N2HET loop (LRP), in which the N2HET updates the data fields of the L1, L2 and L3 instructions. In this case, the HTU could read inconsistent data as shown in the diagram. A wrong (new) value is read from L1 (at time t3), but correct ("old") values are read from L2 and L3 (at times t4 and t5).

Figure 24-10. Timing that Generates No Request Lost Error

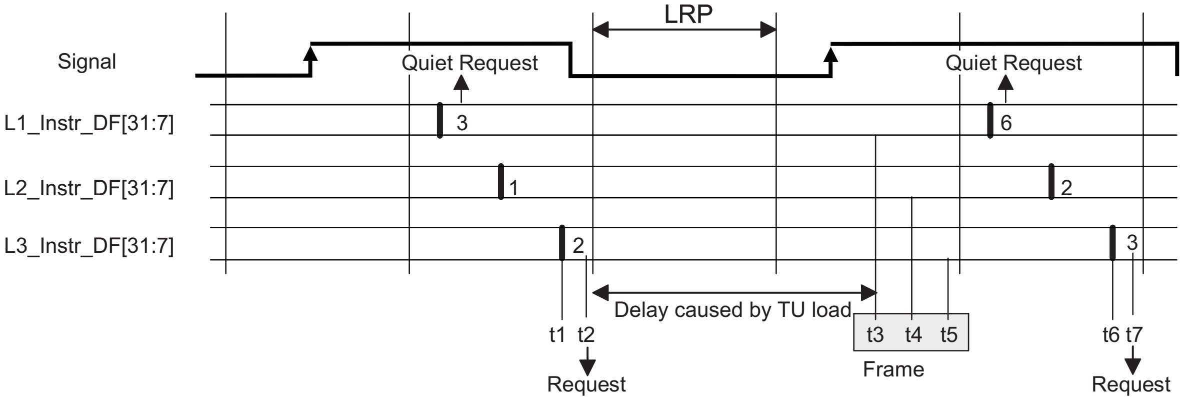
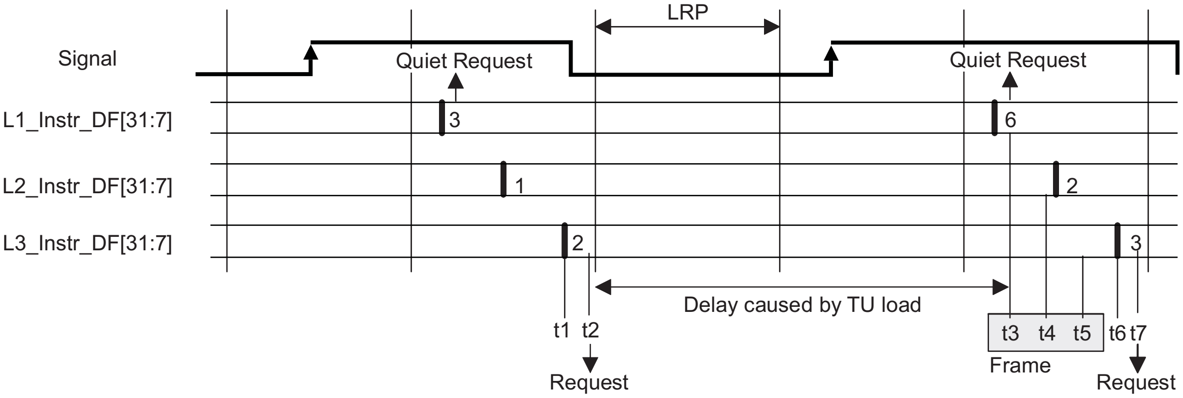


Figure 24-11. Timing that Generates a Request Lost Error



To prevent sending inconsistent data, the N2HET instructions are able to generate a **quiet request**, which does not originate a transfer but is only used by the HTU for consistency check. If a frame has not completed since the last request (or has not even started) at the time the quiet request occurs, then the HTU signals a request lost error. All instructions, which allow to generate a request can be configured to generate a quiet request instead. So in the examples of Figure 24-10 and Figure 24-11, instruction L1 should be configured to generate a quiet request and instruction L3 to generate a normal request. In the case of Figure 24-11, the corresponding bit in the RLOSTFL register will be set.

It is the responsibility of the N2HET software to enable a quiet request for the first instruction of an instruction block, which is addressed by DCP x, and to enable a normal request only for the last instruction of this block. Since enabling the quiet request should enable a proper request lost detection for DCP x, both N2HET instructions need to specify the same DCP x (reqnum=x).

The control fields of the HET instructions provide a 2-bit field to configure one of the following possibilities (as shown in Table 24-2). A 3-bit field in the program field will select which of the 8 Double Control Packets will be triggered by the request.

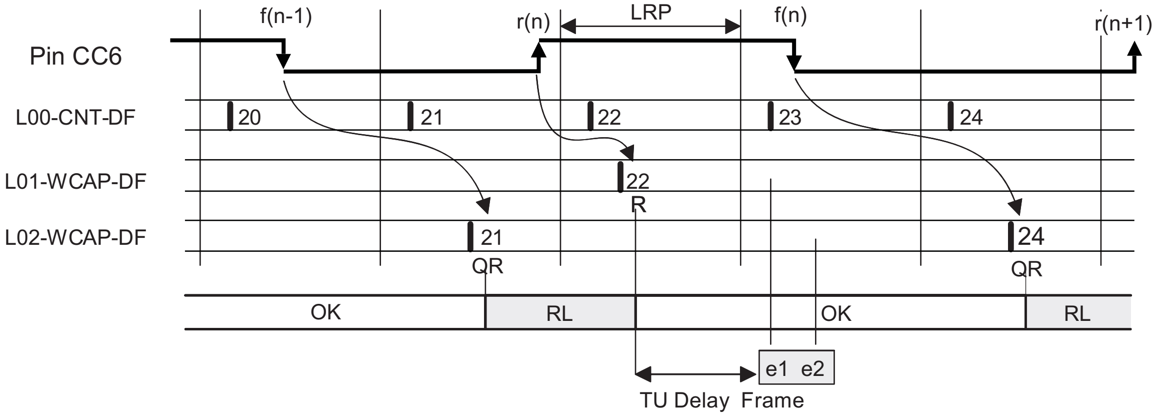
Table 24-2. Triggered Control Packets

Request Type Bit 1	Request Type Bit 0		Request Number
Don't care	0	No request	Specify number 0, 1,... or 7, which selects the HTU or DMA request line.
0	1	Generate normal request	
1	1	Generate quiet request	

In the case of very light HTU load, but higher signal requirements (for example, high frequency), the quiet request could also be used to define periods in which the data read by a control packet is safe. The following HET code will capture counter time stamps to the L1-WCAP data field after rising edges (at pin CC6) and to the L2-WCAP data field after falling edges (at pin CC6):

```
L0 CNT  {reg=A, max=0x1FFFFFF}
L1 WCAP {reqnum=3, request=GENREQ, event=RISE, reg=A, pin=CC6}
L2 WCAP {reqnum=3, request=QUIET, event=FALL, reg=A, pin=CC7}
; HET HRSHARE feature configured to assign both WCAPs to pin CC6
```

Figure 24-12. Timing Example for Two WCAP Instructions



The HTU frame will have two elements: The first gives the time stamp of the rising edge r(n) and the second gives the time stamp of the previous falling edge f(n-1). Using the code above, requests (R) and the quiet requests (QR) will occur at the times shown in Figure 24-12, and a request lost will only be signaled when the frame makes an access during the times marked with RL. So reading [22, 21] as frame elements is correct. If the signal frequency would increase, then a wrong pair [22, 23] could be read, but this will be signaled by a request lost error since at least e2 falls into the RL period.

24.2.5 Memory Protection

This feature allows restricting accesses to certain areas in memory in order to protect critical application data from unintentionally being manipulated by the HTU.

If the HTU memory protection feature is disabled, the full 4 GB address range can be accessed by the HTU without exception. There are two memory regions that start and end addresses can be configured.

With the HTU memory protection feature enabled, read and write accesses by the HTU IFADDRA and IFADDRB registers inside the defined regions are allowed. HTU access to its tightly-coupled memory is independent of the MPU, it routes through the dedicated HTU/N2HET bus using the IHADDR bits in the IHADDRCT register. See Section 24.2 for details on the tightly-coupled bus. For accesses outside the regions, one of two modes is configurable:

- Any access performed by the HTU is forbidden and will be signaled to the ESM module. Write accesses will be blocked.
- Read access is allowed but write access will be blocked and signaled to the ESM module.

To use one region only, REG01ENA must be 0. Bits ACCR01, INTENA01, and register settings of MP1S and MP1E will be ignored.

To use both regions, the following rules must be followed:

1. Memory mapped region 0 covers a lower memory area as Memory mapped region 1.
2. REG01ENA is a 1 and REG0ENA is a 0.
3. ACCR01 is set for the desired access type, ACCR0 is ignored.
4. INTENA01 is set for the desired action, INTENA0 is ignored.

If an element transfer of DCP x generates a memory protection error, then:

1. The element counter of DCP x is cleared.
2. All new element transfers on DCP x are stopped.
3. The active busy bit of DCP x is cleared.
4. DCP x is disabled in the CPENA register. The DCPs other than DCP x will not be affected.
5. The FT flag will be set.
6. An error is signaled to the ESM module.

24.2.6 Control Packet RAM Parity Checking

The HTU module can detect parity errors in the DCP (Double Control Packet) RAM. DCP RAM parity checking is implemented using one parity bit per byte. Even or odd parity checking can be selected in the DEVCR1 register of the system module and can be enabled/disabled by a 4-bit key in the PCR register.

During a read access to the DCP RAM, the parity is calculated based on the data read from the RAM and compared with the good parity value stored in the parity bits. The parity check is performed when the HTU or any other master (for example, CPU) makes a read access to the DCP RAM. A read access within the RAM section of an initial or current DCP checks all 16 bytes of the DCP at a time (see also DCP memory map). For example, if a byte read access happens for DCP RAM address 0, but there is a parity error at byte address Ch then the parity error will occur and the captured parity address will be Ch and not 0. The address of the byte in which the error occurred can be read from the PAR register. If successive DCP RAM read accesses generate multiple parity errors, only the address of the first detected error will be captured and the PAR register will not be updated by subsequent errors until it is read by the application. When multiple errors in a 16 byte word are detected, only the address of the lowest byte will be captured.

The application can decide whether to stop any transfers when a parity error is detected or to continue transferring data. If the COPE (Continue On Parity Error) bit is 0 and parity checking is enabled, then the HTU will not start the frame and the corresponding DCP will be automatically disabled in the CPENA register. If a master other than the HTU (for example, CPU) reads the RAM section of DCP x and a parity error is detected during this read access, while the parity check is enabled and the COPE bit is 0, then the DCP x will be automatically disabled in the CPENA register. If a frame for this DCP x is ongoing during this read access, then in addition the element counter of DCP x is cleared, all new element transfers on DCP x are stopped and the active busy bit of DCP x is cleared. With COPE set to 1 and the parity check enabled, the parity checking will still be performed, but the data transfer of an active DCP continues after a parity error was detected for this DCP. So neither the DCP with the parity error will be disabled nor the frame will be stopped.

After a DCP is enabled (with CPENA using BIM=0), then at the start of the first frame, the HTU performs the parity check only on the initial DCP, since it does not need the current DCP information. For further frames, the HTU performs the parity check for both initial and current DCP, since it needs both information.

On a parity error detection, an error will also be signaled to the ESM module.

24.2.6.1 Parity Bit Mapping and Testing

To test the parity checking mechanism, the parity RAM can be made accessible in order to allow manual fault insertion. Once the TEST bit is set, the parity bits are mapped to address FF4E 0200h.

When in test mode (the parity RAM is accessible), no parity checking will be done when reading from parity RAM, but parity checking will still be performed for read accesses to the DCP RAM.

Table 24-3 and Table 24-4 show how the corresponding parity bits of the DCP RAM bytes are mapped into the memory.

Table 24-3. DCP RAM

Bit	31	24	23	16	15	8	7	0
FF4E 0000h	Byte 0		Byte 1		Byte 2		Byte 3	
FF4E 0004h	Byte 4		Byte 5		Byte 6		Byte 7	
FF4E 0008h	Byte 8		Byte 9		Byte 10		Byte 11	
FF4E 000Ch	Byte 12		Byte 13		Byte 14		Byte 15	

Table 24-4. DCP Parity RAM

Bit		24		16		8		0
FF4E 0200h		P0		P1		P2		P3
FF4E 0204h		P4		P5		P6		P7
FF4E 0208h		P8		P9		P10		P11
FF4E 020Ch		P12		P13		P14		P15

Each byte in DCP RAM has its own parity bit in the DCP Parity RAM. P0 is the parity bit for byte 0, P1 is the parity bit for byte 1, and so on.

24.2.6.2 Initializing Parity Bits

After device power up, the DCP RAM content including the parity bit cannot be guaranteed. In order to avoid parity failures, when reading DCP RAM, the RAM has to be initialized first. This can simply be done by writing known values into the RAM by software and the corresponding parity bit will be automatically calculated.

Another possibility to initialize the DCP memory and its parity bits is to use the system module, which is an on-chip module external to the HTU. This module can start the automatic initialization of all RAMs on the microcontroller including the HTU DCP RAM. This function initializes the complete DCP RAM to 0 when activated by the system module. Depending on the even/odd parity selection, all parity bits will be calculated accordingly. The HTUEN bit must be cleared and the parity functionality must be enabled (by PARITY_ENA) during the automatic DCP RAM initialization. If HTUEN is 1 when the initialization is triggered by the system module, then the initialization will not be performed and the HTU operation is not affected. If a 1 is written to HTUEN during the initialization, then the HTUEN bit will be set but the HTU will not be enabled before the initialization completes.

24.3 Use Cases

24.3.1 Example: Single Element Transfer with One Trigger Request

This example considers the case that the HTU fills a RAM buffer in the main (CPU) data RAM. The HTU reads from the instruction which generates the HTU requests.

This example uses a PCNT instruction. Every time the PCNT has captured a new pulse or period value, it will automatically generate a transfer request to the HTU, which then transfers the value from the N2HET RAM to the buffer RAM. So over time consecutive locations in the RAM buffer can be filled with consecutive measurement values captured into the N2HET RAM data field of the same PCNT instruction without loading or interrupting the CPU.

24.3.2 Example: Multiple Element Transfer with One Trigger Request

The following example shows how the HTU could be used to fill a RAM buffer with a data stream including different types of measurement values belonging to the same N2HET input signal (on one pin): Time stamp values (WCAP), edge counter values (ECNT) and last period values (PCNT).

Figure 24-13 shows the timing and Table 24-5 shows the byte addresses of the program- (PF), control- (CF), data- (DF) and reserved field (res) of the WCAP-ECNT-PCNT instruction block. The timing and code example assumes that all three instructions are assigned to the same N2HET pin.

Figure 24-13. Timing of the WCAP, ECNT, PCNT Example

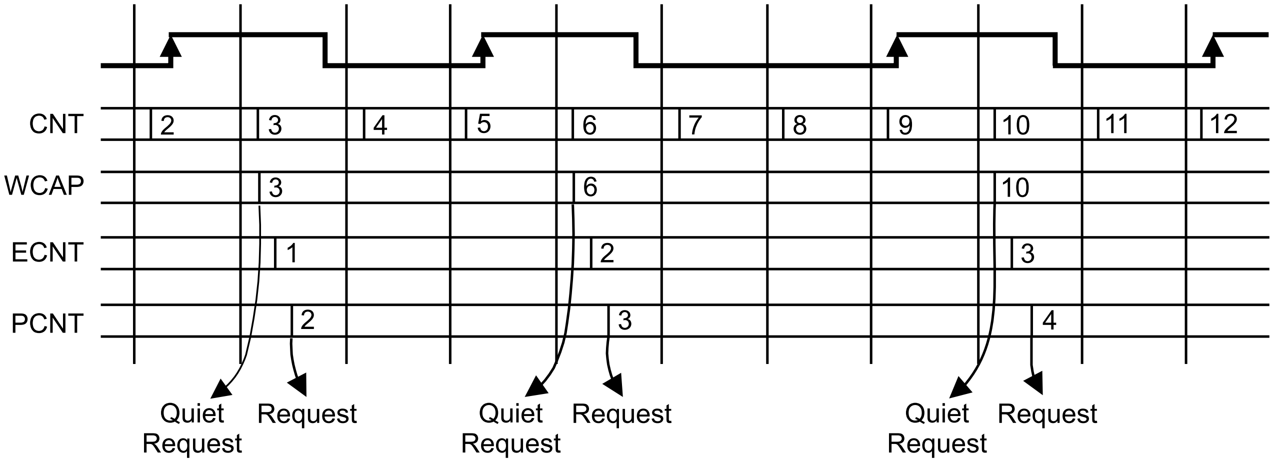


Table 24-5. Field Addresses of the WCAP, ECNT, PCNT Example

	PF	CF	DF	Res
WCAP	30h	34h	38h	3Ch
ECNT	40h	44h	48h	4Ch
PCNT	50h	54h	58h	5Ch

In the HET code the HTU request is enabled only for the last instruction (PCNT) of the WCAP-ECNT-PCNT block. When the PCNT condition is true, it will cause the generated HTU frame to perform three HTU element reads from the data fields of WCAP, ECNT, and PCNT.

32-Bit-Transfer of data fields:

Table 24-6 shows how the internal element counter, frame counter and the address registers change over time for the example described above. Every time the PCNT instruction captures a new value it generates a request to the HTU, which starts a frame. At the end of each frame the frame counter decrements.

Table 24-6. 32-Bit-Transfer of Data Fields[1]

Frame Counter	3			2			1		
Element Counter	3	2	1	3	2	1	3	2	1
Source Address (HET)	38h	48h	58h	38h	48h	58h	38h	48h	58h
Destination Address (main CPU RAM)	70h	74h	78h	7Ch	80h	84h	88h	8Ch	90h

[1] Shows the byte addresses

The destination buffer is filled with the WCAP, ECNT, and PCNT data field values as shown in Table 24-7.

Table 24-7. Destination Buffer Values

Address	Frame Count	Instruction	Value
70h	3	WCAP	3
74h	3	ECNT	1
78h	3	PCNT	2
7Ch	2	WCAP	6
80h	2	ECNT	2
84h	2	PCNT	3
88h	1	WCAP	10
8Ch	1	ECNT	3
90h	1	PCNT	4

The corresponding setup of the HTU control packet for this example is as follows:

```
IHADDR    = 0x38                 // points to WCAP data field
IFADDRA   = 0x70                 // points to buffer
ITCOUNT [frame count = 3] [element count = 3]
IHADDRCT = [DIR: Read HET and write to full address]
           [SIZE: 32 bit]
           [ADDMH: Increment HET address by 16 bytes]
           [ADDMF: Post increment full address mode]
           [Any transfer mode]
```

24.3.3 Example: 64-Bit-Transfer of Control Field and Data Fields

Table 24-8 shows how the internal element counter, frame counter and the address registers change over time assuming the same example as in Section 24.3.2, but now with a transfer size set to 64-bit. The HET address now points to the control field of the instruction, so CF and DF are transferred as 64 bit data.

Table 24-8. 64-Bit-Transfer of Control Field and Data Fields[1]

Frame Counter	3			2			1		
Element Counter	3	2	1	3	2	1	3	2	1
HET (Source) Address	34h	44h	54h	34h	44h	54h	34h	44h	54h
Full (Destination) Address	70h	78h	80h	88h	90h	98h	A0h	A8h	B0h

[1] Shows the byte addresses.

The destination buffer is filled with the WCAP, ECNT, and PCNT control and data field values as shown on the right in Table 24-9.

Table 24-9. Destination Buffer Values

Address	Frame Count	Instruction	Value
70h	3	WCAP	Control Field Value
74h	3	WCAP	3
78h	3	ECNT	Control Field Value
7Ch	3	ECNT	1
80h	3	PCNT	Control Field Value
84h	3	PCNT	2
88h	2	WCAP	Control Field Value
8Ch	2	WCAP	6
90h	2	ECNT	Control Field Value
94h	2	ECNT	2
98h	2	PCNT	Control Field Value
9Ch	2	PCNT	3
A0h	1	WCAP	Control Field Value
A4h	1	WCAP	10
A8h	1	ECNT	Control Field Value
ACh	1	ECNT	3
B0h	1	PCNT	Control Field Value
B4h	1	PCNT	4

The necessary setup of the HTU control packet (see Section 24.5) for this example is as follows:

```
IHADDR   = 0x34  (points to WCAP control field)
IFADDR   = 0x70  (points to buffer)
ITCOUNT [frame count = 3] [element count = 3]
IHADDRCT = [DIR: Read HET and write to full address]
           [SIZE: 64 bit]
           [ADDMH: Increment HET address by 16 bytes]
           [ADDMF: post increment full address mode]
           [Any transfer mode]
```

For different applications, which have the transfer direction set for reading the buffer and writing to HET fields, the 64-bit transfer could be used to change the conditional addresses together with a new data field.

24.4 HTU Control Registers

Table 24-10 provides a summary of the registers. The registers support 8-bit, 16-bit, and 32-bit writes. The offset is relative to the associated peripheral select. See the following sections for detailed descriptions of the registers. The base address for the control registers is FFF7 A400h for HTU1 and FFF7 A500h for HTU2. The address locations not listed, are reserved.

Table 24-10. HTU Control Registers

Offset	Acronym	Register Description	Section
00h	HTU GC	Global Control Register	Section 24.4.1
04h	HTU CPENA	Control Packet Enable Register	Section 24.4.2
08h	HTU BUSY0	Control Packet Busy Register 0	Section 24.4.3
0Ch	HTU BUSY1	Control Packet Busy Register 1	Section 24.4.4
10h	HTU BUSY2	Control Packet Busy Register 2	Section 24.4.5
14h	HTU BUSY3	Control Packet Busy Register 3	Section 24.4.6
18h	HTU ACPE	Active Control Packet and Error Register	Section 24.4.7
20h	HTU RLBECTRL	Request Lost and Bus Error Control Register	Section 24.4.8
24h	HTU BFINTS	Buffer Full Interrupt Enable Set Register	Section 24.4.9
28h	HTU BFINTC	Buffer Full Interrupt Enable Clear Register	Section 24.4.10
2Ch	HTU INTMAP	Interrupt Mapping Register	Section 24.4.11
34h	HTU INTOFF0	Interrupt Offset Register 0	Section 24.4.12
38h	HTU INTOFF1	Interrupt Offset Register 1	Section 24.4.13
3Ch	HTU BIM	Buffer Initialization Mode Register	Section 24.4.14
40h	HTU RLOSTFL	Request Lost Flag Register	Section 24.4.15
44h	HTU BFINTFL	Buffer Full Interrupt Flag Register	Section 24.4.16
48h	HTU BERINTFL	BER Interrupt Flag Register	Section 24.4.17
4Ch	HTU MP1S	Memory Protection 1 Start Address Register	Section 24.4.18
50h	HTU MP1E	Memory Protection 1 End Address Register	Section 24.4.19
54h	HTU DCTRL	Debug Control Register	Section 24.4.20
58h	HTU WPR	Watch Point Register	Section 24.4.21
5Ch	HTU WMR	Watch Mask Register	Section 24.4.22
60h	HTU ID	Module Identification Register	Section 24.4.23
64h	HTU PCR	Parity Control Register	Section 24.4.24
68h	HTU PAR	Parity Address Register	Section 24.4.25
70h	HTU MPCS	Memory Protection Control and Status Register	Section 24.4.26
74h	HTU MP0S	Memory Protection 0 Start Address Register	Section 24.4.27
78h	HTU MP0E	Memory Protection 0 End Address Register	Section 24.4.28

24.4.1 Global Control Register (HTU GC)

Figure 24-14. Global Control Register (HTU GC) [offset = 00]

31 25	24	23 17	16
Reserved	VBUSHOLD	Reserved	HTUEN
R-0	R/WP-0	R-0	R/WP-0
15 9	8	7 1	0
Reserved	DEBM	Reserved	HTURES
R-0	R/WP-0	R-0	R/WP-0

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privilege mode only; -*n* = value after reset

Table 24-11. Global Control Register (HTU GC) Field Descriptions

Bit	Field	Value	Description
31-25	Reserved	0	Reads return 0. Writes have no effect.
24	VBUSHOLD		Hold the VBUS bus
		0	The VBUS is not held.
		1	The VBUSHOLD bit holds the bus used to transfer data between the HTU and the N2HET module. When the BUS_BUSY bit is 0 then the bus is no longer busy. While the bus is held, requests will still be accepted. They will be acted upon when the VBUSHOLD is 0. Request lost conditions will be detected and interrupts generated if they are enabled.
23-17	Reserved	0	Reads return 0. Writes have no effect.
16	HTUEN		Transfer Unit Enable Bit
		0	The Transfer Unit is disabled.
		1	The Transfer Unit is enabled.
			The configuration registers and control packets should be set up first before the HTUEN bit is set to 1 to prevent it from carrying out unintended bus transactions. If the HTUEN bit is cleared to 0 during a frame is transferred, then the frame will be completed before the HTU is disabled. The HTUEN bit must be cleared to 0 and the parity functionality must be enabled (by PARITY_ENA) during the automatic DCP RAM initialization (see Initializing Parity Bits). If HTUEN is 1 when the initialization is triggered by the system module, then the initialization will not be performed and the HTU operation is not affected. If a 1 is written to HTUEN during the initialization, then the HTUEN bit will be set but the HTU will not be enabled before the initialization completes. **Note: If HTU is disabled during a frame transfer, then the ongoing current frame will be completed before the HTU module is disabled. If enabled again, then the transfer will restart from the initial frame count for the CP programmed.**
15-9	Reserved	0	Reads return 0. Writes have no effect.
8	DEBM		Debug Mode
		0	The Transfer Unit is stopped in debug mode. The HTU will complete the current frame, but not start any new frames. It will also ignore all requests from the HET and not generate any request lost signals.
		1	The Transfer Unit continues operation in debug mode. **Note:** Since the HET has also an "ignore suspend" bit, there a several possibilities for the behavior of the HET and HTU in suspend mode.
7-1	Reserved	0	Reads return 0. Writes have no effect.
0	HTURES		HTU Software Reset Request
		0	Reset request is not issued to the HTU module. Writing a 0 has no effect.
		1	Reset request is issued to the HTU module. Ongoing element transfers will be completed, before resetting the complete HTU module, similar to a hardware reset. The HTURES bit will also be cleared. The recommended order of operations is: • Set the software reset bit. This also clears HTUEN. • Wait for the HTURES bit to clear. • Configure the HTU registers and packets. • Set the HTUEN bit to begin operation.

24.4.2 Control Packet Enable Register (HTU CPENA)

This register enables or disables the individual double control packets (DCP).

Figure 24-15. Control Packet Enable Register (HTU CPENA) [offset = 04h]

31	16
Reserved	
R-0	

15	0
CPENA	
R/WP-0	

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privilege mode only; -*n* = value after reset

Table 24-12. Control Packet Enable Register (HTU CPENA) Field Descriptions

Bit	Field	Value	Description
31-16	Reserved	0	Reads return 0. Writes have no effect.
15-0	CPENA		CP Enable Bits Bits (2*x) and (2*x+1) of CPENA control the double control packet (DCP) x (whereby x must be within 0,1,....,7). See Table 24-13 for write rules. See Table 24-14 for read rules.

Table 24-13. CPENA Write Results

Bit (2*x+1)	Bit (2*x)	Control packets (CP) B and A of DCP x are affected as follows:
0	0	CP B and A are not changed.
0	1	CP B is disabled and CP A are enabled simultaneously.
1	0	CP B is enabled and CP A are disabled simultaneously.
1	1	CP B and CP A are both disabled simultaneously.

Table 24-14. CPENA Read Results

Bit (2*x+1)	Bit (2*x)	State of DCP:
0	0	The DCP is disabled.
0	1	CP B is disabled and CP A is enabled.
1	0	CP B is enabled and CP A is disabled.
1	1	Cannot be read.

- The conditions listed in Section 24.2.3 can automatically disable DCP x. In this case, bits (2*x) and (2*x+1) are both automatically set to 0.
- When bits (2*x) and (2*x+1) change from 00 to 01 or from 00 to 10 caused by a write access to CPENA, then old pending requests on the corresponding request line are cleared. This means only new requests which occur **after** this write access cause the first HTU transfer for this DCP. This is **not** the case when **switching** CPs (from 10 to 01 or from 01 to 10).
- CP A and/or CP B of a DCP can be configured to one-shot, circular or auto-switch transfer mode via the TMBA or TMBB bits in the IHADDRCT control packet configuration. If a write access to CPENA occurs during the last frame of a buffer (with frame counter = 1) then the action defined by the write access to CPENA and the action defined by TMBx can contradict. The priority rules for this case are given in Table 24-1.

24.4.3 Control Packet (CP) Busy Register 0 (HTU BUSY0)

This register displays the status of individual control packets.

Figure 24-16. Control Packet (CP) Busy Register 0 (HTU BUSY0) [offset = 08h]

31 25	24	23 17	16
Reserved	BUSY0A	Reserved	BUSY0B
R-0	R/W1CP-0	R-0	R/W1CP-0

15 9	8	7 1	0
Reserved	BUSY1A	Reserved	BUSY1B
R-0	R/W1CP-0	R-0	R/W1CP-0

LEGEND: R/W = Read/Write; R = Read only; W1CP = Write 1 in privilege mode only to clear the bit; -*n* = value after reset

Table 24-15. Control Packet (CP) Busy Register 0 (HTU BUSY0) Field Descriptions

Bit	Field	Value	Description
31-25	Reserved	0	Reads return 0. Writes have no effect.
24	BUSY0A		Busy Flag for CP A of DCP 0
23-17	Reserved	0	Reads return 0. Writes have no effect.
16	BUSY0B		Busy Flag for CP B of DCP 0
15-9	Reserved	0	Reads return 0. Writes have no effect.
8	BUSY1A		Busy Flag for CP A of DCP 1
7-1	Reserved	0	Reads return 0. Writes have no effect.
0	BUSY1B		Busy Flag for CP B of DCP 1

The bit is **set** when the frame on the according control packet **starts** (as shown in the diagram below, there could be a delay between the request and the start of the frame).

The bit is automatically **cleared** at any of the following conditions:

1. At the end of a frame.
2. Writing a 1 to a BUSY bit (of DCP x) if that bit is 1. This will:
   a. clear the element counter of DCP x
   b. stop all new element transfers on DCP x
   c. clear the busy bit
   d. and disable DCP x in the CPENA register.

   There is no effect, if the BUSY bit is 0.
3. At the conditions listed in Section 24.2.3.

A write access to the CPENA register can stop a control packet (CP) in single buffer mode or it can switch to the other CP of a DCP in dual buffer mode. If stopping or switching occurs while a frame runs on the currently active control packet, the CPU can poll the busy bit to determine when it is safe to read the buffer.

24.4.4 Control Packet (CP) Busy Register 1 (HTU BUSY1)

This register displays the status of individual control packets.

Figure 24-17. Control Packet (CP) Busy Register 1 (HTU BUSY1) [offset = 0Ch]

31 25	24	23 17	16
Reserved	BUSY2A	Reserved	BUSY2B
R-0	R/W1CP-0	R-0	R/W1CP-0
15 9	8	7 1	0
Reserved	BUSY3A	Reserved	BUSY3B
R-0	R/W1CP-0	R-0	R/W1CP-0

LEGEND: R/W = Read/Write; R = Read only; W1CP = Write 1 in privilege mode only to clear the bit; -*n* = value after reset

Table 24-16. Control Packet (CP) Busy Register 1 (HTU BUSY1) Field Descriptions

Bit	Field	Value	Description
31-25	Reserved	0	Reads return 0. Writes have no effect.
24	BUSY2A		Busy Flag for CP A of DCP 2
23-17	Reserved	0	Reads return 0. Writes have no effect.
16	BUSY2B		Busy Flag for CP B of DCP 2
15-9	Reserved	0	Reads return 0. Writes have no effect.
8	BUSY3A		Busy Flag for CP A of DCP 3
7-1	Reserved	0	Reads return 0. Writes have no effect.
0	BUSY3B		Busy Flag for CP B of DCP 3

See Section 24.4.3 for more details.

24.4.5 Control Packet (CP) Busy Register 2 (HTU BUSY2)

Figure 24-18. Control Packet (CP) Busy Register 2 (HTU BUSY2) [offset = 10h]

31 25	24	23 17	16
Reserved	BUSY4A	Reserved	BUSY4B
R-0	R/W1CP-0	R-0	R/W1CP-0
15 9	8	7 1	0
Reserved	BUSY5A	Reserved	BUSY5B
R-0	R/W1CP-0	R-0	R/W1CP-0

LEGEND: R/W = Read/Write; R = Read only; W1CP = Write 1 in privilege mode only to clear the bit; -*n* = value after reset

Table 24-17. Control Packet (CP) Busy Register 2 (HTU BUSY2) Field Descriptions

Bit	Field	Value	Description
31-25	Reserved	0	Reads return 0. Writes have no effect.
24	BUSY4A		Busy Flag for CP A of DCP 4
23-17	Reserved	0	Reads return 0. Writes have no effect.
16	BUSY4B		Busy Flag for CP B of DCP 4
15-9	Reserved	0	Reads return 0. Writes have no effect.
8	BUSY5A		Busy Flag for CP A of DCP 5
7-1	Reserved	0	Reads return 0. Writes have no effect.
0	BUSY5B		Busy Flag for CP B of DCP 5

24.4.6 Control Packet (CP) Busy Register 3 (HTU BUSY3)

Figure 24-19. Control Packet (CP) Busy Register 3 (HTU BUSY3) [offset = 14h]

31 - 25	24	23 - 17	16
Reserved	BUSY6A	Reserved	BUSY6B
R-0	R/W1CP-0	R-0	R/W1CP-0

15 - 9	8	7 - 1	0
Reserved	BUSY7A	Reserved	BUSY7B
R-0	R/W1CP-0	R-0	R/W1CP-0

LEGEND: R/W = Read/Write; R = Read only; W1CP = Write 1 in privilege mode only to clear the bit; -*n* = value after reset

Table 24-18. Control Packet (CP) Busy Register 3 (HTU BUSY3) Field Descriptions

Bit	Field	Value	Description
31-25	Reserved	0	Reads return 0. Writes have no effect.
24	BUSY6A		Busy Flag for CP A of DCP 6
23-17	Reserved	0	Reads return 0. Writes have no effect.
16	BUSY6B		Busy Flag for CP B of DCP 6
15-9	Reserved	0	Reads return 0. Writes have no effect.
8	BUSY7A		Busy Flag for CP A of DCP 7
7-1	Reserved	0	Reads return 0. Writes have no effect.
0	BUSY7B		Busy Flag for CP B of DCP 7

24.4.7 Active Control Packet and Error Register (HTU ACPE)

Figure 24-20. Active Control Packet and Error Register (HTU ACPE) [offset = 18h]

31	30 - 29	28 - 24	23 - 20	19 - 16
ERRF	Reserved	ERRETC	Reserved	ERRCPN
R/W1CP-0	R-0	R-0	R-0	R-0

15	14	13	12 - 8	7 - 4	3 - 0
TIPF	BUSBUSY	Rsvd	CETCOUNT	Reserved	NACP
R-0	R-0	R-0	R-0	R-0	R-0

LEGEND: R/W = Read/Write; R = Read only; W1CP = Write 1 in privilege mode only to clear the bit; -*n* = value after reset

Table 24-19. Active Control Packet and Error Register (HTU ACPE) Field Descriptions

Bit	Field	Value	Description
31	ERRF		Error Flag
		0	No error occurred.
		1	This bit is set when one of the conditions listed at ERRETC is fulfilled and ERRETC and ERRCPN are captured. Once ERRF is set, it is cleared when reading the upper 16-bit word of the ACPE register or the complete 32-bit register. It is also cleared when writing a 1 to ERRF. ERRF can be used to indicate if ERRETC and ERRCPN contain new unread data.
30-29	Reserved	0	Reads return 0. Writes have no effect.

Table 24-19. Active Control Packet and Error Register (HTU ACPE) Field Descriptions (continued)

Bit	Field	Value	Description
28-24	ERRETC		Error Element Transfer Count If one of the following conditions happens the current element transfer counter of the control packet (specified by ERRCPN) is captured to ERRETC. Please see Section 24.2.3. • Request Lost Error of control packet specified by ERRCPN. This is independent of the CORL bit. • Parity Error of control packet specified by ERRCPN. This requires the parity check to be enabled, but is independent of the COPE bit. • Bus Error of control packet specified by ERRCPN. • Memory Protection Error of control packet specified by ERRCPN. This requires the memory protection to be enabled. • Writing a 1 to a BUSY bit, which belongs to the control packet specified by ERRCPN, if that bit is 1. There is no effect, if the BUSY bit is 0. ERRETC is frozen from being updated until the upper 16-bit word of the ACPE register or the complete 32-bit register is read by the CPU. After this read, the HTU will update ERRETC if one of the above conditions is fulfilled again. During debugging, ERRETC stays frozen even when reading the upper 16-bit word or the 32-bit register.
23-20	Reserved	0	Reads return 0. Writes have no effect.
19-16	ERRCPN		Error Control Packet Number If one of the conditions listed at ERRETC happens the number of the control packet, which caused the condition, is captured to ERRCPN. **Control Packet** **ERRCPN Value** CP A of DCP x 2 x CP B of DCP x 2 x+1 With x = 0,1,...or 7 ERRCPN is frozen from being updated until the upper 16-bit word of the ACPE register or the complete 32-bit register is read by the CPU. After this read, the HTU will update ERRCPN if one of the above conditions is fulfilled again. During debugging, ERRCPN stays frozen even when reading the upper 16-bit word or the 32-bit register. If one of the conditions is fulfilled, ERRETC and ERRCPN are updated simultaneously.
15	TIPF		Transfer in Progress Flag
		0	No transfers are in progress.
		1	A transfer is currently active. This bit is the result of a logical OR function of all BUSYxx flags of the 4 BUSYx registers.
14	BUSBUSY		Bus is Busy
		0	Bus between N2HET and HTU is not busy.
		1	When BUSBUSY is 1, the bus is busy with a transfer. It is different from TIPF above because BUSBUSY will go low after VBUSHOLD is set to 1 and no transfers are pending between the HTU and the main memory. TIPF will remain 1, if a transfer is still pending and VBUSPHOLD is 1.
13	Reserved	0	Reads return 0. Writes have no effect.
12-8	CETCOUNT		Current Element Transfer Count CETCOUNT shows the current element transfer counter for the frame that is currently processed. If the HTU does not currently transfer any frame, CETCOUNT is 0. CETCOUNT is updated after the write part of a transfer. There is a period of up to 7 cycles between the time the CETCOUNT is 0 and the HTU is finished updating the current DCP (and the CPENA registers, if the required conditions are fulfilled).
7-4	Reserved	0	Reads return 0. Writes have no effect.
3-0	NACP		Number of Active Control Packet Indicates which CP currently processes a frame. **Active or Recent DCP** **NACP Value** CP A of DCP x 2 x CP B of DCP x 2 x+1 With x = 0,1,...or 7 NACP is updated at the time the frame starts on the according CP, and it is updated with a new value when a frame starts on a different CP. Note, that there can be a delay between the request and the start of the frame.

24.4.8 Request Lost and Bus Error Control Register (HTU RLBECTRL)

Figure 24-21. Request Lost and Bus Error Control Register (HTU RLBECTRL) [offset = 20h]

<table>
<tr><td>31</td><td>17</td><td>16</td></tr>
<tr><td colspan="2">Reserved</td><td>BERINTENA</td></tr>
<tr><td colspan="2">R-0</td><td>R/WP-0</td></tr>
</table>

<table>
<tr><td>15</td><td>9</td><td>8</td><td>7</td><td>1</td><td>0</td></tr>
<tr><td colspan="2">Reserved</td><td>CORL</td><td colspan="2">Reserved</td><td>RLINTENA</td></tr>
<tr><td colspan="2">R-0</td><td>R/WP-0</td><td colspan="2">R-0</td><td>R/WP-0</td></tr>
</table>

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privilege mode only; *-n* = value after reset

Table 24-20. Request Lost and Bus Error Control Register (HTU RLBECTRL) Field Descriptions

Bit	Field	Value	Description
31-17	Reserved	0	Reads return 0. Writes have no effect.
16	BERINTENA		Bus Error Interrupt Enable Bit
		0	The bus error interrupt is disabled for all DCPs.
		1	The bus error interrupt is enabled for all DCPs.
15-9	Reserved	0	Reads return 0. Writes have no effect.
8	CORL		Continue On Request Lost Error
		0	Stop current frame on request lost detection. Please see Section 24.2.3.
		1	If CORL is 1 and DCP x is enabled, then DCP x will stay enabled after a request lost condition on DCP x and element transfers will continue.
7-1	Reserved	0	Reads return 0. Writes have no effect.
0	RLINTENA		Request Lost Interrupt Enable Bit
		0	The request lost interrupt is disabled for all DCPs. Disabling RLINTENA will not clear the flags in the RLOSTFL register.
		1	The request lost interrupt is enabled for all DCPs. If bits are set in the RLOSTFL flag register at the time RLINTENA is (re-) enabled, then the according interrupt(s) will occur (in the order of the priority of the request lines).

24.4.9 Buffer Full Interrupt Enable Set Register (HTU BFINTS)

This registers allows to enable the buffer full interrupts for the different control packets. Reading registers BFINTS and BFINTC will return the same bits indicating the status which interrupt is enabled (1) or disabled (0).

Figure 24-22. Buffer Full Interrupt Enable Set Register (HTU BFINTS) [offset = 24h]

31		16
	Reserved	
	R-0	

15		0
	BFINTENA	
	R/WP-0	

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privilege mode only; -*n* = value after reset

Table 24-21. Buffer Full Interrupt Enable Set Register (HTU BFINTS) Field Descriptions

Bit	Field	Value	Description
31-16	Reserved	0	Reads return 0. Writes have no effect.
15-0	BFINTENA		Bus Full Interrupt Enable Bits. If the interrupt for CP A of a DCP is enabled, then the interrupt is generated once buffer A is full, that is, once the frame counter CFTCTA decrements to 0. The same applies for CP B (and CFTCTB).
		0	Interrupt is disabled. Writing a 0 has no effect.
		1	Writing to bit (2*x) enables the interrupt for CP A of DCP x.
			Writing to bit (2*x+1) enables the interrupt for CP B of DCP x.

24.4.10 Buffer Full Interrupt Enable Clear Register (HTU BFINTC)

This registers allows to disable the buffer full interrupts for the different control packets. Reading registers BFINTS and BFINTC will return the same bits indicating the status which interrupt is enabled (1) or disabled (0)

Figure 24-23. Buffer Full Interrupt Enable Clear Register (HTU BFINTC) [offset = 28h]

31		16
	Reserved	
	R-0	

15		0
	BFINTDIS	
	R/WP-0	

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privilege mode only; -*n* = value after reset

Table 24-22. Buffer Full Interrupt Enable Clear Register (HTU BFINTC) Field Descriptions

Bit	Field	Value	Description
31-16	Reserved	0	Reads return 0. Writes have no effect.
15-0	BFINTDIS		Buffer Full Interrupt Disable Bits
		0	Interrupt is disabled. Writing a 0 has no effect.
		1	Writing to bit (2*x) disables the interrupt for CP A of DCP x.
			Writing to bit (2*x+1) disables the interrupt for CP B of DCP x.

24.4.11 Interrupt Mapping Register (HTU INTMAP)

Figure 24-24. Interrupt Mapping Register (HTU INTMAP) [offset = 2Ch]

31	17	16
Reserved		MAPSEL
R-0		R/WP-0

15	0
CPINTMAP	
R/WP-0	

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privilege mode only; -*n* = value after reset

Table 24-23. Interrupt Mapping Register (HTU INTMAP) Field Descriptions

Bit	Field	Value	Description
31-17	Reserved	0	Reads return 0. Writes have no effect.
16	MAPSEL		Interrupt Mapping Select Bit
		0	If MAPSEL is 0, then one bit of CPINTMAP selects one of two interrupt priorities 0 or 1 for the **buffer full** interrupt for the according CP. The **request lost** and **bus error** interrupts of all CPs are set to priority 0, independent of CPINTMAP.
		1	If MAPSEL is 1, then one bit of CPINTMAP determines if the **buffer full**, **request lost** and **bus error** interrupts of the according CP are assigned either to interrupt line 0 or to 1.
15-0	CPINTMAP		CP Interrupt Mapping Bits
		0	Interrupt of CP A (bit 2-x) of DCP x is mapped to interrupt line 0. Interrupt of CP B (bit 2*x+1) of DCP x is mapped to interrupt line 0.
		1	Interrupt of CP A (bit 2-x) of DCP x is mapped to interrupt line 1. Interrupt of CP B (bit 2*x+1) of DCP x is mapped to interrupt line 1.

24.4.12 Interrupt Offset Register 0 (HTU INTOFF0)

The INTOFF0 register reflects the highest priority interrupt flag bit set in the BERINTFL, RLOSTFL, or BFINTFL flag registers with the appropriate CPINTMAP bit set to 0. The priority order (from high to low) is: BER, RLOST, buffer-full. Interrupts for request lines with lower number have higher priority.

Figure 24-25. Interrupt Offset Register 0 (HTU INTOFF0) [offset = 34h]

31 - 16
Reserved
R-0

15 - 10	9 - 8	7 - 4	3 - 0
Reserved	INTTYPE0	Reserved	CPOFF0
R-0	R-0	R-0	R-0

LEGEND: R = Read only; -*n* = value after reset

Table 24-24. Interrupt Offset Register 0 (HTU INTOFF0) Field Descriptions

Bit	Field	Value	Description
31-10	Reserved	0	Reads return 0. Writes have no effect.
9-8	INTTYPE0		Interrupt Type of Interrupt Line 0. Indicates whether a buffer-full, RLOST, or BER interrupt, assigned to interrupt line 0, is currently pending.
		0	No interrupt.
		1h	Interrupt caused by full buffer on CP/DCP specified by CPOFF0.
		2h	RLOST interrupt generated by CP/DCP specified by CPOFF0.
		3h	BER interrupt generated by CP/DCP specified by bits CPOFF0.
7-4	Reserved	0	Reads return 0. Writes have no effect.
3-0	CPOFF0		CP Offset. Indicates for which control packet the interrupt is pending, which is classified by INTTYPE0 and is assigned to interrupt line 0.
		0	DCP 0, CP A
		1h	DCP 0, CP B
		2h	DCP 1, CP A
		3h	DCP 1, CP B
		4h	DCP 2, CP A
		5h	DCP 2, CP B
		6h	DCP 3, CP A
		7h	DCP 3, CP B
		8h	DCP 4, CP A
		9h	DCP 4, CP B
		Ah	DCP 5, CP A
		Bh	DCP 5, CP B
		Ch	DCP 6, CP A
		Dh	DCP 6, CP B
		Eh	DCP 7, CP A
		Fh	DCP 7, CP B

NOTE: Reading CPOFF0 will clear the bit generating the current interrupt from appropriate flag register (BERINTFL, RLOSTFL, or BFINTFL), except when in debug mode where reading CPOFF0 will have no effect on the flag registers.

In order to read INTTYPE0 and CPOFF0 simultaneously, always read this register using word or half-word but not using byte accesses.

24.4.13 Interrupt Offset Register 1 (HTU INTOFF1)

This register is organized identically to the INTOFF0 register. The difference is that INTOFF1 reflects the highest priority interrupt flag bit set in the BERINTFL, RLOSTFL, or BFINTFL flag registers with the appropriate CPINTMAP bit set to 1.

Figure 24-26. Interrupt Offset Register 1 (HTU INTOFF1) [offset = 38h]

31–16
Reserved
R-0

15–10	9–8	7–4	3–0
Reserved	INTTYPE1	Reserved	CPOFF1
R-0	R-0	R-0	R-0

LEGEND: R = Read only; -*n* = value after reset

Table 24-25. Interrupt Offset Register 1 (HTU INTOFF1) Field Descriptions

Bit	Field	Value	Description
31-10	Reserved	0	Reads return 0. Writes have no effect.
9-8	INTTYPE1		Interrupt Type of Interrupt Line 1. Indicates whether a buffer-full, RLOST, or BER interrupt, assigned to interrupt line 1, is currently pending.
		0	No interrupt.
		1h	Interrupt caused by full buffer on CP/DCP specified by CPOFF1.
		2h	RLOST interrupt generated by CP/DCP specified by CPOFF1.
		3h	BER interrupt generated by CP/DCP specified by bits CPOFF1.
7-4	Reserved	0	Reads return 0. Writes have no effect.
3-0	CPOFF1		CP Offset. Indicates for which DCP / CP the interrupt is pending, which is classified by INTTYPE1 and is assigned to interrupt line 1.
		0	DCP 0, CP A
		1h	DCP 0, CP B
		2h	DCP 1, CP A
		3h	DCP 1, CP B
		4h	DCP 2, CP A
		5h	DCP 2, CP B
		6h	DCP 3, CP A
		7h	DCP 3, CP B
		8h	DCP 4, CP A
		9h	DCP 4, CP B
		Ah	DCP 5, CP A
		Bh	DCP 5, CP B
		Ch	DCP 6, CP A
		Dh	DCP 6, CP B
		Eh	DCP 7, CP A
		Fh	DCP 7, CP B

NOTE: Reading CPOFF1 will clear the bit generating the current interrupt from appropriate flag register (BERINTFL, RLOSTFL, or BFINTFL), except when in debug mode where reading CPOFF1 will have no effect on the flag registers.

In order to read INTTYPE1 and CPOFF1 simultaneously, always read this register using word or half-word but not using byte accesses.

24.4.14 Buffer Initialization Mode Register (HTU BIM)

This register enables special applications, where one CP is temporarily disabled, but after having re-enabled the CP, filling the buffer should not start back at its beginning, but should continue after the last element of the previous run.

Table 24-27 shows more details on the BIM usage.

Figure 24-27. Buffer Initialization Mode Register (HTU BIM) [offset = 3Ch]

31 ... 16
Reserved
R-0

15 ... 8	7 ... 0
Reserved	BIM
R-0	R/WP-0

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privilege mode only; -*n* = value after reset

Table 24-26. Buffer Initialization Mode Register (HTU BIM) Field Descriptions

Bit	Field	Value	Description
31-8	Reserved	0	Reads return 0. Writes have no effect.
7-0	BIM		Buffer Initialization Mode The BIM bits and the TMBx bits determine when a buffer is initialized, that means when its initial full address IFADDRx and its initial frame counter IFTCOUNT is used. When initializing (restarting) a buffer the information in the corresponding initial DCP RAM is loaded to a internal state machine but not to the current DCP RAM (CFADDRx, CFTCTx). The current DCP RAM is updated the first time when the first frame has finished. A buffer is initialized: • In circular buffer transfer mode (defined by TMBx) when the end of the buffer is reached. • When CPs are switched or enabled according to Buffer Initialization. The CPENA bits (2*x+1) and (2*x) are changed by write access to CPENA. For the first two rows of the table, the change of the CPENA bits could also be the result of the auto switch feature (as defined by TMBx). BIM bit x only affects DCP x (with x = 0,1,...or 7).

Table 24-27. Buffer Initialization

Case	Change of CPENA bits (2*x+1) and (2*x)		Action on buffer A or B (of DCP x)		
	Old state [1]	New state [2]		BIM bit x = 0 (normal mode)	BIM bit x = 1 (special mode)
A	01	10	Switch from CP A to B	Next frame starts at the initial address of buffer B [3]	Same as for BIM bit x = 0
B	10	01	Switch from CP B to A	Next frame starts at the initial address of buffer A [3]	Same as for BIM bit x = 0
C	01	01	Stay at CP A	Write to CPENA bits (2*x+1) and (2*x) is ignored	Same as for BIM bit x = 0
E	10	10	Stay at CP B	Write to CPENA bits (2*x+1) and (2*x) is ignored	Same as for BIM bit x = 0
E	00	01	Enable CP A	Next frame starts at the **initial** address of buffer A	*Next frame continues at the **current** address of buffer A*
F	00	10	Enable CP B	Next frame starts at the **initial** address of buffer B	*Next frame continues at the **current** address of buffer B*
G	xx	11	Disable both CPs	Stop DCP x	Same as for BIM bit x = 0

[1] See read table of CPENA register (Table 24-14).

[2] See write table of CPENA register (Table 24-13).

[3] This is regardless of whether the switch is done by a write access to CPENA or by the auto-switch feature.

NOTE: For cases E and F above, after the last frame of a buffer, the HTU sets CFTCTx to 0 and CFADDRx to the next address after the buffer. If the DCP was disabled during this state, then both CFTCTx and CFADDRx would contain invalid initialization values. Therefore, if a DCP should continue at its current address, then the software should use one of the following two procedures before it (re-) enables the DCP (as per Table 24-27):

1.
```
If CFTCTx ≠ 0 then set BIM=1
If CFTCTx = 0 then set
                BIM=0
```
2.
```
If CFTCTx ≠ 0 then set
                BIM=1
If CFTCTx = 0 then {set
                BIM=1;
```

```
set CFTCTx = IFTCOUNT;
set CFADDRx = IFADDRx}
```

But note that these procedures are only required for the cases E and F and not for all the other cases shown in Table 24-27. Also, when a buffer reaches its end in circular mode, it uses the initial DCP information to restart independently of the BIM setting (assuming it is not temporarily disabled during CFTCTx = 0).

NOTE: Similarly, care needs to be taken when BIM is set to 1 and a DCP is enabled for the very first time. Also, in this case, CFTCTx and CFADDRx usually contain invalid initialization values. The software can either solve this by setting BIM = 0 for the first time or setting CFADDRx to IFADDRx and CFTCTx to IFTCOUNT before the DCP is enabled.

NOTE: If

- the HTUEN bit is changed to 1 after the HTU was disabled HTUEN = 0
- the CPENA bit pair is 01 or 10 (during this HTUEN change)

then the corresponding BIM bit will decide if the corresponding buffer continues at its initial or current address. Cases E and F in Table 24-27 also apply for this situation. The software should use the procedures explained in the first note before setting HTUEN.

24.4.15 Request Lost Flag Register (HTU RLOSTFL)

Figure 24-28. Request Lost Flag Register (HTU RLOSTFL) [offset = 40h]

31	16
Reserved	
R-0	

15	0
CPRLFL	
R/W1CP-0	

LEGEND: R/W = Read/Write; R = Read only; W1CP = Write 1 in privilege mode to clear the bit; -*n* = value after reset

Table 24-28. Request Lost Flag Register (HTU RLOSTFL) Field Descriptions

Bit	Field	Value	Description
31-16	Reserved	0	Reads return 0. Writes have no effect.
15-0	CPRLFL		CP Request Lost Flags
		0	No request was lost. Writing a 0 has no effect.
		1	If bit (2*x) is set, a request was lost on CP A of DCP x.
			If bit (2*x+1) is set, a request was lost on CP B of DCP x. Reading from INTOFFx in case of a RLOST interrupt clears the corresponding flag. The state of the flag bit can be polled even if RLINTENA is cleared.
			• Reading CPRLFL will not clear the flags or • Reading from INTOFFx clears the corresponding flag. • Writing a 1 clears the corresponding flag.

24.4.16 Buffer Full Interrupt Flag Register (HTU BFINTFL)

Figure 24-29. Buffer Full Interrupt Flag Register (HTU BFINTFL) [offset = 44h]

31	16
Reserved	
R-0	

15	0
BFINTFL	
R/W1CP-0	

LEGEND: R/W = Read/Write; R = Read only; W1CP = Write 1 in privilege mode to clear the bit; -*n* = value after reset

Table 24-29. Buffer Full Interrupt Flag Register (HTU BFINTFL) Field Descriptions

Bit	Field	Value	Description
31-16	Reserved	0	Reads return 0. Writes have no effect.
15-0	BFINTFL		Buffer Full Interrupt Flags
		0	No buffer full condition is detected. Writing a 0 has no effect.
		1	If bit (2*x) is set, a buffer full condition on CP A of DCP x has been detected.
			If bit (2*x+1) is set, a buffer full condition on CP B of DCP x has been detected.
			The BFINTFL flag is set after the last frame finishes on the corresponding buffer regardless of whether the buffer is configured to one shot, circular or auto-switch mode. If BFINTFL is set in circular mode, then a circular overrun has occurred on the corresponding buffer. This can be used to indicate whether the buffer section after the frozen full address contains valid data or not.
			Reading from INTOFFx in case of a buffer-full interrupt clears the corresponding flag. The state of the flag bit can be polled even if the corresponding interrupt enable bit is cleared.
			• Reading BFINTFL will not clear the flags or • Reading INTOFFx will clear the corresponding flags or • Writing a 1 clears the corresponding flag.

24.4.17 BER Interrupt Flag Register (HTU BERINTFL)

A bus error interrupt results due to an address error or a timeout condition on the main memory access. A bus error will stop the frame transfer. Please see Section 24.2.3.

Figure 24-30. BER Interrupt Flag Register (HTU BERINTFL) [offset = 48h]

31	16
Reserved	
R-0	

15	0
BERINTFL	
R/W1CP-0	

LEGEND: R/W = Read/Write; R = Read only; W1CP = Write 1 in privilege mode to clear the bit; -*n* = value after reset

Table 24-30. BER Interrupt Flag Register (HTU BERINTFL) Field Descriptions

Bit	Field	Value	Description
31-16	Reserved	0	Reads return 0. Writes have no effect.
15-0	BERINTFL		Bus Error Interrupt Flags
		0	No bus error condition is detected. Writing a 0 has no effect.
		1	If bit (2*x) is set, then a BER interrupt is pending on CP A of DCP x. If bit (2*x+1) is set, then a BER interrupt is pending on CP B of DCP x. The state of the flag bit can be polled even if BERINTENA is cleared. • Reading BERINTFL will not clear the flags or • Reading from INTOFFx in case of a BER interrupt clears the corresponding flag or • Writing a 1 clears the corresponding flag.

24.4.18 Memory Protection 1 Start Address Register (HTU MP1S)

This register configures the start address of memory protection region 1.

Figure 24-31. Memory Protection 1 Start Address Register (HTU MP1S) [offset = 4Ch]

31	16
STARTADDRESS1	
R/WP-0	

15	2	1	0
STARTADDRESS1		0	0
R/WP-0			

LEGEND: R/W = Read/Write; WP = Write in privilege mode only; -*n* = value after reset

Table 24-31. Memory Protection 1 Start Address Register (HTU MP1S) Field Descriptions

Bit	Field	Description
31-0	STARTADDRESS1	The start address defines at which main memory address the region begins. A memory protection error will be triggered, if the HTU accesses an address smaller than STARTADDRESS1 and the MPCS bit REG01ENA register is configured accordingly. The address is 32-bit aligned, so the 2 LSBs are not significant and will always read 0.

24.4.19 Memory Protection 1 End Address Register (HTU MP1E)

Figure 24-32. Memory Protection 1 End Address Register (HTU MP1E) [offset = 50h]

31	16
ENDADDRESS1	
R/WP-0	

15	2	1	0
ENDADDRESS1		0	0
R/WP-0			

LEGEND: R/W = Read/Write; WP = Write in privilege mode only; -*n* = value after reset

Table 24-32. Memory Protection 1 End Address Register (HTU MP1E) Field Descriptions

Bit	Field	Description
31-0	ENDADDRESS1	The end address defines at which address the region ends. A memory protection error will be triggered, if the HTU accesses an address bigger than ENDADDRESS1 and the register bit REG01ENA is configured accordingly. The address is 32-bit aligned, so the 2 LSBs are not significant and will always read 0. The effective end address is rounded up to the nearest word end address, that is, 0x200 = 0x203.

24.4.20 Debug Control Register (HTU DCTRL)

This register allows to create watch points on access to a certain location. It is intended to help debug the application execution during program development.

Figure 24-33. Debug Control Register (HTU DCTRL) [offset = 54h]

31 28	27 24	23 17	16
Reserved	CPNUM	Reserved	HTUDBGS
R-0	R-0	R-0	R/W1CS-0

15 1	0
Reserved	DBREN
R-0	R/WS-0

LEGEND: R/W = Read/Write; R = Read only; W1CS = Write 1 in suspend mode to clear the bit; WS = Write in suspend mode only; -*n* = value after reset

Table 24-33. Debug Control Register (HTU DCTRL) Field Descriptions

Bit	Field	Value	Description
31-28	Reserved	0	Reads return 0. Writes have no effect.
27-24	CPNUM		CP Number. These bit fields indicate the CP that should cause the watch point to match.
		0	CP A of DCP0
		1h	CP B of DCP0
		2h	CP A of DCP1
		3h	CP B of DCP1
		4h	CP A of DCP2
		5h	CP B of DCP2
		6h	CP A of DCP3
		7h	CP B of DCP3
		8h	CP A of DCP4
		9h	CP B of DCP4
		Ah	CP A of DCP5
		Bh	CP B of DCP5
		Ch	CP A of DCP6
		Dh	CP B of DCP6
		Eh	CP A of DCP7
		Fh	CP B of DCP7
23-17	Reserved	0	Reads return 0. Writes have no effect.
16	HTUDBGS		HTU Debug Status. When the main memory address is equal to the unique address defined by WPR, or lies in the specified range resulting from WMR, then the HTUDBGS is set. If in addition DBREN is set, then the application code execution will be stopped. A 1 must be written to this bit in order to clear it and to release the CPU from debug halting state.
		0	Read: No watch point condition was detected. Write: No effect.
		1	Read: A watch point condition was detected. Write: Clears the bit.
15-1	Reserved	0	Reads return 0. Writes have no effect.
0	DBREN		Debug Request Enable If a watch point matches and DBREN is set, then the application code execution will be stopped. This bit can only be set or cleared when in debug mode. This bit and all other bits of the DCTRL, WPR and WMR registers are reset by the test reset (nTRST) but not by the normal device reset.

24.4.21 Watch Point Register (HTU WPR)

This register defines the main memory address of the watch point.

Figure 24-34. Watch Point Register (HTU WPR) [offset = 58h]

31	16
WP	
R/WS-0	

15	0
WP	
R/WS-0	

LEGEND: R/W = Read/Write; WS = Write in suspend mode only; -*n* = value after reset

Table 24-34. Watch Point Register (HTU WPR) Field Descriptions

Bit	Field	Description
31-0	WP	Watch Point Register A 32-bit address can be programmed into this register as a watch point. The WPR register is used along with the Watch Mask Register (WMR). When the main memory address is equal to the unique address defined by WPR, or lies in the specified range resulting from WMR, then the HTUDBGS is set. If in addition DBREN is set, then the application code execution is stopped. This register can only be programmed during debug mode. This register and all other bits of the DCTRL and WMR registers are reset by the test reset (nTRST) but not by the normal device reset.

24.4.22 Watch Mask Register (HTU WMR)

This register defines a mask of the main memory address of the watch point. It can be used to define a memory range in conjunction with the WPR register.

Figure 24-35. Watch Mask Register (HTU WMR) [offset = 5Ch]

31	16
WM	
R/WS-0	

15	0
WM	
R/WS-0	

LEGEND: R/W = Read/Write; WS = Write in suspend mode only; -*n* = value after reset

Table 24-35. Watch Mask Register (HTU WMR) Field Descriptions

Bit	Field	Description
31-0	WM	Watch Mask Register Setting a bit in the WMR register to 1 has the effect of masking the corresponding bit in of the main memory address, so that this bit is ignored for the address comparison. This register can only be programmed during debug mode. This register and all other bits of the DCTRL and WPR registers are reset by the test reset (nTRST) but not by the normal device reset.

24.4.23 Module Identification Register (HTU ID)

This register is for TI internal purposes and allows to keep track of the HTU module version on different devices.

Figure 24-36. Module Identification Register (HTU ID) [offset = 60h]

31 — 24	23 — 16
Reserved	CLASS
R-0	R - Module Class Number

15 — 8	7 — 0
TYPE	REV
R - Class Subtype Number	R - Module Revision Number

LEGEND: R = Read only; -*n* = value after reset

Table 24-36. Module Identification Register (HTU ID) Field Descriptions

Bit	Field	Value	Description
31-24	Reserved	0	Reads return 0. Writes have no effect.
23-16	CLASS		Module Class This field defines the module class number as read-only constant value for the HTU module. Writes have no effect.
15-8	TYPE		Subtype within a Class This field defines the subtype within a class as read-only constant value for the HTU module. Writes have no effect.
7-0	REV		Module Revision Number This field defines the module revision number as read-only constant value for the HTU module. Writes have no effect.

24.4.24 Parity Control Register (HTU PCR)

Figure 24-37. Parity Control Register (HTU PCR) [offset = 64h]

31 – 17	16
Reserved	COPE
R-0	R/WP-0

15 – 9	8	7 – 4	3 – 0
Reserved	TEST	Reserved	PARITY_ENA
R-0	R/WP-0	R-0	R/WP-5h

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privileged mode only; -*n* = value after reset

Table 24-37. Parity Control Register (HTU PCR) Field Descriptions

Bit	Field	Value	Description
31-17	Reserved	0	Reads return 0. Writes have no effect.
16	COPE		Continue on Parity Error
		0	The HTU performs parity checks every time it reads the RAM section of DCP x (with x = 0, 1,... or 7), before the next frame (of DCP x) is started. If a parity error is detected during this read access and if the parity check is enabled, then the frame will not be started and DCP x will be automatically disabled in the CPENA register. If a master different than the HTU (for example, CPU) reads the RAM section of DCP x and a parity error is detected during this read access, while the parity check is enabled, then the DCP x will automatically be disabled in the CPENA register. If a frame is active on DCP x during this read access, then in addition the element counter of DCP x is cleared and all new element transfers on DCP x are stopped and the active busy bit of DCP x is cleared.
		1	The difference to COPE = 0 is, that the data transfer on a active DCP continues after a parity error was detected on this DCP. So, neither the DCP with the parity error will be disabled nor the frame will be stopped.
15-9	Reserved	0	Reads return 0. Writes have no effect.
8	TEST		Test. When this bit is set, the parity bits are mapped into the peripheral RAM frame to make them accessible by the CPU.
		0	Parity bits are not memory-mapped.
		1	Parity bits are memory-mapped.
7-4	Reserved	0	Reads return 0. Writes have no effect.
3-0	PARITY_ENA		Enable/Disable Parity Checking. This bit field enables or disables the parity check on read operations and the parity calculation on write operations. If parity checking is enabled and a parity error is detected, then the PEFT flag is set, PAOFF is captured if it is not currently frozen and an interrupt is generated if it is enabled.
		5h	Parity check is disabled.
		All Others	Parity check is enabled. **Note:** It is recommended to write Ah to enable error detection, to guard against single bit changes from flipping PARITY_ENA to a disable state.

24.4.25 Parity Address Register (HTU PAR)

Figure 24-38. Parity Address Register (HTU PAR) [offset = 68h]

31 17	16
Reserved	PEFT
R-0	R/W1CP-0

15 9	8 0
Reserved	PAOFF
R-0	R-X

LEGEND: R/W = Read/Write; R = Read only; W1CP = Write 1 in privilege mode to clear the bit; -*n* = value after reset; X = undefined

Table 24-38. Parity Address Register (HTU PAR) Field Descriptions

Bit	Field	Value	Description
31-10	Reserved	0	Reads return 0. Writes have no effect.
16	PEFT		Parity Error Fault Flag. This bit is set, when the HTU detects a parity error and parity checking is enabled.
		0	No fault is detected.
		1	Fault is detected.
			Note: Once PEFT is set, a read access to the lower 16 bits or to the complete 32-bit HTUPAR register will clear the PEFT flag in non-debug mode. Another possibility to clear PEFT is to write a 1 to the PEFT bit.
15-9	Reserved	0	Reads return 0. Writes have no effect.
8-0	PAOFF		Parity Error Address Offset. This bit field holds the address of the first parity error, which is detected in the DCP RAM. PAOFF provides the offset address of the erroneous byte counted from the beginning of the DCP memory. This error address is frozen from being updated until a read access to the lower 16 bits or to the complete 32-bit HTUPAR register happens. During debug mode, this address is frozen even when read. **Note:** The Parity Error Address bits will not be reset, neither by PORRST nor by any other reset source.

24.4.26 Memory Protection Control and Status Register (HTU MPCS)

Figure 24-39. Memory Protection Control and Status Register (HTU MPCS) [offset = 70h]

31	28	27	24
Reserved		CPNUM0	
R-0		R-0	

23	18	17	16
Reserved		MPEFT1	MPEFT0
R-0		R/W1CP-0	R/W1CP-0

15	12	11	8
Reserved		CPNUM1	
R-0		R-0	

7	6	5	4	3	2	1	0
Reserved		INT ENA01	ACCR01	REG01ENA	INT ENA0	ACCR0	REG0ENA
R-0		R/WP-0	R/WP-0	R/WP-0	R/WP-0	R/WP-0	R/WP-0

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privilege mode only; W1CP = Write 1 in privilege mode to clear the bit; -*n* = value after reset

Table 24-39. Memory Protection Control and Status Register (HTU MPCS) Field Descriptions

Bit	Field	Value	Description
31-28	Reserved	0	Reads return 0. Writes have no effect.
27-24	CPNUM0		Control Packet Number for single memory protection region configuration. CPNUM0 holds the number of the CP, which has caused the first memory protection error when only one memory protection region is used. This number is not updated for multiple access violations until it is read by the CPU. During debug mode, CPNUM0 is frozen even when read.
		0	CP A of DCP0
		1h	CP B of DCP0
		2h	CP A of DCP1
		3h	CP B of DCP1
		4h	CP A of DCP2
		5h	CP B of DCP2
		6h	CP A of DCP3
		7h	CP B of DCP3
		8h	CP A of DCP4
		9h	CP B of DCP4
		Ah	CP A of DCP5
		Bh	CP B of DCP5
		Ch	CP A of DCP6
		Dh	CP B of DCP6
		Eh	CP A of DCP7
		Fh	CP B of DCP7
23-18	Reserved	0	Reads return 0. Writes have no effect.
17	MPEFT1		Memory Protection Error Fault Flag 1. This bit is set, when the HTU performs an access outside the region defined by the MP0S and MP0E and the MP1S and MP1E registers, when the access violates the rights defined by ACCR01, and when the REG01ENA bit is set.
		0	No fault detected. Writing a 0 has no effect.
		1	Fault detected. Writing a 1 will clear the bit.
16	MPEFT0		Memory Protection Error Fault Flag 0. This bit is set, when the HTU performs an access outside the region defined by the MP0S and MP0E registers, when the access violates the rights defined by ACCR, and when the REG0ENA bit is set.
		0	No fault detected. Writing a 0 has no effect.
		1	Fault detected. Writing a 1 will clear the bit.

Table 24-39. Memory Protection Control and Status Register (HTU MPCS) Field Descriptions (continued)

Bit	Field	Value	Description
15-12	Reserved	0	Reads return 0. Writes have no effect.
11-8	CPNUM1		Control Packet Number for single memory protection region configuration. CPNUM1 holds the number of the CP, which has caused the first memory protection error when only one memory protection region is used. This number is not updated for multiple access violations until it is read by the CPU. During debug mode, CPNUM1 is frozen even when read.
		0	CP A of DCP0
		1h	CP B of DCP0
		2h	CP A of DCP1
		3h	CP B of DCP1
		4h	CP A of DCP2
		5h	CP B of DCP2
		6h	CP A of DCP3
		7h	CP B of DCP3
		8h	CP A of DCP4
		9h	CP B of DCP4
		Ah	CP A of DCP5
		Bh	CP B of DCP5
		Ch	CP A of DCP6
		Dh	CP B of DCP6
		Eh	CP A of DCP7
		Fh	CP B of DCP7
7-6	Reserved	0	Reads return 0. Writes have no effect.
5	INTENA01		Interrupt Enable 01. This bit needs to be set when working with two memory-mapped regions and a error should be generated to the ESM module on an access violation.
		0	Error signaling is disabled.
		1	Error signaling is enabled.
4	ACCR01		Access Rights 01. This bit defines the access rights for the HTU for accesses outside the region defined by the MP0S and MP0E and the MP1S and MP1E registers.
		0	HTU read access is allowed but write access will be signaled.
		1	Any access performed by the HTU is forbidden and will be signaled.
3	REG01ENA		Region Enable 01. This bit needs to be set when working with two memory-mapped regions. **REG0ENA must be cleared to 0** if this bit is set to a 1. Memory region 0 must be less than memory region 1.
		0	The protection outside the memory region defined by the MP0S and MP0E and the MP1S and MP1E registers is not enabled. This means the HTU can access any implemented memory space. REG0ENA could still enabled to give protection outside the MP0S:MP0E region.
		1	The protection outside the memory region defined by the MP0S and MP0E and the MP1S and MP1E registers is enabled. This means the HTU can perform any access within the regions, but if it attempts to perform a forbidden access outside of both of the regions (according to the ACCR01 configuration), the access is signaled by the MPEFT1 flag. The number of the CP, which has caused the memory protection error, is captured to CPNUM1 if it is not currently frozen and an error is generated if it is enabled.
2	INTENA0		Interrupt Enable 0. This bit needs to be set when working with one memory-mapped region and a error should be generated to the ESM module on an access violation.
		0	Error signaling is disabled.
		1	Error signaling is enabled.
1	ACCR		Access Rights 0. This bit defines the access rights for the HTU for accesses outside the region defined by the MP0S and MP0E registers for a single memory protection region configuration.
		0	HTU read access is allowed but write access will be signaled.
		1	Any access performed by the HTU is forbidden and will be signaled.

Table 24-39. Memory Protection Control and Status Register (HTU MPCS) Field Descriptions (continued)

Bit	Field	Value	Description
0	REG0ENA		Region Enable 0
		0	The protection outside the memory region defined by the MP0S and MP0E registers is not enabled. This means the HTU can access any implemented memory space.
		1	The protection outside the memory region defined by the MP0S and MP0E registers is enabled. This means the HTU can perform any access within the region, but if it attempts to perform a forbidden access outside the region (according to the ACCR configuration), the access is signaled by the MPEFT0 flag, the number of the CP, which has caused the memory protection error, is captured to CPNUM0 if it is not currently frozen and an error is generated if it is enabled.

24.4.27 Memory Protection Start Address Register 0 (HTU MP0S)

This register configures the start address of memory protection region 0

Figure 24-40. Memory Protection Start Address Register 0 (HTU MP0S) [offset = 74h]

31–16
STARTADDRESS0
R/WP-0

15–2	1	0
STARTADDRESS0	0	0
R/WP-0		

LEGEND: R/W = Read/Write; WP = Write in privilege mode only; -*n* = value after reset

Table 24-40. Memory Protection 0 Start Address Register (HTU MP0S) Field Descriptions

Bit	Field	Description
31-0	STARTADDRESS0	The start address defines at which main memory address the region begins. A memory protection error will be triggered, if the HTU accesses an address smaller than STARTADDRESS0 and the MPCS register is configured accordingly. The address is 32-bit aligned, so the 2 LSBs are not significant and will always read 0.

24.4.28 Memory Protection End Address Register (HTU MP0E)

Figure 24-41. Memory Protection End Address Register (HTU MP0E) [offset = 78h]

31–16
ENDADDRESS0
R/WP-0

15–2	1	0
ENDADDRESS0	0	0
R/WP-0		

LEGEND: R/W = Read/Write; WP = Write in privilege mode only; -*n* = value after reset

Table 24-41. Memory Protection End Address Register (HTU MP0E) Field Descriptions

Bit	Field	Description
31-0	ENDADDRESS0	The end address defines at which address the region ends. A memory protection error will be triggered, if the HTU accesses an address bigger than ENDADDRESS0 and the register bit MPCS register is configured accordingly. The address is 32-bit aligned, so the 2 LSBs are not significant and will always read 0. The effective end address is rounded up to the nearest word end address, that is, 0x200 = 0x203.

24.5 Double Control Packet Configuration Memory

All bits marked "reserved' are implemented in RAM and will be initialized to unknown values after power on. Reserved locations can be written and read but should be written with 0 to ensure future compatibility. The HTU RAM can be cleared with the system RAM initialization function.

Table 24-42 provides a summary of the memory configuration. There are eight sets of DCP registers and eight sets of CF registers. The base address for the DCP registers is FF4E 0000h for HTU1 and FF4C 0000h for HTU2.

Table 24-42. Double Control Packet Memory Map

Offset	Acronym	Register Description	Section
00h	HTU DCP0 IFADDRA	Initial Full Address A Register	Section 24.5.1
04h	HTU DCP0 IFADDRB	Initial Full Address B Register	Section 24.5.2
08h	HTU DCP0 IHADDRCT	Initial N2HET Address and Control Register	Section 24.5.3
0Ch	HTU DCP0 ITCOUNT	Initial Transfer Count Register	Section 24.5.4
10h	HTU DCP1 IFADDRA	Initial Full Address A Register	Section 24.5.1
14h	HTU DCP1 IFADDRB	Initial Full Address B Register	Section 24.5.2
18h	HTU DCP1 IHADDRCT	Initial N2HET Address and Control Register	Section 24.5.3
1Ch	HTU DCP1 ITCOUNT	Initial Transfer Count Register	Section 24.5.4
:	:	:	
70h	HTU DCP7 IFADDRA	Initial Full Address A Register	Section 24.5.1
74h	HTU DCP7 IFADDRB	Initial Full Address B Register	Section 24.5.2
78h	HTU DCP7 IHADDRCT	Initial N2HET Address and Control Register	Section 24.5.3
7Ch	HTU DCP7 ITCOUNT	Initial Transfer Count Register	Section 24.5.4
100h	HTU CDCP0 CFADDRA	Current Full Address A Register	Section 24.5.5
104h	HTU CDCP0 CFADDRB	Current Full Address B Register	Section 24.5.6
108h	HTU CDCP0 CFCOUNT	Current Frame Count Register	Section 24.5.7
110h	HTU CDCP1 CFADDRA	Current Full Address A Register	Section 24.5.5
114h	HTU CDCP1 CFADDRB	Current Full Address B Register	Section 24.5.6
118h	HTU CDCP1 CFCOUNT	Current Frame Count Register	Section 24.5.7
:	:	:	
170h	HTU CDCP7 CFADDRA	Current Full Address A Register	Section 24.5.5
174h	HTU CDCP7 CFADDRB	Current Full Address B Register	Section 24.5.6
178h	HTU CDCP7 CFCOUNT	Current Frame Count Register	Section 24.5.7

24.5.1 Initial Full Address A Register (HTU IFADDRA)

Figure 24-42. Initial Full Address A Register (HTU IFADDRA)

31	16
IFADDRA	
R/WP-X	

15	0
IFADDRA	
R/WP-X	

LEGEND: R/W = Read/Write; WP = Write in privilege mode only; -*n* = value after reset; X = Unknown

Table 24-43. Initial Full Address A Register (HTU IFADDRA) Field Descriptions

Bit	Field	Description
31-0	IFADDRA	Initial Address of Buffer A in main memory. Initial (byte) address of buffer A placed in the main memory address range. Bits 0 and 1 are ignored by the logic, due to 32-bit alignment.

24.5.2 Initial Full Address B Register (HTU IFADDRB)

Figure 24-43. Initial Full Address B Register (HTU IFADDRB)

31	16
IFADDRB	
R/WP-X	

15	0
IFADDRB	
R/WP-X	

LEGEND: R/W = Read/Write; WP = Write in privilege mode only; -*n* = value after reset; X = Unknown

Table 24-44. Initial Full Address B Register (HTU IFADDRB) Field Descriptions

Bit	Field	Description
31-0	IFADDRB	Initial Address of Buffer B in main memory. Initial (byte) address of buffer B placed in the main memory address range. Bits 0 and 1 are ignored by the logic, due to 32-bit alignment.

24.5.3 Initial N2HET Address and Control Register (HTU IHADDRCT)

Figure 24-44. Initial N2HET Address and Control Register (HTU IHADDRCT)

31	24
Reserved	
R-0	

23	22	21	20	19	18	17	16
DIR	SIZE	ADDMH	ADDMF	TMBA		TMBB	
R/WP-X	R/WP-X	R/WP-X	R/WP-X	R/WP-X		R/WP-X	

15	13	12	2	1	0
Reserved		IHADDR		Reserved	
R-0		R/WP-X		R-0	

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privilege mode only; -*n* = value after reset; X = Unknown

Table 24-45. Initial N2HET Address and Control Register (HTU IHADDRCT) Field Descriptions

Bit	Field	Value	Description
31-24	Reserved	0	Reads return 0. Writes have no effect.
23	DIR		Direction of Transfer
		0	N2HET address is read and main memory address is written.
		1	Main memory address is read and N2HET address is written.
22	SIZE		Size of Transferred Data
		0	32-bit transfer
		1	64-bit transfer
			64-bit transfer examples: If the N2HET address points to the N2HET instruction Control Field (CF), then the CF and Data Field (DF) will be transferred. If the N2HET address points to the Program Field (PF), then the PF and CF will be transferred.
21	ADDMH		Addressing Mode N2HET Address. This bit determines the N2HET address index from one to the next element of a frame.
		0	Increment by 16 bytes.
			Examples:
			If the initial N2HET address points to data field of instruction (n). Then the N2HET fields to be transferred by the elements of a frame are: data field of instruction (n), data field of instruction (n+1), data field of instruction (n+2) and so on. If the initial N2HET address points to control field of instruction (n), then the N2HET fields to be transferred by the elements of a frame are: control field of instruction (n), control field of instruction (n+1), control field of instruction (n+2) and so on.
		1	Increment by 8 bytes.
			This mode is intended to be used together with the 64-bit transfer size to load short N2HET instruction blocks into the N2HET RAM. So the sequence of transferred 64-bit elements could be: [PF and CF of instruction (n)], [DF and RF of instruction (n)], [PF and CF of instruction (n+1)], [DF and RF of instruction (n+1)] and so on.
20	ADDMF		Addressing Mode Main Memory Address
		0	Post-increment
			Note: When post-increment is selected the HTU will automatically increment by 4 bytes for a 32-bit data size and by 8 bytes for a 64-bit data size.
		1	Constant
19-18	TMBA		Transfer Mode for Buffer A
		0	One-Shot buffer mode
		1h	Circular buffer mode
		2h-3h	Auto Switch mode
17-16	TMBB		Transfer Mode for Buffer B
		0	One-Shot buffer mode
		1h	Circular buffer mode
		2h-3h	Auto Switch mode

Table 24-45. Initial N2HET Address and Control Register (HTU IHADDRCT) Field Descriptions (continued)

Bit	Field	Value	Description
15-13	Reserved	0	Reads return 0. Writes have no effect.
12-2	IHADDR		Initial N2HET Address The initial N2HET Address points to the N2HET field, which is the first element of the frame. The N2HET address (bits 12:2) increments by 1 for each 32-bit N2HET field and starts with 0 at the first 32-bit field in the N2HET RAM. **Note:** When the HTU addresses the N2HET RAM it uses only the number of address bits required for the actual N2HET RAM size. If the N2HET address exceeds the actual N2HET RAM size, the unused MSB bits of the address will be ignored and the address rolls over to the start of the N2HET RAM.
1-0	Reserved	0	Reads return 0. Writes have no effect.

24.5.4 Initial Transfer Count Register (HTU ITCOUNT)

Figure 24-45. Initial Transfer Count Register (HTU ITCOUNT)

31-21	20-16
Reserved	IETCOUNT
R-0	R/WP-X

15-8	7-0
Reserved	IFTCOUNT
R-0	R/WP-X

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privilege mode only; -*n* = value after reset; X = Unknown

Table 24-46. Initial Transfer Count Register (HTU ITCOUNT) Field Descriptions

Bit	Field	Value	Description
31-21	Reserved	0	Reads return 0. Writes have no effect.
20-16	IETCOUNT		Initial Element Transfer Count Defines the number of element transfers.
15-8	Reserved	0	Reads return 0. Writes have no effect.
7-0	IFTCOUNT		Initial Frame Transfer Count Defines the number of frame transfers.

24.5.5 Current Full Address A Register (HTU CFADDRA)

Figure 24-46. Current Full Address A Register (HTU CFADDRA)

31	16
CFADDRA	
R/WP-X	

15	0
CFADDRA	
R/WP-X	

LEGEND: R/W = Read/Write; WP = Write in privilege mode only; -*n* = value after reset; X = Unknown

Table 24-47. Current Full Address A Register (HTU CFADDRA) Field Descriptions

Bit	Field	Description
31-0	CFADDRA	Current (byte) Address of Buffer A The current main memory address register is updated at the end of each frame. Therefore it points to the start address of the frame, which is the next to transfer, if currently no frame is transferred on this DCP. For an ongoing frame transfer, it points to the start address of this frame. After the last element of a buffer was transferred it will point to the buffer end address plus 0x4. The main purpose of the current full address registers for buffer A and buffer B (see next section) is to enable the software to find out the recently transferred element in the frozen buffer while the address of the active buffer increments. **Note:** A frame can be automatically stopped if any of the events listed in Conditions for Frame Transfer Interruption happens. If a frame is stopped before it could complete, then the current full address register is not updated and it will point to the start of the bad frame after the DCP was automatically disabled.

To transfer the first frame of buffer x, the information in the corresponding initial DCP RAM (IFADDRx, IHADDRCT, ITCOUNT) is loaded to an internal state machine but not to the current DCP RAM (CFADDRx, CFTCTx).

This is valid for all of the following modes:

- Buffer x has reached it's end in circular mode and rolls back to its start address.
- CP x is enabled by a CPENA access (and corresponding BIM bit is 0).
- A CPENA access or auto-switch mode causes a switch from CP y to CP x.

This means after starting the transfer to/from buffer x, CFADDRx and CFTCTx is not updated before the end of the first frame. So before the software switches from CP y to CP x using a write access to the CPENA register, it needs to initialize CFADDRx, CFTCTx. This allows the software to find out if the next request on CP x after the switching to CP x was delayed or never occurring.

24.5.6 Current Full Address B Register (HTU CFADDRB)

Figure 24-47. Current Full Address B Register (HTU CFADDRB)

31 … 16
CFADDRB
R/WP-X

15 … 0
CFADDRB
R/WP-X

LEGEND: R/W = Read/Write; WP = Write in privilege mode only; -*n* = value after reset; X = Unknown

Table 24-48. Current Full Address B Register (HTU CFADDRB) Field Descriptions

Bit	Field	Description
31-0	CFADDRB	Current (byte) Address of Buffer B The current main memory address register is updated at the end of each frame. Therefore it points to the start address of the frame, which is the next to transfer, if currently no frame is transferred on this DCP. If currently a frame is transferred, then it points to the start address of this frame. After the last element of a buffer was transferred it will point to the buffer end address plus 0x4. The main purpose of the current full address registers for buffer A and buffer B (see next section) is to enable the software to find out the recently transferred element in the frozen buffer while the address of the active buffer increments. **Note:** A frame can be automatically stopped if any of the events listed in Conditions for Frame Transfer Interruption happens. If a frame is stopped before it could complete, then the current full address register is not updated and it will point to the start of the bad frame after the DCP was automatically disabled.

To transfer the first frame of buffer x, the information in the corresponding initial DCP RAM (IFADDRx, IHADDRCT, ITCOUNT) is loaded to an internal state machine but not to the current DCP RAM (CFADDRx, CFTCTx).

This is valid for all of the following modes:

- Buffer x has reached it's end in circular mode and rolls back to its start address.
- CP x is enabled by a CPENA access (and corresponding BIM bit is 0).
- A CPENA access or auto-switch mode causes a switch from CP y to CP x.

This means after starting the transfer to/from buffer x, CFADDRx and CFTCTx is not updated before the end of the first frame. So before the software switches from CP y to CP x using a write access to the CPENA register, it needs to initialize CFADDRx, CFTCTx. This allows the software to find out if the next request on CP x after the switching to CP x was delayed or never occurring.

24.5.7 Current Frame Count Register (HTU CFCOUNT)

The current frame count register enables the software to find out the recent frame in the buffer while the counter of the active buffer decrements.

Figure 24-48. Current Frame Count Register (HTU CFCOUNT)

31 24	23 16
Reserved	CFTCTA
R-0	R/WP-X

15 8	7 0
Reserved	CFTCTB
R-0	R/WP-X

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privilege mode only; -*n* = value after reset; X = Unknown

Table 24-49. Current Frame Count Register (HTU CFCOUNT) Field Descriptions

Bit	Field	Value	Description
31-24	Reserved	0	Reads return 0. Writes have no effect.
23-16	CFTCTA		Current Frame Transfer Count for CP A. It is updated at the end of each frame.
15-8	Reserved	0	Reads return 0. Writes have no effect.
7-0	CFTCTB		Current Frame Transfer Count for CP B. It is updated at the end of each frame.

24.6 Examples

24.6.1 Application Examples for Setting the Transfer Modes of CP A and B of a DCP

Table 24-50. Application Examples for Setting the Transfer Modes of CP A and B of a DCP

CP A	CP B	
One shot	Not used	Buffer A can be used as a "one shot" buffer. A buffer full interrupt enabled for CP A can signal reaching the end of the buffer.
Auto switch	One shot	Can double the buffer size for a "one shot" buffer. A buffer full interrupt enabled for CP B can signal reaching the end of the buffer.
Circular	Circular	The CPU can switch the buffers at arbitrary times. It will fill or read the frozen buffer during the other buffer is filled or read by the HTU. Interrupts are not required for this case.
Auto switch	Auto switch	Buffer full interrupts (enabled for CP A and B) signal when the end of a buffer is reached. After one buffer is completed the according CPU interrupt routine will read or refill this buffer. At the same time the other buffer is read or filled by the HTU. Here the time when the buffer must be read is determined by the time of the interrupt (determined by the frequency of the N2HET transfer requests).

24.6.2 Software Example Sequence Assuming Circular Mode for Both CP A and B

The example assumes the N2HET address to be read and the main memory address to be written.

I1 CPU initializes initial DCP: IFADDRA, IFADDRB, IHADDRCT, ITCOUNT

I2 CPU clears current DCP: CFADDRA, CFADDRB, CFTCTA, CFTCTB

I3 CPU clears BFINTFL flag of CP A and B

I4 Enable CP A with the CPENA register. Now the HTU fills buffer A

After some time the CPU intends to read buffer A:

A1 CPU enables CP B and disables CP A by writing to the CPENA register. After this switch, the HTU fills buffer B. Filling buffer B starts with its initial full address and initial frame counter.

A2 CPU waits for CP A busy bit equals 0

A3 Optional: CPU verifies that the CP A request lost flag is not set. The bus error flag of CP A could also be checked.

A4 CPU reads the frozen CFTCTA, which indicates the fill level in the buffer

A5 CPU sets current CP A (CFTCTA and/or CFADDRA) to 0. This allows to find out if any request has happened during the next time buffer A is active.

A6 CPU reads BFINTFL flag of buffer A

A7 CPU clears the BFINTFL flag of buffer A. This is an initialization for the next time buffer A is used.

A8 CPU reads valid values of frozen buffer A. After reading the CPU does not need to clear the frozen buffer A.

After some time the CPU intends to read buffer B:

B1 CPU enables CP A and disables CP B by writing to the CPENA register. After this switch, the HTU fills buffer A. Filling buffer A starts with its initial full address and initial frame counter.

B2 CPU waits for CP B busy bit equals 0

B3 Optional: CPU verifies that the CP B request lost flag is not set. The bus error flag of CP B could also be checked.

B4 CPU reads the frozen CFTCTB, which indicates the fill level in the buffer

B5 CPU sets current CP B (CFTCTB and/or CFADDRB) to 0. This allows to find out if any request has happened during the next time buffer B is active.

B6 CPU reads BFINTFL flag of buffer B

B7 CPU clears the BFINTFL flag of buffer B. This is an initialization for the next time buffer B is used.

B8 CPU reads valid values of frozen buffer B. After reading the CPU does not need to clear the frozen buffer B.

After some time the CPU intends to read buffer A:

A1) ... see above...

NOTE: The buffer full interrupt doesn't need to be enabled. The BFINTFL flag is used to indicate a circular overrun of the buffer. If the BFINTFL flag is set, also the buffer section after the frozen full address could be read.

Steps A3 and B3 in the example sequence above imply that request lost interrupts are disabled. The example below assumes that request lost interrupts are enabled.

Request lost detection with interrupt enabled.

24.6.3 Example of an Interrupt Dispatch Flow for a Request Lost Interrupt

- A request lost occurs and the interrupt routine starts.
- Reading INTOFFx.INTYPEx shows that RLOSTFL is the interrupt source.
- Reading INTOFFx.CPOFFx = Ah shows that DCP 5 / CP A has caused the RLOSTFL interrupt. The hardware automatically clears bit (2·5+0) in RLOSTFL.
- Reading RLOSTFL= 84h shows that also another request lost event happened on DCP 1 / CP A [bit (2·1+0)] and on DCP 3 / CP B [bit (2·3+1)] at the same time or after the request lost occurred on DCP 5 / CP A.
- Writing back 84h to RLOSTFL clears bits 2 and 7 and the according pending interrupts.

Chapter 25

SPNU563A – March 2018

General-Purpose Input/Output (GIO) Module

This chapter describes the general-purpose input/output (GIO) module. The GIO module provides the family of devices with input/output (I/O) capability. The I/O pins are bidirectional and bit-programmable. The GIO module also supports external interrupt capability.

Topic **Page**

25.1 Overview

The GIO module offers general-purpose input and output capability. It supports up to eight 8-bit ports for a total of up to 64 GIO terminals. Each of these 64 terminals can be independently configured as input or output and configured as required by the application. The GIO module also supports generation of interrupts whenever a rising edge or falling edge or any toggle is detected on up to 32 of these GIO terminals. Refer to the device datasheet for identifying the number of GIO ports supported and the GIO terminals capable of generating an interrupt.

The main features of the GIO module are summarized as follows:

- Allows each GIO terminal to be configured for general-purpose input or output functions
- Supports programmable pull directions on each input GIO terminal
- Supports GIO output in push/pull or open-drain modes
- Allows up to 32 GIO terminals to be used for generating interrupt requests

25.2 Quick Start Guide

The GIO module comprises two separate components: an input/output (I/O) block and an interrupt generation block. Figure 25-1 and Figure 25-2 show what you should do after reset to configure the GIO module as I/O or for generating interrupts.

In GIO interrupt service routine, you shall read the GIO offset register (GIOOFF1 or GIOOFF2, depending on high-/low-level interrupt) to clear the flag and find the pending interrupt GIO channel.

Figure 25-1. I/O Function Quick Start Flow Chart

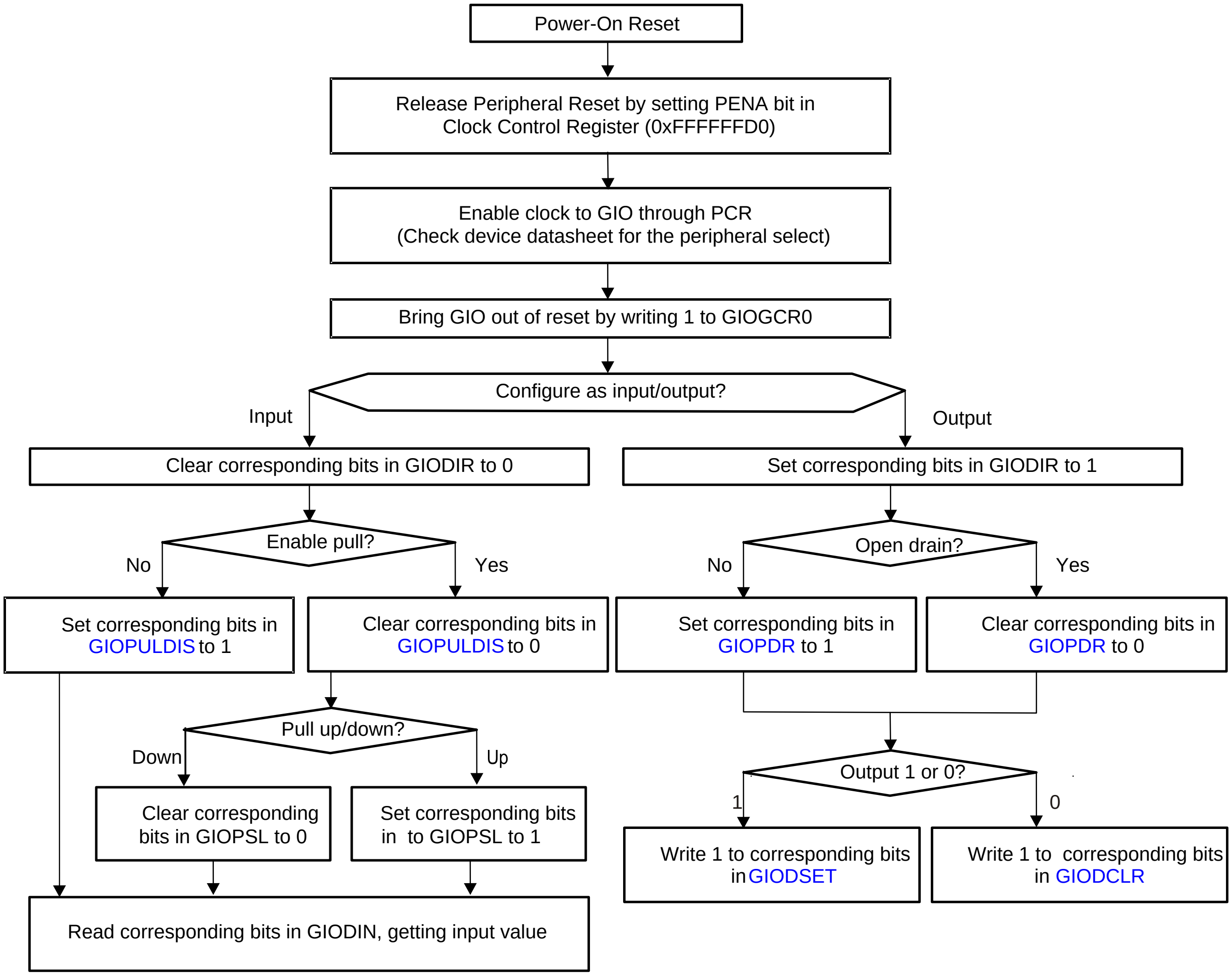
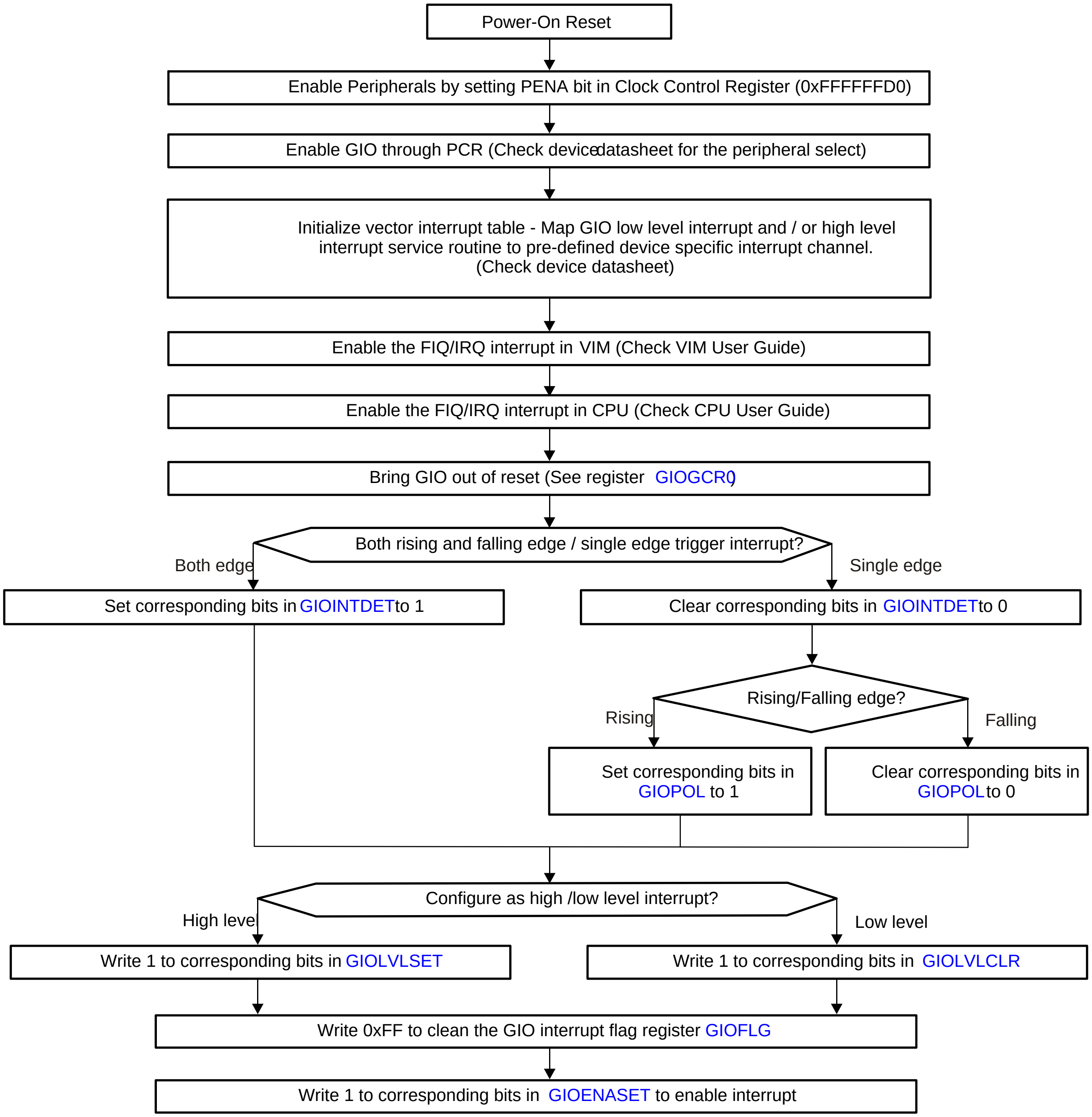

Power-On Reset
Release Peripheral Reset by setting PENA bit in Clock Control Register (0xFFFFFFD0)
Enable clock to GIO through PCR (Check device datasheet for the peripheral select)
Bring GIO out of reset by writing 1 to GIOGCR0
Configure as input/output?
Input
Output
Clear corresponding bits in GIODIR to 0
Set corresponding bits in GIODIR to 1
Enable pull?
No
Yes
Open drain?
No
Yes
Set corresponding bits in GIOPULDIS to 1
Clear corresponding bits in GIOPULDIS to 0
Set corresponding bits in GIOPDR to 1
Clear corresponding bits in GIOPDR to 0
Pull up/down?
Down
Up
Clear corresponding bits in GIOPSL to 0
Set corresponding bits in to GIOPSL to 1
Output 1 or 0?
1
0
Write 1 to corresponding bits in GIODSET
Write 1 to corresponding bits in GIODCLR
Read corresponding bits in GIODIN, getting input value

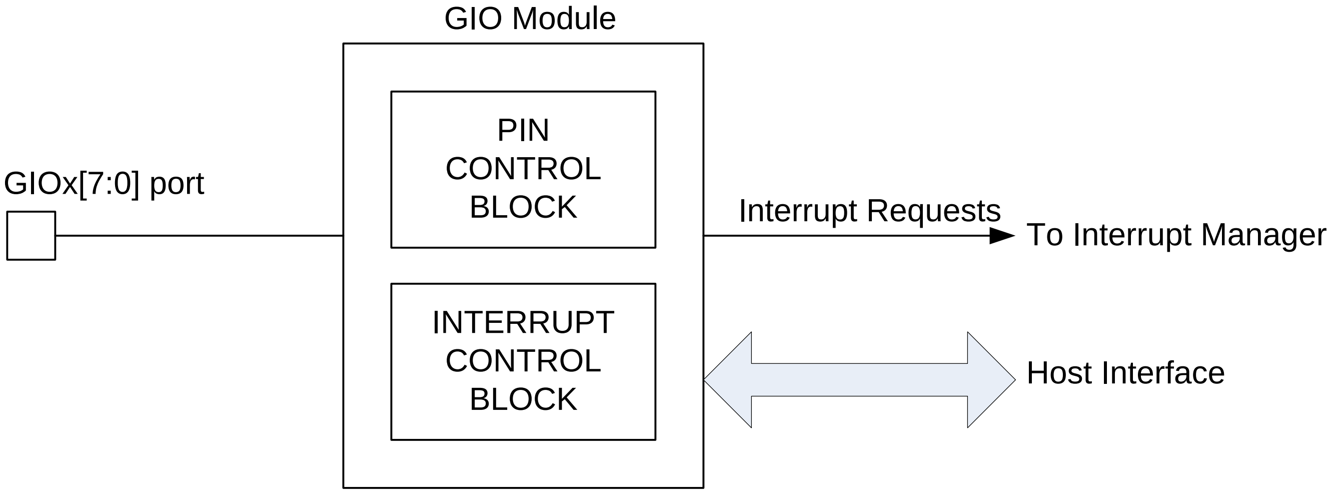
Figure 25-2. Interrupt Generation Function Quick Start Flow Chart


Power-On Reset
Enable Peripherals by setting PENA bit in Clock Control Register (0xFFFFFFD0)
Enable GIO through PCR (Check devicedatasheet for the peripheral select)
Initialize vector interrupt table - Map GIO low level interrupt and / or high level interrupt service routine to pre-defined device specific interrupt channel. (Check device datasheet)
Enable the FIQ/IRQ interrupt in VIM (Check VIM User Guide)
Enable the FIQ/IRQ interrupt in CPU (Check CPU User Guide)
Bring GIO out of reset (See register GIOGCR0)
Both rising and falling edge / single edge trigger interrupt?
Both edge
Single edge
Set corresponding bits in GIOINTDETto 1
Clear corresponding bits in GIOINTDETto 0
Rising/Falling edge?
Rising
Falling
Set corresponding bits in GIOPOL to 1
Clear corresponding bits in GIOPOLto 0
Configure as high /low level interrupt?
High level
Low level
Write 1 to corresponding bits in GIOLVLSET
Write 1 to corresponding bits in GIOLVLCLR
Write 0xFF to clean the GIO interrupt flag register GIOFLG
Write 1 to corresponding bits in GIOENASET to enable interrupt

25.3 Functional Description of GIO Module

As shown in Figure 25-3, the GIO module comprises of two separate components: an input/output (I/O) block and an interrupt block.

Figure 25-3. GIO Module Diagram



25.3.1 I/O Functions

The I/O block allows each GIO terminal to be configured for use as a general-purpose input or output in the application. The GIO module supports multiple registers to control the various aspects of the input and output functions. These are described as follows.

- Data direction (GIODIR)

  Configures GIO terminal(s) as input (default) or output through the GIODIRx registers.

- Data input (GIODIN)

  Reflects the logic level on GIO terminals in the GIODINx registers. A high voltage (V_{IH} or greater) applied to the pin causes a high value (1) in the data input register (GIODIN[7:0]). When a low voltage (V_{IL} or less) is applied to the pin, the data input register reads a low value (0). The V_{IH} and V_{IL} values are device specific and can be found in the device datasheet.

- Data output (GIODOUT)

  Configures the logic level to be output on GIO terminal(s) configured as outputs. A low value (0) written to the data output register forces the pin to a low output voltage (V_{OL} or lower). A high value (1) written to the data output register (GIODOUTx) forces the pin to a high output voltage (V_{OH} or higher) if the open drain functionality is disabled (GIOPDRx[7:0]). If open drain functionality is enabled, a high value (1) written to the data output register forces the pin to a high-impedance state (Z).

- Data set (GIODSET)

  Allows logic HIGH to be output on GIO terminal(s) configured as outputs by writing 1's to the required bits in the GIODSETx registers. If open drain functionality is enabled, a high value (1) written to the data output register forces the pin to a high-impedance state (Z). The GIODSETx registers eliminate the need for the application to perform a read-modify-write operation when it needs to set one or more GIO pin(s).

- Data clear (GIODCLR)

  Allows logic LOW to be output on GIO terminal(s) configured as outputs by writing 1s to the required bits in the GIODCLRx registers. The GIODCLRx registers eliminate the need for the application to perform a read-modify-write operation when it needs to clear one or more GIO pin(s).

- Open drain (GIOPDR)

  Open drain functionality is enabled or disabled (default) using the open drain register GIOPDR[7:0] register. If open-drain mode output is enabled on a pin, a high value (1) written to the data output register (GIODOUTx[7:0]) forces the pin to a high impedance state (Z).

- Pull disable (GIOPULDIS)

  Disables the internal pull on GIO terminal(s) configured as inputs by writing to the GIOPULDISx registers.
- Pull select (GIOPSL)

  Selects internal pull down (default) or pull up on GIO terminal(s) configured as inputs by writing to the GIOPULSELx registers.

Refer to the specific device's datasheet to identify the number of GIO ports as well as the input and output functions supported. Some devices may not support the programmable pull controls. In that case, the pull disable and the pull select register controls will not work.

25.3.2 Interrupt Function

The GIO module supports up to 32 terminals to be configured for generating an interrupt to the host processor through the Vectored Interrupt Manager (VIM). The main functions of the interrupt block are:

- Select the GIO pin(s) that is/are used to generate interrupt(s)

  This is done via the interrupt enable set and clear registers, GIOENASET and GIOENACLR.
- Select the edge on the selected GIO pin(s) that is/are used to generate interrupt(s): rising/falling/both

  Rising or falling edge can be selected via the GIOPOL register. If interrupt is required to be generated on both rising and falling edges, this can be configured via the GIOINTDET register.
- Select the interrupt priority

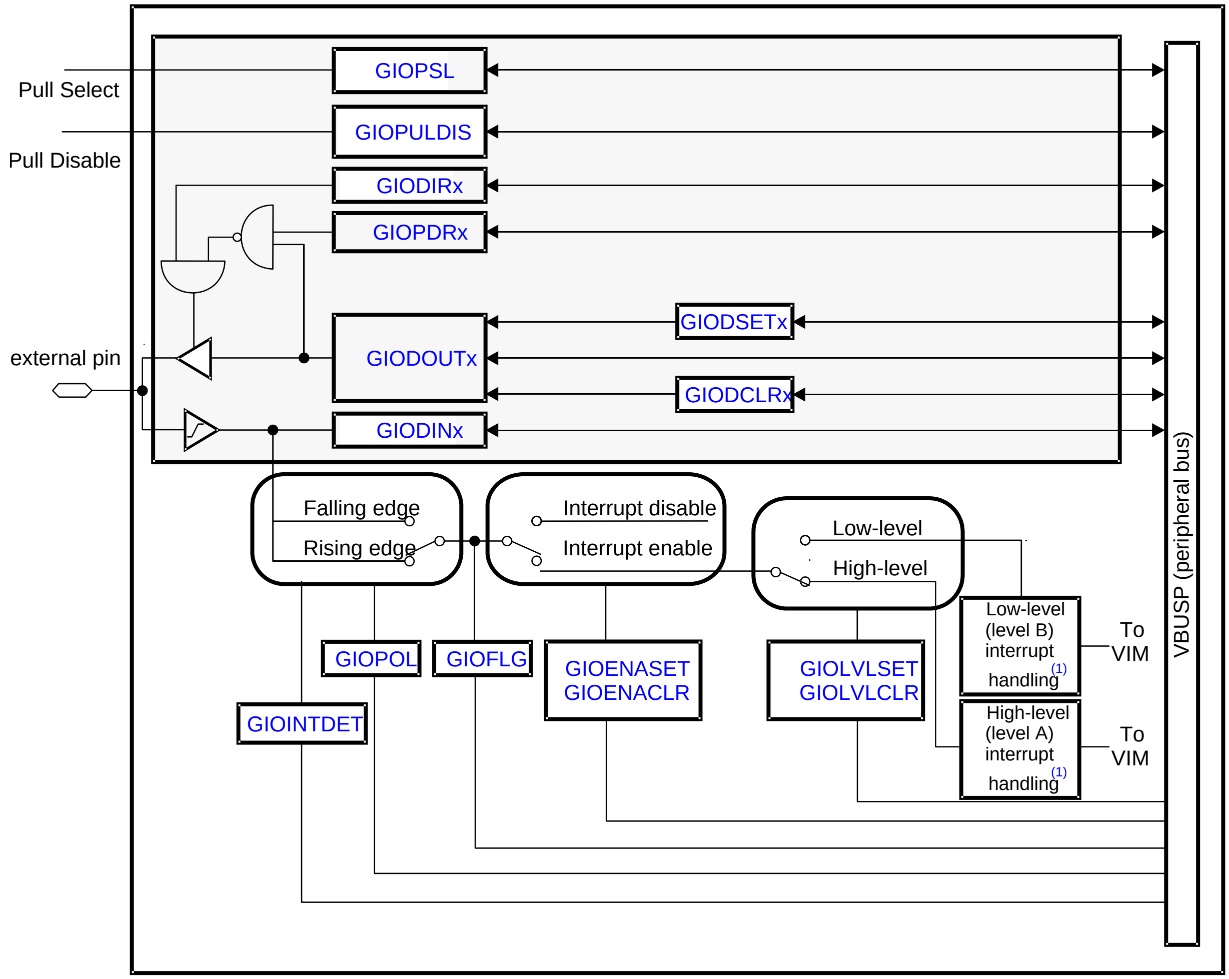
  Low- or high-level interrupt can be selected through the GIOLVLSET and GIOLVLCLR registers.
- Individual interrupt flags are set in the GIOFLG register

The terminals on GIO ports A through D are all interrupt-capable and can be used to handle either general I/O functions or interrupt requests. Each interrupt request can be connected to the VIM at one of two different levels – High (or A) and Low (or B), depending on the VIM channel number. The VIM has an inherent priority scheme so that a request on a lower number channel has a higher priority than a request on a higher number channel. Refer the device datasheet to identify the VIM channel numbers for the GIO level A and level B interrupt requests. Also note that the interrupt priority of level A and level B interrupt handling blocks can be re-programmed in the VIM.

25.3.3 GIO Block Diagram

The GIO block diagram (Figure 25-4) represents the flow of information through a pin. The shaded area corresponds to the I/O block; the unshaded area corresponds to the interrupt block.

Figure 25-4. GIO Block Diagram



(1) A single low-level-interrupt-handling block and a single high-level-interrupt-handling block service all of the interrupt-capable external pins, but only one pin can be serviced by an interrupt block at a time.

25.4 Device Modes of Operation

The GIO module behaves differently in different modes of operation. There are two main modes:

- Emulation mode
- Power-down mode (low-power mode)

25.4.1 Emulation Mode

Emulation mode is used by debugger tools to stop the CPU at breakpoints to read registers.

> **NOTE: Emulation Mode and Emulation Registers**
>
> Emulation mode is a mode of operation of the device and is separate from the GIO emulation registers (GIOEMU1 and GIOEMU2). The contents of these emulation registers are identical to the contents of GIO offset registers (GIOOFF1 and GIOOFF2). Both emulation registers and GIO offset registers are NOT cleared when they are read in emulation mode. GIO offset registers are cleared when they are read in normal mode (other than emulation mode). The emulation registers are NOT cleared when they are read in normal mode. The intention for the emulation registers is that software can use them without clearing the flags.

During emulation mode:

- External interrupts are not captured because the VIM is unable to service interrupts.
- Any register can be read without affecting the state of the system.
- A write to a register still does affect the state of the system.

25.4.2 Power-Down Mode (Low-Power Mode)

In power-down mode, the clock signal to the GIO module is disabled. Thus, there is no switching and the only current draw comes from leakage current. In power-down mode, interrupt pins become level-sensitive rather than edge-sensitive. The polarity bit changes function from falling-edge-triggered to low-level-triggered and rising-edge-triggered to high-level-triggered. A corresponding level on an interrupt pin pulls the module out of low-power mode, if the interrupt is also enabled to wake up the device out of a low-power mode.

25.4.2.1 Module-Level Power Down

The GIO module can be placed into a power down state by disabling the GIO peripheral module via the appropriate bit in the peripheral power down register. Please refer to the Peripheral Central Resource Registers (Section 2.5.3) for details.

25.4.2.2 Device-Level Power Down

The entire device can be placed in one of the pre-defined low-power modes: doze, snooze, or sleep using the clock source and clock domain disable registers in the system module.

25.5 GIO Control Registers

Table 25-1 shows the summary of the GIO registers. The registers are accessible in 8-, 16-, and 32-bit reads or writes.

The start address for the GIO module is FFF7 BC00h.

The GIO module supports up to 8 ports. Refer to your device-specific data manual to identify the actual number of GIO ports and the number of pins in each GIO port implemented on this device.

The GIO module supports up to 4 interrupt-capable ports. Refer to the device datasheet to identify the actual number of interrupt-capable GIO ports and the number of pins in each GIO port implemented on this device.

Table 25-1. GIO Control Registers

Offset	Acronym	Register Description	Section
00h	GIOGCR0	GIO Global Control Register	Section 25.5.1
08h	GIOINTDET	GIO Interrupt Detect Register	Section 25.5.2
0Ch	GIOPOL	GIO Interrupt Polarity Register	Section 25.5.3
10h	GIOENASET	GIO Interrupt Enable Set Register	Section 25.5.4.1
14h	GIOENACLR	GIO Interrupt Enable Clear Register	Section 25.5.4.2
18h	GIOLVLSET	GIO Interrupt Priority Set Register	Section 25.5.5.1
1Ch	GIOLVLCLR	GIO Interrupt Priority Clear Register	Section 25.5.5.2
20h	GIOFLG	GIO Interrupt Flag Register	Section 25.5.6
24h	GIOOFF1	GIO Offset 1 Register	Section 25.5.7
28h	GIOOFF2	GIO Offset 2 Register	Section 25.5.8
2Ch	GIOEMU1	GIO Emulation 1 Register	Section 25.5.9
30h	GIOEMU2	GIO Emulation 2 Register	Section 25.5.10
34h	GIODIRA	GIO Data Direction Register	Section 25.5.11
38h	GIODINA	GIO Data Input Register	Section 25.5.12
3Ch	GIODOUTA	GIO Data Output Register	Section 25.5.13
40h	GIODSETA	GIO Data Set Register	Section 25.5.14
44h	GIODCLRA	GIO Data Clear Register	Section 25.5.15
48h	GIOPDRA	GIO Open Drain Register	Section 25.5.16
4Ch	GIOPULDISA	GIO Pull Disable Register	Section 25.5.17
50h	GIOPSLA	GIO Pull Select Register	Section 25.5.18
54h	GIODIRB	GIO Data Direction Register	Section 25.5.11
58h	GIODINB	GIO Data Input Register	Section 25.5.12
5Ch	GIODOUTB	GIO Data Output Register	Section 25.5.13
60h	GIODSETB	GIO Data Set Register	Section 25.5.14
64h	GIODCLRB	GIO Data Clear Register	Section 25.5.15
68h	GIOPDRB	GIO Open Drain Register	Section 25.5.16
6Ch	GIOPULDISB	GIO Pull Disable Register	Section 25.5.17
70h	GIOPSLB	GIO Pull Select Register	Section 25.5.18

25.5.1 GIO Global Control Register (GIOGCR0)

The GIOGCR0 register contains one bit that controls the module reset status. Writing a 0 to this bit puts the module in a reset state. After system reset, this bit must be set to 1 before configuring any other register of the GIO module. Figure 25-5 and Table 25-2 describe this register.

Figure 25-5. GIO Global Control Register (GIOGCR0) [offset = 00h]

31–16
Reserved
R-0

15–1	0
Reserved	RESET
R-0	R/WP-0

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privileged mode only; -*n* = value after reset

Table 25-2. GIO Global Control Register (GIOGCR0) Field Descriptions

Bit	Field	Value	Description
31-1	Reserved	0	Reads return 0. Writes have no effect.
0	RESET		GIO reset.
		0	The GIO is in reset state.
		1	The GIO is operating normally.

NOTE: Note that putting the GIO module in reset state is not the same as putting it in a low-power state.

25.5.2 GIO Interrupt Detect Register (GIOINTDET)

The GIO module supports generation of an interrupt request to CPU when a rising edge, falling edge, or both edges is detected on one or more GIO pin(s). The GIOINTDET register allows both rising and falling edges to be detected, while the GIOPOL register allows the application to define whether a rising edge or a falling edge is to be detected. Figure 25-6 and Table 25-3 describe this register.

Figure 25-6. GIO Interrupt Detect Register (GIOINTDET) [offset = 08h]

31 – 24	23 – 16
GIOINTDET 3	GIOINTDET 2
R/W-0	R/W-0

15 – 8	7 – 0
GIOINTDET 1	GIOINTDET 0
R/W-0	R/W-0

LEGEND: R/W = Read/Write; -*n* = value after reset

Table 25-3. GIO Interrupt Detect Register (GIOINTDET) Field Descriptions

Bit	Field	Value	Description
31-24	GIOINTDET 3		Interrupt detection select for pins GIOD[7:0]
		0	The flag sets on either a falling or a rising edge on the corresponding pin, depending on the polarity setup in the polarity register (GIOPOL).
		1	The flag sets on both the rising and falling edges on the corresponding pin.
23-16	GIOINTDET 2		Interrupt detection select for pins GIOC[7:0]
		0	The flag sets on either a falling or a rising edge on the corresponding pin, depending on the polarity setup in the polarity register (GIOPOL).
		1	The flag sets on both the rising and falling edges on the corresponding pin.
15-8	GIOINTDET 1		Interrupt detection select for pins GIOB[7:0]
		0	The flag sets on either a falling or a rising edge on the corresponding pin, depending on the polarity setup in the polarity register (GIOPOL).
		1	The flag sets on both the rising and falling edges on the corresponding pin.
7-0	GIOINTDET 0		Interrupt detection select for pins GIOA[7:0]
		0	The flag sets on either a falling or a rising edge on the corresponding pin, depending on the polarity setup in the polarity register (GIOPOL).
		1	The flag sets on both the rising and falling edges on the corresponding pin.

25.5.3 GIO Interrupt Polarity Register (GIOPOL)

The GIOPOL register configures the polarity of the edge, rising edge or falling edge, that needs to be detected. When the device is in low-power mode, the GIOPOL register controls the ***level***, high or low, which will be detected by the GIO module. Figure 25-7 and Table 25-4 describe this register.

Figure 25-7. GIO Interrupt Polarity Register (GIOPOL) [offset = 0Ch]

31 24	23 16
GIOPOL 3	GIOPOL 2
R/W-0	R/W-0

15 8	7 0
GIOPOL 1	GIOPOL 0
R/W-0	R/W-0

LEGEND: R/W = Read/Write; -*n* = value after reset

Table 25-4. GIO Interrupt Polarity Register (GIOPOL) Field Descriptions

Bit	Field	Value	Description
31-24	GIOPOL 3		Interrupt polarity select for pins GIOD[7:0]
			Normal operation (user or privileged mode):
		0	The flag is set on the falling edge on the corresponding pin.
		1	The flag is set on the rising edge on the corresponding pin.
			Low-power mode (GIO module clocks off):
		0	The interrupt is triggered on the low level.
		1	The interrupt is triggered on the high level.
23-16	GIOPOL 2		Interrupt polarity select for pins GIOC[7:0]
			Normal operation (user or privileged mode):
		0	The flag is set on the falling edge on the corresponding pin.
		1	The flag is set on the rising edge on the corresponding pin.
			Low-power mode (GIO module clocks off):
		0	The interrupt is triggered on the low level.
		1	The interrupt is triggered on the high level.
15-8	GIOPOL 1		Interrupt polarity select for pins GIOB[7:0]
			Normal operation (user or privileged mode):
		0	The flag is set on the falling edge on the corresponding pin.
		1	The flag is set on the rising edge on the corresponding pin.
			Low-power mode (GIO module clocks off):
		0	The interrupt is triggered on the low level.
		1	The interrupt is triggered on the high level.
7-0	GIOPOL 0		Interrupt polarity select for pins GIOA[7:0]
			Normal operation (user or privileged mode):
		0	The flag is set on the falling edge on the corresponding pin.
		1	The flag is set on the rising edge on the corresponding pin.
			Low-power mode (GIO module clocks off):
		0	The interrupt is triggered on the low level.
		1	The interrupt is triggered on the high level.

25.5.4 GIO Interrupt Enable Registers (GIOENASET and GIOENACLR)

The GIOENASET and GIOENACLR registers control which interrupt-capable pins are actually configured as interrupts. If the interrupt is enabled, the rising edge, falling edge, or both edges on the selected pin lead to an interrupt request.

25.5.4.1 GIOENASET Register

Figure 25-8 and Table 25-5 describe this register.

NOTE: Enabling Interrupt at the Device Level

The interrupt channel in the Vectored Interrupt Manager (VIM) must be enabled for the interrupt request to be forwarded to the CPU. Additionally, the ARM CPU (CPSR bit 7 or 6) must be cleared to respond to interrupt requests (IRQ/FIQ).

Figure 25-8. GIO Interrupt Enable Set Register (GIOENASET) [offset = 10h]

31 24	23 16
GIOENASET 3	GIOENASET 2
R/W-0	R/W-0

15 8	7 0
GIOENASET 1	GIOENASET 0
R/W-0	R/W-0

LEGEND: R/W = Read/Write; -*n* = value after reset

Table 25-5. GIO Interrupt Enable Set Register (GIOENASET) Field Descriptions

Bit	Field	Value	Description
31-24	GIOENASET 3		Interrupt enable for pins GIOD[7:0]
		0	Read: The interrupt is disabled. Write: Writing a 0 to this bit has no effect.
		1	Read: The interrupt is enabled. Write: Enables the interrupt.
23-16	GIOENASET 2		Interrupt enable for pins GIOC[7:0]
		0	Read: The interrupt is disabled. Write: Writing a 0 to this bit has no effect.
		1	Read: The interrupt is enabled. Write: Enables the interrupt.
15-8	GIOENASET 1		Interrupt enable for pins GIOB[7:0]
		0	Read: The interrupt is disabled. Write: Writing a 0 to this bit has no effect.
		1	Read: The interrupt is enabled. Write: Enables the interrupt.
7-0	GIOENASET 0		Interrupt enable for pins GIOA[7:0]
		0	Read: The interrupt is disabled. Write: Writing a 0 to this bit has no effect.
		1	Read: The interrupt is enabled. Write: Enables the interrupt.

25.5.4.2 GIOENACLR Register

This register disables the interrupt. Figure 25-9 and Table 25-6 describe this register.

Figure 25-9. GIO Interrupt Enable Clear Register (GIOENACLR) [offset = 14h]

31 24	23 16
GIOENACLR 3	GIOENACLR 2
R/W-0	R/W-0

15 8	7 0
GIOENACLR 1	GIOENACLR 0
R/W-0	R/W-0

LEGEND: R/W = Read/Write; -*n* = value after reset

Table 25-6. GIO Interrupt Enable Clear Register (GIOENACLR) Field Descriptions

Bit	Field	Value	Description
31-24	GIOENACLR 3		Interrupt disable for pins GIOD[7:0]
		0	Read: The interrupt is disabled. Write: Writing a 0 to this bit has no effect.
		1	Read: The interrupt is enabled. Write: Disables the interrupt.
23-16	GIOENACLR 2		Interrupt disable for pins GIOC[7:0]
		0	Read: The interrupt is disabled. Write: Writing a 0 to this bit has no effect.
		1	Read: The interrupt is enabled. Write: Disables the interrupt.
15-8	GIOENACLR 1		Interrupt disable for pins GIOB[7:0]
		0	Read: The interrupt is disabled. Write: Writing a 0 to this bit has no effect.
		1	Read: The interrupt is enabled. Write: Disables the interrupt.
7-0	GIOENACLR 0		Interrupt disable for pins GIOA[7:0]
		0	Read: The interrupt is disabled. Write: Writing a 0 to this bit has no effect.
		1	Read: The interrupt is enabled. Write: Disables the interrupt.

25.5.5 GIO Interrupt Priority Registers (GIOLVLSET and GIOLVLCLR)

The GIOLVLSET and GIOLVLCLR registers configure the interrupts as high-level (level A) or low-level (level B) going to the Vectored Interrupt Manager (VIM). Each interrupt is individually configured.

- The high-level interrupts are recorded to GIOOFF1 and GIOEMU1.
- The low-level interrupts are recorded to GIOOFF2 and GIOEMU2.

NOTE: The GIO module can generate two interrupt requests. These are connected to two separate channels on the Vectored Interrupt Manager (VIM). The lower-numbered VIM channels are higher priority. The GIO interrupt connected to a lower-number channel is the high-level (also called level A) GIO interrupt, while the GIO interrupt connected to a higher-number channel is the low-level (also called level B) GIO interrupt.

25.5.5.1 GIOLVLSET Register

The GIOLVLSET register is used to configure an interrupt as a high-level interrupt going to the VIM. An interrupt can be configured as a high-level interrupt by writing a 1 into the corresponding bit of the GIOLVLSET register. Writing a 0 has no effect. Figure 25-10 and Table 25-7 describe this register.

Figure 25-10. GIO Interrupt Priority Register (GIOLVLSET) [offset = 18h]

31			16
GIOLVLSET 3		GIOLVLSET 2	
R/W-0		R/W-0	

15	8	7	0
GIOLVLSET 1		GIOLVLSET 0	
R/W-0		R/W-0	

LEGEND: R/W = Read/Write; -*n* = value after reset

Table 25-7. GIO Interrupt Priority Register (GIOLVLSET) Field Descriptions

Bit	Field	Value	Description
31-24	GIOLVLSET 3		GIO high-priority interrupt for pins GIOD[7:0].
		0	Read: The interrupt is a low-level interrupt. The low-level interrupts are recorded to GIOOFF2 and GIOEMU2. Write: Writing a 0 to this bit has no effect.
		1	Read: The interrupt is set as a high-level interrupt. The high-level interrupts are recorded to GIOOFF1 and GIOEMU1. Write: Sets the interrupt as a high-level interrupt. The high-level interrupts are recorded to GIOOFF1 and GIOEMU1.
23-16	GIOLVLSET 2		GIO high-priority interrupt for pins GIOC[7:0].
		0	Read: The interrupt is a low-level interrupt. The low-level interrupts are recorded to GIOOFF2 and GIOEMU2. Write: Writing a 0 to this bit has no effect.
		1	Read: The interrupt is set as a high-level interrupt. The high-level interrupts are recorded to GIOOFF1 and GIOEMU1. Write: Sets the interrupt as a high-level interrupt. The high-level interrupts are recorded to GIOOFF1 and GIOEMU1.
15-8	GIOLVLSET 1		GIO high-priority interrupt for pins GIOB[7:0].
		0	Read: The interrupt is a low-level interrupt. The low-level interrupts are recorded to GIOOFF2 and GIOEMU2. Write: Writing a 0 to this bit has no effect.
		1	Read: The interrupt is set as a high-level interrupt. The high-level interrupts are recorded to GIOOFF1 and GIOEMU1. Write: Sets the interrupt as a high-level interrupt. The high-level interrupts are recorded to GIOOFF1 and GIOEMU1.

Table 25-7. GIO Interrupt Priority Register (GIOLVLSET) Field Descriptions (continued)

Bit	Field	Value	Description
7-0	GIOLVLSET 0		GIO high-priority interrupt for pins GIOA[7:0].
		0	Read: The interrupt is a low-level interrupt. The low-level interrupts are recorded to GIOOFF2 and GIOEMU2. Write: Writing a 0 to this bit has no effect.
		1	Read: The interrupt is set as a high-level interrupt. The high-level interrupts are recorded to GIOOFF1 and GIOEMU1. Write: Sets the interrupt as a high-level interrupt. The high-level interrupts are recorded to GIOOFF1 and GIOEMU1.

25.5.5.2 GIOLVLCLR Register

The GIOLVLCLR register is used to configure an interrupt as a low-level interrupt going to the VIM. An interrupt can be configured as a low-level interrupt by writing a 1 into the corresponding bit of the GIOLVLCLR register. Writing a 0 has no effect. Figure 25-11 and Table 25-8 describe this register.

Figure 25-11. GIO Interrupt Priority Register (GIOLVLCLR) [offset = 1Ch]

31	16
GIOLVLCLR 3	GIOLVLCLR 2
R/W-0	R/W-0

15 8	7 0
GIOLVLCLR 1	GIOLVLCLR 0
R/W-0	R/W-0

LEGEND: R/W = Read/Write; -*n* = value after reset

Table 25-8. GIO Interrupt Priority Register (GIOLVLCLR) Field Descriptions

Bit	Field	Value	Description
31-24	GIOLVLCLR 3		GIO low-priority interrupt for pins GIOD[7:0]
		0	Read: The interrupt is a low-level interrupt. Write: Writing a 0 to this bit has no effect.
		1	Read: The interrupt is set as a high-level interrupt. The high-level interrupts are recorded to GIOOFF1 and GIOEMU1. Write: Sets the interrupt as a low-level interrupt. The low-level interrupts are recorded to GIOOFF2 and GIOEMU2.
23-16	GIOLVLCLR 2		GIO low-priority interrupt for pins GIOC[7:0]
		0	Read: The interrupt is a low-level interrupt. Write: Writing a 0 to this bit has no effect.
		1	Read: The interrupt is set as a high-level interrupt. The high-level interrupts are recorded to GIOOFF1 and GIOEMU1. Write: Sets the interrupt as a low-level interrupt. The low-level interrupts are recorded to GIOOFF2 and GIOEMU2.
15-8	GIOLVLCLR 1		GIO low-priority interrupt for pins GIOB[7:0]
		0	Read: The interrupt is a low-level interrupt. Write: Writing a 0 to this bit has no effect.
		1	Read: The interrupt is set as a high-level interrupt. The high-level interrupts are recorded to GIOOFF1 and GIOEMU1. Write: Sets the interrupt as a low-level interrupt. The low-level interrupts are recorded to GIOOFF2 and GIOEMU2.
7-0	GIOLVLCLR 0		GIO low-priority interrupt for pins GIOA[7:0]
		0	Read: The interrupt is a low-level interrupt. Write: Writing a 0 to this bit has no effect.
		1	Read: The interrupt is set as a high-level interrupt. The high-level interrupts are recorded to GIOOFF1 and GIOEMU1. Write: Sets the interrupt as a low-level interrupt. The low-level interrupts are recorded to GIOOFF2 and GIOEMU2.

25.5.6 GIO Interrupt Flag Register (GIOFLG)

The GIOFLG register contains flags indicating that the transition edge (as set in GIOINTDET and GIOPOL registers) has occurred. The flag can be cleared by the CPU writing a 1 to the flag that is set. The flag is also cleared by reading the appropriate interrupt offset register (GIOOFF1 or GIOOFF2). Figure 25-12 and Table 25-9 describe this register.

Figure 25-12. GIO Interrupt Flag Register (GIOFLG) [offset = 20h]

31 – 24	23 – 16
GIOFLG 3	GIOFLG 2
R/W1C-0	R/W1C-0

15 – 8	7 – 0
GIOFLG 1	GIOFLG 0
R/W1C-0	R/W1C-0

LEGEND: R/W = Read/Write; W1C = Write 1 to clear; -*n* = value after reset

Table 25-9. GIO Interrupt Flag Register (GIOFLG) Field Descriptions

Bit	Field	Value	Description
31-24	GIOFLG 3		GIO flag for pins GIOD[7:0]
		0	Read: A transition has not occurred since the last clear. Write: Writing a 0 to this bit has no effect.
		1	Read: The selected transition on the corresponding pin has occurred. Write: The corresponding bit is cleared to 0. **Note: This bit is also cleared by a read to the corresponding bit in the appropriate offset register.**
23-16	GIOFLG 2		GIO flag for pins GIOC[7:0]
		0	Read: A transition has not occurred since the last clear. Write: Writing a 0 to this bit has no effect.
		1	Read: The selected transition on the corresponding pin has occurred. Write: The corresponding bit is cleared to 0. **Note: This bit is also cleared by a read to the corresponding bit in the appropriate offset register.**
15-8	GIOFLG 1		GIO flag for pins GIOB[7:0]
		0	Read: A transition has not occurred since the last clear. Write: Writing a 0 to this bit has no effect.
		1	Read: The selected transition on the corresponding pin has occurred. Write: The corresponding bit is cleared to 0. **Note: This bit is also cleared by a read to the corresponding bit in the appropriate offset register.**
7-0	GIOFLG 0		GIO flag for pins GIOA[7:0]
		0	Read: A transition has not occurred since the last clear. Write: Writing a 0 to this bit has no effect.
		1	Read: The selected transition on the corresponding pin has occurred. Write: The corresponding bit is cleared to 0. **Note: This bit is also cleared by a read to the corresponding bit in the appropriate offset register.**

NOTE: An interrupt flag gets set when the selected transition happens on the corresponding GIO pin ***regardless of whether the interrupt generation is enabled or not***. It is recommended to clear a flag before enabling the interrupt generation for a transition on the corresponding GIO pin.

25.5.7 GIO Offset Register 1 (GIOOFF1)

The GIOOFF1 register provides a numerical offset value that represents the pending external interrupt with high priority. The offset value can be used to locate the position of the interrupt routine in a vector table in application software. Figure 25-13 and Table 25-10 describe this register.

NOTE: Reading this register clears it, GIOEMU1 and the corresponding flag bit in the GIOFLG register. However, in emulation mode, a read to this register does not clear any register or flag. If more than one GIO interrupts are pending, then reading the GIOOFF1 register will change the contents of GIOOFF1 and GIOEMU1 to show the offset value for the next highest-priority pending interrupt. The application can choose to service all GIO interrupts from the same service routine by continuing to read the GIOOFF1 register until it reads zeros.

Figure 25-13. GIO Offset 1 Register (GIOOFF1) [offset = 24h]

31–16
Reserved
R-0

15–6	5–0
Reserved	GIOOFF1
R-0	R-0

LEGEND: R = Read only; -*n* = value after reset

Table 25-10. GIO Offset 1 Register (GIOOFF1) Field Descriptions

Bit	Field	Value	Description
31-6	Reserved	0	Reads return 0. Writes have no effect.
5-0	GIOOFF1		GIO offset 1. These bits index the currently pending high-priority interrupt. This register and the flag bit (in the GIOFLG register) are also cleared when this register is read, except in emulation mode.
		0	No interrupt is pending.
		1h	Interrupt 0 (corresponding to GIOA0) is pending with a high priority.
		:	:
		8h	Interrupt 7 (corresponding to GIOA7) is pending with a high priority.
		9h	Interrupt 8 (corresponding to GIOB0) is pending with a high priority.
		:	:
		10h	Interrupt 16 (corresponding to GIOB7) is pending with a high priority.
		:	:
		20h	Interrupt 32 (corresponding to GIOD7) is pending with a high priority.
		21h-3Fh	Reserved

25.5.8 GIO Offset B Register (GIOOFF2)

The GIOOFF2 register provides a numerical offset value that represents the pending external interrupt with low priority. The offset value can be used to locate the position of the interrupt routine in a vector table in application software. Figure 25-14 and Table 25-11 describe this register.

NOTE: Reading this register clears it, GIOEMU2 and the corresponding flag bit in the GIOFLG register. However, in emulation mode, a read to this register does not clear any register or flag. If more than one GIO interrupts are pending, then reading the GIOOFF1 register will change the contents of GIOOFF2 and GIOEMU2 to show the offset value for the next highest-priority pending interrupt. The application can choose to service all GIO interrupts from the same service routine by continuing to read the GIOOFF1 register until it reads zeros.

Figure 25-14. GIO Offset 2 Register (GIOOFF2) [offset = 28h]

31–16
Reserved
R-0

15–6	5–0
Reserved	GIOOFF2
R-0	R-0

LEGEND: R = Read only; -*n* = value after reset

Table 25-11. GIO Offset 2 Register (GIOOFF2) Field Descriptions

Bit	Field	Value	Description
31-6	Reserved	0	Reads return 0. Writes have no effect.
5-0	GIOOFF2		GIO offset 2. These bits index the currently pending low-priority interrupt. This register and the flag bit (in the GIOFLG register) are also cleared when this register is read, except in emulation mode.
		0	No interrupt is pending.
		1h	Interrupt 0 (corresponding to GIOA0) is pending with a low priority.
		:	:
		8h	Interrupt 7 (corresponding to GIOA7) is pending with a low priority.
		9h	Interrupt 8 (corresponding to GIOB0) is pending with a low priority.
		:	:
		10h	Interrupt 16 (corresponding to GIOB7) is pending with a low priority.
		:	:
		20h	Interrupt 32 (corresponding to GIOD7) is pending with a low priority.
		21h-3Fh	Reserved

25.5.9 GIO Emulation A Register (GIOEMU1)

The GIOEMU1 register is a read-only register. The contents of this register are identical to the contents of GIOOFF1. The intention for the this register is that software can use it without clearing the flags. Figure 25-15 and Table 25-12 describe this register.

NOTE: The corresponding flag in the GIOFLG register is not cleared when the GIOEMU1 register is read.

Figure 25-15. GIO Emulation 1 Register (GIOEMU1) [offset = 2Ch]

31	16
Reserved	
R-0	

15	6	5	0
Reserved		GIOEMU1	
R-0		R-0	

LEGEND: R = Read only; -*n* = value after reset

Table 25-12. GIO Emulation 1 Register (GIOEMU1) Field Descriptions

Bit	Field	Value	Description
31-6	Reserved	0	Reads return 0. Writes have no effect.
5-0	GIOEMU1		GIO offset emulation 1. These bits index the currently pending high-priority interrupt. No register or flag is cleared by reading this register.
		0	No interrupt is pending.
		1h	Interrupt 0 (corresponding to GIOA0) is pending with a high priority.
		:	:
		8h	Interrupt 7 (corresponding to GIOA7) is pending with a high priority.
		9h	Interrupt 8 (corresponding to GIOB0) is pending with a high priority.
		:	:
		10h	Interrupt 16 (corresponding to GIOB7) is pending with a high priority.
		:	:
		20h	Interrupt 32 (corresponding to GIOD7) is pending with a high priority.
		21h-3Fh	Reserved

25.5.10 GIO Emulation B Register (GIOEMU2)

The GIOEMU2 register is a read-only register. The contents of this register are identical to the contents of GIOOFF2. The intention for the this register is that software can use it without clearing the flags. Figure 25-16 and Table 25-13 describe this register.

NOTE: The corresponding flag in the GIOFLG register is not cleared when the GIOEMU2 register is read.

Figure 25-16. GIO Emulation 2 Register (GIOEMU2) [offset = 30h]

31–16
Reserved
R-0

15–6	5–0
Reserved	GIOEMU2
R-0	R-0

LEGEND: R = Read only; -*n* = value after reset

Table 25-13. GIO Emulation 2 Register (GIOEMU2) Field Descriptions

Bit	Field	Value	Description
31-6	Reserved	0	Reads return 0. Writes have no effect.
5-0	GIOEMU2		GIO offset emulation 2. These bits index the currently pending low-priority interrupt. No register or flag is cleared by reading this register.
		0	No interrupt is pending.
		1h	Interrupt 0 (corresponding to GIOA0) is pending with a low priority.
		:	:
		8h	Interrupt 7 (corresponding to GIOA7) is pending with a low priority.
		9h	Interrupt 8 (corresponding to GIOB0) is pending with a low priority.
		:	:
		10h	Interrupt 16 (corresponding to GIOB7) is pending with a low priority.
		:	:
		20h	Interrupt 32 (corresponding to GIOD7) is pending with a low priority.
		21h-3Fh	Reserved

25.5.11 GIO Data Direction Registers (GIODIR[A-B])

The GIODIR register controls whether the pins of a given port are configured as inputs or outputs. Figure 25-17 and Table 25-14 describe this register.

Figure 25-17. GIO Data Direction Registers (GIODIR[A-B]) [offset = 34h, 54h]

31	16
Reserved	
R-0	

15	8	7	0
Reserved		GIODIR[7:0]	
R-0		R/W-0	

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 25-14. GIO Data Direction Registers (GIODIR[A-B]) Field Descriptions

Bit	Field	Value	Description
31-8	Reserved	0	Reads return 0. Writes have no effect.
7-0	GIODIR[*n*]		GIO data direction of port n, pins [7:0]
		0	The GIO pin is an input. Note: If the pin direction is set as an input, the output buffer is tristated.
		1	The GIO pin is an output.

25.5.12 GIO Data Input Registers (GIODIN[A-B])

Values in the GIODIN register reflect the current state (high = 1 or low = 0) on the pins of the port. Figure 25-18 and Table 25-15 describe this register.

Figure 25-18. GIO Data Input Registers (GIODIN[A-B]) [offset = 38h, 58h]

31	16
Reserved	
R-0	

15	8	7	0
Reserved		GIODIN[7:0]	
R-0		R/W-0	

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 25-15. GIO Data Input Registers (GIODIN[A-B]) Field Descriptions

Bit	Field	Value	Description
31-8	Reserved	0	Reads return 0. Writes have no effect.
7-0	GIODIN[*n*]		GIO data input for port n, pins [7:0]
		0	The pin is at logic low (0).
		1	The pin is at logic high (1).

25.5.13 GIO Data Output Registers (GIODOUT[A-B])

Values in the GIODOUT register specify the output state (high = 1 or low = 0) of the pins of the port when they are configured as outputs. Figure 25-19 and Table 25-16 describe this register.

NOTE: Values in the GIODSET register set the data output control register bits to 1 regardless of the current value in the GIODOUT bits.

Figure 25-19. GIO Data Output Registers (GIODOUT[A-B]) [offset = 3Ch, 5Ch]

31	16
Reserved	
R-0	

15	8	7	0
Reserved		GIODOUT[7:0]	
R-0		R/W-0	

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 25-16. GIO Data Output Registers (GIODOUT[A-B]) Field Descriptions

Bit	Field	Value	Description
31-8	Reserved	0	Reads return 0. Writes have no effect.
7-0	GIODOUT[*n*]		GIO data output of port n, pins[7:0].
		0	The pin is driven to logic low (0).
		1	The pin is driven to logic high (1).
			Note: Output is in high impedance state if the GIOPDRx bit = 1 and GIODOUTx bit = 1.
			Note: GIO pin is placed in output mode by setting the GIODIRx bit to 1.

25.5.14 GIO Data Set Registers (GIODSET[A-B])

Values in this register set the data output control register bits to 1 regardless of the current value in the GIODOUT bits. The contents of this register reflect the contents of GIODOUT. Figure 25-20 and Table 25-17 describe this register.

Figure 25-20. GIO Data Set Registers (GIODSET[A-B]) [offset = 40h, 60h]

31	16
Reserved	
R-0	

15	8	7	0
Reserved		GIODSET[7:0]	
R-0		R/W-0	

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 25-17. GIO Data Set Registers (GIODSET[A-B]) Field Descriptions

Bit	Field	Value	Description
31-8	Reserved	0	Reads return 0. Writes have no effect.
7-0	GIODSET[*n*]		GIO data set for port n, pins[7:0]. This bit drives the output of GIO pin high.
		0	Write: Writing a 0 has no effect.
		1	Write: The corresponding GIO pin is driven to logic high (1).
			Note: The current logic state of the GIODOUT bit will also be displayed by this bit.
			Note: GIO pin is placed in output mode by setting the GIODIRx bit to 1.

25.5.15 GIO Data Clear Registers (GIODCLR[A-B])

Values in this register clear the data output register (GIO Data Output Register [A-H]) bit to 0 regardless of its current value. The contents of this register reflect the contents of GIODOUT. Figure 25-21 and Table 25-18 describe this register.

Figure 25-21. GIO Data Clear Registers (GIODCLR[A-B]) [offset = 44h, 64h]

31	16
Reserved	
R-0	

15	8	7	0
Reserved		GIODCLR[7:0]	
R-0		R/W-0	

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 25-18. GIO Data Clear Registers (GIODCLR[A-B]) Field Descriptions

Bit	Field	Value	Description
31-8	Reserved	0	Reads return 0. Writes have no effect.
7-0	GIODCLR[*n*]		GIO data clear for port n, pins[7:0]. This bit drives the output of GIO pin low.
		0	Write: Writing a 0 has no effect.
		1	Write: The corresponding GIO pin is driven to logic low (0).
			Note: The current logic state of the GIODOUT bit will also be displayed by this bit.
			Note: GIO pin is placed in output mode by setting the GIODIRx bit to 1.

25.5.16 GIO Open Drain Registers (GIOPDR[A-B])

Values in this register enable or disable the open drain capability of the data pins. Figure 25-22 and Table 25-19 describe this register.

Figure 25-22. GIO Open Drain Registers (GIOPDR[A-B]) [offset = 48h, 68h]

31	16
Reserved	
R-0	

15	8	7	0
Reserved		GIOPDR[7:0]	
R-0		R/W-0	

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 25-19. GIO Open Drain Registers (GIOPDR[A-B]) Field Descriptions

Bit	Field	Value	Description
31-8	Reserved	0	Reads return 0. Writes have no effect.
7-0	GIOPDR[*n*]		GIO open drain for port n, pins[7:0]
		0	The GIO pin is configured in push/pull (normal GIO) mode. The output voltage is V_{OL}or lower if GIODOUT bit = 0 and V_{OH} or higher if GIODOUT bit = 1.
		1	The GIO pin is configured in open drain mode. The GIODOUTx bit controls the state of the GIO output buffer: GIODOUTx = 0, the GIO output buffer is driven low; GIODOUTx = 1, the GIO output buffer is tristated.

25.5.17 GIO Pull Disable Registers (GIOPULDIS[A-B])

Values in this register enable or disable the pull control capability of the pins. Figure 25-23 and Table 25-20 describe this register.

Figure 25-23. GIO Pull Disable Registers (GIOPULDIS[A-B]) [offset = 4Ch, 6Ch]

31–16
Reserved
R-0

15–8	7–0
Reserved	GIOPULDIS[7:0]
R-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 25-20. GIO Pull Disable Registers (GIOPULDIS[A-B]) Field Descriptions

Bit	Field	Value	Description
31-8	Reserved	0	Reads return 0. Writes have no effect.
7-0	GIOPULDIS[*n*]		GIO pull disable for port n, pins[7:0]. Writes to this bit will only take effect when the GIO pin configured as an input pin.
		0	The pull functionality is enabled.
		1	The pull functionality is disabled.
			Note: The GIO pin is placed in input mode by clearing the GIODIRx bit to 0.

25.5.18 GIO Pull Select Registers (GIOPSL[A-B])

Values in this register select the pull up or pull down functionality of the pins. Figure 25-24 and Table 25-21 describe this register.

Figure 25-24. GIO Pull Select Registers (GIOPSL[A-B]) [offset = 50h, 70h]

31–16
Reserved
R-0

15–8	7–0
Reserved	GIOPSL[7:0]
R-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 25-21. GIO Pull Select Registers (GIOPSL[A-B]) Field Descriptions

Bit	Field	Value	Description
31-8	Reserved	0	Reads return 0. Writes have no effect.
7-0	GIOPSL[*n*]		GIO pull select for port n, pins[7:0]
		0	The pull down functionality is select, when pull up/pull down logic is enabled.
		1	The pull up functionality is select, when pull up/pull down logic is enabled.
			Note: The pull up/pull down functionality is enabled by clearing corresponding bit in GIOPULDIS to 0.

25.6 I/O Control Summary

The behavior of the output buffer and the pull control is summarized in Table 25-22.

Table 25-22. Output Buffer and Pull Control Behavior for GIO Pins

Module under Reset?	Pin Direction (GIODIR)[1][2]	Open Drain Enable (GIOPDR)[1][3]	Pull Disable (GIOPULDIS)[1][4]	Pull Select (GIOPSL)[1][5]	Pull Control	Output Buffer[6]
Yes	X	X	X	X	Enabled	Disabled
No	0	X	0	0	Pull down	Disabled
No	0	X	0	1	Pull up	Disabled
No	0	X	1	0	Disabled	Disabled
No	0	X	1	1	Disabled	Disabled
No	1	0	X	X	Disabled	Enabled
No	1	1	X	X	Disabled	Enabled

[1] X = Don't care

[2] GIODIR = 0 for input; = 1 for output

[3] See Section 25.5.16

[4] GIOPULDIS = 0 for enabling pull control; = 1 for disabling pull control

[5] GIOPSL= 0 for pull-down functionality; = 1 for pull-up functionality

[6] If open drain is enabled, output buffer will be disabled if a high level (1) is being output.

Chapter 26

SPNU563A–March 2018

FlexRay Module

This chapter provides the specification for TI's FlexRay module and its features from the application programmer's point of view.

Topic **Page**

26.1 Overview

The FlexRay module performs communication according to the FlexRay protocol specification v2.1 Rev. A. The sample clock bit rate can be programmed to values up to 10 Mbit/s. Additional bus driver (BD) hardware is required for connection to the physical layer.

For communication on a FlexRay network, individual message buffers with up to 254 data bytes are configurable. The message storage consists of a single-ported message RAM that holds up to 128 message buffers. All functions concerning the handling of messages are implemented in the message handler. Those functions are the acceptance filtering, the transfer of messages between the two FlexRay Channel Protocol Controllers and the message RAM, maintaining the transmission schedule as well as providing message status information.

The register set of the FlexRay module can be accessed directly by the CPU via the VBUS interface. These registers are used to control, configure and monitor the FlexRay channel protocol controllers, message handler, global time unit, system universal control, frame and symbol processing, network management, interrupt control, and to access the message RAM via the input / output buffer.

26.1.1 Feature List

- Conformance with FlexRay protocol specification v2.1 Rev. A
- Data rates of up to 10 Mbit/s on each channel
- Up to 128 message buffers
- 8 Kbyte of message RAM for storage of, for example, 128 message buffers with maximum of 48-byte data section or up to 30 message buffers with 254-byte data section
- Configuration of message buffers with different payload lengths
- One configurable receive FIFO
- Each message buffer can be configured as receive buffer, as transmit buffer or as part of the receive FIFO
- CPU access to message buffers via input and output buffer
- Specialized DMA like FlexRay Transfer Unit (FTU) for automatic data transfer between data memory and message buffers without CPU interaction
- Filtering for slot counter, cycle counter, and channel
- Maskable module interrupts
- Supports Network Management

26.1.2 FlexRay Module Block Diagram

The TI FlexRay module, Figure 26-1, contains the following blocks:

- Peripheral Interface (VBUS IF)

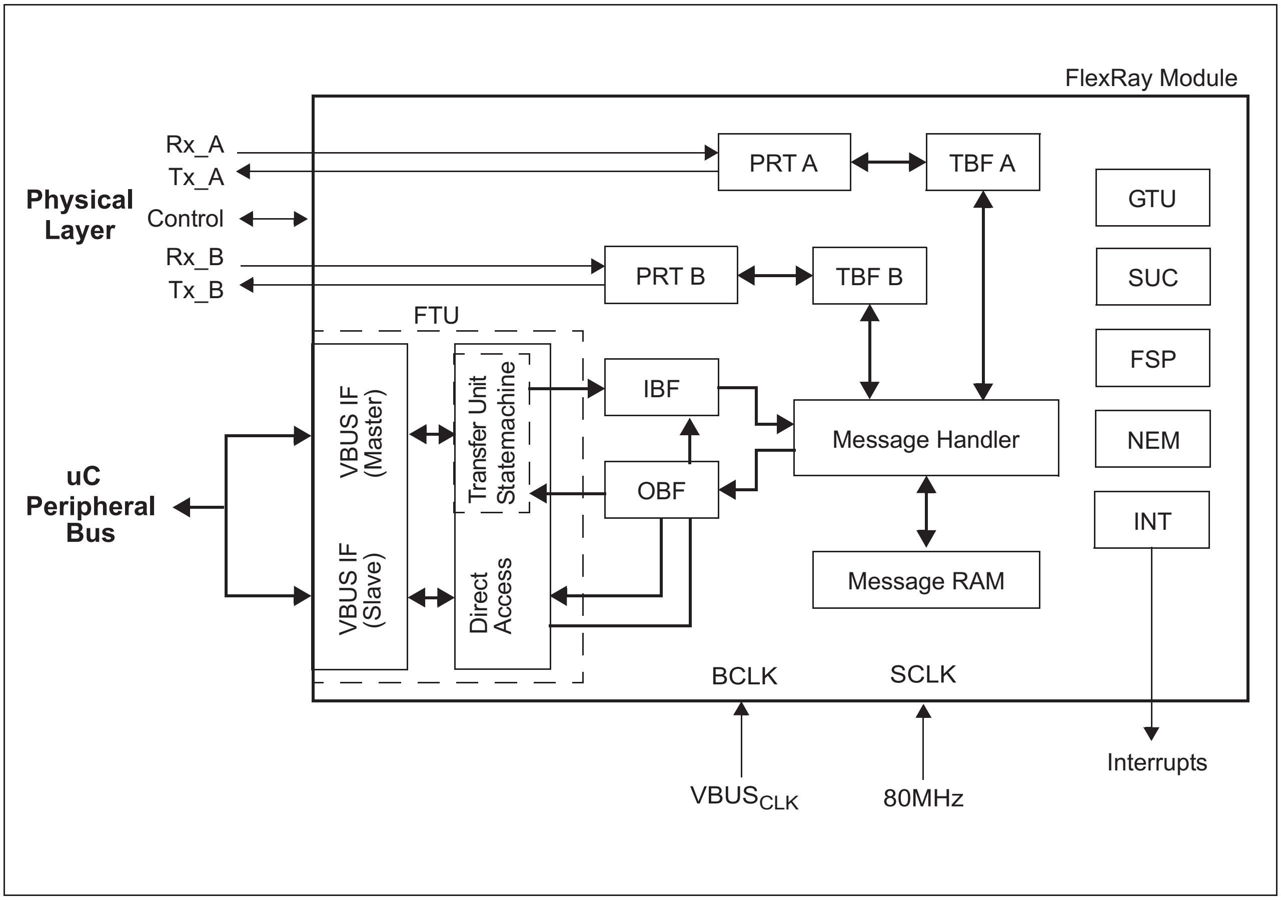
  Interface to the Peripheral Bus of the TMS570 microcontroller architecture. The FlexRay module can either act as a VBUS master or VBUS slave.

- FlexRay Transfer Unit (FTU)

  The internal intelligent state-machine (Transfer Unit State Machine) is able to transfer data between the input buffer (IBF) and output buffer (OBF) of the communication controller and the system memory without CPU interaction.

NOTE: Since the FlexRay module is accessed through the FTU, the FTU must be powered up by the corresponding bit in the Peripheral Power Down Registers of the System Module before accessing any FlexRay module register. For details, refer to the *Architecture* chapter and the device-specific data manual.

Figure 26-1. FlexRay Module Block Diagram



- Input Buffer (IBF)

  For write access to the message buffers configured in the message RAM, the CPU or the FTU can write the header and data section for a specific message buffer to the input buffer. The message handler then transfers the data from the input buffer to the selected message buffer in the message RAM.
- Output Buffer (OBF)

  For read access to a message buffer configured in the message RAM the message handler transfers the selected message buffer to the output buffer. After the transfer has completed, the CPU or the FTU can read the header and data section of the transferred message buffer from the output buffer.
- Message Handler (MHD)

  The message handler controls data transfers between the following components:
  - Input / output buffer and message RAM
  - Transient buffer RAMs of the two FlexRay protocol controllers and message RAM
- Message RAM

  The message RAM stores up to 128 FlexRay message buffers together with the related configuration data (header and data partition).
- The Transient Buffer RAM (TBF A/B):

  Stores the data section of two complete messages.

- FlexRay Channel Protocol Controller (PRT A/B)

  The FlexRay channel protocol controllers consist of a shift register and the FlexRay protocol FSM (Finite State Machine). They are connected to the transient buffer RAMs for intermediate message storage and to the physical layer via bus drivers (BD).

  They perform the following functionality:

  – Control and check of bit timing
  – Reception / transmission of FlexRay frames and symbols
  – Check of header CRC
  – Generation / check of frame CRC
  – Interfacing to bus driver

  The FlexRay channel protocol controllers have interfaces to:

  – Physical layer (bus driver)
  – Transient buffer RAM
  – Message handler
  – Global Time Unit
  – System universal control
  – Frame and symbol processing
  – Network management
  – Interrupt control
- Global time unit (GTU)

  The GTU performs the following functions:

  – Generation of microtick
  – Generation of macrotick
  – Fault tolerant clock synchronization by FTM algorithm
    - rate and offset correction
    - offset correction
  – Cycle counter
  – Timing control of static segment
  – Timing control of dynamic segment (minislotting)
  – Support of external clock correction
- System Universal Control (SUC)

  The SUC controls the following functions:

  – Configuration
  – Wakeup
  – Startup
  – Normal Operation
  – Passive Operation
  – Monitor Mode
- Frame and Symbol Processing (FSP)

  The frame and symbol processing controls the following functions:

  – Checks the correct timing of frames and symbols
  – Tests the syntactical and semantic correctness of received frames
  – Sets the slot status flags
- Network Management (NEM)

  Handles the network management vector.

- Interrupt Control (INT)

  The interrupt controller performs the following functions:
  - Provides error and status interrupt flags
  - Enable / disable interrupt sources
  - Assignment of interrupt sources to the two module interrupt lines
  - Enable / disable module interrupt lines
  - Manages the two interrupt timers
  - Stop watch time capturing
- 80-MHz Clock Signal

  NOTE: VCLKA2 is used to provide the 80-MHz clock to the FlexRay Module. The second PLL / Clock Source 6 in the microcontroller is typically used as source for VCLKA2.

  Clock signal for the sample clock (SCLK) of the FlexRay module.
- Module Clock ($VBUS_{CLK}$)

  The FlexRay module clock (BCLK) is derived from the Peripheral Clock $VBUS_{CLK}$ of the microcontroller.

26.1.3 FlexRay Module Blocks

Figure 26-2 shows the different module blocks of the FlexRay module: the communication controller, the transfer unit, and the transfer unit RAM. The RAM of the communication controller is only memory-mapped in test mode, where it is mapped to the register set address range. The address ranges of the three FlexRay blocks are shown in Table 26-1.

Figure 26-2. FlexRay Module Blocks

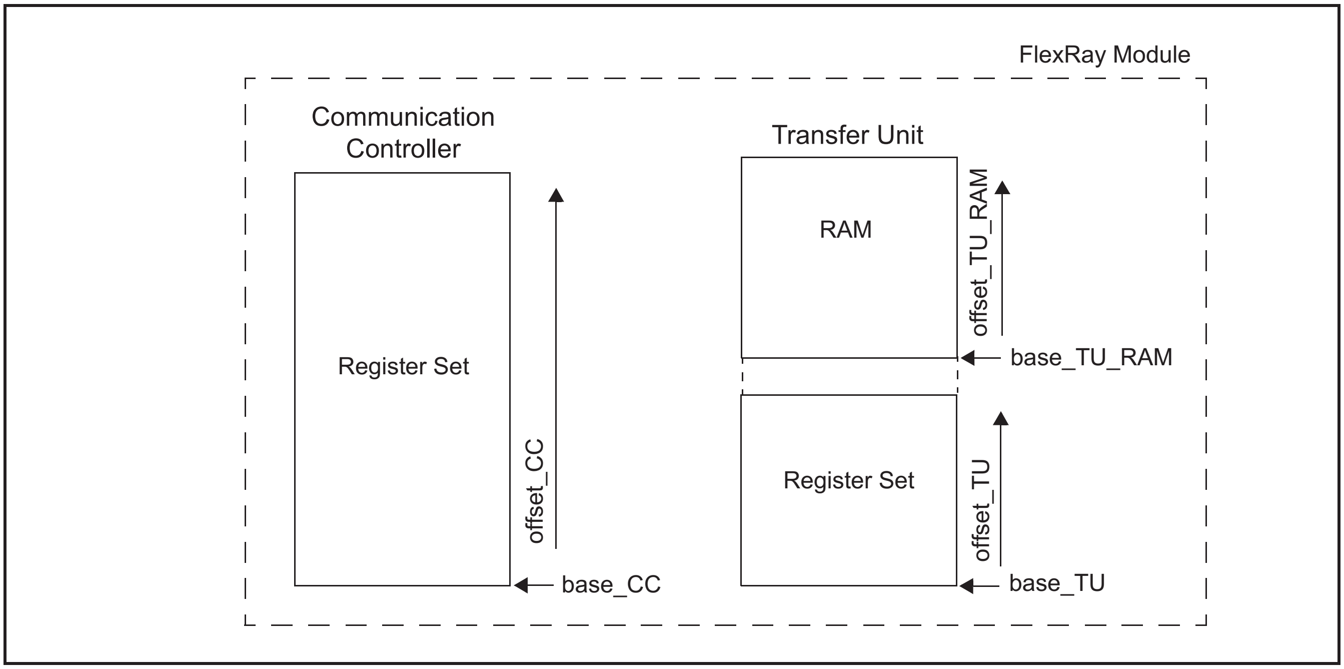


Table 26-1. FlexRay Address Range Table

Module	Address Range
FlexRay Communication Controller	0xFFF7_C800 - 0xFFF7_CFFF
FlexRay TU	0xFFF7_A000 - 0xFFF7_A1FF
FlexRay TU RAM	0xFF50_0000 - 0xFF51_FFFF

26.2 Module Operation

26.2.1 Transfer Unit

The FlexRay Transfer Unit (FTU), Figure 26-3, has an internal intelligent state-machine (Transfer Unit State Machine) to transfer data between the Input and Output Buffer Interfaces of the FlexRay core module and the system memory of the microcontroller without CPU interaction. It operates in a similar manner to a DMA (Direct Memory Access) module.

The FlexRay Input Buffer (IBF) and FlexRay Output Buffer (OBF) can also be accessed directly by the CPU. In this case the IBF and OBF are 8-, 16-, and 32-bit accessible. For transfers using the Transfer Unit State Machine only 4 × 32-bit data packages (4 word bursts) are supported.

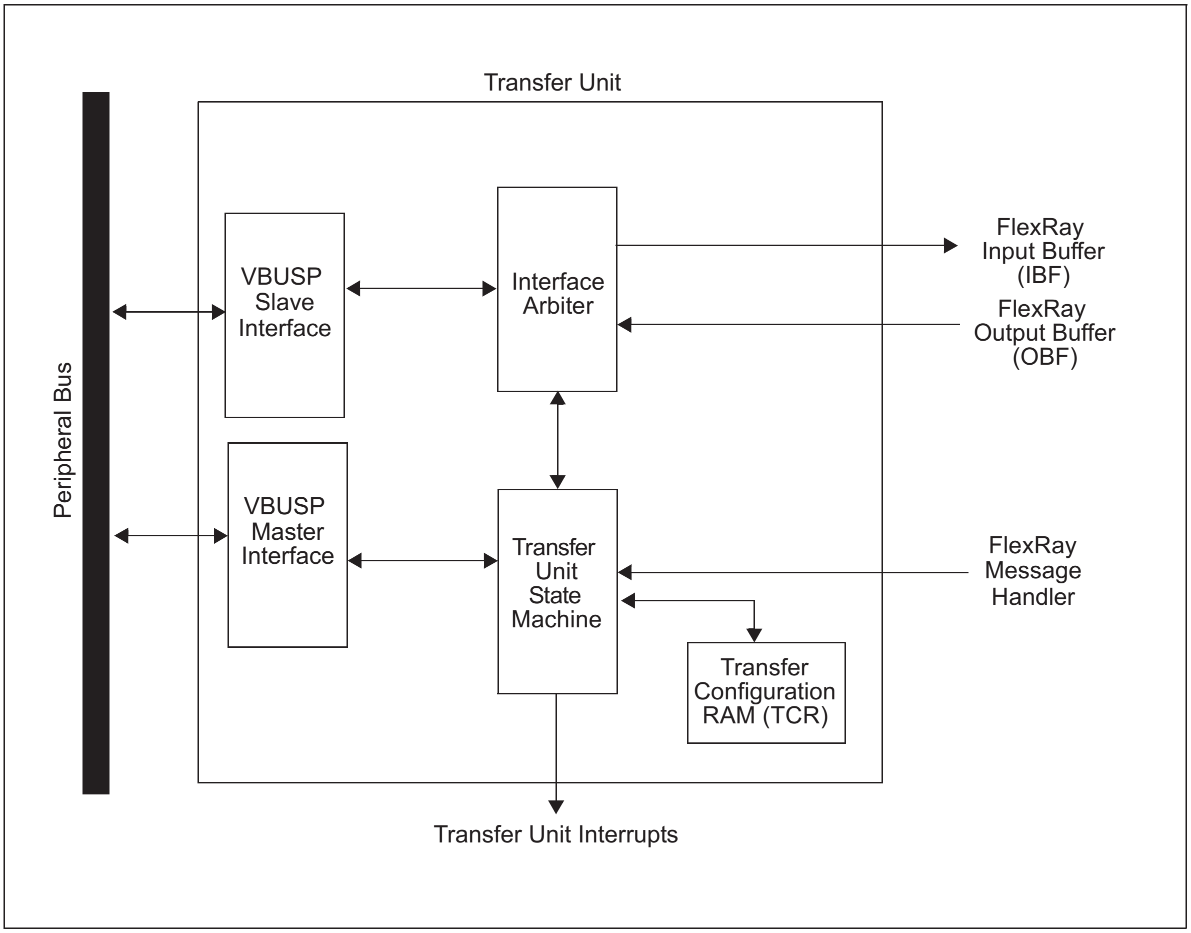
The Interface Arbiter controls the access to the IBF and OBF. Direct CPU accesses to IBF and OBF are not possible, if the Transfer Unit State Machine is switched on. Accesses will be ignored and the associated error interrupt will be generated.

The Transfer Unit State Machine is the head of all manual, event driven and automatic message transfer activities. It controls the Transfer Unit interrupt generation related to transfer protocol correctness, status and violations of the message transfers.

With the Transfer Configuration RAM (TCR) the transfer sequence, executed by the Transfer Unit State Machine, can be configured.

The usage of the Transfer Unit allows the user to setup a mirror of the FlexRay message RAM in the fast accessible data RAM of the microcontroller. The Transfer Unit can handle the data transfers between the data RAM and the FlexRay message RAM in the ‘background’ without CPU interaction.

Figure 26-3. Transfer Unit



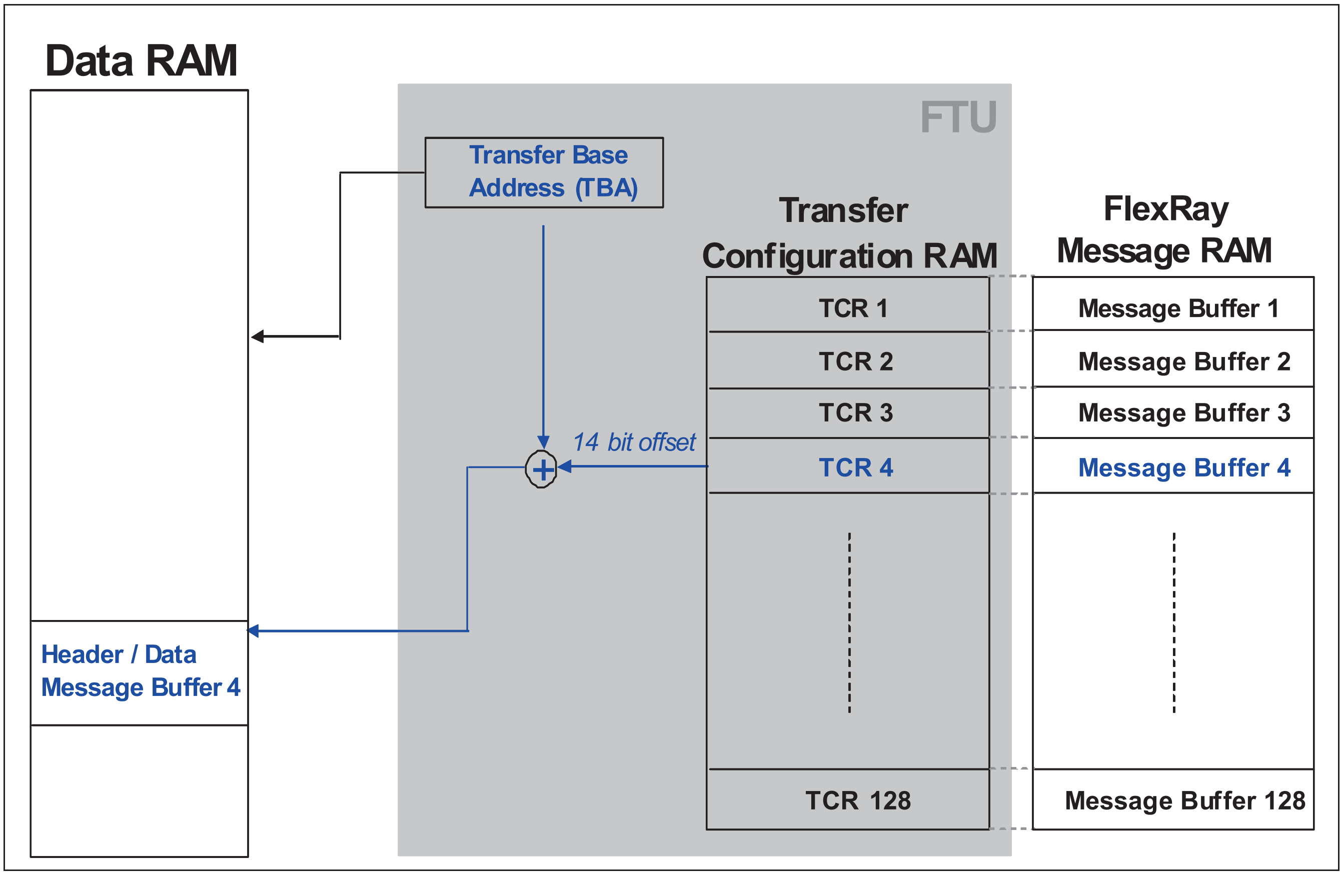
26.2.1.1 Transfer Unit Functional Description

Figure 26-4 shows the principle of the Transfer Unit operation.

Each FlexRay message buffer of the FlexRay message buffer RAM has one Transfer Configuration RAM (TCR) entry assigned to it, that is, message buffer 1 is assigned to TCR1, message buffer 2 is assigned to TCR2, and so on.

The Transfer Base Address (TBA) register of the Transfer Unit holds the message buffer base address in the data RAM. Each Transfer Configuration RAM (TCR) entry contains a 14 bit offset value to the dedicated message buffer area in the data RAM.

Figure 26-4. FlexRay Transfer Unit Operation Principle



The following two diagrams show the principle of the Transfer Unit operation including Transfer State Machine (see Figure 26-5) and Event State Machine (see Figure 26-6).

Figure 26-5. FlexRay Transfer Unit Operation Principle for Transfer FSM (simplified)

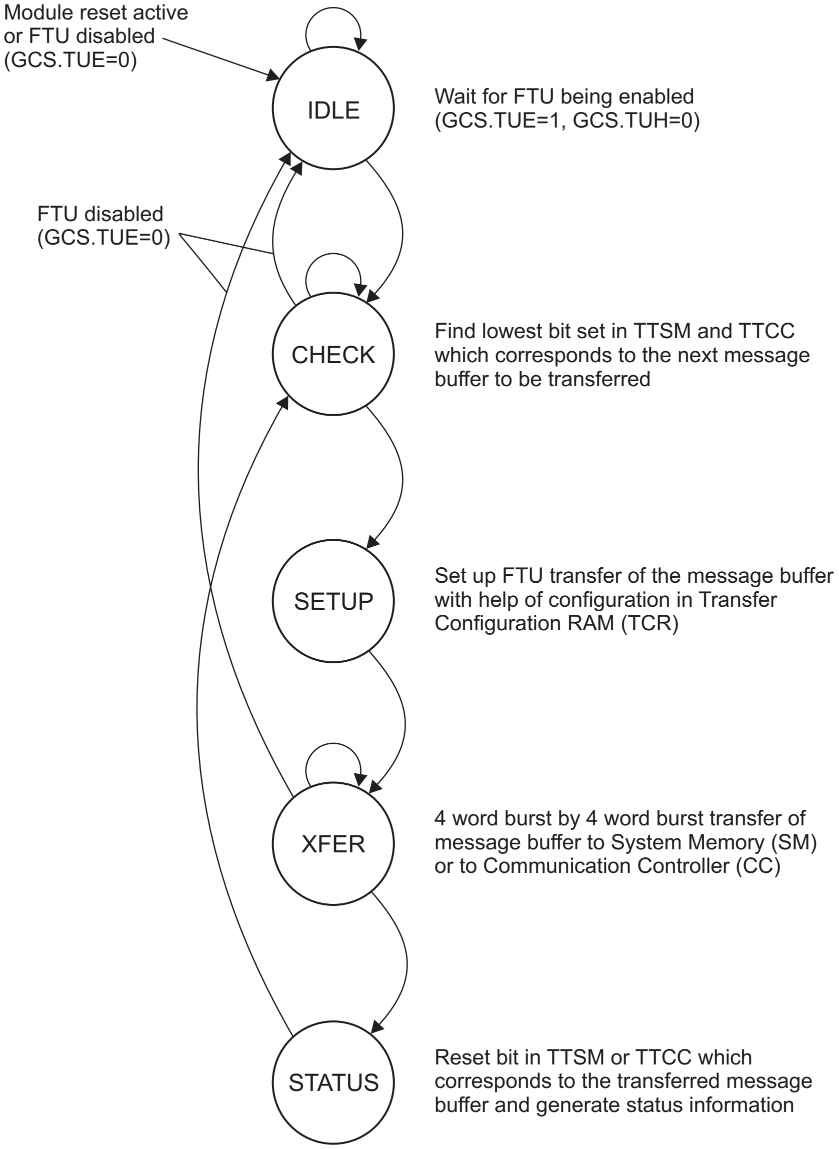
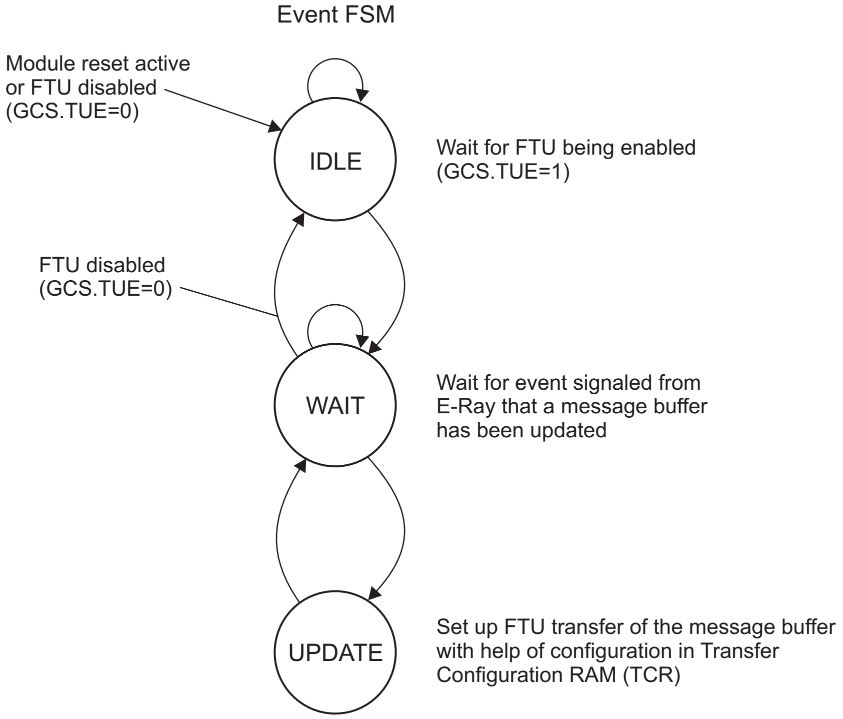

Module reset active
or FTU disabled
(GCS.TUE=0)
IDLE
Wait for FTU being enabled
(GCS.TUE=1, GCS.TUH=0)
FTU disabled
(GCS.TUE=0)
CHECK
Find lowest bit set in TTSM and TTCC
which corresponds to the next message
buffer to be transferred
SETUP
Set up FTU transfer of the message buffer
with help of configuration in Transfer
Configuration RAM (TCR)
XFER
4 word burst by 4 word burst transfer of
message buffer to System Memory (SM)
or to Communication Controller (CC)
STATUS
Reset bit in TTSM or TTCC which
corresponds to the transferred message
buffer and generate status information

Figure 26-6. FlexRay Transfer Unit Operation Principle for Event FSM (simplified)


Event FSM
Module reset active or FTU disabled (GCS.TUE=0)
IDLE
Wait for FTU being enabled (GCS.TUE=1)
FTU disabled (GCS.TUE=0)
WAIT
Wait for event signaled from E-Ray that a message buffer has been updated
UPDATE
Set up FTU transfer of the message buffer with help of configuration in Transfer Configuration RAM (TCR)

ETESMS/R	CESMS/R	Set bit TTSM	Clear bit ETESMS/R
0	-	False	False
1	0	True	False
1	1	True	True

26.2.1.1.1 Transfer Control

26.2.1.1.1.1 Transfer Start and Halt

The Transfer Unit State Machine can be halted, effectively stopping the Transfer Unit transfer sequence (after completion of the current 4 word burst transfer cycle). After releasing from halt state, the Transfer Unit resumes exactly, where it was halted without data loss.

NOTE: It is the software’s responsibility to ensure data coherency when the FlexRay module continues to receive data, but the Transfer Unit doesn't transfer it.

26.2.1.1.1.2 Transfer Abort

A Transfer Unit transfer will be aborted and the Transfer Unit will be disabled automatically in case of:

- an ECC multi-bit error while accessing the Transfer Configuration RAM (TCR)
- an uncorrected bit error while accessing the Transfer Configuration RAM (TCR), when ECC single-bit error correction is disabled
- a memory protection error while accessing the data RAM of the microcontroller. In this case, the ongoing transfer is aborted but the TUE bit in GCS/R may not get reset. User shall clear the TUE bit manually by software.

26.2.1.1.1.3 Transfer Reset

The Transfer Unit State Machine can be reset by the Transfer Unit Enable (TUE) bit in the Global Control register. Though the Transfer Unit State Machine can be reset with the above, the module register contents and the Transfer Configuration RAM (TCR). So, after re-enabling the Transfer Unit no reconfiguration of the Transfer Unit is required.

26.2.1.1.1.4 Transfer Modes

Possible transfer sequence modes are:

- Manual by triggering the desired transfer by setting the corresponding bit in the Trigger Transfer to System Memory (TTSM) register or the Trigger Transfer to Communication Controller (TTCC) register
- Event-Driven (transfers from FlexRay Communication Controller to the System Memory only) using the Enable Transfer on Event to System Memory (ETESM) register.
- Single or continuous event driven transfers by using the Clear on Event to System Memory (CESM)

The transfer event trigger in general occurs upon completion of a reception or transmission of a frame through the FlexRay bus. Table 26-2 shows more details: Conditions marked with 'X' per row must match to trigger a FTU transfer event as configured in the Transfer Configuration RAM (TCR):

Table 26-2. FlexRay Transfer Unit Event Trigger Conditions

	Event on Channel A	Event on Channel B	Frame belonging to static segment or first slot of dynamic segment	Frame belonging to dynamic segment, except first slot of dynamic segment	Bus activity detected on Channel A (MBS.ESA = 0)	Bus activity detected on Channel B (MBS.ESB = 0)
FTU Event Trigger for Receive Message Buffers	X		X			
		X	X			
	X			X	X	
		X		X		X
FTU Event Trigger for Transmit Message Buffers	X				X	
		X				X

NOTE: By setting the corresponding bit in the Enable Transfer on Event to System Memory (ETESM) register prior to an on-demand transfer to the Communication Controller by way of the Trigger Transfer to Communication Controller (TTCC) register, an event-triggered transmission back to the System Memory can be initiated, once the buffer has been sent out on the FlexRay bus. This mechanism can be used, for instance, to automatically read back the header status information to the system memory after a transmission occurred.

The transmission or reception of null frames in the static segment of a FlexRay communication cycle triggers transfers of the transfer unit. The header and/or payload is transferred to the system memory, if the corresponding bits THTSM and/or TPTSM in the Transfer Configuration RAM (TCR) are set. If neither THTSM nor TPTSM bit is set in TCR, neither header nor payload gets transferred. The corresponding bit in the Transfer to System Memory Occurred register (TSMO) gets set in all cases.

26.2.1.1.1.5 Transfer Size and Types

The data transferred by the Transfer Unit can be selected as:

- data and header section
- header section only
- data section only

The number of transferred payload words is derived from the Payload Length Configured (PLC) information configured in the Write Header Section 2 (WRHS2) register.

As only 4 word bursts are supported for the Transfer Unit transfers, only multiple of 4x32-bit data packets are supported. Additional transferred words are undefined, as indicated in Figure 26-7 and Figure 26-8.

Figure 26-7. Example: FTU Read Transfer of 6 Words

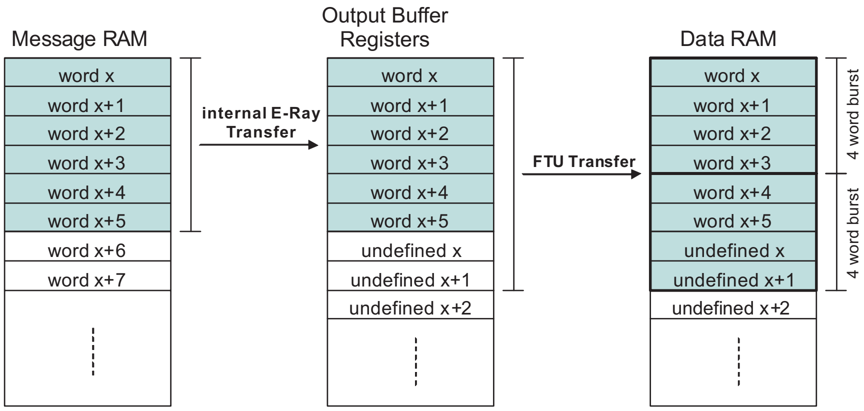
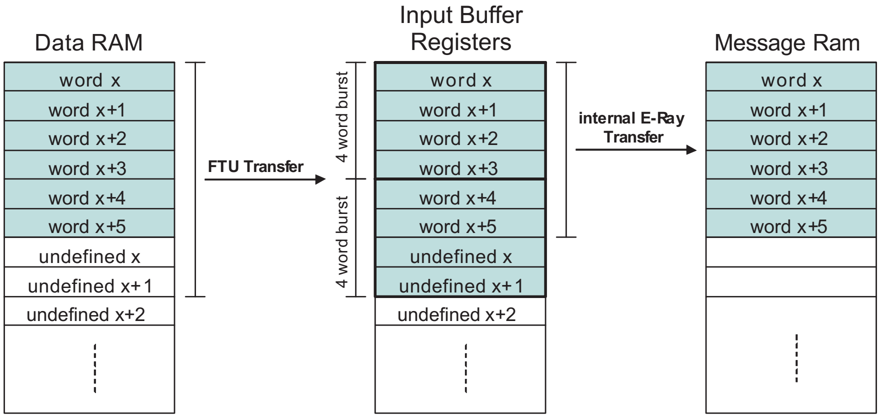


Figure 26-8. Example: FTU Write Transfer of 6 Words



Physically the FTU continues reading the additional words from the source location it started the burst transfer. Therefore, on reads, the additional transferred words depend on the contents of the Communication Controller Output Buffer Registers as indicated in Figure 26-7. On writes the additional words depend on the contents of the data RAM, as shown in Figure 26-8. The additional data will be written to the Communication Controller's Input Buffer Registers, but not transferred to the message RAM.

NOTE: It should be ensured that the allocated data RAM space for FTU transfers ends on 4x32 bit boundary to avoid possible data overwrites or memory protection issues on FTU reads and avoid reading the additional data from the source location on FTU writes.

26.2.1.1.1.6 Transfer Status Indication

There are 3 registers indicating the transfer status:

- Transfer Status Current Buffer (TSCB) shows the current transfer buffer status
- Last Transferred Buffer to Communication Controller (LTBCC) indicates the last completed buffer transfer to the communication controller
- Last Transferred Buffer to System Memory (LTBSM) shows the last completed buffer transfer to system memory

26.2.1.1.1.7 Transfer Mirror Function

In order to efficiently access the transfer unit status registers in the system memory, the following registers can be mirrored to the system memory starting at the base address defined in the Base Address of Mirrored Status (BAMS) register:

- Transfer Status Current Buffer (TSCB)
- Last Transferred Buffer to Communication Controller (LTBCC)
- Last Transferred Buffer to System Memory (LTBSM)
- Transfer to System Memory Occurred 1/2/3/4 (TSMO1-4)
- Transfer to Communication Controller Occurred 1/2/3/4 (TCCO1-4)
- Transfer Occurred OFFset (TOOFF)

The mirrored values are updated after completion of a buffer transfer.

The mirroring of these registers can be disabled if not needed.

Table 26-3. Mirroring Address Mapping

Address	Register
BAMS+0x00	TSCB
BAMS+0x04	LTBCC
BAMS+0x08	LTBSM
BAMS+0x0C	TSMO1
BAMS+0x10	TSMO2
BAMS+0x14	TSMO3
BAMS+0x18	TSMO4
BAMS+0x1C	TCCO1
BAMS+0x20	TCCO2
BAMS+0x24	TCCO3
BAMS+0x28	TCCO4
BAMS+0x2C	TOOFF

26.2.1.1.1.8 Endianness Correction

For the data transfer by the Transfer Unit an Endianness correction mechanism can be used to switch big Endianness data to little Endianness data and vice versa.

For maximum flexibility, 6 bits are available in the Global Control Set/Reset Register (GCS/R) to control.

- Header Data byte-order
- Payload Data byte-order
- Byte-order of the FlexRay Core registers and the Transfer Configuration RAM data of the Transfer Unit

independently and in both directions.

26.2.1.1.1.9 Transfer Data Package

Table 26-4 shows the data of a transfer data package. Independent of whether the header gets transferred or not, the buffer address always points to element Header1.

Table 26-4. Mirroring Address Mapping

Address	Register
0x0000	Header1
0x0004	Header2
0x0008	Header3
0x000C	Buffer Status[1]
0x0010	Payload1
0x0014	Payload2
:	:
0x010C	Payload64

[1] Transferred only from Communication Controller to System Memory

26.2.1.1.1.10 Transfer Start Address to Message Buffer Number Assignment

The assignment of a FlexRay message buffer number to the transfer location in system memory is done by the combination of:

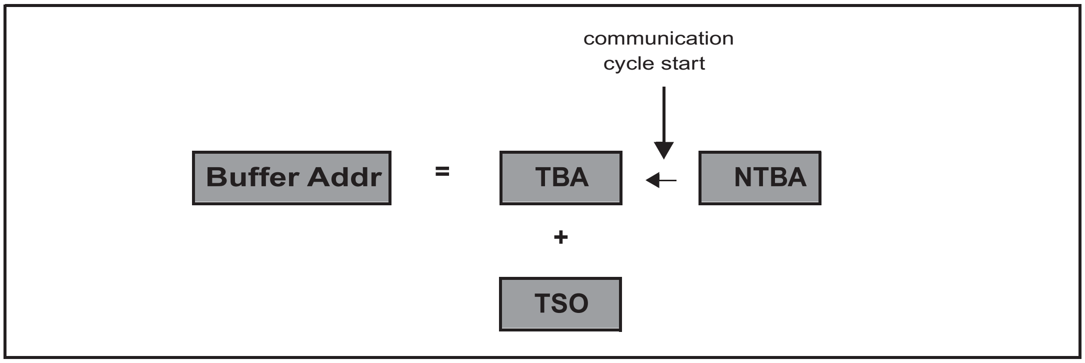
- the Transfer Start Offset (TSO) field in a Transfer Configuration RAM (TCR) entry
- the Transfer Base Address (TBA) register

Each entry of the TCR holds a 14 bit offset value (TSO). The TSO offset will be added to the content of the TBA register. The TBA register holds the 32bit base address-pointer to a location of the data RAM.

A value written to Next Transfer Base Address (NTBA) will be loaded in the TBA at the next communication cycle start. This allows efficient multi-buffering of the message buffers in the system memory. The Transfer Not Ready (NTR) flag in the Transfer Error Interrupt Flag (TEIF) register can be used to determine, if NTBA can be reloaded by the CPU.

NOTE: If a value is written to TBA, NTBA is set to the same value.

Figure 26-9. Transfer Start Address to Message Buffer Number Assignment



26.2.1.1.1.11 Transfer Priority

The Transfer Unit will transfer the message buffers from low to high message buffer numbers.

In case the same buffer is pending in both the Trigger Transfer to Communication (TTCC) register and the Trigger Transfer to System Memory (TTSM) register, the priority between TTCC and TTSM is determined by the Transfer Priority bit (GC.PRIO) in the Transfer Unit Global Control Set/Reset Register (GCS/R).

26.2.1.1.1.12 Read Transfers

A read transfer is the data transfer from FlexRay message buffer RAM to the system memory of the microcontroller.

For read transfers the registers Trigger Transfer to System Memory (TTSM), Enable Transfer on Event to System Memory (ETESM) and Clear on Event to System Memory (CESM) have to be setup.

The amount and type of data to be transferred can be selected as:

- data and header section
- header section only
- data section only

which can be configured on the Transmit Configuration RAM (TCR).

The number of 32 bit words per buffer to be transferred is read from the Payload Length Configured (RDHS2.PLC) configuration information. This information is part of the header section stored in the message RAM of the Communication Controller.

26.2.1.1.1.13 Write Transfers

A write transfer is the data transfer from the system memory of the microcontroller to the FlexRay message buffer RAM.

For write transfers the Trigger Transfer to Communication (TTCC) register has to be setup.

The amount and type of data transferred can be selected as:

- data and header section
- header section only
- data section only

which can be configured on the Transmit Configuration RAM (TCR).

It can be configured in the TCR, if Set Transmission Request Host (STXRH) bit in the Input Buffer Command Mask (IBCM) of the Communication Controller should be set. This would trigger the transfer to the FlexRay bus.

If a data and header section transfer is selected, the number of 32 bit words to be transferred is read from the Payload Length Configured (PLC) configuration information stored in Header2 word in the system memory.

If a data section only transfer is selected, the number of 32 bit words to be transferred is read from the Payload Length Configured (RDHS2.PLC) configuration information. This information is part of the header section stored in the message RAM of the Communication Controller.

26.2.1.1.1.14 Transfer Unit Event Interface

The Transfer Unit Event Control generates transfer trigger signals for transfers in the following cases:

- For transmit (TX) message buffers, a write transfer trigger is generated, if a transmit event occurs. The configured TX message buffers generate a transfer trigger, except when a Nullframe in static segment or no frame in the dynamic segment is sent.
- For receive (RX) message buffers, a read transfer trigger is generated, if a receive event occurs in the static segment.
- For receive (RX) message buffers, a read transfer trigger is generated if a receive event occurs in the dynamic segment, updated in the current cycle and no Nullframe.

If a buffer is part of the FIFO, no transfer trigger is generated!

When the Transfer Unit is disabled (TUE bit in Global Control Register (GCS/R) is 0), no transfer trigger is generated, whereas if the Transfer Unit is enabled, but in halt mode (TUH bit and TUE bit in Global Control Register (GCS/R) are 1), the occurring triggers remain pending and get executed when the Transfer Unit will be resumed from halt mode.

26.2.1.1.2 Transfer Configuration RAM

The Transfer Configuration RAM (TCR) consists of 128 entries, one entry for each possible FlexRay buffer. Entry 1 is assigned to FlexRay buffer 1, entry 2 to FlexRay buffer 2,..., and entry 128 is assigned to FlexRay buffer 128.

Each TCR entry defines:

- data transfer size (header + data, header only or data only)
- whether the transmit request flag (STRXH) should be set for the data transferred by the FTU to the CC to send out the data to the FlexRay bus.
- the 14-bit buffer address offset, which, in combination with the Transfer Base Address defined in TBA, specifies the start of the corresponding FlexRay message buffer in the system memory RAM.

NOTE: It is recommended to clear the whole TCR before configuring it, in order to avoid unexpected transfer behavior due to old configuration contents or random TCR RAM contents after power on reset.

If a transfer is triggered but no transfer size (header or data) is setup in the TCR, no data will be transferred, but the corresponding flag in the Transfer to Communication Controller Occurred (TCCOx) or the Transfer to System Memory Occurred (TSMOx) will be set.

26.2.1.1.2.1 ECC Protection

The Transfer Configuration RAM (TCR) is ECC protected. The ECC multi-bit error interrupt generation is disabled by default and can be switched on by writing a 4 bit key to dedicated ECC lock bits in the Global Control Set/Reset Register (GCS/R).

The ECC protection supports single-bit error correction and double-bit error detection mechanism (SECDED). The ECC information is stored together with the corresponding 19-bit data word entry.

The ECC is checked each time a data word is read from the TCR RAM. If an ECC error is detected, the PE error flag is set in the Transfer Error Interrupt Flag (TEIF) register. The detection of an ECC single-bit error will be indicated by the SE flag in the TCR Single-Bit Error Status (TSBESTAT).

Additionally an uncorrectable RAM error interrupt/event will be generated. The uncorrectable RAM error interrupt/event is non maskable and therefore cannot be switched off. For ECC single-bit errors, the uncorrectable RAM error interrupt/event is generated, if the ECC single-bit error correction is disabled. The uncorrectable RAM error is hooked up to the ESM module (event).

The faulty TCR RAM address can be read from the ECC Error Address (PEADR) register. Equivalent information is available for ECC single-bit errors in the TCR Single-Bit Error Status (TSBESTAT) register. Independent of the ECC single-bit error correction being enabled or disabled, the TSBESTAT is updated. See Figure 26-26 for more details.

26.2.1.1.3 Memory Protection Mechanism

This feature allows to restrict accesses to certain areas in memory in order to protect critical application data from unintentionally being accessed by the Transfer Unit State Machine.

One memory section (start and end address) can be defined, which allows read and write accesses for the Transfer Unit State Machine.

If the end address is smaller or equal to the start address, data transfers will be blocked. Any accesses performed outside this memory area by the Transfer Unit State Machine result in no transfers being performed. In case of a protection violation a flag will be set and the Memory Protection Violation interrupt will be activated. The Transfer Unit State Machine will be disabled in this case.

The default setting of the Transfer Unit State Machine memory protection address range setup is 0x00000000 for start address and 0x00000000 for end address.

This means a valid address range must be setup, before the Transfer Unit can be used.

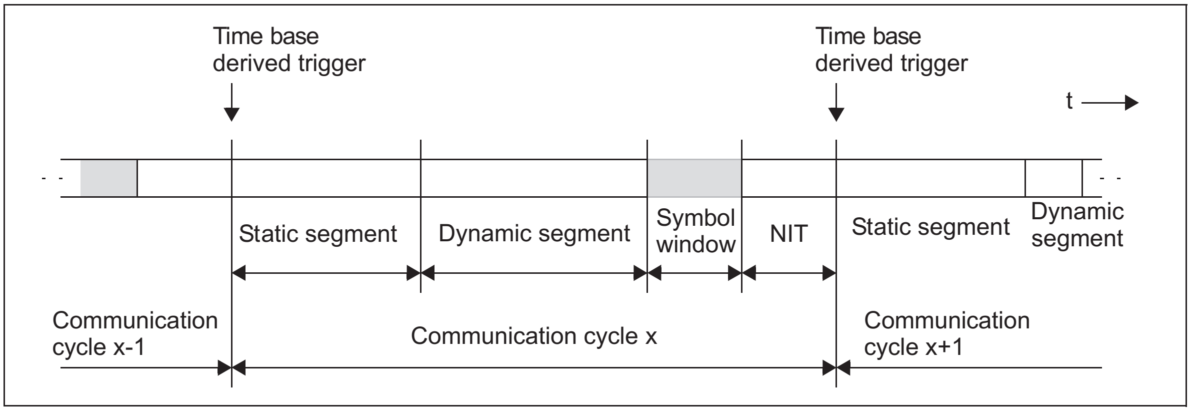
26.2.2 Communication Cycle

A communication cycle in FlexRay (Figure 26-10) consists of the following elements:

- Static segment
- Dynamic segment
- Symbol window
- Network idle time (NIT)

Static segment, dynamic segment, and symbol window form the network communication time (NCT). For each communication channel the slot counter starts at 1 and counts up until the end of the dynamic segment is reached. Both channels share the same arbitration grid which means that they use the same synchronized macrotick.

Figure 26-10. Structure of Communication Cycle



26.2.2.1 Static Segment

The Static Segment is characterized by the following features:

- Time slots of fixed length (optionally protected by bus guardian)
- Start of frame transmission at action point of the corresponding static slot
- Payload length same for all frames on both channels

Parameters: number of static slots GTUC7.NSS(9-0), static slot length GTUC7.SSL(9-0), Payload Length Static MHDC.SFDL(6-0), action point offset GTUC9.APO(5-0)

26.2.2.2 Dynamic Segment

The Dynamic Segment is characterized by the following features:

- All controllers have bus access (no bus guardian protection possible)
- Variable payload length and duration of slots, different for both channels
- Start of transmission at minislot action point

Parameters: number of minislots GTUC8.NMS(12-0), minislot length GTUC8.MSL(5-0), minislot action point offset GTUC9.MAPO(4-0), start of latest transmit (last minislot) MHDC.SLT(12-0)

26.2.2.3 Symbol Window

During the symbol window only one media access test symbol (MTS) may be transmitted per channel. MTS symbols are sent in NORMAL_ACTIVE state to test the bus guardian.

The symbol window is characterized by the following features:

- Send single symbol
- Transmission of the MTS symbol starts at the symbol windows action point

Parameters: Symbol Window Action Point Offset GTUC9.APO(4-0) (same as for static slots), Network Idle Time Start GTUC4.NIT(13-0)

26.2.2.4 Network Idle Time (NIT)

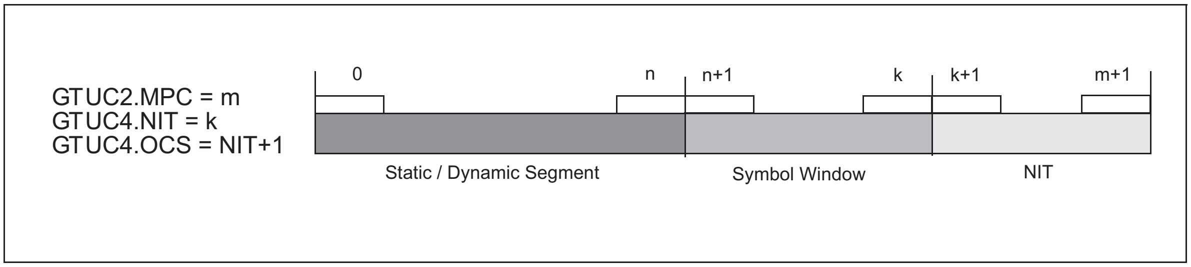
During network idle time the communication controller has to perform the following tasks:

- Calculate clock correction terms (offset and rate)
- Distribute offset correction over multiple macroticks after offset correction start
- Perform cluster cycle related tasks

Parameters: network idle time start GTUC4.NIT(13-0), offset correction start GTUC4.OCS(13-0)

26.2.2.5 Configuration of NIT Start and Offset Correction Start

Figure 26-11. Configuration of NIT Start and Offset Correction Start



The number of macroticks per cycle is assumed to be m. It is configured by programming GTUC2.MPC = m in the GTU Configuration register 2.

The static / dynamic segment starts with macrotick 0 and ends with macrotick n: n = static segment length + dynamic segment offset + dynamic segment length - 1MT

The static segment length is configured by GTUC7.SSL and GTUC7.NSS. The dynamic segment length is configured by GTUC8.MSL and GTUC8.NMS.

The dynamic segment offset is ActionPointOffset - MinislotActionPointOffset or 0 MT if the result is negative. For details, refer to the FlexRay Communications System Protocol Specification from the FlexRay Consortium.

The NIT starts with macrotick k+1 and ends with the last macrotick of cycle m-1. It has to be configured by setting GTUC4.NIT = k.

For this FlexRay module the offset correction start is required to be GTUC4.OCS >= GTUC4.NIT + 1 = k+1.

The length of symbol window results from the number of macroticks between the end of the static / dynamic segment and the beginning of the NIT. It can be calculated by k - n.

26.2.3 Communication Modes

The FlexRay protocol specification v2.1 Rev. A defines the Time-Triggered Distributed (TT-D) mode.

26.2.3.1 Time-Triggered Distributed (TT-D)

In TT-D mode the following configurations are possible:

- Pure static: minimum 2 static slots + symbol window (optional)
- Mixed static/dynamic: minimum 2 static slots + dynamic segment + symbol window (optional)

A minimum of two coldstart nodes need to be configured for distributed time-triggered operation. Two fault-free coldstart nodes are necessary for the cluster startup. Each startup frame must be a sync frame, therefore all coldstart nodes are sync nodes.

26.2.4 Clock Synchronization

In TT-D mode a distributed clock synchronization is used. Each node individually synchronizes itself to the cluster by observing the timing of received sync frames from other nodes.

26.2.4.1 Global Time

Activities in a FlexRay node, including communication, are based on the concept of a global time, even though each individual node maintains its own view of it. It is the clock synchronization mechanism that differentiates the FlexRay cluster from other node collections with independent clock mechanisms. The global time is a vector of two values; the cycle (cycle counter) and the cycle time (macrotick counter).

Cluster specific:

- Macrotick (MT) = basic unit of time measurement in a FlexRay network, a macrotick consists of an integer number of microticks (μT)
- Cycle length = duration of a communication cycle in units of macroticks (MT)

26.2.4.2 Local Time

Internally, nodes time their behavior with microtick resolution. Microticks are time units derived from the oscillator clock tick of the specific node. Therefore microticks are controller-specific units. They may have different duration in different controllers. The precision of a nodes local time difference measurements is a microtick (μT).

Node specific:

- Sample clock -> prescaler -> microtick (µT); typically 25ns.
- μT = basic unit of time measurement in a communication controller, clock correction is done in units of μTs
- Cycle counter + macrotick counter = nodes local view of the global time

26.2.4.3 Synchronization Process

Clock synchronization is performed by means of sync frames. Only preconfigured nodes (sync nodes) are allowed to send sync frames. In a two-channel cluster, a sync node has to send its sync frame on both channels.

For synchronization in FlexRay the following constraints have to be considered:

- Max. one sync frame per node in one communication cycle
- Max. 15 sync frames per cluster in one communication cycle
- Every node has to use a preconfigured number of sync frames (GTUC2.SNM(3-0)) for clock synchronization
- Minimum of two sync nodes required for clock synchronization and startup

For clock synchronization the time difference between expected and observed arrival time of sync frames received during the static segment is measured. In a two channel cluster the sync node has to be configured to send sync frames on both channels. The calculation of correction terms is done during NIT (offset: every cycle, rate: odd cycle) by using a FTA / FTM algorithm. For details see FlexRay protocol specification v2.1 Rev. A.

26.2.4.3.1 Offset (Phase) Correction

- Only deviation values measured and stored in the current cycle used
- For a two channel node the smaller value will be taken
- Calculation during NIT of every communication cycle
- Offset correction value calculated in even cycles used for error checking only
- Checked against limit values
- Correction value is a signed integer number of μTs

- Correction done in odd numbered cycles, distributed over the macroticks beginning at offset correction start up to cycle end (end of NIT) to shift nodes next start of cycle (MTs lengthened / shortened)

26.2.4.3.2 Rate (Frequency) Correction

- Pairs of deviation values measured and stored in even / odd cycle pair used
- For a two channel node the average of the differences from the two channels is used
- Calculated during NIT of odd numbered cycles
- Cluster drift damping is performed using global damping value
- Checked against limit values
- Correction value is a signed integer number of μTs
- Distributed over macroticks comprising the next even/odd cycle pair (MTs lengthened / shortened)

26.2.4.4 Sync Frame Transmission

Sync frame transmission is only possible from buffer 0 and 1. Message buffer 1 may be used for sync frame transmission in case that sync frames should have different payloads on the two channels. In this case bit MRC.SPLM has to be programmed to 1.

Message buffers used for sync frame transmission have to be configured with the key slot ID and can be (re)configured in DEFAULT_CONFIG or CONFIG state only. For nodes transmitting sync frames SUCC1.TXSY must be set to 1.

26.2.4.5 External Clock Synchronization

During normal operation, independent clusters can drift significantly. If synchronous operation across independent clusters is desired, external synchronization is necessary; even though the nodes within each cluster are synchronized. This can be accomplished with synchronous application of host-deduced rate and offset correction terms to the clusters.

- External offset / rate correction value is a signed integer
- External offset / rate correction value is added to calculated offset / rate correction value
- Aggregated offset / rate correction term (external + internal) is not checked against configured limits

26.2.5 Error Handling

The implemented error handling concept of the FlexRay protocol is intended to ensure that in the presence of a lower layer protocol error in a single node, communication between non-affected nodes can be maintained. In some cases, higher layer program command activity is required for the communication controller to resume normal operation. A change of the error handling state will set bit EIR.PEMC and can trigger an interrupt to the host if enabled. The current error mode is signaled by CCEV.ERRM(1-0).

Table 26-5. Error Modes of the POC (Degradation Model)

Error Mode	Activity
ACTIVE	**Full operation**, State: NORMAL_ACTIVE The communication controller is fully synchronized and supports the cluster wide clock synchronization. The host is informed of any error condition(s) or status change by interrupt (if enabled) or by reading the error and status flags from registers EIR and SIR.
PASSIVE	**Reduced operation**, State: NORMAL_PASSIVE, communication controller self rescue allowed The communication controller stops transmitting frames and symbols, but received frames are still processed. Clock synchronization mechanisms are continued based on received frames. No active contribution to the cluster wide clock synchronization. The host is informed of any error condition(s) or status change by interrupt (if enabled) or by reading the error and status flags from registers EIR and SIR.
COMM_HALT	**Operation halted**, State: HALT, communication controller self rescue not allowed The communication controller stops frame and symbol processing, clock synchronization processing, and the macrotick generation. The host has still access to error and status information by reading the error and status flags from registers EIR and SIR. The bus drivers are disabled.

26.2.5.1 Clock Correction Failed Counter

When the Clock Correction Failed Counter reaches the “maximum without clock correction passive” limit defined by SUCC3.WCP(3-0), the POC transits from NORMAL_ACTIVE to NORMAL_PASSIVE state. When it reaches the “maximum without clock correction fatal” limit defined by SUCC3.WCF(3-0), it transits NORMAL_ACTIVE or NORMAL_PASSIVE to the HALT state.

The Clock Correction Failed Counter CCEV.CCFC(3-0) allows the host to monitor the duration of the inability of a node to compute clock correction terms after the communication controller passed protocol startup phase. It will be incremented by 1 at the end of any odd numbered communication cycle where either the Missing Offset Correction flag SFS.MOCS or the Missing Rate Correction flag SFS.MRCS is set.

The clock correction failed counter is reset to 0 at the end of an odd communication cycle if neither the Missing Offset Correction flag SFS.MOCS nor the Missing Rate Correction flag SFS.MRCS is set.

The Clock Correction Failed Counter stops incrementing when the “maximum without clock correction fatal” value SUCC3.WCF(3-0) is reached (incrementing the counter at its maximum value will not cause it to wraparound back to 0). The clock correction failed counter will be initialized to 0 when the communication controller enters READY state or when NORMAL_ACTIVE state is entered.

NOTE: The transition to HALT state is prevented if SUCC1.HCSE is not set.

26.2.5.2 Passive to Active Counter

The passive to active counter controls the transition of the POC from NORMAL_PASSIVE to NORMAL_ACTIVE state. SUCC1.SUCC1.PTA(4-0) defines the number of consecutive even / odd cycle pairs that must have valid clock correction terms before the communication controller is allowed to transit from NORMAL_PASSIVE to NORMAL_ACTIVE state. If SUCC1.PTA(4-0) is cleared to 0, the communication controller is not allowed to transit from NORMAL_PASSIVE to NORMAL_ACTIVE state.

26.2.5.3 HALT Command

In case the host wants to stop FlexRay communication of the local node it can bring the communication controller into HALT state by asserting the HALT command. This can be done by writing SUCC1.CMD(3-0) = 0110. In order to shut down communication on an entire FlexRay network, a higher layer protocol is required to assure that all nodes apply the HALT command at the same time.

The POC state from which the transition to HALT state took place can be read from CCSV.PSL(5-0).

When called in NORMAL_ACTIVE or NORMAL_PASSIVE state the POC transits to HALT state at the end of the current cycle. When called in any other state SUCC1.CMD(3-0) will be reset to 0000 = “command_not_accepted” and bit EIR.CNA in the error interrupt register is set to 1. If enabled an interrupt to the host is generated.

26.2.5.4 FREEZE Command

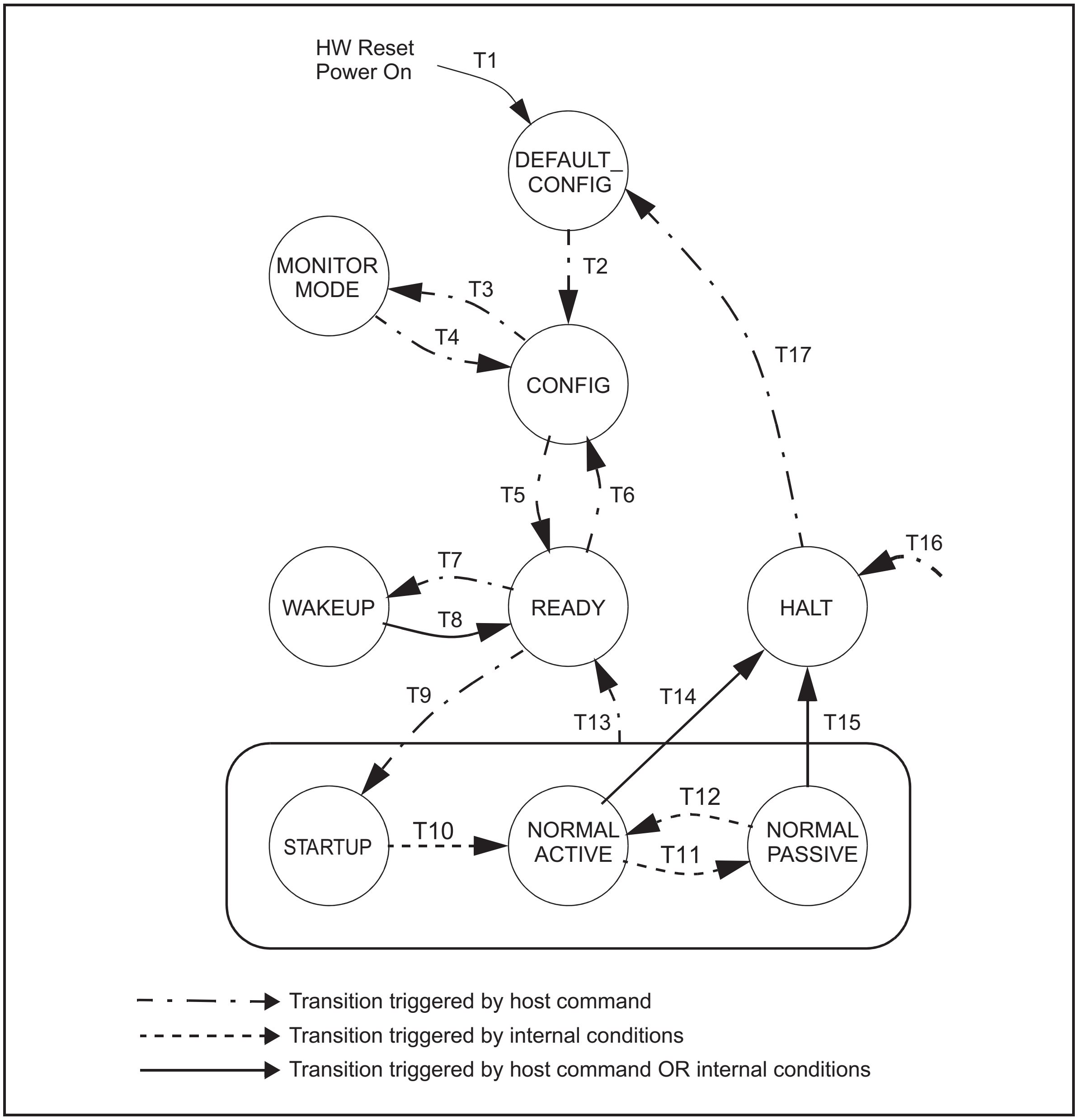
In case the host detects a severe error condition it can bring the communication controller into HALT state by asserting the FREEZE command. This can be done by writing SUCC1.CMD(3-0) = 0111. The FREEZE command triggers the entry of the HALT state immediately regardless of the current POC state.

The POC state from which the transition to HALT state took place can be read from CCSV.PSL(5-0).

26.2.6 Communication Controller States

26.2.6.1 Communication Controller State Diagram

Figure 26-12. Overall State Diagram of Communication Controller


HW Reset
Power On
T1
DEFAULT_
CONFIG
MONITOR
MODE
T2
T3
T4
CONFIG
T17
T5
T6
T7
T16
WAKEUP
READY
HALT
T8
T9
T14
T13
T15
T12
STARTUP
T10
NORMAL
ACTIVE
T11
NORMAL
PASSIVE
Transition triggered by host command
Transition triggered by internal conditions
Transition triggered by host command OR internal conditions

State transitions are controlled by the reset and FlexRay receive (rxd1, 2) pins, the POC state machine, and by the CHI command vector SUCC1.CMD(3-0).

The Communication Controller exits from all states to HALT state after application of the FREEZE command (SUCC1.CMD(3-0) = 0111b).

Table 26-6. State Transitions of Communication Controller Overall State Machine

T#	Condition	From	To
1	Hardware reset	All States	DEFAULT_CONFIG
2	Command CONFIG, SUCC1.CMD(3-0) = 0001	DEFAULT_CONFIG	CONFIG
3	Unlock sequence followed by command MONITOR_MODE, SUCC1.CMD(3-0) = 1011	CONFIG	MONITOR_MODE
4	Command CONFIG, SUCC1.CMD(3-0) = 0001	MONITOR_MODE	CONFIG
5	Unlock sequence followed by command READY, SUCC1.CMD(3-0) = 0010	CONFIG	READY
6	Command CONFIG, SUCC1.CMD(3-0) = 0001	READY	CONFIG
7	Command WAKEUP, SUCC1.CMD(3-0) = 0011	READY	WAKEUP
8	Complete, non-aborted transmission of wakeup pattern OR received WUP OR received frame header OR command READY, SUCC1.CMD(3-0) = 0010	WAKEUP	READY
9	Command RUN, SUCC1.CMD(3-0) = 0100	READY	STARTUP
10	Successful startup	STARTUP	NORMAL_ACTIVE
11	Clock correction failed counter reached “maximum without clock correction passive” limit configured by SUCC3.WCP(3-0)	NORMAL_ACTIVE	NORMAL_PASSIVE
12	Number of valid correction terms reached the Passive to Active limit configured by SUCC1.PTA(4-0)	NORMAL_PASSIVE	NORMAL_ACTIVE
13	Command READY, SUCC1.CMD(3-0) = 0010	STARTUP, NORMAL_ACTIVE,NORMAL_PASSIVE	READY
14	Clock Correction Failed counter reached “maximum without clock correction fatal” limit configured by SUCC3.WCF(3-0) AND bit SUCC1.HCSE set to 1 OR command HALT, SUCC1.CMD(3-0) = 0110	NORMAL_ACTIVE	HALT
15	Clock Correction Failed counter reached “maximum without clock correction fatal” limit configured by SUCC3.WCF(3-0) AND bit SUCC1.HCSE set to 1 OR command HALT, SUCC1.CMD(3-0) = 0110	NORMAL_PASSIVE	HALT
16	Command FREEZE, SUCC1.CMD(3-0) = 0111	All States	HALT
17	Command CONFIG, SUCC1.CMD(3-0) = 0001	HALT	DEFAULT_CONFIG

26.2.6.2 DEFAULT_CONFIG State

In DEFAULT_CONFIG state, the communication controller is stopped. All configuration registers are accessible and the pins to the physical layer are in their inactive state.

The communication controller enters this state:

- When leaving hardware reset
- When exiting from HALT state

To leave DEFAULT_CONFIG state the host has to write SUCC1.CMD(3-0) = 0001. The communication controller then transits to CONFIG state.

26.2.6.3 CONFIG State

In CONFIG state, the communication controller is stopped. All configuration registers are accessible and the pins to the physical layer are in their inactive state. This state is used to initialize the communication controller configuration.

The communication controller enters this state:

- When exiting from DEFAULT_CONFIG state
- When exiting from MONITOR_MODE or READY state

When the state has been entered by HALT and DEFAULT_CONFIG state, the host can analyze status information and configuration. Before leaving CONFIG state the host has to assure that the configuration is fault-free.

To leave CONFIG state, the host has to perform the unlock sequence. Directly after unlocking the CONFIG state the host has to write SUCC1.CMD(3-0) to enter the next state.

NOTE: The message buffer status registers (MHDS, TXRQ1/2/3/4, NDAT1/2/3/4, MBSC1/2/3/4) and status data stored in the message RAM and are not affected by the transition of the POC from CONFIG to READY state.

When the communication controller is in CONFIG state it is also possible to bring the communication controller into a power saving mode by halting the module clocks. To do this the host has to assure that all Message RAM transfers have finished before turning off the clocks.

26.2.6.4 MONITOR_MODE

After unlocking CONFIG state and writing SUCC1.CMD(3-0) = 1011 the communication controller enters MONITOR_MODE. In this mode the communication controller is able to receive FlexRay frames and to detect wakeup pattern. The temporal integrity of received frames is not checked, and therefore cycle counter filtering is not supported. This mode can be used for debugging purposes in case e.g. that startup of a FlexRay network fails. After writing SUCC1.CMD(3-0) = 0001 the communication controller transits back to CONFIG state.

In MONITOR_MODE the pick first valid mechanism is disabled. This means that a receive message buffer may only be configured to receive on one channel. Received frames are stored into message buffers according to frame ID and receive channel. Null frames are handled like data frames. After frame reception only status bits MBS.VFRA, MBS.VFRB, MBS.MLST, MBS.RCIS, MBS.SFIS, MBS.SYNS, MBS.NFIS, MBS.PPIS, MBS.RESS have valid values.

In MONITOR_MODE the communication controller is not able to distinguish between CAS and MTS symbols. In case one of these symbols is received on one or both of the two channels, the flags SIR.MTSA/SIR.MTSB are set. SIR.CAS has no function in MONITOR_MODE.

26.2.6.5 READY State

After unlocking CONFIG state and writing SUCC1.CMD(3-0) = 0010 the communication controller enters READY state. From this state the communication controller can transit to WAKEUP state and perform a cluster wakeup or to STARTUP state to perform a coldstart or to integrate into a running cluster.

The communication controller enters this state:

- When exiting from CONFIG, WAKEUP, STARTUP, NORMAL_ACTIVE, or NORMAL_PASSIVE state by writing SUCC1.CMD(3-0) = 0010 (READY command).

The communication controller exits from this state:

- To CONFIG state by writing SUCC1.CMD(3-0) = 0001 (CONFIG command)
- To WAKEUP state by writing SUCC1.CMD(3-0) = 0011 (WAKEUP command)
- To STARTUP state by writing SUCC1.CMD(3-0) = 0100 (RUN command)

Internal counters and the communication controller status flags are reset when the communication controller enters STARTUP state.

NOTE: Status bits MHDS(14-0), registers TXRQ1/2/3/4, and status data stored in the Message RAM are not affected by the transition of the POC from READY to STARTUP state.

26.2.6.6 WAKEUP State

The following description is intended to help configuring wakeup for the FlexRay module. A detailed description of the wakeup procedure can be found in the FlexRay protocol specification v2.1 Rev. A.

The communication controller enters this state:

- When exiting from READY state by writing SUCC1.CMD(3-0) = 0011 (WAKEUP command).

The communication controller exits from this state to READY state:

- After complete non-aborted transmission of wakeup pattern
- After WUP reception
- After detecting a WUP collision
- After reception of a frame header
- By writing SUCC1.CMD(3-0) = 0010 (READY command)

The communication controller exits from this state to HALT state:

- By writing SUCC1.CMD(3-0) = 0111 (FREEZE command)

The cluster wakeup must precede the communication startup in order to ensure that all nodes in a cluster are awake. The minimum requirement for a cluster wakeup is that all bus drivers are supplied with power. A bus driver has the ability to wake up the other components of its node when it receives a wakeup pattern on its channel. At least one node in the cluster needs an external wakeup source.

The host completely controls the wakeup procedure. It is informed about the state of the cluster by the bus driver and the communication controller and configures bus guardian (if available) and communication controller to perform the cluster wakeup. The communication controller provides to the host the ability to transmit a special wakeup pattern on each of its available channels separately. The communication controller needs to recognize the wakeup pattern only during wakeup and startup phase.

Wakeup may be performed on only one channel at a time. The host has to configure the wakeup channel while the communication controller is in CONFIG state by writing bit WUCS in the SUC configuration register 1. The communication controller ensures that ongoing communication on this channel is not disturbed. The communication controller cannot guarantee that all nodes connected to the configured channel awake upon the transmission of the wakeup pattern, since these nodes cannot give feedback until the startup phase. The wakeup procedure enables single-channel devices in a two-channel system to trigger the wakeup, by only transmitting the wakeup pattern on the single channel to which they are connected. Any coldstart node that deems a system startup necessary will then wake the remaining channel before initiating communication startup.

The wakeup procedure tolerates any number of nodes simultaneously trying to wakeup a single channel and resolves this situation such that only one node transmits the pattern. Additionally the wakeup pattern is collision resilient, so even in the presence of a fault causing two nodes to simultaneously transmit a wakeup pattern, the resulting collided signal can still wake the other nodes.

After wakeup the communication controller returns to READY state and signals the change of the wakeup status to the host by setting bit SIR.WST in the status interrupt register. The wakeup status vector CCSV.WSV(2-0) can be read from the communication controller status vector register. If a valid wakeup pattern was received also either bit SIR.WUPA or bit SIR.WUPB in the status interrupt register is set.

Figure 26-13. Structure of POC State WAKEUP

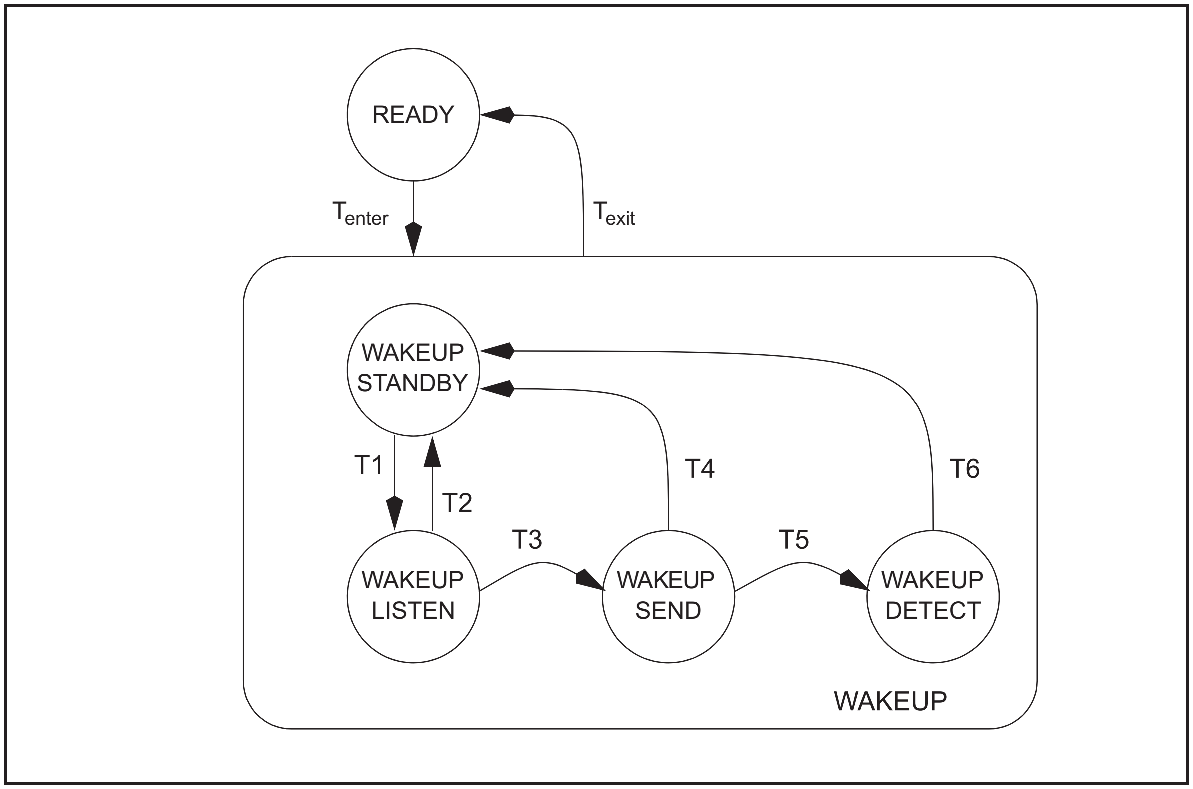


Table 26-7. State Transitions WAKEUP

T#	Condition	From	To
enter	Host commands change to WAKEUP state by writing SUCC1.CMD(3-0) = 0011 (WAKEUP command)	READY	WAKEUP
1	CHI command WAKEUP triggers wakeup FSM to transit to WAKEUP_LISTEN state	WAKEUP_STANDBY	WAKEUP_LISTEN
2	Received WUP on wakeup channel selected by bit SUCC1.WUCS OR frame header on either available channel	WAKEUP_LISTEN	WAKEUP_STANDBY
3	Timer event	WAKEUP_LISTEN	WAKEUP_SEND
4	Complete, non-aborted transmission of wakeup pattern	WAKEUP_SEND	WAKEUP_STANDBY
5	Collision detected	WAKEUP_SEND	WAKEUP_DETECT
6	Wakeup timer expired OR WUP detected on wakeup channel selected by bit SUCC1.WUCS OR frame header received on either available channel	WAKEUP_DETECT	WAKEUP_STANDBY
exit	Wakeup completed (after T2 or T4 or T6) OR host commands change to READY state by writing SUCC1.CMD(3-0) = 0010 (READY command). This command also resets the wakeup FSM to WAKEUP_STANDBY state.	WAKEUP	READY

The WAKEUP_LISTEN state is controlled by the wakeup timer and the wakeup noise timer. The two timers are controlled by the parameters Listen Timeout SUCC2.LT(20-0) and Listen Timeout Noise SUCC2.LTN(3-0). Listen timeout enables a fast cluster wakeup in case of a noise free environment, while listen timeout noise enables wakeup under more difficult conditions regarding noise interference.

In WAKEUP_SEND state the communication controller transmits the wakeup pattern on the configured channel and checks for collisions. After return from wakeup the host has to bring the communication controller into STARTUP state by CHI command RUN.

In WAKEUP_DETECT state the communication controller attempts to identify the reason for the wakeup collision detected in WAKEUP_SEND state. The monitoring is bounded by the expiration of listen timeout as configured by SUCC2.LT(20-0) in the SUC configuration register 2. Either the detection of a wakeup pattern indicating a wakeup attempt by another node or the reception of a frame header indication existing communication, causes the direct transition to READY state. Otherwise WAKEUP_DETECT is left after expiration of listen timeout; in this case the reason for wakeup collision is unknown.

The host has to be aware of possible failures of the wakeup and act accordingly. It is advisable to delay any potential startup attempt of the node having instigated the wakeup by the minimal time it takes another coldstart node to become awake and to be configured.

The FlexRay Protocol Specification recommends that two different communication controllers shall wake the two channels.

26.2.6.6.1 Host Activities

The host must coordinate the wakeup of the two channels and must decide whether, or not, to wake a specific channel. The sending of the wakeup pattern is initiated by the host. The wakeup pattern is detected by the remote BDs and signaled to their local hosts.

Wakeup procedure controlled by host (single-channel wakeup):

- Configure the communication controller in CONFIG state
  - Select wakeup channel by programming bit SUCC1.WUCS
- Check local BDs whether a WUP was received
- Activate BD of selected wakeup channel
- Command communication controller to enter READY state
- Command communication controller to start wakeup on the configured channel by writing SUCC1.CMD(3-0) = 0011
  - communication controller enters WAKEUP_LISTEN
  - communication controller returns to READY state and signals status of wakeup attempt to host
- Wait predefined time to allow the other nodes to wakeup and configure themselves
- Coldstart node:
  - in dual channel cluster wait for WUP on the other channel
  - Reset Coldstart Inhibit flag CCSV.CSI by writing SUCC1.CMD(3-0) = 1001 (ALLOW_COLDSTART command)
- Command communication controller to enter startup by writing SUCC1.CMD(3-0) = 0100 (RUN command)

Wakeup procedure triggered by the bus driver:

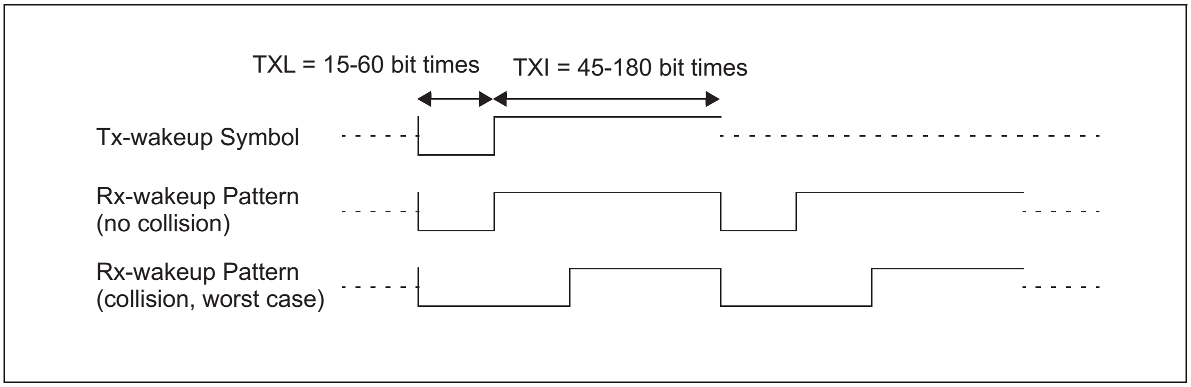
- Wakeup recognized by bus driver
- bus driver triggers power-up of host (if required)
- bus driver signals wakeup event to host
- Host configures its local communication controller
- If necessary host commands wakeup of second channel and waits predefined time to allow the other nodes to wakeup and configure themselves
- Host commands communication controller to enter STARTUP state by writing SUCC1.CMD(3-0) = 0100 (RUN command)

26.2.6.6.2 Wake Up Pattern (WUP)

The wake up pattern (WUP) is composed of at least two wakeup symbols (WUS). Wakeup symbol and wakeup pattern are configured by the PRT configuration registers 1,2.

- Single channel wakeup, wake up symbol may not be sent on both channels at the same time
- Wakeup symbol collision resilient for up to two sending nodes (two overlapping wakeup symbols still recognizable)
- Wakeup symbol must be configured identical in all nodes of a cluster
- Wakeup symbol transmit low time configured by PRTC2.TXL(5-0)
- Wakeup symbol idle time configured by PRTC2.TXI(7-0), used to listen for activity on the bus
- A wakeup pattern composed of at least two Tx-wakeup symbols needed for wakeup
- Number of repetitions configurable by PRTC1.RWP(5-0) (2 to 63 repetitions)
- Wakeup symbol receive window length configured by PRTC1.RXW(8-0)
- Wakeup symbol receive low time configured by PRTC2.RXL(5-0)
- Wakeup symbol receive idle time configured by PRTC2.RXI(5-0)

Figure 26-14. Timing of Wake Up Pattern



26.2.6.7 STARTUP State

The following description is intended to help configuring startup for the FlexRay module. A detailed description of the startup procedure can be found in the FlexRay protocol specification v2.1 Rev. A.

Any node entering STARTUP state that has coldstart capability should assure that both channels attached have been awakened before initiating coldstart.

It cannot be assumed that all nodes and stars need the same amount of time to become completely awake and to be configured. Since at least two nodes are necessary to start up the cluster communication, it is advisable to delay any potential startup attempt of the node having instigated the wakeup by the minimal amount of time it takes another coldstart node to become awake, to be configured and to enter startup. It may require several hundred milliseconds (depending on the hardware used) before all nodes and stars are completely awakened and configured.

Startup is performed on all channels synchronously. During startup, a node only transmits startup frames. Startup frames are both sync frames and null frames during startup.

A fault-tolerant, distributed startup strategy is specified for initial synchronization of all nodes. In general, a node may enter NORMAL_ACTIVE state by:

- Coldstart path initiating the schedule synchronization (leading coldstart node)
- Coldstart path joining other coldstart nodes (following coldstart node)
- Integration path integrating into an existing communication schedule (all other nodes)

See also Figure 26-15 for more information.

A coldstart attempt begins with the transmission of a collision avoidance symbol (CAS). Only a coldstart node that transmitted the CAS, transmits frames in the first four cycles after the CAS. it is then joined firstly by the other coldstart nodes and afterwards by all other nodes.

A coldstart node has bits SUCC1.TXST and SUCC1.TXSY set to 1. Message buffer 0 holds the key slot ID which defines the slot number where the startup frame is sent. The startup frame indicator bit is set in the frame header of the startup frame.

In clusters consisting of three or more nodes, at least three nodes shall be configured to be coldstart nodes. In clusters consisting of two nodes, both nodes must be coldstart nodes. At least two fault-free coldstart nodes are necessary for the cluster to startup.

Each startup frame must also be a sync frame; therefore each coldstart node will also be a sync node. The number of coldstart attempts is configured by SUCC1.CSA(4-0) in the SUC configuration register 1.

A non-coldstart node requires at least two startup frames from distinct nodes for integration. It may start integration before the coldstart nodes have finished their startup. It will not finish its startup until at least two coldstart nodes have finished their startup.

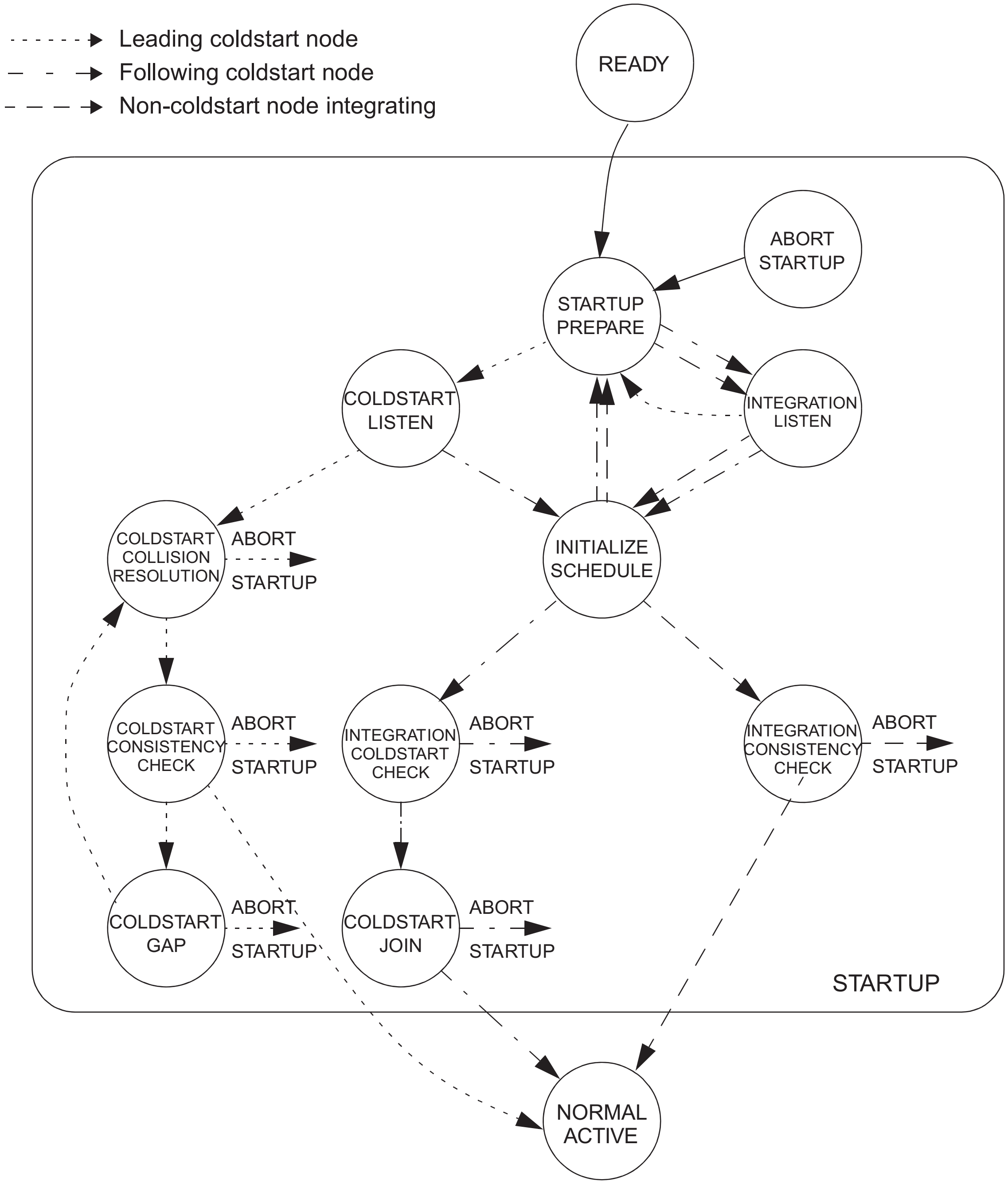
Both non-coldstart nodes and coldstart nodes start passive integration through the integration path as soon as they receive sync frames from which to derive the TDMA schedule information. During integration the node has to adapt its own clock to the global clock (rate and offset) and has to make its cycle time consistent with the global schedule observable at the network. Afterwards, these settings are checked for consistency with all available network nodes. The node can only leave the integration phase and actively participate in communication when these checks are passed.

26.2.6.7.1 Coldstart Inhibit Mode

In coldstart inhibit mode, the node is prevented from initializing the TDMA communication schedule. If the CCSV.CSI bit in the communication controller status vector register is set, the node is not allowed to initialize the cluster communication, that is, entering the coldstart path is prohibited. The node is allowed to integrate to a running cluster or to transmit startup frames after another coldstart node starts the initialization of the cluster communication.

The coldstart inhibit bit CCSV.CSI is set whenever the POC enters READY state. The bit has to be cleared under control of the host by CHI command ALLOW_COLDSTART (SUCC1.CMD(3-0) = 1001).

Figure 26-15. State Diagram Time-Triggered Startup



26.2.6.7.2 Startup Timeouts

The communication controller supplies two different μT timers supporting two timeout values, startup timeout and startup noise timeout. The two timers are started when the communication controller enters the COLDSTART_LISTEN state. The expiration of either of these timers causes the node to leave the initial sensing phase (COLDSTART_LISTEN state) with the intention of starting up communication.

NOTE: The startup and startup noise timers are identical with the wakeup and wakeup noise timers and use the same configuration values SUCC2.LT(20-0) and SUCC2.LTN(3-0) from the SUC configuration register 2.

26.2.6.7.2.1 Startup Timeout

The startup timeout limits the listen time used by a node to determine if there is already communication between other nodes or at least one coldstart node actively requesting the integration of others. The startup timer is configured by programming SUCC2.LT(20-0) in the SUC configuration register 2.

The startup timeout time can be calculated from the contents of SUCC2.LT(20-0) (Refer to the FlexRay Protocol Specification: pdListenTimeout)

The startup timer is restarted upon:

- Entering the COLDSTART_LISTEN state
- Both channels reaching idle state while in COLDSTART_LISTEN state

The startup timer is stopped:

- If communication channel activity is detected on one of the configured channels while the node is in the COLDSTART_LISTEN state
- When the COLDSTART_LISTEN state is left

Once the startup timeout expires, neither an overflow nor a cyclic restart of the timer is performed. The timer status is kept for further processing by the startup state machine.

26.2.6.7.2.2 Startup Noise Timeout

At the same time the startup timer is started for the first time (transition from STARTUP_PREPARE state to COLDSTART_LISTEN state), the startup noise timer is started. This additional timeout is used to improve reliability of the startup procedure in the presence of noise. The startup noise timer is configured by programming SUCC2.LTN(3-0) in the SUC configuration register 2.

The startup noise timeout time can be calculated as the product of SUCC2.LT(20-0) * SUCC2.LTN(3-0) (Refer to the FlexRay Protocol Specification: pdListenTimeout • gListenNoise)

The startup noise timer is restarted upon:

- Entering the COLDSTART_LISTEN state
- Reception of correctly decoded headers or CAS symbols while the node is in COLDSTART_LISTEN state

The startup noise timer is stopped when the COLDSTART_LISTEN state is left.

Once the startup noise timeout expires, neither an overflow nor a cyclic restart of the timer is performed. The status is kept for further processing by the startup state machine. Since the startup noise timer won’t be restarted when random channel activity is sensed, this timeout defines the fall-back solution that guarantees that a node will try to start up the communication cluster even in the presence of noise.

26.2.6.7.3 Path of Leading Coldstart Node (Initiating Coldstart)

When a coldstart node enters COLDSTART_LISTEN, it listens to its attached channels.

If no communication is detected, the node enters the COLDSTART_COLLISION_RESOLUTION state and commences a coldstart attempt. The initial transmission of a CAS symbol is succeeded by the first regular cycle. This cycle has the number 0.

From cycle 0 on, the node transmits its startup frame. Since each coldstart node is allowed to perform a coldstart attempt, it may occur that several nodes simultaneously transmit the CAS symbol and enter the coldstart path. This situation is resolved during the first four cycles after CAS transmission.

As soon as a node that initiates a coldstart attempt receives a CAS symbol or a frame header during these four cycles, it re-enters the COLDSTART_LISTEN state. Thereby, only one node remains in this path. In cycle four, other coldstart nodes begin to transmit their startup frames.

After four cycles in COLDSTART_COLLISION_RESOLUTION state, the node that initiated the coldstart enters the COLDSTART_CONSISTENCY_CHECK state. It collects all startup frames from cycle four and five and performs clock correction. If the clock correction does not deliver any errors and it has received at least one valid startup frame pair, the node leaves COLDSTART_CONSISTENCY_CHECK and enters NORMAL_ACTIVE state.

The number of coldstart attempts that a node is allowed to perform is configured by SUCC1.CSA(4-0) in the SUC configuration register 1. The number of remaining coldstarts attempts can be read from CCSV.RCA(4-0) of communication controller status vector register. The number of remaining attempts is reduced by 1 for each attempted coldstart. A node may enter the COLDSTART_LISTEN state only if this value is larger than 1 and it may enter the COLDSTART_COLLISION_RESOLUTION state only if this value is larger than 0. If the number of coldstart attempts is 1, coldstart is inhibited but integration is still possible.

26.2.6.7.4 Path of Following Coldstart Node (Responding to Leading Coldstart Node)

When a coldstart node enters the COLDSTART_LISTEN state, it tries to receive a valid pair of startup frames to derive its schedule and clock correction from the leading coldstart node.

As soon as a valid startup frame has been received, the INITIALIZE_SCHEDULE state is entered. If the clock synchronization can successfully receive a matching second valid startup frame and derive a schedule from this, the INTEGRATION_COLDSTART_CHECK state is entered.

In INTEGRATION_COLDSTART_CHECK state, it is assured that the clock correction can be performed correctly and that the coldstart node from which this node has initialized its schedule is still available. The node collects all sync frames and performs clock correction in the following double-cycle. If clock correction does not signal any errors and if the node continues to receive sufficient frames from the same node it has integrated on, the COLDSTART_JOIN state is entered.

In COLDSTART_JOIN state, following coldstart nodes begin to transmit their own startup frames and continue to do so in subsequent cycles. Thereby, the leading coldstart node and the nodes joining it can check if their schedules agree with each other. If the clock correction signals any error, the node aborts the integration attempt. If a node in this state sees at least one valid startup frame during all even cycles in this state and at least one valid startup frame pair during all double cycles in this state, the node leaves COLDSTART_JOIN state and enters NORMAL_ACTIVE state. Thereby it leaves STARTUP at least one cycle after the node that initiated the coldstart.

26.2.6.7.5 Path of Non-Coldstart Node

When a non-coldstart node enters the INTEGRATION_LISTEN state, it listens to its attached channels.

As soon as a valid startup frame has been received the INITIALIZE_SCHEDULE state is entered. If the clock synchronization can successfully receive a matching second valid startup frame and derive a schedule from this, the INTEGRATION_CONSISTENCY_CHECK state is entered.

In INTEGRATION_CONSISTENCY_CHECK state it is verified that the clock correction can be performed correctly and that enough coldstart nodes (at least 2) send startup frames that agree with the nodes own schedule. Clock correction is activated, and if any errors are signaled, the integration attempt is aborted.

During the first even cycle in this state, either two valid startup frames or the startup frame of the node that this node has integrated on must be received; otherwise the node aborts the integration attempt.

During the first double-cycle in this state, either two valid startup frame pairs or the startup frame pair of the node that this node has integrated on must be received; otherwise the node aborts the integration attempt.

If after the first double-cycle less than two valid startup frames are received within an even cycle, or less than two valid startup frame pairs are received within a double-cycle, the startup attempt is aborted.

Nodes in this state need to see two valid startup frame pairs for two consecutive double-cycles each to be allowed to leave STARTUP and enter NORMAL_OPERATION. Consequently, they leave startup at least one double-cycle after the node that initiated the coldstart and only at the end of a cycle with an odd cycle number.

26.2.6.8 NORMAL_ACTIVE State

As soon as the node that transmitted the first CAS symbol (resolving the potential access conflict and entering STARTUP through the coldstart path) and one additional node have entered the NORMAL_ACTIVE state, the startup phase for the cluster has finished. In the NORMAL_ACTIVE state, all configured messages are scheduled for transmission. This includes all data frames as well as the sync frames. Rate and offset measurement is started in all even cycles (even/odd cycle pairs required).

In NORMAL_ACTIVE state the communication controller supports regular communication functions:

- The communication controller performs transmissions and reception on the FlexRay bus as configured
- Clock synchronization is running
- The host interface is operational

The communication controller exits from that state to:

- HALT state by writing SUCC1.CMD(3-0) = 0110 (HALT command, at the end of the current cycle)
- HALT state by writing SUCC1.CMD(3-0) = 0111 (FREEZE command, immediately)
- HALT state due to change of the error state from ACTIVE to COMM_HALT
- NORMAL_PASSIVE state due to change of the error state from ACTIVE to PASSIVE
- READY state by writing SUCC1.CMD(3-0) = 0010 (READY command)

26.2.6.9 NORMAL_PASSIVE State

NORMAL_PASSIVE state is entered from NORMAL_ACTIVE state when the error state changes from ACTIVE to PASSIVE.

In NORMAL_PASSIVE state, the node is able to receive all frames (node is fully synchronized and performs clock synchronization). Contrary to the NORMAL_ACTIVE state, the node does not actively participate in communication, that is, neither symbols nor frames are transmitted.

In NORMAL_PASSIVE state:

- The communication controller performs reception on the FlexRay bus
- The communication controller does not transmit any frames or symbols on the FlexRay bus
- Clock synchronization is running
- The host interface is operational

The communication controller exits from this state to

- HALT state by writing SUCC1.CMD(3-0) = 0110 (HALT command, at the end of the current cycle)
- HALT state by writing SUCC1.CMD(3-0) = 0111 (FREEZE command, immediately)
- HALT state due to change of the error state from PASSIVE to COMM_HALT
- NORMAL_ACTIVE state due to change of the error state from PASSIVE to ACTIVE. The transition takes place when CCEV.PTAC(4-0) equals SUCC1.PTA(4-0) - 1.
- To READY state by writing SUCC1.CMD(3-0) = 0010 (READY command)

26.2.6.10 HALT State

In this state all communication (reception and transmission) is stopped.

The communication controller enters this state:

- By writing SUCC1.CMD(3-0) = 0110 (HALT command) while the communication controller is in NORMAL_ACTIVE or NORMAL_PASSIVE state

- By writing SUCC1.CMD(3-0) = 0111 (FREEZE command) from all states
- When exiting from NORMAL_ACTIVE state because the clock correction failed counter reached the “maximum without clock correction fatal” limit and SUCC1.HCSE is set
- When exiting from NORMAL_PASSIVE state because the clock correction failed counter reached the “maximum without clock correction fatal” limit and SUCC1.HCSE is set

The communication controller exits from this state to DEFAULT_CONFIG state

- By writing SUCC1.CMD(3-0) = 0001 (CONFIG command)

When the communication controller transits from HALT state to DEFAULT_CONFIG state all configuration and status data is maintained for analyzing purposes.

When the host writes SUCC1.CMD(3-0) = 0110 (HALT command), the communication controller sets bit CCSV.HRQ and enters HALT state at next end of cycle.

When the host writes SUCC1.CMD(3-0) = 0111 (FREEZE command), the communication controller enters HALT state immediately and sets the CCSV.FSI bit in the communication controller status vector register.

The POC state from which the transition to HALT state took place can be read from CCSV.PSL(5-0).

26.2.7 Network Management

The accrued network management (NM) vector is located in the Network Management Registers (NMV1/2/3). The communication controller performs a logical OR operation over all NM vectors out of all received valid NM frames with the Payload Preamble Indicator (PPI) bit set. Only a static frame may be configured to hold NM information. The communication controller updates the NM vector at the end of each cycle.

The length of the NM vector can be configured from 0 to 12 bytes by NEMC.NML(3-0). The NM vector length must be configured identically in all nodes of a cluster.

To configure a transmit buffer to send FlexRay frames with the PPI bit set, the PPIT bit in the header section of the corresponding transmit buffer has to be set WRHS1.PPIT. In addition the host has to write the NM information to the data section of the corresponding transmit buffer.

The evaluation of the NM vector has to be done by the application running on the host.

NOTE: In case a message buffer is configured for transmission / reception of network management frames, the payload length configured in header 2 of that message buffer should be equal or greater than the length of the NM vector configured by NEMC.NML(3-0).When the Communication Controller transits to HALT state, the cycle count is not incremented and therefore the NM vector is not updated. In this case NMV1/2/3 holds the value from the cycle before.

26.2.8 Filtering and Masking

Filtering is done by comparison of the configuration of assigned message buffers against current slot and cycle counter values and channel ID (channel A, B). A message buffer is only updated / transmitted if the required matches occur.

Filtering is done on:

- Slot counter
- Cycle counter
- Channel ID

The following filter combinations for acceptance / transmit filtering are allowed:

- Slot counter + Channel ID
- Slot counter + Cycle counter + Channel ID

All configured filters must match in order to store a received message in a message buffer.

NOTE: For the FIFO the acceptance filter is configured by the FIFO Rejection Filter and the FIFO Rejection Filter mask.

A message will be transmitted in the time slot corresponding to the configured frame ID on the configured channel(s). If cycle counter filtering is enabled the configured cycle filter value must also match.

26.2.8.1 Slot Counter Filtering

Every transmit and receive buffer contains a frame ID stored in the header section. This frame ID is compared against the current slot counter value in order to assign receive and transmit buffers to the corresponding slot.

If two or more message buffers are configured with the same frame ID and channel ID, and if they have a matching cycle counter filter value for the same slot, then the message buffer with the lowest message buffer number is used.

26.2.8.2 Cycle Counter Filtering

Cycle counter filtering is based on the notion of a cycle set. For filtering purposes, a match is detected if any one of the elements of the cycle set is matched. The cycle set is defined by the cycle code field in the header section 1 of each message buffer.

If message buffer 0 or 1 is configured to hold the startup / sync frame or the single slot frame by bits TXST, TXSY, and TSM of SUC Configuration Register 1, cycle counter filtering for message buffer 0 or 1 respectively shall be disabled.

NOTE: Sharing of a static time slot by cycle counter filtering between different nodes of a FlexRay network is not allowed.

The set of cycle numbers belonging to a cycle set is determined as described in Table 26-8.

Table 26-8. Definition of Cycle Set

Cycle Code	Matching Cycle Counter Values		
0b000000x	All cycles		
0b000001***c***	Every second cycle	at (cycle count)mod2	= ***c***
0b00001***cc***	Every fourth cycle	at (cycle count)mod4	= ***cc***
0b0001***ccc***	Every eighth cycle	at (cycle count)mod8	= ***ccc***
0b001***cccc***	Every sixteenth cycle	at (cycle count)mod16	= ***cccc***
0b01***ccccc***	Every thirty-second cycle	at (cycle count)mod32	= ***ccccc***
0b1***cccccc***	Every sixty-fourth cycle	at (cycle count)mod64	= ***cccccc***

Table 26-9 gives some examples for valid cycle sets to be used for cycle counter filtering.

Table 26-9. Examples for Valid Cycle Sets

Cycle Code	Matching Cycle Counter Values
0b00000**1*1***	1-3-5-7- …. -63 ↲
0b0000**1*00***	0-4-8-12- …. -60 ↲
0b000**1*110***	6-14-22-30- …. -62 ↲
0b00**1*1000***	8-24-40-56 ↲
0b0**1*00011***	3-35 ↲
0b**1*001001***	9 ↲

The received message is stored only if the cycle counter value of the cycle during which the message is received matches an element of the receive buffer's cycle set. Other filter criteria must also be met.

The content of a transmit buffer is transmitted on the configured channel(s) when an element of the cycle set matches the current cycle counter value. Other filter criteria must also be met.

26.2.8.3 Channel ID Filtering

There is a 2-bit channel filtering field (CHA, CHB) located in the header section of each message buffer in the message RAM. It serves as a filter for receive buffers, and as a control field for transmit buffers (see Table 26-10).

Table 26-10. Channel Filtering Configuration

		Transmit Buffer	Receive Buffer
CHA	CHB	Transmit frame	Store valid receive frame
1	1	On both channels (static segment only)	Received on channel A or B (store first semantically valid frame, static segment only)
1	0	On channel A	Received on channel A
0	1	On channel B	Received on channel B
0	0	No transmission	Ignore frame

The contents of a transmit buffer is transmitted on the channels specified in the channel filtering field when the slot counter filtering and cycle counter filtering criteria are also met. Only in static segment a transmit buffer may be set up for transmission on both channels (CHA and CHB set).

Valid received frames are stored if they are received on the channels specified in the channel filtering field when the slot counter filtering and cycle counter filtering criteria are also met. Only in static segment a receive buffer may be setup for reception on both channels (CHA and CHB set).

NOTE: If a message buffer is configured for the dynamic segment and both bits of the channel filtering field are set to 1, no frames are transmitted and received frames are ignored (same function as CHA = CHB = 0)

26.2.8.4 FIFO Filtering

For FIFO filtering there is one rejection filter and one rejection filter mask available. The FIFO filter consists of channel filter FRF.CH(1-0), frame ID filter FRF.FID(10-0), and cycle counter filter FRF.CYF(6-0). Registers FRF and FRFM can be configured in DEFAULT_CONFIG or CONFIG state only. The filter configuration in the header section of message buffers belonging to the FIFO is ignored.

The 7-bit cycle counter filter determines the cycle set to which frame ID and channel rejection filter are applied. In cycles not belonging to the cycle set specified by FRF.CYF(6-0), all frames are rejected.

A valid received frame is stored in the FIFO if channel ID, frame ID, and cycle counter are not rejected by the configured rejection filter and rejection filter mask, and if there is no matching dedicated receive buffer.

26.2.9 Transmit Process

26.2.9.1 Static Segment

For the static segment, if there are several messages pending for transmission, the message with the frame ID corresponding to the next sending slot is selected for transmission.

The data section of transmit buffers assigned to the static segment can be updated until the end of the preceding time slot. This means that a transfer from the input buffer has to be started by writing to the Input Buffer Command Request Register latest at this time.

26.2.9.2 Dynamic Segment

In the dynamic segment, if several messages are pending, the message with the highest priority (lowest frame ID) is selected next. In the dynamic segment different slot counter sequences on channel A and channel B are possible (concurrent sending of different frame IDs on both channels).

The data section of transmit buffers assigned to the dynamic segment can be updated until the end of the preceding slot. This means that a transfer from the input buffer has to be started by writing to the Input Buffer Command Request Register latest at this time.

The start of latest transmit configured by MHDC.SLT(12-0) in the MHD configuration register 1 defines the maximum minislot value allowed before inhibiting new frame transmission in the dynamic segment of the current cycle.

26.2.9.3 Transmit Buffers

Communication Controller message buffers can be configured as transmit buffers by programming bit CFG in the header section of the corresponding message buffer to 1 in WRHS1.

There exist the following possibilities to assign a transmit buffer to the communication controller channels:

- Static segment:
  - channel A or channel B
  - channel A and channel B
- Dynamic segment:
  - channel A or channel B

Message buffer 0 or 1 is dedicated to hold the startup frame, the sync frame, or the designated single slot frame as configured by SUCC1.TXST, SUCC1.TXSY, and SUCC1.TSM in the SUC Configuration register 1. In this case it can be reconfigured in DEFAULT_CONFIG or CONFIG state only. This ensures that any node transmits at most one startup / sync frame per communication cycle. Transmission of startup / sync frames from other message buffers is not possible.

All other message buffers configured for transmission in static or dynamic segment are reconfigurable during runtime depending on the configuration of MRC.SEC(1-0). Due to the organization of the data partition in the message RAM (reference by data pointer), reconfiguration of the configured payload length and the data pointer in the header section of a message buffer may lead to erroneous configurations.

If a message buffer is reconfigured (header section updated) during runtime, it may happen that this message buffer is not sent out in the currently active communication cycle.

The communication controller does not have the capability to calculate the header CRC. The host is supposed to provide the header CRCs for all transmit buffers. If network management is required the host has to set the PPIT bit in the header section of the corresponding message buffer to 1 and write the network management information to the data section of the message buffer.

The payload length field configures the data payload length in 2-byte words. If the configured payload length of a static transmit buffer is shorter than the payload length configured for the static segment by MHDC.SFDL(6-0) in the message handler configuration register 1, the communication controller generates padding bytes to ensure that frames have proper physical length. The padding pattern is logical 0.

NOTE: In case of an odd payload length (PLC=1,3,5,...) the application needs to write zeros to the last 16 bit of the message buffers data section to ensure that the padding pattern is all zeros.

Each transmit buffer provides a transmission mode flag TXM that allows the host to configure the transmission mode for the transmit buffer. If this bit is set, the transmitter operates in the single-shot mode. If this bit is cleared, the transmitter operates in the continuous mode.

In single-shot mode the Communication Controller resets the corresponding TXR flag after transmission has completed after which the host may update the transmit buffer.

In continuous mode, the Communication Controller does not reset the corresponding transmission request flag TXR after successful transmission. In this case a frame is sent out each time the filter criteria match. The TXR flag can be reset by the Host by writing the corresponding message buffer number to the IBCR register while bit IBCM.STXRH is set to 0.

If two or more transmit buffers meet the filter criteria simultaneously, the transmit buffer with the lowest message buffer number will be transmitted in the corresponding slot.

26.2.9.4 Frame Transmission

The following steps are required to prepare a message buffer for transmission:

- Configure the transmit buffer in the Message RAM through WRHS1, WRHS2, and WRHS3
- Write the data section of the transmit buffer through WRDSn
- Transfer the configuration and message data from Input Buffer to the Message RAM by writing the number of the target message buffer to register IBCR
- If configured in the Input Buffer Command Mask (IBCM) register the Transmission request flag (TXR) for the corresponding message buffer will be set as soon as the transfer has completed, and the message buffer is ready for transmission.
- Check whether the message buffer has been transmitted by checking the TXR bits (TXR = 0) in the Transmission request 1,2,3,4 registers (single-shot mode only).

After transmission has completed, the corresponding TXR flag in the Transmission request 1,2,3,4 register is reset (single- shot mode), and, if bit MBI in the header section of the message buffer is set, flag SIR.TXI in the Status Interrupt register is set to 1. If enabled, an interrupt is generated.

26.2.9.5 Null Frame Transmission

If in static segment the host does not set the transmission request flag before transmit time, and if there is no other transmit buffer with matching filter criteria, the communication controller transmits a null frame with the null frame indication bit set and the payload data cleared to 0.

In the following cases the communication controller transmits a null frame:

- If the message buffer with the lowest message buffer number matching the filter criteria does not have its transmission request flag set (TXR = 0).
- No transmit buffer configured for the slot has a cycle counter filter that matches the current cycle. In this case, no message buffer status MBS is updated.

Null frames are not transmitted in the dynamic segment.

26.2.10 Receive Process

26.2.10.1 Dedicated Receive Buffers

A portion of the Communication Controller message buffers can be configured as dedicated receive buffers by programming bit CFG in the header section of the corresponding message buffer to 0. This can be done through the Write Header Section 1 register.

The following possibilities exist to assign a receive buffer to the Communication Controller channels:

- Static segment:
  - channel A or channel B
  - channel A and channel B (the communication controller stores the first semantically valid frame)
- Dynamic segment:
  - channel A or channel B

The communication controller transfers payload data of valid received messages from the shift registers of the FlexRay protocol controller (channel A or B) to the receive buffer with the matching filter configuration. A receive buffer stores all frame elements except the frame CRC.

All message buffers configured for reception in static or dynamic segment are reconfigurable during runtime depending on the configuration of MRC.SEC(1-0) of the Message RAM Configuration register. If a message buffer is reconfigured (header section updated) during runtime it may happen that in the currently active communication cycle a received message is lost.

If two or more receive buffers meet the filter criteria simultaneously, the receive buffer with the lowest message buffer number is updated with the received message.

26.2.10.2 Frame Reception

The following steps are required to prepare a dedicated message buffer for reception:

- Configure the receive buffer in the Message RAM through WRHS1, WRHS2, and WRHS3
- Transfer the configuration from input buffer to the message RAM by writing the number of the target message buffer to the Input Buffer Command Request (IBCR) register.

Once these steps are performed, the message buffer functions as an active receive buffer and participates in the internal acceptance filtering process, which takes place every time the communication controller receives a message. The first matching receive buffer is updated from the received message.

If a valid payload segment was stored in the data section of a message buffer, the corresponding ND flag in the NDAT1,2,3,4 registers is set, and, if bit MBI in the header section of that message buffer is set, flag SIR.RXI in the Status Interrupt Register is set to 1. If enabled, an interrupt is generated.

In case that bit ND was already set when the Message Handler updates the message buffer, bit MBS.MLST of the corresponding message buffer is set and the unprocessed message data is lost.

If no frame, a null frame, or a corrupted frame is received in a slot, the data section of the message buffer configured for this slot is not updated. In this case only the flags in the corresponding message buffer status (MBS) is updated.

When the Message Handler changes the message buffer status MBS in the header section of a message buffer, the corresponding MBC flag in the Message Buffer Status Changed 1,2,3 or 4 register is set, and if bit MBI in the header section of that message buffer is set, flag SIR.MBSI in the Status Interrupt Register is set to 1. If enabled an interrupt is generated.

If the payload length of a received frame PLR(6-0) is longer than the value programmed by PLC(6-0) in the header section of the corresponding message buffer, the data field stored in the message buffer is truncated to that length.

NOTE: The ND and MBS flags are automatically cleared by the message handler when the payload data and the header of a received message have been transferred to the output buffer, respectively.

26.2.10.3 Null Frame Reception

The payload segment of a received null frame is not copied into the matching dedicated receive buffer. If a null frame has been received, only the message buffer status MBS of the matching message buffer is updated from the received null frame. All bits in header 2 and 3 of the matching message buffer remain unchanged. They are updated from received data frames only.

26.2.11 FIFO Function

26.2.11.1 Description

A group of the message buffers can be configured as a cyclic First-In-First-Out (FIFO) buffer. The group of message buffers belonging to the FIFO is contiguous in the register map starting with the message buffer referenced by MRC.FFB(7-0) and ending with the message buffer referenced by MRC.LCB(7-0) in the message RAM configuration register. Up to 128 message buffers can be assigned to the FIFO.

Every valid incoming message not matching with any dedicated receive buffer but passing the programmable FIFO filter is stored into the FIFO. In this case frame ID, payload length, receive cycle count, and the message buffer status MBS of the addressed FIFO message buffer are overwritten with frame ID, payload length, receive cycle count, and the status from the received frame. Bit SIR.RFNE in the status interrupt register shows that the FIFO is not empty, bit SIR.RFCL is set when the receive FIFO fill level FSR.RFFL(7-0) is equal or greater than the critical level as configured by FCL.CL(7-0), bit EIR.RFO shows that a FIFO overrun has been detected. If enabled, interrupts are generated.

If null frames are not rejected by the FIFO rejection filter, the null frames will be treated like data frames when they are stored into the FIFO.

There are two index registers associated with the FIFO. The PUT Index register (PIDX) is an index to the next available location in the FIFO. When a new message has been received it is written into the message buffer addressed by the PIDX register. The PIDX register is then incremented and addresses the next available message buffer. If the PIDX register is incremented past the highest numbered message buffer of the FIFO, the PIDX register is loaded with the number of the first (lowest numbered) message buffer in the FIFO chain. The GET Index register (GIDX) is used to address the next message buffer of the FIFO to be read. The GIDX register is incremented after transfer of the contents of a message buffer belonging to the FIFO to the output buffer. The PUT Index register and the GET Index register are not memory mapped and are not accessible by the host CPU.

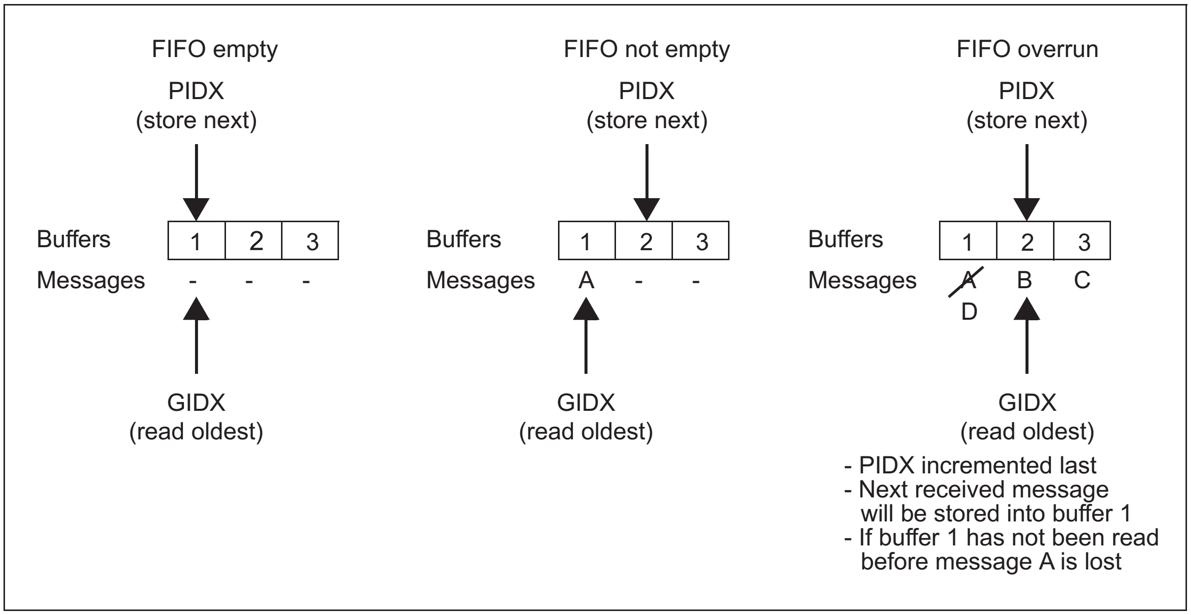
The FIFO is completely filled when the PUT index (PIDX) reaches the value of the GET index (GIDX). When the next message is written to the FIFO before the oldest message has been read, both PUT index and GET index are incremented and the new message overwrites the oldest message in the FIFO. This will set FIFO overrun flag EIR.RFO in the error interrupt register.

A FIFO not empty status is detected when the PUT index (PIDX) differs from the GET index (GIDX). In this case flag SIR.RFNE is set. This indicates that there is at least one received message in the FIFO. The FIFO empty, FIFO not empty, and the FIFO overrun states are explained in Figure 26-16 for a three message buffer FIFO.

The programmable FIFO Rejection Filter register (FRF) defines a filter pattern for messages to be rejected. The FIFO rejection filter consists of channel filter, frame ID filter, and cycle counter filter. If bit FRF.RSS is set to 1 (default), all messages received in the static segment are rejected by the FIFO. If bit FRF.RNF is set to 1 (default), received null frames are not stored in the FIFO.

The FIFO Rejection Filter mask register (FRFM) specifies which bits of the frame ID filter in the FIFO Rejection Filter register are marked don't care for rejection filtering.

Figure 26-16. FIFO Status: Empty, Not Empty, and Overrun



26.2.11.2 Configuration of the FIFO

(Re)configuration of message buffers belonging to the FIFO is only possible when the Communication Controller is in DEFAULT_CONFIG or CONFIG state. While the Communication Controller is in DEFAULT_CONFIG or CONFIG state, the FIFO function is not available.

For all message buffers belonging to the FIFO should have the same payload length configured in WRHS2.PLC(6-0) of the Write Header Section 2 register. The data pointer to the first 32-bit word in the data section of the corresponding message buffer has to be configured by WRHS3.DP(10-0).

All information required for acceptance filtering is taken from the FIFO rejection filter and the FIFO rejection filter mask. With the exception of DP and PLC, the values configured in the header sections of the message buffers belonging to the FIFO are irrelevant.

NOTE: It is recommended to program the MBI bits of the message buffers belonging to the FIFO to 0 by WRHS1.MBI to avoid RX interrupts to be generated.

If the payload length of a received frame is longer than the value programmed by WRHS2.PLC(6-0) in the header section of the corresponding message buffer, the data field stored in a message buffer of the FIFO is truncated to that length.

26.2.11.3 Access to the FIFO

For FIFO access outside DEFAULT_CONFIG and CONFIG state, the Host has to trigger a transfer from the Message RAM to the Output Buffer by writing the number of the first message buffer of the FIFO (referenced by MRC.FFB(7-0)) to the Output Buffer Command Request (OBCR) register. The message handler then transfers the message buffer addressed by the GET Index register (GIDX) to the output buffer. After this transfer the GET Index register (GIDX) is incremented.

26.2.12 Message Handling

The message handler controls data transfers between the input / output buffer and the message RAM and between the message RAM and the two transient buffer RAMs. All accesses to the internal RAMs are 32 bit accesses.

Access to the message buffers stored in the message RAM is done under control of the message handler state machine. This avoids conflicts between accesses of the two protocol controllers and the host CPU to the message RAM.

Frame IDs of message buffers assigned to the static segment have to be in the range from 1 to GTU7.NSS(9-0) as configured in the GTU configuration register 7. Frame IDs of message buffers assigned to the dynamic segment have to be in the range from GTU7.NSS(9-0) + 1 to 2047.

Received messages with no matching dedicated receive buffer (static or dynamic segment) are stored in the receive FIFO (if configured) if they pass the FIFO rejection filter.

26.2.12.1 Reconfiguration of Message Buffers

In case that an application needs to operate with more than 128 different messages, static and dynamic message buffers may be reconfigured during FlexRay operation. This is done by updating the header section of the corresponding message buffer through Input Buffer registers WRHS1,2,3.

Reconfiguration has to be enabled through control bits MRC.SEC(1-0) in the Message RAM Configuration register.

If a message buffer has not been transmitted / updated from a received frame before reconfiguration starts, the corresponding message is lost.

The point in time when a reconfigured message buffer is ready for transmission / reception according to the reconfigured frame ID depends on the current state of the slot counter when the update of the header section has completed. Therefore it may happen that a reconfigured message buffer is not transmitted / updated from a received frame in the cycle where it was reconfigured.

The Message RAM is scanned according to Table 26-11.

Table 26-11. Scan of Message RAM

Start of Scan in Slot	Scan for Slots
1	2...15, 1 (next cycle)
8	16...23, 1 (next cycle)
16	24...31, 1 (next cycle)
24	32...39, 1 (next cycle)
....	...

A Message RAM scan is terminated with the start of NIT irrespective of it’s completion. The scan of the Message RAM for slots 2 to 15 starts at the beginning of slot 1 of the current cycle. The scan of the Message RAM for slot 1 is done in the cycle before by checking in parallel to each scan of the Message RAM whether there is a message buffer configured for slot 1 of the next cycle.

The number of the first dynamic message buffer is configured by MRC.FDB(7-0) in the Message RAM Configuration register. In case a Message RAM scan starts while the Communication Controller is in dynamic segment, the scan starts with the message buffer number configured by MRC.FDB(7-0).

In case a message buffer needs to be reconfigured to be used in slot 1 of the next cycle, the following has to be considered:

- If the message buffer to be reconfigured for slot 1 is part of the Static Buffers, it will only be found if it is reconfigured before the last Message RAM scan in the static segment of the current cycle evaluates this message buffer.
- If the message buffer to be reconfigured for slot 1 is part of the Static + Dynamic Buffers, it will be found if it is reconfigured before the last Message RAM scan in the current cycle evaluates this message buffer.
- The start of NIT terminates the Message RAM scan. In case the Message RAM scan has not evaluated the reconfigured message buffer until this point in time, the message buffer will not be considered for the next cycle.

NOTE: Reconfiguration of message buffers may lead to the loss of messages and therefore has to be used very carefully. In worst case (reconfiguration in consecutive cycles) it may happen that a message buffer is never transmitted / updated from a received frame.

26.2.12.2 Host Access to Message RAM

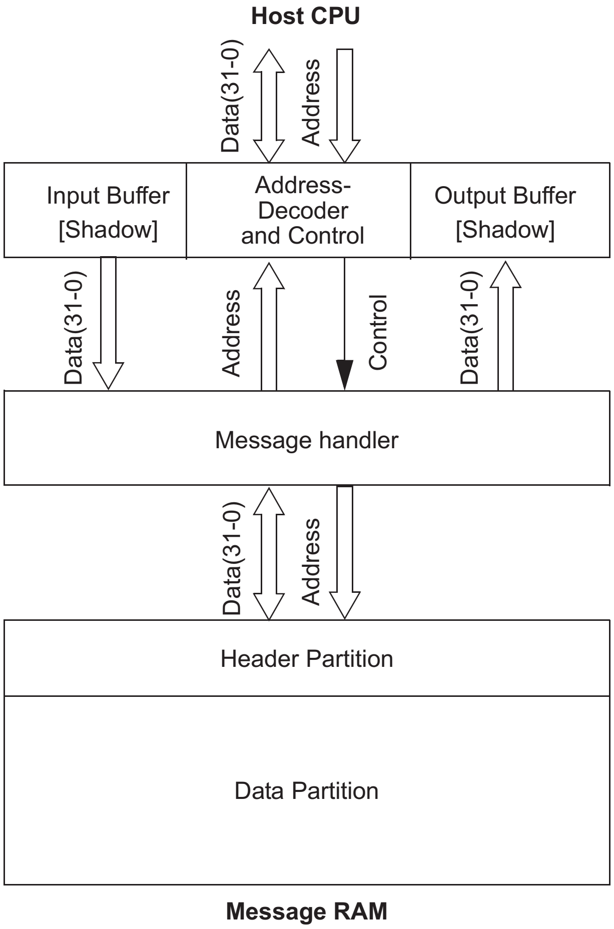
The message transfer between input buffer and message RAM as well as between message RAM and output buffer is triggered by the host CPU by writing the number of the target / source message buffer to be accessed to the input or output buffer command request register (IBCR/OBCR).

The input / output buffer command mask registers can be used to write / read header and data section of the selected message buffer separately.

If bit IBCM.STXR in the input buffer command mask register is set (STXR = 1), the transmission request flag TXR of the selected message buffer is automatically set after the message buffer has been updated. If bit IBCM.STXR in the input buffer command mask register is reset (STXR = 0), the transmission request flag TXR of the selected message buffer is reset. This can be used to stop transmission from message buffers operated in continuous mode.

Input buffer (IBF) and the output buffer (OBF) are built up as a double buffer structure. One half of this double buffer structure is accessible by the host CPU (IBF host / OBF host), while the other half (IBF shadow / OBF shadow) is accessed by the message handler for data transfers between IBF / OBF and message RAM.

Figure 26-17. Host Access to Message RAM



26.2.12.2.1 Data Transfer from Input Buffer to Message RAM

To configure / update a message buffer in the message RAM, the host has to write the data to WRDSn and the header to WRHS1…3. The specific action is selected by configuring the input buffer command mask IBCM.

When the host writes the number of the target message buffer in the message RAM to IBCR.IBRH(6-0) in the input buffer command request register IBCR, IBF host and IBF shadow are swapped (Figure 26-18).

Figure 26-18. Double Buffer Structure Input Buffer

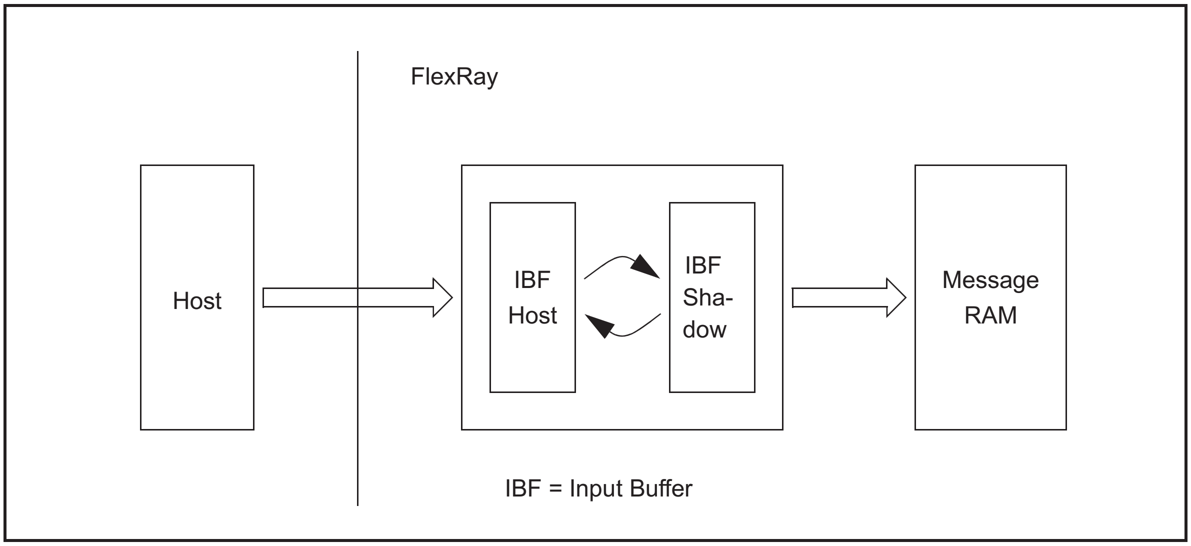
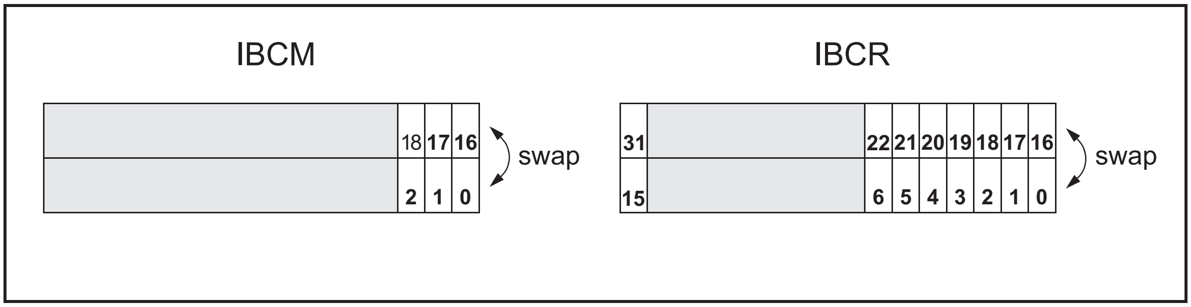


Figure 26-19. Swapping of IBCM and IBCR Bits



With this write operation the IBCR.IBSYS bit in the input buffer command request register is set to 1. The message handler then starts to transfer the contents of IBF shadow to the message buffer in the message RAM selected by IBCR.IBRS(6-0).

While the message handler transfers the data from IBF shadow to the target message buffer in the message RAM, the host may write the next message to IBF host. After the transfer between IBF shadow and the message RAM has completed, the IBCR.IBSYS bit is set back to 0 and the next transfer to the message RAM may be started by the host by writing the corresponding target message buffer number to IBCR.IBRH(6-0) in the input buffer command request register.

If a write access to IBCR.IBRH(6-0) occurs while IBCR.IBSYS is 1, IBCR.IBSYH is set to 1. After completion of the ongoing data transfer from IBF shadow to the message RAM, IBF host and IBF shadow are swapped, IBCR.IBSYH is reset to 0, IBCR.IBSYS remains set to 1, and the next transfer to the message RAM is started. In addition the message buffer numbers stored under IBCR.IBRH(6-0) and IBCR.IBRS(6-0) and the command mask flags are also swapped.

Example of a 8/16/32-bit host access sequence:

- Configure / update n-th message buffer through IBF
- Wait until IBCR.IBSYH is reset
- Write data section to WRDSn
- Write header section to WRHS1,2,3
- Write command mask: write IBCM.STXRH, IBCM.LHSH, IBCM.LDSH
- Demand data transfer to target message buffer: write IBCR.IBRH(6-0)

Configure / update further message buffer through IBF in the same way.

NOTE: Any write access to IBF while IBCR.IBSYH is 1 will set error flag EIR.IIBA in the Error Interrupt Register to 1. In this case the write access has no effect.

Table 26-12. Assignment of Input Buffer Command Mask Bits

Position	Access	Bit	Function
18	r	STXRS	Set Transmission Request shadow ongoing or finished
17	r	LDSS	Load Data Section shadow ongoing or finished
16	r	LHSS	Load Header Section shadow ongoing or finished
2	r/w	STXRH	Set Transmission Request Host
1	r/w	LDSH	Load Data Section Host
0	r/w	LHSH	Load Header Section Host

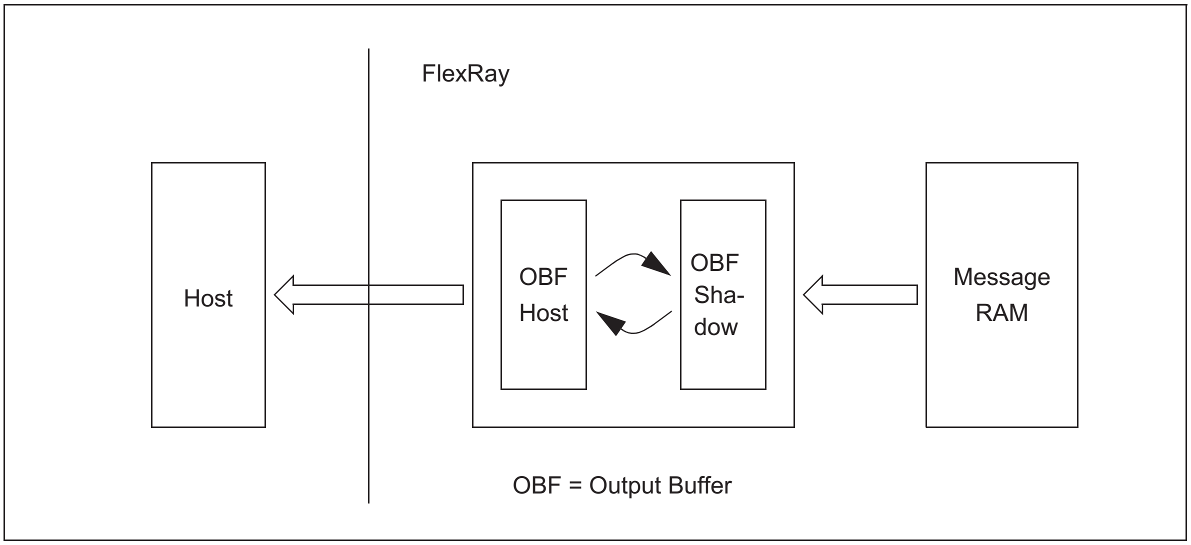
Table 26-13. Assignment of Input Buffer Command Request Bits

Position	Access	Bit	Function
31	r	IBSYS	IBF Busy Shadow, signals ongoing transfer from IBF shadow to message RAM
22-16	r	IBRS(6-0)	IBF Request Shadow, number of message buffer currently / last updated
15	r	IBSYH	IBF Busy Host, transfer request pending for message buffer referenced by IBRH(6-0)
6-0	r/w	IBRH(6-0)	IBF Request Host, number of message buffer to be updated next

26.2.12.2.2 Data Transfer from Message RAM to Output Buffer

To read out a message buffer from the message RAM, the host has to write to the output buffer command mask and command request register to trigger the data transfer. After a transfer has completed the host can read the transferred data from the RDDSn, RDHS1,2,3, and MBS.

Figure 26-20. Double Buffer Structure Output Buffer

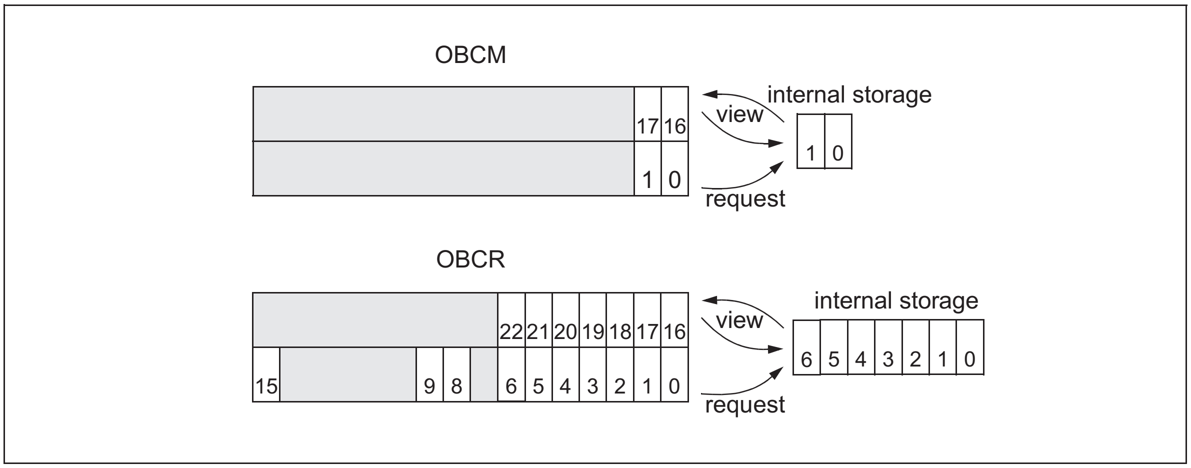


OBF host and OBF shadow as well as bits OBCM.RHSS, OBCM.RDSS, OBCM.RHSH, OBCM.RDSH from the output buffer command mask register and bits OBCM.OBRS(6-0), OBCM.OBRH(6-0) from the output buffer command request register are swapped under control of bits OBCR.VIEW and OBCR.REQ from the output buffer command request register.

Writing bit OBCR.REQ in the output buffer command request register to 1 copies bits OBCM.RHSS, OBCM.RDSS from the output buffer command mask register and bits OBCR.OBRS(6-0) from the output buffer command request register to an internal storage (see Figure 26-21).

After setting OBCR.REQ to 1, OBCR.OBSYS is set to 1, and the transfer of the message buffer selected by OBCR.OBRS(6-0) from the message RAM to OBF shadow is started. After the transfer between the message RAM and OBF shadow has completed, the OBCR.OBSYS bit is set back to 0. Bits OBCR.REQ and OBCR.VIEW can only be set to 1 while OBCR.OBSYS is 0.

Figure 26-21. Swapping of OBCM and OBCR Bits



OBF host and OBF shadow are swapped by setting bit OBCR.VIEW in the output buffer command request register to 1 while bit OBCR.OBSYS is 0 (see Figure 26-20).

In addition bits OBCR.OBRH(6-0) are swapped with the output buffer command request registers internal storage and bits OBCM.RHSH, OBCM.RDSH are swapped with the output buffer command mask registers internal storage thus assuring that the message buffer number stored in OBCR.OBRH(6-0) and the mask configuration stored in OBCM.RHSH, OBCM.RDSH matches the transferred data stored in OBF host (see Figure 26-21).

Now the host can read the transferred message buffer from OBF host while the message handler may transfer the next message from the message RAM to OBF shadow.

NOTE: If bits REQ and VIEW are set to 1 with the same write access while OBSYS is 0, OBSYS is automatically set to 1 and OBF shadow and OBF host are swapped. Additionally mask bits OBCM.RDSH and OBCM.RHSH are swapped with the registers internal storage to keep them attached to the corresponding Output Buffer transfer. Afterwards OBRS (6-0) is copied to the register internal storage, mask bits OBCM.RDSS and OBCM.RHSS are copied to register OBCM internal storage, and the transfer of the selected message buffer from the Message RAM to OBF shadow is started. While the transfer is ongoing the Host can read the message buffer transferred by the previous transfer from OBF Host. When the current transfer between Message RAM and OBF shadow has completed, this is signaled by setting OBSYS back to 0.

Example of an 8/16/32-bit host access to a single message buffer:

If a single message buffer has to be read out, two separate write accesses to OBCR.REQ and OBCR.VIEW are necessary:

- Wait until OBCR.OBSYS is reset
- Write Output Buffer Command Mask OBCM.RHSS, OBCM.RDSS
- Request transfer of message buffer to OBF Shadow by writing OBCR.OBRS(6-0) and OBCR.REQ (in case of and 8-bit Host interface, OBCR.OBRS(6-0) has to be written before OBCR.REQ).
- Wait until OBCR.OBSYS is reset
- Toggle OBF Shadow and OBF Host by writing OBCR.VIEW = 1
- Read out transferred message buffer by reading RDDSn, RDHS1,2,3, and MBS

Example of an 8/16/32-bit host access sequence:

Request transfer of 1st message buffer to OBF shadow

- Wait until OBCR.OBSYS is reset
- Write Output Buffer Command Mask OBCM.RHSS, OBCM.RDSS for 1st message buffer
- Request transfer of 1st message buffer to OBF Shadow by writing OBCR.OBRS(6-0) and OBCR.REQ (in case of an 8-bit Host interface, OBCR.OBRS(6-0) has to be written before OBCR.REQ).

Toggle OBF Shadow and OBF Host to read out 1st transferred message buffer and request transfer of 2nd message buffer:

Request transfer of 2nd message buffer to OBF shadow, read out 1st message buffer from OBF host

- Wait until OBCR.OBSYS is reset
- Write Output Buffer Command Mask OBCM.RHSS, OBCM.RDSS for 2nd message buffer
- Toggle OBF Shadow and OBF Host and start transfer of 2nd message buffer to OBF Shadow simultaneously by writing OBCR.OBRS(6-0) of 2nd message buffer, OBCR.REQ, and OBCR.VIEW (in case of and 8-bit Host interface, OBCR.OBRS(6-0) has to be written before OBCR.REQ and OBCR.VIEW).
- Read out 1st transferred message buffer by reading RDDSn, RDHS13, and MBS

For further transfers continue the same way.

Demand access to last requested message buffer without request of another message buffer:

- Wait until OBCR.OBSYS is reset
- Demand access to last transferred message buffer by writing OBCR.VIEW
- Read out last transferred message buffer by reading RDDSn, RDHS1,2,3, and MBS

Table 26-14. Assignment of Output Buffer Command Mask Bits

Position	Access	Bit	Function
17	r	RDSH	Read Data Section Host access
16	r	RHSH	Read Header Section Host access
1	r/w	RDSS	Read Data Section Shadow
0	r/w	RHSS	Read Header Section Shadow

Table 26-15. Assignment of Output Buffer Command Request Bits

Position	Access	Bit	Function
22-16	r	OBRH(6-0)	OBF Request Host, number of message buffer available for host access
15	r	OBSYS	OBF Busy Shadow, signals ongoing transfer from message RAM to OBF Shadow
9	r/w	REQ	Request Transfer from message RAM to OBF Shadow
8	r/w	VIEW	View OBF Shadow, swap OBF Shadow and OBF Host
6-0	r/w	OBRS(6-0)	OBF Request Shadow, number of message buffer for next request

26.2.12.3 FlexRay Protocol Controller Access to Message RAM

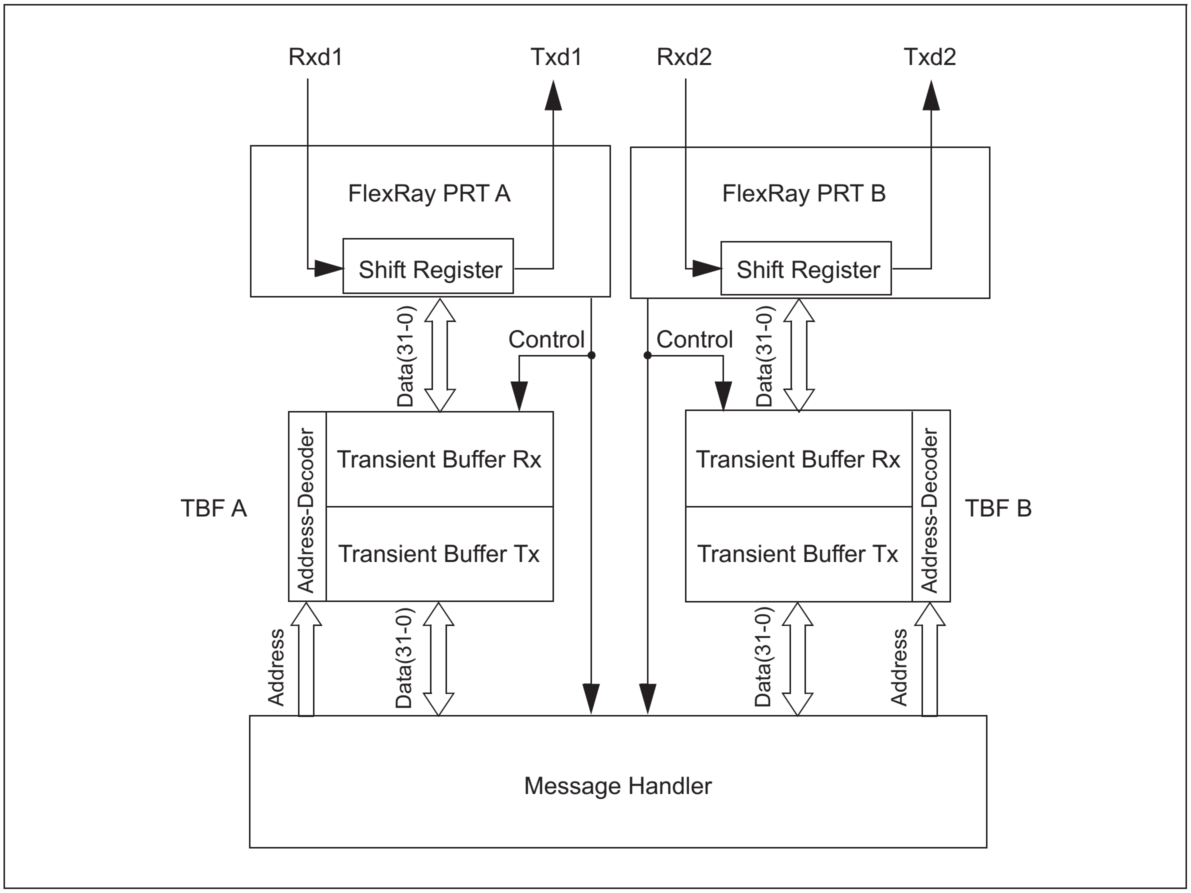
The two transient buffer RAMs (TBF A,B) are used to buffer the data for transfer between the two FlexRay channel protocol controllers and the message RAM.

Each transient buffer RAM is built up as a double buffer, able to store two complete FlexRay messages. There is always one buffer assigned to the corresponding protocol controller while the other one is accessible by the message handler.

If, for example, the message handler writes the next message to be sent to transient buffer Tx, the FlexRay Channel protocol controller can access transient buffer Rx to store the message it is currently receiving. During transmission of the message stored in transient buffer Tx, the message handler transfers the last received message stored in transient buffer Rx to the message RAM (if it passes acceptance filtering) and updates the corresponding message buffer.

Data transfers between the transient buffer RAMs and the shift registers of the FlexRay channel protocol controllers are done in words of 32 bit. This enables the use of a 32 bit shift register independent of the length of the FlexRay messages.

Figure 26-22. Access to Transient Buffer RAMs



26.2.13 Module RAMs

The FlexRay module contains the following RAM portions:

- Message RAM
- Transient Buffer RAM Channel A (TBF A)
- Transient Buffer RAM Channel B (TBF B)
- Input Buffer (IBF)
- Input Buffer Shadow (IBFS)
- Output Buffer (OBF)
- Output Buffer Shadow (OBFS)
- Transfer Configuration RAM (TCR)

All RAMs except the TCR are part of the Communication Controller core.

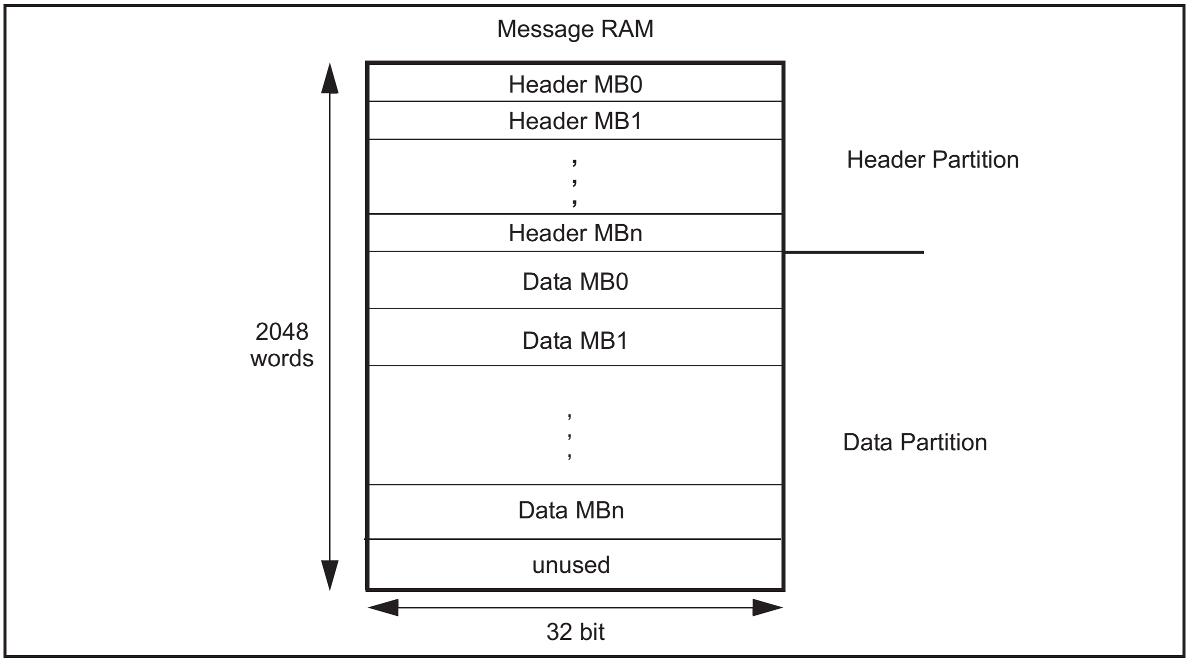
26.2.13.1 Message RAM

To avoid conflicts between host access to the message RAM and FlexRay message reception / transmission, the host CPU cannot directly access the message buffers in the message RAM. These accesses are handled through the input and output buffers. The message RAM is able to store up to 128 message buffers depending on the configured payload length.

The message RAM has a structure as shown in Figure 26-23.

The data partition is allowed to start at Message RAM word number: (MRC.LCB + 1) • 4

Figure 26-23. Configuration Example of Message Buffers in the Message RAM



Header Partition

Stores header segments of FlexRay frames:

- Supports a maximum of 128 message buffers
- Each message buffer has a header of four 32 bit words
- Header 3 of each message buffer holds the 11-bit data pointer to the corresponding data section in the data partition

Data Partition

Flexible storage of data sections with different length. Some maximum values are:

- 30 message buffers with 254 byte data section each
- Or 56 message buffers with 128 byte data section each
- Or 128 message buffers with 48 byte data section each

Restriction: header partition + data partition may not occupy more than 2048 x 32 bit words.

26.2.13.1.1 Header Partition

The elements used for configuration of a message buffer as well as the current message buffer status are stored in the header partition of the message RAM as shown in Figure 26-24. Configuration of the header sections of the message buffers is done through IBF (Write Header Section 1,2,3). Read access to the header sections is done through OBF (read header section 1,2,3 + message buffer status). The data pointer has to be calculated by the programmer to define the starting point of the data section for the corresponding message buffer in the data partition of the message RAM. The data pointer should not be modified during runtime. For message buffers belonging to the receive FIFO (re)configuration should be done in DEFAULT_CONFIG or CONFIG state only.

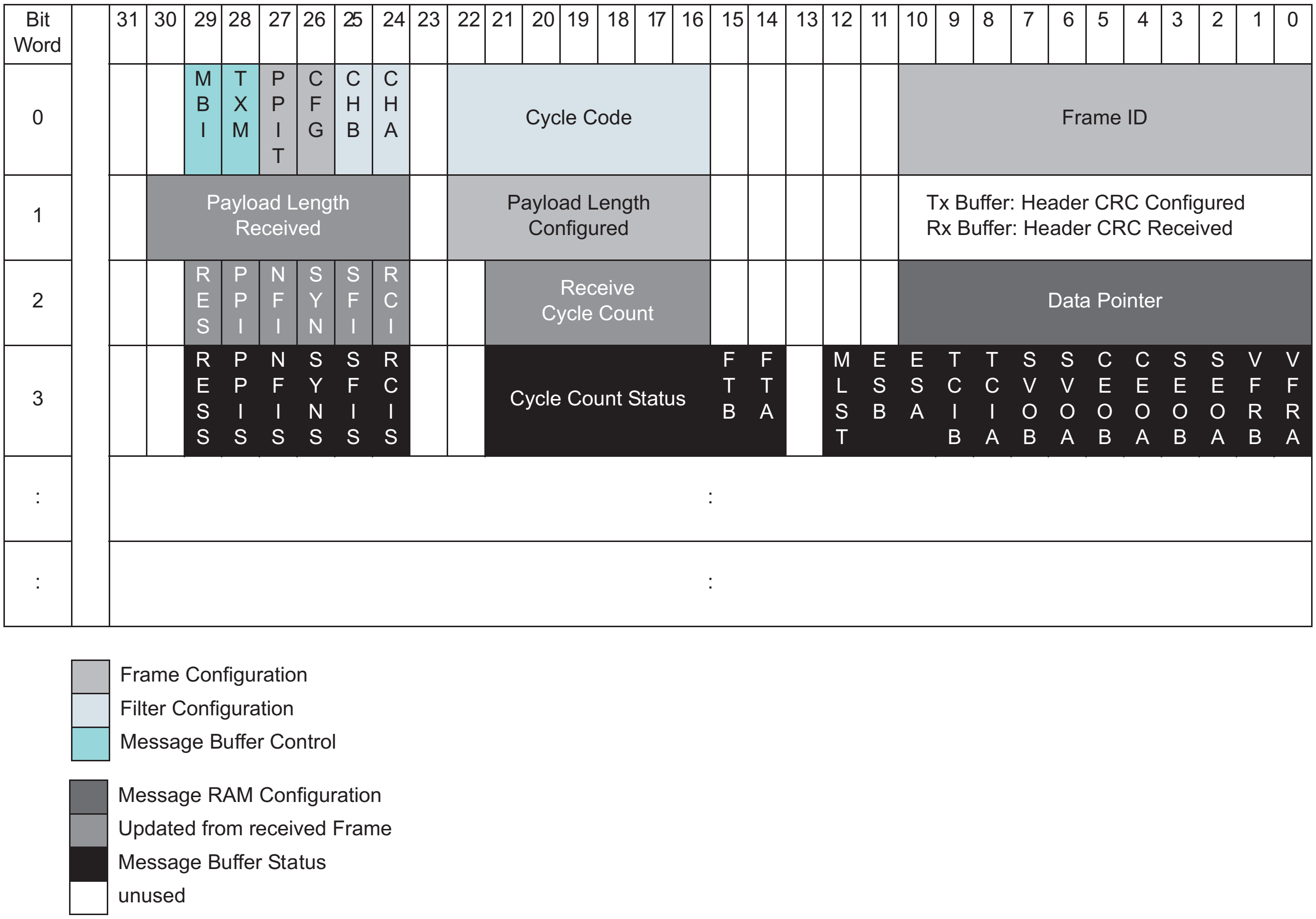
The header section of each message buffer occupies four 32 bit words in the header partition of the message RAM. The header of message buffer 0 starts with the first word in the message RAM.

For transmit buffers the Header CRC has to be calculated by the host CPU.

Payload length received (PLR(6-0)), receive cycle count (RCC(5-0)), Received on Channel Indication (RCI), Startup Frame Indication bit (SFI), sync bit (SYN), null frame indication bit (NFI), payload preamble indication bit (PPI), and reserved bit (RES) are only updated from received valid data frames only.

Header word 3 of each configured message buffer holds the corresponding message buffer status MBS.

Figure 26-24. Header Section of Message Buffer in Message RAM



Bit Word		31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0				MBI	TXM	PPIT	CFG	CHB	CHA		Cycle Code												Frame ID										
1			Payload Length Received								Payload Length Configured												Tx Buffer: Header CRC Configured Rx Buffer: Header CRC Received										
2				RES	PPI	NFI	SYN	SFI	RCI			Receive Cycle Count											Data Pointer										
3				RESS	PPIS	NFIS	SYNS	SFIS	RCIS			Cycle Count Status						FTB	FTA		MLST	ESB	ESA	TCIB	TCIA	SVOB	SVOA	CEOB	CEOA	SEOB	SEOA	VFRB	VFRA
:		:																															
:		:																															

- Frame Configuration
- Filter Configuration
- Message Buffer Control

- Message RAM Configuration
- Updated from received Frame
- Message Buffer Status
- unused

Header 1 (Word 0)

Write access through WRHS1, read access through RDHS1:

- Frame ID- Slot counter filtering configuration
- Cycle Code- Cycle counter filtering configuration
- CHA, CHB- Channel filtering configuration
- CFG- Message buffer configuration: receive / transmit
- PPIT- Payload Preamble Indicator Transmit
- TXM- Transmit mode configuration: single-shot / continuous
- MBI- Message buffer receive / transmit interrupt enable

Header 2 (Word 1)

Write access through WRHS2, read access through RDHS2:

- Header CRC Transmit Buffer: Configured by the host (calculated from frame header)
  - Receive Buffer: Updated from received frame
- Payload Length Configured: Length of data section (2-byte words) as configured by the host
- Payload Length Received: Length of payload segment (2-byte words) stored from received frame

Header 3 (Word 2)

Write access through WRHS3, read access through RDHS3:

- Data Pointer- Pointer to the beginning of the corresponding data section in the data partition

Read access through RDHS3, valid for receive buffers only, updated from received frames:

- Receive Cycle Count - Cycle count from received frame
- RCI- Received on Channel Indicator
- SFI- Startup Frame Indicator
- SYN- Sync Frame Indicator
- NFI- Null Frame Indicator
- PPI- Payload Preamble Indicator
- RES- REServed bit

Message Buffer Status MBS (Word 3)

Read access through MBS, updated by the communication controller at the end of the configured slot.

- VFRA- Valid Frame received on channel A
- VFRB- Valid Frame received on channel B
- SEOA- Syntax Error Observed on channel A
- SEOB- Syntax Error Observed on channel B
- CEOA- Content Error Observed on channel A
- CEOB- Content Error Observed on channel B
- SVOA- Slot Boundary Violation Observed on channel A
- SVOB- Slot Boundary Violation Observed on channel B
- TCIA- Transmission Conflict Indication channel A
- TCIB- Transmission Conflict Indication channel B
- ESA- Empty Slot Channel A
- ESB- Empty Slot Channel B
- MLST- Message Lost
- FTA- Frame Transmitted on Channel A
- FTB- Frame Transmitted on Channel B
- Cycle Count Status- Current cycle count when status was updated
- RCIS- Received on Channel Indicator Status
- SFIS- Startup Frame Indication Status
- SYNS- Sync Frame Indicator Status
- NFIS- Null Frame Indicator Status
- PPIS- Payload Preamble Indicator Status
- RESS- Reserved Bit Status

26.2.13.1.2 Data Partition

The data partition of the message RAM stores the data sections of the message buffers configured for reception / transmission as defined in the header partition. The number of data bytes for each message buffer can vary from 0 to 254. In order to optimize the data transfer between the shift registers of the two FlexRay protocol controllers and the message RAM as well as between the host interface and the message RAM, the physical width of the message RAM is word wise (4 bytes).

The data partition starts right after the last word of the header partition. When configuring the message buffers in the message RAM the programmer has to assure that the data pointers point to addresses within the data partition.

Figure 26-25 shows an example how the payload of the configured message buffers can be stored in the data partition of the message RAM. Message buffers 0 to 2 are static buffers with a payload of 3, whereas message buffers 3 to n are dynamic buffers with variable payload.

The beginning of a message buffer's data section is determined by the data pointer and the payload length configured in the message buffer's header section. This enables a flexible usage of the available RAM space for storage of message buffers with different data lengths.

The storage of the payload data is word aligned. If the size of a message buffer payload is an odd number of 2-byte words, the remaining 16 bits in the last 32-bit word are unused (see Figure 26-25).

Figure 26-25. Example Structure of Data Partition in Message RAM

<table>
<tr><th>Bit / Word</th><th>31</th><th>30</th><th>29</th><th>28</th><th>27</th><th>26</th><th>25</th><th>24</th><th>23</th><th>22</th><th>21</th><th>20</th><th>19</th><th>18</th><th>17</th><th>16</th><th>15</th><th>14</th><th>13</th><th>12</th><th>11</th><th>10</th><th>9</th><th>8</th><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td>:</td><td colspan="8">MB0 Data3</td><td colspan="8">MB0 Data2</td><td colspan="8">MB0 Data1</td><td colspan="8">MB0 Data0</td></tr>
<tr><td>:</td><td colspan="8">unused</td><td colspan="8">unused</td><td colspan="8">MB0 Data5</td><td colspan="8">MB0 Data4</td></tr>
<tr><td>:</td><td colspan="8">MB1 Data3</td><td colspan="8">MB1 Data2</td><td colspan="8">MB1 Data1</td><td colspan="8">MB1 Data0</td></tr>
<tr><td>:</td><td colspan="8">unused</td><td colspan="8">unused</td><td colspan="8">MB1 Data5</td><td colspan="8">MB1 Data4</td></tr>
<tr><td>:</td><td colspan="8">MB2 Data3</td><td colspan="8">MB2 Data2</td><td colspan="8">MB2 Data1</td><td colspan="8">MB2 Data0</td></tr>
<tr><td>:</td><td colspan="8">unused</td><td colspan="8">unused</td><td colspan="8">MB2 Data5</td><td colspan="8">MB2 Data4</td></tr>
<tr><td>:</td><td colspan="8">MB3 Data3</td><td colspan="8">MB3 Data2</td><td colspan="8">MB3 Data1</td><td colspan="8">MB3 Data0</td></tr>
<tr><td>:</td><td colspan="8">o</td><td colspan="8">o</td><td colspan="8">o</td><td colspan="8">o</td></tr>
<tr><td>:</td><td colspan="8">MB3 Data(k)</td><td colspan="8">MB3 Data(k-1)</td><td colspan="8">MB3 Data(k-2)</td><td colspan="8">MB3 Data(k-3)</td></tr>
<tr><td>:</td><td colspan="8">MBn Data3</td><td colspan="8">MBn Data2</td><td colspan="8">MBn Data1</td><td colspan="8">MBn Data0</td></tr>
<tr><td>:</td><td colspan="8">o</td><td colspan="8">o</td><td colspan="8">o</td><td colspan="8">o</td></tr>
<tr><td>:</td><td colspan="8">o</td><td colspan="8">o</td><td colspan="8">o</td><td colspan="8">o</td></tr>
<tr><td>:</td><td colspan="8">MBn Data(m)</td><td colspan="8">MBn Data(m-1)</td><td colspan="8">MBn Data(m-2)</td><td colspan="8">MBn Data(m-3)</td></tr>
<tr><td>:</td><td colspan="8">unused</td><td colspan="8">unused</td><td colspan="8">unused</td><td colspan="8">unused</td></tr>
<tr><td>:</td><td colspan="8">unused</td><td colspan="8">unused</td><td colspan="8">unused</td><td colspan="8">unused</td></tr>
<tr><td>2046</td><td colspan="8">unused</td><td colspan="8">unused</td><td colspan="8">unused</td><td colspan="8">unused</td></tr>
<tr><td>2047</td><td colspan="8">unused</td><td colspan="8">unused</td><td colspan="8">unused</td><td colspan="8">unused</td></tr>
</table>

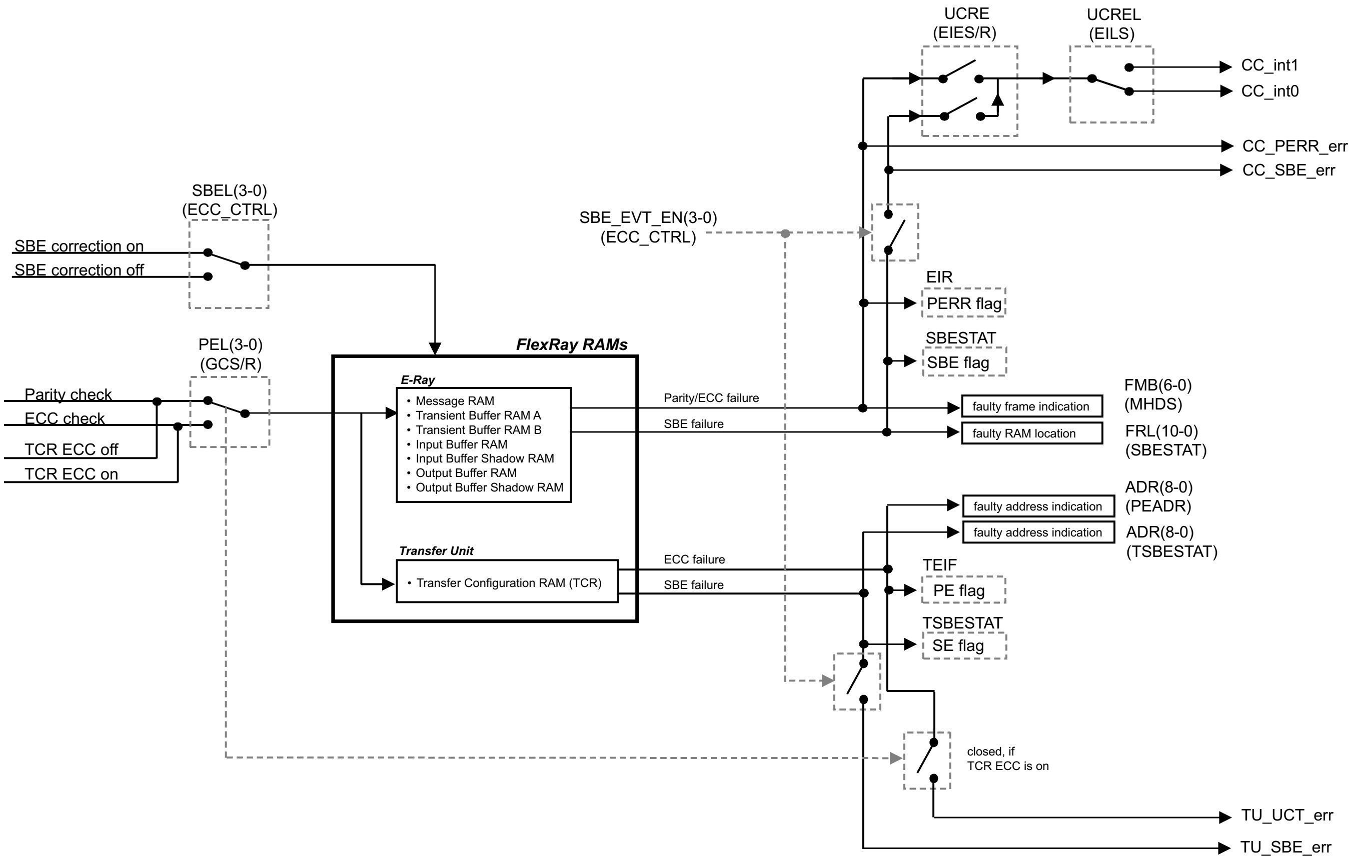
26.2.13.2 ECC Check

In order to assure the integrity of the data stored in the different RAM blocks of the module (message RAM, 2 transient buffer RAMs, 2 input buffer RAMs, 2 output buffer RAMs, Transfer Configuration RAM), the FlexRay module RAMs are optionally ECC protected.

The ECC protection is switched off by default and ECC protection is activated by writing a 4 bit key to the dedicated ECC lock bits (PEL(3-0)) in the Global Control Register (GCS/R) of the Transfer Unit register frame. ECC single-bit error correction is enabled by default and can be disabled by the 4-bit key bits SBEL in the ECC Control Register (ECC_CTRL). Only the Transfer Configuration RAM has the exceptional functionality that ECC protection can either be switched on or off by PEL(3-0). By default the ECC protection is switched off and the TCR is not protected.

Figure 26-26 shows the ECC structure concerning enabling/disabling and error indication.

Figure 26-26. Parity/ECC Structure



NOTE: There is no parity protection for FlexRay RAM blocks.

For the seven RAM blocks of the Communication Controller portion (message RAM, 2 transient buffer RAMs, 2 input buffer RAMs and 2 output buffer RAMs), ECC protection is added, which can be selected by the ECC lock bits (PEL(3-0)). ECC protection can not be switched off completely. For the TCR of the Transfer Unit, actually the ECC multi-bit error generation will be switched on and off by the ECC lock bits, the ECC generation itself remains always on.

ECC should be activated before initializing all RAMs by the CLEAR_RAMS command and should not be switched off during FlexRay usage.

The following paragraphs describe the protection for the Communication Controller related RAM blocks. For details about the protection of the Transfer Configuration RAM (TCR) of the Transfer Unit, see Section 26.2.1.1.2.1.

All the Communication Controller related RAM blocks have an ECC generator and an ECC checker attached as shown in Figure 26-27. When data is written to a RAM block, the local ECC generator generates the corresponding ECC information.

The ECC protection supports single-bit error correction and double-bit error detection mechanism (SECDED). The ECC information is stored together with the corresponding data word.

The ECC is checked each time a data word is read from any of the RAM blocks. The module internal data buses have a width of 32 bits. If an ECC multi-bit error is detected, the PERR error flag is set in the error interrupt register (EIR). Additionally, the correction of an ECC single-bit error will be indicated by the SBE flag in the Single-Bit Error Status Register (SBESTAT).

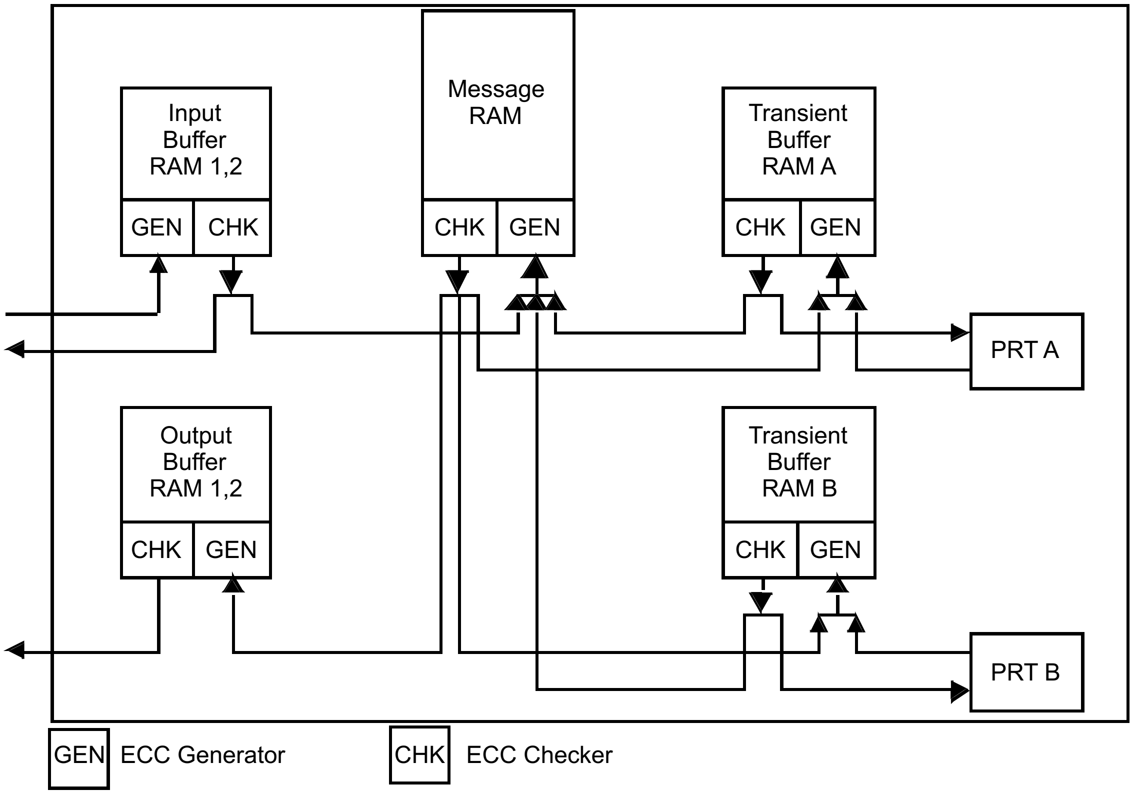
An uncorrectable RAM error interrupt can be generated, if enabled by the UCRE bit in the error interrupt enable register (EIES/R). For ECC single-bit error, the uncorrectable RAM error interrupt is only generated, if the ECC single-bit error correction is disabled and the single-bit error indication key (SBE_EVT_EN in ECC_CTRL) is enabled. When single-bit error correction is turned off, the ECC algorithm will detect up to 3 bits in error in a word.

For ECC multi-bit errors the faulty message buffer number, together with the information of the failing RAM, can be read from the message handler status (MHDS) register. Equivalent information is available for ECC single-bit errors in the single-bit error location (SBESTAT) register, irrespective of ECC single-bit error correction being enabled.

Figure 26-27 shows the data paths between the RAM blocks and the ECC generators and checkers.

The ECC generation is done according to the ECC syndrome tables as shown in Figure 26-28 and Figure 26-29.

Figure 26-27. ECC Generation and Check



NOTE: The ECC generator and ECC checker are not part of the RAM blocks, but of the RAM access logic.

Figure 26-28. ECC Syndrome Table

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	ECC
x	x	x	x	x	x	x	x																	x	x	x	x	x	x	x	x	6
x	x	x	x	x	x	x	x									x	x	x	x	x	x	x	x									5
x	x							x	x	x	x	x	x			x	x							x	x	x	x	x	x			4
		x	x	x				x	x	x				x	x			x	x	x				x	x	x				x	x	3
x		x			x	x		x			x	x			x	x		x			x	x		x			x	x			x	2
			x		x		x		x		x		x	x	x				x		x		x		x		x		x	x	x	1
x		x	x		x					x		x	x	x			x			x		x	x	x	x		x				x	0

ECC Error Bits for Syndrome Decode

6	5	4	3	2	1	0	ECC
							7
x							6
	x						5
		x					4
			x				3
				x			2
					x		1
						x	0

Figure 26-29. ECC Syndrome Table (TCR)

18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	ECC
			x	x	x	x	x	x	x	x									5
		x	x	x							x	x	x	x	x	x			4
	x	x			x	x	x				x	x	x				x	x	3
x	x	x	x		x			x	x		x			x	x			x	2
x		x				x		x		x		x		x		x	x	x	1
x	x	x		x			x		x	x			x		x	x	x		0

ECC Error Bits for Syndrome Decode

5	4	3	2	1	0	ECC
						7
						6
x						5
	x					4
		x				3
			x			2
				x		1
					x	0

When an ECC error has been detected the following actions will be performed:

In all cases:

- The corresponding error flag in the message handler status (MHDS) register is set and the faulty message buffer is indicated. On ECC single-bit error equivalent information is available in the single-bit error location (SBESTAT) register.
- The error flag EIR.PERR in the error interrupt register is set and, if enabled, a module interrupt to the CPU will be generated. An ECC single-bit error is indicated by the SBESTAT.SBE flag irrespective of ECC single-bit error correction being enabled. Additionally an ECC single-bit error can generate an interrupt to the CPU.

Additionally in specific cases of ECC multi-bit errors:

1. ECC multi-bit error during data transfer from input buffer RAM 1,2 ⇒ message RAM
   a. Transfer of header and/or data section and ECC multi-bit error occurs during header and/ or data section transfer to message RAM:
      - MHDS.PIBF bit is set
      - MHDS.FMBD bit is set to indicate that MHDS.FMB(6-0) points to a faulty message buffer
      - MHDS.FMB(6-0) indicates the number of the faulty message buffer
      - Header and/or data section of the corresponding message buffer is updated
      - Transmission request for the corresponding message buffer is not set (no transfer to the FlexRay bus)
   b. Transfer of data section only and ECC multi-bit error occurs when reading header section of the corresponding message buffer from the message RAM.
      - MHDS.PMR bit is set
      - MHDS.FMBD bit is set to indicate that MHDS.FMB(6-0) points to a faulty message buffer
      - MHDS.FMB(6-0) indicates the number of the faulty message buffer
      - The data section of the corresponding message buffer is not updated
      - Transmission request for the corresponding message buffer is not set (no transfer to the FlexRay bus)
2. ECC multi-bit error during host CPU reading input buffer RAM 1,2
   - MHDS.PIBF bit is set
3. ECC multi-bit error during scan of header sections in message RAM
   - MHDS.PMR bit is set
   - MHDS.FMBD bit is set to indicate that MHDS.FMB(6-0) points to a faulty message buffer
   - MHDS.FMB(6-0) indicates the number of the faulty message buffer
   - Ignore message buffer (the transfer of the message buffer is skipped)
4. ECC multi-bit error during data transfer from message RAM to transient buffer RAM 1,2
   - MHDS.PMR bit is set
   - MHDS.FMBD bit is set to indicate that MHDS.FMB(6-0) points to the faulty message buffer
   - MHDS.FMB(6-0) indicates the number of the faulty message buffer
   - Frame not transmitted, frames already in transmission are invalidated by clearing the frame CRC to 0
5. ECC multi-bit error during data transfer from transient buffer RAM 1,2 to protocol controller 1, 2
   - MHDS.PTBF1,2 bit is set
   - Frames already in transmission are invalidated by clearing the frame CRC to 0

6. ECC multi-bit error during data transfer from transient buffer RAM 1,2 to message RAM
    a. ECC multi-bit error when reading header section of corresponding message buffer from message RAM
        - MHDS.PMR bit is set
        - MHDS.FMBD bit is set to indicate that MHDS.FMB(6-0) points to a faulty message buffer
        - MHDS.FMB(6-0) indicates the number of the faulty message buffer
        - The data section of the corresponding message buffer is not updated

    b. ECC multi-bit error when reading transient buffer RAM 1,2:
        - MHDS.PTBF1,2 bit is set
        - MHDS.FMBD bit is set to indicate that MHDS.FMB(6-0) points to a faulty message buffer
        - MHDS.FMB(6-0) indicates the number of the faulty message buffer
        - The data section of the corresponding message buffer is updated
7. ECC multi-bit error during data transfer from message RAM to output buffer RAM
    - MHDS.PMR bit is set
    - MHDS.FMBD bit is set to indicate that MHDS.FMB(6-0) points to faulty message buffer
    - MHDS.FMB(6-0) indicates the number of the faulty message buffer
    - Header and/or data section of the output buffer is updated, but should not be used by the host CPU
8. ECC multi-bit error during host CPU reading output buffer RAM 1,2
    - MHDS.POBF bit is set
9. ECC multi-bit error during data read of transient buffer RAM 1,2

    When an ECC multi-bit error occurs during the Message Handler reads a frame with network management information (PPI = 1) from the transient buffer RAM 1,2 the corresponding network management vector register NMV1,2,3 is not updated from that frame.

Additionally in specific cases of ECC single-bit errors:

1. ECC single-bit error during data transfer from input buffer RAM 1,2 ⇒ message RAM
    a. Transfer of header and/or data section and ECC single-bit error occurs during header and/or data section transfer to message RAM:
        - SBESTAT.SIBF bit is set
        - SBESTAT.FMBD bit is set to indicate that SBESTAT.FMB(6-0) points to a faulty message buffer
        - SBESTAT.FMB(6-0) indicates the number of the faulty message buffer
        - Header and/or data section of the corresponding message buffer is updated

        If ECC single-bit error correction is disabled:

        – Transmission request for the corresponding message buffer is not set (no transfer to the FlexRay bus)

    b. Transfer of data section only and ECC single-bit error occurs when reading header section of corresponding message buffer from the message RAM.
        - SBESTAT.SMR bit is set
        - SBESTAT.FMBD bit is set to indicate that SBESTAT.FMB(6-0) points to a faulty message buffer
        - SBESTAT.FMB(6-0) indicates the number of the faulty message buffer

        If ECC single-bit error correction is disabled:

        – The data section of the corresponding message buffer is not updated

        – Transmission request for the corresponding message buffer is not set (no transfer to the FlexRay bus)
2. ECC single-bit error during host CPU reading input buffer RAM 1,2
    - SBESTAT.SIBF bit is set

3. ECC single-bit error during scan of header sections in message RAM
   - SBESTAT.SMR bit is set
   - SBESTAT.FMBD bit is set to indicate that SBESTAT.FMB(6-0) points to a faulty message buffer
   - SBESTAT.FMB(6-0) indicates the number of the faulty message buffer

   If ECC single-bit error correction is disabled:
   – Ignore message buffer (the transfer of the message buffer is skipped)
4. ECC single-bit error during data transfer from message RAM to transient buffer RAM 1,2
   - SBESTAT.SMR bit is set
   - SBESTAT.FMBD bit is set to indicate that SBESTAT.FMB(6-0) points to the faulty message buffer
   - SBESTAT.FMB(6-0) indicates the number of the faulty message buffer

   If ECC single-bit error correction is disabled:
   – Frame not transmitted, frames already in transmission are invalidated by clearing the frame CRC to 0
5. ECC single-bit error during data transfer from transient buffer RAM 1,2 to protocol controller 1, 2
   - SBESTAT.STBF1,2 bit is set

   If ECC single-bit error correction is disabled:
   – Frames already in transmission are invalidated by setting the frame CRC to 0
6. ECC single-bit error during data transfer from transient buffer RAM 1,2 to message RAM
   a. ECC single-bit error when reading header section of corresponding message buffer from message RAM
      - SBESTAT.SMR bit is set
      - SBESTAT.FMBD bit is set to indicate that SBESTAT.FMB(6-0) points to a faulty message buffer
      - SBESTAT.FMB(6-0) indicates the number of the faulty message buffer

      If ECC single-bit error correction is disabled:
      – The data section of the corresponding message buffer is not updated
   b. ECC single-bit error when reading transient buffer RAM 1,2:
      - SBESTAT.STBF1,2 bit is set
      - SBESTAT.FMBD bit is set to indicate that SBESTAT.FMB(6-0) points to a faulty message buffer
      - SBESTAT.FMB(6-0) indicates the number of the faulty message buffer
      - The data section of the corresponding message buffer is updated
7. ECC single-bit error during data transfer from message RAM to output buffer RAM
   - SBESTAT.SMR bit is set
   - SBESTAT.FMBD bit is set to indicate that SBESTAT.FMB(6-0) points to faulty message buffer
   - SBESTAT.FMB(6-0) indicates the number of the faulty message buffer

   If ECC single-bit error correction is disabled:
   – Header and/or data section of the output buffer is updated, but should not be used by the host CPU
8. ECC single-bit error during host CPU reading output buffer RAM 1,2
   - SBESTAT.SOBF bit is set
9. ECC single-bit error during data read of transient buffer RAM 1,2, when single-bit error correction is disabled.

   When an ECC single-bit error occurs during when the Message Handler reads a frame, with network management information (PPI = 1), from the transient buffer RAM 1,2, the corresponding network management vector register NMV1,2,3 is not updated from that frame.

26.2.13.2.1 Host Handling of Uncorrectable ECC Multi-bit Errors

Uncorrectable errors caused by transient bit flips can be fixed by:

Self-healing

Uncorrectable errors located in:

- Input Buffer RAM 1,2
- Output Buffer RAM 1,2
- Data Section of Message RAM
- Transient Buffer RAM A
- Transient Buffer RAM B
- Transfer Configuration RAM (TCR)

are overwritten with the next write access to the disturbed bit(s) caused by host access or by FlexRay communication.

CLEAR_RAMS Command

When called in DEFAULT_CONFIG or CONFIG state POC command CLEAR_RAMS initializes all module-internal RAMs to 0 and the ECC bits are initialized accordingly, depending what mode is enabled.

Temporary Unlocking of Header Section

An uncorrectable error in the header section of a locked message buffer can be fixed by a transfer from the input buffer to the locked buffer header section. For this transfer, the write-access to the IBCR (specifying the message buffer number) must be immediately preceded by the unlock sequence normally used to leave CONFIG state. For that single transfer the corresponding message buffer header is unlocked, regardless whether it belongs to the FIFO or whether its locking is controlled by MRC.SEC(1-0), and will be updated with new data.

NOTE: In case the previous methods do not work, it is recommended to execute the PBIST test at device level to confirm a hard error in the module internal RAMs.

26.2.14 Interrupts

This section describes the transfer unit interrupts and the communication controller interrupts.

26.2.14.1 Transfer Unit Interrupts

26.2.14.1.1 Interrupt Structure

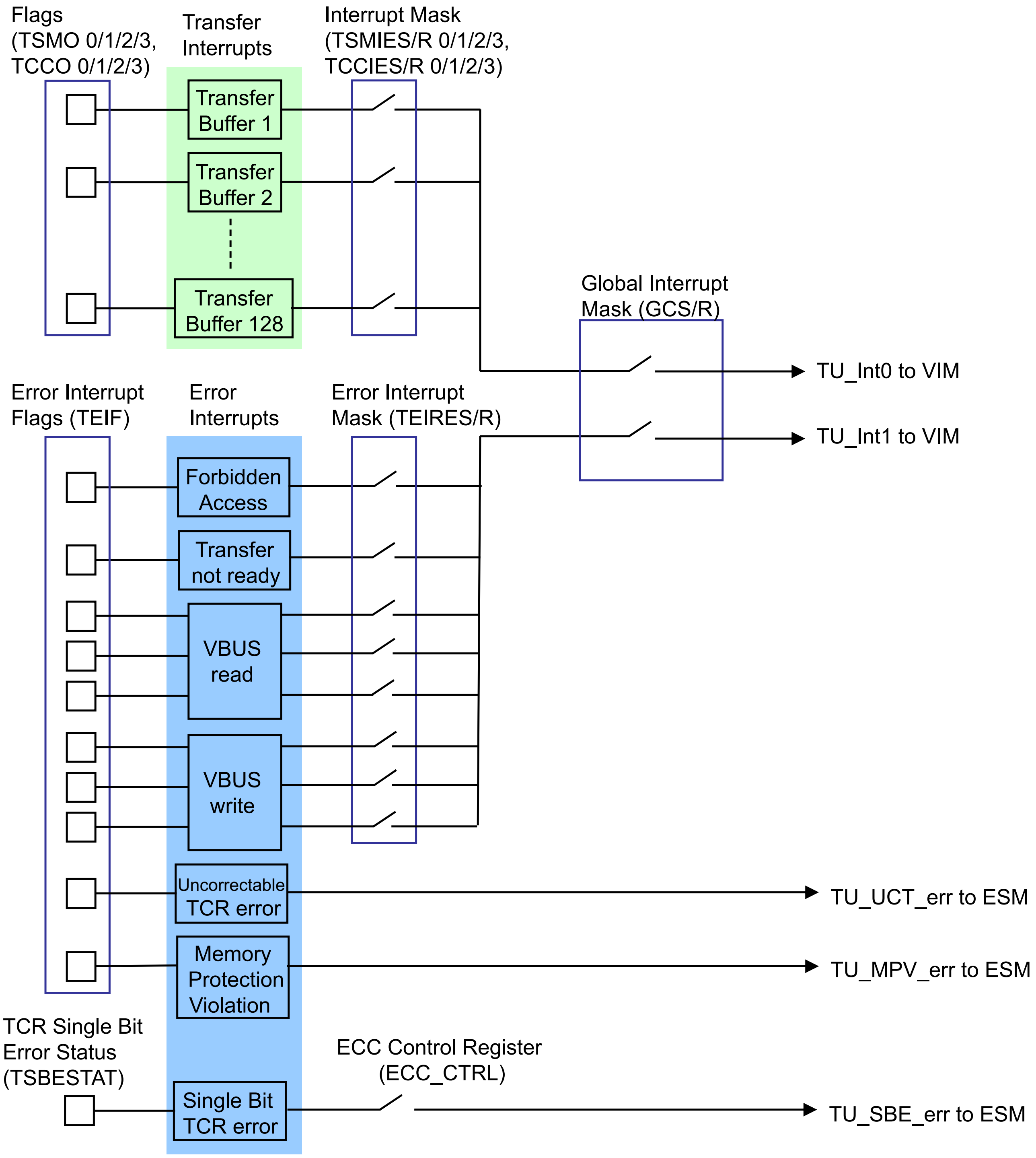
For transfer interrupts, one enable bit is provided for each bit in the transfer occurred status registers. Maskable error interrupts are possible for all error conditions except ECC multi-bit error and memory protection error.

The ECC multi-bit error and the memory protection error have separate non-maskable lines. Both turn off the Transfer Unit after finishing the current word access cycle.

The single-bit error interrupt is maskable. On single-bit error, if single-bit error correction is turned off, the Transfer Unit is turned off after finishing the current word access cycle.

Figure 26-30 shows the interrupt structure of the FlexRay Transfer Unit.

Figure 26-30. Transfer Unit (TU) Interrupt Structure



26.2.14.1.2 Enable Interrupts

TSMIES/R and TCCIES/R control the buffer transfer interrupts for each buffer in both directions. The TEIRES/R registers controls the maskable error interrupt sources which are:

- VBUS transaction errors

  If an error occurs during VBUS read or write transfer a error interrupt will be generated.
- Forbidden access to IBF or OBF

  Since host accesses to communication controller through the IBF and the OBF (0x400-0x7FF) are forbidden, as long as the Transfer Unit State Machine is enabled, accesses will be ignored and an error interrupt will be generated.
- Transfer not ready when TBA should be loaded

  When a transfer is ongoing/pending during base address reload on FlexRay communication cycle start (only occurs if NTBA != TBA) the TBA will not be loaded and an error interrupt will be generated.

The transfer interrupts use a separate interrupt line (TU_int0) than the error interrupts (TU_int1).

26.2.14.1.3 Interrupt Flags

The TSMO and TCCO flags indicate buffer transfer status interrupts whereas the TEIF flags indicate interrupt sources for maskable and non-maskable error interrupts.

The error interrupt flags are set by the Transfer Unit State Machine and can be cleared by the CPU by writing a 1. If the CPU clears the flag, while the Transfer Unit State Machine sets it at the same time, the flag remains set.

26.2.14.1.4 Nonmaskable Error Indication

Memory protection violation and uncorrectable TCR error have their own nonmaskable error lines, which can be connected to the Vectored Interrupt Module (VIM) and/or the Error Signaling Module (ESM). Refer to the device-specific data manual on the hookup.

- If a memory protection violation occurs, the Memory Protection Violation Error (TU_MPV_err) line will be activated.
- If an uncorrectable TCR error occurs while accessing the TCR, the ECC Error (TU_UCT_err) line will be activated. An uncorrectable TCR error can be caused by an ECC error in TCR.

26.2.14.2 Communication Controller Interrupts

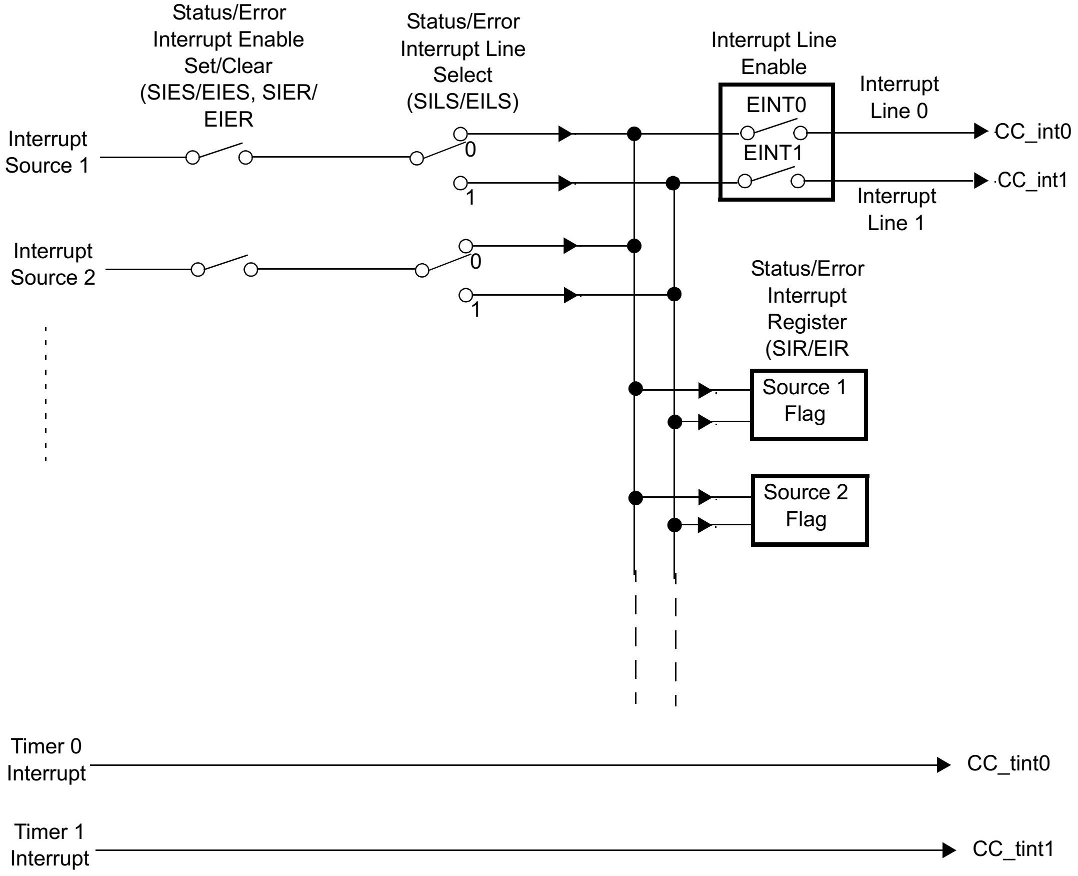
In general, interrupts provide a close link to the protocol timing as they are triggered almost immediately when an error or status change is detected by the controller, a frame is received or transmitted, a configured timer interrupt is activated, or a stop watch event occurred. This enables the host CPU to react very quickly on specific error conditions, status changes, or timer events. To remain flexible though, the communication controller supports disable / enable controls for each individual interrupt source separately.

An interrupt may be triggered, for example, when:

- a frame is received or transmitted
- an error was detected
- a status flag is set
- a timer reaches a preconfigured value
- a message transfer from input buffer to message RAM or from message RAM to output buffer has completed
- a stop watch event occurred

NOTE: For specific information about error interrupt generation on uncorrectable RAM errors, see Figure 26-30.

Figure 26-31. Communication Controller (CC) Interrupt Structure



Tracking status and generating interrupts when a status change or an error occurs are two independent tasks. Independent of an interrupt being enabled, the corresponding status is tracked and indicated by the Communication Controller. The host has access to the current status and error information by reading the error interrupt register and the status interrupt register.

The interrupt lines to the host, CC_int0 and CC_int1, are controlled by the enabled interrupts. In addition each of the two interrupt lines to the host CPU can be enabled / disabled separately by programming bit EINT0 and EINT1 in the Interrupt Line Enable register.

The two timer interrupts generated by interrupt timer 0 and 1 are available on pins CC_tint0 and CC_tint1. They can be configured via the timer 0 and timer 1 configuration register.

When a transfer between IBF / OBF and the Message RAM has completed bit SIR.TIBC or SIR.TOBC is set.

Table 26-16. Module Interrupt Flags and Interrupt Line Enable

Register	Bit	Function
EIR	PEMC	Protocol error Mode Changed
	CNA	Command Not Valid
	SFBM	Sync Frames Below Minimum
	SFO	Sync Frame Overflow
	CCF	Clock Correction Failure
	CCL	CHI Command Locked
	PERR	ECC Error
	RFO	Receive FIFO Overrun
	EFA	Empty FIFO Access
	IIBA	Illegal Input Buffer Access
	IOBA	Illegal Output Buffer Access
	MHF	Message Handler Constraints Flag
	EDA	Error Detected on Channel A
	LTVA	Latest Transmit Violation Channel A
	TABA	Transmission Across Boundary Channel A
	EDB	Error Detected on Channel B
	LTVB	Latest Transmit Violation Channel B
	TABB	Transmission Across Boundary Channel B
SIR	WST	Wakeup Status
	CAS	Collision Avoidance Symbol
	CYCS	Cycle Start Interrupt
	TXI	Transmit Interrupt
	RXI	Receive Interrupt
	RFNE	Receive FIFO not Empty
	RFCL	Receive FIFO Critical Level
	NMVC	Network Management Vector Changed
	TI0	Timer Interrupt 0
	TI1	Timer Interrupt 1
	TIBC	Transfer Input Buffer Completed
	TOBC	Transfer Output Buffer Completed
	SWE	Stop Watch Event
	SUCS	Startup Completed Successfully
	MBSI	Message Buffer Status Interrupt
	SDS	Start of Dynamic Segment
	WUPA	Wakeup Pattern Channel A
	MTSA	MTS Received on Channel A
	WUPB	Wakeup Pattern Channel B
	MTSB	MTS Received on Channel B
ILE	EINT0	Enable Interrupt Line 0
	EINT1	Enable Interrupt Line 1

26.2.15 Minimum Peripheral Clock Frequency

In order to calculate the minimum peripheral clock frequency (VBUSclk) the worst case scenario has to be considered. The worst case scenario depends on the following parameters:

- maximum payload length
- minimum minislot length
- number of configured message buffers (excluding FIFO)
- used channels (single/dual channel)

Worst case scenario:

- reception of a message with a maximum payload length in slot n (n is 7,15,23,31,39,...)
- slot n+1 to n+7 are empty dynamic slots (minislot) and configured as receive buffer
- the find-sequence (usually started in slot 8,16,24,32,40,...) has to scan the maximum number of configured buffers
- the number of concurrent tasks has its maximum value of 3

The find-sequence is executed each 8 slots (slot 8,16,24,32,40,...). It has to be finished until the next find-sequence is requested.

The duration of a Transient Buffer RAM (TBF) transfer to the Message Buffer RAM (MBF) varies from 4 (header section only) to 68 (header + maximum data section) time steps plus a setup time of 6 time steps.

$\text{VBUScycles}_{\text{t2m}}$ = (number of concurrent tasks) x (6 + (number of 4-byte words))

A Slot Status (SS) transfer to the Message Buffer RAM (MBF) has a length of 1 time step plus a setup time of 4 time steps.

$\text{VBUSclk}_{\text{ss2m}}$ = (number of concurrent tasks) x 5

The find sequence has a maximum length of 128 (maximum number of buffers) time steps plus a setup time of 2 time steps.

$\text{VBUSclk}_{\text{find}}$ = (number of concurrent tasks) x (2 + (number of configured buffers))

A minislot has a length of 2 to 63 macroticks (MTicks). The minimum nominal macrotick period (MTcycle) is 1µs. A sequence of 8 minislots has a length of

$t_{\text{8minislots}}$ = 8 x MTicks x MTcycle

The worst case VCLK cycle period can be calculated as follows:

$$t_{8\min islots} \geq \frac{VBUScycles_{t2m} + \left(7 \times VBUScycles_{ss2m}\right) + VBUScycles_{find}}{VBUSclk} \tag{32}$$

$$\frac{1}{VBUS_{clk}} \leq \frac{t_{8\min islots}}{VBUScycle_{t2m} + \left(7 \times VBUScycle_{ss2m}\right) + VBUScycle_{find}} [\mu s] \tag{33}$$

minimum $t_{\text{8minislots}}$ = 8 * 2 * 1 µs = 16 µs

maximum $\text{VBUScycle}_{\text{t2m}}$ = 3 * (6 + 68) = 222

maximum $\text{VBUScycle}_{\text{ss2m}}$ = 3 * 5 = 15

maximum $\text{VBUScycle}_{\text{find}}$ = 3 * (2 + 128) = 390

$$\frac{1}{VBUSclk} \leq = \frac{16\mu s}{222 + 7 \times 15 + 390} = 22.32ns \tag{34}$$

The minimum peripheral clock frequency for this worst case scenario is 44,8125 MHz.

26.2.16 Assignment of FlexRay Configuration Parameters

The following table shows the assignment of the FlexRay parameters as defined in the FlexRay Protocol Specification and the corresponding bit fields of the FlexRay module.

Table 26-17. Assignment of FlexRay Configuration Parameters

Parameter	Bit(field)
pKeySlotusedForStartup	SUCC1.TXST
pKeySlotUsedForSync	SUCC1.TXSY
gColdStartAttempts	SUCC1.CSA(4-0)
pAllowPassiveToActive	SUCC1.PTA(4-0)
pWakeupChannel	SUCC1.WUCS
pSingleSlotEnabled	SUCC1.TSM
pAllowHaltDueToClock	SUCC1.HCSE
pChannels	SUCC1.CCHASUCC1.CCHB
pdListenTimeOut	SUCC2.LT(20-0)
gListenNoise	SUCC2.LTN(3-0)
gMaxWithoutClockCorrectionPassive	SUCC3.WCP(3-0)
gMaxWithoutClockCorrectionFatal	SUCC3.WCF(3-0)
gNetworkManagementVectorLength	NEMC.NML(3-0)
gdTSSTransmitter	PRTC1.TSST(3-0)
gdCASRxLowMax	PRTC1.CASM(6-0)
gdSampleClockPeriod	PRTC1.BRP(1-0)
pSamplesPerMicrotick	PRTC1.BRP(1-0)
gdWakeupSymbolRxWindow	PRTC1.RXW(8-0)
pWakeupPattern	PRTC1.RWP(5-0)
gdWakeupSymbolRxIdle	PRTC2.RXI(5-0)
gdWakeupSymbolRxLow	PRTC2.RXL(5-0)
gdWakeupSymbolTxIdle	PRTC2.TXI(7-0)
gdWakeupSymbolTxLow	PRTC2.TXL(5-0)
gPayloadLengthStatic	MHDC.SFDL(6-0)
pLatestTx	MHDC.SLT(12-0)
pMicroPerCycle	GTUC1.UT(19-0)
gMacroPerCycle	GTUC2.MPC(13-0)
gSyncNodeMax	GTUC2.SNM(3-0)
pMicroInitialOffset[A]	GTUC3.UIOA(7-0)
pMicroInitialOffset[B]	GTUC3.UIOB(7-0)
pMacroInitialOffset[A]	GTUC3.MIOA(6-0)
pMacroInitialOffset[B]	GTUC3.MIOB(6-0)
gdNIT	GTUC4.NIT(13-0)
gOffsetCorrectionStart	GTUC4.OCS(13-0)
pDelayCompensation[A]	GTUC5.DCA(7-0)
pDelayCompensation[B]	GTUC5.DCB(7-0)
pClusterDriftDamping	GTUC5.CDD(4-0)
pDecodingCorrection	GTUC5.DEC(7-0)
pdAcceptedStartupRange	GTUC6.ASR(10-0)
pdMaxDrift	GTUC6.MOD(10-0)
gdStaticSlot	GTUC7.SSL(9-0)
gNumberOfStaticSlots	GTUC7.NSS(9-0)
gdMinislot	GTUC8.MSL(5-0)

Table 26-17. Assignment of FlexRay Configuration Parameters (continued)

Parameter	Bit(field)
gNumberOfMinislots	GTUC8.NMS(12-0)
gdActionPointOffset	GTUC9.APO(5-0)
gdMinislotActionPointOffset	GTUC9.MAPO(4-0)
gdDynamicSlotIdlePhase	GTUC9.DSI(1-0)
pOffsetCorrectionOut	GTUC10.MOC(13-0)
pRateCorrectionOut	GTUC10.MRC(10-0)
pExternOffsetCorrection	GTUC11.EOC(2-0)
pExternRateCorrection	GTUC11.ERC(2-0)

26.2.17 Emulation/Debug Support

For any debug transactions on the bus interface (VBUSP or OCP), the responses are normal except for the following:

1. Reads will not clear “read-clear” type of bits during the same.

2. User mode writes are allowed to "privilege mode write only" bits as well.

26.3 FlexRay Module Registers

26.3.1 Transfer Unit Registers

Table 26-18 provides a summary of the registers. All registers are organized as 32-bit registers. 8-, 16-, and 32-bit accesses are supported. For FlexRayTU transfers only, 4 × 32-bit data packages are supported. The base address for the Transfer Unit registers is FFF7 A000h.

The Transfer Unit State Machine registers use the offset address range 00h to 1FCh.

Transfer Configuration RAM uses the offset address range 00h to 1FCh in normal mode and 00h to 3FCh in ECC test mode.

Table 26-18. Transfer Unit Registers

Offset Address	Acronym	Register Description	Section
000h	GSN0	Global Static Number 0	Section 26.3.1.1
004h	GSN1	Global Static Number 1	Section 26.3.1.2
010h	GCS	Global Control Set	Section 26.3.1.3
014h	GCR	Global Control Reset	Section 26.3.1.3
018h	TSCB	Transfer Status Current Buffer	Section 26.3.1.4
01Ch	LTBCC	Last Transferred Buffer to Communication Controller	Section 26.3.1.5
020h	LTBSM	Last Transferred Buffer to System Memory	Section 26.3.1.6
024h	TBA	Transfer Base Address	Section 26.3.1.7
028h	NTBA	Next Transfer Base Address	Section 26.3.1.8
02Ch	BAMS	Base Address of Mirrored Status	Section 26.3.1.9
030h	SAMP	Start Address of Memory Protection	Section 26.3.1.10
034h	EAMP	End Address of Memory Protection	Section 26.3.1.11
040h	TSMO1	Transfer to System Memory Occurred 1	Section 26.3.1.12
044h	TSMO2	Transfer to System Memory Occurred 2	Section 26.3.1.12
048h	TSMO3	Transfer to System Memory Occurred 3	Section 26.3.1.12
04Ch	TSMO4	Transfer to System Memory Occurred 4	Section 26.3.1.12
050h	TCCO1	Transfer to Communication Controller Occurred 1	Section 26.3.1.13
054h	TCCO2	Transfer to Communication Controller Occurred 2	Section 26.3.1.13
058h	TCCO3	Transfer to Communication Controller Occurred 3	Section 26.3.1.13
05Ch	TCCO4	Transfer to Communication Controller Occurred 4	Section 26.3.1.13
060h	TOOFF	Transfer Occurred Offset	Section 26.3.1.14
06Ch	TSBESTAT	TCR ECC Single-Bit Error Status	Section 26.3.1.15
070h	PEADR	ECC Error Address	Section 26.3.1.16
074h	TEIF	Transfer Error Interrupt	Section 26.3.1.17
078h	TEIRES	Transfer Error Interrupt Enable Set	Section 26.3.1.18
07Ch	TEIRER	Transfer Error Interrupt Enable Reset	Section 26.3.1.18
080h	TTSMS1	Trigger Transfer to System Memory Set 1	Section 26.3.1.19
084h	TTSMR1	Trigger Transfer to System Memory Reset 1	Section 26.3.1.19
088h	TTSMS2	Trigger Transfer to System Memory Set 2	Section 26.3.1.19
08Ch	TTSMR2	Trigger Transfer to System Memory Reset 2	Section 26.3.1.19
090h	TTSMS3	Trigger Transfer to System Memory Set 3	Section 26.3.1.19
094h	TTSMR3	Trigger Transfer to System Memory Reset 3	Section 26.3.1.19
098h	TTSMS4	Trigger Transfer to System Memory Set 4	Section 26.3.1.19
09Ch	TTSMR4	Trigger Transfer to System Memory Reset 4	Section 26.3.1.19
0A0h	TTCCS1	Trigger Transfer to Communication Controller Set 1	Section 26.3.1.20
0A4h	TTCCR1	Trigger Transfer to Communication Controller Reset 1	Section 26.3.1.20
0A8h	TTCCS2	Trigger Transfer to Communication Controller Set 2	Section 26.3.1.20

Table 26-18. Transfer Unit Registers (continued)

Offset Address	Acronym	Register Description	Section
0ACh	TTCCR2	Trigger Transfer to Communication Controller Reset 2	Section 26.3.1.20
0B0h	TTCCS3	Trigger Transfer to Communication Controller Set 3	Section 26.3.1.20
0B4h	TTCCR3	Trigger Transfer to Communication Controller Reset 3	Section 26.3.1.20
0B8h	TTCCS4	Trigger Transfer to Communication Controller Set 4	Section 26.3.1.20
0BCh	TTCCR4	Trigger Transfer to Communication Controller Reset 4	Section 26.3.1.20
0C0h	ETESMS1	Enable Transfer on Event to System Memory Set 1	Section 26.3.1.21
0C4h	ETESMR1	Enable Transfer on Event to System Memory Reset 1	Section 26.3.1.21
0C8h	ETESMS2	Enable Transfer on Event to System Memory Set 2	Section 26.3.1.21
0CCh	ETESMR2	Enable Transfer on Event to System Memory Reset 2	Section 26.3.1.21
0D0h	ETESMS3	Enable Transfer on Event to System Memory Set 3	Section 26.3.1.21
0D4h	ETESMR3	Enable Transfer on Event to System Memory Reset 3	Section 26.3.1.21
0D8h	ETESMS4	Enable Transfer on Event to System Memory Set 4	Section 26.3.1.21
0DCh	ETESMR4	Enable Transfer on Event to System Memory Reset 4	Section 26.3.1.21
0E0h	CESMS1	Clear on Event to System Memory Set 1	Section 26.3.1.22
0E4h	CESMR1	Clear on Event to System Memory Reset 1	Section 26.3.1.22
0E8h	CESMS2	Clear on Event to System Memory Set 2	Section 26.3.1.22
0ECh	CESMR2	Clear on Event to System Memory Reset 2	Section 26.3.1.22
0F0h	CESMS3	Clear on Event to System Memory Set 3	Section 26.3.1.22
0F4h	CESMR3	Clear on Event to System Memory Reset 3	Section 26.3.1.22
0F8h	CESMS4	Clear on Event to System Memory Set 4	Section 26.3.1.22
0FCh	CESMR4	Clear on Event to System Memory Reset 4	Section 26.3.1.22
100h	TSMIES1	Transfer to System Memory Interrupt Enable Set 1	Section 26.3.1.23
104h	TSMIER1	Transfer to System Memory Interrupt Enable Reset 1	Section 26.3.1.23
108h	TSMIES2	Transfer to System Memory Interrupt Enable Set 2	Section 26.3.1.23
10Ch	TSMIER2	Transfer to System Memory Interrupt Enable Reset 2	Section 26.3.1.23
110h	TSMIES3	Transfer to System Memory Interrupt Enable Set 3	Section 26.3.1.23
114h	TSMIER3	Transfer to System Memory Interrupt Enable Reset 3	Section 26.3.1.23
118h	TSMIES4	Transfer to System Memory Interrupt Enable Set 4	Section 26.3.1.23
11Ch	TSMIER4	Transfer to System Memory Interrupt Enable Reset 4	Section 26.3.1.23
120h	TCCIES1	Transfer to Communication Controller Interrupt Enable Set 1	Section 26.3.1.24
124h	TCCIER1	Transfer to Communication Controller Interrupt Enable Reset 1	Section 26.3.1.24
128h	TCCIES2	Transfer to Communication Controller Interrupt Enable Set 2	Section 26.3.1.24
12Ch	TCCIER2	Transfer to Communication Controller Interrupt Enable Reset 2	Section 26.3.1.24
130h	TCCIES3	Transfer to Communication Controller Interrupt Enable Set 3	Section 26.3.1.24
134h	TCCIER3	Transfer to Communication Controller Interrupt Enable Reset 3	Section 26.3.1.24
138h	TCCIES4	Transfer to Communication Controller Interrupt Enable Set 4	Section 26.3.1.24
13Ch	TCCIER4	Transfer to Communication Controller Interrupt Enable Reset 4	Section 26.3.1.24
0-1FCh	TCR	Transfer Configuration RAM	Section 26.3.1.25
200h-3FCh	TCR ECC	TCR ECC Test Mode	Section 26.3.1.26

26.3.1.1 Global Static Number 0 (GSN0)

This register contains a constant to check correctness of data transfers.

Figure 26-32. Global Static Number 0 (GSN0) [offset_TU = 00h]

31 … 16
Data_A
R-5432h

15 … 0
Data_B
R-ABCDh

LEGEND: R = Read only; -*n* = value after reset

Table 26-19. Global Static Number 0 (GSN0) Field Descriptions

Bit	Field	Value	Description
31-16	Data_A	0-FFFFh	Data_A
15-0	Data_B	0-FFFFh	Complement of Data_A

26.3.1.2 Global Static Number 1 (GSN1)

This register contains a constant to check correctness of data transfers.

Figure 26-33. Global Static Number 1 (GSN1) [offset_TU = 04h]

31 … 16
Data_C
R-ABCDh

15 … 0
Data_D
R-5432h

LEGEND: R = Read only; -*n* = value after reset

Table 26-20. Global Static Number 1 (GSN1) Field Descriptions

Bit	Field	Value	Description
31-16	Data_C	0-FFFFh	Data_C
15-0	Data_D	0-FFFFh	Complement of Data_C

26.3.1.3 Global Control Set/Reset (GCS/GCR)

The GCx Registers reflects the configuration mode and allows to configure the basic Transfer Unit behavior.

The GCx registers consist of a set register (GCS) and a reset register (GCR). Bits are set by writing 1 to GCS and reset by writing 1 to GCR. Writing a 0 has no effect. Reading from both addresses will result in the same value.

For Global Control Reset (GCR) bit descriptions, see Table 26-21.

Figure 26-34. Global Control Set (GCS) [offset_TU = 10h]

31	30	29	28	27	26	25	24
ENDVBM	ENDVBS	ENDR		ENDH		ENDP	
R/S-0	R/S-0	R/S-0		R/S-0		R/S-0	

23	22	21	20	19			16
Reserved		PRIO	Reserved	PEL			
R-0		R/S-0	R-0	R/S-5h			

15	14	13	12	11		9	8
Reserved	CETESM	CTTCC	CTTSM	Reserved			ETSM
R-0	R/S-0	R/S-0	R/S-0	R-0			R/S-0

7	6	5	4	3	2	1	0
Reserved		SILE	EILE	Reserved		TUH	TUE
R-0		R/S-0	R/S-0	R-0		R/S-0	R/S-0

LEGEND: R/W = Read/Write; R = Read only; S = Set; -*n* = value after reset

Figure 26-35. Global Control Reset (GCR) [offset_TU = 14h]

31	30	29	28	27	26	25	24
ENDVBM	ENDVBS	ENDR		ENDH		ENDP	
R/S-0	R/S-0	R/S-0		R/S-0		R/S-0	

23	22	21	20	19			16
Reserved		PRIO	Reserved	PEL			
R-0		R/S-0	R-0	R/S-5h			

15	14	13	12	11		9	8
Reserved	CETESM	CTTCC	CTTSM	Reserved			ETSM
R-0	R/S-0	R/S-0	R/S-0	R-0			R/S-0

7	6	5	4	3	2	1	0
Reserved		SILE	EILE	Reserved		TUH	TUE
R-0		R/S-0	R/S-0	R-0		R/S-0	R/S-0

LEGEND: R/W = Read/Write; R = Read only; S = Set; -*n* = value after reset

Table 26-21. Global Control Set/Reset (GCS/R) Field Descriptions

Bit	Field	Value	Description
31	ENDVBM		Endianness Correction on VBusp Master.
		0	Endianness correction switched off (Endianness is default: Little Endianness equal to Big Endian word invariant (ARM:BE-32), same as all other peripherals) (Example 32 Bit Word = ABCD).
		1	Endianness correction switched on (E-Ray Register, Header and Payload Endianness is according the configuration of bits ENDR0/1 ENDH0/1, ENDP0/1).
30	ENDVBS		Endianness correction on VBusp Slave.
		0	Endianness correction switched off (Endianness is default: Little Endianness equal to Big Endian word invariant (ARM:BE-32), same as all other peripherals) (Example 32 Bit Word = ABCD).
		1	Endianness correction switched on (E-Ray Register, Header and Payload Endianness is according the configuration of bits ENDR0/1, ENDH0/1, ENDP0/1).
29-28	ENDR		Endianness Correction for No (header or payload) Data Sink Access. Byte-order control of CPU access to E-Ray register, Transfer Unit register and Transfer Unit ram data. Data transferred between CPU and data sink will be corrected.
		0	Remapped to ABCDh.
		1h	Remapped to BADCh.
		2h	Remapped to CDABh.
		3h	Remapped to DCBAh.
27-26	ENDH		Endianness Correction for Header.
		0	Remapped to ABCDh.
		1h	Remapped to BADCh.
		2h	Remapped to CDABh.
		3h	Remapped to DCBAh.
25-24	ENDP		Endianness Correction for Payload.
		0	Remapped to ABCDh.
		1h	Remapped to BADCh.
		2h	Remapped to CDABh.
		3h	Remapped to DCBAh.
23-22	Reserved	0	Reads return 0. Writes have no effect.
21	PRIO		Transfer Priority.
		0	TTSM gets higher priority than TTCC.
		1	TTCC gets higher priority than TTSM.
20	Reserved	0	Reserved
19-16	PEL		ECC Lock.
		5h	ECC interrupt generation for TCR is switched off. ECC protection for message RAM, transient buffer RAMs, input buffer RAMs and output buffer RAMs is switched off.
		Others	ECC interrupt generation for TCR is switched on. ECC protection for message RAM, transient buffer RAMs, input buffer RAMs and output buffer RAMs is switched on. **Note:** For TCR, PEL enables or disables the ECC multi-bit error interrupt generation. While the ECC feature is disabled, the ECC generation is still ongoing and the error indication by the ECC interrupt flag (PE) in the Transfer Error Interrupt Flag register (TEIF) remains active. Only the ECC interrupt generation gets disabled.
15	Reserved	0	Reads return 0. Writes have no effect.
14	CETESM		Clear ETESM Register. Clear all bits of Enable Transfer on Event to System Memory register.
		0	Do not clear the register.
		1	Clear the register when bit is set from 0 to 1.
13	CTTCC		Clear TTCC Register.
		0	Do not clear the register.
		1	Clear the register when bit is set from 0 to 1.

Table 26-21. Global Control Set/Reset (GCS/R) Field Descriptions (continued)

Bit	Field	Value	Description
12	CTTSM		Clear TTSM Register.
		0	Do not clear the register.
		1	Clear the register when bit is set from 0 to 1.
11-9	Reserved	0	Reads return 0. Writes have no effect.
8	ETSM		Enable Transfer Status Mirrored. Mirror technique must be adjustable.
		0	Disable mirror function for TSCB, LTBCC, LTBSM, TSMO1-4, TCCO1-4, and TOOFF.
		1	Enable mirror function for TSCB, LTBCC, LTBSM, TSMO1-4, TCCO1-4, and TOOFF.
7-6	Reserved	0	Reads return 0. Writes have no effect.
5	SILE		Status Interrupt Line Enable. Enable status line interrupt.
		0	TU_Int0 is disabled.
		1	TU_Int0 is enabled.
4	EILE		Error Interrupt Line Enable. Enable error interrupt line.
		0	TU_Int1 is disabled.
		1	TU_Int1 is enabled.
3-2	Reserved	0	Reads return 0. Writes have no effect.
1	TUH		Transfer Unit Halted. When halted, the Transfer Unit State Machine finishes the ongoing VBUSM access before it stops working. After deassertion, the Transfer Unit State Machine continues at the point it was halted before. No reconfiguration is required.
		0	Transfer Unit is not halted.
		1	Transfer Unit is halted. **Note: If the Transfer Unit State Machine halts, all mirroring registers contained the last transfer not the current transfer information.**
0	TUE		Transfer Unit Enabled. Enable transfer unit.
		0	Transfer Unit is disabled, reset Transfer Unit State Machine, completion of the current VBUS transfer cycle but data could be corrupt.
		1	Transfer Unit is enabled. **Note: Before switching on the Transfer Unit, the registers must be set up. After re-enabling of the Transfer Unit State Machine the contents of the module registers and the TCR is still valid (assuming it was continuously powered).**

26.3.1.4 Transfer Status Current Buffer (TSCB)

The Transfer Status Current Buffer displays the current buffer in progress and indicates if the Transfer Unit State Machine is idle and is halt. The IDLE flag is cleared by writing a 1 to it.

Figure 26-36. Transfer Status Current Buffer (TSCB) [offset_TU = 18h]

31–21	20–16
Reserved	TSMS
R-0	R-0

15–13	12	11–9	8	7	6–0
Reserved	STUH	Reserved	IDLE	RSVD	BN
R-0	R-0	R-0	R/W-1	R-0	R-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 26-22. Transfer Status Current Buffer (TSCB) Field Descriptions

Bit	Field	Value	Description
31-21	Reserved	0	Reads return 0. Writes have no effect.
20-16	TSMS		Transfer State Machine Status.
			Reflects the current status of the transfer state machine for debug purpose (only available in debug mode in combination with a debugger). In Normal Operation Mode, the value of TSMS is always read as 0.
		1h	IDLE state
			Transfer Trigger to System Memory:
		2h	Start state (TTSM_START)
		3h	Output Buffer Command Mask access state (TTSM_OBCM)
		4h	Request state (TTSM_REQ)
		5h	View state (TTSM_VIEW)
		6h	Check state (TTSM_CHECK)
		7h	Read Header Section access state (TTSM_RDHS)
		8h	Read Data Section access state (TTSM_RDDS)
			Transfer Trigger to Communication Controller:
		9h	Start state (TTCC_START)
		Ah	Busy state (TTCC_IBUSY)
		Bh	Check state (TTCC_CHECK)
		Ch	Write Header Section access state (TTCC_WRHS)
		Dh	Payload Read state (TTCC_PLC_READ)
		Eh	Payload Calculation state (TTCC_PLC_CALC)
		Fh	Write Data Section access state (TTCC_WRDS)
		10h	Input Buffer Command Mask access state (TTCC_IBCM)
		11h	Input Buffer Command Request access state (TTCC_IBCR)
		12h	Mirror state (TTCC_MIRROR)
		13h	End state (TTSM_END)
		14h-1Eh	Reserved
		1Fh	Undefined state
15-13	Reserved	0	Reads return 0. Writes have no effect.
12	STUH		Status of Transfer Unit State Machine for Halt Detection.
		0	Not in HALT status.
		1	In HALT status.
11-9	Reserved	0	Reads return 0. Writes have no effect.

Table 26-22. Transfer Status Current Buffer (TSCB) Field Descriptions (continued)

Bit	Field	Value	Description
8	IDLE		Detects Transfer State Machine State IDLE. Will be set if the transfer unit state machine is in IDLE state and ready to start the next transfer, but nothing is requested.
		0	IDLE state is not reached since last clear.
		1	IDLE state is reached.
7	Reserved	0	Reads return 0. Writes have no effect.
6-0	BN	0-7Fh	Buffer number. 7-bit value of buffer number, which is currently in transfer. If state machine enters IDLE mode the last transferred buffer number is shown.

26.3.1.5 Last Transferred Buffer to Communication Controller (LTBCC)

Shows the number of the last completely transferred message buffer from system memory to the communication controller.

Figure 26-37. Last Transferred Buffer to Communication Controller (LTBCC) [offset_TU = 1Ch]

31 – 7	6 – 0
Reserved	BN
R-0	R-0

LEGEND: R = Read only; -*n* = value after reset

Table 26-23. Last Transferred Buffer to Communication Controller (LTBCC) Field Descriptions

Bit	Field	Value	Description
31-7	Reserved.	0	Reads return 0. Writes have no effect.
6-0	BN	0-7Fh	Buffer number. 7-bit value of last completely transferred message buffer from system memory to the communication controller.

26.3.1.6 Last Transferred Buffer to System Memory (LTBSM)

Shows the number of the last completely transferred message buffer from communication controller to the system memory.

Figure 26-38. Last Transferred Buffer to System Memory (LTBSM) [offset_TU = 20h]

31 – 7	6 – 0
Reserved	BN
R-0	R-0

LEGEND: R = Read only; -*n* = value after reset

Table 26-24. Last Transferred Buffer to System Memory (LTBSM) Field Descriptions

Bit	Field	Value	Description
31-7	Reserved	0	Reads return 0. Writes have no effect.
6-0	BN	0-7Fh	Buffer number. 7-bit value of last completely transferred message buffer from system memory to the communication controller to the system memory.

26.3.1.7 Transfer Base Address (TBA)

The Transfer Base Address register holds a 32-bit aligned 32-bit base-pointer, which defines the base address for the data to be transferred.

NOTE: A write to this register also modifies the NTBA register.

Figure 26-39. Transfer Base Address (TBA) [offset_TU = 24h]

31	16
TBA	
R/W-0	

15	0
TBA	
R/W-0	

LEGEND: R/W = Read/Write; -*n* = value after reset

Table 26-25. Transfer Base Address (TBA) Field Descriptions

Bit	Field	Description
31-0	TBA	Transfer Base Address. 32-bit base pointer, 2 LSB are not significant (32-bit accesses only) and will always be 0.

26.3.1.8 Next Transfer Base Address (NTBA)

The Next Transfer Base Address hold a 32-bit aligned 32-bit base-pointer to be loaded into TBA during next cycle start.

NOTE: A write on TBA register also modifies the NTBA register.

Figure 26-40. Next Transfer Base Address (NTBA) [offset_TU = 28h]

31	16
NTBA	
R/W-0	

15	0
NTBA	
R/W-0	

LEGEND: R/W = Read/Write; -*n* = value after reset

Table 26-26. Next Transfer Base Address (NTBA) Field Descriptions

Bit	Field	Description
31-0	NTBA	Next Transfer Base Address. 32-bit base pointer, 2 LSB are not significant (32-bit accesses only) will always be 0.

26.3.1.9 Base Address of Mirrored Status (BAMS)

The Base Address of Mirrored Status hold a 32-bit aligned 32-bit base-pointer to be use for mirror transactions. Further details about the transfer mirror mechanism can be found in Section 26.2.1.1.1.7.

Figure 26-41. Base Address of Mirrored Status (BAMS) [offset_TU = 2Ch]

31 … 16
BAMS
R/W-0

15 … 0
BAMS
R/W-0

LEGEND: R/W = Read/Write; -*n* = value after reset

Table 26-27. Base Address of Mirrored Status (BAMS) Field Descriptions

Bit	Field	Description
31-0	BAMS	Base Address of Mirrored Status. 32-bit base pointer, 2 LSB are not significant (32-bit accesses only) will always be 0.

26.3.1.10 Start Address of Memory Protection (SAMP)

The Start Address of Memory Protection hold a 32-bit address.

Figure 26-42. Start Address of Memory Protection (SAMP) [offset_TU = 30h]

31	16
SAMP	
R/W-0	

15	0
SAMP	
R/W-0	

LEGEND: R/W = Read/Write; -*n* = value after reset

Table 26-28. Start Address of Memory Protection (SAMP) Field Descriptions

Bit	Field	Description
31-0	SAMP	Start Address Memory Protection. Start address of the memory area, which allows read and write accesses for the Transfer Unit State Machine. 32-bit base pointer, 2 LSB are not significant (32-bit accesses only) will always be 0.

26.3.1.11 End Address of Memory Protection (EAMP)

The End Address of Memory Protection hold a 32-bit address.

Figure 26-43. End Address of Memory Protection (EAMP) [offset_TU = 34h]

31	16
EAMP	
R/W-0	

15	0
EAMP	
R/W-0	

LEGEND: R/W = Read/Write; -*n* = value after reset

Table 26-29. End Address of Memory Protection (EAMP) Field Descriptions

Bit	Field	Description
31-0	EAMP	End Address Memory Protection. End address of the memory area, which allows read and write accesses for the Transfer Unit State Machine. 32-bit address, 2 LSB are not significant (32-bit accesses only) will always be 0.

26.3.1.12 Transfer to System Memory Occurred (TSMO[1-4])

The Transfer to System Memory Occurred register reflects the message buffer transfer status for a transfer transaction to the system memory. Four 32-bit registers reflect all possible 128 message buffers.

NOTE: Writing 1 will clear a bit. Writing 0 will leave a bit unchanged.

Figure 26-44. Transfer to System Memory Occurred 1 (TSMO1) [offset_TU = 40h]

31	16
TSMO1[31-16]	
R/W1C-0	

15	0
TSMO1[15-0]	
R/W1C-0	

LEGEND: R/W = Read/Write; W1C = Write 1 to clear; -*n* = value after reset

Figure 26-45. Transfer to System Memory Occurred 2 (TSMO2) [offset_TU = 44h]

31	16
TSMO2[63-48]	
R/W1C-0	

15	0
TSMO2[47-32]	
R/W1C-0	

LEGEND: R/W = Read/Write; W1C = Write 1 to clear; -*n* = value after reset

Figure 26-46. Transfer to System Memory Occurred 3 (TSMO3) [offset_TU = 48h]

31	16
TSMO3[95-80]	
R/W1C-0	

15	0
TSMO3[79-64]	
R/W1C-0	

LEGEND: R/W = Read/Write; W1C = Write 1 to clear; -*n* = value after reset

Figure 26-47. Transfer to System Memory Occurred 4 (TSMO4) [offset_TU = 4Ch]

31	16
TSMO4[127-112]	
R/W1C-0	

15	0
TSMO4[111-96]	
R/W1C-0	

LEGEND: R/W = Read/Write; W1C = Write 1 to clear; -*n* = value after reset

Table 26-30. Transfer to System Memory Occurred (TSMOn) Field Descriptions

Bit	Field	Value	Description
31-0	TSMO(1-4)[*n*]		Transfer to System Memory Occurred Register. The register bits correspond to message buffers 0 to 127. Each bit of the register reflects a finished message buffer transfer to the system memory.
		0	Read: No transfer occurred. Write: Bit is unchanged.
		1	Read: Transfer occurred. Write: Clears the bit.

26.3.1.13 Transfer to Communication Controller Occurred (TCCO[1-4])

The Transfer to Communication Controller Occurred reflects the message buffer transfer status for a VBUSP master transfer transaction from the system memory. Four 32-bit registers reflect all possible 128 message buffers.

NOTE: Writing 1 will clear a bit. Writing 0 will leave a bit unchanged.

Figure 26-48. Transfer to Communication Controller Occurred 1 (TCCO1) [offset_TU = 50h]

31	16
TCCO1[31-16]	
R/W1C-0	

15	0
TCCO1[15-0]	
R/W1C-0	

LEGEND: R/W = Read/Write; W1C = Write 1 to clear; -*n* = value after reset

Figure 26-49. Transfer to Communication Controller Occurred 2 (TCCO2) [offset_TU = 54h]

31	16
TCCO2[63-48]	
R/W1C-0	

15	0
TCCO2[47-32]	
R/W1C-0	

LEGEND: R/W = Read/Write; W1C = Write 1 to clear; -*n* = value after reset

Figure 26-50. Transfer to Communication Controller Occurred 3 (TCCO3) [offset_TU = 58h]

31	16
TCCO3[95-80]	
R/W1C-0	

15	0
TCCO3[79-64]	
R/W1C-0	

LEGEND: R/W = Read/Write; W1C = Write 1 to clear; -*n* = value after reset

Figure 26-51. Transfer to Communication Controller Occurred 4 (TCCO4) [offset_TU = 5Ch]

31	16
TCCO4[127-112]	
R/W1C-0	

15	0
TCCO4[111-96]	
R/W1C-0	

LEGEND: R/W = Read/Write; W1C = Write 1 to clear; -*n* = value after reset

Table 26-31. Transfer to Communication Controller Occurred (TCCOn) Field Descriptions

Bit	Field	Value	Description
31-0	TCCO(1-4)[*n*]		Transfer to Communication Controller Occurred Register. The register bits correspond to message buffers 0 to 127. Each bit of the register reflects a finished message buffer transfer from the system memory.
		0	Read: No transfer occurred. Write: Bit is unchanged.
		1	Read: Transfer occurred. Write: Clears the bit.

26.3.1.14 Transfer Occurred Offset (TOOFF)

The Transfer Occurred Offset register contains the offset vector to the highest prior pending transfer occurred interrupt and the transfer direction.

After a read access the transfer occurred flag is cleared and the register contents will be updated automatically.

Figure 26-52. Transfer Occurred Offset (TOOFF) [offset_TU = 60h]

31 … 16
Reserved
R-0

15 … 9	8	7 … 0
Reserved	TDIR	OFF
R-0	R-0	R-0

LEGEND: R = Read only; -*n* = value after reset

Table 26-32. Transfer Occurred Offset (TOOFF) Field Descriptions

Bit	Field	Value	Description
31-9	Reserved	0	Reads return 0. Writes have no effect.
8	TDIR		Transfer Direction. In case the same interrupt occurs for communication controller and Transfer Unit State Machine transfers the PRIO bit in the Global Control register decides about the higher priority.
		0	A transfer to System Memory occurred.
		1	A transfer to the Communication Controller occurred.
7-0	OFF		Offset Vector
		0	Offset not valid (no transfer occurred, interrupt pending).
		1h	Interrupt pending for buffer 0.
		2h	Interrupt pending for buffer 1.
		3h	Interrupt pending for buffer 2.
		:	:
		80h	Interrupt pending for buffer 127.
		81h-FFh	Reserved

26.3.1.15 TCR Single-Bit Error Status (TSBESTAT)

After an ECC single-bit error in the Transfer Configuration RAM (TCR) occurred, the SE flag is set and the affected address is stored in this register. The register is updated without regard to the ECC single-bit error correction activation in the ECC Control Register (ECC_CTRL).

The contents of this register is cleared automatically when reading the register.

NOTE: ECC single-bit error can only be indicated by the SE bit when ADR is cleared. Since the contents of ADR is undefined after reset, it is recommended to clear the register by reading it.

Figure 26-53. TCR Single-Bit Error Status (TSBESTAT) [offset_TU = 6Ch]

31	30	16
SE	Reserved	
R-0	R-0	

15	9	8	0
Reserved		ADR	
R-0		RC-U	

LEGEND: R = Read only; RC = Clear on read; U = value is undefined; -*n* = value after reset

Table 26-33. TCR Single-Bit Error Status (TSBESTAT) Field Descriptions

Bit	Field	Value	Description
31	SE		ECC Single-Bit Error. The flag signals an ECC single-bit error to the host. The flag is set when an ECC single-bit error in TCR is detected. The flag is set without regard to the single-bit error lock setting of ECC Control Register (ECC_CTRL). ECC multi-bit errors are indicated by a separate PE bit in the Transfer Error Interrupt Flag (TEIF) register.
		0	No ECC single-bit error occurred.
		1	ECC single-bit error occurred.
30-9	Reserved	0	Reads return 0. Writes have no effect.
8-0	ADR		Address of failing TCR word location. ADR[8-2] is the TCR word address where the ECC single-bit error occurred. ADR[1-0] are always driven as 00.

26.3.1.16 ECC Error Address (PEADR)

After an ECC multi-bit error in the Transfer Configuration RAM occurred, the affected address is stored in this not resettable register.

The contents of the ECC Error Address register as well as the PE bit in the Transfer Error Interrupt Flag (TEIF) register is cleared automatically when reading the ECC Error Address register.

NOTE: An ECC multi-bit error can only be indicated by the PE bit of TEIF register when PEADR is cleared. Since the contents of PEADR is undefined after reset, it is recommended to clear the register by reading it.

Figure 26-54. ECC Error Address (PEADR) [offset_TU = 70h]

31 9	8 0
Reserved	ADR
R-0	RC-U

LEGEND: R = Read only; RC = Clear on read; U = value is undefined; -*n* = value after reset

Table 26-34. ECC Error Address (PEADR) Field Descriptions

Bit	Field	Value	Description
31-9	Reserved	0	Reads return 0. Writes have no effect.
8-0	ADR	0-1FFh	Address of failing TCR location. ADR[8-2] is the TCR word address where the ECC multi-bit error occurred. ADR[1-0] are always driven as 11.

26.3.1.17 Transfer Error Interrupt Flag (TEIF)

The Transfer Error Interrupt Flag register includes the Transfer Unit error flags. The bits in the TEIF are cleared by writing a 1.

NOTE: Memory Protection Violation (MPV) and ECC Error (PE) interrupts are nonmaskable and can not be disabled. Therefore, the MPV and PE bits are not part of the Transfer Error Interrupt Enable Set/Reset (TEIRES/R) registers.

Figure 26-55. Transfer Error Interrupt Flag (TEIF) [offset_TU = 74h]

31-18	17	16
Reserved	MPV	PE
R-0	R/W1C-0	R/W1C-0

15-11	10-8	7	6-4	3-2	1	0
Reserved	RSTAT	RSVD	WSTAT	Reserved	TNR	FAC
R-0	R/W1C-0	R-0	R/W1C-0	R-0	R/W1C-0	R/W1C-0

LEGEND: R/W = Read/Write; R = Read only; W1C = Write 1 to clear (writing 0 has no effect); -*n* = value after reset

Table 26-35. Transfer Error Interrupt Flag (TEIF) Field Descriptions

Bit	Field	Value	Description
31-18	Reserved	0	Reads return 0. Writes have no effect.
17	MPV		Memory Protection Violation.
		0	No MPV occurred.
		1	MPV occurred.
16	PE		ECC Error. The flag signals an ECC multi-bit error to the host. The flag is set when an ECC multi-bit error in TCR is detected.
			Note: ECC single-bit errors in TCR are indicated by a separate SE bit in TCR Single-Bit Error Status (TSBESTAT).
		0	No ECC multi-bit error occurred.
		1	ECC multi-bit error occurred.
15-11	Reserved	0	Reads return 0. Writes have no effect.
10-8	RSTAT		Status of VBUS on read transfers.
		0	Success
		1h	Addressing error
		2h	Protection error
		3h	Timeout error
		4h	Data error
		5h	Unsupported addressing mode error
		6h	Reserved
		7h	Exclusive read failure
			Note: Any value other than 000 indicates a VBUS read error. The information of the specific VBUS fault is for debug reasons only and is not relevant for normal usage.
7	Reserved	0	Reads return 0. Writes have no effect.
6-4	WSTAT		Status of VBUS on write transfers.
		0	Success
		1h	Addressing error
		2h	Protection error
		3h	Timeout error
		4h	Reserved
		5h	Unsupported addressing mode error
		6h	Reserved

Table 26-35. Transfer Error Interrupt Flag (TEIF) Field Descriptions (continued)

Bit	Field	Value	Description
		7h	Exclusive write failure
			Note: Any value other than 000 indicates a VBUS read error. The information of the specific VBUS fault is for debug reasons only and is not relevant for normal usage.
3-2	Reserved	0	Reads return 0. Writes have no effect.
1	TNR		Transfer Not Ready.
		0	Transfer started and NTBA is loaded to TBA.
		1	Transfer is not ready on communication cycle start and therefore NTBA is not loaded to TBA.
0	FAC		Forbidden Access.
		0	No forbidden access occurred.
		1	A forbidden CPU access to IBF or OBF occurred when the Transfer Unit State Machine is enabled.

26.3.1.18 Transfer Error Interrupt Enable Set/Reset (TEIRES/TEIRER)

The Transfer Error Interrupt Enable Set controls the interrupt activation of interrupt line TU_Int1. An interrupt is generated if both the interrupt flag in TEIF and the corresponding bit in TEIRES are set.

Exceptions are the memory protection violation (MPV) and the ECC (PE) error, which are related to nonmaskable interrupts, and therefore are not part of the TEIRS/R registers. Those errors have private error lines (TU_MPV_err and TU_UCT_err), which can be connected to the Vectored Interrupt Module (VIM) and/or the Error Signaling Module (ESM). Refer to the device-specific data manual for more details about the signal hookup.

A Transfer Error Interrupt is enabled by writing 1 to TEIRES register and disabled by writing 1 to TIERER register. Writing of 0 has no effect. Reading from both addresses will result in the same value.

Figure 26-56. Transfer Error Interrupt Enable Set (TEIRES) [offset_TU = 78h]

<table>
<tr><td>31</td><td>16</td></tr>
<tr><td colspan="2">Reserved</td></tr>
<tr><td colspan="2">R-0</td></tr>
</table>

<table>
<tr><td>15</td><td>11</td><td>10</td><td>8</td><td>7</td><td>6</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td></tr>
<tr><td colspan="2">Reserved</td><td colspan="2">RSTATE</td><td>RSVD</td><td colspan="2">WSTATE</td><td colspan="2">Reserved</td><td>TNRE</td><td>FACE</td></tr>
<tr><td colspan="2">R-0</td><td colspan="2">R/WS-0</td><td>R-0</td><td colspan="2">R/WS-0</td><td colspan="2">R-0</td><td>R/WS-0</td><td>R/WS-0</td></tr>
</table>

LEGEND: R/W = Read/Write; R = Read only; S = Set; -*n* = value after reset

Table 26-36. Transfer Error Interrupt Enable Set (TEIRES)

Bit	Field	Value	Description
31-11	Reserved	0	Reads return 0. Writes have no effect.
10-8	RSTATE		Read Error Interrupt Generation (interrupt generation on VBUS read transfer errors).
		0	Interrupt generation on VBUS read transfer error is disabled.
		7h	Interrupt generation on VBUS read transfer error is enabled.
			Note: Any value different from 111 does not assure the interrupt error generation of all possible VBUS read errors.
7	Reserved	0	Reads return 0. Writes have no effect.
6-4	WSTATE		Write Error Interrupt Generation (interrupt generation on VBUS write transfer errors).
		0	Interrupt generation on VBUS write transfer error is disabled.
		7h	Interrupt generation on VBUS write transfer error is enabled.
			Note: Any value different from 111 does not assure the interrupt error generation of all possible VBUS read errors.
3-2	Reserved	0	Reads return 0. Writes have no effect.
1	TNRE		Transfer Not Ready Enable.
		0	TNR interrupt is disabled.
		1	TNR interrupt is enabled.
0	FACE		Forbidden Access Enable.
		0	FAC interrupt is disabled.
		1	FAC interrupt is enabled.

Figure 26-57. Transfer Error Interrupt Enable Reset (TEIRER) [offset_TU = 7Ch]

31 16
Reserved
R-0

15 11	10 8	7	6 4	3 2	1	0
Reserved	RSTATE	RSVD	WSTATE	Reserved	TNRE	FACE
R-0	R/WC-0	R-0	R/WC-0	R-0	R/WC-0	R/WC-0

LEGEND: R/W = Read/Write; R = Read only; C = Clear; -*n* = value after reset

Table 26-37. Transfer Error Interrupt Enable Reset (TEIRER)

Bit	Field	Value	Description
31-11	Reserved	0	Reads return 0. Writes have no effect.
10-8	RSTATE		Read Error Interrupt Generation (interrupt generation on VBUS read transfer errors).
		0	Interrupt generation on VBUS read transfer error is disabled.
		7h	Interrupt generation on VBUS read transfer error is enabled.
			Note: Any value different from 111 does not assure the interrupt error generation of all possible VBUS read errors.
7	Reserved	0	Reads return 0. Writes have no effect.
6-4	WSTATE		Write Error Interrupt Generation (interrupt generation on VBUS write transfer errors).
		0	Interrupt generation on VBUS write transfer error is disabled.
		7h	Interrupt generation on VBUS write transfer error is enabled.
			Note: Any value different from 111 does not assure the interrupt error generation of all possible VBUS read errors.
3-2	Reserved	0	Reads return 0. Writes have no effect.
1	TNRE		Transfer Not Ready Enable.
		0	TNR interrupt is disabled.
		1	TNR interrupt is enabled.
0	FACE		Forbidden Access Enable.
		0	FAC interrupt is disabled.
		1	FAC interrupt is enabled.

26.3.1.19 Trigger Transfer to System Memory Set/Reset (TTSMS[1-4]/TTSMR[1-4])

The Trigger Transfer to System Memory register selects the current message buffer for a Transfer Unit State Machine transfer transaction to system memory. Four 32-bit registers reflect all possible 128 message buffers.

The bits are set by writing 1 to TTSMSx and reset by writing 1 to TTSMRx or after the transfer occurred. Writing a 0 has no effect. Reading from both addresses will result in the same value.

Figure 26-58. Trigger Transfer to System Memory Set 1 (TTSMS1) [offset_TU = 80h]

31	16
TTSMS1[31-16]	
R/WS-0	

15	0
TTSMS1[15-0]	
R/WS-0	

LEGEND: R/W = Read/Write; R = Read only; S = Set; -*n* = value after reset

Table 26-38. Trigger Transfer to System Memory Set 1 (TTSMS1) Field Descriptions

Bit	Field	Value	Description
31-0	TTSMS1[*n*]		Trigger Transfer to System Memory Set 1. The register bits 0 to 31 correspond to message buffers 0 to 31. Each bit of the register controls the message buffer transfer to the system memory in the following manner (not that only the least significant bit of all four combined TTSM registers will currently scheduled for transmission).
		0	No transfer request.
		1	Transfer based on address defined in TBA

Figure 26-59. Trigger Transfer to System Memory Reset 1 (TTSMR1) [offset_TU = 84h]

31	16
TTSMR1	
R/WC-0	

15	0
TTSMR1	
R/WC-0	

LEGEND: R/W = Read/Write; R = Read only; C = Clear; -*n* = value after reset

Table 26-39. Trigger Transfer to System Memory Reset 1 (TTSMR1) Field Descriptions

Bit	Field	Description
31-0	TTSMR1	Trigger Transfer to System Memory Reset 1. The TTSMR1 register shows the identical values to TTSMS1 if read.

Figure 26-60. Trigger Transfer to System Memory Set 2 (TTSMS2) [offset_TU = 88h]

31	16
TTSMS2[63-48]	
R/WS-0	

15	0
TTSMS2[47-32]	
R/WS-0	

LEGEND: R/W = Read/Write; R = Read only; S = Set; -*n* = value after reset

Table 26-40. Trigger Transfer to System Memory Set 2 (TTSMS2) Field Descriptions

Bit	Field	Value	Description
31-0	TTSMS2[*n*]		Trigger Transfer to System Memory Set 2. The register bits 0 to 31 correspond to message buffers 32 to 63. Each bit of the register controls the message buffer transfer to the system memory in the following manner (note that only the least-significant bit of all four combined TTSM registers will be currently scheduled for transmission).
		0	No transfer request.
		1	Transfer based on address defined in TBA

Figure 26-61. Trigger Transfer to System Memory Reset 2 (TTSMR2) [offset_TU = 8Ch]

31	16
TTSMR2	
R/WC-0	

15	0
TTSMR2	
R/WC-0	

LEGEND: R/W = Read/Write; R = Read only; C = Clear; -*n* = value after reset

Table 26-41. Trigger Transfer to System Memory Reset 2 (TTSMR2) Field Descriptions

Bit	Field	Description
31-0	TTSMR2	Trigger Transfer to System Memory Reset 2. The TTSMR2 register shows the identical values to TTSMS2 if read.

Figure 26-62. Trigger Transfer to System Memory Set 3 (TTSMS3) [offset_TU = 90h]

31	16
TTSMS3[95-80]	
R/WS-0	

15	0
TTSMS3[79-64]	
R/WS-0	

LEGEND: R/W = Read/Write; R = Read only; S = Set; -*n* = value after reset

Table 26-42. Trigger Transfer to System Memory Set 3 (TTSMS3) Field Descriptions

Bit	Field	Value	Description
31-0	TTSMS3[*n*]		Trigger Transfer to System Memory Set 3. The register bits 0 to 31 correspond to message buffers 64 to 95. Each bit of the register controls the message buffer transfer to the system memory in the following manner (note that only the least-significant bit of all four combined TTSM registers will be currently scheduled for transmission).
		0	No transfer request.
		1	Transfer based on address defined in TBA.

Figure 26-63. Trigger Transfer to System Memory Reset 3 (TTSMR3) [offset_TU = 94h]

31	16
TTSMR3	
R/WC-0	

15	0
TTSMR3	
R/WC-0	

LEGEND: R/W = Read/Write; R = Read only; C = Clear; -*n* = value after reset

Table 26-43. Trigger Transfer to System Memory Reset 3 (TTSMR3) Field Descriptions

Bit	Field	Description
31-0	TTSMR3	Trigger Transfer to System Memory Reset 3. The TTSMR3 register shows the identical values to TTSMS3 if read.

Figure 26-64. Trigger Transfer to System Memory Set 4 (TTSMS4) [offset_TU = 98h]

31	16
TTSMS4[127-112]	
R/WS-0	

15	0
TTSMS4[111-96]	
R/WS-0	

LEGEND: R/W = Read/Write; R = Read only; S = Set; -*n* = value after reset

Table 26-44. Trigger Transfer to System Memory Set 4 (TTSMS4) Field Descriptions

Bit	Field	Value	Description
31-0	TTSMS4[*n*]		Trigger Transfer to System Memory Set 4. The register bits 0 to 31 correspond to message buffers 96 to 127. Each bit of the register controls the message buffer transfer to the system memory in the following manner (note that only the least-significant bit of all four combined TTSM registers will be currently scheduled for transmission).
		0	No transfer request.
		1	Transfer based on address defined in TBA.

Figure 26-65. Trigger Transfer to System Memory Reset 4 (TTSMR4) [offset_TU = 9Ch]

31	16
TTSMR4	
R/WC-0	

15	0
TTSMR4	
R/WC-0	

LEGEND: R/W = Read/Write; R = Read only; C = Clear; -*n* = value after reset

Table 26-45. Trigger Transfer to System Memory Reset 4 (TTSMR4) Field Descriptions

Bit	Field	Description
31-0	TTSMR4	Trigger Transfer to System Memory Reset 4. The TTSMR4 register shows the identical values to TTSMS4 if read.

26.3.1.20 Trigger Transfer to Communication Controller Set/Reset (TTCCS[1-4]/TTCCR[1-4])

The Trigger Transfer to Communication Controller registers select the current message buffer for a Transfer Unit State Machine transfer transaction from system memory. Four 32-bit registers reflect all possible 128 message buffers.

The bits are set by writing 1 to TTCCSx and reset by writing 1 to TTCCRx or after the transfer occurred. Writing a 0 has no effect. Reading from both addresses will result in the same value.

Figure 26-66. Trigger Transfer to Communication Controller Set 1 (TTCCS1) [offset_TU = A0h]

31	16
TTCCS1[31-16]	
R/WS-0	

15	0
TTCCS1[15-0]	
R/WS-0	

LEGEND: R/W = Read/Write; R = Read only; S = Set; -*n* = value after reset

Table 26-46. Trigger Transfer to Communication Controller Set 1 (TTCCS1) Field Descriptions

Bit	Field	Value	Description
31-0	TTCCS1[*n*]		Trigger Transfer to Communication Controller Set 1. The register bits 0 to 31 correspond to message buffers 0 to 31. Each bit of the register controls the message buffer transfer to the communication controller in the following manner:
		0	No transfer request.
		1	Transfer based on address defined in TBA.

Figure 26-67. Trigger Transfer to Communication Controller Reset 1 (TTCCR1) [offset_TU = A4h]

31	16
TTCCR1	
R/WC-0	

15	0
TTCCR1	
R/WC-0	

LEGEND: R/W = Read/Write; R = Read only; C = Clear; -*n* = value after reset

Table 26-47. Trigger Transfer to Communication Controller Reset 1 (TTCCR1) Field Descriptions

Bit	Field	Description
31-0	TTCCR1	Trigger Transfer to Communication Controller Reset 1. The TTCCR1 register shows the identical values to TTCCS1 if read.

Figure 26-68. Trigger Transfer to Communication Controller Set 2 (TTCCS2) [offset_TU = A8h]

31	16
TTCCS2[63-48]	
R/WS-0	
15	0
TTCCS2[47-32]	
R/WS-0	

LEGEND: R/W = Read/Write; R = Read only; S = Set; -*n* = value after reset

Table 26-48. Trigger Transfer to Communication Controller Set 2 (TTCCS2) Field Descriptions

Bit	Field	Value	Description
31-0	TTCCS2[*n*]		Trigger Transfer to Communication Controller Set 2. The register bits 0 to 31 correspond to message buffers 32 to 63. Each bit of the register controls the message buffer transfer to the communication controller in the following manner.
		0	No transfer request.
		1	Transfer based on address defined in TBA.

Figure 26-69. Trigger Transfer to Communication Controller Reset 2 (TTCCR2) [offset_TU = ACh]

31	16
TTCCR2	
R/WC-0	
15	0
TTCCR2	
R/WC-0	

LEGEND: R/W = Read/Write; R = Read only; C = Clear; -*n* = value after reset

Table 26-49. Trigger Transfer to Communication Controller Reset 2 (TTCCR2) Field Descriptions

Bit	Field	Description
31-0	TTCCR2	Trigger Transfer to Communication Controller Reset 2. The TTCCR2 register shows the identical values to TTCCS2 if read.

Figure 26-70. Trigger Transfer to Communication Controller Set 3 (TTCCS3) [offset_TU = B0h]

31	16
TTCCS3[95-80]	
R/WS-0	

15	0
TTCCS3[79-64]	
R/WS-0	

LEGEND: R/W = Read/Write; R = Read only; S = Set; -*n* = value after reset

Table 26-50. Trigger Transfer to Communication Controller Set 3 (TTCCS3) Field Descriptions

Bit	Field	Value	Description
31-0	TTCCS3[*n*]		Trigger Transfer to Communication Controller Set 3. The register bits 0 to 31 correspond to message buffers 64 to 95. Each bit of the register controls the message buffer transfer to the communication controller in the following manner.
		0	No transfer request.
		1	Transfer based on address defined in TBA.

Figure 26-71. Trigger Transfer to Communication Controller Reset 3 (TTCCR3) [offset_TU = B4h]

31	16
TTCCR3	
R/WC-0	

15	0
TTCCR3	
R/WC-0	

LEGEND: R/W = Read/Write; R = Read only; C = Clear; -*n* = value after reset

Table 26-51. Trigger Transfer to Communication Controller Reset 3 (TTCCR3) Field Descriptions

Bit	Field	Description
31-0	TTCCR3	Trigger Transfer to Communication Controller Reset 3. The TTCCR3 register shows the identical values to TTCCS3 if read.

Figure 26-72. Trigger Transfer to Communication Controller Set 4 (TTCCS4) [offset_TU = B8h]

31	16
TTCCS4[127-112]	
R/WS-0	

15	0
TTCCS4[111-96]	
R/WS-0	

LEGEND: R/W = Read/Write; R = Read only; S = Set; -*n* = value after reset

Table 26-52. Trigger Transfer to Communication Controller Set 4 (TTCCS4) Field Descriptions

Bit	Field	Value	Description
31-0	TTCCS4[*n*]		Trigger Transfer to Communication Controller Set 4. The register bits 0 to 31 correspond to message buffers 96 to 127. Each bit of the register controls the message buffer transfer to the communication controller in the following manner:
		0	No transfer request.
		1	Transfer based on address defined in TBA.

Figure 26-73. Trigger Transfer to Communication Controller Reset 4 (TTCCR4) [offset_TU = BCh]

31	16
TTCCR4	
R/WC-0	

15	0
TTCCR4	
R/WC-0	

LEGEND: R/W = Read/Write; R = Read only; C = Clear; -*n* = value after reset

Table 26-53. Trigger Transfer to Communication Controller Reset 4 (TTCCR4) Field Descriptions

Bit	Field	Description
31-0	TTCCR4	Trigger Transfer to Communication Controller Reset 4. The TTCCR4 register shows the identical values to TTCCS4 if read.

26.3.1.21 Enable Transfer on Event to System Memory Set/Reset (ETESMS[1-4]/ETESMR[1-4])

The Enable Transfer on Event to System Memory Set registers enable a message buffer transfer to the system memory after a receive or transmit event. Four 32-bit registers reflect all possible 128 message buffers.

The bits are set by writing 1 to ETESMSx and reset by writing 1 to ETESMRx. Writing a 0 has no effect. Reading from both addresses will result in the same value.

Figure 26-74. Enable Transfer on Event to System Memory Set 1 (ETESMS1) [offset_TU = C0h]

31 ... 16
ETESMS1[31-16]
R/WS-0

15 ... 0
ETESMS1[15-0]
R/WS-0

LEGEND: R/W = Read/Write; R = Read only; S = Set; -*n* = value after reset

Table 26-54. Enable Transfer on Event to System Memory Set 1 Field Descriptions

Bit	Field	Value	Description
31-0	ETESMS1[*n*]		Enable Transfer on Event to System Memory Set 1. The register bits 0 to 31 correspond to message buffers 0 to 31. Each bit of the register enables a message buffer transfer on event to the system memory:
		0	Transfer on event is disabled.
		1	Transfer on event is enabled.

Figure 26-75. Enable Transfer on Event to System Memory Reset 1 (ETESMR1) [offset_TU = C4h]

31 ... 16
ETESMR1
R/WC-0

15 ... 0
ETESMR1
R/WC-0

LEGEND: R/W = Read/Write; R = Read only; C = Clear; -*n* = value after reset

Table 26-55. Enable Transfer on Event to System Memory Reset 1 (ETESMR1) Field Descriptions

Bit	Field	Description
31-0	ETESMR1	Enable Transfer on Event to System Memory Reset 1. The ETESMR1 register shows the identical values to ETESMS1 if read.

Figure 26-76. Enable Transfer on Event to System Memory Set 2 (ETESMS2) [offset_TU = C8h]

31 16
ETESMS2[63-48]
R/WS-0

15 0
ETESMS2[47-32]
R/WS-0

LEGEND: R/W = Read/Write; R = Read only; S = Set; -*n* = value after reset

Table 26-56. Enable Transfer on Event to System Memory Set 2 Field Descriptions

Bit	Field	Value	Description
31-0	ETESMS2[*n*]		Enable Transfer on Event to System Memory Set 2. The register bits 0 to 31 correspond to message buffers 32 to 63. Each bit of the register enables a message buffer transfer on event to the system memory:
		0	Transfer on event is disabled.
		1	Transfer on event is enabled.

Figure 26-77. Enable Transfer on Event to System Memory Reset 2 (ETESMR2) [offset_TU = CCh]

31 16
ETESMR2
R/WC-0

15 0
ETESMR2
R/WC-0

LEGEND: R/W = Read/Write; R = Read only; C = Clear; -*n* = value after reset

Table 26-57. Enable Transfer on Event to System Memory Reset 2 (ETESMR2) Field Descriptions

Bit	Field	Description
31-0	ETESMR2	Enable Transfer on Event to System Memory Reset 2. The ETESMR2 register shows the identical values to ETESMS2 if read.

Figure 26-78. Enable Transfer on Event to System Memory Set 3 (ETESMS3) [offset_TU = D0h]

31	16
ETESMS3[95-80]	
R/WS-0	

15	0
ETESMS3[79-64]	
R/WS-0	

LEGEND: R/W = Read/Write; R = Read only; S = Set; -*n* = value after reset

Table 26-58. Enable Transfer on Event to System Memory Set 3 Field Descriptions

Bit	Field	Value	Description
31-0	ETESMS3[*n*]		Enable Transfer on Event to System Memory Set 3. The register bits 0 to 31 correspond to message buffers 64 to 95. Each bit of the register enables a message buffer transfer on event to the system memory:
		0	Transfer on event is disabled.
		1	Transfer on event is enabled.

Figure 26-79. Enable Transfer on Event to System Memory Reset 3 (ETESMR3) [offset_TU = D4h]

31	16
ETESMR3	
R/WC-0	

15	0
ETESMR3	
R/WC-0	

LEGEND: R/W = Read/Write; R = Read only; C = Clear; -*n* = value after reset

Table 26-59. Enable Transfer on Event to System Memory Reset 3 (ETESMR3) Field Descriptions

Bit	Field	Description
31-0	ETESMR3	Enable Transfer on Event to System Memory Reset 3. The ETESMR3 register shows the identical values to ETESMS3 if read.

Figure 26-80. Enable Transfer on Event to System Memory Set 4 (ETESMS4) [offset_TU = D8h]

31	16
ETESMS4[127-112]	
R/WS-0	

15	0
ETESMS4[111-96]	
R/WS-0	

LEGEND: R/W = Read/Write; R = Read only; S = Set; -*n* = value after reset

Table 26-60. Enable Transfer on Event to System Memory Set 4 Field Descriptions

Bit	Field	Value	Description
31-0	ETESMS4[*n*]		Enable Transfer on Event to System Memory Set 4. The register bits 0 to 31 correspond to message buffers 96 to 127. Each bit of the register enables a message buffer transfer on event to the system memory:
		0	Transfer on event is disabled.
		1	Transfer on event is enabled.

Figure 26-81. Enable Transfer on Event to System Memory Reset 4 (ETESMR4) [offset_TU = DCh]

31	16
ETESMR4	
R/WC-0	

15	0
ETESMR4	
R/WC-0	

LEGEND: R/W = Read/Write; R = Read only; C = Clear; -*n* = value after reset

Table 26-61. Enable Transfer on Event to System Memory Reset 4 (ETESMR4) Field Descriptions

Bit	Field	Description
31-0	ETESMR4	Enable Transfer on Event to System Memory Reset 4. The ETESMR4 register shows the identical values to ETESMS4 if read.

26.3.1.22 Clear on Event to System Memory Set/Reset (CESMS[1-4]/CESMR[1-4])

The Clear on Event to System Memory registers disables an enabled transfer on event (enabled in ETESM) after a receive or transmit event. Four 32-bit registers reflect all possible 128 message buffers.

The bits are set by writing 1 to CESMSx and reset by writing 1 to CESMRx. Writing a 0 has no effect. Reading from both addresses will result in the same value.

Figure 26-82. Clear on Event to System Memory Set 1 (CESMS1) [offset_TU = E0h]

31	16
CESMS1[31-16]	
R/WS-0	

15	0
CESMS1[15-0]	
R/WS-0	

LEGEND: R/W = Read/Write; R = Read only; S = Set; -*n* = value after reset

Table 26-62. Clear on Event to System Memory Set 1 (CESMS1) Field Descriptions

Bit	Field	Value	Description
31-0	CESMS1[*n*]		Clear on Event to System Memory Set 1. The register bits 0 to 31 correspond to message buffers 0 to 31. Each bit of the register enables an automatic clear of the corresponding ETESM1 bit after a receive or transmit event:
		0	No clear.
		1	Activate clear.

Figure 26-83. Clear on Event to System Memory Reset 1 (CESMR1) [offset_TU = E4h]

31	16
CESMR1	
R/WC-0	

15	0
CESMR1	
R/WC-0	

LEGEND: R/W = Read/Write; R = Read only; C = Clear; -*n* = value after reset

Table 26-63. Clear on Event to System Memory Reset 1 (CESMR1) Field Descriptions

Bit	Field	Description
31-0	CESMR1	Clear on Event to System Memory Reset 1. The CESMR1 register shows the identical values to CESMS1 if read.

Figure 26-84. Clear on Event to System Memory Set 2 (CESMS2) [offset_TU = E8h]

31	16
CESMS2[63-48]	
R/WS-0	

15	0
CESMS2[47-32]	
R/WS-0	

LEGEND: R/W = Read/Write; R = Read only; S = Set; -*n* = value after reset

Table 26-64. Clear on Event to System Memory Set 2 (CESMS2) Field Descriptions

Bit	Field	Value	Description
31-0	CESMS2[*n*]		Clear on Event to System Memory Set 2. The register bits 0 to 31 correspond to message buffers 32 to 63. Each bit of the register enables an automatic clear of the corresponding ETESM2 bit after a receive or transmit event:
		0	No clear.
		1	Activate clear.

Figure 26-85. Clear on Event to System Memory Reset 2 (CESMR2) [offset_TU = ECh]

31	16
CESMR2	
R/WC-0	

15	0
CESMR2	
R/WC-0	

LEGEND: R/W = Read/Write; R = Read only; C = Clear; -*n* = value after reset

Table 26-65. Clear on Event to System Memory Reset 2 (CESMR2) Field Descriptions

Bit	Field	Description
31-0	CESMR2	Clear on Event to System Memory Reset 2. The CESMR2 register shows the identical values to CESMS2 if read.

Figure 26-86. Clear on Event to System Memory Set 3 (CESMS3) [offset_TU = F0h]

31	16
CESMS3[95-80]	
R/WS-0	

15	0
CESMS3[79-64]	
R/WS-0	

LEGEND: R/W = Read/Write; R = Read only; S = Set; -*n* = value after reset

Table 26-66. Clear on Event to System Memory Set 3 (CESMS3) Field Descriptions

Bit	Field	Value	Description
31-0	CESMS3[*n*]		Clear on Event to System Memory Set 3. The register bits 0 to 31 correspond to message buffers 64 to 95. Each bit of the register enables an automatic clear of the corresponding ETESM3 bit after a receive or transmit event:
		0	No clear.
		1	Activate clear.

Figure 26-87. Clear on Event to System Memory Reset 3 (CESMR3) [offset_TU = F4h]

31	16
CESMR3	
R/WC-0	

15	0
CESMR3	
R/WC-0	

LEGEND: R/W = Read/Write; R = Read only; C = Clear; -*n* = value after reset

Table 26-67. Clear on Event to System Memory Reset 3 (CESMR3) Field Descriptions

Bit	Field	Description
31-0	CESMR3	Clear on Event to System Memory Reset 3. The CESMR3 register shows the identical values to CESMS3 if read.

Figure 26-88. Clear on Event to System Memory Set 4 (CESMS4) [offset_TU = F8h]

31	16
CESMS4[127-112]	
R/WS-0	

15	0
CESMS4[111-96]	
R/WS-0	

LEGEND: R/W = Read/Write; R = Read only; S = Set; -*n* = value after reset

Table 26-68. Clear on Event to System Memory Set 4 (CESMS4) Field Descriptions

Bit	Field	Value	Description
31-0	CESMS4[*n*]		Clear on Event to System Memory Set 4. The register bits 0 to 31 correspond to message buffers 96 to 127. Each bit of the register enables an automatic clear of the corresponding ETESM4 bit after a receive or transmit event:
		0	No clear.
		1	Activate clear.

Figure 26-89. Clear on Event to System Memory Reset 4 (CESMR4) [offset_TU = FCh]

31	16
CESMR4	
R/WC-0	

15	0
CESMR4	
R/WC-0	

LEGEND: R/W = Read/Write; R = Read only; C = Clear; -*n* = value after reset

Table 26-69. Clear on Event to System Memory Reset 4 (CESMR4) Field Descriptions

Bit	Field	Description
31-0	CESMR4	Clear on Event to System Memory Reset 4. The CESMR4 register shows the identical values to CESMS4 if read.

26.3.1.23 Transfer to System Memory Interrupt Enable Set/Reset (TSMIES[1-4]/TSMIER[1-4])

The Transfer to System Memory Interrupt Enable registers enable the interrupt generation on interrupt line TU_Int0, after a transfer to the system memory occurred (flagged in TSMO). Four 32-bit Registers reflect all 128 MB's.

The bits are set by writing 1 to TSMIESx and reset by writing 1 to TSMIERx. Writing a 0 has no effect. Reading from both addresses will result in the same value.

Figure 26-90. Transfer to System Memory Interrupt Enable Set 1 (TSMIES1) [offset_TU = 100h]

31	16
TSMIES1[31-16]	
R/WS-0	

15	0
TSMIES1[15-0]	
R/WS-0	

LEGEND: R/W = Read/Write; R = Read only; S = Set; -*n* = value after reset

Table 26-70. Transfer to System Memory Interrupt Enable Set 1 (TSMIES1) Field Descriptions

Bit	Field	Value	Description
31-0	TTSMIES1[*n*]		Transfer to System Memory Interrupt Enable Set 1. The register bits 0 to 31 correspond to message buffers 0 to 31. Each bit of the register enables a potential interrupt, which occurs if the corresponding TSMO1 bit is set:
		0	No interrupt.
		1	Interrupt is generated.

Figure 26-91. Transfer to System Memory Interrupt Enable Reset 1 (TSMIER1) [offset_TU = 104h]

31	16
TSMIER1	
R/WC-0	

15	0
TSMIER1	
R/WC-0	

LEGEND: R/W = Read/Write; R = Read only; C = Clear; -*n* = value after reset

Table 26-71. Transfer to System Memory Interrupt Enable Reset 1 (TSMIER1) Field Descriptions

Bit	Field	Description
31-0	TSMIER1	Transfer to System Memory Interrupt Enable Reset 1. The TSMIER1 register shows the identical values to TSMIES1 if read.

Figure 26-92. Transfer to System Memory Interrupt Enable Set 2 (TSMIES2) [offset_TU = 108h]

31	16
TSMIES2[63-48]	
R/WS-0	

15	0
TSMIES2[47-32]	
R/WS-0	

LEGEND: R/W = Read/Write; R = Read only; S = Set; -*n* = value after reset

Table 26-72. Transfer to System Memory Interrupt Enable Set 2 (TSMIES2) Field Descriptions

Bit	Field	Value	Description
31-0	TSMIES2[*n*]		Transfer to System Memory Interrupt Enable Set 2. The register bits 0 to 31 correspond to message buffers 32 to 63. Each bit of the register enables a potential interrupt, which occurs if the corresponding TSMO2 bit is set:
		0	No interrupt.
		1	Interrupt is generated.

Figure 26-93. Transfer to System Memory Interrupt Enable Reset 2 (TSMIER2) [offset_TU = 10Ch]

31	16
TSMIER2	
R/WC-0	

15	0
TSMIER2	
R/WC-0	

LEGEND: R/W = Read/Write; R = Read only; C = Clear; -*n* = value after reset

Table 26-73. Transfer to System Memory Interrupt Enable Reset 2 (TSMIER2) Field Descriptions

Bit	Field	Description
31-0	TSMIER2	Transfer to System Memory Interrupt Enable Reset 2. The TSMIER2 register shows the identical values to TSMIES2 if read.

Figure 26-94. Transfer to System Memory Interrupt Enable Set 3 (TSMIES3) [offset_TU = 110h]

31	16
TSMIES3[95-80]	
R/WS-0	

15	0
TSMIES3[79-64]	
R/WS-0	

LEGEND: R/W = Read/Write; R = Read only; S = Set; -*n* = value after reset

Table 26-74. Transfer to System Memory Interrupt Enable Set 3 (TSMIES3) Field Descriptions

Bit	Field	Value	Description
31-0	TSMIES3[*n*]		Transfer to System Memory Interrupt Enable Set 3. The register bits 0 to 31 correspond to message buffers 64 to 95. Each bit of the register enables a potential interrupt, which occurs if the corresponding TSMO3 bit is set:
		0	No interrupt.
		1	Interrupt is generated.

Figure 26-95. Transfer to System Memory Interrupt Enable Reset 3 (TSMIER3) [offset_TU = 114h]

31	16
TSMIER3	
R/WC-0	

15	0
TSMIER3	
R/WC-0	

LEGEND: R/W = Read/Write; R = Read only; C = Clear; -*n* = value after reset

Table 26-75. Transfer to System Memory Interrupt Enable Reset 3 (TSMIER3) Field Descriptions

Bit	Field	Description
31-0	TSMIER3	Transfer to System Memory Interrupt Enable Reset 3. The TSMIER3 register shows the identical values to TSMIES3 if read.

Figure 26-96. Transfer to System Memory Interrupt Enable Set 4 (TSMIES4) [offset_TU = 118h]

31	16
TSMIES4[127-112]	
R/WS-0	

15	0
TSMIES4[111-96]	
R/WS-0	

LEGEND: R/W = Read/Write; R = Read only; S = Set; -*n* = value after reset

Table 26-76. Transfer to System Memory Interrupt Enable Set 4 (TSMIES4) Field Descriptions

Bit	Field	Value	Description
31-0	TSMIES4[*n*]		Transfer to System Memory Interrupt Enable Set 4. The register bits 0 to 31 correspond to message buffers 96 to 127. Each bit of the register enables a potential interrupt, which occurs if the corresponding TSMO4 bit is set:
		0	No interrupt.
		1	Interrupt is generated.

Figure 26-97. Transfer to System Memory Interrupt Enable Reset 4 (TSMIER4) [offset_TU = 11Ch]

31	16
TSMIER4	
R/WC-0	

15	0
TSMIER4	
R/WC-0	

LEGEND: R/W = Read/Write; R = Read only; C = Clear; -*n* = value after reset

Table 26-77. Transfer to System Memory Interrupt Enable Reset 4 (TSMIER4) Field Descriptions

Bit	Field	Description
31-0	TSMIER4	Transfer to System Memory Interrupt Enable Reset 4. The TSMIER4 register shows the identical values to TSMIES4 if read.

26.3.1.24 Transfer to Communication Controller Interrupt Enable Set/Reset (TCCIES[1-4]/TCCIER[1-4])

The Transfer to Communication Controller Interrupt Enable registers enables the interrupt generation on interrupt line TU_Int0, after a transfer to the communication controller occurred (flagged in TCCO). Four 32-bit Registers reflect all 128 MBs.

The bits are set by writing 1 to TCCIESx and reset by writing 1 to TCCIERx. Writing a 0 has no effect. Reading from both addresses will result in the same value.

Figure 26-98. Transfer to Communication Controller Interrupt Enable Set 1 (TCCIES1) [offset_TU = 120h]

31	16
TCCIES1[31-16]	
R/WS-0	

15	0
TCCIES1[15-0]	
R/WS-0	

LEGEND: R/W = Read/Write; R = Read only; S = Set; -*n* = value after reset

Table 26-78. Transfer to Communication Controller Interrupt Enable Set 1 (TCCIES1) Field Descriptions

Bit	Field	Value	Description
31-0	TCCIES1[*n*]		Transfer to Communication Controller Interrupt Enable Set 1. The register bits 0 to 31 correspond to message buffers 0 to 31. Each bit of the register enables a potential interrupt, which occurs if the corresponding TCCO1 bit is set:
		0	No interrupt.
		1	Interrupt is generated.

Figure 26-99. Transfer to Communication Controller Interrupt Enable Reset 1 (TCCIER1) [offset_TU = 124h]

31	16
TCCIER1	
R/WC-0	

15	0
TCCIER1	
R/WC-0	

LEGEND: R/W = Read/Write; R = Read only; C = Clear; -*n* = value after reset

Table 26-79. Transfer to Communication Controller Interrupt Enable Reset 1 (TCCIER1) Field Descriptions

Bit	Field	Description
31-0	TCCIER1	Transfer to Communication Controller Interrupt Enable Reset 1. The TCCIER1 register shows the identical values to TCCIES1 if read.

Figure 26-100. Transfer to Communication Controller Interrupt Enable Set 2 (TCCIES2) [offset_TU = 128h]

31	16
TCCIES2[63-48]	
R/WS-0	

15	0
TCCIES2[47-32]	
R/WS-0	

LEGEND: R/W = Read/Write; R = Read only; S = Set; -*n* = value after reset

Table 26-80. Transfer to Communication Controller Interrupt Enable Set 2 (TCCIES2) Field Descriptions

Bit	Field	Value	Description
31-0	TCCIES2[*n*]		Transfer to Communication Controller Interrupt Enable Set 2. The register bits 0 to 31 correspond to message buffers 32 to 63. Each bit of the register enables a potential interrupt, which occurs if the corresponding TCCO2 bit is set:
		0	No interrupt.
		1	Interrupt is generated.

Figure 26-101. Transfer to Communication Controller Interrupt Enable Reset 2 (TCCIER2) [offset_TU = 12Ch]

31	16
TCCIER2	
R/WC-0	

15	0
TCCIER2	
R/WC-0	

LEGEND: R/W = Read/Write; R = Read only; C = Clear; -*n* = value after reset

Table 26-81. Transfer to Communication Controller Interrupt Enable Reset 2 (TCCIER2) Field Descriptions

Bit	Field	Description
31-0	TCCIER2	Transfer to Communication Controller Interrupt Enable Reset 2. The TCCIER2 register shows the identical values to TCCIES2 if read.

Figure 26-102. Transfer to Communication Controller Interrupt Enable Set 3 (TCCIES3) [offset_TU = 130h]

31	16
TCCIES3[95-80]	
R/WS-0	

15	0
TCCIES3[79-64]	
R/WS-0	

LEGEND: R/W = Read/Write; R = Read only; S = Set; -*n* = value after reset

Table 26-82. Transfer to Communication Controller Interrupt Enable Set 3 (TCCIES3) Field Descriptions

Bit	Field	Value	Description
31-0	TCCIES3[*n*]		Transfer to Communication Controller Interrupt Enable Set 3. The register bits 0 to 31 correspond to message buffers 64 to 95. Each bit of the register enables a potential interrupt, which occurs if the corresponding TCCO3 bit is set:
		0	No interrupt.
		1	Interrupt is generated.

Figure 26-103. Transfer to Communication Controller Interrupt Enable Reset 3 (TCCIER3) [offset_TU = 134h]

31	16
TCCIER3	
R/WC-0	

15	0
TCCIER3	
R/WC-0	

LEGEND: R/W = Read/Write; R = Read only; C = Clear; -*n* = value after reset

Table 26-83. Transfer to Communication Controller Interrupt Enable Reset 3 (TCCIER3) Field Descriptions

Bit	Field	Description
31-0	TCCIER3	Transfer to Communication Controller Interrupt Enable Reset 3. The TCCIER3 register shows the identical values to TCCIES3 if read.

Figure 26-104. Transfer to Communication Controller Interrupt Enable Set 4 (TCCIES4) [offset_TU = 138h]

31	16
TCCIES4[127-112]	
R/WS-0	

15	0
TCCIES4[111-96]	
R/WS-0	

LEGEND: R/W = Read/Write; R = Read only; S = Set; -*n* = value after reset

Table 26-84. Transfer to Communication Controller Interrupt Enable Set 4 (TCCIES4) Field Descriptions

Bit	Field	Value	Description
31-0	TCCIES4[*n*]		Transfer to Communication Controller Interrupt Enable Set 4. The register bits 0 to 31 correspond to message buffers 96 to 127. Each bit of the register enables a potential interrupt, which occurs if the corresponding TCCO4 bit is set:
		0	No interrupt.
		1	Interrupt is generated.

Figure 26-105. Transfer to Communication Controller Interrupt Enable Reset 4 (TCCIER4) [offset_TU = 13Ch]

31	16
TCCIER4	
R/WC-0	

15	0
TCCIER4	
R/WC-0	

LEGEND: R/W = Read/Write; R = Read only; C = Clear; -*n* = value after reset

Table 26-85. Transfer to Communication Controller Interrupt Enable Reset 4 (TCCIER4) Field Descriptions

Bit	Field	Description
31-0	TCCIER4	Transfer to Communication Controller Interrupt Enable Reset 4. The TCCIER4 register shows the identical values to TCCIES4 if read.

26.3.1.25 Transfer Configuration RAM (TCR)

The TCR consists of 128 entries, each 19 bit wide. The TCR is ECC protected. The ECC protection can be switched on or off by the 4-bit key (PEL(3-0)) in the Global Control Set/Reset (GCS/R) registers.

Figure 26-106. Transfer Configuration RAM (TCR) [offset_TU_RAM = 0000h - 01FFh]

31-19	18	17	16
Reserved	STXR	THTSM	TPTSM
R-0	R/W-0	R/W-0	R/W-0

15	14	13-0
THTCC	TPTCC	TSO
R/W-0	R/W-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 26-86. Transfer Configuration RAM (TCR) Field Descriptions

Bit	Field	Value	Description
31-19	Reserved	0	Reads return 0. Writes have no effect.
18	STXR		Set Transmit Request. Control set/reset of buffer transmit requests in the communication controller.
		0	Transfer Unit State Machine will set IBCM.STXRH to 0 during a transfer to the communication controller.
		1	Transfer Unit State Machine will set IBCM.STXRH to 1 during a transfer to the communication controller.
17	THTSM		Transfer Header to System Memory.
		0	Transfer Unit State Machine will not transfer buffer header to system memory.
		1	Transfer Unit State Machine will transfer buffer header to system memory.
16	TPTSM		Transfer Payload to System Memory.
		0	Transfer Unit State Machine will not transfer buffer payload to system memory.
		1	Transfer Unit State Machine will transfer buffer payload to system memory.
15	THTCC		Transfer Header to Communication Controller.
		0	Transfer Unit State Machine will not transfer buffer header to the communication controller.
		1	Transfer Unit State Machine will transfer buffer header to the communication controller.
14	TPTCC		Transfer Payload to Communication Controller.
		0	Transfer Unit State Machine will not transfer buffer payload to the communication controller.
		1	Transfer Unit State Machine will transfer buffer payload to the communication controller.
13-0	TSO		Transfer Start Offset. 14-bit buffer address offset in system memory. The resulting address in system memory is computed by adding the 32-bit aligned buffer address offset (TSO = buffer address offset bits 15:2) to the base address defined in the TBA register. Example: A TSO contents of 0x40 results in a Transfer Start Offset of 0x40 × 4 = 0x100

26.3.1.26 TCR ECC Test Mode

In ECC test mode (diagnostic mode is enabled in ECC Control Register (ECC_CTRL)) the ECC information is visible and can be read or written. The corresponding TCR entry can be found by subtracting 0x200 from the TCR offset.

Figure 26-107. ECC Information in TCR ECC Test Mode [offset_TU_RAM = 200h-3FCh]

31 16
Reserved
R-0

15 6	5 0
Reserved	ECCINF(5-0)
R-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 26-87. ECC Information in TCR ECC Test Mode Field Descriptions

Bit	Field	Value	Description
31-6	Reserved	0	Reads return 0. Writes have no effect.
5-0	ECCINF(5-0)		ECC Data of TCR RAM locations.

26.3.2 Communication Controller Registers

The FlexRay Communication Controller module allocates an address space of 2 Kbytes (0000h to 07FFh). The registers are organized as 32-bit registers. 8/16-bit accesses are also supported. CPU access to the message RAM is done through the input and output buffers. They buffer data to be transferred to and from the message RAM under control of the message handler, avoiding conflicts between CPU accesses and message reception / transmission. Alternatively to increase performance the Transfer Unit can be used for transferring buffer data.

The test registers located on address 0010h and 0014h are writable only under the conditions.

The assignment of the message buffers is done according to the scheme shown in Figure 26-108. The number N of available message buffers depends on the payload length of the configured message buffers. The maximum number of message buffers is 128. The maximum payload length supported is 254 bytes.

The message buffers are separated into three consecutive groups; see Figure 26-108:

- Static buffers - Transmit / receive buffers assigned to static segment
- Static + Dynamic buffers - Transmit / receive buffers assigned to static or dynamic segment
- FIFO - Receive FIFO

The message buffer separation configuration can be changed in DEFAULT_CONFIG or CONFIG state only by programming register MRC.

The first group starts with message buffer 0 and consists of static message buffers only. Message buffer 0 is dedicated to hold the startup / sync frame or the single slot frame, if the node transmits one, as configured by SUCC1.TXST, SUCC1.TXSY, and SUCC1.TSM. In addition, message buffer 1 may be used for sync frame transmission in case that sync frames or single-slot frames should have different payloads on the two channels. In this case bit MRC.SPLM has to be programmed to 1 and message buffers 0 and 1 have to be configured with the key slot ID and can be (re)configured in DEFAULT_CONFIG or CONFIG state only.

The second group consists of message buffers assigned to the static or to the dynamic segment. Message buffers belonging to this group may be reconfigured during run time from dynamic to static or vice versa depending on the state of MRC.SEC.

The message buffers belonging to the third group are concatenated to a single receive FIFO.

Figure 26-108. Message Buffer Assignment

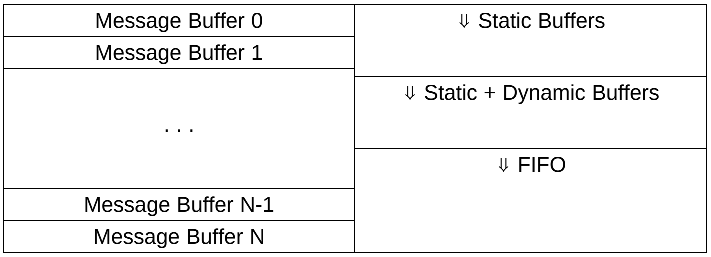


Table 26-88 provides a summary of the registers. The base address for the Communication Controller registers is FFF7 C800h.

Table 26-88. Communication Controller Registers

Offset	Acronym	Register Description	Section
		Special Registers	
00h	ECC_CTRL	ECC Control Register	Section 26.3.2.1.1
04h	ECCDSTAT	ECC Diagnostic Status Register	Section 26.3.2.1.2
08h	ECCTEST	ECC Test Register	Section 26.3.2.1.3
0Ch	SBESTAT	Single-Bit Error Status Register	Section 26.3.2.1.4
10h	TEST1	Test Register 1	Section 26.3.2.1.5
14h	TEST2	Test Register 2	Section 26.3.2.1.6
1Ch	LCK	Lock Register	Section 26.3.2.1.7
		Interrupt Registers	
20h	EIR	Error Interrupt Register	Section 26.3.2.2.1
24h	SIR	Status Interrupt Register	Section 26.3.2.2.2
28h	EILS	Error Interrupt Line Select Register	Section 26.3.2.2.3
2Ch	SILS	Status Interrupt Line Select Register	Section 26.3.2.2.4
30h	EIES	Error Interrupt Enable Set Register	Section 26.3.2.2.5
34h	EIER	Error Interrupt Enable Reset Register	Section 26.3.2.2.5
38h	SIES	Status Interrupt Enable Set Register	Section 26.3.2.2.6
3Ch	SIER	Status Interrupt Enable Reset Register	Section 26.3.2.2.6
40h	ILE	Interrupt Line Enable Register	Section 26.3.2.2.7
44h	T0C	Timer 0 Configuration Register	Section 26.3.2.2.8
48h	T1C	Timer 1 Configuration Register	Section 26.3.2.2.9
4Ch	STPW1	Stop Watch Register 1	Section 26.3.2.2.10
50h	STPW2	Stop Watch Register 2	Section 26.3.2.2.11
		Communication Controller Control Registers	
80h	SUCC1	SUC Configuration Register 1	Section 26.3.2.3.1
84h	SUCC2	SUC Configuration Register 2	Section 26.3.2.3.2
88h	SUCC3	SUC Configuration Register 3	Section 26.3.2.3.3
8Ch	NEMC	NEM Configuration Register	Section 26.3.2.3.4
90h	PRTC1	PRT Configuration Register 1	Section 26.3.2.3.5
94h	PRTC2	PRT Configuration Register 2	Section 26.3.2.3.6
98h	MHDC	MHD Configuration Register 1	Section 26.3.2.3.7
A0h	GTUC1	GTU Configuration Register 1	Section 26.3.2.3.8
A4h	GTUC2	GTU Configuration Register 2	Section 26.3.2.3.9
A8h	GTUC3	GTU Configuration Register 3	Section 26.3.2.3.10
ACh	GTUC4	GTU Configuration Register 4	Section 26.3.2.3.11
B0h	GTUC5	GTU Configuration Register 5	Section 26.3.2.3.12
B4h	GTUC6	GTU Configuration Register 6	Section 26.3.2.3.13
B8h	GTUC7	GTU Configuration Register 7	Section 26.3.2.3.14
BCh	GTUC8	GTU Configuration Register 8	Section 26.3.2.3.15
C0h	GTUC9	GTU Configuration Register 9	Section 26.3.2.3.16
C4h	GTUC10	GTU Configuration Register 10	Section 26.3.2.3.17
C8h	GTUC11	GTU Configuration Register 11	Section 26.3.2.3.18
		Communication Controller Status Registers	
100h	CCSV	Communication Controller Status Vector Register	Section 26.3.2.4.1
104h	CCEV	Communication Controller Error Vector Register	Section 26.3.2.4.2
110h	SCV	Slot Counter Value Register	Section 26.3.2.4.3
114h	MTCCV	Macrotick and Cycle Counter Value Register	Section 26.3.2.4.4
118h	RCV	Rate Correction Value Register	Section 26.3.2.4.5
11Ch	OCV	Offset Correction Value Register	Section 26.3.2.4.6

Table 26-88. Communication Controller Registers (continued)

Offset	Acronym	Register Description	Section
120h	SFS	Sync Frame Status Register	Section 26.3.2.4.7
124h	SWNIT	Symbol Window and NIT Status Register	Section 26.3.2.4.8
128h	ACS	Aggregated Channel Status Register	Section 26.3.2.4.9
130h-168h	ESIDn	Even Sync ID Register [1 to 15]	Section 26.3.2.4.10
170h-1A8h	OSIDn	Odd Sync ID Register [1 to 15]	Section 26.3.2.4.11
1B0h-1B8h	NMVn	Network Management Vector Register [1 to 3]	Section 26.3.2.4.12
		Message Buffer Control Registers	
300h	MRC	Message RAM Configuration Register	Section 26.3.2.5.1
304h	FRF	FIFO Rejection Filter Register	Section 26.3.2.5.2
308h	FRFM	FIFO Rejection Filter Mask Register	Section 26.3.2.5.3
30Ch	FCL	FIFO Critical Level Register	Section 26.3.2.5.4
		Message Buffer Status Registers	
310h	MHDS	Message Handler Status	Section 26.3.2.6.1
314h	LDTS	Last Dynamic Transmit Slot	Section 26.3.2.6.2
318h	FSR	FIFO Status Register	Section 26.3.2.6.3
31Ch	MHDF	Message Handler Constraints Flags	Section 26.3.2.6.4
320h	TXRQ1	Transmission Request Register 1	Section 26.3.2.6.5
324h	TXRQ2	Transmission Request Register 2	Section 26.3.2.6.5
328h	TXRQ3	Transmission Request Register 3	Section 26.3.2.6.5
32Ch	TXRQ4	Transmission Request Register 4	Section 26.3.2.6.5
330h	NDAT1	New Data Register 1	Section 26.3.2.6.6
334h	NDAT2	New Data Register 2	Section 26.3.2.6.6
338h	NDAT3	New Data Register 3	Section 26.3.2.6.6
33Ch	NDAT4	New Data Register 4	Section 26.3.2.6.6
340h	MBSC1	Message Buffer Status Changed Register 1	Section 26.3.2.6.7
344h	MBSC2	Message Buffer Status Changed Register 2	Section 26.3.2.6.7
348h	MBSC3	Message Buffer Status Changed Register 3	Section 26.3.2.6.7
34Ch	MBSC4	Message Buffer Status Changed Register 4	Section 26.3.2.6.7
		Identification Registers	
3F0h	CREL	Core Release Register	Section 26.3.2.7.1
3F4h	ENDN	Endian Register	Section 26.3.2.7.2
		Input Buffer	
400h-4FCh	WRDSn	Write Data Section Register [1 to 64]	Section 26.3.2.8.1
500h	WRHS1	Write Header Section Register 1	Section 26.3.2.8.2
504h	WRHS2	Write Header Section Register 2	Section 26.3.2.8.3
508h	WRHS3	Write Header Section Register 3	Section 26.3.2.8.4
510h	IBCM	Input Buffer Command Mask Register	Section 26.3.2.8.5
514h	IBCR	Input Buffer Command Request Register	Section 26.3.2.8.6
		Output Buffer	
600h-6FCh	RDDSn	Read Data Section Register [1 to 64]	Section 26.3.2.9.1
700h	RDHS1	Read Header Section Register 1	Section 26.3.2.9.2
704h	RDHS2	Read Header Section Register 2	Section 26.3.2.9.3
708h	RDHS3	Read Header Section Register 3	Section 26.3.2.9.4
70Ch	MBS	Message Buffer Status Register	Section 26.3.2.9.5
710h	OBCM	Output Buffer Command Mask Register	Section 26.3.2.9.6
714h	OBCR	Output Buffer Command Request Register	Section 26.3.2.9.7

26.3.2.1 Special Registers

26.3.2.1.1 ECC Control Register (ECC_CTRL)

ECC Control Register holds three 4-bit keys. SBEL to turn ECC single-bit error correction on or off, SBE_EVT_EN to enable a single-bit error event and DIAGSEL to enable the diagnostic mode to test the ECC single-bit error correction and double-bit error detection (SECDED) mechanism. Write access to key DIAGSEL is only possible in privilege mode.

Figure 26-109 and Table 26-89 illustrate this register.

NOTE: Diagnostic mode should be used only for RAM test purpose in RAM test mode. Therefore, when entering diagnostic mode, the FlexRay module should be in RAM test mode (TMC(1-0) set to 1 in Test Register 1 (TEST1)) before performing ECC testing.

Single-bit error correction can only be active when ECC is enabled.

Figure 26-109. ECC Control Register (ECC_CTRL) [offset_CC = 00h]

31 - 28	27 - 24	23 - 20	19 - 16
Reserved	SBE_EVT_EN	Reserved	SBEL
R-0	R/W-5h	R-0	R/W-Ah

15 - 4	3 - 0
Reserved	DIAGSEL
R-0	R/WP-Ah

LEGEND: R/W = Read/Write; R = Read only; WP = Write in Privilege Mode only; -*n* = value after reset

Table 26-89. ECC Control Register (ECC_CTRL) Field Descriptions

Bit	Field	Value	Description
31-28	Reserved	0	Reads return 0. Writes have no effect.
27-24	SBE_EVT_EN		ECC Single-Bit Error Indication.
		5h	ECC single-bit error indication is disabled. On ECC single-bit error detection when reading from message RAM, transient buffer RAMs, input buffer RAMs and output buffer RAMs, the single-bit error event signal of the communication controller (CC_SBE_err) is activated. On ECC single-bit error detection when reading from TCR, the single-bit error event signal of the transfer unit (TU_SBE_err) is activated.
		All other values	ECC single-bit error indication is enabled. On ECC single-bit error detection when reading from message RAM, transient buffer RAMs, input buffer RAMs and output buffer RAMs, the single-bit error event signal of the communication controller (CC_SBE_err) is not activated. On ECC single-bit error detection when reading from TCR, the single-bit error event signal of the transfer unit (TU_SBE_err) is not activated.
23-20	Reserved	0	Reads return 0. Writes have no effect.
19-16	SBEL		ECC Single-Bit Error Lock.
		5h	ECC single-bit error correction is turned off. ECC single-bit errors in the FlexRay RAMs do not get corrected and the ECC algorithm will detect up to 3 bits in error in a word.
		All other values	ECC single-bit error correction is turned on. ECC single-bit errors in the FlexRay RAMs get corrected.
15-4	Reserved	0	Reads return 0. Writes have no effect.
3-0	DIAGSEL		Diagnostic Mode select Key. The 4-bit key enables or disables the diagnostic mode.
		5h	Diagnostic mode is enabled. Double-bit errors will not trigger the peripheral ECC interrupt.
		All other values	Diagnostic mode is disabled. Double-bit errors will trigger the peripheral ECC interrupt.

26.3.2.1.2 ECC Diagnostic Status Register (ECCDSTAT)

ECC Diagnostic Status Register holds the ECC single-bit error and the double-bit error flags when in diagnostic mode. A flag is cleared by writing a 1 to it.

Figure 26-110 and Table 26-90 illustrate this register.

NOTE: In normal operation mode, double-bit errors are indicated in the Message Handler Status (MHDS) register, except for FTU RAM. A double-bit error in FTU RAM is indicated by the dedicated ECC error flag (PE) in the Transfer Error Interrupt Flag (TEIF) register.

Figure 26-110. ECC Diagnostic Status Register (ECCDSTAT) [offset_CC = 04h]

31							24
Reserved							
R-0							

23	22	21	20	19	18	17	16
DEFH	DEFG	DEFF	DEFE	DEFD	DEFC	DEFB	DEFA
R/W1C-0	R/W1C-0	R/W1C-0	R/W1C-0	R/W1C-0	R/W1C-0	R/W1C-0	R/W1C-0

15							8
Reserved							
R-0							

7	6	5	4	3	2	1	0
SEFH	SEFG	SEFF	SEFE	SEFD	SEFC	SEFB	SEFA
R/W1C-0	R/W1C-0	R/W1C-0	R/W1C-0	R/W1C-0	R/W1C-0	R/W1C-0	R/W1C-0

LEGEND: R/W = Read/Write; R = Read only; W1C = Write 1 to clear; -*n* = value after reset

Table 26-90. ECC Diagnostic Status Register (ECCDSTAT) Field Descriptions

Bit	Field	Value	Description
31-24	Reserved	0	Reads return 0. Writes have no effect.
23	DEFH		ECC Double-Bit Error Flag for FTU RAM.
		0	No ECC double-bit error is detected.
		1	ECC double-bit error is detected and diagnostic mode is enabled.
22	DEFG		ECC Double-Bit Error Flag for FlexRay Message RAM.
		0	No ECC double-bit error is detected.
		1	ECC double-bit error is detected and diagnostic mode is enabled.
21	DEFF		ECC Double-Bit Error Flag for Transient Buffer B RAM.
		0	No ECC double-bit error is detected.
		1	ECC double-bit error is detected and diagnostic mode is enabled.
20	DEFE		ECC Double-Bit Error Flag for Transient Buffer A RAM.
		0	No ECC double-bit error is detected.
		1	ECC double-bit error is detected and diagnostic mode is enabled.
19	DEFD		ECC Double-Bit Error Flag for Output Buffer 2 RAM.
		0	No ECC double-bit error is detected.
		1	ECC double-bit error is detected and diagnostic mode is enabled.
18	DEFC		ECC Double-Bit Error Flag for Output Buffer 1 RAM.
		0	No ECC double-bit error is detected.
		1	ECC double-bit error is detected and diagnostic mode is enabled.
17	DEFB		ECC Double-Bit Error Flag for Input Buffer 2 RAM.
		0	No ECC double-bit error is detected.
		1	ECC double-bit error is detected and diagnostic mode is enabled.

Table 26-90. ECC Diagnostic Status Register (ECCDSTAT) Field Descriptions (continued)

Bit	Field	Value	Description
16	DEFA		ECC Double-Bit Error Flag for Input Buffer 1 RAM.
		0	No ECC double-bit error is detected.
		1	ECC double-bit error is detected and diagnostic mode is enabled.
15-8	Reserved	0	Reads return 0. Writes have no effect.
7	SEFH		ECC Single-Bit Error Flag for FTU RAM.
		0	No ECC single-bit error is detected.
		1	ECC single-bit error is detected and diagnostic mode is enabled.
6	SEFG		ECC Single-Bit Error Flag for FlexRay Message RAM.
		0	No ECC single-bit error is detected.
		1	ECC single-bit error is detected and diagnostic mode is enabled.
5	SEFF		ECC Single-Bit Error Flag for Transient Buffer B RAM.
		0	No ECC single-bit error is detected.
		1	ECC single-bit error is detected and diagnostic mode is enabled.
4	SEFE		ECC Single-Bit Error Flag for Transient Buffer A RAM.
		0	No ECC single-bit error is detected.
		1	ECC single-bit error is detected and diagnostic mode is enabled.
3	SEFD		ECC Single-Bit Error Flag for Output Buffer 2 RAM.
		0	No ECC single-bit error is detected.
		1	ECC single-bit error is detected and diagnostic mode is enabled.
2	SEFC		ECC Single-Bit Error Flag for Output Buffer 1 RAM.
		0	No ECC single-bit error is detected.
		1	ECC single-bit error is detected and diagnostic mode is enabled.
1	SEFB		ECC Single-Bit Error Flag for Input Buffer 2 RAM.
		0	No ECC single-bit error is detected.
		1	ECC single-bit error is detected and diagnostic mode is enabled.
0	SEFA		ECC Single-Bit Error Flag for Input Buffer 1 RAM.
		0	No ECC single-bit error is detected.
		1	ECC single-bit error is detected and diagnostic mode is enabled.

26.3.2.1.3 ECC Test Register (ECCTEST)

The ECC Test Register can be used in diagnostic mode to read or write ECC information of message RAM, transient buffer RAM, input buffer RAM and output buffer RAM locations. Write access to this register is only possible when in diagnostic mode.

In order to be able to directly access the above mentioned RAM portions, RAM test mode must be selected in test register 1 and the corresponding RAM section must be selected in test register 2. When reading a certain RAM location, the corresponding ECC value is shown in RDECC bitfield. Writing to a certain ECC location copies the contents of WRECC bitfield to the corresponding ECC location.

Figure 26-111 and Table 26-91 illustrate this register.

NOTE: For FTU RAM, a separate portion of memory-mapped RAM is available in TCR ECC test mode, which can be accessed directly for reading or writing ECC information. See Section 26.3.1.26 for more details.

Figure 26-111. ECC Test Register (ECCTEST) [offset_CC = 08h]

31 23	22 16
Reserved	RDECC
R-0	R/W-0

15 7	6 0
Reserved	WRECC
R-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 26-91. ECC Test Register (ECCTEST) Field Descriptions

Bit	Field	Value	Description
31-23	Reserved	0	Reads return 0. Writes have no effect.
22-16	RDECC	0-7Fh	Holds ECC bits when reading a RAM location.
15-7	Reserved	0	Reads return 0. Writes have no effect.
6-0	WRECC	0-7Fh	ECC bits to be written in ECC location when writing to a RAM location.

26.3.2.1.4 Single-Bit Error Status Register (SBESTAT)

In normal operation mode, the Single-Bit Error Status Register indicates an ECC single-bit error by setting the SBE flag. In addition, it holds the faulty message buffer number and the indication of the buffer RAM when an ECC single-bit error occurred. The register is updated without regard to the single-bit error lock setting of ECC Control Register (ECC_CTRL).

A flag is cleared by writing a 1 to the corresponding bit position. Writing a 0 has no effect on the flag. A hardware reset or CHI command CLEAR_RAMS will also clear the register.

Figure 26-112 and Table 26-92 illustrate this register.

NOTE: An ECC single-bit error in the FTU RAM (TCR) is indicated by a dedicated TCR Single-Bit Error Status (TSBESTAT) register.

When one of the flags SBEFA, SBEFB, SBEFC, SBEFD, SBEFE, SBEFF and SBEFG changes from 0 to 1, the SBE flag is set to 1.

Figure 26-112. Single-Bit Error Status Register (SBESTAT) [offset_CC = 0Ch]

31	30 … 16
SBE	Reserved
R/W1C-0	R-0

15	14 … 8
Reserved	FMB
R-0	R/W1C-0

7	6	5	4	3	2	1	0
Reserved	MFMB	FMBD	STBF2	STBF1	SMR	SOBF	SIBF
R-0	R/W1C-0	R/W1C-0	R/W1C-0	R/W1C-0	R/W1C-0	R/W1C-0	R/W1C-0

LEGEND: R/W = Read/Write; R = Read only; W1C = Write 1 to clear; -*n* = value after reset

Table 26-92. Single-Bit Error Status Register (SBESTAT) Field Descriptions

Bit	Field	Value	Description
31	SBE		ECC Single-Bit Error. The flag signals an ECC single-bit error to the host. The flag is set by the ECC logic of the communication controller, when it detects an ECC single-bit error while reading from one of the FlexRay RAM blocks. This flag is set without regard to the SBEL bit setting in the ECC Control Register (ECC_CTRL). Note: ECC multi-bit errors are indicated by a separate PERR bit in the Error Interrupt Register (EIR).
		0	No ECC single-bit error occurred.
		1	ECC single-bit error occurred.
30-15	Reserved	0	Reads return 0. Writes have no effect.
14-8	FMB	0-7Fh	Faulty message buffer. An ECC single-bit error occurred when reading from a message buffer or when transferring data from Input Buffer or Transient Buffer 1,2 to the message buffer referenced by FMB. This value is only valid when one of the flags SIBF, SMR, STBF1, STBF2, and flag FMBD is set. It is not updated while flag FMBD is set.
7	Reserved	0	Reads return 0. Writes have no effect.
6	MFMB		Multiple message buffers with ECC single-bit error fault detected.
		0	No additional faulty message buffer.
		1	Another faulty message buffer was detected while flag FMBD is set.
5	FMBD		Message buffer with ECC single-bit error fault detected.
		0	No faulty message buffer.
		1	Message buffer referenced by FMB holds faulty data due to an ECC single-bit error.
4	STBF2		ECC single-bit error in transient buffer RAM B.
		0	No ECC single-bit error.
		1	ECC single-bit error occurred when reading transient buffer RAM B.

Table 26-92. Single-Bit Error Status Register (SBESTAT) Field Descriptions (continued)

Bit	Field	Value	Description
3	STBF1		ECC single-bit error in transient buffer RAM A.
		0	No ECC single-bit error.
		1	ECC single-bit error occurred when reading transient buffer RAM A.
2	SMR		ECC single-bit error in message RAM.
		0	No ECC single-bit error.
		1	ECC single-bit error occurred when reading message RAM.
1	SOBF		ECC single-bit error in output buffer RAM 1, 2.
		0	No ECC single-bit error.
		1	ECC single-bit error occurred when message handler read output buffer RAM 1,2.
0	SIBF		ECC single-bit error in input buffer RAM 1, 2.
		0	No ECC single-bit error.
		1	ECC single-bit error occurred when message handler read input buffer RAM 1,2.

26.3.2.1.5 Test Register 1 (TEST1)

Test register 1 holds the control bits to configure the test modes of the FlexRay module. Write access to these bits is only possible if the WRTEN bit is set. Figure 26-113 and Table 26-93 illustrate this register.

When the FlexRay module is operated in one of its test modes that requires WRTEN to be set (RAM Test Mode, I/O Test Mode, Asynchronous Transmit Mode, and Loop Back Mode) only the selected test mode functionality is available.

NOTE: To return from test mode operation to regular FlexRay operation we strongly recommend to apply a hardware reset (Power on Reset or nReset) to reset all FlexRay internal state machines to their initial state.

The test functions are not available in addition to the normal operational mode functions, they change the functions of parts of the FlexRay module. Therefore, normal operation as specified outside this chapter and as required by the FlexRay protocol specification and the FlexRay conformance test is not possible. Test mode functions may not be combined with each other or with FlexRay protocol functions.

NOTE: The FlexRay module should be kept in CONFIG state, while RAM Test Mode TMC = 01 is enabled.

The test mode features are intended for hardware testing or for FlexRay bus analyzer tools. They are not intended to be used in FlexRay applications.

Figure 26-113. Test Register 1 (TEST1) [offset_CC = 10h]

31 28	27 24	23 22	21	20	19	18	17	16
CERB	CERA	Reserved	TXENB	TXENA	TXB	TXA	RXB	RXA
R-0	R-0	R-0	R/W-0	R/W-0	R/W-0	R/W-0	R-0	R-0

15 10	9	8	7 6	5 4	3 2	1	0
Reserved	AOB	AOA	Reserved	TMC	Reserved	ELBE	WRTEN
R-0	R/W-1		R-0	R/W-0	R-0	R/W-0	R/W-0

LEGEND: R = Read only; R/W = Read/Write; -*n* = value after reset

Table 26-93. Test Register 1 (TEST1) Field Descriptions

Bit	Field	Value	Description
31-28	CERB		Coding Error Report Channel B.
			Set when a coding error is detected on channel B. Reset to 0 when register TEST1 is read or written. Once the CERB is set it will remain unchanged until the Host accesses the TEST1 register.
		0	No coding error is detected.
		1h	Header CRC error is detected.
		2h	Frame CRC error is detected.
		3h	Frame Start Sequence FSS too long.
		4h	First bit of Byte Start Sequence BSS seen LOW.
		5h	Second bit of Byte Start Sequence BSS seen HIGH.
		6h	First bit of Frame End Sequence FES seen HIGH.
		7h	Second bit of Frame End Sequence FES seen LOW.
		8h	CAS / MTS symbol seen too short.
		9h	CAS / MTS symbol seen too long.
		Ah-Fh	Reserved

Table 26-93. Test Register 1 (TEST1) Field Descriptions (continued)

Bit	Field	Value	Description
27-24	CERA		Coding Error Report Channel A.
			Set when a coding error is detected on channel A. Reset to 0 when register TEST1 is read or written. Once the CERA is set it will remain unchanged until the Host accesses the TEST1 register
		0	No coding error is detected.
		1h	Header CRC error is detected.
		2h	Frame CRC error is detected.
		3h	Frame Start Sequence FSS too long.
		4h	First bit of Byte Start Sequence BSS seen LOW.
		5h	Second bit of Byte Start Sequence BSS seen HIGH.
		6h	First bit of Frame End Sequence FES seen HIGH.
		7h	Second bit of Frame End Sequence FES seen LOW.
		8h	CAS / MTS symbol seen too short.
		9h	CAS / MTS symbol seen too long.
		Ah-Fh	Reserved
			Note: Coding errors are also signaled when the communication controller is in MONITOR_MODE. The error codes regarding CAS / MTS symbols concern only the monitored bit pattern, irrelevant whether those bit patterns occurred in the symbol window or elsewhere.
23-22	Reserved	0	Reads return 0. Writes have no effect.
21	TXENB		Control of channel B transmit enable pin.
		0	txen2 pin drives a 0.
		1	txen2 pin drives a 1.
20	TXENA		Control of channel A transmit enable pin.
		0	txen1 pin drives a 0.
		1	txen1 pin drives a 1.
19	TXB		Control of channel B transmit pin.
		0	txd2 pin drives a 0.
		1	txd2 pin drives a 1.
18	TXA		Control of channel A transmit pin.
		0	txd1 pin drives a 0.
		1	txd1 pin drives a 1.
17	RXB		Monitor channel B receive pin.
		0	rxd2 = 0
		1	rxd2 = 1
16	RXA		Monitor channel A receive pin.
		0	rxd1 = 0
		1	rxd1 = 1
15-10	Reserved	0	Reads return 0. Writes have no effect.
9	AOB		Activity on B. The channel idle condition is specified in the FlexRay protocol spec v2.1, BITSTRB process.
		0	No activity is detected, channel B is idle.
		1	Activity is detected, channel B is not idle.
8	AOA		Activity on A. The channel idle condition is specified in the FlexRay protocol spec v2.1, BITSTRB process.
		0	No activity is detected, channel A is idle.
		1	Activity is detected, channel A is not idle.
7-6	Reserved	0	Reads return 0. Writes have no effect.

Table 26-93. Test Register 1 (TEST1) Field Descriptions (continued)

Bit	Field	Value	Description
5-4	TMC		Test mode control.
		0	Normal operation mode, default.
		1h	RAM test mode. All RAM blocks of the FlexRay module are directly accessible by the host. This mode is intended to enable testing of the embedded RAM blocks during production testing.
		2h	I/O test mode. The output pins txd1, txd2, txen1, txen2 are driven to the values defined by bits TXA, TXB, TXENA, TXENB. The values applied to the input pins rxd1, rxd2 can be read from register bits RXA, RXB.
		3h	Unused. Mapped to normal operation mode.
3-2	Reserved	0	Reads return 0. Writes have no effect.
1	ELBE		External Loop Back Enable. There are two possibilities to perform a loop back test. External loop back via physical layer or internal loop back for in-system self-test (default). In case of an internal loop back pins txen1,2 are in their inactive state, pins txd1,2 are set to HIGH, pins rxd1,2 are not evaluated. Bit ELBE is evaluated only when POC is in loop back mode and test mode control is in normal operation mode TMC = 00.
		0	Internal loop back (default).
		1	External loop back.
0	WRTEN		Write test register enable. Enables write access to the test registers. To set the bit from 0 to 1, the test mode key has to be written as defined in Lock Register (LCK). The unlock sequence is not required when WRTEN is kept at 1 while other bits of the register are changed. The bit can be reset to 0 at any time.
		0	Write access to the test register is disabled.
		1	Write access to the test register is enabled.

26.3.2.1.5.1 Asynchronous Transmit Mode (ATM)

The asynchronous transmit mode is entered by writing 1110 to the controller host interface command vector CMD in the SUC configuration register 1 (controller host interface command: ATM) while the communication controller is in CONFIG state and bit WRTEN in the test register 1 is set to 1. When called in any other state or when bit WRTEN is not set, CMD will be reset to 0000 = command_not_accepted. POCS in the communication controller status vector will show 00 1110 while the FlexRay module is in ATM mode.

Asynchronous transmit mode can be left by writing 0001 (controller host interface command: CONFIG) to the controller host interface command vector CMD in the SUC configuration register 1.

In ATM mode transmission of a FlexRay frame is triggered by writing the number of the corresponding message buffer to the input buffer command request register while bit STXR in the input buffer command mask register is set to 1. In this mode wakeup, startup, and clock synchronization are bypassed, the controller host interface command SEND_MTS results in the immediate transmission of a MTS symbol.

MTS symbols received while operating in ATM mode will set the status interrupt flags MTSA,B in the Status Interrupt Register like in monitor mode.

26.3.2.1.5.2 Loop Back Mode

The loop back mode is entered by writing 1111 to the controller host interface command vector CMD(3-0) in the SUC configuration register 1 (controller host interface command: LOOP_BACK) while the communication controller is in CONFIG state and bit WRTEN in the test register 1 is set to 1. This write operation has to be directly preceded by two consecutive write accesses to the Configuration Lock Key (unlock sequence). When called in any other state or when bit WRTEN is not set, CMD will be reset to 0000 = command_not_accepted. POCS in the communication controller status vector will show 00 1101 while the FlexRay module is in loop back mode.

Loop back mode can be left by writing 0001 (controller host interface command: CONFIG) to the controller host interface command vector CMD in the SUC configuration register 1.

The loop back mode is intended to check the modules internal data paths. Normal, time triggered operation is not possible in loop back mode.

There are two possibilities to perform a loop back test. External loop back through the physical layer (TEST1.ELBE = 1) or internal loop back for in-system self-test (TEST1.ELBE = 0). In case of an internal loop back pins txen1,2_n are in their inactive state, pins txd1,2 are set, pins rxd1,2 are not evaluated.

When the communication controller is in loop back mode, a loop back test is started by the host writing a message to the input buffer and requesting the transmission by writing to the input buffer command request register. The message handler will transfer the message into the message RAM and then into the transient buffer of the selected channel. The channel protocol controller (PRT) will read (in 32-bit words) the message from the transmit part of the transient buffer and load it into its Rx / Tx shift register. The serial transmission is looped back into the shift register; its content is written into the receive part of the channels transient buffer before the next word is loaded.

The PRT and the message handler will then treat this transmitted message like a received message, perform an acceptance filtering on frame ID and receive channel, and store the message into the message RAM (assuming the message passed the acceptance filter, thus testing the acceptance filter logic). The loop back test ends with the host requesting this received message from the message RAM and then checking the contents of the output buffer.

Each FlexRay channel is tested separately. The FlexRay module cannot receive messages from the FlexRay bus while it is in the loop back mode.

The cycle counter value of frames used in loop back mode can be programmed by writing to the CCV bits of the MTCCV register (writable in ATM and loop back mode only).

NOTE: In case of an odd payload the last two bytes of the looped-back payload will be right aligned (shifted by 16 bits to the right) inside the last 32-bit data word.

The controller host interface command SEND_MTS results in the immediate transmission of an MTS symbol. Transmitted MTS symbols will not cause status interrupt flags MTSA,B to be set in the Status Interrupt Register. MTS symbols received while operating in loop back mode will set status interrupt flags MTSA,B in System Interrupt Register like in monitor mode. The reception of an MTS symbol can be emulated by driving the FlexRay receive pins RxD1,2 to low for the duration of the symbol in external loop back mode, or by driving the FlexRay pins TxD1,2 and TxEN1,2 to low using the TXA,B and TXENA,B of Test Register1 in internal or external loop back mode.

26.3.2.1.6 Test Register 2 (TEST2)

Test register 2 holds all bits required for RAM test of the embedded RAM blocks of the communication controller. Write access to this register is only possible when bit WRTEN in the test register 1 is set.

Figure 26-114 and Table 26-94 illustrate this register.

Figure 26-114. Test Register 2 (TEST2) [offset_CC = 14h]

31 16
Reserved
R-0

15	14	13 7	6 4	3	1 0
RDPB	WRPB	Reserved	SSEL	Rsvd	RS
R-0	R/W-0	R-0	R/W-0	R-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 26-94. Test Register 2 (TEST2) Field Descriptions

Bit	Field	Value	Description
31-16	Reserved	0	Reads return 0. Writes have no effect.
15	RDPB	0-1	When ECC mode is enabled, this bit is always read as 0.
14	WRPB	0-1	When ECC mode is enabled, this bit has no effect.
13-7	Reserved	0	Reads return 0. Writes have no effect.
6-4	SSEL		Segment select. To enable access to the complete message RAM (8192 byte addresses) the message RAM is segmented.
		0	Access to RAM bytes 0000h to 03FFh is enabled.
		1h	Access to RAM bytes 0400h to 07FFh is enabled.
		2h	Access to RAM bytes 0800h to 0BFFh is enabled.
		3h	Access to RAM bytes 0C00h to 0FFFh is enabled.
		4h	Access to RAM bytes 1000h to 13FFh is enabled.
		5h	Access to RAM bytes 1400h to 17FFh is enabled.
		6h	Access to RAM bytes 1800h to 1BFFh is enabled.
		7h	Access to RAM bytes 1C00h to 1FFFh is enabled.
3	Reserved	0	Reads return 0. Writes have no effect.
2-0	RS		RAM select. In RAM test mode, the RAM blocks selected by RS are mapped to module address 400h to 7FFh (1024 byte addresses).
		0	Input buffer RAM 1
		1h	Input buffer RAM 2
		2h	Output buffer RAM 1
		3h	Output buffer RAM 2
		4h	Transient buffer RAM A
		5h	Transient buffer RAM B
		6h	Message RAM
		7h	Reserved

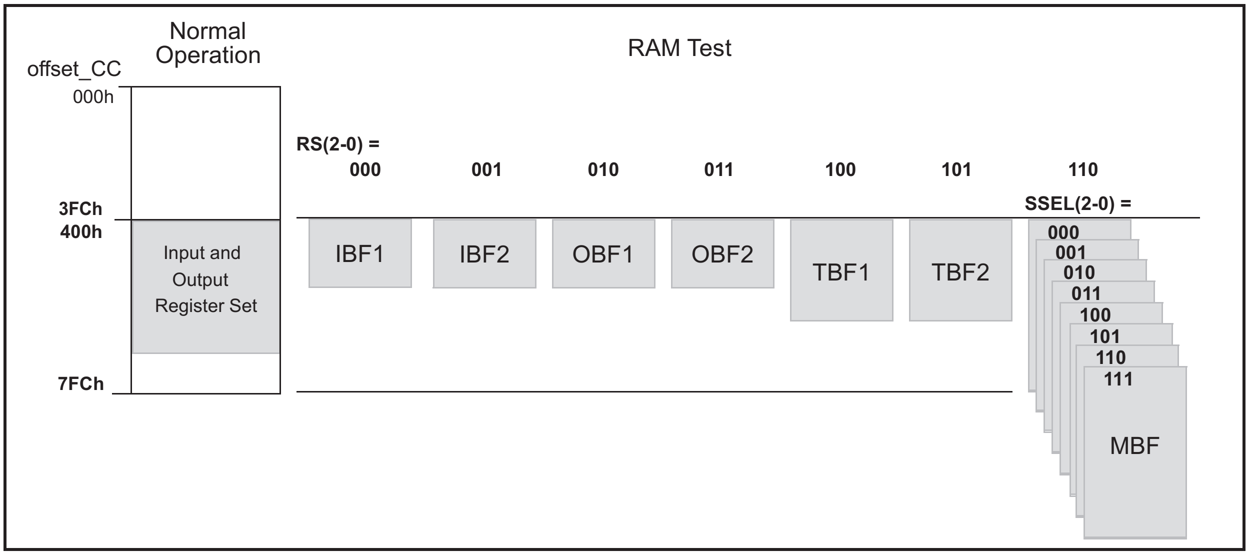
26.3.2.1.6.1 RAM Test Mode

In RAM test mode [TMC = 01], one of the seven RAM blocks can be selected for direct read and write access by programming the RS field to the corresponding value; see Figure 26-115.

For external RAM access in RAM test mode, the selected RAM block is mapped to the address range offset_CC 400h to 7FFh, which is the address space for the input and output buffer register sets in normal operation. Hence, the functionality of the input and output buffer register sets is not available in RAM test mode.

With the available address space (offset_CC 400h to 7FFh) in RAM test mode, 1024 bytes of RAM can be addressed for direct access. Since the length of the Message RAM exceeds the available address space, the Message RAM is segmented into segments of 1024 bytes. The segments can be selected by programming the bits SSEL(2-0) of test register 2.

Figure 26-115. Test Mode Access to Communication Controller RAM Blocks



26.3.2.1.7 Lock Register (LCK)

The lock register is write-only. Reading the register will return 00.

Figure 26-116 and Table 26-95 illustrate this register.

Figure 26-116. Lock Register (LCK) [offset_CC = 1Ch]

31	16
Reserved	
R-0	

15	8	7	0
TMK		CLK	
R/W-0		R/W-0	

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 26-95. Lock Register (LCK) Field Descriptions

Bit	Field	Value	Description
31-16	Reserved	0	Reads return 0. Writes have no effect.
15-8	TMK	0-1FFh	Test mode key. To write bit WRTEN in the test register to 1, the write operation has to be directly preceded by two consecutive write accesses to the test mode key (unlock sequence). If this write sequence is interrupted by other write accesses, bit WRTEN is not set to 1 and the sequence has to be repeated. First write (LCK.TMK): 75h = 0b0111 0101 Second write (LCK.TMK): 8Ah = 0b1000 1010 Third write: TEST1.WRTEN = 1
7-0	CLK	0-FFh	Configuration lock key. To leave CONFIG state by writing to CMD in the SUC configuration register 1 (commands READY; MONITOR_MODE; ATM; LOOP_BACK), the write operation has to be directly preceded by two consecutive write accesses to the configuration lock key (unlock sequence). If this write sequence is interrupted by other write accesses, the communication controller remains in CONFIG state and the sequence has to be repeated. First write (LCK.CLK): CEh = 0b1100 1110 Second write (LCK.CLK): 31h = 0b0011 0001 Third write (SUCC.CMD)

NOTE: In case that the Host uses 8/16-bit accesses to write the listed bit fields, the programmer has to ensure that no "dummy accesses" (for example, the remaining register bytes / words) are inserted by the compiler.

26.3.2.2 Interrupt Registers

26.3.2.2.1 Error Interrupt Register (EIR)

The flags are set when the communication controller detects one of the listed error conditions. They remain set until the host clears them. A flag is cleared by writing a 1 to the corresponding bit position. Writing a 0 has no effect on the flag. A reset will also clear the register.

Figure 26-117 and Table 26-96 illustrate this register.

Figure 26-117. Error Interrupt Register (EIR) [offset_CC = 20h]

31 27	26	25	24	23 19	18	17	16
Reserved	TABB	LTVB	EDB	Reserved	TABA	LTVA	EDA
R-0	R/W-0	R/W-0	R/W-0	R-0	R/W-0	R/W-0	R/W-0

15 12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	MHF	IOBA	IIBA	EFA	RFO	PERR	CCL	CCF	SFO	SFBM	CNA	PEMC
R-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 26-96. Error Interrupt Register (EIR) Field Descriptions

Bit	Field	Value	Description
31-27	Reserved	0	Reads return 0. Writes have no effect.
26	TABB		Transmission Across Boundary Channel B. The flag signals to the Host that a transmission across a slot boundary occurred for channel B.
		0	No transmission across slot boundary is detected on channel B.
		1	Transmission across slot boundary is detected on channel B.
25	LTVB		Latest transmit violation channel B. The flag signals a latest transmit violation on channel B to the host.
		0	No latest transmit violation is detected on channel B.
		1	Latest transmit violation is detected on channel B.
24	EDB		Error detected on channel B. This bit is set whenever one of the flags SEDB, CEDB, CIB, SBVB in the Aggregated channel status register is set.
		0	No error is detected on channel B.
		1	Error is detected on channel B.
23-19	Reserved	0	Reads return 0. Writes have no effect.
18	TABA		Transmission Across Boundary Channel A. The flag signals to the Host that a transmission across a slot boundary occurred for channel A.
		0	No transmission across slot boundary is detected on channel A.
		1	Transmission across slot boundary is detected on channel A.
17	LTVA		Latest transmit violation channel A. The flag signals a latest transmit violation on channel A to the host.
		0	No latest transmit violation is detected on channel A.
		1	Latest transmit violation is detected on channel A.
16	EDA		Error detected on channel A. This bit is set whenever one of the flags SEDA, CEDA, CIA, SBVA in the Aggregated channel status register is set.
		0	No error is detected on channel A.
		1	Error is detected on channel A.
15-12	Reserved	0	Reads return 0. Writes have no effect.
11	MHF		Message Handler Constraints Flag. The flag signals a Message Handler constraints violation condition. It is set whenever one of the flags MHDF.SNUA, MHDF.SNUB, MHDF.FNFA, MHDF.FNFB, MHDF.TBFA, MHDF.TBFB, MHDF.WAHP changes from 0 to 1.
		0	No Message Handler failure is detected.
		1	Message Handler failure is detected.

Table 26-96. Error Interrupt Register (EIR) Field Descriptions (continued)

Bit	Field	Value	Description
10	IOBA		Illegal Output buffer Access. This flag is set by the communication controller when the Host requests the transfer of a message buffer from the Message RAM to the Output Buffer while OBCR.OBSYS is set to 1.
		0	No illegal Host access to Output Buffer occurred.
		1	Illegal Host access to Output Buffer occurred.
9	IIBA		Illegal Input Buffer Access. This flag is set by the communication controller when the Host wants to modify a message buffer via Input Buffer and one of the following conditions applies: • The communication controller is not in CONFIG or DEFAULT_CONFIG state and the Host writes to the Input Buffer Command Request register to modify the following: – the Header section of message buffer 0, 1 if configured for transmission in key slot – the Header section of static message buffers with buffer number < MRC.FDB while MRC.SEC = 01 – the Header section of any static or dynamic message buffer while MRC.SEC = 1x – Header and / or data section of any message buffer belonging to the receive FIFO • The Host writes to any register of the Input Buffer while IBCR.IBSYH is set to 1.
		0	No illegal Host access to Input Buffer occurred.
		1	Illegal Host access to Input Buffer occurred.
8	EFA		Empty FIFO Access. This flag is set by the communication controller when the Host requests the transfer of a message from the receive FIFO via Output Buffer while the receive FIFO is empty.
		0	No Host access to empty FIFO occurred.
		1	Host access to empty FIFO occurred.
7	RFO		Receive FIFO overrun. This flag is set by the communication controller when a receive FIFO overrun was detected. The flag is cleared by the next FIFO read access of the host. After this read access one position in the FIFO is empty again.
		0	No receive FIFO overrun is detected.
		1	A receive FIFO overrun is detected.
6	PERR		ECC error. The flag signals an ECC multi-bit error to the host. The flag is set by the ECC logic of the communication controller, when it detects an ECC multi-bit error while reading from one of the FlexRay RAM blocks. Note: ECC single-bit errors are indicated by a separate SBE bit in the Single-Bit Error Status Register (SBESTAT).
		0	No ECC multi-bit error is detected.
		1	ECC multi-bit error is detected.
5	CCL		CHI Command Locked. The flag signals that the write access to the CHI command vector SUCC1.CMD was not successful because it coincided with a POC state change triggered by protocol functions. In this case bit CNA is also set to 1.
		0	CHI command is accepted.
		1	CHI command is not accepted.
4	CCF		Clock correction failure. This flag is set at the end of the cycle whenever one of the following errors occurred: • Missing rate correction signal • Missing offset correction signal • Clock correction Failed counter stopped at 15 • Clock correction Limit Reached The clock correction status is monitored in the communication controller error vector and sync frame status register.
		0	No clock correction error.
		1	Clock correction failed.
3	SFO		Sync frame overflow. Set when either the number of sync frames received during the last communication cycle or the total number of different sync frame IDs received during the last double cycle exceeds the maximum number of sync frames as defined by SNM in the GTU configuration register 2.
		0	Number of received sync frames in the configured range.
		1	More sync frames received than configured by SNM.

Table 26-96. Error Interrupt Register (EIR) Field Descriptions (continued)

Bit	Field	Value	Description
2	SFBM		Sync frames below minimum. This flag signals that the number of sync frames received during the last communication cycle was below the limit required by the FlexRay protocol. The minimum number of sync frames per communication cycle is 2.
		0	Two or more sync frames are received during last communication cycle.
		1	Less than two sync frames are received during last communication cycle.
1	CNA		Command not accepted. The flag signals that the controller host interface command vector CMD in the SUC configuration register 1 was reset to 0000 due to an unaccepted controller host interface command.
		0	Controller host interface command is accepted.
		1	Controller host interface command is not accepted.
0	PEMC		POC error mode changed. This flag is set whenever the error mode signaled by ERRM in the communication controller error vector register has changed.
		0	Error mode has not changed.
		1	Error mode has changed.

26.3.2.2.2 Status Interrupt Register (SIR)

The flags are set by the communication controller when a corresponding event occurs. They remain set until the host clears them. If enabled, an interrupt is pending while one of the bits is set. A flag is cleared by writing a 1 to the corresponding bit position. Writing a 0 has no effect on the flag. A hardware reset will also clear the register.

Figure 26-118 and Table 26-97 illustrate this register.

Figure 26-118. Status Interrupt Register (SIR) [offset_CC = 24h]

31–26	25	24	23–18	17	16
Reserved	MTSB	WUPB	Reserved	MTSA	WUPA
R-0	R/W-0	R/W-0	R-0	R/W-0	R/W-0

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SDS	MBSI	SUCS	SWE	TOBC	TIBC	TI1	TI0	NMVC	RFCL	RFNE	RXI	TXI	CYCS	CAS	WST
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 26-97. Status Interrupt Register (SIR) Field Descriptions

Bit	Field	Value	Description
31-26	Reserved	0	Reads return 0. Writes have no effect.
25	MTSB		MTS received on channel B. Media access test symbol received on channel B during the last symbol window. Updated by the communication controller for each channel at the end of the symbol window.
		0	No MTS symbol is received.
		1	MTS symbol is received.
24	WUPB		Wakeup pattern channel B. This flag is set by the communication controller when a wakeup pattern was received on channel B.
		0	No wakeup pattern is on channel B.
		1	Wakeup pattern is on channel B.
23-18	Reserved	0	Reads return 0. Writes have no effect.
17	MTSA		MTS received on channel A. Media access test symbol received on channel A during the last symbol window. Updated by the communication controller for each channel at the end of the symbol window.
		0	No MTS symbol is received.
		1	MTS symbol is received.

Table 26-97. Status Interrupt Register (SIR) Field Descriptions (continued)

Bit	Field	Value	Description
16	WUPA		Wakeup pattern channel A. This flag is set by the communication controller when a wakeup pattern was received on channel A.
		0	No wakeup pattern is on channel A.
		1	Wakeup pattern is on channel A.
15	SDS		Start of Dynamic Segment. This flag is set by the communication controller when the dynamic segment starts.
		0	Dynamic segment is not yet started.
		1	Dynamic segment is started.
14	MBSI		Message buffer status interrupt. This flag is set by the communication controller if bit MBI of a dedicated receive buffer is set to 1 and when the status of that message buffer has been updated due to reception of a: • valid frame with payload • valid frame with payload zero • null frame • corrupted frame or an empty slot
		0	No message buffer status has been updated.
		1	Message buffer status of at least one receive buffer has been updated.
13	SUCS		Startup completed successfully. This flag is set whenever a startup completed successfully and the communication controller entered NORMAL_ACTIVE state.
		0	No startup is completed successfully.
		1	Startup is completed successfully.
12	SWE		Stop watch event. If enabled by the respective control bits located in the Stop watch register, a detected edge on external stop watch pin or a software trigger event will generate a stop watch event.
		0	No stop watch event.
		1	Stop watch event occurred.
11	TOBC		Transfer output buffer completed. This flag is set whenever a transfer from the message RAM to the output buffer has completed and bit OBSYS in the output buffer command request register has been reset by the message handler.
		0	No transfer is completed since bit was reset.
		1	Transfer between message RAM and output buffer is completed.
10	TIBC		Transfer input buffer completed. This flag is set whenever a transfer from input buffer to the message RAM has completed and bit IBSYS in the input buffer command request register has been reset by the message handler.
		0	No transfer is completed since bit was reset.
		1	Transfer between input buffer and message RAM is completed.
9	TI1		Timer interrupt 1. This flag is set whenever the conditions programmed in the timer interrupt 1 configuration register are met. A timer interrupt 1 is also signaled on pin CC_tint1.
		0	No timer interrupt 1.
		1	Timer interrupt 1 occurred.
8	TI0		Timer interrupt 0. This flag is set whenever the conditions programmed in the timer interrupt 0 configuration register are met. A timer interrupt 0 is also signaled on pin CC_tint0.
		0	No timer interrupt 0.
		1	Timer interrupt 0 occurred.
7	NMVC		Network management vector changed. This interrupt flag signals a change in the Network management vector visible to the host.
		0	No change in the network management vector.
		1	Network management vector is changed.
6	RFCL		Receive FIFO critical level. This flag is set when the receive FIFO fill level FSR.RFFL is equal or greater than the critical level as configured by FCL.CL.
		0	Receive FIFO is below critical level.
		1	Receive FIFO critical level is reached.

Table 26-97. Status Interrupt Register (SIR) Field Descriptions (continued)

Bit	Field	Value	Description
5	RFNE		Receive FIFO not empty. This flag is set by the communication controller when a received valid frame was stored into the empty receive FIFO. The actual state of the receive FIFO is monitored in register FSR.
		0	Receive FIFO is empty.
		1	Receive FIFO is not empty.
4	RXI		Receive interrupt. This flag is set by the communication controller when the payload segment of a received valid frame was stored into the data section of a matching dedicated receive buffer and if bit MBI of that message buffer is set to 1.
		0	No data section has been updated.
		1	At least one data section has been updated.
3	TXI		Transmit interrupt. This flag is set by the communication controller after successful frame transmission if bit MBI in the respective message buffer is set to 1.
		0	No frame is transmitted.
		1	At least one frame was transmitted successfully.
2	CYCS		Cycle start interrupt. This flag is set by the communication controller when a communication cycle starts.
		0	No communication cycle is started.
		1	Communication cycle is started.
1	CAS		Collision avoidance symbol. This flag is set by the communication controller when a CAS was received.
		0	No CAS symbol is received.
		1	CAS symbol is received.
0	WST		This flag is set when WSV in the communication controller status vector register changes to a value other than UNDEFINED.
		0	Wakeup status is unchanged.
		1	Wakeup status is changed.

26.3.2.2.3 Error Interrupt Line Select (EILS)

The settings in the error interrupt line select register assigns an interrupt generated by a specific error interrupt flag to one of the two module interrupt lines (CC_int0 or CC_int1).

Figure 26-119 and Table 26-98 illustrate this register.

Figure 26-119. Error Interrupt Line Select Register (EILS) [offset_CC = 28h]

31 27	26	25	24	23 19	18	17	16
Reserved	TABBL	LTVBL	EDBL	Reserved	TABAL	LTVAL	EDAL
R-0	R/W-0	R/W-0	R/W-0	R-0	R/W-0	R/W-0	R/W-0

15 12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	MHFL	IOBAL	IIBAL	EFAL	RFOL	UCREL	CCLL	CCFL	SFOL	SFBML	CNAL	PEMCL
R-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 26-98. Error Interrupt Line Select Register (EILS) Field Descriptions

Bit	Field	Value	Description
31-27	Reserved	0	Reads return 0. Writes have no effect.
26	TABBL		Transmission across boundary channel B interrupt line.
		0	Interrupt is assigned to interrupt line CC_int0.
		1	Interrupt is assigned to interrupt line CC_int1.
25	LTVBL		Latest transmit violation channel B interrupt line.
		0	Interrupt is assigned to interrupt line CC_int0.
		1	Interrupt is assigned to interrupt line CC_int1.
24	EDBL		Error detected on channel B interrupt line.
		0	Interrupt is assigned to interrupt line CC_int0.
		1	Interrupt is assigned to interrupt line CC_int1.
23-19	Reserved	0	Reads return 0. Writes have no effect.
18	TABAL		Transmission across boundary channel A interrupt line.
		0	Interrupt is assigned to interrupt line CC_int0.
		1	Interrupt is assigned to interrupt line CC_int1.
17	LTVAL		Latest transmit violation channel A interrupt line.
		0	Interrupt is assigned to interrupt line CC_int0.
		1	Interrupt is assigned to interrupt line CC_int1.
16	EDAL		Error detected on channel A interrupt line.
		0	Interrupt is assigned to interrupt line CC_int0.
		1	Interrupt is assigned to interrupt line CC_int1.
15-12	Reserved	0	Reads return 0. Writes have no effect.
11	MHFL		Message handler constraints flag interrupt line.
		0	Interrupt is assigned to interrupt line CC_int0.
		1	Interrupt is assigned to interrupt line CC_int1.
10	IOBAL		Illegal output buffer access interrupt line.
		0	Interrupt is assigned to interrupt line CC_int0.
		1	Interrupt is assigned to interrupt line CC_int1.
9	IIBAL		Illegal output buffer access interrupt line.
		0	Interrupt is assigned to interrupt line CC_int0.
		1	Interrupt is assigned to interrupt line CC_int1.
8	EFAL		Empty FIFO access interrupt line.
		0	Interrupt is assigned to interrupt line CC_int0.
		1	Interrupt is assigned to interrupt line CC_int1.

Table 26-98. Error Interrupt Line Select Register (EILS) Field Descriptions (continued)

Bit	Field	Value	Description
7	RFOL		Receive FIFO overrun interrupt line.
		0	Interrupt is assigned to interrupt line CC_int0.
		1	Interrupt is assigned to interrupt line CC_int1.
6	UCREL		Uncorrectable RAM error interrupt line.
		0	Interrupt is assigned to interrupt line CC_int0.
		1	Interrupt is assigned to interrupt line CC_int1.
5	CCLL		CHI command locked interrupt line.
		0	Interrupt is assigned to interrupt line CC_int0.
		1	Interrupt is assigned to interrupt line CC_int1.
4	CCFL		Clock correction failure interrupt line.
		0	Interrupt is assigned to interrupt line CC_int0.
		1	Interrupt is assigned to interrupt line CC_int1.
3	SFOL		Sync frame overflow interrupt line.
		0	Interrupt is assigned to interrupt line CC_int0.
		1	Interrupt is assigned to interrupt line CC_int1.
2	SFBML		Sync frames below minimum interrupt line.
		0	Interrupt is assigned to interrupt line CC_int0.
		1	Interrupt is assigned to interrupt line CC_int1.
1	CNAL		Command not accepted interrupt line.
		0	Interrupt is assigned to interrupt line CC_int0.
		1	Interrupt is assigned to interrupt line CC_int1.
0	PEMCL		POC error mode changed interrupt line.
		0	Interrupt is assigned to interrupt line CC_int0.
		1	Interrupt is assigned to interrupt line CC_int1.

26.3.2.2.4 Status Interrupt Line Select (SILS)

The settings in the status interrupt line select register assign an interrupt generated by a specific status interrupt flag to one of the two module interrupt lines (CC_int0 or CC_int1).

Figure 26-120 and Table 26-99 illustrate this register.

Figure 26-120. Status Interrupt Line Select Register (SILS) [offset_CC = 2Ch]

31-26	25	24	23-18	17	16
Reserved	MTSBL	WUPBL	Reserved	MTSAL	WUPAL
R-0	R/W-1	R/W-1	R-0	R/W-1	R/W-1

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SDSL	MBSIL	SUCSL	SWEL	TOBCL	TIBCL	TI1L	TI0L	NMVCL	RFFL	RFNEL	RXIL	TXIL	CYCSL	CASL	WSTL
R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 26-99. Status Interrupt Line Select Register (SILS) Field Descriptions

Bit	Field	Value	Description
31-26	Reserved	0	Reads return 0. Writes have no effect.
25	MTSBL		Media access test symbol channel B interrupt line.
		0	Interrupt is assigned to interrupt line CC_int0.
		1	Interrupt is assigned to interrupt line CC_int1.
24	WUPBL		Wakeup pattern channel B interrupt line.
		0	Interrupt is assigned to interrupt line CC_int0.
		1	Interrupt is assigned to interrupt line CC_int1.
23-18	Reserved	0	Reads return 0. Writes have no effect.
17	MTSAL		Media access test symbol channel A interrupt line.
		0	Interrupt is assigned to interrupt line CC_int0.
		1	Interrupt is assigned to interrupt line CC_int1.
16	WUPAL		Wakeup pattern channel A interrupt line.
		0	Interrupt is assigned to interrupt line CC_int0.
		1	Interrupt is assigned to interrupt line CC_int1.
15	SDSL		Start of Dynamic Segment Interrupt Line
		0	Interrupt is assigned to interrupt line CC_int0.
		1	Interrupt is assigned to interrupt line CC_int1.
14	MBSIL		Message buffer status interrupt line.
		0	Interrupt is assigned to interrupt line CC_int0.
		1	Interrupt is assigned to interrupt line CC_int1.
13	SUCSL		Startup completed Successfully interrupt line.
		0	Interrupt is assigned to interrupt line CC_int0.
		1	Interrupt is assigned to interrupt line CC_int1.
12	SWEL		Stop watch event interrupt line.
		0	Interrupt is assigned to interrupt line CC_int0.
		1	Interrupt is assigned to interrupt line CC_int1.
11	TOBCL		Transfer output buffer completed interrupt line.
		0	Interrupt is assigned to interrupt line CC_int0.
		1	Interrupt is assigned to interrupt line CC_int1.
10	TIBCL		Transfer input buffer completed interrupt line.
		0	Interrupt is assigned to interrupt line CC_int0.
		1	Interrupt is assigned to interrupt line CC_int1.

Table 26-99. Status Interrupt Line Select Register (SILS) Field Descriptions (continued)

Bit	Field	Value	Description
9	TI1L		Timer interrupt 1 line.
		0	Interrupt is assigned to interrupt line CC_int0.
		1	Interrupt is assigned to interrupt line CC_int1.
8	TI0L		Timer interrupt 0 line.
		0	Interrupt is assigned to interrupt line CC_int0.
		1	Interrupt is assigned to interrupt line CC_int1.
7	NMVCL		Network management vector changed interrupt line.
		0	Interrupt is assigned to interrupt line CC_int0.
		1	Interrupt is assigned to interrupt line CC_int1.
6	RFCLL		Receive FIFO full interrupt line.
		0	Interrupt is assigned to interrupt line CC_int0.
		1	Interrupt is assigned to interrupt line CC_int1.
5	RFNEL		Receive FIFO not empty interrupt line.
		0	Interrupt is assigned to interrupt line CC_int0.
		1	Interrupt is assigned to interrupt line CC_int1.
4	RXIL		Receive interrupt line.
		0	Interrupt is assigned to interrupt line CC_int0.
		1	Interrupt is assigned to interrupt line CC_int1.
3	TXIL		Transmit interrupt line.
		0	Interrupt is assigned to interrupt line CC_int0.
		1	Interrupt is assigned to interrupt line CC_int1.
2	CYCSL		Cycle start interrupt line.
		0	Interrupt is assigned to interrupt line CC_int0.
		1	Interrupt is assigned to interrupt line CC_int1.
1	CASL		Collision Avoidance symbol interrupt line.
		0	Interrupt is assigned to interrupt line CC_int0.
		1	Interrupt is assigned to interrupt line CC_int1.
0	WSTL		Wakeup status interrupt line.
		0	Interrupt is assigned to interrupt line CC_int0.
		1	Interrupt is assigned to interrupt line CC_int1.

26.3.2.2.5 Error Interrupt Enable Set / Reset (EIES/EIER)

The settings in the error interrupt enable register determine which status changes in the error interrupt register will result in an interrupt. The enable bits are set by writing to EIES (address 30h) and reset by writing to EIER (address 34h). Writing 1 sets or resets the specific enable bit, writing 0 has no effect.

Figure 26-121 and Table 26-100 illustrate this register.

Figure 26-121. Error Interrupt Enable Set/Reset Register (EIES/EIER) [offset_CC = 30h/34h]

31-27	26	25	24	23-19	18	17	16
Reserved	TABBE	LTVBE	EDBE	Reserved	TABAE	LTVAE	EDAE
R-0	R/W-0	R/W-0	R/W-0	R-0	R/W-0	R/W-0	R/W-0

15-12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	MHFE	IOBAE	IIBAE	EFAE	RFOE	UCREE	CCLE	CCFE	SFOE	SFBME	CNAE	PEMCE
R-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 26-100. Error Interrupt Set/Reset Register (EIES/EIER) Field Descriptions

Bit	Field	Value	Description
31-27	Reserved	0	Reads return 0. Writes have no effect.
26	TABBE		Transmission across boundary channel B interrupt enable.
		0	Interrupt is disabled.
		1	Transmission across boundary channel B interrupt is enabled.
25	LTVBE		Latest transmit violation channel B interrupt enable.
		0	Interrupt is disabled.
		1	Latest transmit violation channel B interrupt is enabled.
24	EDBE		Error detected on channel B interrupt enable.
		0	Interrupt is disabled.
		1	Error detected on channel B interrupt is enabled.
23-19	Reserved	0	Reads return 0. Writes have no effect.
18	TABAE		Transmission across boundary channel A interrupt enable.
		0	Interrupt is disabled.
		1	Transmission across boundary channel A interrupt is enabled.
17	LTVAE		Latest transmit violation channel A interrupt enable.
		0	Interrupt is disabled.
		1	Latest transmit violation channel A interrupt is enabled.
16	EDAE		Error detected on channel A interrupt enable.
		0	Interrupt is disabled.
		1	Error detected on channel A interrupt is enabled.
15-12	Reserved	0	Reads return 0. Writes have no effect.
11	MHFE		Message handler constraints flag interrupt enable.
		0	Interrupt is disabled.
		1	Message handler constraints flag interrupt is enabled.
10	IOBAE		Illegal output buffer access interrupt enable.
		0	Interrupt is disabled.
		1	Illegal output buffer access interrupt is enabled.
9	IIBAE		Illegal input buffer access interrupt enable.
		0	Interrupt is disabled.
		1	Illegal input buffer access interrupt is enabled.
8	EFAE		Empty FIFO access interrupt enable.
		0	Interrupt is disabled.
		1	Empty FIFO access interrupt is enabled.

Table 26-100. Error Interrupt Set/Reset Register (EIES/EIER) Field Descriptions (continued)

Bit	Field	Value	Description
7	RFOE		Receive FIFO overrun interrupt enable.
		0	Interrupt is disabled.
		1	Receive FIFO overrun interrupt is enabled.
6	UCREE		Uncorrectable RAM error interrupt enable. An uncorrectable RAM error can be caused by: • an ECC single-bit error, if ECC single-bit error correction is disabled • an ECC double-bit error
		0	Interrupt is disabled.
		1	Uncorrectable RAM error interrupt is enabled.
5	CCLE		CHI command locked interrupt enable.
		0	Interrupt is disabled.
		1	CHI command locked interrupt is enabled.
4	CCFE		Clock correction failure interrupt enable.
		0	Interrupt is disabled.
		1	Clock correction failure interrupt is enabled.
3	SFOE		Sync frame overflow interrupt enable.
		0	Interrupt is disabled.
		1	Sync frame overflow interrupt is enabled.
2	SFBME		Sync frames below minimum interrupt enable.
		0	Interrupt is disabled.
		1	Sync frames below minimum interrupt is enabled.
1	CNAE		Command not Accepted interrupt enable.
		0	Interrupt is disabled.
		1	Command not valid interrupt is enabled.
0	PEMCE		POC error mode changed interrupt enable.
		0	Interrupt is disabled.
		1	Protocol error mode changed interrupt is enabled.

26.3.2.2.6 Status Interrupt Enable Set / Reset Register (SIES/SIER)

The settings in the status interrupt enable register determine which status changes in the status interrupt register will result in an interrupt. The enable bits are set by writing to SIES (address 38h) and reset by writing to SIER (address 3Ch). Writing 1 sets or resets the specific enable bit, writing 0 has no effect.

Figure 26-122 and Table 26-101 illustrate this register.

Figure 26-122. Status Interrupt Enable Set/Reset Register (SIES/SIER) [offset_CC = 38h/3Ch]

31-26	25	24	23-18	17	16
Reserved	MTSBE	WUPBE	Reserved	MTSAE	WUPAE
R-0	R/W-0	R/W-0	R-0	R/W-0	R/W-0

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SDSE	MBSIE	SUCSE	SWEE	TOBCE	TIBCE	TI1E	TI0E	NMVCE	RFFE	RFNEE	RXIE	TXIE	CYCSE	CASE	WSTE
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 26-101. Status Interrupt Enable Set/Reset Register (SIES/SIER) Field Descriptions

Bit	Field	Value	Description
31-26	Reserved	0	Reads return 0. Writes have no effect.
25	MTSBE		MTS received on channel B interrupt enable.
		0	Interrupt is disabled.
		1	MTS received on channel B interrupt is enabled.
24	WUPBE		Wakeup pattern channel B interrupt enable.
		0	Interrupt is disabled.
		1	Wakeup pattern channel B interrupt is enabled.
23-18	Reserved	0	Reads return 0. Writes have no effect.
17	MTSAE		MTS received on channel A interrupt enable.
		0	Interrupt is disabled.
		1	MTS received on channel A interrupt is enabled.
16	WUPAE		Wakeup pattern channel A interrupt enable.
		0	Interrupt is disabled.
		1	Wakeup pattern channel A interrupt is enabled.
15	SDSE		Start of dynamic segment interrupt enable.
		0	Interrupt is disabled.
		1	Start of dynamic segment interrupt is enabled.
14	MBSIE		Message buffer status interrupt enable.
		0	Interrupt is disabled.
		1	Message buffer status interrupt is enabled.
13	SUCSE		Startup completed successfully interrupt enable.
		0	Interrupt is disabled.
		1	Startup completed successfully interrupt is enabled.
12	SWEE		Stop watch event interrupt enable.
		0	Interrupt is disabled.
		1	Stop watch event interrupt is enabled.
11	TOBCE		Transfer output buffer completed interrupt enable.
		0	Interrupt is disabled.
		1	Transfer output buffer completed interrupt is enabled.
10	TIBCE		Transfer input buffer completed interrupt enable.
		0	Interrupt is disabled.
		1	Transfer input buffer completed interrupt is enabled.

Table 26-101. Status Interrupt Enable Set/Reset Register (SIES/SIER) Field Descriptions (continued)

Bit	Field	Value	Description
9	TI1E		Timer interrupt 1 enable.
		0	Interrupt is disabled.
		1	Timer interrupt 1 is enabled.
8	TI0E		Timer interrupt 0 enable.
		0	Interrupt is disabled.
		1	Timer interrupt 0 is enabled.
7	NMVCE		Network management vector changed interrupt enable.
		0	Interrupt is disabled.
		1	Network management vector changed interrupt is enabled.
6	RFCLE		Receive FIFO full interrupt enable.
		0	Interrupt is disabled.
		1	Receive FIFO overrun interrupt is enabled.
5	RFNEE		Receive FIFO not empty interrupt enable.
		0	Interrupt is disabled.
		1	Receive FIFO not empty interrupt is enabled.
4	RXIE		Receive interrupt enable.
		0	Interrupt is disabled.
		1	Receive interrupt is enabled.
3	TXIE		Transmit interrupt enable.
		0	Interrupt is disabled.
		1	Transmit interrupt is enabled.
2	CYCSE		Cycle start interrupt enable.
		0	Interrupt is disabled.
		1	Cycle start interrupt is enabled.
1	CASE		Collision avoidance symbol interrupt enable.
		0	Interrupt is disabled.
		1	Collision Avoidance symbol interrupt is enabled.
0	WSTE		Wakeup status interrupt enable.
		0	Interrupt is disabled.
		1	Wakeup status interrupt is enabled.

26.3.2.2.7 Interrupt Line Enable Register (ILE)

Each of the two interrupt lines (CC_int0, CC_int1) can be enabled separately by programming bit EINT0 and EINT1.

Figure 26-123 and Table 26-102 illustrate this register.

Figure 26-123. Interrupt Line Enable Register (ILE) [offset_CC = 40h]

31	16
Reserved	
R-0	

15	2	1	0
Reserved		EINT	
R-0		R/W-0	

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 26-102. Interrupt Line Enable Register (ILE) Field Descriptions

Bit	Field	Value	Description
31-2	Reserved	0	Reads return 0. Writes have no effect.
1-0	EINT		Enable interrupt line (1-0).
		0	Interrupt line CC_int1 and CC_int0 are disabled.
		1h	Interrupt line CC_int1 is disabled and CC_int0 is enabled.
		2h	Interrupt line CC_int1 is enabled and CC_int0 is disabled.
		3h	Interrupt line CC_int1 and CC_int0 are enabled.

26.3.2.2.8 Timer 0 Configuration Register (T0C)

This absolute timer specifies, in terms of cycle count and macrotick, the point in time when the timer 0 interrupt occurs. The timer 0 interrupt generates a non maskable interrupt signal on CC_tint0.

Timer 0 can be activated as long as the POC is either in NORMAL_ACTIVE state or in NORMAL_PASSIVE state. Timer 0 is deactivated when leaving NORMAL_ACTIVE state or NORMAL_PASSIVE state except for transitions between the two states.

Before reconfiguration of the timer, the timer has to be halted first by writing bit T0RC to 0.

Figure 26-124 and Table 26-103 illustrate this register.

Figure 26-124. Timer 0 Configuration Register (T0C) [offset_CC = 44h]

31-30	29-16
Reserved	TOMO
R-0	R/W-0

15	14-8	7-2	1	0
Rsvd	TOCC	Reserved	TOMS	TORC
R-0	R/W-0	R-0	R/W-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 26-103. Timer 0 Configuration Register (T0C) Field Descriptions

Bit	Field	Value	Description
31-30	Reserved	0	Reads return 0. Writes have no effect.
29-16	TOMO	0-3FFFh	Timer 0 macrotick offset. Configures the macrotick offset from the beginning of the cycle where the interrupt is to occur. The Timer 0 interrupt occurs at this offset for each cycle in the cycle set.
15	Reserved	0	Reads return 0. Writes have no effect.
14-8	TOCC	0-FFh	Timer 0 cycle code. The 7-bit timer 0 cycle code determines the cycle set used for generation of the timer 0 interrupt.
7-2	Reserved	0	Reads return 0. Writes have no effect.
1	TOMS		Time 0 mode select.
		0	Single-shot mode.
		1	Continuous mode.
0	TORC		Timer 0 run control.
		0	Timer 0 is halted.
		1	Timer 0 is running.

26.3.2.2.9 Timer 1 Configuration Register (T1C)

This relative timer generates an interrupt on the non maskable interrupt signal CC_tint1 after the specified number of macroticks has expired.

Timer 1 can be activated as long as the POC is either in NORMAL_ACTIVE state or in NORMAL_PASSIVE state. Timer 1 is deactivated when leaving NORMAL_ACTIVE state or NORMAL_PASSIVE state except for transitions between the two states.

Before reconfiguration of the timer, the timer has to be halted first by writing bit T1RC to 0.

Figure 26-125 and Table 26-104 illustrate this register.

Figure 26-125. Timer 1 Configuration Register (T1C) [offset_CC = 48h]

31-30	29-16
Reserved	TIMC
R-0	R/W-2h

15-2	1	0
Reserved	T1MS	T1RC
R-0	R/W-0	R/W-0

LEGEND: R = Read only; -*n* = value after reset

Table 26-104. Timer 1 Configuration Register (T1C) Field Descriptions

Bit	Field	Value	Description
31-30	Reserved	0	Reads return 0. Writes have no effect.
29-16	TIMC		Timer 1 macrotick count. When the configured macrotick count is reached the timer 1 interrupt is generated. In case the configured macrotick count is not within the valid range, timer 1 will not start. Valid values: • 2 to 16383 macroticks in continuous mode • 1 to 16383 macroticks in single-shot mode
		2h-3FFFh	Continuous mode.
		1h-3FFFh	Single-shot mode.
15-2	Reserved	0	Reads return 0. Writes have no effect.
1	T1MS		Timer 1 mode select.
		0	Single-shot mode.
		1	Continuous mode.
0	T1RC		Timer 1 run control.
		0	Timer 1 is halted.
		1	Timer 1 is running.

26.3.2.2.10 Stop Watch Register 1 Register (STPW1)

The stop watch is activated by an interrupt event (CC_int0 or CC_int1), by writing bit SSWT to 1, or by an external event.

With the macrotick counter increment following next to the stop watch activation the actual cycle counter and macrotick value is stored in the stop watch register 1 (stop watch event) and the slot counter values for channel A and channel B are stored in stop watch register 2.

Figure 26-126 and Table 26-105 illustrate this register.

Figure 26-126. Stop Watch Register 1 (STPW1) [offset_CC = 4Ch]

31-30	29-16
Reserved	SMTV
R-0	R-0

15-14	13-8	7	6	5	4	3	2	1	0
Reserved	SCCV	Rsvd	EINT1	EINT0	EETP	SSWT	EDGE	SWMS	ESWT
R-0	R-0	R-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 26-105. Stop Watch Register 1 (STPW1) Field Descriptions

Bit	Field	Value	Description
31-30	Reserved	0	Reads return 0. Writes have no effect.
29-16	SMTV	0-3E80h	Stopped macrotick value. State of the macrotick counter when the stop watch event occurred.
15-14	Reserved	0	Reads return 0. Writes have no effect.
13-8	SCCV	0-3Fh	Stopped cycle counter value. State of the cycle counter when the stop watch event occurred.
7	Reserved	0	Reads return 0. Writes have no effect.
6	EINT1		Enable interrupt 1 trigger. Enables stop watch trigger by CC_int1 event if ESWT = 1.
		0	Stop watch trigger by CC_int1 is disabled.
		1	CC_int1 event triggers stop watch.
5	EINT0		Enable interrupt 0 trigger. Enables stop watch trigger by CC_int0 event if ESWT = 1.
		0	Stop watch trigger by CC_int0 is disabled.
		1	CC_int0 event triggers stop watch.
4	EETP		Enable external trigger pin. Enables stop watch trigger event from external pin, if ESWT = 1.
		0	External trigger is disabled.
		1	Stop watch is activated by external trigger.
3	SSWT		Software stop watch trigger. When the host writes this bit to 1, the stop watch is activated. After the actual cycle counter and macrotick value are stored in the stop watch register, this bit is reset to 0. The bit is only writable while ESWT = 0.
		0	Software trigger is reset.
		1	Stop watch is activated by software trigger.
2	EDGE		Stop watch trigger edge select.
		0	Falling edge.
		1	Rising edge.
1	SWMS		Stop watch mode select.
		0	Single-shot mode.
		1	Continuous mode.
0	ESWT		External stop watch trigger. If enabled, an external event activates the stop watch. In single-shot mode, this bit is reset to 0 after the stop watch event occurred.
		0	External stop watch trigger is disabled.
		1	External stop watch trigger is enabled.

NOTE: Bits ESWT and SSWT cannot be set to 1 simultaneously. In this case the write access to the register is ignored, and both bits keep their previous values. Either the external stop watch trigger or the software stop watch trigger may be used.

The availability of an external stop watch pin is device dependant. Refer to the device data sheet for details.

26.3.2.2.11 Stop Watch Register 2 Register (STPW2)

Figure 26-127 and Table 26-106 illustrate this register.

Figure 26-127. Stop Watch Register 2 (STPW2) [offset_CC = 50h]

31-27	26-16
Reserved	SSCVB
R-0	R-0

15-11	10-0
Reserved	SSCVA
R-0	R-0

LEGEND: R = Read only; -*n* = value after reset

Table 26-106. Stop Watch Register 2 (STPW2) Field Descriptions

Bit	Field	Value	Description
31-27	Reserved	0	Reads return 0. Writes have no effect.
26-16	SSCVB	0-7FFh	Stop watch captured slot counter value channel B. State of the slot counter for channel B when the stop watch event occurred.
15-11	Reserved	0	Reads return 0. Writes have no effect.
10-0	SSCVA	0-7FFh	Stop watch captured slot counter value channel A. State of the slot counter for channel A when the stop watch event occurred.

26.3.2.3 Control Registers

This section describes the registers provided by the communication controller to allow the host to control the operation of the communication controller. The FlexRay protocol specification requires the host to write application configuration data in CONFIG state only.

NOTE: Be aware that the configuration registers are not locked for writing in DEFAULT_CONFIG state.

The configuration data is reset when DEFAULT_CONFIG state is entered from hardware reset. To change POC state from DEFAULT_CONFIG to CONFIG state the host has to apply the controller host interface command CONFIG. If the host wants the communication controller to leave CONFIG state, the host has to proceed as described in Lock Register (LCK).

NOTE: All bits marked with an asterisk (*) can be updated in DEFAULT_CONFIG or CONFIG state only.

26.3.2.3.1 SUC Configuration Register 1 (SUCC1)

The communication controller accepts modifications of the register in DEFAULT_CONFIG or CONFIG state only.

Figure 26-128 and Table 26-107 illustrate this register.

Figure 26-128. SUC Configuration Register 1 (SUCC1) [offset_CC = 80h]

31 28	27	26	25	24	23	22	21	20 16
Reserved	CCHB*	CCHA*	MTSB*	MTSA*	HCSE*	TSM*	WUCS*	PTA*
R-0	R/W-1	R/W-1	R/W-0	R/W-0	R/W-0	R/W-1	R/W-0	R/W-0

15 11	10	9	8	7	6 4	3 0
CSA*	Rsvd	TXSY*	TXST*	PBSY	Reserved	CMD*
R/W-2h	R-0	R/W-0	R/W-0	R-1	R-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset; *These bits can be updated in DEFAULT_CONFIG or CONFIG state only

Table 26-107. SUC Configuration Register 1 (SUCC1) Field Descriptions

Bit	Field	Value	Description
31-28	Reserved	0	Reads return 0. Writes have no effect.
27	CCHB		Connected to channel B. Configures whether the node is connected to channel B.
		0	Node is not connected to channel B.
		1	Node is connected to channel B (default by hardware reset).
26	CCHA		Connected to channel A. Configures whether the node is connected to channel A.
		0	Node is not connected to channel A.
		1	Node is connected to channel A (default by hardware reset).
25	MTSB		Select channel B for MTS Transmission. The bit selects channel B for MTS symbol transmission if requested by writing CMD = 8h. The flag is reset by default and may be modified only in DEFAULT_CONFIG or CONFIG state.
		0	Channel B is not selected for MTS transmission.
		1	Channel B is selected for MTS transmission.
			Note: MTSB may also be changed outside DEFAULT_CONFIG or CONFIG state when the write to SUC Configuration Register 1 (SUCC1) is directly preceded by the unlock sequence for the Configuration Lock Key as described in the Lock Register (LCK). This may be combined with CHI command SEND_MTS. If both bits MTSA and MTSB are set to 1 an MTS symbol will be transmitted on both channels when requested by writing CMD = 8h.

Table 26-107. SUC Configuration Register 1 (SUCC1) Field Descriptions (continued)

Bit	Field	Value	Description
24	MTSA		Select channel A for MTS Transmission. The bit selects channel A for MTS symbol transmission if requested by writing CMD = 8h. The flag is reset by default and may be modified only in DEFAULT_CONFIG or CONFIG state.
		0	Channel A is not selected for MTS transmission.
		1	Channel A is selected for MTS transmission.
			Note: MTSA may also be changed outside DEFAULT_CONFIG or CONFIG state when the write to SUC Configuration Register 1 (SUCC1) is directly preceded by the unlock sequence for the Configuration Lock Key as described in the Lock Register (LCK). This may be combined with CHI command SEND_MTS. If both bits MTSA and MTSB are set to 1 an MTS symbol will be transmitted on both channels when requested by writing CMD = 8h.
23	HCSE		Halt due to clock sync error. Controls reaction of the communication controller to a clock synchronization error. The bit can be modified in DEFAULT_CONFIG or CONFIG state only.
		0	Communication controller will enter or remain in NORMAL_PASSIVE.
		1	Communication controller will enter HALT state.
22	TSM		Transmission slot mode. Selects the initial transmission slot mode. In SINGLE slot mode the communication controller may only transmit in the pre-configured key slot. This slot is defined by the key slot ID, which is configured in the header section of message buffer 0. In all slot mode the communication controller may transmit in all slots. The bit can be written in DEFAULT_CONFIG or CONFIG state only. The communication controller changes to all slot mode when the host successfully applied the ALL_SLOTS command by writing CMD = 5h in POC states NORMAL_ACTIVE or NORMAL_PASSIVE. The actual slot mode is monitored by SLM in register CCSV.
		0	All slot mode.
		1	Single slot mode (default by hardware reset).
21	WUCS		Wakeup channel select. With this bit the host selects the channel on which the communication controller sends the Wakeup pattern. The communication controller ignores any attempt to change the status of this bit when not in DEFAULT_CONFIG or CONFIG state.
		0	Send wakeup pattern on channel A.
		1	Send wakeup pattern on channel B.
20-16	PTA	0-1Fh even/odd cycle pairs	Passive to active. Defines the number of consecutive even/odd cycle pairs that must have valid clock correction terms before the communication controller is allowed to transit from NORMAL_PASSIVE to NORMAL_ACTIVE state. If set to 0, the communication controller is not allowed to transit from NORMAL_PASSIVE to NORMAL_ACTIVE state. It can be modified in DEFAULT_CONFIG or CONFIG state only.
15-11	CSA	2h-1Fh	Cold start attempts. Configures the maximum number of attempts that a cold starting node is permitted to try to start up the network without receiving any valid response from another node. It can be modified in DEFAULT_CONFIG or CONFIG state only. Must be identical in all nodes of a cluster.
10	Reserved	0	Reads return 0. Writes have no effect.
9	TXSY		Transmit sync frame in key slot. Defines whether the key slot is used to transmit a sync frame. The bit can be modified inDEFAULT_CONFIG or CONFIG state only.
			Note: The protocol requires that both bits TXST and TXSY are set for coldstart nodes.
		0	No sync frame transmission in key slot, node is neither sync nor coldstart node.
		1	Key slot used to transmit sync frame, node is sync node.
8	TXST		Transmit startup frame in key slot. Defines whether the key slot is used to transmit a startup frame. The bit can be modified in DEFAULT_CONFIG or CONFIG state only.
			Note: The protocol requires that both bits TXST and TXSY are set for coldstart nodes.
		0	No startup frame transmitted in key slot, node is non-coldstarter.
		1	Key slot used to transmit startup frame, node is leading or following coldstarter.
7	PBSY		POC busy. Signals that the POC is busy and cannot accept a command from the host. CMD is locked against write accesses.
		0	POC is not busy, CMD is writable.
		1	POC is busy, CMD is locked.
6-4	Reserved	0	Reads return 0. Writes have no effect.

Table 26-107. SUC Configuration Register 1 (SUCC1) Field Descriptions (continued)

Bit	Field	Value	Description
3-0	CMD		The controller host interface command vector. The host may write any controller host interface command at any time, but certain commands are enabled only in certain POC states. If a command is not enabled, it will not be executed, the controller host interface command vector CMD will be reset to 0000 = command_not_accepted, and flag CNA in the error interrupt register will be set to 1. In case the previous controller host interface (CHI) command has not yet completed, EIR.CCL is set to 1 together with EIR.CNA; the CHI command needs to be repeated. Except for HALT state, a POC state change command applied while the communication controller is already in the requested POC state neither causes a state change nor will EIR.CNA be set.
		0	command_not_accepted
		1h	CONFIG
		2h	READY
		3h	WAKEUP
		4h	RUN
		5h	ALL_SLOTS
		6h	HALT
		7h	FREEZE
		8h	SEND_MTS
		9h	ALLOW_COLDSTART
		Ah	RESET_STATUS_INDICATORS
		Bh	MONITOR_MODE
		Ch	CLEAR_RAMS
		Dh-Eh	Reserved
		Fh	LOOPBACK MODE

Controller Host Interface Command Vector:

The following gives more information about the controller host interface commands.

- Reading CMD shows whether the last controller host interface command was accepted.
- The actual POC state is monitored by POCS in the communication controller status vector
- In most cases the Host must check SUCC1.PBSY before writing a new CHI command.

command_not_accepted

CMD is reset to 0000 due to one of the following conditions:

- Illegal command applied by the host
- Host applied command to leave CONFIG state without preceding configuration lock key
- Host applied new command while execution of the previous host command has not completed
- Host writes command_not_accepted

When CMD is reset to 0000 due to an unaccepted command, bit CNA in the error interrupt register is set, and, if enabled, an interrupt is generated. Commands which are not accepted are not executed.

CONFIG

Go to POC state CONFIG when called in POC states DEFAULT_CONFIG, READY, or in MONITOR_MODE. When called in HALT state the communication controller transits to POC state DEFAULT_CONFIG. When called in any other state, CMD will be reset to 0000 = command_not_accepted.

READY

Go to POC state READY when called in POC states CONFIG, NORMAL_ACTIVE, NORMAL_PASSIVE, STARTUP, or WAKEUP. When called in any other state, CMD will be reset to 0000 = command_not_accepted.

WAKEUP

Go to POC state WAKEUP when called in POC state READY. When called in any other state, CMD will be reset to 0000 = command_not_accepted.

RUN

Go to POC state STARTUP when called in POC state READY. When called in any other state, CMD will be reset to 0000 = command_not_accepted.

ALL_SLOTS

Leave single slot mode after successful startup / integration at the next end of cycle when called in POC states NORMAL_ACTIVE or NORMAL_PASSIVE. When called in any other state, CMD will be reset to 0000 = command_not_accepted.

HALT

Set the Halt request HRQ bit in the communication controller status vector register and go to POC state HALT at the next end of cycle when called in POC states NORMAL_ACTIVE or NORMAL_PASSIVE. When called in any other state, CMD will be reset to 0000 = command_not_accepted.

FREEZE

Go to POC state HALT immediately and set the Freeze status Indicator FSI bit in the communication controller status vector register. Can be called from any state.

SEND_MTS

Send single MTS symbol during the symbol window of the following cycle on the channel configured by MTSA, MTSB, when called in POC state NORMAL_ACTIVE. When called in any other state, CMD will be reset to 0000 = command_not_accepted.

ALLOW_COLDSTART

The command resets bit CSI in the CCSV register to enable coldstarting of the node when called in any POC state except DEFAULT_CONFIG, CONFIG or HALT. When called in these states, CMD will be reset to 0000 = command_not_accepted.

RESET_STATUS_INDICATORS

Reset status flags FSI, HRQ, CSNI, and CSAI in the communication controller status vector register.

CLEAR_RAMS

Sets bit CRAM in the message handler status register when called in DEFAULT_CONFIG or CONFIG state. When called in any other state, CMD will be reset to 0000 = command_not_accepted. Bit CRAM is also set when the communication controller leaves hardware reset. By setting bit CRAM, all internal RAM blocks are initialized to 0 and the ECC bits are initialized accordingly, depending what mode is enabled. During the initialization of the RAMs, PBSY will show POC busy. Access to the configuration and status registers is possible during execution of CHI command CLEAR_RAMS.

The initialization of the Communication Controller internal RAM blocks takes 2048 VBUS clock cycles. There should be no host access to IBF or OBF during initialization of the internal RAM blocks after hardware reset or after assertion of controller host interface command CLEAR_RAMS. Before asserting controller host interface command CLEAR_RAMS the host should be aware that no transfer between message RAM and IBF / OBF or the transient buffer RAMs is ongoing. This command also resets the message buffer status registers (MHDS, TXRQ1/2/3/4, NDAT1/2/3/4, MBSC1/2/3/4).

NOTE: All accepted commands with exception of CLEAR_RAMS and SEND_MTS will cause a change of the POC state in the VBUS clock domain after at most 8 cycles of the slower of the two clocks VBUS clock and 80MHz sample clock coming from the PLL. It is assumed that POC was not busy when the command was applied and that no POC state change was forced by bus activity in that time frame. Reading register Communication Controller Status Vector (CCSV) will show data that is additionally delayed by synchronization from sample clock to VBUS clock domain and by the CPU interface. The maximum additional delay is 12 cycles of the slower of the two clocks VBUS clock and sample clock.

MONITOR_MODE

Enter MONITOR_MODE when called in POC state CONFIG. In this mode the communication controller is able to receive FlexRay frames and CAS / MTS symbols. It is also able to detect coding errors. The temporal integrity of received frames is not checked. This mode can be used for debugging purposes, for example, in case that the startup of a FlexRay network fails. When called in any other state, CMD will be reset to 0000 = command_not_accepted.

26.3.2.3.2 SUC Configuration Register 2 (SUCC2)

The communication controller accepts modifications of the register in DEFAULT_CONFIG or CONFIG state only.

Figure 26-129 and Table 26-108 illustrate this register.

Figure 26-129. SUC Configuration Register 2 (SUCC2) [offset_CC = 84h]

31 - 28	27 - 24	23 - 21	20 - 16
Reserved	LTN*	Reserved	LT*
R-0	R/W-1h	R-0	R/W-504h

15 - 0
LT*
R/W-504h

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset; *These bits can be updated in DEFAULT_CONFIG or CONFIG state only.

Table 26-108. SUC Configuration Register 2 (SUCC2) Field Descriptions

Bit	Field	Value	Description
31-28	Reserved.	0	Reads return 0. Writes have no effect.
27-24	LTN	2-Fh	Listen timeout noise. Configures the upper limit for the startup and wakeup listen timeout in the presence of noise. Must be identical in all nodes of a cluster. The wakeup / startup noise timeout is calculated as follows: LT × (LTN + 1).
23-21	Reserved.	0	Reads return 0. Writes have no effect.
20-0	LT	504h-139706h µT	Listen timeout. Configures the upper limit of the startup and wakeup listen timeout.

26.3.2.3.3 SUC Configuration Register 3 (SUCC3)

The communication controller accepts modifications of the register in DEFAULT_CONFIG or CONFIG state only.

Figure 26-130 and Table 26-109 illustrate this register.

Figure 26-130. SUC Configuration Register 3 (SUCC3) [offset_CC = 88h]

31–16
Reserved
R-0

15–8	7–4	3–0
Reserved	WCF*	WCP*
R-0	R/W-1h	R/W-1h

LEGEND: R/W = Read/Write; R = Read only; D = Device-specific reset value; -*n* = value after reset; *These bits can be updated in DEFAULT_CONFIG or CONFIG state only

Table 26-109. SUC Configuration Register 3 (SUCC3) Field Descriptions

Bit	Field	Value	Description
31-8	Reserved.	0	Reads return 0. Writes have no effect.
7-4	WCF	1-Fh	Maximum without clock correction fatal. These bits define the number of consecutive even/odd cycle pairs with missing clock correction terms that will cause a transition from NORMAL_ACTIVE or NORMAL_PASSIVE state. These must be identical in all nodes of a cluster. **Note: The transition to HALT state is prevented if SUCC1.HCSE is not set.**
3-0	WCP	1-Fh	Maximum without clock correction passive. These bits define the number of consecutive even/odd cycle pairs with missing clock correction terms that will cause a transition from NORMAL_ACTIVE to NORMAL_PASSIVE to HALT state. These must be identical in all nodes of a cluster.

26.3.2.3.4 NEM Configuration Register (NEMC)

The communication controller accepts modifications of the register in DEFAULT_CONFIG or CONFIG state only.

Figure 26-131 and Table 26-110 illustrate this register.

Figure 26-131. NEM Configuration Register (NEMC) [offset_CC = 8Ch]

31–16
Reserved
R-0

15–4	3–0
Reserved	NML*
R-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset; *These bits can be updated in DEFAULT_CONFIG or CONFIG state only

Table 26-110. NEM Configuration Register (NEMC) Field Descriptions

Bit	Field	Value	Description
31-7	Reserved.	0	Reads return 0. Writes have no effect.
6-0	NML	0-Ch bytes	Network management vector length (in bytes). These bits configure the length of the NM vector. The configured length must be identical in all nodes of a cluster.

26.3.2.3.5 PRT Configuration Register 1 (PRTC1)

The communication controller accepts modifications of the register in DEFAULT_CONFIG or CONFIG state only.

Figure 26-132 and Table 26-111 illustrate this register.

Figure 26-132. PRT Configuration Register 1 (PRTC1) [offset_CC = 90h]

31 26	25	24 16
RPW*	Rsvd	RXW*
R/W-2h	R-0	R/W-4Ch

15 14	13 12	11	10 4	3 0
BRP*	SSP*	Rsvd	CASM*	TSST*
R/W-0	R/W-0	R-0	R/W-23h	R/W-3h

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset; *These bits can be updated in DEFAULT_CONFIG or CONFIG state only

Table 26-111. PRT Configuration Register 1 (PRTC1) Field Descriptions

Bit	Field	Value	Description
31-26	RWP	2h-3Fh	Repetition of transmission wakeup pattern. These bits configure the number of repetitions (sequences) of the transmission wakeup symbol.
25	Reserved	0	Reads return 0. Writes have no effect.
24-16	RXW	4Ch-12Dh	Wakeup symbol receive window length. Configures the number of bit times used by the node to test the duration of the received wakeup pattern. Must be identical in all nodes of a cluster.
15-14	BRP		Baud rate prescaler. These bits configure the baud rate on the FlexRay bus. The baud rates listed below are valid with a sample clock of 80 MHz. One bit time always consists of 8 samples independent of the configured baud rate.
		0	10 Mbit/s (Sample Clock Period = 12.5ns; 1 µT = 25ns; Samples per µT = 2)
		1h	5 Mbit/s (Sample Clock Period = 25ns; 1 µT = 25ns; Samples per µT = 1)
		2h-3h	2.5 Mbit/s (Sample Clock Period = 50ns; 1 µT = 50ns; Samples per µT = 1)
13-12	SPP		Strobe Point Position. Defines the sample count value for strobing. The strobed bit value is set to the voted value when the sample count is incremented to the value configured by SPP. **Note: The current revision 2.1 of the FlexRay protocol requires that SPP = 00. The alternate strobe point positions could be used to compensate for asymmetries in the physical layer.**
		0, 3h	Sample 5 (default)
		1h	Sample 4
		2h	Sample 6
11	Reserved	0	Reads return 0. Writes have no effect.
10-4	CASM	43h-63h bit times	Collision avoidance symbol max (in bit times). These bits configure the upper limit of the acceptance window for a collision avoidance symbol (CAS). CASM6 is always 1.
3-0	TSST	3h-Fh bit times	Transmission start sequence transmitter (in bit times). These bits configure the duration of the transmission start sequence (TSS) in terms of bit times (1 bit time = 4 µT = 100ns @ 10Mbps). Must be identical in all nodes of a cluster.

26.3.2.3.6 PRT Configuration Register 2 (PRTC2)

The communication controller accepts modifications of the register in DEFAULT_CONFIG or CONFIG state only.

Figure 26-133 and Table 26-112 illustrate this register.

Figure 26-133. PRT Configuration Register 2 (PRTC2) [offset_CC = 94h]

31 30	29 24	23 16
Reserved	TXL*	TXI*
R-0	R/W-Fh	R/W-2Dh

15 14	13 8	7 6	5 0
Reserved	RXL*	Reserved	RXI*
R-0	R/W-Ah	R-0	R/W-Eh

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset; *These bits can be updated in DEFAULT_CONFIG or CONFIG state only

Table 26-112. PRT Configuration Register 2 (PRTC2) Field Descriptions

Bit	Field	Value	Description
31-30	Reserved	0	Reads return 0. Writes have no effect.
29-24	TXL	Fh-3Ch bit times	Wakeup symbol transmit low (in bit times). These bits configure the active low phase of the wakeup symbol. The duration must be identical in all nodes of a cluster.
23-16	TXI	2Dh-B4h bit times	Wakeup symbol transmit idle (in bit times). These bits configure the number of bit times used by the node to transmit the idle phase of the wakeup symbol. Durations must be identical in all nodes of a cluster.
15-14	Reserved	0	Reads return 0. Writes have no effect.
13-8	RXL	Ah-37h bit times	Wakeup symbol receive low (in bit times). These bits configure the number of bit times used by the node to test the duration of the low phase of the received wakeup symbol. Must be identical in all nodes of a cluster.
7-6	Reserved	0	Reads return 0. Writes have no effect.
5-0	RXI	Eh-37h bit times	Wakeup symbol receive idle (in bit times). These bits configure the number of bit times used by the node to test the duration of the idle phase of the received wakeup symbol. Must be identical in all nodes of a cluster.

26.3.2.3.7 MHD Configuration Register (MHDC)

The communication controller accepts modifications of the register in DEFAULT_CONFIG or CONFIG state only.

Figure 26-134 and Table 26-113 illustrate this register.

Figure 26-134. MHD Configuration Register (MHDC) [offset_CC = 98h]

31 - 29	28 - 16
Reserved	SLT*
R-0	R/W-2h

15 - 7	6 - 0
Reserved	SFDL*
R-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset; *These bits can be updated in DEFAULT_CONFIG or CONFIG state only

Table 26-113. MHD Configuration Register (MHDC) Field Descriptions

Bit	Field	Value	Description
31-29	Reserved	0	Reads return 0. Writes have no effect.
28-16	SLT	0-1F2Dh minislots	Start of latest transmit (in minislots). These bits configure the maximum minislot value allowed before inhibiting new frame transmissions in the Dynamic Segment of the cycle. There is no transmission in dynamic segment if SLT is cleared to 0.
15-8	Reserved	0	Reads return 0. Writes have no effect.
7-0	SFDL	0-7Fh	Static frame data length. These bits configure the cluster-wide payload length for all frames sent in the static segment in double bytes. The payload length must be identical in all nodes of a cluster.

26.3.2.3.8 GTU Configuration Register 1 (GTUC1)

The communication controller accepts modifications of the register in DEFAULT_CONFIG or CONFIG state only.

Figure 26-135 and Table 26-114 illustrate this register.

Figure 26-135. GTU Configuration Register 1 (GTUC1) [offset_CC = A0h]

31-20	19-16
Reserved	UT*
R-0	R/W-0

15-0
UT*
R/W-0280h

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset; *These bits can be updated in DEFAULT_CONFIG or CONFIG state only

Table 26-114. GTU Configuration Register 1 (GTUC1) Field Descriptions

Bit	Field	Value	Description
31-20	Reserved.	0	Reads return 0. Writes have no effect.
19-0	UT	280h-9C400h µT	Microtick per cycle (in microticks). These bits configure the duration of the communication cycle in microticks.

26.3.2.3.9 GTU Configuration Register 2 (GTUC2)

The communication controller accepts modifications of the register in DEFAULT_CONFIG or CONFIG state only.

Figure 26-136 and Table 26-115 illustrate this register.

Figure 26-136. GTU Configuration Register 2 (GTUC2) [offset_CC = A4h]

31-20	19-16
Reserved	SNM*
R-0	R/W-2h

15-14	13-0
Reserved	MPC*
R-0	R/W-Ah

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset; *These bits can be updated in DEFAULT_CONFIG or CONFIG state only

Table 26-115. GTU Configuration Register 2 (GTUC2) Field Descriptions

Bit	Field	Value	Description
31-20	Reserved	0	Reads return 0. Writes have no effect.
19-16	SNM	2h-Fh frames	Sync node max (in frames). These bits configure the maximum number of frames within a cluster with sync frame indicator bit SYN set. The number of frames must be identical in all nodes of a cluster.
15-14	Reserved	0	Reads return 0. Writes have no effect.
13-0	MPC	Ah-3E80h MT	Macrotick per cycle (in macroticks). These bits configure the duration of one communication cycle in macroticks. The cycle length must be identical in all nodes of a cluster.

26.3.2.3.10 GTU Configuration Register 3 (GTUC3)

The communication controller accepts modifications of the register in DEFAULT_CONFIG or CONFIG state only.

Figure 26-137 and Table 26-116 illustrate this register.

Figure 26-137. GTU Configuration Register 3 (GTUC3) [offset_CC = A8h]

31	30 — 24	23	22 — 16
Rsvd	MIOB*	Rsvd	MIOA*
R-0	R/W-2h	R-0	R/W-2h

15 — 8	7 — 0
UIOB*	UIOA*
R/W-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset; *These bits can be updated in DEFAULT_CONFIG or CONFIG state only.

Table 26-116. GTU Configuration Register 3 (GTUC3) Field Descriptions

Bit	Field	Value	Description
31	Reserved	0	Reads return 0. Writes have no effect.
30-24	MIOB	2h-48h MT	Macrotick initial offset channel B (in macroticks). These bits configure the number of macroticks between the static slot boundary and the subsequent macrotick boundary of the secondary time reference point based on the nominal macrotick duration. Must be identical in all nodes of a cluster.
23	Reserved	0	Reads return 0. Writes have no effect.
22-16	MIOA	2h-48h MT	Macrotick initial offset channel A (in macroticks). These bits configure the number of macroticks between the static slot boundary and the subsequent macrotick boundary of the secondary time reference point based on the nominal macrotick duration. Must be identical in all nodes of a cluster.
15-8	UIOB	0-F0h µT	Microtick initial offset channel B (in microticks). These bits configure the number of microticks between the actual time reference point on channel B and the subsequent macrotick boundary of the secondary time reference point. The parameter has to be set for each channel independently.
7-0	UIOA	0-F0h µT	Microtick initial offset channel A (in microticks). These bits configure the number of microticks between the actual time reference point on channel A and the subsequent macrotick boundary of the secondary time reference point. The parameter has to be set for each channel independently.

26.3.2.3.11 GTU Configuration Register 4 (GTUC4)

The communication controller accepts modifications of the register in DEFAULT_CONFIG or CONFIG state only. Figure 26-138 and Table 26-117 illustrate this register.

Figure 26-138. GTU Configuration Register 4 (GTUC4) [offset_CC = ACh]

31 30	29 16
Reserved	OCS*
R-0	R/W-Ah

15 14	13 0
Reserved	NIT*
R-0	R/W-9h

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset; *These bits can be updated in DEFAULT_CONFIG or CONFIG state only

Table 26-117. GTU Configuration Register 4 (GTUC4) Field Descriptions

Bit	Field	Value	Description
31-30	Reserved	0	Reads return 0. Writes have no effect.
29-16	OCS	8h-3E7Eh MT	Offset correction start (in macroticks). These bits determine the start of the offset correction within the NIT phase, calculated from start of cycle. Must be identical in all nodes of a cluster.
15-14	Reserved	0	Reads return 0. Writes have no effect.
13-0	NIT	7h-3E7Dh MT	Network idle time start (in macroticks). These bits configure the starting point of the network idle time (NIT) at the end of the communication cycle expressed in terms of macroticks from the beginning of the cycle. The number must be identical in all nodes of a cluster.

26.3.2.3.12 GTU Configuration Register 5 (GTUC5)

The communication controller accepts modifications of the register in DEFAULT_CONFIG or CONFIG state only. Figure 26-139 and Table 26-118 illustrate this register.

Figure 26-139. GTU Configuration Register 5 (GTUC5) [offset_CC = B0h]

31 24	23 21	20 16
DEC*	Reserved	CDD*
R/W-Eh	R-0	R/W-0

15 8	7 0
DCB*	DCA*
R/W-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset; *These bits can be updated in DEFAULT_CONFIG or CONFIG state only

Table 26-118. GTU Configuration Register 5 (GTUC5) Field Descriptions

Bit	Field	Value	Description
31-24	DEC	Eh-8Fh µT	Decoding correction (in microticks). These bits configure the decoding correction value used to determine the primary time reference point.
23-21	Reserved.	0	Reads return 0. Writes have no effect.
20-16	CDD	0-14h µT	Cluster drift damping (in microticks). These bits configure the cluster drift damping value used in clock synchronization to minimize accumulation of rounding errors.
15-8	DCB	0-C8h µT	Delay compensation channel B (in microticks). These bits are used to compensate for reception delays on the indicated channel. This compensates propagation delays for microticks in the range of 0.0125 to 0.05s. In practice, the minimum propagation delay of all sync nodes should be applied.
7-0	DCA	0-C8h µT	Delay compensation channel A (in microticks). These bits are used to compensate for reception delays on the indicated channel. This compensates propagation delays for microticks in the range of 0.0125 to 0.05s. In practice, the minimum propagation delay of all sync nodes should be applied.

26.3.2.3.13 GTU Configuration Register 6 (GTUC6)

The communication controller accepts modifications of the register in DEFAULT_CONFIG or CONFIG state only.

Figure 26-140 and Table 26-119 illustrate this register.

Figure 26-140. GTU Configuration Register 6 (GTUC6) [offset_CC = B4h]

31 — 27	26 — 16
Reserved	MOD*
R-0	R/W-2h

15 — 11	10 — 0
Reserved	ASR*
R-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset; *These bits can be updated in DEFAULT_CONFIG or CONFIG state only

Table 26-119. GTU Configuration Register 6 (GTUC6) Field Descriptions

Bit	Field	Value	Description
31-27	Reserved	0	Reads return 0. Writes have no effect.
26-16	MOD	2h-783h µT	Maximum oscillator drift (in microticks). Maximum drift offset between two nodes that operate with unsynchronized clocks over one communication cycle in µT.
15-11	Reserved	0	Reads return 0. Writes have no effect.
10-0	ASR	0-753h µT	Accepted startup range (in microticks). Number of microticks constituting the expanded range of measured deviation for startup frames during integration.

26.3.2.3.14 GTU Configuration Register 7 (GTUC7)

The communication controller accepts modifications of the register in DEFAULT_CONFIG or CONFIG state only.

Figure 26-141 and Table 26-120 illustrate this register.

Figure 26-141. GTU Configuration Register 7 (GTUC7) [offset_CC = B8h]

31 — 26	25 — 16
Reserved	NSS*
R-0	R/W-2h

15 — 10	9 — 0
Reserved	SSL*
R-0	R/W-4h

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset; *These bits can be updated in DEFAULT_CONFIG or CONFIG state only

Table 26-120. GTU Configuration Register 7 (GTUC7) Field Descriptions

Bit	Field	Value	Description
31-26	Reserved.	0	Reads return 0. Writes have no effect.
25-16	NSS	2h-3FFh	Number of static slots. These bits configure the number of static slots in a cycle. At least two coldstart nodes must be configured to startup a FlexRay network. The number of static slots must be identical in all nodes of a cluster.
15-10	Reserved.	0	Reads return 0. Writes have no effect.
9-0	SSL	4h-293h	Static slot length (in macroticks). These bits configure the duration of a static slot. The static slot length must be identical in all nodes of a cluster.

26.3.2.3.15 GTU Configuration Register 8 (GTUC8)

The communication controller accepts modifications of the register in DEFAULT_CONFIG or CONFIG state only.

Figure 26-142 and Table 26-121 illustrate this register.

Figure 26-142. GTU Configuration Register 8 (GTUC8) [offset_CC = BCh]

31 29	28 16
Reserved	NMS*
R-0	R/W-0

15 6	5 0
Reserved	MSL*
R-0	R/W-2h

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset; *These bits can be updated in DEFAULT_CONFIG or CONFIG state only

Table 26-121. GTU Configuration Register 8 (GTUC8) Field Descriptions

Bit	Field	Value	Description
31-29	Reserved	0	Reads return 0. Writes have no effect.
28-16	NMS	0-1F32h	Number of minislots. These bits configure the number of minislots in the dynamic segment of a cycle. The number of minislots must be identical in all nodes of a cluster.
15-6	Reserved	0	Reads return 0. Writes have no effect.
5-0	MSL	2h-3Fh MT	Minislot length (in macroticks). These bits configure the duration of a minislot. The minislot length must be identical in all nodes of a cluster.

26.3.2.3.16 GTU Configuration Register 9 (GTUC9)

The communication controller accepts modifications of the register in DEFAULT_CONFIG or CONFIG state only.

Figure 26-143 and Table 26-122 illustrate this register.

Figure 26-143. GTU Configuration Register 9 (GTUC9) [offset_CC = C0h]

31 18	17 16
Reserved	DSI*
R-0	R/W-0

15 13	12 8	7 6	5 0
Reserved	MAPO*	Reserved	APO*
R-0	R/W-1h	R-0	R/W-1h

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset; *These bits can be updated in DEFAULT_CONFIG or CONFIG state only

Table 26-122. GTU Configuration Register 9 (GTUC9) Field Descriptions

Bit	Field	Value	Description
31-18	Reserved	0	Reads return 0. Writes have no effect.
17-16	DSI	0-2h	Dynamic slot idle phase (in minislots). The duration of the dynamic slot idle phase has to be greater or equal than the idle detection time. Must be identical in all nodes of a cluster.
15-13	Reserved	0	Reads return 0. Writes have no effect.
12-8	MAPO	1h-1Fh MT	Minislot action point offset (in macroticks). These bits configure the minislot action point offset within the minislots of the dynamic segment. The minislot action point offset must be identical in all nodes of a cluster.
7-6	Reserved	0	Reads return 0. Writes have no effect.
5-0	APO	1h-3Fh MT	Action point offset (in macroticks). These bits configure the action point offset within static slots and symbol window. The action point offset must be identical in all nodes of a cluster.

26.3.2.3.17 GTU Configuration Register 10 (GTUC10)

The communication controller accepts modifications of the register in DEFAULT_CONFIG or CONFIG state only.

Figure 26-144 and Table 26-123 illustrate this register.

Figure 26-144. GTU Configuration Register 10 (GTUC10) [offset_CC = C4h]

31 27	26 16
Reserved	MRC*
R-0	R/W-2h

15 14	13 0
Reserved	MOC*
R-0	R/W-5h

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset; *These bits can be updated in DEFAULT_CONFIG or CONFIG state only

Table 26-123. GTU Configuration Register 10 (GTUC10) Field Descriptions

Bit	Field	Value	Description
31-27	Reserved	0	Reads return 0. Writes have no effect.
26-16	MRC	2h-783h µT	Maximum rate correction (in microticks). Holds the maximum permitted rate correction value to be applied by the internal clock synchronization algorithm. The communication controller checks the internal rate correction value against the maximum rate correction value (absolute value).
15-14	Reserved	0	Reads return 0. Writes have no effect.
13-0	MOC	5h-3BA2h µT	Maximum offset correction (in microticks). Holds the maximum permitted offset correction value to be applied by the internal clock synchronization algorithm (absolute value). The communication controller checks the internal offset correction value against the maximum offset correction value.

26.3.2.3.18 GTU Configuration Register 11 (GTUC11)

Figure 26-145 and Table 26-124 illustrate this register.

Figure 26-145. GTU Configuration Register 11 (GTUC11) [offset_CC = C8h]

31-27	26-24	23-19	18-16
Reserved	ERC*	Reserved	EOC*
R-0	R/W-0	R-0	R/W-0

15-10	9-8	7-2	1-0
Reserved	ERCC*	Reserved	EOCC*
R-0	R/W-0	R-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset; *These bits can be updated in DEFAULT_CONFIG or CONFIG state only

Table 26-124. GTU Configuration Register 11 (GTUC11) Field Descriptions

Bit	Field	Value	Description
31-27	Reserved	0	Reads return 0. Writes have no effect.
26-24	ERC	0-7h µT	External rate correction (in microticks). Holds the external clock rate correction value to be applied by the internal clock synchronization algorithm. The value is subtracted/added from/to the calculated rate correction value. The value is applied during NIT. May be modified in DEFAULT_CONFIG or CONFIG state only.
23-19	Reserved	0	Reads return 0. Writes have no effect.
18-16	EOC	0-7h µT	External offset correction (in microticks). Holds the external clock offset correction value to be applied by the internal clock synchronization algorithm. The value is subtracted/added from/to the calculated offset correction value. The value is applied during NIT. May be modified in DEFAULT_CONFIG or CONFIG state only.
15-10	Reserved	0	Reads return 0. Writes have no effect.
9-8	ERCC		External rate correction control. By writing to ERCC, the external rate correction is enabled as specified below. Should be modified only outside NIT.
		0, 1h	No external rate correction.
		2h	External rate correction value is subtracted from calculated rate correction value.
		3h	External rate correction value is added to calculated rate correction value.
7-2	Reserved	0	Reads return 0. Writes have no effect.
1-0	EOCC		External offset correction control. By writing to EOCC, the external offset correction is enabled as specified below. Should be modified only outside NIT.
		0-1h	No external offset correction.
		2h	External offset correction value is subtracted from calculated offset correction value.
		3h	External offset correction value is added to calculated offset correction value.

26.3.2.4 Status Registers

During 8/16-bit accesses to status variables coded with more than 8/16-bit, the variable might be updated by the communication controller between two accesses (non-atomic read accesses). All internal counters and the communication controller status flags are reset when the communication controller transits from CONFIG to READY state.

26.3.2.4.1 Communication Controller Status Vector (CCSV)

Figure 26-146 and Table 26-125 illustrate this register.

Figure 26-146. Communication Controller Status Vector Register (CCSV) [offset_CC = 100h]

31-30	29-24	23-19	18-16
Reserved	PSL	RCA	WSV
R-0	R-0	R-2h	R-0

15	14	13	12	11-10	9-8	7	6	5-0
Rsvd	CSI	CSAI	CSNI	Reserved	SLM	HQR	FSI	POCS
R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0

LEGEND: R = Read only; -*n* = value after reset

Table 26-125. Communication Controller Status Vector Register (CCSV) Field Descriptions

Bit	Field	Value	Description
31-30	Reserved	0	Reads return 0. Writes have no effect.
29-24	PSL		POC Status Log. Status of POCS immediately before entering HALT state. Set when entering HALT state. Set to HALT when FREEZE command is applied during HALT state and FSI is not already set, that is, the HALT state was not reached by FREEZE command. Reset to 0 when leaving HALT state.
23-19	RCA	0-1Fh	Remaining coldstart attempts. Indicates the number of remaining coldstart attempts. The maximum number of coldstart attempts is configured by CSA(4-0) in the SUC configuration register 1.
18-16	WSV		Wakeup status. Indicates the status of the current wakeup attempt. Reset by CHI command RESET_STATUS_INDICATORS or by transition from HALT to EFAULT_CONFIG state.
		0	UNDEFINED. No wakeup attempt since CONFIG state was left.
		1h	RECEIVED_HEADER. Set when the communication controller finishes wakeup due to the reception of a frame header without coding violation on either channel in WAKEUP_LISTEN or WAKEUP_DETECT state.
		2h	RECEIVED_WUP. Set when the communication controller finishes wakeup due to the reception of a valid wakeup pattern on the configured wakeup channel in WAKEUP_LISTEN or WAKEUP_DETECT state.
		3h	COLLISION_HEADER. Set when the communication controller stops wakeup due to a detected collision during wakeup pattern transmission by receiving a valid header on either channel.
		4h	COLLISION_WUP. Set when the communication controller stops wakeup due to a detected collision during wakeup pattern transmission by receiving a valid wakeup pattern on the configured wakeup channel.
		5h	COLLISION_UNKNOWN. Set when the communication controller stops wakeup by leaving WAKEUP_DETECT state after expiration of the wakeup timer without receiving a valid wakeup pattern or a valid frame header.
		6h	TRANSMITTED. Set when the communication controller has successfully completed the transmission of the wakeup pattern.
		7h	Reserved
15	Reserved	0	Reads return 0. Writes have no effect.
14	CSI		Cold start inhibit. Indicates that the node is disabled from cold starting. The flag is set whenever the POC enters READY state. The flag has to be reset under control of the host by the controller host interface command ALLOW_COLDSTART (CMD = 1001).
		0	Cold starting of node is enabled.
		1	Cold starting of node is disabled.
13	CSAI	0-1	Coldstart abort indicator. Coldstart aborted. Reset by CHI command RESET_STATUS_INDICATORS or by transition from HALT to DEFAULT_CONFIG state or from READY to STARTUP state.

Table 26-125. Communication Controller Status Vector Register (CCSV) Field Descriptions (continued)

Bit	Field	Value	Description
12	CSNI	0-1	Coldstart noise indicator. Indicates that the cold start procedure occurred under noisy conditions. Reset by CHI command RESET_STATUS_INDICATORS or by transition from HALT to DEFAULT_CONFIG state or from READY to STARTUP state.
11-10	Reserved	0	Reads return 0. Writes have no effect.
9-8	SLM		Slot mode. Indicates the actual slot mode of the POC in states READY, WAKEUP, STARTUP, NORMAL_ACTIVE, and NORMAL_PASSIVE. Default is SINGLE. Changes to ALL, depending on SUCC1.TSM. In NORMAL_ACTIVE or NORMAL_PASSIVE state the CHI command ALL_SLOTS will change the slot mode from SINGLE over ALL_PENDING to ALL. Set to SINGLE in all other states.
		0	SINGLE
		1h	Reserved
		2h	ALL_PENDING
		3h	ALL
7	HRQ	0-1	Halt request. Indicates that a request from the Host has been received to halt the POC at the end of the communication cycle. Reset by CHI command RESET_STATUS_INDICATORS or by transition from HALT to DEFAULT_CONFIG state or when entering READY state.
6	FSI	0-1	Freeze status indicator. Indicates that the POC has entered the HALT state due to CHI command FREEZE or due to an error condition requiring an immediate POC halt. Reset by CHI command RESET_STATUS_INDICATORS or by transition from HALT to DEFAULT_CONFIG state.
5-0	POCS		Protocol operation control status.
			Indicates the actual state of operation of the Communication Controller Protocol Operation Control:
		0	DEFAULT_CONFIG state
		1h	READY state
		2h	NORMAL_ACTIVE state
		3h	NORMAL_PASSIVE state
		4h	HALT state
		5h	MONITOR_MODE state
		6h-Ch	Reserved
		Dh	LOOPBACK MODE state
		Eh	Reserved
		Fh	CONFIG state
			Indicates the actual state of operation of the POC in the wakeup path:
		10h	WAKEUP_STANDBY state
		11h	WAKEUP_LISTEN state
		12h	WAKEUP_SEND state
		13h	WAKEUP_DETECT state
		14h-1Fh	Reserved
			Indicates the actual state of operation of the POC in the startup path:
		20h	STARTUP_PREPARE state
		21h	COLDSTART_LISTEN state
		22h	COLDSTART_COLLISION_RESOLUTION state
		23h	COLDSTART_CONSISTENCY_CHECK state
		24h	COLDSTART_GAP state
		25h	COLDSTART_JOIN state
		26h	INTEGRATION_COLDSTART_CHECK state
		27h	INTEGRATION_LISTEN state
		28h	INTEGRATION_CONSISTENCY_CHECK state
		29h	INITIALIZE_SCHEDULE state
		2Ah	ABORT_STARTUP state
		2Bh-3Fh	Reserved

NOTE: CHI command RESET_STATUS_INDICATORS (SUCC1.CMD = Ah) resets flags FSI, HRQ, CSNI, CSAI, the slot mode SLM, and the wakeup status WSV.

26.3.2.4.2 Communication Controller Error Vector (CCEV)

Reset by CHI command RESET_STATUS_INDICATORS or by transition from HALT to DEFAULT_CONFIG state or when entering READY state.

Figure 26-147 and Table 26-126 illustrate this register.

Figure 26-147. Communication Controller Error Vector Register (CCEV) [offset_CC = 104h]

31-16
Reserved
R-0

15-13	12-8	7-6	5-4	3-0
Reserved	PTAC	ERRM	Reserved	CCFC
R-0	R-0	R-0	R-0	R-0

LEGEND: R = Read only; -*n* = value after reset

Table 26-126. Communication Controller Error Vector Register (CCEV) Field Descriptions

Bit	Field	Value	Description
31-13	Reserved	0	Reads return 0. Writes have no effect.
12-8	PTAC	0-1Fh	Passive to active count. Indicates the number of consecutive even / odd cycle pairs that have passed with valid rate and offset correction terms, while the node is waiting to transit from NORMAL_PASSIVE state to NORMAL_ACTIVE state. The transition takes place when PTAC equals PTA - 1 as defined in the SUC configuration register 1.
7-6	ERRM		Error mode. Indicates the actual error mode of the POC.
		0	ACTIVE
		1h	PASSIVE
		2h	COMM_HALT
		3h	Reserved
5-4	Reserved	0	Reads return 0. Writes have no effect.
3-0	CCFC	0-Fh	Clock correction failed counter. The clock correction failed counter is incremented by 1 at the end of any odd communication cycle where either the missing offset correction error or missing rate correction error are active. The clock correction failed counter is reset to 0 at the end of an odd communication cycle if neither the offset correction failed nor the rate correction failed errors are active. The clock correction failed counter stops at 15.

26.3.2.4.3 Slot Counter Value (SCV)

This register is reset when the Communication Controller leaves CONFIG state or enters STARTUP state. Figure 26-148 and Table 26-127 illustrate this register.

Figure 26-148. Slot Counter Vector Register (SCV) [offset_CC = 110h]

31-27	26-16
Reserved	SCCB
R-0	R-0

15-11	10-0
Reserved	SCCA
R-0	R-0

LEGEND: R = Read only; -*n* = value after reset

Table 26-127. Slot Counter Vector Register (SCV) Field Descriptions

Bit	Field	Value	Description
31-27	Reserved	0	Reads return 0. Writes have no effect.
26-16	SCCB	1h-7FFh	Slot counter channel B. Current slot counter value channel B. The value is incremented by the communication controller and reset at the start of a communication cycle.
15-11	Reserved	0	Reads return 0. Writes have no effect.
10-0	SCCA	1h-7FFh	Slot counter channel A. Current slot counter value channel A. The value is incremented by the communication controller and reset at the start of a communication cycle.

26.3.2.4.4 Macrotick and Cycle Counter Value (MTCCV)

Figure 26-149 and Table 26-128 illustrate this register.

Figure 26-149. Macrotick and Cycle Counter Register (MTCCV) [offset_CC = 114h]

31-22	21-16
Reserved	CCV
R-0	R-0

15-14	13-0
Reserved	MTV
R-0	R-0

LEGEND: R = Read only; -*n* = value after reset

Table 26-128. Macrotick and Cycle Counter Register (MTCCV) Field Descriptions

Bit	Field	Value	Description
31-22	Reserved	0	Reads return 0. Writes have no effect.
21-16	CCV	0-3Fh	Cycle counter value. Current cycle counter value. The value is incremented by the communication controller at the start of a communication cycle.
15-14	Reserved	0	Reads return 0. Writes have no effect.
13-0	MTV	0-3E80h	Macrotick value. Current macrotick value. The value is incremented by the communication controller and reset at the start of a communication cycle.

26.3.2.4.5 Rate Correction Value (RCV)

This register is reset when the Communication Controller leaves CONFIG state or enters STARTUP state. Figure 26-150 and Table 26-129 illustrate this register.

Figure 26-150. Rate Correction Value Register (RCV) [offset_CC = 118h]

31 16
Reserved
R-0

15 12	11 0
Reserved	RCV
R-0	R-0

LEGEND: R = Read only; -*n* = value after reset

Table 26-129. Rate Correction Value Register (RCV) Field Descriptions

Bit	Field	Value	Description
31-12	Reserved	0	Reads return 0. Writes have no effect.
11-0	RCV		Rate correction value (in microticks). Rate correction value (two's complement). Calculated internal rate correction value before limitation. If the RCV value exceeds the limits defined by GTUC10.MRC, flag SFS.RCLR is set to 1.

NOTE: The external rate correction value is added to the limited rate correction value.

26.3.2.4.6 Offset Correction Value (OCV)

Figure 26-151 and Table 26-130 illustrate this register.

Figure 26-151. Offset Correction Value Register (OCV) [offset_CC = 11Ch]

31 20	19 16
Reserved	OCV
R-0	R-0

15 0
OCV
R-0

LEGEND: R = Read only; -*n* = value after reset

Table 26-130. Offset Correction Value Register (OCV) Field Descriptions

Bit	Field	Value	Description
31-20	Reserved	0	Reads return 0. Writes have no effect.
19-0	OCV		Offset correction value (in microticks). Offset correction value (two's complement). Calculated internal offset correction value before limitation. If the OCV value exceeds the limits defined by GTUC10.MOC, flag SFS.OCLR is set to 1.

NOTE: The external offset correction value is added to the limited offset correction value.

26.3.2.4.7 Sync Frame Status (SFS)

This register is reset when the Communication Controller leaves CONFIG state or enters STARTUP state. Figure 26-152 and Table 26-131 illustrate this register.

Figure 26-152. Sync Frame Status Register (SFS) [offset_CC = 120h]

31 20	19	18	17	16
Reserved	RCLR	MRCS	OCLR	MOCS
R-0	R-0	R-0	R-0	R-0

15 12	11 8	7 4	3 0
VSBO	VSBE	VSAO	VSAE
R-0	R-0	R-0	R-0

LEGEND: R = Read only; -*n* = value after reset

Table 26-131. Sync Frame Status Register (SFS) Field Descriptions

Bit	Field	Value	Description
31-20	Reserved	0	Reads return 0. Writes have no effect.
19	RCLR		Rate correction limit reached. The Rate Correction Limit Reached flag signals to the Host, that the rate correction value has exceeded its limit as defined by GTUC10.MRC. The flag is updated by the communication controller at start of offset correction phase.
		0	Rate correction is below limit.
		1	Rate correction limit is reached.
18	MRCS		Missing rate correction signal. The missing rate correction signal signals to the host that no rate correction can be performed because no pairs of even/odd sync frames were received. The flag is updated by the communication controller at start of offset correction phase.
		0	Rate correction signal is valid.
		1	Missing rate correction signal.
17	OCLR		Offset correction limit reached. The offset correction limit reached flag signals to the host that the offset correction value has reached its limit as defined by GTUC10.MOC. The flag is updated by the communication controller at start of offset correction phase.
		0	Offset correction is below limit.
		1	Offset correction limit is reached.
16	MOCS		Missing offset correction signal. The missing offset correction signal signals to the host that no rate correction can be performed because no pairs of even / odd sync frames were received. The flag is updated by the communication controller at start of offset correction phase.
		0	Offset correction signal is valid.
		1	Missing offset correction signal.
15-12	VSBO	0-Fh	Valid sync frames channel B, odd communication cycle. Holds the number of valid sync frames received on channel B in the odd communication cycle. If transmission of sync frames is enabled by SUCC1.TXSY, the value is incremented by 1. The value is updated during the NIT of each odd communication cycle.
11-8	VSBE	0-Fh	Valid synch frames channel B, even communication cycle. Holds the number of valid sync frames received and transmitted on channel B in the even communication cycle. If transmission of sync frames is enabled by SUCC1.TXSY, the value is incremented by 1. The value is updated during the NIT of each even communication cycle.
7-4	VSAO	0-Fh	Valid synch frames channel A, odd communication cycle. Holds the number of valid sync frames received and transmitted on channel A in the odd communication cycle. If transmission of sync frames is enabled by SUCC1.TXSY, the value is incremented by 1. The value is updated during the NIT of each odd communication cycle.
3-0	VSAE	0-Fh	Valid synch frames channel A, even communication cycle. Holds the number of valid sync frames received and transmitted on channel A in the even communication cycle. If transmission of sync frames is enabled by SUCC1.TXSY, the value is incremented by 1. The value is updated during the NIT of each even communication cycle.

NOTE: The bit fields **VSBO**, **VSBE, VSAO, VSAE** are only valid if the respective channel is assigned to the communication controller by SUCC1.CCHA or SUCC1.CCHB.

26.3.2.4.8 Symbol Window and NIT Status (SWNIT)

Symbol window related status information. Updated by the communication controller at the end of the symbol window for each channel. During startup the status data is not updated. This register is reset when the Communication Controller leaves CONFIG state or enters STARTUP state.

Figure 26-153 and Table 26-132 illustrate this register.

Figure 26-153. Symbol Window and NIT Status Register (SWNIT) [offset_CC = 124h]

31–16
Reserved
R-0

15–12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	SBNB	SENB	SBNA	SENA	MTSB	MTSA	TCSB	SBSB	SESB	TCSA	SBSA	SESA
R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0

LEGEND: R = Read only; -*n* = value after reset

Table 26-132. Symbol Window and NIT Status Register (SWNIT) Field Descriptions

Bit	Field	Value	Description
31-12	Reserved	0	Reads return 0. Writes have no effect.
11	SBNB		Slot boundary violation during NIT channel B.
		0	No slot boundary violation is detected.
		1	Slot boundary violation during NIT is detected on channel B.
10	SENB		Syntax error during NIT channel B.
		0	No syntax error is detected.
		1	Syntax error during NIT is detected on channel B.
9	SBNA		Slot boundary violation during NIT channel A.
		0	No slot boundary violation is detected.
		1	Slot boundary violation during NIT is detected on channel A.
8	SENA		Syntax error during NIT channel A.
		0	No syntax error is detected.
		1	Syntax error during NIT is detected on channel A.
7	MTSB		MTS Received on Channel B. Media Access Test symbol received on channel B during the last symbol window. Updated by the communication controller for each channel at the end of the symbol window. When this bit is set to 1, also interrupt flag SIR.MTSB is set to 1.
		0	No MTS symbol is received on channel B.
		1	MTS symbol is received on channel B.
6	MTSA		MTS Received on Channel A. Media Access Test symbol received on channel A during the last symbol window. Updated by the communication controller for each channel at the end of the symbol window. When this bit is set to 1, also interrupt flag SIR.MTSB is set to 1.
		0	No MTS symbol is received on channel A.
		1	MTS symbol is received on channel A.
5	TCSB		Transmission conflict in symbol window channel B.
		0	No transmission conflict is detected.
		1	Transmission conflict in symbol window is detected on channel B.
4	SBSB		Slot boundary violation in symbol window channel B.
		0	No slot boundary violation is detected.
		1	Slot boundary violation during symbol window is detected on channel B.
3	SESB		Syntax error in symbol window channel B.
		0	No syntax error is detected.
		1	Syntax error during symbol window is detected on channel B.

Table 26-132. Symbol Window and NIT Status Register (SWNIT) Field Descriptions (continued)

Bit	Field	Value	Description
2	TCSA		Transmission conflict in symbol window channel A.
		0	No transmission conflict is detected.
		1	Transmission conflict in symbol window is detected on channel A.
1	SBSA		Slot boundary violation in symbol window channel A.
		0	No slot boundary violation is detected.
		1	Slot boundary violation during symbol window is detected on channel A.
0	SESA		Syntax error in symbol window channel A.
		0	No syntax error is detected.
		1	Syntax error during symbol window is detected on channel A.

26.3.2.4.9 Aggregated Channel Status (ACS)

The aggregated channel status provides the host with an accrued status of channel activity for all communication slots regardless of whether they are assigned for transmission or subscribed for reception. The aggregated channel status also includes status data from the symbol phase and the network idle time. The status data is updated (set) after each slot and aggregated until it is reset by the host. During startup the status data is not updated. A flag is cleared by writing a 1 to the corresponding bit position. Writing a 0 has no effect on the flag. This register is reset when the Communication Controller leaves CONFIG state or enters STARTUP state.

Figure 26-154 and Table 26-133 illustrate this register.

Figure 26-154. Aggregated Channel Status Register (ACS) [offset_CC = 128h]

31-16
Reserved
R-0

15-13	12	11	10	9	8	7-5	4	3	2	1	0
Reserved	SBVB	CIB	CEDB	SEDB	VFRB	Reserved	SBVA	CIA	CEDA	SEDA	VFRA
R-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 26-133. Aggregated Channel Status Register (ACS) Field Descriptions

Bit	Field	Value	Description
31-13	Reserved	0	Reads return 0. Writes have no effect.
12	SBVB		Slot boundary violation on channel B. One or more slot boundary violations were observed on channel B at any time during the observation period (static or dynamic slots including symbol window and NIT).
		0	No slot boundary violation is observed.
		1	Slot boundary violation is observed on channel B.
11	CIB		Communication indicator channel B. One or more valid frames were received on channel B in slots that also contained any additional communication during the observation period, that is, one or more slots received a valid frame AND had any combination of either syntax error OR content error OR slot boundary violation.
		0	No valid frame is received in slots containing any additional communication.
		1	Valid frame is received on channel B in slots containing any additional communication.
10	CEDB		Content error detected on channel B. One or more frames with a content error were received on channel B in any static or dynamic slot during the observation period.
		0	No frame with content error is received.
		1	Frame with content error is received on channel B.

Table 26-133. Aggregated Channel Status Register (ACS) Field Descriptions (continued)

Bit	Field	Value	Description
9	SEDB		Syntax error detected on channel B. One or more syntax errors in static or dynamic slots including symbol window and NIT were observed on channel B.
		0	No syntax error is observed.
		1	Syntax error is observed on channel B.
8	VFRB		Valid frame received on channel B. One or more valid frames were received on channel B in any static or dynamic slot during the observation period. Reset is under control of the host.
		0	No valid frame is received.
		1	Valid frame is received on channel B.
7-5	Reserved	0	Reads return 0. Writes have no effect.
4	SBVA		Slot boundary violation on channel A. One or more slot boundary violations were observed on channel A at any time during the observation period (static or dynamic slots including symbol window and NIT).
		0	No slot boundary violation is observed.
		1	Slot boundary violation is observed on channel A.
3	CIA		Communication indicator channel A. One or more valid frames were received on channel A in slots that also contained any additional communication during the observation period, that is, one or more slots received a valid frame AND had any combination of either syntax error OR content error OR slot boundary violation.
		0	No valid frame is received in slots containing any additional communication.
		1	Valid frame is received on channel A in slots containing any additional communication.
2	CEDA		Content error detected on channel A. One or more frames with a content error were received on channel A in any static or dynamic slot during the observation period.
		0	No frame with content error is received.
		1	Frame with content error is received on channel A.
1	SEDA		Syntax error detected on channel A. One or more syntax errors in static or dynamic slots including symbol window and NIT were observed on channel A.
		0	No syntax error is observed.
		1	Syntax error is observed on channel A.
0	VFRA		Valid frame received on channel A. One or more valid frames were received on channel A in any static or dynamic slot during the observation period.
		0	No valid frame is received.
		1	Valid frame is received on channel A.

26.3.2.4.10 Even Sync ID Registers (ESID[1-15])

Registers ESID1 to ESID15 hold the frame IDs of the sync frames received in **even** communication cycles, assorted in ascending order, with register ESID1 holding the lowest received sync frame ID. If the node transmits a sync frame in an even communication cycle by itself, register ESID1 holds the respective sync frame ID as configured in message buffer 0. The value is updated during the NIT of each even communication cycle. This register is reset when the Communication Controller leaves CONFIG state or enters STARTUP state.

Figure 26-155 and Table 26-134 illustrate this register.

Figure 26-155. Even Sync ID Registers (ESIDn) [offset_CC = 130h-168h]

<table>
<tr><td>31</td><td>16</td></tr>
<tr><td colspan="2">Reserved</td></tr>
<tr><td colspan="2">R-0</td></tr>
</table>

<table>
<tr><td>15</td><td>14</td><td>13</td><td>10</td><td>9</td><td>0</td></tr>
<tr><td>RXEB</td><td>RXEA</td><td colspan="2">Reserved</td><td colspan="2">EID</td></tr>
<tr><td>R-0</td><td>R-0</td><td colspan="2">R-0</td><td colspan="2">R-0</td></tr>
</table>

LEGEND: R = Read only; -*n* = value after reset

Table 26-134. Even Sync ID Registers (ESIDn) Field Descriptions

Bit	Field	Value	Description
31-16	Reserved	0	Reads return 0. Writes have no effect.
15	RXEB		Received even sync ID on channel B. A sync frame corresponding to the stored even sync ID was received on channel B.
		0	Sync frame is not received on channel B.
		1	Sync frame is received on channel B.
14	RXEA		Received even sync ID on channel A. A sync frame corresponding to the stored even sync ID was received on channel A.
		0	Sync frame is not received on channel A.
		1	Sync frame is received on channel A.
13-10	Reserved	0	Reads return 0. Writes have no effect.
9-0	EID		Even Sync ID. Sync frame ID even communication cycle.

26.3.2.4.11 Odd Sync ID Registers (OSID[1-15])

Registers OSID1 to OSID15 hold the frame IDs of the sync frames received in **odd** communication cycles, assorted in ascending order, with register OSID1 holding the lowest received sync frame ID. If the node transmits a sync frame in an odd communication cycle by itself, register OSID1 holds the respective sync frame ID as configured in message buffer 0. The value is updated during the NIT of each odd communication cycle. This register is reset when the Communication Controller leaves CONFIG state or enters STARTUP state.

Figure 26-156 and Table 26-135 illustrate this register.

Figure 26-156. Odd Sync ID Registers (OSIDn) [offset_CC = 170h-1A8h]

31-16
Reserved
R-0

15	14	13-10	9-0
RXOB	RXOA	Reserved	OID
R-0	R-0	R-0	R-0

LEGEND: R = Read only; -*n* = value after reset

Table 26-135. Odd Sync ID Registers (OSIDn) Field Descriptions

Bit	Field	Value	Description
31-16	Reserved	0	Reads return 0. Writes have no effect.
15	RXOB		Received odd sync ID on channel B. A sync frame corresponding to the stored even sync ID was received on channel B.
		0	Sync frame is not received on channel B.
		1	Sync frame is received on channel B.
14	RXOA		Received odd sync ID on channel A. A sync frame corresponding to the stored even sync ID was received on channel A.
		0	Sync frame is not received on channel A.
		1	Sync frame is received on channel A.
13-10	Reserved	0	Reads return 0. Writes have no effect.
9-0	OID		Odd Sync ID. Sync frame ID odd communication cycle.

26.3.2.4.12 Network Management Vector Registers (NMV[1-3])

The three network management registers hold the accrued NM vector (configurable 0-12 bytes). The accrued NM vector is generated by the communication controller by bit-wise ORing each NM vector received (valid frames with PPI = 1) on each channel.

The communication controller updates the NM vector at the end of each communication cycle as long as the communication controller is either in NORMAL_ACTIVE or NORMAL_PASSIVE state.

NMVn-bytes exceeding the configured NM vector length are not valid.

Figure 26-157 illustrates these registers and Table 26-136 shows the assignment of the data bytes to the network management vector.

Figure 26-157. Network Management Vector Registers (NMVn) [offset_CC = 1B0h-1B8h]

31	16
NMI	
R-0	

15	0
NMI	
R-0	

LEGEND: R = Read only; -*n* = value after reset

Table 26-136. Assignment of Data Bytes to Network Management Vector

Bit Word	31 24	23 16	15 8	7 0
NMV1	Data3	Data2	Data1	Data0
NMV2	Data7	Data6	Data5	Data4
NMV3	Data11	Data10	Data9	Data8

26.3.2.5 Message Buffer Control Registers

26.3.2.5.1 Message RAM Configuration (MRC)

The message RAM Configuration register defines the number of message buffers assigned to the static segment, dynamic segment, and FIFO. The register can be written during DEFAULT_CONFIG or CONFIG state only.

Figure 26-158 and Table 26-137 illustrate this register.

Figure 26-158. Message RAM Configuration Register (MRC) [offset_CC = 300h]

31 27	26	25 24	23 16
Reserved	SPLM*	SEC*	LCB*
R-0	R-1h	R-0	R/W-80h

15 8	7 0
FFB*	FDB*
R/W-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset; *These bits can be updated in DEFAULT_CONFIG or CONFIG state only

Table 26-137. Message RAM Configuration Register (MRC) Field Descriptions

Bit	Field	Value	Description
31-27	Reserved	0	Reads return 0. Writes have no effect.
26	SPLM		Sync Frame Payload Multiplex. This bit is only evaluated if the node is configured as sync node (SUCC1.TXSY = 1) or for single slot mode operation (SUCC1.TSM = 1). When this bit is set to 1 message buffers 0 and 1 are dedicated for sync frame transmission with different payload data on channel A and B. When this bit is set to 0, sync frames are transmitted from message buffer 0 with the same payload data on both channels. Note that the channel filter configuration for message buffer 0 resp. message buffer 1 has to be chosen accordingly.
		0	Only message buffer 0 is locked against reconfiguration.
		1	Both message buffers 0 and 1 are locked against reconfiguration.
25-24	SEC		Secure Buffers. Not evaluated when the communication controller is in DEFAULT_CONFIG or CONFIG state.
		0	Reconfiguration of message buffers is enabled with numbers < FFBh enabled. Exception: In nodes configured for sync frame transmission or for single slot mode operation message buffer 0 (and if SPLM = 1, also message buffer 1) is always locked.
		1h	Reconfiguration of message buffers with numbers < FDB and with numbers FFB is locked and transmission of message buffers for static segment with numbers FDB is disabled.
		2h	Reconfiguration of all message buffers is locked.
		3h	Reconfiguration of all message buffers is locked and transmission of message buffers for static segment with numbers FDB is disabled.
23-16	LCB		Last configured buffer.
		0-7Fh	Number of message buffers is LCB + 1.
		≥ 80h	No message buffer is configured.
15-8	FFB		First buffer of FIFO.
		0	All message buffers are assigned to the FIFO.
		0-7Fh	Message buffers from FFB to LCB are assigned to the FIFO.
		≥ 80h	No message buffer is assigned to the FIFO.
7-0	FDB		First dynamic buffer.
		0	No group of pure static buffers is configured.
		0-7Fh	Message buffers 0 to FDB - 1 are reserved for static segment.
		≥ 80h	No dynamic buffers are configured.

NOTE: In case the node is configured as sync node (SUCC1.TXSY = 1) or for single slot mode operation (SUCC1.TSM = 1), message buffer 0 resp. 1 is reserved for sync frames or single slot frames and have to be configured with the node-specific key slot ID. In case the node is neither configured as sync node nor for single slot operation message buffer 0 resp. 1 is treated like all other message buffers.

Table 26-138. Buffer Configuration

Message Buffer 0	↓ Static Buffers		
Message Buffer 1			
. . .	↓ Static + Dynamic Buffers	← **FDB**	
	↓ FIFO	← **FFB**	FIFO configured: FBB > FDB No FIFO configured: FFB ≥ 128
Message Buffer N-1			
Message Buffer N		← **LCB**	LCB ≥ FDB, LCB ≥ FFB

The programmer must ensure that the configuration defined by FDB(7-0), FFB(7-0), and LCB(7-0) is valid.

NOTE: The communication controller does not check for erroneous configurations.

NOTE: **Maximum Number of Header Sections**

The maximum number of header sections is 128. This means a maximum of 128 message buffers can be configured. The maximum length of the data sections is 254 bytes. The length of the data section may be configured different for each message buffer. In case two or more message buffers are assigned to slot 1 by use of cycle filtering, all of them must be located either in the "Static Buffers" or at the beginning of the "Static + Dynamic Buffers" section. The FlexRay protocol specification requires that each node has to send a frame in its key slot. Therefore at least message buffer 0 is reserved for transmission in the key slot. Due to this requirement a maximum number of 127 message buffers can be assigned to the FIFO. Nevertheless, a non protocol conform configuration without a transmission slot in the static segment would still be operational. The payload length configured and the length of the data sections need to be configured identical for all message buffers belonging to the FIFO via WRHS2. PLC and WRHS3.DP. When the communication controller is not in DEFAULT_CONFIG or CONFIG state reconfiguration of message buffers belonging to the FIFO is locked.

26.3.2.5.2 FIFO Rejection Filter (FRF)

The FIFO rejection filter defines a user specified sequence of bits with which channel, frame ID, and cycle count of the incoming frames are compared. Together with the FIFO rejection filter mask (FRFM), this register determines whether a message is rejected by the FIFO. The FRF register can be written during DEFAULT_CONFIG or CONFIG state only.

Figure 26-159 and Table 26-139 illustrate this register.

Figure 26-159. FIFO Rejection Filter Register (FRF) [offset_CC = 304h]

31 25	24	23	22 16
Reserved	RNF*	RSS*	CYF*
R-0	R-1h	R-1h	R/W-0

15 13	12 2	1 0
Reserved	FID*	CH*
R-0	R/W-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset; *These bits can be updated in DEFAULT_CONFIG or CONFIG state only

Table 26-139. FIFO Rejection Filter Register (FRF) Field Descriptions

Bit	Field	Value	Description
31-25	Reserved	0	Reads return 0. Writes have no effect.
24	RNF		Reject null frames. If this bit is set, received null frames are not stored in the FIFO.
		0	Null frames are stored in the FIFO.
		1	Reject all null frames.
23	RSS		Reject in static segment. If this bit is set, the FIFO is used only for the dynamic segment.
		0	FIFO also used in static segment.
		1	Reject messages in static segment.
22-16	CYF		Cycle counter filter. The 7-bit cycle counter filter determines the cycle set to which the frame ID FIFO rejection filter and the channel FIFO rejection filter are applied. In cycles not belonging to the cycle set specified by CYF, all frames are rejected. For details about the configuration of the cycle counter filter.
15-13	Reserved	0	Reads return 0. Writes have no effect.
12-2	FID	0-7FFh	Frame ID filter. A frame ID filter value of 0 means that no frame ID is rejected.
1-0	CH		Channel filter. Note: If reception on both channels is configured, also in the static segment both frames (from channel A and B) are always stored in the FIFO, even if they are identical.
		0	Receive on both channels.
		1h	Receive only on channel B.
		2h	Receive only on channel A.
		3h	No reception.

26.3.2.5.3 FIFO Rejection Filter Mask (FRFM)

The FIFO rejection filter mask specifies which of the corresponding frame ID filter bits are relevant for rejection filtering. If a bit is set, it indicates that the state of the corresponding bit in the FRF register will not be considered for rejection filtering. The FRFM register can be written during DEFAULT_CONFIG or CONFIG state only.

Figure 26-160 and Table 26-140 illustrate this register.

Figure 26-160. FIFO Rejection Filter Mask Register (FRFM) [offset_CC = 308h]

31-16
Reserved
R-0

15-13	12-2	1-0
Reserved	MFID*	Reserved
R-0	R/W-0	R-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset; *These bits can be updated in DEFAULT_CONFIG or CONFIG state only

Table 26-140. FIFO Rejection Filter Mask Register (FRFM) Field Descriptions

Bit	Field	Value	Description
31-13	Reserved	0	Reads return 0. Writes have no effect.
12-2	MFID		Mask Frame ID Filter.
		0	Corresponding frame ID filter bit is used for rejection filtering.
		1	Ignore corresponding frame ID filter bit.
1-0	Reserved	0	Reads return 0. Writes have no effect.

26.3.2.5.4 FIFO Critical Level (FCL)

The communication controller accepts modifications of the register in DEFAULT_CONFIG or CONFIG state only.

Figure 26-161 and Table 26-141 illustrate this register.

Figure 26-161. FIFO Critical Level Register (FCL) [offset_CC = 30Ch]

31-16
Reserved
R-0

15-8	7-0
Reserved	CL*
R-0	R/W-810h

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset; *These bits can be updated in DEFAULT_CONFIG or CONFIG state only

Table 26-141. FIFO Critical Level Register (FCL) Field Descriptions

Bit	Field	Value	Description
31-8	Reserved	0	Reads return 0. Writes have no effect.
7-0	CL		Critical Level. When the receive FIFO fill level FSR.RFFL is equal or greater than the critical level configured by CL, the receive FIFO critical level flag FSR.RFCL is set. If CL is programmed to values > 128, bit FSR.RFCL is never set. When FSR.RFCL changes from 0 to 1 bit SIR.RFCL is set to 1, and if enabled, an interrupt is generated.

26.3.2.6 Message Buffer Status Registers

26.3.2.6.1 Message Handler Status (MHDS)

A flag is cleared by writing a 1 to the corresponding bit position. Writing a 0 has no effect on the flag. A hardware reset will also clear the register.

Figure 26-162 and Table 26-142 illustrate this register.

Figure 26-162. Message Handler Status (MHDS) [offset_CC = 310h]

31	30	24	23	22	16
Rsvd	MBU		Rsvd	MBT	
R-0	R-0		R-0	R-0	

15	14	8	7	6	5	4	3	2	1	0
Rsvd	FMB		CRAM	MFMB	FMBD	PTFB2	PTFB1	PMR	POBF	PIBF
R-0	R-0		R-1h	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 26-142. Message Handler Status (MHDS) Field Descriptions

Bit	Field	Value	Description
31	Reserved	0	Reads return 0. Writes have no effect.
30-24	MBU	0-7Fh	Message buffer updated. Number of the message buffer that was updated last by the communication controller. For this message buffer, the respective ND and/or MBC flag in the new data 1...4 (NDAT1...4) and the message buffer status changed 1...4 (MBSC1...4) registers are also set. **Note: MBU are reset when the communication controller leaves CONFIG state or enters STARTUP state.**
23	Reserved	0	Reads return 0. Writes have no effect.
22-16	MBT	0-7Fh	Message buffer transmitted. Number of the last successfully transmitted message buffer. If the message buffer is configured for single-shot mode, the respective TXR flag in the Transmission request register 1...4 (TXRQ1..4) was reset. **Note: MBT are reset when the communication controller leaves CONFIG state or enters STARTUP state.**
15	Reserved	0	Reads return 0. Writes have no effect.
14-8	FMB	0-7Fh	Faulty message buffer. An ECC multi-bit error occurred when reading from a message buffer or when transferring data from Input Buffer or Transient Buffer 1,2 to the message buffer referenced by FMB. This value is only valid when one of the flags PIBF, PMR, PTBF1, PTBF2, and flag FMBD is set. Is not updated while flag FMBD is set.
7	CRAM		Clear all internal RAMs. Signals that execution of the controller host interface command CLEAR_RAMS is ongoing (all bits of all internal RAM blocks are written to 0). The bit is set by hardware reset or by the controller host interface command CLEAR_RAMS.
		0	No execution of the controller host interface command CLEAR_RAMS.
		1	Execution of the controller host interface command CLEAR_RAMS is ongoing.
6	MFMB		Multiple faulty message buffers detected.
		0	No additional faulty message buffer.
		1	Another faulty message buffer was detected while flag FMBD is set.
5	FMBD		Faulty message buffer detected.
		0	No faulty message buffer.
		1	Message buffer referenced by FMB holds faulty data due to an ECC multi-bit error.
4	PTBF2		ECC error transient buffer RAM B.
		0	No ECC multi-bit error.
		1	ECC multi-bit error occurred when reading transient buffer RAM B.
3	PTBF1		ECC error transient buffer RAM A.
		0	No ECC multi-bit error.
		1	ECC multi-bit error occurred when reading transient buffer RAM A.

Table 26-142. Message Handler Status (MHDS) Field Descriptions (continued)

Bit	Field	Value	Description
2	PMR		ECC error message RAM.
		0	No ECC multi-bit error.
		1	ECC multi-bit error occurred when reading message RAM.
1	POBF		ECC error output buffer RAM 1, 2.
		0	No ECC multi-bit error.
		1	ECC multi-bit error occurred when message handler read output buffer RAM 1,2.
0	PIBF		ECC error input buffer RAM 1, 2.
		0	No ECC multi-bit error.
		1	ECC multi-bit error occurred when message handler read input buffer RAM 1,2.

NOTE: When one of the flags PIBF, POBF, PMR, PTBF1, PTBF2 changes from 0 to 1, the PERR flag in the Error Interrupt Register (EIR) is set to 1.

26.3.2.6.2 Last Dynamic Transmit Slot (LDTS)

The register is reset when the communication controller leaves CONFIG state or enters STARTUP state

Figure 26-163 and Table 26-143 illustrate this register.

Figure 26-163. Last Dynamic Transmit Slot (LDTS) [offset_CC = 314h]

31 – 27	26 – 16
Reserved	LDTB
R-0	R-0

15 – 11	10 – 0
Reserved	LDTA
R-0	R-0

LEGEND: R = Read only; -*n* = value after reset

Table 26-143. Last Dynamic Transmit Slot (LDTS) Field Descriptions

Bit	Field	Value	Description
31-27	Reserved	0	Reads return 0. Writes have no effect.
26-16	LDTB		Last Dynamic Transmission Channel B. Value of Slot Counter B at the time of the last frame transmission on channel A in the dynamic segment of this node. It is updated at the end of the dynamic segment and is reset to 0 if no frame was transmitted during the dynamic segment.
15-11	Reserved	0	Reads return 0. Writes have no effect.
10-0	LDTA		Last Dynamic Transmission Channel A. Value of Slot Counter A at the time of the last frame transmission on channel A in the dynamic segment of this node. It is updated at the end of the dynamic segment and is reset to 0 if no frame was transmitted during the dynamic segment.

26.3.2.6.3 FIFO Status Register (FSR)

The register is reset when the communication controller leaves CONFIG state, enters STARTUP state, or by CHI command CLEAR_RAMS..

Figure 26-164 and Table 26-144 illustrate this register.

Figure 26-164. FIFO Status Register (FSR) [offset_CC = 318h]

31 16
Reserved
R-0

15 8	7 3	2	1	0
RFFL	Reserved	RFO	RFCL	RFNE
R-0	R-0	R-0	R-0	R-0

LEGEND: R = Read only; -*n* = value after reset

Table 26-144. FIFO Status Register (FSR) Field Descriptions

Bit	Field	Value	Description
31-16	Reserved	0	Reads return 0. Writes have no effect.
15-8	RFFL	0-7Fh	Receive FIFO Fill Level. Number of FIFO buffers filled with new data not yet read by the Host.
7-7	Reserved	0	Reads return 0. Writes have no effect.
2	RFO		Receive FIFO Overrun. The flag is set by the communication controller when a receive FIFO overrun is detected. When a receive FIFO overrun occurs, the oldest message is overwritten with the actual received message. In addition, interrupt flag RFO in the Error Interrupt Register (EIR) is set. The flag is cleared by the next FIFO read access issued by the Host.
		0	No receive FIFO overrun is detected.
		1	A receive FIFO overrun is detected.
1	RFCL		Receive FIFO Critical Level. This flag is set when the receive FIFO fill level RFFL is equal or greater than the critical level as configured by CL in the FIFO Critical Level register (FCL). The flag is cleared by the communication controller as soon as RFFL drops below FCL.CL. When RFCL changes from 0 to 1, the RFCL flag in the Status Interrupt register (SIR) is set to 1, and if enabled, an interrupt is generated.
		0	Receive FIFO is below critical level.
		1	Receive FIFO critical level is reached.
0	RFNE		Receive FIFO Not Empty. This flag is set by the communication controller when a received valid frame (data or null frame depending on rejection mask) was stored in the FIFO. In addition, interrupt flag RFNE in the Status Interrupt register (SR) is set. The bit is reset after the Host has read all message from the FIFO.
		0	Receive FIFO is empty.
		1	Receive FIFO is not empty.

26.3.2.6.4 Message Handler Constraints Flags (MHDF)

Some constraints exist for the Message Handler regarding VBUSclk frequency, Message RAM configuration, and FlexRay bus traffic. In order to simplify software development, constraints violations are reported by setting flags in the MHDF.

A flag is cleared by writing a 1 to the corresponding bit position. Writing a 0 has no effect on the flag. A hardware reset will also clear the register. The register is reset when the communication controller leaves CONFIG state, enters STARTUP state, or by CHI command CLEAR_RAMS.

Figure 26-165 and Table 26-145 illustrate this register.

Figure 26-165. Message Handler Constraints Flags (MHDF) [offset_CC = 31Ch]

31-16
Reserved
R-0

15-9	8	7	6	5	4	3	2	1	0
Reserved	WAHP	TNSA	TNSB	TBFB	TBFA	FNFB	FNFA	SNUB	SNUA
R-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 26-145. Message Handler Constraint Flags (MHDF) Field Descriptions

Bit	Field	Value	Description
31-9	Reserved	0	Reads return 0. Writes have no effect.
8	WAHP		Write attempt to header partition. This flag is set by the communication controller when the message handler tries to write message data into the header partition of the Message RAM due to faulty configuration of a message buffer. The write attempt is not executed, to protect the header partition from unintended write accesses.
		0	No write attempt to header partition.
		1	Write attempt to header partition.
7	TNSA		Transmission Not Started Channel A. This flag is set by the CC when the Message Handler was not ready to start a scheduled transmission on channel A at the action point of the configured slot.
		0	No transmission is not started on channel A.
		1	Transmission is not started on channel A.
6	TNSB		Transmission Not Started Channel B. This flag is set by the CC when the Message Handler was not ready to start a scheduled transmission on channel B at the action point of the configured slot.
		0	No transmission is not started on channel B.
		1	Transmission is not started on channel B.
5	TBFB		Transient buffer access failure B. This flag is set by the communication controller when a read or write access to TBF B requested by PRT B could not complete within the available time.
		0	No TBF B access failure.
		1	TBF B access failure.
4	TBFA		Transient buffer access failure A. This flag is set by the communication controller when a read or write access to TBF A requested by PRT A could not complete within the available time.
		0	No TBF A access failure.
		1	TBF A access failure.
3	FNFB		Find sequence not finished channel B. This flag is set by the communication controller when the Message Handler, due to overload condition, was not able to finish a find sequence (scan of Message RAM for matching message buffer) with respect to channel B.
		0	No find sequence is not finished for channel B.
		1	Find sequence is not finished for channel B.
2	FNFA		Find sequence not finished channel A. This flag is set by the communication controller when the Message Handler, due to overload condition, was not able to finish a find sequence (scan of Message RAM for matching message buffer) with respect to channel A.
		0	No find sequence is not finished for channel A.
		1	Find sequence is not finished for channel A.

Table 26-145. Message Handler Constraint Flags (MHDF) Field Descriptions (continued)

Bit	Field	Value	Description
1	SNUB		Status not updated channel B. This flag is set by the communication controller when the Message Handler, due to overload condition, was not able to update a message buffer's status MBS with respect to channel B.
		0	No overload condition occurred when updating MBS for channel B.
		1	MBS for channel B is not updated.
0	SNUA		Status not updated channel A. This flag is set by the communication controller when the Message Handler, due to overload condition, was not able to update a message buffer's status MBS with respect to channel A.
		0	No overload condition occurred when updating MBS for channel A.
		1	MBS for channel A is not updated.

NOTE: When one of the flags SNUA, SNUB, FNFA, FNFB, TBFA, TBFB, WAHP changes from 0 to 1, interrupt flag MHF in the Error Interrupt register (EIR) is set to 1.

26.3.2.6.5 Transmission Request Registers (TXRQ[1-4])

These four registers reflect the state of the TXR flags of all configured message buffers. The flags are evaluated for transmit buffers only. If the number of configured message buffers is less than 128, the remaining TXR flags have no meaning.

Figure 26-166 through Figure 26-169 and Table 26-146 illustrate these registers.

Figure 26-166. Transmission Request Register 4 (TXRQ4) [offset_CC = 32Ch]

<table>
<tr><td>31</td><td>16</td></tr>
<tr><td colspan="2">TXR[127:112]</td></tr>
<tr><td colspan="2">R-0</td></tr>
<tr><td>15</td><td>0</td></tr>
<tr><td colspan="2">TXR[111:96]</td></tr>
<tr><td colspan="2">R-0</td></tr>
</table>

LEGEND: R = Read only; -*n* = value after reset

Figure 26-167. Transmission Request Register 3 (TXRQ3) [offset_CC = 328h]

<table>
<tr><td>31</td><td>16</td></tr>
<tr><td colspan="2">TXR[95:80]</td></tr>
<tr><td colspan="2">R-0</td></tr>
<tr><td>15</td><td>0</td></tr>
<tr><td colspan="2">TXR[79:64]</td></tr>
<tr><td colspan="2">R-0</td></tr>
</table>

LEGEND: R = Read only; -*n* = value after reset

Figure 26-168. Transmission Request Register 2 (TXRQ2) [offset_CC = 324h]

<table>
<tr><td>31</td><td>16</td></tr>
<tr><td colspan="2">TXR[63:48]</td></tr>
<tr><td colspan="2">R-0</td></tr>
<tr><td>15</td><td>0</td></tr>
<tr><td colspan="2">TXR[47:32]</td></tr>
<tr><td colspan="2">R-0</td></tr>
</table>

LEGEND: R = Read only; -*n* = value after reset

Figure 26-169. Transmission Request Register 1 (TXRQ1) [offset_CC = 320h]

<table>
<tr><td>31</td><td>16</td></tr>
<tr><td colspan="2">TXR[31:16]</td></tr>
<tr><td colspan="2">R-0</td></tr>
<tr><td>15</td><td>0</td></tr>
<tr><td colspan="2">TXR[15:0]</td></tr>
<tr><td colspan="2">R-0</td></tr>
</table>

LEGEND: R = Read only; -*n* = value after reset

Table 26-146. Transmission Request Registers (TXRQn) Field Description

Bit	Field	Value	Description
127-0	TXR[*n*]		Transmission request.
		0	The respective message buffer is not ready for transmission.
		1	If the flag is set, the respective message buffer is ready for transmission. Respectively, transmission of this message buffer is in progress. In single-shot mode the flags are reset after transmission has completed.

26.3.2.6.6 New Data Registers (NDAT[1-4])

The four registers reflect the state of the ND flags of all configured message buffers. **ND** flags corresponding to transmit buffers have no meaning. If the number of configured message buffers is less than 128, the remaining ND flags have no meaning. The registers are reset when the communication controller leaves CONFIG state or enters STARTUP state.

Figure 26-170 through Figure 26-173 and Table 26-147 illustrate these registers.

Figure 26-170. New Data Register 4 (NDAT4) [offset_CC = 33Ch]

31	16
ND[127:112]	
R-0	

15	0
ND[111:96]	
R-0	

LEGEND: R = Read only; -*n* = value after reset

Figure 26-171. New Data Register 3 (NDAT3) [offset_CC = 338h]

31	16
ND[95:80]	
R-0	

15	0
ND[79:64]	
R-0	

LEGEND: R = Read only; -*n* = value after reset

Figure 26-172. New Data Register 2 (NDAT2) [offset_CC = 334h]

31	16
ND[63:48]	
R-0	

15	0
ND[47:32]	
R-0	

LEGEND: R = Read only; -*n* = value after reset

Figure 26-173. New Data Register 1 (NDAT1) [offset_CC = 330h]

31	16
ND[31:16]	
R-0	

15	0
ND[15:0]	
R-0	

LEGEND: R = Read only; -*n* = value after reset

Table 26-147. New Data Registers (NDATn) Field Descriptions

Bit	Field	Value	Description
127-0	ND[*n*]		New data.
		0	The flags are reset when the header section of the corresponding message buffer is reconfigured or when the data section has been transferred to the output buffer.
		1	The flags are set when a valid received data frame matches the message buffer's filter configuration, independent of the payload length received or the payload length configured for that message buffer. The flags are not set after reception of null frames except for message buffers belonging to the receive FIFO.

26.3.2.6.7 Message Buffer Status Changed Registers (MBSC[1-4])

The four registers reflect the state of the MBC flags of all configured message buffers. If the number of configured message buffers is less than 128, the remaining MBC flags have no meaning.

Figure 26-174 through Figure 26-177 and Table 26-148 illustrate these registers.

Figure 26-174. Message Buffer Status Changed Register 4 (MBSC4) [offset_CC = 34Ch]

31	16
MBS[127:112]	
R-0	

15	0
MBS[111:96]	
R-0	

LEGEND: R = Read only; -*n* = value after reset

Figure 26-175. Message Buffer Status Changed Register 3 (MBSC3) [offset_CC = 348h]

31	16
MBS[95:80]	
R-0	

15	0
MBS[79:64]	
R-0	

LEGEND: R = Read only; -*n* = value after reset

Figure 26-176. Message Buffer Status Changed Register 2 (MBSC2) [offset_CC = 344h]

31	16
MBS[63:48]	
R-0	

15	0
MBS[47:32]	
R-0	

LEGEND: R = Read only; -*n* = value after reset

Figure 26-177. Message Buffer Status Changed Register 1 (MBSC1) [offset_CC = 340h]

31	16
MBS[31:16]	
R-0	

15	0
MBS[15:0]	
R-0	

LEGEND: R = Read only; -*n* = value after reset

Table 26-148. Message Buffer Status Changed Registers (MBSCn) Field Descriptions

Bit	Field	Value	Description
127-0	MBS[*n*]		Message buffer status changed.
		0	A flag is reset when the header section of the corresponding message buffer is reconfigured or when it has been transferred to the Output Buffer.
		1	The flag is set whenever the Message Handler changes one of the status flags VFRA, VFRB, SEOA, SEOB, CEOA, CEOB, SVOA, SVOB, TCIA, TCIB, ESA, ESB, MLST, FTA, FTB in the header section (see Message Buffer Status (MBS)) of the respective message buffer.

26.3.2.7 Identification Registers

26.3.2.7.1 Core Release Register (CREL)

Figure 26-178 and Table 26-149 illustrate this register.

Figure 26-178. Core Release Register (CREL) [offset_CC = 3F0h]

<table>
<tr><td>31</td><td>28</td><td>27</td><td>20</td><td>19</td><td>16</td></tr>
<tr><td colspan="2">REL</td><td colspan="2">STEP</td><td colspan="2">YEAR</td></tr>
<tr><td colspan="2">R-release info</td><td colspan="2">R-release info</td><td colspan="2">R-release info</td></tr>
</table>

<table>
<tr><td>15</td><td>8</td><td>7</td><td>0</td></tr>
<tr><td colspan="2">MON</td><td colspan="2">DAY</td></tr>
<tr><td colspan="2">R-release info</td><td colspan="2">R-release info</td></tr>
</table>

LEGEND: R = Read only; -*n* = value after reset

Table 26-149. Core Release Register (CREL) Field Descriptions

Bit	Field	Value	Description
31-28	REL	0-Fh	Core Release. One digit, BCD-coded.
27-20	STEP	0-FFh	Step of Core Release. Two digits, BCD-coded.
19-16	YEAR	0-Fh	Design Time Stamp, Year. One digit, BCD-coded.
15-8	MON	0-FFh	Design Time Stamp, Month. Two digits, BCD-coded.
7-0	DAY	0-FFh	Design Time Stamp, Day. Two digits, BCD-coded.

Table 26-150 shows the release coding in register CREL.

Table 26-150. Release Coding

Release	Step	Sub-Step	Core Release Register Contents	Name
1	0	0	1006 0519	Revision 1.0.0
1	0	1	1016 1211	Revision 1.0.1
1	0	2	10271031	Revision 1.0.2
1	0	3	10390206	Revision 1.0.3

26.3.2.7.2 Endian Register (ENDN)

Figure 26-179 and Table 26-151 illustrate this register.

Figure 26-179. Endian Register (ENDN) [offset_CC = 3F4h]

<table>
<tr><td>31</td><td>16</td></tr>
<tr><td colspan="2">ETV</td></tr>
<tr><td colspan="2">R-8765h</td></tr>
</table>

<table>
<tr><td>15</td><td>0</td></tr>
<tr><td colspan="2">ETV</td></tr>
<tr><td colspan="2">R-4321h</td></tr>
</table>

LEGEND: R = Read only; -*n* = value after reset

Table 26-151. Endian Register (ENDN) Field Descriptions

Bit	Field	Description
31-0	ETV	Endianness Test Value. The Endianness test value is 87654321h.

26.3.2.8 Input Buffer

Double buffer structure consisting of input buffer host and input buffer shadow. While the host can write to input buffer host, the transfer to the message RAM is done from input buffer shadow. The input buffer holds the header and data sections to be transferred to the selected message buffer in the message RAM. It is used to configure the message buffers in the message RAM and to update the data sections of transmit buffers.

When updating the header section of a message buffer in the Message RAM from the Input Buffer, the Message Buffer Status as described in Message Buffer Status (MBS), Message Buffer Status (MBS) is automatically reset to 0.

The header sections of message buffers belonging to the receive FIFO can only be (re)configured when the communication controller is in DEFAULT_CONFIG or CONFIG state. For those message buffers only the payload length configured and the data pointer need to be configured by bits PLC of the Write Header Section 2 (WRHS2) and by bits DP of Write Header Section 3 (WRHS3). All information required for acceptance filtering is taken from the FIFO rejection filter and the FIFO rejection filter mask.

26.3.2.8.1 Write Data Section Registers (WRDS[1-64])

Holds the data words to be transferred to the data section of the addressed message buffer. The data words (DW $_n$) are written to the message RAM in transmission order from DW $_1$(byte0, byte1) to DW $_{PL}$(DW $_{PL}$= number of data words as defined by the payload length configured in PLC of the Write Header Section 2 (WRHS2).

Figure 26-180 and Table 26-152 illustrate this register.

Figure 26-180. Write Data Section Registers (WRDSn) [offset_CC = 400h-4FCh]

31 ... 16
MD
R/W-0

15 ... 0
MD
R/W-0

LEGEND: R/W = Read/Write; -*n* = value after reset

Table 26-152. Write Data Section Registers (WRDSn) Field Descriptions

Bit	Field	Description
31-0	MD	Message data. **Note: DW127 is located on WRDS64.MD. In this case WRDS64.MD is unused (no valid data).The input buffer RAMs are initialized to 0 when leaving hardware reset or by the controller host interface command CLEAR_RAMS.** MD(31-24) = DW $_{2n}$, byte $_{4n-1}$ MD(23-16) = DW $_{2n}$, byte $_{4n-2}$ MD(15-8) = DW $_{2n-1}$, byte $_{4n-3}$ MD(7-0) = DW $_{2n-1}$, byte $_{4n-4}$

26.3.2.8.2 Write Header Section Register 1 (WRHS1)

Figure 26-181 and Table 26-153 illustrate this register.

Figure 26-181. Write Header Section Register 1 (WRHS1) [offset_CC = 500h]

31 30	29	28	27	26	25	24	23	22 16
Reserved	MBI	TXM	PPIT	CFG	CHB	CHA	Rsvd	CYC
R-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R-0	R/W-0

15 11	10 0
Reserved	FID
R-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 26-153. Write Header Section Register 1 (WRHS1) Field Descriptions

Bit	Field	Value	Description
31-30	Reserved	0	Reads return 0. Writes have no effect.
29	MBI		Message buffer interrupt. This bit enables the receive/transmit interrupt for the corresponding message buffer. After a dedicated receive buffer has been updated by the message handler, flag RXI and/or MBSI in the status interrupt register are set. After successful transmission the TXI flag in the status interrupt register is set.
		0	The corresponding message buffer interrupt is enabled.
		1	The corresponding message buffer interrupt is disabled.
28	TXM		Transmission mode. This bit is used to select the transmission mode.
		0	Continuous mode.
		1	Single-shot mode.
27	PPIT		Payload preamble indicator transmit. This bit is used to control the state of the Payload Preamble Indicator in transmit frames. If the bit is set in a static message buffer, the respective message buffer holds network management information. If the bit is set in a dynamic message buffer, the first two bytes of the payload segment may be used for message ID filtering by the receiver. Message ID filtering of received FlexRay frames is not supported by the FlexRay module, but can be done by the host CPU.
		0	Payload Preamble Indicator is not set.
		1	Payload Preamble Indicator is set.
26	CFG		Message buffer configuration bit. This bit is used to configure the corresponding buffer as transmit buffer or as receive buffer. For message buffers belonging to the receive FIFO the bit is not evaluated.
		0	The corresponding buffer is configured as receive buffer.
		1	The corresponding buffer is configured as transmit buffer.
25-24	CHB, CHA	0-3h	Channel filter control. The 2-bit channel filtering field associated with each buffer serves as a filter for receive buffers and as a control field for transmit buffers. See Table 26-154 for bit descriptions. **Note: If a message buffer is configured for the dynamic segment and both bits of the channel filtering field are set to 1, no frames are transmitted resp. received frames are ignored (same function as CHA = CHB = 0)**
23	Reserved	0	Reads return 0. Writes have no effect.
22-16	CYC	0-7Fh	Cycle code. The 7-bit cycle code determines the cycle set used for cycle counter filtering.
15-11	Reserved	0	Reads return 0. Writes have no effect.
10-0	FID	0-7FFh	Frame ID. Frame ID of the selected message buffer. The frame ID defines the slot number for transmission / reception of the respective message. **Note: Message buffers with frame ID = 0 are considered not valid.**

Table 26-154. Channel Filter Control Bit Descriptions

CHA	CHB	Transmit Buffer transmit frame on	Receive Buffer store frame received from
1	1	both channels (static segment only)	channel A or B (store first semantically valid frame, static segment only)
1	0	channel A	channel A
0	1	channel B	channel B
0	0	no transmission	ignore frame

26.3.2.8.3 Write Header Section Register 2 (WRHS2)

Figure 26-182 and Table 26-155 illustrate this register.

Figure 26-182. Write Header Section Register 2 (WRHS2) [offset_CC = 504h]

31 - 23	22 - 16
Reserved	PLC
R-0	R/W-0

15 - 11	10 - 0
Reserved	CRC
R-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 26-155. Write Header Section Register 2 (WRHS2) Field Descriptions

Bit	Field	Value	Description
31-23	Reserved	0	Reads return 0. Writes have no effect.
22-16	PLC	0-7Fh	Payload length configured. Length of data section (number of 2-byte words) as configured by the host. During static segment the static frame data length as configured by SFDL in the MHD configuration register defines the payload length for all static frames. If the payload length configured by PLC is shorter than this value padding bytes are inserted to ensure that frames have proper physical length. The padding pattern is logical 0.
15-11	Reserved	0	Reads return 0. Writes have no effect.
10-0	CRC	0-7FFh	Header CRC. Receive Buffer: configuration not required. Transmit buffer: Header CRC calculated and configured by the host. For calculation of the header CRC the payload length of the frame send on the bus has to be considered. In static segment the payload length of all frames is configured by MHDC.SFDL.

26.3.2.8.4 Write Header Section Register 3 (WRHS3)

Figure 26-183 and Table 26-156 illustrate this register.

Figure 26-183. Write Header Section Register 3 (WRHS3) [offset_CC = 508h]

<table>
<tr><td>31</td><td>16</td></tr>
<tr><td colspan="2">Reserved</td></tr>
<tr><td colspan="2">R-0</td></tr>
</table>

<table>
<tr><td>15</td><td>11</td><td>10</td><td>0</td></tr>
<tr><td colspan="2">Reserved</td><td colspan="2">DP</td></tr>
<tr><td colspan="2">R-0</td><td colspan="2">R/W-0</td></tr>
</table>

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 26-156. Write Header Section Register 3 (WRHS3) Field Descriptions

Bit	Field	Value	Description
31-11	Reserved	0	Reads return 0. Writes have no effect.
10-0	DP	1-7FFh	Data pointer. Pointer to the first 32-bit word of the data section of the addressed message buffer in the message RAM.

26.3.2.8.5 Input Buffer Command Mask Register (IBCM)

Configures how the message buffer in the message RAM selected by the input buffer command request register is updated. When IBF host and IBF shadow are swapped, also mask bits LHSH, LDSH, and STXRH are swapped with bits LHSS, LDSS, and STXRS to keep them attached to the respective input buffer transfer.

Figure 26-184 and Table 26-157 illustrate this register.

Figure 26-184. Input Buffer Command Mask Register (IBCM) [offset_CC = 510h]

31-19	18	17	16
Reserved	STXRS	LDSS	LHSS
R-0	R-0	R-0	R-0

15-3	2	1	0
Reserved	STXRH	LDSH	LHSH
R-0	R/W-0	R/W-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 26-157. Input Buffer Command Mask Register (IBCM) Field Descriptions

Bit	Field	Value	Description
31-19	Reserved	0	Reads return 0. Writes have no effect.
18	STXRS		Set transmission request shadow.
		0	Reset TXR flag.
		1	Set TXR flag; transmit buffer is released for transmission (operation ongoing or finished).
17	LDSS		Load data section shadow.
		0	Data section is not updated.
		1	Data section is selected for transfer from input buffer to the message RAM (transfer ongoing or finished).
16	LHSS		Load header section shadow.
		0	Header section is not updated.
		1	Header section is selected for transfer from input buffer to the message RAM (transfer ongoing or finished).
15-3	Reserved	0	Reads return 0. Writes have no effect.
2	STXRH		Set transmission request host. If this bit is set to 1, the transmission request flag TXR for the selected message buffer is set in the transmission request registers to release the message buffer for transmission. In single-shot mode the flag is cleared by the communication controller after transmission has completed. The flags is evaluated for transmit buffers only.
		0	Reset transmission request flag.
		1	Set transmission request flag; transmit buffer is released for transmission.
1	LDSH		Load data section host.
		0	Data section is not updated.
		1	Data section is selected for transfer from input buffer to the message RAM.
0	LHSH		Load header section host.
		0	Header section is not updated.
		1	Header section is selected for transfer from input buffer to the message RAM.

26.3.2.8.6 Input Buffer Command Request Register (IBCR)

When the host writes the number of a target message buffer in the message RAM to IBRH in the input buffer command request register, IBF host and IBF shadow are swapped. In addition the message buffer numbers stored under IBRH and IBRS are also swapped.

With this write operation the IBSYS bit in the input buffer command request register is set to 1. The message handler then starts to transfer the contents of IBF shadow to the message buffer in the message RAM selected by IBRS.

While the message handler transfers the data from IBF shadow to the target message buffer in the message RAM, the host may configure the next message in the IBF host. After the transfer between IBF shadow and the message RAM has completed, the IBSYS bit is set back to 0 and the next transfer to the message RAM may be started by the host by writing the respective target message buffer number to IBRH.

If a write access to IBRH occurs while IBSYS is 1, IBSYH is set to 1. After completion of the ongoing data transfer from IBF shadow to the message RAM, IBF host and IBF shadow are swapped, IBSYH is reset to 0. IBSYS remains set to 1, and the next transfer to the message RAM is started. In addition the message buffer numbers stored under IBRH and IBRS are also swapped.

Any write access to an Input Buffer Register while both IBSYS and IBSYH are set will cause the error flag IIBA in the Error Interrupt Register (EIR) to be set. In this case the Input Buffer will not be changed.

Figure 26-185 and Table 26-158 illustrate this register.

Figure 26-185. Input Buffer Command Request Register (IBCR) [offset_CC = 514h]

31	30 - 23	22 - 16
IBSYS	Reserved	IBRS
R-0	R-0	R-0

15	14 - 7	6 - 0
IBSYH	Reserved	IBRH
R-0	R-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset; *These bits can be updated in DEFAULT_CONFIG or CONFIG state only

Table 26-158. Input Buffer Command Request Register (IBCR) Field Descriptions

Bit	Field	Value	Description
31	IBSYS		Input buffer busy shadow. Set to 1 after writing IBRH. When the transfer between IBF shadow and the message RAM has completed, IBSYS is set back to 0.
		0	Transfer between IBF shadow and message RAM is completed.
		1	Transfer between IBF shadow and message RAM is in progress.
30-23	Reserved	0	Reads return 0. Writes have no effect.
22-16	IBRS	0-7Fh	Input buffer request shadow. Number of the target message buffer actually updated / lately updated.
15	IBSYH		Input buffer busy host. Set to 1 by writing IBRH while IBSYS is still 1. After the ongoing transfer between IBF shadow and the message RAM has completed, the IBSYH is set back to 0.
		0	No request is pending.
		1	Request while transfer between IBF shadow and message RAM is in progress.
14-7	Reserved	0	Reads return 0. Writes have no effect.
6-0	IBRH	0-7Fh	Input buffer request host. Selects the target message buffer in the Message RAM for data transfer from Input Buffer.

26.3.2.9 Output Buffer

Double buffer structure consisting of output buffer host and output buffer shadow. While the host can read from output buffer host, the transfer from the message RAM is done to output buffer shadow. The output buffer holds the header and data sections of requested message buffers transferred from the message RAM. Used to read out message buffers from the message RAM.

26.3.2.9.1 Read Data Section Registers (RDDS[1-64])

Holds the data words read from the data section of the addressed message buffer. The data words (DW $_n$) are read from the message RAM in reception order from DW $_1$(byte0, byte1) to DW $_{PL}$(DW $_{PL}$= number of data words as defined by the payload length configured in bits PLC(6-0) of the Read Header Section 2 (RDHS2)).

Figure 26-186 and Table 26-159 illustrate this register.

Figure 26-186. Read Data Section Registers (RDDSn) [offset_CC = 600h-6FCh]

31	16
MD	
R/W-0	

15	0
MD	
R/W-0	

LEGEND: R/W = Read/Write; -*n* = value after reset

Table 26-159. Read Data Section Registers (RDDSn) Field Descriptions

Bit	Field	Description
31-0	MD	Message data. **Note: DW127 is located on RDDS64.MD. In this case, RDDS64.MD is unused (no valid data).The input buffer RAMs are initialized to 0 when leaving hardware reset or by the controller host interface command CLEAR_RAMS.** MD(31-24) = DW $_{2n}$, byte $_{4n-1}$ MD(23-16) = DW $_{2n}$, byte $_{4n-2}$ MD(15-8) = DW $_{2n-1}$, byte $_{4n-3}$ MD(7-0) = DW $_{2n-1}$, byte $_{4n-4}$

26.3.2.9.2 Read Header Section Register 1 (RDHS1)

Figure 26-187 and Table 26-160 illustrate this register.

Figure 26-187. Read Header Section Register 1 (RDHS1) [offset_CC = 700h]

31 30	29	28	27	26	25	24	23	22 16
Reserved	MBI	TXM	PPIT	CFG	CHB	CHA	Rsvd	CYC
R-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R-0	R/W-0

15 11	10 0
Reserved	FID
R-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 26-160. Read Header Section Register 1 (RDHS1) Field Descriptions

Bit	Field	Value	Description
31-30	Reserved	0	Reads return 0. Writes have no effect.
29	MBI		Message buffer interrupt.
		0	The corresponding message buffer interrupt is enabled.
		1	The corresponding message buffer interrupt is disabled.
28	TXM		Transmission mode. This bit is used to select the transmission mode.
		0	Continuous mode.
		1	Single-shot mode.
27	PPIT		Payload preamble indicator transmit.
		0	Payload Preamble Indicator is not set.
		1	Payload Preamble Indicator is set.
26	CFG		Message buffer configuration bit.
		0	The corresponding buffer is configured as receive buffer.
		1	The corresponding buffer is configured as Transmit buffer.
25-24	CHB, CHA		Channel filter control. See Table 26-154 for bit descriptions.
23	Reserved	0	Reads return 0. Writes have no effect.
22-16	CYC	0-7Fh	Cycle code. The 7-bit cycle code determines the cycle set used for cycle counter filtering.
15-11	Reserved	0	Reads return 0. Writes have no effect.
10-0	FID	0-7FFh	Frame ID. Frame ID of the selected message buffer. **Note: Message buffers with frame ID = 0 are considered not valid.**

NOTE: In case the message buffer read from the message RAM belongs to the receive FIFO, FID, and CHA, CHB were updated from the received frame while CYC, CFG, PPIT, TXM, and MBI are reset to 0.

For bit description, see also Section 26.3.2.8.2.

26.3.2.9.3 Read Header Section Register 2 (RDHS2)

Figure 26-188 and Table 26-161 illustrate this register.

Figure 26-188. Read Header Section Register 2 (RDHS2) [offset_CC = 704h]

31	30 24	23	22 16
Rsvd	PLR	Rsvd	PLC
R-0	R-0	R-0	R-0

15 11	10 0
Reserved	CRC
R-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 26-161. Read Header Section Register 2 (RDHS2) Field Descriptions

Bit	Field	Value	Description
31	Reserved	0	Reads return 0. Writes have no effect.
30-24	PLR	0-7Fh	Payload length received. Payload length value updated from received frame (exception: if message buffer belongs to the receive FIFO PLR is also updated from received null frames). When a message is stored into a message buffer the following behavior with respect to payload length received and payload length configured is implemented: **PLR > PLC:**The payload data stored in the message buffer is truncated to the payload length configured if PLC even or else truncated to PLC + 1. **PLR <= PLC:** The received payload data is stored into the message buffers data section. The remaining data bytes of the data section as configured by PLC are filled with undefined data. **PLR = zero:** The message buffers data section is filled with undefined data. **PLC = zero:** Message buffer has no data section configured. No data is stored into the message buffers data section.
23	Reserved	0	Reads return 0. Writes have no effect.
22-16	PLC	0-7Fh	Payload length configured. Length of data section (number of 2-byte words) as configured by the host.
15-11	Reserved	0	Reads return 0. Writes have no effect.
10-0	CRC	0-7FFh	Header CRC. Receive buffer: Header CRC is updated from receive frame. Transmit buffer: Header CRC is calculated and configured by the host.

NOTE: The Message RAM is organized in 4-byte words. When received data is stored into a message buffer's data section, the number of 2-byte data words written into the message buffer is PLC rounded to the next even value. PLC should be configured identical for all message buffers belonging to the receive FIFO. Header 2 is updated from data frames only.

26.3.2.9.4 Read Header Section Register 3 (RDHS3)

Figure 26-189 and Table 26-162 illustrate this register.

Figure 26-189. Read Header Section Register 3 (RDHS3) [offset_CC = 708h]

31 30	29	28	27	26	25	24	23 22	21 16
Reserved	RES	PPI	NFI	SYN	SFI	RCI	Reserved	RCC
R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0

15 11	10 0
Reserved	DP
R-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 26-162. Read Header Section Register 3 (RDHS3) Field Descriptions

Bit	Field	Value	Description
31-30	Reserved	0	Reads return 0. Writes have no effect.
29	RES	0-1	Reserved bit. Reflects the state of the received reserved bit. The reserved bit is transmitted as 0.
28	PPI		Payload preamble indicator. The payload preamble indicator defines whether a network management vector or message ID is contained within the payload segment of the received frame.
		0	The payload segment of the received frame does not contain a network management vector or a message ID.
		1	*Static segment:* Network management vector at the beginning of the payload.
			Dynamic segment: Message ID at the beginning of the payload.
27	NFI		Null frame indicator. Is set to 1 after storage of the first received data frame.
		0	Up to now no data frame has been stored into the respective message buffer.
		1	At least one data frame has been stored into the respective message buffer.
26	SYN		Sync frame indicator. A sync frame is marked by the sync frame indicator.
		0	The received frame is not a sync frame.
		1	The received frame is a sync frame.
25	SFI		Startup frame indicator. A startup frame is marked by the startup frame indicator.
		0	The received frame is not a startup frame.
		1	The received frame is a startup frame.
24	RCI		Received on channel indicator. Indicates the channel from which the received data frame was taken to update the respective receive buffer.
		0	Frame is received on channel B.
		1	Frame is received on channel A.
23-22	Reserved	0	Reads return 0. Writes have no effect.
21-16	RCC		Receive cycle count. Cycle counter value updated from received frame.
15-11	Reserved	0	Reads return 0. Writes have no effect.
10-0	DP		Data pointer. Pointer to the first 32-bit word of the data section of the addressed message buffer in the message RAM.

NOTE: Header 3 is updated from data frames only.

26.3.2.9.5 Message Buffer Status Register (MBS)

The message buffer status is updated by the communication controller with respect to the assigned channel(s) latest at the end of the slot following the slot assigned to the message buffer. The flags are updated only when the communication controller is in NORMAL_ACTIVE or NORMAL_PASSIVE state. If only one channel (A or B) is assigned to a message buffer, the channel-specific status flags of the other channel are written to 0. If both channels are assigned to a message buffer, the channel-specific status flags of both channels are updated. The message buffer status is updated only when the slot counter reached the configured frame ID and when the cycle counter filter matched. When the Host updates a message buffer via Input Buffer, all MBS flags are reset to 0 independent of which IBCM bits are set or not.

Whenever the Message Handler changes one of the flags VFRA, VFRB, SEOA, SEOB, CEOA, CEOB, SVOA, SVOB, TCIA, TCIB, ESA, ESB, MLST, FTA, FTB the respective message buffer's MBC flag in registers MBSC1/2/3/4 is set.

Figure 26-190 and Table 26-163 illustrate this register.

Figure 26-190. Message Buffer Status Register (MBS) [offset_CC = 70Ch]

31	30	29	28	27	26	25	24	23	22	21					16
Reserved		RESS	PPIS	NFIS	SYNS	SFIS	RCIS	Reserved		CCS					
R-0		R-0	R-0	R-0	R-0	R-0	R-0	R-0		R-0					

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FTB	FTA	Rsvd	MLST	ESB	ESA	TCIB	TCIA	SVOB	SVOA	CEOB	CEOA	SEOB	SEOA	VFRB	VFRA
R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0

LEGEND: R = Read only; -*n* = value after reset

Table 26-163. Message Buffer Status Register (MBS) Field Descriptions

Bit	Field	Value	Description
31-30	Reserved	0	Reads return 0. Writes have no effect.
29	RESS	0-1	Reserved bit status. Reflects the state of the received reserved bit. The reserved bit is transmitted as 0.
28	PPIS		Payload preamble indicator status. The payload preamble indicator defines whether a network management vector or message ID is contained within the payload segment of the received frame.
		0	The payload segment of the received frame does not contain a network management vector or a message ID.
		1	Static segment: Network management vector at the beginning of the payload Dynamic segment: Message ID at the beginning of the payload.
27	NFIS		Null frame indicator status. If set to 0, the payload segment of the received frame contains no usable data.
		0	Received frame is a null frame.
		1	Received frame is not a null frame.
26	SYNS		Sync frame indicator status. A sync frame is marked by the sync frame indicator.
		0	No sync frame is received.
		1	The received frame is a sync frame.
25	SFIS		Startup frame indicator status. A startup frame is marked by the startup frame indicator.
		0	No startup frame is received.
		1	The received frame is a startup frame.
24	RCIS		Received on channel indicator status. Indicates the channel on which the frame was received.
		0	Frame is received on channel B.
		1	Frame is received on channel A.
23-22	Reserved	0	Reads return 0. Writes have no effect.
21-16	CCS	0-3Fh	Cycle count status. Actual cycle count when status was updated.

Table 26-163. Message Buffer Status Register (MBS) Field Descriptions (continued)

Bit	Field	Value	Description
15	MTB		Frame transmitted on channel B. Indicates that this node has transmitted a data frame in the configured slot on channel B.
		0	No data frame is transmitted on channel B.
		1	Data frame is transmitted on channel B.
14	MTA		Frame transmitted on channel A. Indicates that this node has transmitted a data frame in the configured slot on channel A.
		0	No data frame is transmitted on channel A.
		1	Data frame is transmitted on channel A.
13	Reserved	0	Reads return 0. Writes have no effect.
12	MLST		Message lost. The flag is set in case the Host did not read the message before the message buffer was updated from a received data frame. Not affected by reception of null frames except for message buffers belonging to the receive FIFO. The flag is reset by a host write to the message buffer via IBF or when a new message is stored into the message buffer after the message buffers ND flag was reset by reading out the message buffer via OBF.
		0	No message is lost.
		1	Unprocessed message was overwritten.
11	ESB		Empty slot channel B. In an empty slot there is no activity on the bus. The condition is checked in static and dynamic slots.
		0	Bus activity is detected in the configured slot on channel B.
		1	No bus activity is detected in the configured slot on channel B.
10	ESA		Empty slot channel A. In an empty slot there is no activity on the bus. The condition is checked in static and dynamic slots.
		0	Bus activity is detected in the configured slot on channel A.
		1	No bus activity is detected in the configured slot on channel A.
9	TCIB		Transmission conflict indication channel B. A transmission conflict indication is set if a transmission conflict has occurred on channel B.
		0	No transmission conflict occurred on channel B.
		1	Transmission conflict occurred on channel B.
8	TCIA		Transmission conflict indication channel A. A transmission conflict indication is set if a transmission conflict has occurred on channel A.
		0	No transmission conflict occurred on channel A.
		1	Transmission conflict occurred on channel A.
7	SVOB		Slot boundary violation observed on channel B. A slot boundary violation (channel active at the start or at the end of the assigned slot) was observed on channel B.
		0	No slot boundary violation is observed on channel B.
		1	Slot boundary violation is observed on channel B.
6	SVOA		Slot boundary violation observed on channel A. A slot boundary violation (channel active at the start or at the end of the assigned slot) was observed on channel A.
		0	No slot boundary violation is observed on channel A.
		1	Slot boundary violation is observed on channel A.
5	CEOB		Content error observed on channel B. A content error was observed in the configured slot on channel B.
		0	No content error is observed on channel B.
		1	Content error is observed on channel B.
4	CEOA		Content error observed on channel A. A content error was observed in the configured slot on channel A.
		0	No content error is observed on channel A.
		1	Content error is observed on channel A.
3	SEOB		Syntax error observed on channel B. A syntax error was observed in the assigned slot on channel B.
		0	No syntax error is observed on channel B.
		1	Syntax error is observed on channel B.

Table 26-163. Message Buffer Status Register (MBS) Field Descriptions (continued)

Bit	Field	Value	Description
2	SEOA		Syntax error observed on channel A. A syntax error was observed in the assigned slot on channel A.
		0	No syntax error is observed on channel A.
		1	Syntax error is observed on channel A.
1	VFRB		Valid frame received on channel B. A valid frame indication is set if a valid frame was received on channel B.
		0	No valid frame is received on channel B.
		1	Valid frame is received on channel B.
0	VFRA		Valid frame received on channel A. A valid frame indication is set if a valid frame was received on channel A.
		0	No valid frame is received on channel A.
		1	Valid frame is received on channel A.

NOTE: The status bits RESS, PPPIS, NFIS, FYNS, SFIS and RCIS are updated from both valid data and null frames. If no valid frame was received, the previous value is maintained.

The FlexRay protocol specification requires that FTA, and FTB can only be reset by the CPU. Therefore the Cycle Count Status CCS for these bits is only valid for the cycle where the bits are set to 1.

26.3.2.9.6 Output Buffer Command Mask Register (OBCM)

Configures how the Output Buffer is updated from the message buffer in the Message RAM selected by bits OBRS of the output buffer command request register. Mask bits RDSS and RHSS are copied to the register internal storage when a Message RAM transfer is requested by OBCR.REQ. When OBF host and OBF shadow are swapped, also mask bits RDSH and RHSH are swapped with bits RDSS and RHSS to keep them attached to the respective output buffer transfer.

Figure 26-191 and Table 26-164 illustrate this register.

Figure 26-191. Output Buffer Command Mask Register (OBCM) [offset_CC = 700h]

31-18	17	16
Reserved	RDSH	RHSH
R-0	R-0	R-0

15-2	1	0
Reserved	RDSS	RHSS
R-0	R/W-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 26-164. Output Buffer Command Mask Register (OBCM) Field Descriptions

Bit	Field	Value	Description
31-18	Reserved	0	Reads return 0. Writes have no effect.
17	RDSH		Read data section host.
		0	Data section is not read.
		1	Data section is selected for transfer from message RAM to output buffer.
16	RHSH		Read header section host.
		0	Header section is not read.
		1	Header section is selected for transfer from message RAM to output buffer.
15-2	Reserved	0	Reads return 0. Writes have no effect.
1	RDSS		Read Data Section shadow.
		0	Data section is not read.
		1	Data section is selected for transfer from message RAM to output buffer.
0	RHSS		Read header section shadow.
		0	Header section is not read.
		1	Header section is selected for transfer from message RAM to output buffer.

NOTE: After the transfer of the header section from the message RAM to OBF shadow has completed, the message buffer status Changed flag MBS of the selected message buffer in the message buffer Changed 1,2,3,4 registers is cleared. After the transfer of the data section from the message RAM to OBF shadow has completed, the New Data flag ND of the selected message buffer in the New Data 1,2,3,4 registers is cleared.

26.3.2.9.7 Output Buffer Command Request Register (OBCR)

After setting bit REQ to 1 while OBSYS is 0, OBSYS is automatically set to 1, OBRS(6-0) is copied to the register internal storage, mask bits OBCM.RDSS and OBCM.RHSS are copied to register OBCM internal storage, and the transfer of the message buffer selected by OBRS(6-0) from the Message RAM to OBF Shadow is started. When the transfer between the Message RAM and OBF shadow has completed, this is signaled by setting OBSYS back to 0.

By setting bit VIEW to 1 while OBSYS is 0, OBF Host and OBF shadow are swapped. Additionally mask bits OBCM.RDSH and OBCM.RHSH are swapped with the register OBCM internal storage to keep them attached to the respective output buffer transfer. OBRH(6-0) signals the number of the message buffer currently accessible by the Host.

If bits REQ and VIEW are set to 1 with the same write access while OBSYS is 0, OBSYS is automatically set to 1 and OBF shadow and OBF host are swapped. Additionally mask bits OBCM.RDSH and OBCM.RHSH are swapped with the registers internal storage to keep them attached to the respective output buffer transfer. Afterwards OBRS(6-0) is copied to the register

internal storage, and the transfer of the selected message buffer from the Message RAM to OBF shadow is started. While the transfer is ongoing the Host can read the message buffer transferred by the previous transfer from OBF host. When the current transfer between Message RAM and OBF shadow has completed, this is signaled by setting OBSYS back to 0.

Any write access to OBCR(15-8) while OBSYS is set will cause the error flag IOBA in the Error Interrupt Register to be set.

In this case the output buffer will not be changed.

Figure 26-192 and Table 26-165 illustrate this register.

Figure 26-192. Output Buffer Command Mask Register (OBCR) [offset_CC = 714h]

31–23	22–16
Reserved	OBRH
R-0	R/W-0

15	14–10	9	8	7	6–0
OBSYS	Reserved	REQ	VIEW	Rsvd	OBRS
R-0	R-0	R/W-0	R/W-0	R-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset; *These bits can be updated in DEFAULT_CONFIG or CONFIG state only

Table 26-165. Output Buffer Command Mask Register (OBCR) Field Descriptions

Bit	Field	Value	Description
31-23	Reserved	0	Reads return 0. Writes have no effect.
22-16	OBRH	0-7Fh	Output buffer request host. Number of message buffer currently accessible by the Host via RDHS[1..3], MBS, and RDDS[1..64]. By writing VIEW to 1 OBF Shadow and OBF host are swapped and the transferred message buffer is accessible by the host.
15	OBSYS		Output buffer shadow busy. Set to 1 after setting bit REQ. When the transfer between the message RAM and OBF shadow has completed, OBSYS is set back to 0.
		0	No transfer is in progress.
		1	Transfer between message RAM and OBF shadow is in progress.
14-10	Reserved	0	Reads return 0. Writes have no effect.
9	REQ		Request message RAM Transfer. Requests transfer of message buffer addressed by OBRS from message RAM to OBF shadow. Only writable while OBSYS = 0.
		0	No request.
		1	Transfer to OBF shadow is requested.
8	VIEW		View shadow buffer. Toggles between OBF shadow and OBF host. Only writable while OBSYS = 0.
		0	No action.
		1	Swap OBF shadow and OBF.

Table 26-165. Output Buffer Command Mask Register (OBCR) Field Descriptions (continued)

Bit	Field	Value	Description
7	Reserved	0	Reads return 0. Writes have no effect.
6-0	OBRS	0-7Fh	Output buffer request shadow. Number of source message buffer to be transferred from the message RAM to OBF shadow. If the number of the first message buffer of the receive FIFO is written to this register, the message handler transfers the message buffer addressed by the GET Index register (GIDX) to OBF shadow.

Chapter 27
SPNU563A–March 2018

Controller Area Network (DCAN) Module

This chapter describes the controller area network (DCAN) module.

NOTE: This chapter describes a superset implementation of the DCAN module. Consult your device-specific datasheet to identify the applicability of the DMA-related features, the number of instantiations of the DCAN IP, and the number of mailboxes supported on your specific device being used.

Topic **Page**

27.1 Overview

The Controller Area Network is a high-integrity, serial, multi-master communication protocol for distributed real-time applications. This CAN module is implemented according to ISO 11898-1 and is suitable for industrial, automotive and general embedded communications.

27.1.1 Features

The DCAN module provides the following features:

Protocol

- Supports CAN protocol version 2.0 part A, B

Speed

- Bit rates up to 1 MBit/s

MailBox

- Configurable Message objects
- Individual identifier masks for each message object
- Programmable FIFO mode for message objects

High Speed MailBox Access

- DMA access to Message RAM.

Power

- Global power down and wakeup support
- Local power down and wakeup support

Debug

- Suspend mode for debug support
- Programmable loop-back modes for self-test operation
- Direct access to Message RAM in test mode
- Supports Two interrupt lines - Level 0 and Level 1

Others

- Automatic Message RAM initialization
- Automatic bus on after Bus-Off state by a programmable 32-bit timer
- CAN Rx / Tx pins configurable as general purpose IO pins
- Software module reset
- Message RAM with SECDED mechanism
- Dual clock source to reduce jitter

27.1.2 Functional Description

The CAN protocol is an ISO standard (ISO 11898) for serial data communication. This protocol uses Non-Return To Zero (NRZ) with bit-stuffing. And the communication is carried over a two-wire balanced signaling scheme.

The DCAN data communication happens through the CAN_TX and CAN_RX pins. An additional transceiver hardware is required for the connection to the physical layer (CAN bus) CAN_High and CAN_Low.

The DCAN register set can be accessed directly by the CPU. These registers are used to control and configure the CAN module and the Message RAM.

Individual CAN message objects should be configured for communication over a CAN network. The message objects and identifier masks are stored in the Message RAM.

The CAN module internally handles functions such acceptance filtering, transfer of messages from and to the Message RAM, handling of transmission requests as well as the generation of interrupts or DMA requests.

27.2 CAN Blocks

The DCAN Module, shown in Figure 27-1, comprises of the following basic blocks.

27.2.1 CAN Core

The CAN Core consists of the CAN Protocol Controller and the Rx/Tx Shift Register. It handles all ISO 11898-1 protocol functions.

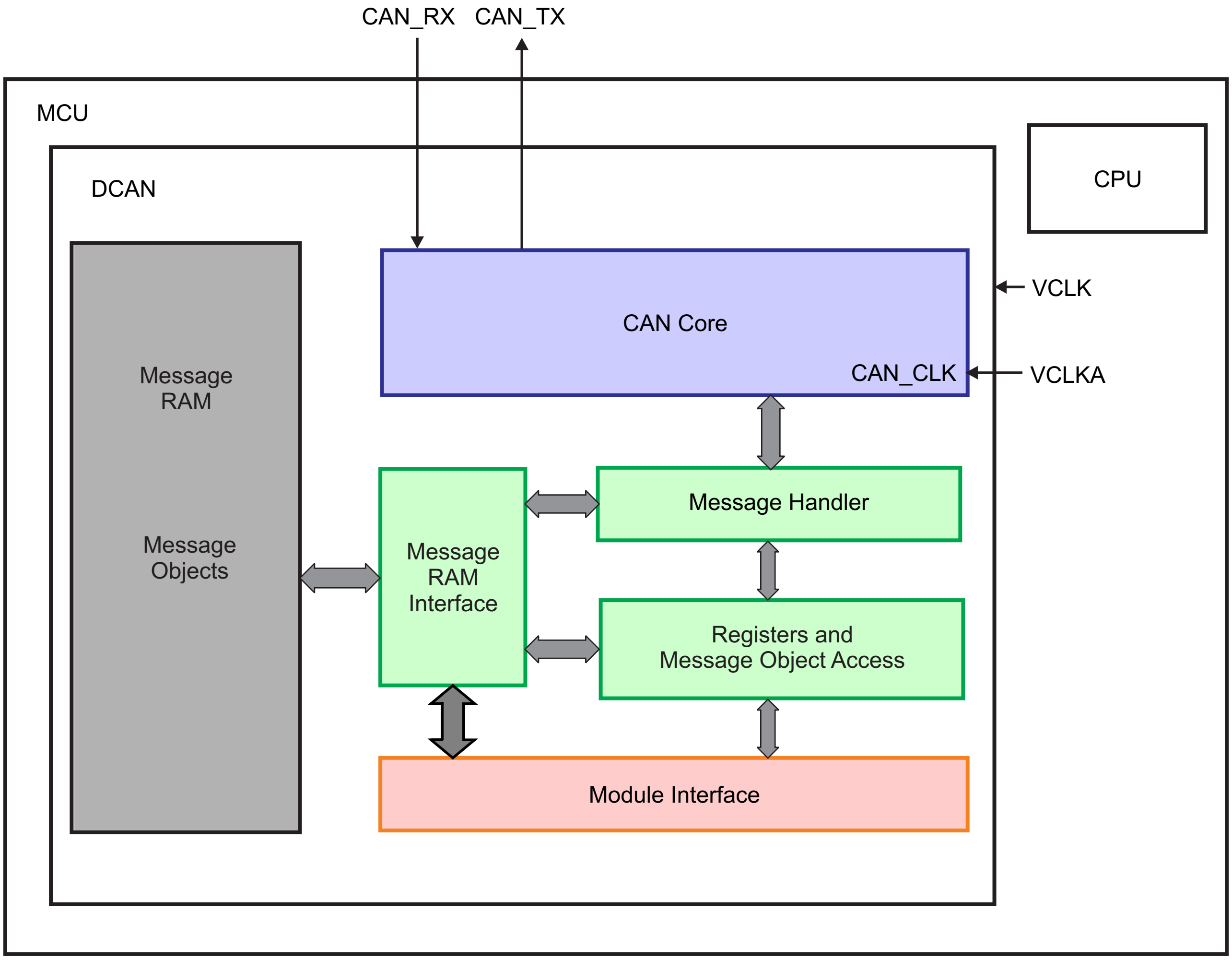
27.2.2 Message RAM

The DCAN Message RAM enables storage of CAN messages. Actual Device datasheet provides the details of the Message RAM address.

27.2.3 Message Handler

The Message Handler is a state machine that controls the data transfer between the single ported Message RAM and the CAN Core’s Rx/Tx Shift Register. It also handles acceptance filtering and the interrupt/DMA request generation as programmed in the control registers.

Figure 27-1. DCAN Block Diagram



27.2.4 Message RAM Interface

The Interface Register sets control the CPU read and write accesses to the Message RAM. There are three interface registers IF1, IF2, and IF3:

- IF1 and IF2 Interface Registers sets for read and write access.
- IF3 Interface Register set for read access only.

The Interface Registers have the same word-length as the Message RAM. Additional information can be found in Section 27.6.

27.2.5 Register and Message Object Access

During normal operation, data consistency of the message objects is guaranteed by indirectly accessing the message objects through the interface registers IF1 and IF2.

In order to be able to perform tests on the message object memory, a dedicated test mode has been implemented, that allows direct access by either the CPU or DMA. During normal operation direct access has to be avoided.

27.2.6 Dual Clock Source

Two clock domains are provided to the DCAN module:

1. VCLK - The peripheral synchronous clock domain as the general module clock source.
2. VCLKA - The peripheral asynchronous clock source domain provided to the CAN core as clock source (CAN_CLK) for generating the CAN Bit Timing.

If a frequency modulated clock output from FMPLL is used as the VCLK source, then VCLKA should be derived from an unmodulated clock source (for example, OSCIN source).

The clock source for VCLKA is selected by the Peripheral Asynchronous Clock Source Register in the system module.

Both clock domains can be derived from the same clock source (so that VCLK = VCLKA). However, if frequency modulation in the FMPLL is enabled (spread spectrum clock), then due to the high precision clocking requirements of the CAN Core, the FMPLL clock source should not be used for VCLKA. Alternatively, a separate clock without any modulation (for example, derived directly from the OSCIN clock) should be used for VCLKA.

Refer to the system module reference guide and the device datasheet for more information how to configure the relevant clock source registers in the system module.

Between the two clock domains, a synchronization mechanism is implemented in the DCAN module in order to ensure correct data transfer.

NOTE: If the dual clock functionality is used, then VCLK must always be higher or equal to CAN_CLK (derived from the asynchronous clock source), in order to achieve a stable functionality of the DCAN. Here also the frequency shift of the modulated VCLK has to be considered:

$$f_{0,\ \mathrm{VCLK}} \pm \Delta f_{\mathrm{FM,VCLK}} \geq f_{\mathrm{CANCLK}}$$

The CAN Core has to be programmed to at least 8 clock cycles per bit time. To achieve a transfer rate of 1 MBaud when using the asynchronous clock domain as the clock source for CAN_CLK, an oscillator frequency of 8MHz or higher has to be used.

27.3 CAN Bit Timing

The DCAN supports bit rates between less than 1 kBit/s and 1000 kBit/s.

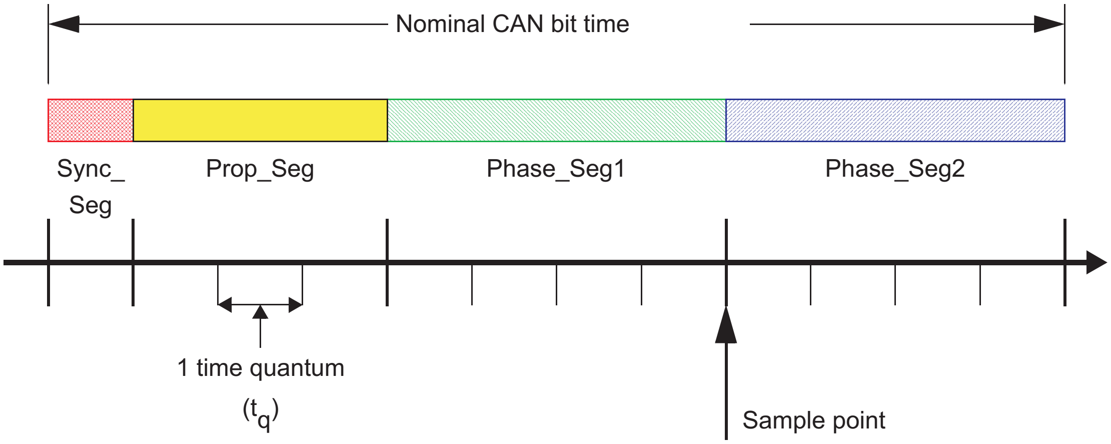
Each member of the CAN network has its own clock generator, typically derived from a crystal oscillator. The Bit timing parameters can be configured individually for each CAN node, creating a common Bit rate even though the CAN nodes' oscillator periods (f_{osc}) may be different.

27.3.1 Bit Time and Bit Rate

According to the CAN specification, the Bit time is divided into four segments (see Figure 27-2):

- Synchronization Segment (Sync_Seg)
- Propagation Time Segment (Prop_Seg)
- Phase Buffer Segment 1 (Phase_Seg1)
- Phase Buffer Segment 2 (Phase_Seg2)

Figure 27-2. Bit Timing



Each segment consists of a specific number of time quanta. The length of one time quantum, (t_q), which is the basic time unit of the bit time, is given by the CAN_CLK and the Baud Rate Prescalers (BRPE and BRP). With these two Baud Rate Prescalers combined, divider values from 1 to 1024 can be programmed:

t_q = Baud Rate Prescaler / CAN_CLK

Apart from the fixed length of the synchronization segment, these numbers are programmable.

Table 27-1 describes the minimum programmable ranges required by the CAN protocol. A given bit rate may be met by different Bit time configurations.

NOTE: For proper functionality of the CAN network, the physical delay times and the oscillator's tolerance range have to be considered.

Table 27-1. Parameters of the CAN Bit Time

Parameter	Range	Remark
Sync_Seg	1 t_q (fixed)	Synchronization of bus input to CAN_CLK
Prop_Seg	[1 … 8] t_q	Compensates for the physical delay times
Phase_Seg1	[1 … 8] t_q	May be lengthened temporarily by synchronization
Phase_Seg2	[1 … 8] t_q	May be shortened temporarily by synchronization
Synchronization Jump Width (SJW)	[1 … 4] t_q	May not be longer than either Phase Buffer Segment

27.3.1.1 Synchronization Segment

The Synchronization Segment (Sync_Seg) is the part of the bit time where edges of the CAN bus level are expected to occur. If an edge occurs outside of Sync_Seg, its distance to the Sync_Seg is called the phase error of this edge.

27.3.1.2 Propagation Time Segment

This part of the bit time is used to compensate physical delay times within the CAN network. These delay times consist of the signal propagation time on the bus and the internal delay time of the CAN nodes.

27.3.1.3 Phase Buffer Segments and Synchronization

The Phase Buffer Segments (Phase_Seg1 and Phase_Seg2) and the Synchronization Jump Width (SJW) are used to compensate for the oscillator tolerance.

The Phase Buffer Segments surround the sample point. The Phase Buffer Segments may be lengthened or shortened by synchronization.

The Synchronization Jump Width (SJW) defines how far the resynchronizing mechanism may move the sample point inside the limits defined by the Phase Buffer Segments to compensate for edge phase errors.

Synchronizations occur on edges from recessive to dominant. Their purpose is to control the distance between edges and sample points.

Edges are detected by sampling the actual bus level in each time quantum and comparing it with the bus level at the previous sample point. A synchronization may be done only if a recessive bit was sampled at the previous sample point and if the actual time quantum's bus level is dominant.

An edge is synchronous if it occurs inside of Sync_Seg, otherwise its distance to the Sync_Seg is the edge phase error, measured in time quanta. If the edge occurs before Sync_Seg, the phase error is negative, else it is positive.

27.3.1.4 Oscillator Tolerance Range

With the introduction of CAN protocol version 1.2, the option to synchronize on edges from dominant to recessive became obsolete. Only edges from recessive to dominant are considered for synchronization. The protocol update to version 2.0 (A and B) had no influence on the oscillator tolerance.

The tolerance range df for an oscillator's frequency f_{osc} around the nominal frequency f_{nom} with:

$$(1\text{-df}) \text{ x } f_{nom} \le f_{osc} \le (1\text{+df}) \text{ x } f_{nom} \quad (35)$$

depends on the proportions of Phase_Seg1, Phase_Seg2, SJW, and the bit time. The maximum tolerance df is the defined by two conditions (both shall be met):

$$\text{I:}\quad \text{df} \le \frac{\text{min(Phase\_Seg1, Phase\_Seg2)}}{[2\text{x}(13\text{xbit\_time - Phase\_Seg2})]}$$

$$\text{II:}\quad \text{df} \le \frac{\text{SJW}}{20\text{xbit\_time}} \quad (36)$$

It has to be considered that SJW may not be larger than the smaller of the Phase Buffer Segments and that the Propagation Time Segment limits that part of the bit time that may be used for the Phase Buffer Segments.

The combination Prop_Seg = 1 and Phase_Seg1 = Phase_Seg2 = SJW = 4 allows the largest possible oscillator tolerance of 1.58%. This combination with a Propagation Time Segment of only 10% of the bit time is not suitable for short bit times; it can be used for bit rates of up to 125 kBit/s (bit time = 8 μs) with a bus length of 40 m.

27.3.2 DCAN Bit Timing Registers

In the DCAN, the bit timing configuration is programmed in two register bytes, additionally a third byte for a baud rate prescaler extension of 4 bits (BREP) is provided. The sum of Prop_Seg and Phase_Seg1 (as TSEG1) is combined with Phase_Seg2 (as TSEG2) in one byte, SJW and BRP (plus BRPE in third byte) are combined in the other byte

In this bit timing register, the components TSEG1, TSEG2, SJW and BRP have to be programmed to a numerical value that is one less than its functional value; so instead of values in the range of [1… n], values in the range of [0 … n-1] are programmed. That way, SJW (functional range of [1 … 4]) is represented by only two bits.

Therefore the length of the Bit time is (programmed values) [TSEG1 + TSEG2 + 3] t_q or (functional values) [Sync_Seg + Prop_Seg + Phase_Seg1 + Phase_Seg2] t_q.

The data in the Bit Timing Register (BTR) is the configuration input of the CAN protocol controller. The Baud Rate Prescaler (configured by BRPE/BRP) defines the length of the time quantum (the basic time unit of the bit time); the Bit Timing Logic (configured by TSEG1, TSEG2, and SJW) defines the number of time quanta in the bit time.

27.3.2.1 Calculation of the Bit Timing Parameters

Usually, the calculation of the bit timing configuration starts with a desired bit rate or bit time. The resulting Bit time (1 / Bit rate) must be an integer multiple of the CAN clock period.

NOTE: 8 MHz is the minimum CAN clock frequency required to operate the DCAN at a bit rate of 1 MBit/s.

The bit time may consist of 8 to 25 time quanta. The length of the time quantum t_q is defined by the Baud Rate Prescaler with t_q = (Baud Rate Prescaler) / CAN_CLK. Several combinations may lead to the desired bit time, allowing iterations of the following steps.

First part of the bit time to be defined is the Prop_Seg. Its length depends on the delay times measured in the system. A maximum bus length as well as a maximum node delay has to be defined for expandible CAN bus systems. The resulting time for Prop_Seg is converted into time quanta (rounded up to the nearest integer multiple of t_q).

The Sync_Seg is 1 t_q long (fixed), leaving (bit time – Prop_Seg – 1) t_q for the two Phase Buffer Segments. If the number of remaining t_q is even, the Phase Buffer Segments have the same length, Phase_Seg2 = Phase_Seg1, else Phase_Seg2 = Phase_Seg1 + 1.

The minimum nominal length of Phase_Seg2 has to be regarded as well. Phase_Seg2 may not be shorter than any CAN controller's Information Processing Time in the network, which is device dependent and can be in the range of [0 … 2] t_q.

The length of the Synchronization Jump Width is set to its maximum value, which is the minimum of 4 and Phase_Seg1.

If more than one configurations are possible to reach a certain Bit rate, it is recommended to choose the configuration that allows the highest oscillator tolerance range.

CAN nodes with different clocks require different configurations to come to the same bit rate. The calculation of the propagation time in the CAN network, based on the nodes with the longest delay times, is done once for the whole network.

The CAN system's oscillator tolerance range is limited by the node with the lowest tolerance range.

The calculation may show that bus length or bit rate have to be decreased or that the oscillator frequencies' stability has to be increased in order to find a protocol compliant configuration of the CAN bit timing.

The resulting configuration is written into the Bit Timing Register:

Tseg2 = Phase_Seg2 - 1
Tseg1 = Phase_Seg1 + Prop_Seg - 1
SJW = SynchronizationJumpWidth - 1
BRP = Prescaler - 1

27.3.2.2 Calculation of BRP Values

If Baud and CAN_CLK(VCLK) are already known, the BRP/BRPE values need to be calculated to be programmed into the register. It is calculated using the following equation:

$$BRP = CAN\_CLK / (BAUD)(1 + TSEG1 + TSEG2) \quad (37)$$

27.3.2.3 Example for Bit Timing at High Baudrate

In this example, the frequency of CAN_CLK is 10 MHz, BRP is 0, the bit rate is 1 MBit/s.

t_q	100	ns	=	t_{CAN_CLK}	
delay of bus driver	60	ns			
delay of receiver circuit	40	ns			
delay of bus line (40m)	220	ns			
t_{Prop}	700	ns	=	INT (2 × delays + 1) = 7 • t_q	
t_{SJW}	100	ns	=	1 × t_q	
t_{TSeg1}	800	ns	=	$t_{Prop} + t_{SJW}$	
t_{TSeg2}	100	ns	=	Information Processing Time + 1 • t_q	
$t_{Sync-Seg}$	100	ns	=	1 × t_q	
bit time	1000	ns	=	$t_{Sync-Seg} + t_{TSeg1} + t_{TSeg2}$	
tolerance for CAN_CLK	1.58	%	=	$\frac{\min(TSeg1, TSeg2)}{[2 \times (13 \times bit\_time - TSeg2)]}$	(38)
			=	$\frac{0.1\,\mu s}{[2 \times (13 \times 1\mu s - 0.1\mu s)]}$	(39)
			=	0.38%	

In this example, the concatenated bit time parameters are $(1-1)_3$ & $(8-1)_4$ & $(1-1)_2$ & $(1-1)_6$, so the Bit Timing Register is programmed to 0000 0700h.

27.3.2.4 Example for Bit Timing at Low Baudrate

In this example, the frequency of CAN_CLK is 2 MHz, BRP is 1, the bit rate is 100 KBit/s.

t_q	1	µs	=	$2 \times t_{CAN\_CLK}$	
delay of bus driver	200	ns			
delay of receiver circuit	80	ns			
delay of bus line (40m)	220	ns			
t_{Prop}	1	µs	=	$1 \times t_q$	
t_{SJW}	4	µs	=	$4 \times t_q$	
t_{TSeg1}	5	µs	=	$t_{Prop} + t_{SJW}$	
t_{TSeg2}	3	µs	=	Information Processing Time + $3 \times t_q$	
$t_{Sync\text{-}Seg}$	1	µs	=	$1 \times t_q$	
bit time	9	µs	=	$t_{Sync\text{-}Seg} + t_{TSeg1} + t_{TSeg2}$	
tolerance for CAN_CLK	0.43	%	=	$\dfrac{\min(TSeg1, TSeg2)}{[2 \times (13 \times bit\_time - TSeg2)]}$	(40)
			=	$\dfrac{3µs}{[2 \times (13 \times 9µs - 3µs)]}$	(41)
			=	1.32%	

In this example, the concatenated bit time parameters are $(3\text{-}1)_3$ & $(5\text{-}1)_4$ & $(4\text{-}1)_2$ & $(2\text{-}1)_6$, so the Bit Timing Register is programmed to 0000 24C1h.

27.4 CAN Module Configuration

After a hardware reset all CAN protocol functions are disabled.The CAN module must be initialized and configured before it can participate on the CAN bus.

27.4.1 DCAN RAM Initialization Through Hardware

To start with a clean DCAN RAM ,the complete DCAN RAM has to be initialized with zeros and the ECC bits set accordingly by configuring the following registers in the system module:

1. Memory Hardware Initialization Global Control Register (MINITGCR)
2. Memory Initialization Enable Register (MSINENA)

For more details on RAM hardware initialization support, refer to the system module reference guide.

27.4.2 CAN Module Initialization

To initialize the CAN Controller, you have to set up the CAN Bit timing and those message objects that have to be used for CAN communication. Message objects that are not needed, can be deactivated.

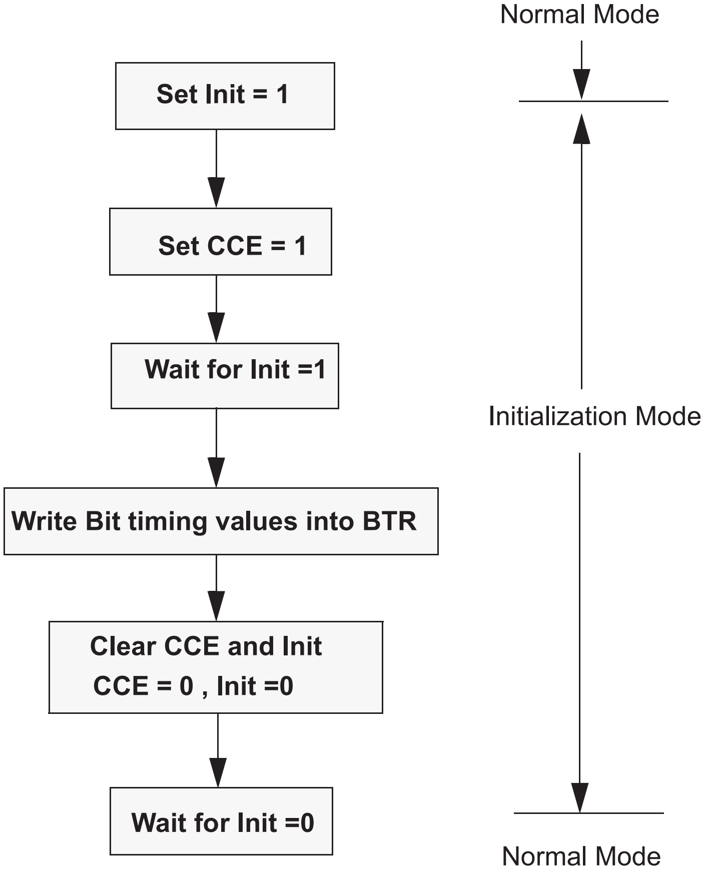
So the two critical steps are:

1. Configuration of CAN Bit Timings
2. Configuration of Message Objects

27.4.2.1 Software Configuration of CAN Bit Timings

This step involves configuring the CAN baud rate register with the calculated CAN bit timing value. The calculation procedure of CAN bit timing values for BTR register are mentioned in Section 27.3. Refer to Figure 27-3 for CAN bit timing software configuration flow.

Figure 27-3. CAN Bit-timing Configuration



Step 1: Enter “initialization mode” by setting the Init (Initialization) bit in the CAN Control Register.

While the Init bit is set, the message transfer from and to the CAN bus is stopped, and the status of the CAN_TX output is recessive (high).

The CAN error counters are not updated. Setting the Init bit does not change any other configuration register.

Also note that the CAN module is also in initialization mode on hardware reset and during Bus-Off.

Step 2: Set the CCE (Configure Change Enable) bit in the CAN Control Register.

The access to the Bit Timing Register for the configuration of the Bit timing is enabled when both Init and CCE bits in the CAN Control Register are set.

Step 3: Wait for the Init bit to get set. This would make sure that the module has entered “Initialization mode”.

Step 4: Write the Bit-Timing values into the Bit-Timing Register (BTR).

Refer to Section 27.3.2.1 for BTR value calculation for a given bit-timing.

Step 5: Clear the CCE bit followed by Init bit.

Step 6: Wait for the Init bit to clear. This would make sure that the module has come out of “initialization mode”.

After step 6 (Init bit cleared), the module will attempt a synchronization on the CAN bus, provided that the BTR settings are meeting the CAN bus parameters.

NOTE: The module would not come out of the “initialization mode” if any incorrect BTR values are written in step 4.

27.4.2.2 Configuration of Message Objects

The whole Message RAM should be configured before putting the CAN into operation. All the message objects are deactivated by default. You should configure the message object that are to be used to a particular identifier. you can change the configuration of any message object or deactivate it when required.

The message objects can be configured only through the Interface registers (IFx) and the CPU does not have direct access to the message object (Message RAM) when DCAN is in operation.

To configure the message objects, you must know about:

1. The message object structure (Section 27.5)
2. The interface register set (IFx) (Section 27.6)

NOTE: The message objects initialization is independent of the bit-timing configuration procedure.

27.5 Message RAM

The DCAN Message RAM contains message objects and ECC bits for the message objects.

27.5.1 Structure of Message Objects

Figure 27-4 shows the structure of a message object.

The grayed fields are those parts of the message object that are represented in dedicated registers. For example, the transmit request flags of all message objects are represented in centralized transmit request registers.

Figure 27-4. Structure of a Message Object

Message Object												
UMask	Msk[28:0]	MXtd	MDir	EoB	unused	NewDat	MsgLst	RxIE	TxIE	IntPnd	RmtEn	TxRqst
MsgVal	ID[28:0]	Xtd	Dir	DLC[3:0]	Data 0	Data 1	Data 2	Data 3	Data 4	Data 5	Data 6	Data 7

Table 27-2. Message Object Field Descriptions

Name	Value	Description
MsgVal		Message valid
	0	The message object is ignored by the Message Handler.
	1	The message object is to be used by the Message Handler.
		Note: The CPU should reset the MsgVal bit of all unused Messages Objects during the initialization before it resets bit Init in the CAN Control Register. MsgVal must also be reset if the messages object is no longer used in operation. For reconfiguration of message objects during normal operation see Section 27.7.6 and Section 27.7.7.
UMask		Use Acceptance Mask
	0	Mask bits (Msk[28:0], MXtd and MDir) are ignored and not used for acceptance filtering.
	1	Mask bits are used for acceptance filtering.
		Note: If the UMask bit is set to 1, the message object's mask bits have to be programmed during initialization of the message object before MsgVal is set to 1.
ID[28:0]		Message Identifier
	ID[28:0]	29-bit ("extended") identifier bits
	ID[28:18]	11-bit ("standard") identifier bits
Msk[28:0]		Identifier Mask
	0	The corresponding bit in the message identifier is not used for acceptance filtering (don't care).
	1	The corresponding bit in the message identifier is used for acceptance filtering.
Xtd		Extended Identifier
	0	The 11-bit ("standard") identifier will be used for this message object.
	1	The 29-bit ("extended") identifier will be used for this message object.
MXtd		Mask Extended Identifier
	0	The extended identifier bit (IDE) has no effect on the acceptance filtering.
	1	The extended identifier bit (IDE) is used for acceptance filtering.
		Note: When 11-bit ("standard") identifiers are used for a message object, the identifiers of received data frames are written into bits ID[28:18]. For acceptance filtering, only these bits together with mask bits Msk[28:18] are considered.
Dir		Message Direction
	0	Direction = receive: On TxRqst, a remote frame with the identifier of this message object is transmitted. On reception of a data frame with matching identifier, the message is stored in this message object.
	1	Direction = transmit: On TxRqst, a data frame is transmitted. On reception of a remote frame with matching identifier, the TxRqst bit of this message object is set (if RmtEn = 1).
MDir		Mask Message Direction
	0	The message direction bit (Dir) has no effect on the acceptance filtering.
	1	The message direction bit (Dir) is used for acceptance filtering.

Table 27-2. Message Object Field Descriptions (continued)

Name	Value	Description
EOB		End of Block
	0	The message object is part of a FIFO Buffer block and is not the last message object of this FIFO Buffer block.
	1	The message object is a single message object or the last message object in a FIFO Buffer block.
		Note: This bit is used to concatenate multiple message objects to build a FIFO Buffer. For single message objects (not belonging to a FIFO Buffer), this bit must always be set to 1.
NewDat		New Data
	0	No new data has been written into the data bytes of this message object by the Message Handler since the last time when this flag was cleared by the CPU.
	1	The Message Handler or the CPU has written new data into the data bytes of this message object.
MsgLst		Message Lost (only valid for message objects with direction = receive)
	0	No message was lost since the last time when this bit was reset by the CPU.
	1	The Message Handler stored a new message into this message object when NewDat was still set, so the previous message has been overwritten.
RxIE		Receive Interrupt Enable
	0	IntPnd will not be triggered after the successful reception of a frame.
	1	IntPnd will be triggered after the successful reception of a frame.
TxIE		Transmit Interrupt Enable
	0	IntPnd will not be triggered after the successful transmission of a frame.
	1	IntPnd will be triggered after the successful transmission of a frame.
IntPnd		Interrupt Pending
	0	This message object is not the source of an interrupt.
	1	This message object is the source of an interrupt. The Interrupt Identifier in the Interrupt Register will point to this message object if there is no other interrupt source with higher priority.
RmtEn		Remote Enable
	0	At the reception of a Remote Frame, TxRqst is not changed.
	1	At the reception of a Remote Frame, TxRqst is set.
TxRqst		Transmit Request
	0	This message object is not waiting for a transmission.
	1	The transmission of this message object is requested and is not yet done.
DLC[3:0]		Data Length Code
	0-8	Data Frame has 0-8 data bytes.
	9-15	Data Frame has 8 data bytes.
		Note: The Data Length Code of a message object must be defined to the same value as in the corresponding objects with the same identifier at other nodes. When the Message Handler stores a data frame, it will write the DLC to the value given by the received message.
Data 0		1st data byte of a CAN Data Frame
Data 1		2nd data byte of a CAN Data Frame
Data 2		3rd data byte of a CAN Data Frame
Data 3		4th data byte of a CAN Data Frame
Data 4		5th data byte of a CAN Data Frame
Data 5		6th data byte of a CAN Data Frame
Data 6		7th data byte of a CAN Data Frame
Data 7		8th data byte of a CAN Data Frame
		Note: Byte Data 0 is the first data byte shifted into the shift register of the CAN Core during a reception, byte Data 7 is the last. When the Message Handler stores a data frame, it will write all the eight data bytes into a message object. If the Data Length Code is less than 8, the remaining bytes of the message object may be overwritten by undefined values.

27.5.2 Addressing Message Objects in RAM

The starting location of a particular message object in RAM is:

Message RAM base address + (message object number) × 0x20.

This means that Message Object 1 starts at offset 0x0020; Message Object 2 starts at offset 0x0040, and so on.

NOTE: 0 is not a valid message object number. At address 0x0000, message object number 64 is located. Writing to the address of an unimplemented message object may overwrite an implemented message object.

The base address for DCAN1 RAM is FF1E 0000h, DCAN2 RAM is FF1C 0000h, DCAN3 RAM is FF1A 0000h, and DCAN4 RAM base address is FF18 0000h.

Message Object number 1 has the highest priority.

Table 27-3. Message RAM Addressing in Debug/Suspend and RDA Mode

Message Object Number	Base Address Offset	Word Number	Debug/Suspend mode, see Section 27.5.3	RDA mode, see Section 27.5.4
1	0x0020	1	Reserved	Data Bytes 4-7
	0x0024	2	MXtd, MDir, Mask	Data Bytes 0-3
	0x0028	3	Xtd, Dir, ID	ID[27:0], DLC
	0x002C	4	Ctrl	Mask, Xtd, Dir, ID[28]
	0x0030	5	Data Bytes 3-0	Reserved, Ctrl, MXtd, MDir
	0x0034	6	Data Bytes 7-4	--
:	:	:	:	:
31	0x03E0	1	Reserved	Data Bytes 4-7
	0x03E4	2	MXtd, MDir, Mask	Data Bytes 0-3
	0x03E8	3	Xtd, Dir, ID	ID[27]:0, DLC
	0x03EC	4	Ctrl	Mask, Xtd, Dir, ID[28]
	0x03F0	5	Data Bytes 3-0	Reserved, Ctrl, MXtd, MDir
	0x03F4	6	Data Bytes 7-4	--
:	:	:	:	:
63	0x07E0	1	Reserved	Data Bytes 4-7
	0x07E4	2	MXtd, MDir, Mask	Data Bytes 0-3
	0x07E8	3	Xtd, Dir, ID	ID[27:0], DLC
	0x07EC	4	Ctrl	Mask, Xtd, Dir, ID[28]
	0x07F0	5	Data Bytes 3-0	Reserved, Ctrl, MXtd, MDir
	0x07F4	6	Data Bytes 7-4	--
64	0x0000	1	Reserved	Data Bytes 4-7
	0x0004	2	MXtd, MDir, Mask	Data Bytes 0-3
	0x0008	3	Xtd, Dir, ID	ID[27]:0, DLC
	0x000C	4	Ctrl	Mask, Xtd, Dir, ID[28]
	0x0010	5	Data Bytes 3-0	Reserved, Ctrl, MXtd, MDir
	0x0014	6	Data Bytes 7-4	--

27.5.3 Message RAM Representation in Debug/Suspend Mode

In Debug/Suspend mode, the Message RAM will be memory mapped. This allows the external debug unit to access the Message RAM.

NOTE: During Debug/Suspend Mode, the Message RAM cannot be accessed via the IFx register sets.

Figure 27-5. Message RAM Representation in Debug/Suspend Mode

Bit	31 15	30 14	29 13	28 12	27 11	26 10	25 9	24 8	23 7	22 6	21 5	20 4	19 3	18 2	17 1	16 0
MsgAddr + 0x00	Reserved															
	Reserved											Reserved				
MsgAddr + 0x04	MXtd	MDir	Rsvd	Msk[28:16]												
	Msk[15:0]															
MsgAddr + 0x08	Rsvd	Xtd	Dir	ID[28:16]												
	ID[15:0]															
MsgAddr + 0x0C	Reserved															
	Rsvd	MsgLst	Rsvd	UMask	TxIE	RxIE	RmtEn	Rsvd	EOB	Reserved			DLC[3:0]			
MsgAddr + 0x10	Data 3								Data 2							
	Data 1								Data 0							
MsgAddr + 0x14	Data 7								Data 6							
	Data 5								Data 4							

27.5.4 Message RAM Representation in Direct Access Mode

When the RDA bit in Test Register is set while the DCAN module is in Test Mode (Test bit in CAN control register is set), the CPU has direct access to the Message RAM. Due to the 32-bit bus structure, the RAM is split into word lines to support this feature. The CPU has access to one word line at a time only.

In RAM Direct Access mode, the RAM is represented by a continuous memory space within the address frame of the DCAN module, starting at the Message RAM base address.

NOTE: During Direct Access Mode, the Message RAM cannot be accessed via the IFx register sets. Before entering RDA mode, it must be ensured that the Init bit is set to avoid any conflicts with the message handler accessing the message RAM.

Any read or write to the RAM addresses for RamDirectAccess during normal operation mode (TestMode bit or RDA bit is not set) will be ignored.

Writes to Reserved bits have no effect.

Figure 27-6. Message RAM Representation in RAM Direct Access Mode

Bit	31 15	30 14	29 13	28 12	27 11	26 10	25 9	24 8	23 7	22 6	21 5	20 4	19 3	18 2	17 1	16 0
MsgAddr + 0x00	Data 4								Data 5							
	Data 6								Data 7							
MsgAddr + 0x04	Data 0								Data 1							
	Data 2								Data 3							
MsgAddr + 0x08	ID[27:12]															
	ID[11:0]												DLC[3:0]			
MsgAddr + 0x0C	Msk[28:13]															
	Msk[12:0]													Xtd	Dir	ID[28]
MsgAddr + 0x10	Reserved															
	Reserved				Reserved				MsgLst	UMask	TxIE	RxIE	RmtEn	EOB	MXtd	MDir

27.5.5 ECC RAM

On devices with SECDED implementation for the message RAM, the ECC bits are stored in a dedicated ECC RAM area that is memory-mapped as follows: The location of the ECC bits for a particular message object in RAM is: Message RAM base address + 0x1000 + (message object number) * 0x20.

NOTE: A 0 is not a valid message object number. At address 0x1000, the ECC bits of the last implemented message object are located.

As shown in Figure 27-7, the ECC bits for the last implemented Message Object (here: 128) are located at offset 0x1000; the ECC bits for Message Object 1 are located at offset 0x1020, and the ECC bits for Message Object 127 are located at offset 0x1FE0. The ECC RAM is only memory mapped if SECDED diagnostic mode is enabled.

Figure 27-7. ECC RAM Representation

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Msg RAM base + 0x1000	Reserved															
	Reserved							ECC[8:0] last implemented Message Object (here: 128)								
Msg RAM base + 0x1020	Reserved															
	Reserved							ECC[8:0] Message Object 1								
:																
Msg RAM base + 0x1FE0	Reserved															
	Reserved							ECC[8:0] Message Object 127								

27.6 Message Interface Register Sets

Accesses to the Message RAM are performed via the Interface Register sets:

- Interface Register 1 and 2 (IF1 and IF2)
- Interface Register 3 (IF3)

The IF3 register set can be configured to automatically receive control and user data from the Message RAM when a message object has been updated after reception of a CAN message. The CPU does not need to initiate the transfer from Message RAM to IF3 register set.

The Message Handler avoids potential conflicts between concurrent accesses to Message RAM and CAN frame reception/transmission.

There are two modes where the Message RAM can be directly accessed by the CPU:

1. In Debug/Suspend mode (see Section 27.5.3)
2. In RAM Direct Access (RDA) mode (see Section 27.5.4)

For the Message RAM Base address, refer to the device datasheet.

A complete message object (see Section 27.5.1) or parts of the message object may be transferred between the Message RAM and the IF1/IF2 Register set (see Section 27.17.24) in one single transfer.

27.6.1 Message Interface Register Sets 1 and 2

The Interface Register sets IF1 and IF2 provide indirect read/write access from the CPU to the Message RAM. The IF1 and IF2 register sets can buffer control and user data to be transferred to and from the message objects.

Table 27-4. Message Interface Register Sets 1 and 2

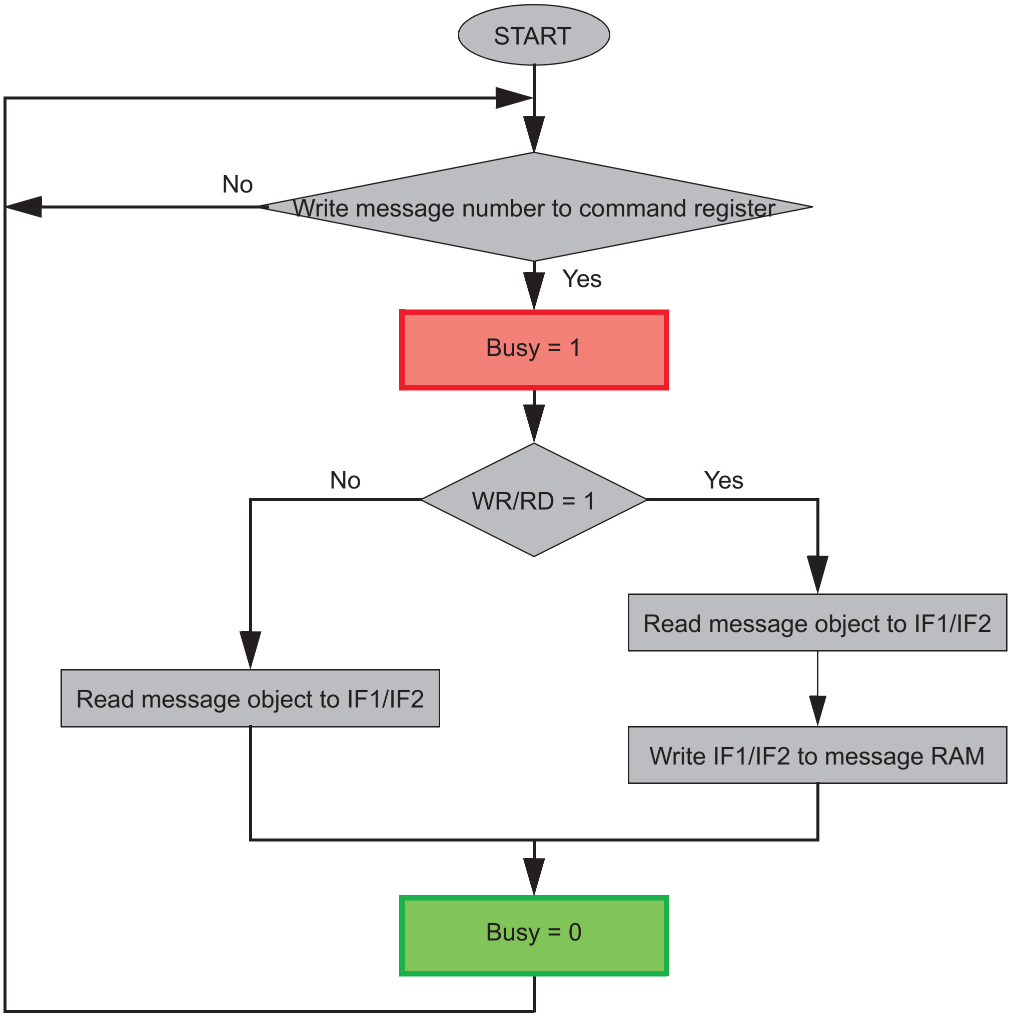
Address	IF1 Register Set		Address	IF2 Register Set	
[CAN Base +]	31 16	15 0	[CAN Base +]	31 16	15 0
0x100	IF1 Command Mask	IF1 Command Request	0x120	IF2 Command Mask	IF2 Command Request
0x104	IF1 Mask 2	IF1 Mask 1	0x124	IF2 Mask 2	IF2 Mask 1
0x108	IF1 Arbitration 2	IF1 Arbitration 1	0x128	IF2 Arbitration 2	IF2 Arbitration 1
0x10C	Rsvd	IF1 Message Control	0x12C	Rsvd	IF2 Message Control
0x110	IF1 Data A 2	IF1 Data A 1	0x130	IF2 Data A 2	IF2 Data A 1
0x114	IF1 Data B 2	IF1 Data B 1	0x134	IF2 Data B 2	IF2 Data B 1

27.6.2 Using Message Interface Register Sets 1 and 2

The Command Register addresses the desired message object in the Message RAM and specifies whether a complete message object or only parts should be transferred. The data transfer is initiated by writing the message number to the bits [7:0] of the Command Register.

When the CPU initiates a data transfer between the IF1/IF2 Registers and Message RAM, the Message Handler sets the Busy bit in the respective Command Register to ‘1’. After the transfer has completed, the Busy bit is set back to ‘0’ (see Figure 27-8).

Figure 27-8. Data Transfer Between IF1 / IF2 Registers and Message RAM


START
No
Write message number to command register
Yes
Busy = 1
No
WR/RD = 1
Yes
Read message object to IF1/IF2
Read message object to IF1/IF2
Write IF1/IF2 to message RAM
Busy = 0

27.6.3 Message Interface Register 3

The IF3 register set can automatically be updated with received message objects without the need to initiate the transfer from Message RAM by CPU. The intention of this feature of IF3 is to provide an interface for the DMA to read packets efficiently.

Table 27-5. Message Interface Register 3

Address	**IF3 Register Set**	
[CAN Base +]	**31 16**	**15 0**
0x140	reserved	IF3 Observation
0x144	IF3 Mask 2	IF3 Mask 1
0x148	IF3 Arbitration 2	IF3 Arbitration 1
0x14C	reserved	IF3 Message Control
0x150	IF3 Data A 2	IF3 Data A 1
0x154	IF3 Data B 2	IF3 Data B 1
:	:	:
0x160	IF3 Update Enable 2	IF3 Update Enable 1
0x164	IF3 Update Enable 4	IF3 Update Enable 3
0x168	IF3 Update Enable 6	IF3 Update Enable 5
0x16C	IF3 Update Enable 8	IF3 Update Enable 7

The automatic update functionality can be programmed for each message object (see IF3 Update Enable Register, Section 27.17.33).

All valid message objects in Message RAM that are configured for automatic update, will be checked for active NewDat flags. If such a message object is found, it will be transferred to the IF3 register (if no previous DMA transfers are ongoing), controlled by IF3 Observation register. If more than one NewDat flag is active, the message object with the lowest number has the highest priority for automatic IF3 update.

The NewDat bit in the message object will be reset by a transfer to IF3.

If DCAN internal IF3 update is complete, a DMA request is generated. The DMA request stays active until first read access to one of the IF3 registers. The DMA functionality has to be enabled by setting bit DE3 in CAN Control register. Please refer to the device datasheet to find out if this DMA source is available.

NOTE: The IF3 register set can not be used for transferring data into message objects.

27.7 Message Object Configurations

This section describes the possible message object configurations for CAN communication.

27.7.1 Configuration of a Transmit Object for Data Frames

Figure 27-9 shows how a Transmit Object can be initialized.

Figure 27-9. Initialization of a Transmit Object

MsgVal	Arb	Data	Mask	EoB	Dir	NewDat	MsgLst	RxIE	TxIE	IntPnd	RmtEn	TxRqst
1	appl.	appl.	appl.	1	1	0	0	0	appl.	0	appl.	0

The Arbitration bits (ID[28:0] and Xtd bit) are given by the application. They define the identifier and type of the outgoing message. If an 11-bit Identifier (Standard Frame) is used (Xtd = 0), it is programmed to ID[28:18]. In this case, ID[17:0] can be ignored.

The Data Registers (DLC[3:0] and Data0-7) are given by the application, TxRqst and RmtEn should not be set before the data is valid.

If the TxIE bit is set, the IntPnd bit will be set after a successful transmission of the message object.

If the RmtEn bit is set, a matching received Remote Frame will cause the TxRqst bit to be set; the Remote Frame will autonomously be answered by a Data Frame.

The Mask bits (Msk[28:0], UMask, MXtd, and MDir bits) may be used (UMask = 1) to allow groups of Remote Frames with similar identifiers to set the TxRqst bit. The Dir bit should not be masked. For details, see Section 27.8.8. Identifier masking must be disabled (UMask = 0) if no Remote Frames are allowed to set the TxRqst bit (RmtEn = 0).

27.7.2 Configuration of a Transmit Object for Remote Frames

It is not necessary to configure Transmit Objects for the transmission of Remote Frames. Setting TxRqst for a Receive Object will cause the transmission of a Remote Frame with the same identifier as the Data Frame for which this receive Object is configured.

27.7.3 Configuration of a Single Receive Object for Data Frames

Figure 27-10 shows how a Receive Object for Data Frames can be initialized.

Figure 27-10. Initialization of a Single Receive Object for Data Frames

MsgVal	Arb	Data	Mask	EoB	Dir	NewDat	MsgLst	RxIE	TxIE	IntPnd	RmtEn	TxRqst
1	appl.	appl.	appl.	1	0	0	0	appl.	0	0	0	0

The Arbitration bits (ID[28:0] and Xtd bit) are given by the application. They define the identifier and type of accepted received messages. If an 11-bit Identifier (Standard Frame) is used (Xtd = 0), it is programmed to ID[28:18]. In this case, ID[17:0] can be ignored. When a Data Frame with an 11-bit Identifier is received, ID[17:0] will be set to 0.

The Data Length Code (DLC[3:0]) is given by the application. When the Message Handler stores a Data Frame in the message object, it will store the received Data Length Code and eight data bytes. If the Data Length Code is less than 8, the remaining bytes of the message object may be overwritten by non specified values.

The Mask bits (Msk[28:0], UMask, MXtd, and MDir bits) may be used (UMask = ’1’) to allow groups of Data Frames with similar identifiers to be accepted. The Dir bit should not be masked in typical applications. If some bits of the Mask bits are set to “don’t care”, the corresponding bits of the Arbitration Register will be overwritten by the bits of the stored Data Frame.

If the RxIE bit is set, the IntPnd bit will be set when a received Data Frame is accepted and stored in the message object.

If the TxRqst bit is set, the transmission of a Remote Frame with the same identifier as actually stored in the Arbitration bits will be triggered. The content of the Arbitration bits may change if the Mask bits are used (UMask = 1 for acceptance filtering.

27.7.4 Configuration of a Single Receive Object for Remote Frames

Figure 27-11 shows how a Receive Object for Remote Frames can be initialized.

Figure 27-11. Initialization of a Single Receive Object for Remote Frames

MsgVal	Arb	Data	Mask	EoB	Dir	NewDat	MsgLst	RxIE	TxIE	IntPnd	RmtEn	TxRqst
1	appl.	appl.	appl.	1	1	0	0	appl.	0	0	0	0

Receive Objects for Remote Frames may be used to monitor Remote Frames on the CAN bus. The Remote Frame stored in the Receive Object will not trigger the transmission of a Data Frame. Receive Objects for Remote Frames may be expanded to a FIFO buffer, see Section 27.7.5.

UMask must be set to 1. The Mask bits (Msk[28:0], UMask, MXtd, and MDir bits) may be set to “must-match” or to “don’t care”, to allow groups of Remote Frames with similar identifiers to be accepted. The Dir bit should not be masked in typical applications. For details, see Section 27.8.8.

The Arbitration bits (ID[28:0] and Xtd bit) may be given by the application. They define the identifier and type of accepted received Remote Frames. If some bits of the Mask bits are set to “don’t care”, the corresponding bits of the Arbitration bits will be overwritten by the bits of the stored Remote Frame. If an 11-bit Identifier (Standard Frame) is used (Xtd = 0), it is programmed to ID[28:18]. In this case, ID[17:0] can be ignored. When a Remote Frame with an 11-bit Identifier is received, ID[17:0] will be set to 0.

The Data Length Code (DLC[3:0]) may be given by the application. When the Message Handler stores a Remote Frame in the message object, it will store the received Data Length Code. The data bytes of the message object will remain unchanged.

If the RxIE bit is set, the IntPnd bit will be set when a received Remote Frame is accepted and stored in the message object.

27.7.5 Configuration of a FIFO Buffer

With the exception of the EoB bit, the configuration of Receive Objects belonging to a FIFO Buffer is the same as the configuration of a single Receive Object.

To concatenate multiple message objects to a FIFO Buffer, the identifiers and masks (if used) of these message objects have to be programmed to matching values. Due to the implicit priority of the message objects, the message object with the lowest number will be the first message object of the FIFO Buffer. The EoB bit of all message objects of a FIFO Buffer except the last one have to be programmed to zero. The EoB bits of the last message object of a FIFO Buffer is set to one, configuring it as the end of the block.

27.7.6 Reconfiguration of Message Objects for the Reception of Frames

A message object with Dir = ‘0’ is configured for the reception of data frames, with Dir = ‘1’ AND Umask = ‘1’ AND RmtEn = ‘0’ it is configured for the reception of remote frames.

It is necessary to reset MsgVal to not valid before changing any of the following configuration and control bits: ID[28:0], Xtd, Dir, DLC[3:0], RxIE, TxIE, RmtEn, EoB, Umask, Msk[28:0], MXtd, and MDir.

These parts of a message object may be changed without clearing MsgVal: Data[7:0], TxRqst, NewDat, MsgLst, and IntPnd.

27.7.7 Reconfiguration of Message Objects for the Transmission of Frames

A message object with Dir = ‘1’ AND (Umask = ‘0’ OR RmtEn = ‘1’) is configured for the transmission of data frames.

It is necessary to reset MsgVal to not valid before changing any of the following configuration and control bits: Dir, RxIE, TxIE, RmtEn, EoB, Umask, Msk[28:0], MXtd, and MDir

These parts of a message object may be changed without clearing MsgVal: ID[28-0], Xtd, DLC[3:0], Data[7:0], TxRqst, NewDat, MsgLst, and IntPnd.

27.8 Message Handling

When initialization is finished, the DCAN module synchronizes itself to the traffic on the CAN bus. It does acceptance filtering on received messages and stores those frames that are accepted into the designated message objects.

The application has to update the data of the messages to be transmitted and enable and request their transmission. The transmission is requested automatically when a matching Remote Frame is received.

The application may read messages that are received and accepted. Messages that are not read before the next messages is accepted for the same message object will be overwritten.

Messages may be read based on interrupts or by polling.

27.8.1 Message Handler Overview

The Message Handler state machine controls the data transfer between the Rx/Tx Shift Register of the CAN Core and the Message RAM. It performs the following tasks:

- Data Transfer from Message RAM to CAN Core (messages to be transmitted).
- Data Transfer from CAN Core to the Message RAM (received messages).
- Data Transfer from CAN Core to the Acceptance Filtering unit.
- Scanning of Message RAM for a matching message object (acceptance filtering).
- Scanning the same message object after being changed by IF1/IF2 registers when priority is same or higher as message the object found by last scanning.
- Handling of TxRqst flags.
- Handling of interrupt flags.

The Message Handler registers contains status flags of all message objects grouped into the following topics:

- Transmission Request flags
- New Data flags
- Interrupt Pending Flags
- Message Valid Registers

Instead of collecting the listed status information of each message object via IFx registers separately, these Message Handler registers provides a fast and easy way to get an overview (for example, about all pending transmission requests).

All Message Handler registers are read-only.

27.8.2 Receive/Transmit Priority

The receive/transmit priority for the message objects is attached to the message number, not to the CAN identifier. Message object 1 has the highest priority, while the last implemented message object has the lowest priority. If more than one transmission request is pending, they are serviced due to the priority of the corresponding message object, so messages with the highest priority can be placed in the message objects with the lowest numbers.

The acceptance filtering for received Data Frames or Remote Frames is also done in ascending order of message objects, so a frame that has been accepted by a message object cannot be accepted by another message object with a higher Message Number. The last message object may be configured to accept any Data Frame or Remote Frame that was not accepted by any other message object, for nodes that need to log the complete message traffic on the CAN bus.

27.8.3 Transmission of Messages in Event Driven CAN Communication

If the shift register of the CAN Core is ready for loading and if there is no data transfer between the IFx Registers and Message RAM, the d bits in the Message Valid Register and the TxRqst bits in the Transmission Request Register are evaluated. The valid message object with the highest priority pending transmission request is loaded into the shift register by the Message Handler and the transmission is started. The message object’s NewDat bit is reset.

After a successful transmission and if no new data was written to the message object (NewDat = ‘0’) since the start of the transmission, the TxRqst bit will be reset. If TxIE is set, IntPnd will be set after a successful transmission. If the DCAN has lost the arbitration or if an error occurred during the transmission, the message will be retransmitted as soon as the CAN bus is free again. If meanwhile the transmission of a message with higher priority has been requested, the messages will be transmitted in the order of their priority.

If Automatic Retransmission mode is disabled by setting the DAR bit in the CAN Control Register, the behavior of bits TxRqst and NewDat in the Message Control Register of the Interface Register set is as follows:

- When a transmission starts, the TxRqst bit of the respective Interface Register set is reset, while bit NewDat remains set.
- When the transmission has been successfully completed, the NewDat bit is reset.

When a transmission failed (lost arbitration or error) bit NewDat remains set. To restart the transmission, the application has to set TxRqst again.

Received Remote Frames do not require a Receive Object. They will automatically trigger the transmission of a Data Frame, if in the matching Transmit Object the RmtEn bit is set.

27.8.4 Updating a Transmit Object

The CPU may update the data bytes of a Transmit Object any time via the IF1/IF2 Interface Registers, neither d nor TxRqst have to be reset before the update.

Even if only a part of the data bytes are to be updated, all four bytes in the corresponding IF1/IF2 Data A Register or IF1/IF2 Data B Register have to be valid before the content of that register is transferred to the message object. Either the CPU has to write all four bytes into the IF1/IF2 Data Register or the message object is transferred to the IF1/IF2 Data Register before the CPU writes the new data bytes.

When only the data bytes are updated, first 0x87 can be written to bits [23:16] of the Command Register and then the number of the message object is written to bits [7:0] of the Command Register, concurrently updating the data bytes and setting TxRqst with NewDat.

To prevent the reset of TxRqst at the end of a transmission that may already be in progress while the data is updated, NewDat has to be set together with TxRqst in event driven CAN communication. For details, see Section 27.8.3.

When NewDat is set together with TxRqst, NewDat will be reset as soon as the new transmission has started.

27.8.5 Changing a Transmit Object

If the number of implemented message objects is not sufficient to be used as permanent message objects only, the Transmit Objects may be managed dynamically. The CPU can write the whole message (Arbitration, Control, and Data) into the Interface Register. The bits [23:16] of the Command Register can be set to 0xB7 for the transfer of the whole message object content into the message object. Before changing the configuration of a message object, MsgVal has to be reset (see Section 27.7.7).

If a previously requested transmission of this message object is not completed but already in progress, it will be continued; however it will not be repeated if it is disturbed.

To only update the data bytes of a message to be transmitted, bits [23:16] of the Command Register should be set to 0x87.

NOTE: After the update of the Transmit Object, the Interface Register set will contain a copy of the actual contents of the object, including the part that had not been updated.

27.8.6 Acceptance Filtering of Received Messages

When the arbitration and control bits (Identifier + IDE + RTR + DLC) of an incoming message is completely shifted into the shift register of the CAN Core, the Message Handler starts to scan of the Message RAM for a matching valid message object:

- The Acceptance Filtering unit is loaded with the arbitration bits from the CAN Core shift register.
- Then the arbitration and mask bits (including MsgVal, UMask, NewDat, and EoB) of Message Object 1 are loaded into the Acceptance Filtering unit and are compared with the arbitration bits from the shift register. This is repeated for all following message objects until a matching message object is found, or until the end of the Message RAM is reached.
- If a match occurs, the scanning is stopped and the Message Handler proceeds depending on the type of the frame (Data Frame or Remote Frame) received.

27.8.7 Reception of Data Frames

The Message Handler stores the message from the CAN Core shift register into the respective message object in the Message RAM. Not only the data bytes, but all arbitration bits and the Data Length Code are stored into the corresponding message object. This ensures that the data bytes stay associated to the identifier even if arbitration mask registers are used.

The NewDat bit is set to indicate that new data (not yet seen by the CPU) has been received. The CPU should reset the NewDat bit when it reads the message object. If at the time of the reception the NewDat bit was already set, MsgLst is set to indicate that the previous data (supposedly not seen by the CPU) is lost. If the RxIE bit is set, the IntPnd bit is set, causing the Interrupt Register to point to this message object.

The TxRqst bit of this message object is reset to prevent the transmission of a Remote Frame, while the requested Data Frame has just been received.

27.8.8 Reception of Remote Frames

When a Remote Frame is received, three different configurations of the matching message object have to be considered:

1. Dir = 1 (direction = transmit), RmtEn = 1, UMask = 1 or 0: The TxRqst bit of this message object is set at the reception of a matching Remote Frame. The rest of the message object remains unchanged.
2. Dir = 1 (direction = transmit), RmtEn = 0, UMask = 0: The Remote Frame is ignored, this message object remains unchanged.
3. Dir = 1 (direction = transmit), RmtEn = 0, UMask = 1: The Remote Frame is treated similar to a received Data Frame. At the reception of a matching Remote Frame, the TxRqst bit of this message object is reset. The arbitration and control bits (Identifier + IDE + RTR + DLC) from the shift register are stored in the message object in the Message RAM and the NewDat bit of this message object is set. The data bytes of the message object remain unchanged.

27.8.9 Reading Received Messages

The CPU may read a received message any time via the IFx Interface Registers, the data consistency is guaranteed by the Message Handler state machine.

Typically the CPU will write first 0x7F to bits [23:16] and then the number of the message object to bits [7:0] of the Command Register. That combination will transfer the whole received message from the Message RAM into the Interface Register set. Additionally, the bits NewDat and IntPnd are cleared in the Message RAM (not in the Interface Register set). The values of these bits in the Message Control Register always reflect the status before resetting the bits.

If the message object uses masks for acceptance filtering, the arbitration bits show which of the different matching messages has been received.

The actual value of NewDat shows whether a new message has been received since last time when this message object was read. The actual value of MsgLst shows whether more than one message have been received since the last time when this message object was read. MsgLst will not be automatically reset.

27.8.10 Requesting New Data for a Receive Object

By means of a Remote Frame, the CPU may request another CAN node to provide new data for a receive object. Setting the TxRqst bit of a receive object will cause the transmission of a Remote Frame with the receive object's identifier. This Remote Frame triggers the other CAN node to start the transmission of the matching Data Frame. If the matching Data Frame is received before the Remote Frame could be transmitted, the TxRqst bit is automatically reset.

Setting the TxRqst bit without changing the contents of a message object requires the value 0x84 in bits [23:16] of the Command Register.

27.8.11 Storing Received Messages in FIFO Buffers

Several message objects may be grouped to form one or more FIFO Buffers. Each FIFO Buffer configured to store received messages with a particular (group of) Identifier(s). Arbitration and Mask Registers of the FIFO Buffer's message objects are identical. The EoB (End of Buffer) bits of all but the last of the FIFO Buffer's message objects are '0', in the last one the EoB bit is 1.

Received messages with identifiers matching to a FIFO Buffer are stored into a message object of this FIFO Buffer, starting with the message object with the lowest message number.

When a message is stored into a message object of a FIFO Buffer the NewDat bit of this message object is set. By setting NewDat while EoB is 0 the message object is locked for further write accesses by the Message Handler until the CPU has cleared the NewDat bit.

Messages are stored into a FIFO Buffer until the last message object of this FIFO Buffer is reached. If none of the preceding message objects is released by writing NewDat to 0, all further messages for this FIFO Buffer will be written into the last message object of the FIFO Buffer (EoB = 1) and therefore overwrite previous messages in this message object.

27.8.12 Reading from a FIFO Buffer

Several messages may be accumulated in a set of message objects that are concatenated to form a FIFO Buffer before the application program is required (in order to avoid the loss of data) to empty the buffer.

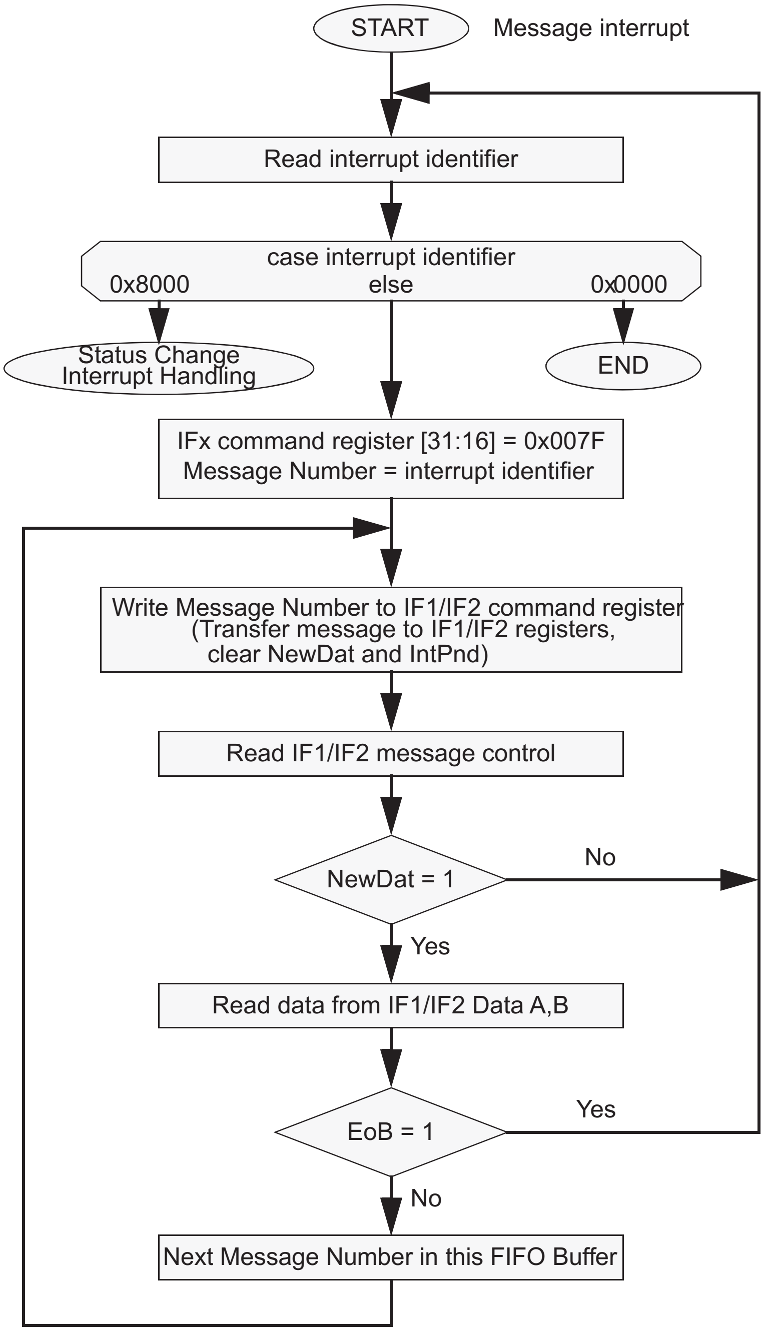
A FIFO Buffer of length N will store N-1 plus the last received message since last time it was cleared.

A FIFO Buffer is cleared by reading and resetting the NewDat bits of all its message objects, starting at the FIFO Object with the lowest message number. This should be done in a subroutine following the example shown in Figure 27-12.

NOTE: All message objects of a FIFO buffer needs to be read and cleared before the next batch of messages can be stored. Otherwise true FIFO functionality can not be guaranteed, since the message objects of a partly read buffer will be re-filled according to the normal (descending) priority.

Reading from a FIFO Buffer message object and resetting its NewDat bit is handled the same way as reading from a single message object.

Figure 27-12. CPU Handling of a FIFO Buffer (Interrupt Driven)


START
Message interrupt
Read interrupt identifier
case interrupt identifier
0x8000
else
0x0000
Status Change Interrupt Handling
END
IFx command register [31:16] = 0x007F
Message Number = interrupt identifier
Write Message Number to IF1/IF2 command register
(Transfer message to IF1/IF2 registers,
clear NewDat and IntPnd)
Read IF1/IF2 message control
NewDat = 1
No
Yes
Read data from IF1/IF2 Data A,B
EoB = 1
Yes
No
Next Message Number in this FIFO Buffer

27.9 CAN Message Transfer

Once the DCAN is initialized and Init bit is reset to zero, the CAN Core synchronizes itself to the CAN bus and is ready for message transfer as per the configured message objects.

The CPU may enable the interrupt lines (setting IE0 and IE1 to 1) at the same time when it clears Init and CCE. The status interrupts EIE and SIE may be enabled simultaneously.

The CAN communication can be carried out in any of the following two modes:

1. Interrupt mode
2. Polling mode.

The Interrupt Register points to those message objects with IntPnd = 1. It is updated even if the interrupt lines to the CPU are disabled (IE0 / IE1 are zero).

The CPU may poll all Message Object's NewDat and TxRqst bits in parallel from the NewData X Registers and the Transmission Request X Registers. Polling can be made easier if all Transmit Objects are grouped at the low numbers, all Receive Objects are grouped at the high numbers.

Received messages are stored into their appropriate message objects if they pass acceptance filtering.

The whole message (including all arbitration bits, DLC and up to eight data bytes) is stored into the message object. As a consequence, when the identifier mask is used, the arbitration bits that are masked to “don't care” may change in the message object when a received message is stored.

The CPU may read or write each message at any time via the Interface Registers, as the Message Handler guarantees data consistency in case of concurrent accesses (for reconfiguration, see Section 27.7.6)

If a permanent message object (arbitration and control bits set up during configuration and leaving unchanged for multiple CAN transfers) exists for the message, it is possible to only update the data bytes.

If several transmit messages should be assigned to one message object, the whole message object has to be configured before the transmission of this message is requested.

The transmission of multiple message objects may be requested at the same time. They are subsequently transmitted, according to their internal priority.

Messages may be updated or set to not valid at any time, even if a requested transmission is still pending (for reconfiguration, see Section 27.7.7). However, the data bytes will be discarded if a message is updated before a pending transmission has started.

Depending on the configuration of the message object, a transmission may be automatically requested by the reception of a remote frame with a matching identifier.

27.9.1 Automatic Retransmission

According to the CAN Specification (ISO11898), the DCAN provides a mechanism to automatically retransmit frames that have lost arbitration or have been disturbed by errors during transmission. The frame transmission service will not be confirmed to you before the transmission is successfully completed.

By default, this automatic retransmission is enabled. It can be disabled by setting bit DAR (Disable Automatic Retransmission) in CAN Control Register. Further details to this mode are provided in Section 27.8.3.

27.9.2 Auto-Bus-On

Per default, after the DCAN has entered Bus-Off state, the CPU can start a Bus-Off-Recovery sequence by resetting Init bit. If this is not done, the module will stay in Bus-Off state.

The DCAN provides an automatic Auto-Bus-On feature that is enabled by bit ABO in CAN Control Register. If set, the DCAN will automatically start the Bus-Off-Recovery sequence. The sequence can be delayed by a user-defined number of VCLK cycles that can be defined in Auto-Bus-On Time Register.

NOTE: If the DCAN goes Bus-Off due to massive occurrence of CAN bus errors, it stops all bus activities and automatically sets the Init bit. Once the Init bit has been reset by the CPU or due to the Auto-Bus-On feature, the device will wait for 129 occurrences of Bus Idle (equal to 129 × 11 consecutive recessive bits) before resuming normal operation. At the end of the Bus-Off recovery sequence, the error counters will be reset.

27.10 Interrupt Functionality

Interrupts can be generated on two interrupt lines:

1. DCAN0INT line
2. DCAN1INT line

These lines can be enabled by setting the IE0 and IE1 bits, respectively, in the CAN Control Register.

The DCAN provides three groups of interrupt sources: Message Object Interrupts, Status Change Interrupts and Error Interrupts (see Figure 27-13 and Figure 27-14).

The source of an interrupt can be determined by the interrupt identifiers Int0ID / Int1ID in the Interrupt Register (see Section 27.17.5). When no interrupt is pending, the register will hold the value zero.

Each interrupt line remains active until the dedicated field in the Interrupt Register DCAN INT (Int0ID / Int1ID) again reach zero (this means the cause of the interrupt is reset), or until IE0 / IE1 are reset.

The value 0x8000 in the Int0ID field indicates that an interrupt is pending because the CAN Core has updated (not necessarily changed) the Error and Status Register (Error Interrupt or Status Interrupt). This interrupt has the highest priority. The CPU can update (reset) the status bits WakeUpPnd, RxOk, TxOk and LEC by reading the Error and Status Register DCAN ES, but a write access of the CPU will never generate or reset an interrupt.

Values between 1 and the number of the last message object indicates that the source of the interrupt is one of the message objects, Int0ID resp. Int1ID will point to the pending message interrupt with the highest priority. The Message Object 1 has the highest priority, the last message object has the lowest priority.

An interrupt service routine that reads the message that is the source of the interrupt, may read the message and reset the message object's IntPnd at the same time (ClrIntPnd bit in the IF1/IF2 Command Register). When IntPnd is cleared, the Interrupt Register will point to the next message object with a pending interrupt.

27.10.1 Message Object Interrupts

Message Object interrupts are generated by events from the message objects. They are controlled by the flags IntPND, TxIE and RxIE, that are described in Section 27.5.1.

Message Object interrupts can be routed to either DCAN0INT or DCAN1INT line, controlled by the Interrupt Multiplexer Register (see Section 27.17.22).

27.10.2 Status Change Interrupts

The events WakeUpPnd, RxOk, TxOk and LEC in Error and Status Register (DCAN ES) belong to the Status Change Interrupts. The Status Change Interrupt group can be enabled by bit in CAN Control Register.

If SIE is set, a Status Change Interrupt will be generated at each CAN frame, independent of bus errors or valid CAN communication, and also independent of the Message RAM configuration.

Status Change interrupts can only be routed to interrupt line DCAN0INT that has to be enabled by setting IE0 in the CAN Control Register.

NOTE: Reading the Error and Status Register will clear the WakeUpPnd flag. If in global power down mode, the WakeUpPnd flag is cleared by such a read access before the DCAN module has been waken up by the system, the DCAN may re-assert the WakeUpPnd flag, and a second interrupt may occur (additional information can be found in Section 27.11.2).

27.10.3 Error Interrupts

The events PER, BOff and EWarn (monitored in Error and Status Register, DCAN ES) belong to the Error Interrupts. The Error Interrupt group can be enabled by setting bit EIE in CAN Control Register.

Error interrupts can only be routed to interrupt line DCAN0INT that has to be enabled by setting IE0 in the CAN Control Register.

Figure 27-13. CAN Interrupt Topology 1

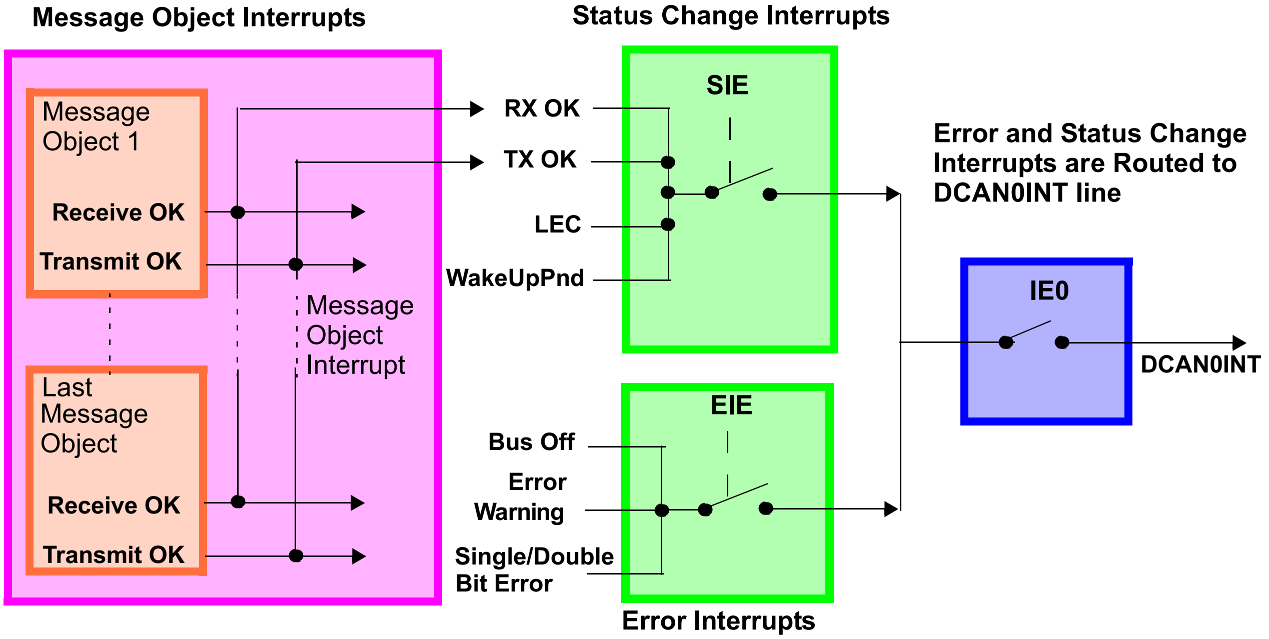
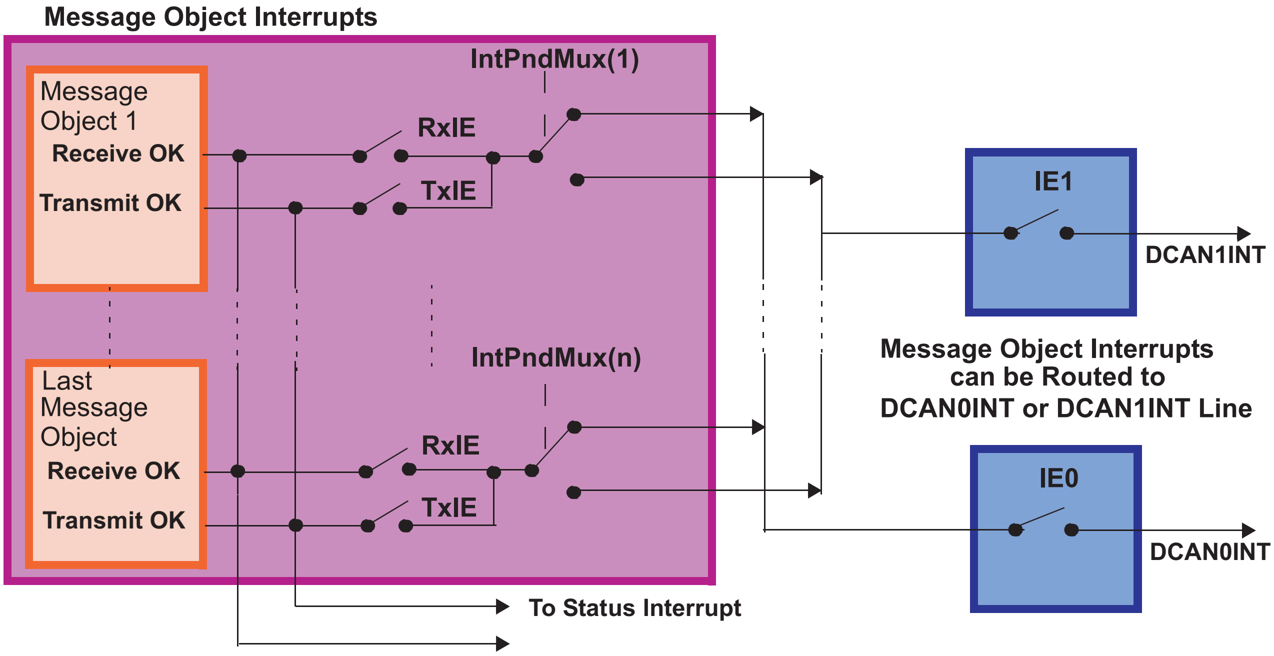


Figure 27-14. CAN Interrupt Topology 2



Details of Interrupt Mapping for actual device will be described in the device specific data sheet.

27.11 Global Power Down Mode

The device architecture supports a centralized global power down control over the peripheral modules through the Peripheral Central Resource (PCR) module (Additional information can be found in Platform Architecture Specification).

27.11.1 Entering Global Power Down Mode

The global power down mode for the DCAN is requested by setting the appropriate Peripheral Power Down Set bit (**PSPWRDWNSETx**) in the PCR module.

The DCAN then finishes all transmit requests of the message objects. When all requests are done, the DCAN waits until a bus idle state is recognized. Then it will automatically set the Initbit to indicate that the global power down mode has been entered.

27.11.2 Wakeup From Global Power Down Mode

When the DCAN module is in global power down mode, a CAN bus activity detection circuit exists, which can be active, if enabled. If this circuit is active,on occurrence of a dominant CAN bus level, the DCAN will set the **WakeUpPnd** bit in Error and Status Register (DCAN ES).

If Status Interrupts are enabled, also an interrupt will be generated. This interrupt could be used by the application to wakeup the DCAN. For this, the application needs to set the appropriate Peripheral Power Down Clear bit (**PSPWRDWNCLRx**) in the PCR module, and to clear the Init bit in CAN Control Register.

After the Init bit has been cleared, the DCAN module waits until it detects 11 consecutive recessive bits on the CAN_RX pin and then goes Bus-Active again.

NOTE: The CAN transceiver circuit has to stay active during CAN bus activity detection. The first CAN message, which initiates the bus activity, cannot be received. This means that the first message received in power down mode is lost.

27.12 Local Power Down Mode

Besides the centralized power down mechanism controlled by the PCR module (global power down, see Section 27.15), the DCAN supports a local power down mode that can be controlled within the DCAN control registers.

27.12.1 Entering Local Power Down Mode

The local power down mode is requested by setting the PDR bit in CAN Control Register.

The DCAN then finishes all transmit requests of the message objects. When all requests are done, DCAN waits until a bus idle state is recognized. Then it will automatically set the Initbit in CAN Control Register to prevent any further CAN transfers, and it will also set the PDA bit in CAN Error and Status Register. With setting the PDA bits, the DCAN module indicates that the local power down mode has been entered.

During local power down mode, the internal clocks of the DCAN module are turned off, but there is a wake up logic (see Section 27.12.2) that can be active, if enabled. Also the actual contents of the control registers can be read back.

NOTE: In local low power mode, the application should not clear the Init bit while PDR is set. If there are any messages in the Message RAM configured as to be transmitted and the application resets the init bit, these messages may be sent.

27.12.2 Wakeup From Local Power Down

There are two ways to wake up the DCAN from local power down mode:

1. The application could wake up the DCAN module manually by clearing the PDR bit and then clearing the Init bit in CAN Control Register.
2. Alternatively, a CAN bus activity detection circuit can be activated by setting the wake up on bus activity bit (WUBA) in CAN Control Register. If this circuit is active, on occurrence of a dominant CAN bus level, the DCAN will automatically start the wake up sequence. It will clear the PDR bit in CAN Control Register and also clear the PDA bit in Error and Status Register. The WakeUpPnd bit in CAN Error and Status Register will be set. If Status Interrupts are enabled, also an interrupt will be generated. Finally the Init bit in CAN control register will be cleared.

After the Init bit has been cleared, the module waits until it detects 11 consecutive recessive bits on the CAN_RX pin and then goes Bus-Active again.

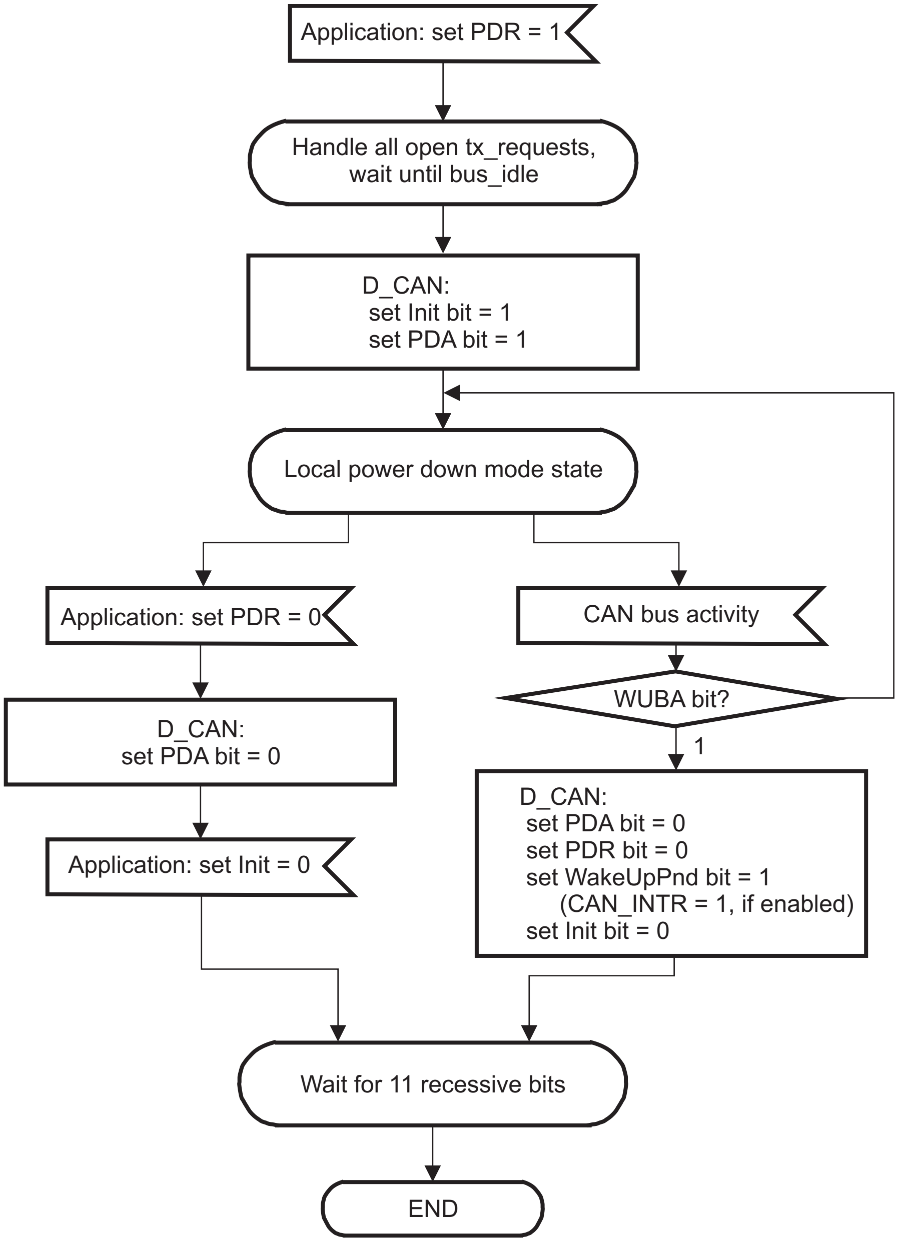
NOTE: The CAN transceiver circuit has to stay active while CAN bus observation. The first CAN message, which initiates the bus activity, cannot be received. This means that the first message received in power down and automatic wake-up mode, is lost.

Figure 27-15 shows a flow diagram about entering and leaving local power down mode.

27.13 GIO Support

The CAN_RX and CAN_TX pins of each DCAN module can be used as general purpose IO pins, if CAN functionality is not needed. This function is controlled by the CAN TX IO Control register (see Section 27.17.34) and the CAN RX IO Control register (see Section 27.17.35).

Figure 27-15. Local Power Down Mode Flow Diagram



27.14 Test Modes

The DCAN provides several test modes that are mainly intended for production tests or self test.

For all test modes, Test bit in the CAN Control Register needs to be set to one. This enables write access to the Test Register.

NOTE: When using any of the Loop Back modes, it must be ensured by software that all message transfers are finished before setting the Init bit to ‘1’.

27.14.1 Silent Mode

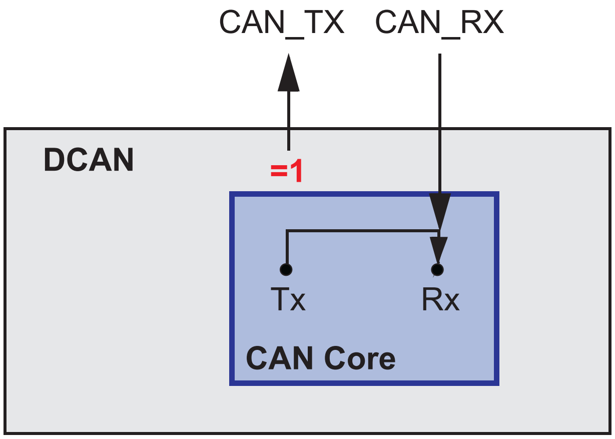
The Silent Mode may be used to analyze the traffic on the CAN bus without affecting it by sending dominant bits (for example, acknowledge bit, overload flag, active error flag). The DCAN is still able to receive valid data frames and valid remote frames, but it will not send any dominant bits. However, these are internally routed to the CAN Core.

Figure 27-16 shows the connection of signals CAN_TX and CAN_RX to the CAN Core in Silent Mode.

Silent Mode can be activated by setting the Silent bit in Test Register to 1.

In ISO 11898-1, the Silent Mode is called the Bus Monitoring Mode.

Figure 27-16. CAN Core in Silent Mode



27.14.2 Loop Back Mode

The Loop Back Mode is mainly intended for hardware self-test functions. In this mode, the CAN Core uses internal feedback from Tx output to Rx input. Transmitted messages are treated as received messages, and can be stored into message objects if they pass acceptance filtering. The actual value of the CAN_RX input pin is disregarded by the CAN Core. Transmitted messages still can be monitored at the CAN_TX pin.

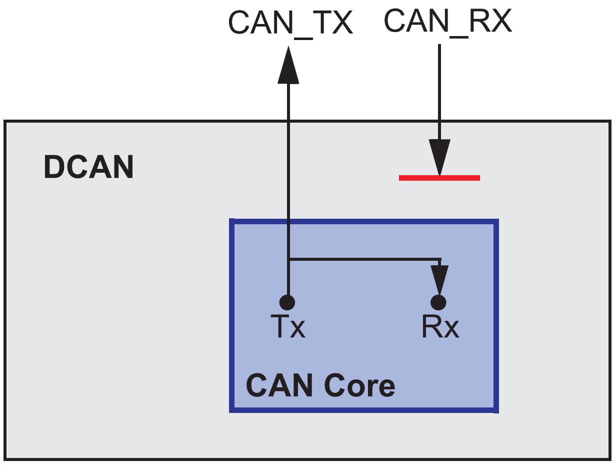
In order to be independent from external stimulation, the CAN Core ignores acknowledge errors (recessive bit sampled in the acknowledge slot of a data/remote frame) in Loop Back Mode.

Figure 27-17 shows the connection of signals CAN_TX and CAN_RX to the CAN Core in Loop Back Mode.

Loop Back Mode can be activated by setting bit LBack in Test Register to 1.

NOTE: In Loop Back mode, the signal path from CAN Core to Tx pin, the Tx pin itself, and the signal path from Tx pin back to CAN Core are disregarded. For including these into the testing, see External Loop Back mode (Section 27.14.3).

Figure 27-17. CAN Core in Loop Back Mode



27.14.3 External Loop Back Mode

The External Loop Back Mode is similar to the Loop Back Mode, however it includes the signal path from CAN Core to Tx pin, the Tx pin itself, and the signal path from Tx pin back to CAN Core. When External Loop Back Mode is selected, the input of the CAN core is connected to the input buffer of the Tx pin.

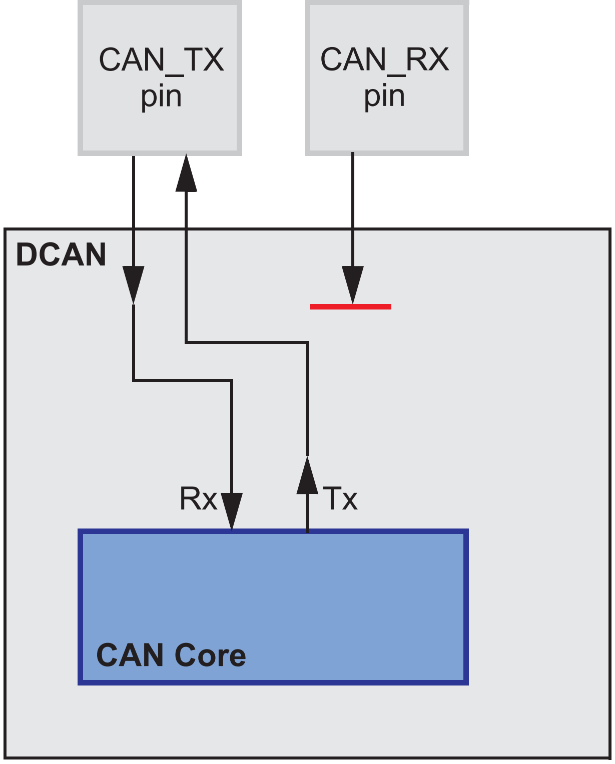
With this configuration, the Tx pin IO circuit can be tested.

External Loop Back Mode can be activated by setting bit ExL in Test Register to 1.

Figure 27-18 shows the connection of signals CAN_TX and CAN_RX to the CAN Core in External Loop Back Mode.

NOTE: When Loop Back Mode is active (LBack bit set), the ExL bit will be ignored.

Figure 27-18. CAN Core in External Loop Back Mode

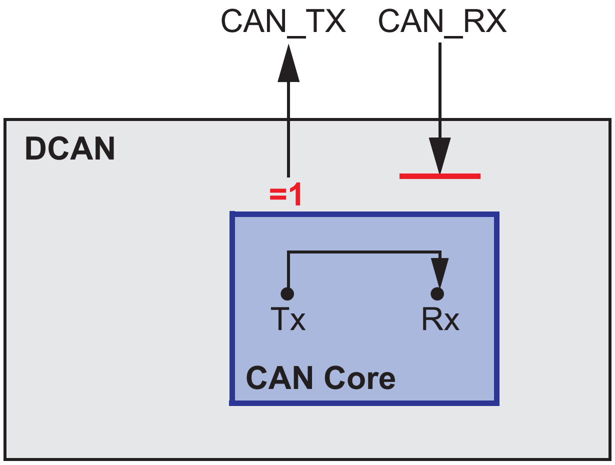


27.14.4 Loop Back Combined with Silent Mode

It is also possible to combine Loop Back Mode and Silent Mode by setting bits LBack and Silent at the same time. This mode can be used for a “Hot Selftest”, that is, the DCAN hardware can be tested without affecting the CAN network. In this mode, the CAN_RX pin is disconnected from the CAN Core and no dominant bits will be sent on the CAN_TX pin.

Figure 27-19 shows the connection of the signals CAN_TX and CAN_RX to the CAN Core in case of the combination of Loop Back Mode with Silent Mode.

Figure 27-19. CAN Core in Loop Back Combined with Silent Mode



27.14.5 Software Control of CAN_TX Pin

Four output functions are available for the CAN transmit pin CAN_TX. Additionally to its default function (serial data output), the CAN_TX pin can drive constant dominant or recessive values, or it can drive the CAN Sample Point signal to monitor the CAN Core’s bit timing.

Combined with the readable value of the CAN_RX pin, this can be used to check the physical layer of the CAN bus.

The output mode of pin CAN_TX is selected by programming the Test Register bits Tx[1:0] as described in Section 27.17.6.

NOTE: The software control for pin CAN_TX interferes with CAN protocol functions. For CAN message transfer or any of the test modes Loop Back Mode, External Loop Back Mode or Silent Mode, the CAN_TX pin should operate in its default functionality.

27.15 SECDED Mechanism

The DCAN module provides a single-bit error correction and double-bit error detection (SECDED) mechanism to ensure data integrity of Message RAM data. For each message object (136 bits) in the Message RAM, 9 ECC bits will be calculated. See Section 27.5.5.

The ECC bits are stored in a dedicated RAM. They will be generated on write accesses and will be checked on read accesses.

The SECDED functionality can be enabled or disabled by PMD bit field in CAN Control Register. If SECDED is enabled, ECC bits will be automatically generated and checked.

With the ECCMODE field in the ECC Control and Status register the single-bit error correction can be enabled or disabled (default: enabled).

NOTE: During RAM initialization, no ECC check will be done, but if the PMD bit is set, the ECC bits will be generated.

27.15.1 Behavior on Single-Bit Error

If a single-bit error is detected with single-bit error correction enabled, the correction will be done and the SEFLG in the ECC Control and Status register will be set.

If single-bit error correction is disabled and a single-bit error is detected then the SEFLG in the ECC Control and Status register and the the PER bit in the Error and Status register will be set. If error interrupts are enabled, also an interrupt would be generated. In order to avoid the transmission of invalid data over the CAN bus, the MsgVal bit of the message object will be reset.

The message object number where the single-bit error has occurred will be indicated in the ECC single-bit Error Code Register.

When single-bit error correction is disabled the message object data can be read by the host CPU, independently of single-bit errors. Thus, the application has to ensure that the read data is valid, for example, by immediately checking the ECC single-bit Error Code Register on single-bit error interrupt.

27.15.2 Behavior on Double-Bit Error

If a double-bit error is detected, then the DEFLG in the ECC Control and Status register and the PER bit in Error and Status Register will be set. If error interrupts are enabled, also an interrupt would be generated. In order to avoid the transmission of invalid data over the CAN bus, the MsgVal bit of the message object will be reset. The message object number will be indicated in the Parity Error Code Register.

The message object data can be read by the host CPU, independently of double-bit errors. Thus, the application has to ensure that the read data is valid, for example, by immediately checking the Parity Error Code register on double-bit error interrupt.

27.15.3 SECDED Testing

Testing of the SECDED mechanism can be implemented by using the diagnostic mode, which is enabled with the ECCDIAG register. The following procedure can be used:

1. Disable SECDED using DCAN control register. Enable diagnostic mode using the ECCDIAG register

2. Write to corrupt the data (in RDA mode) or ECC bits.

3. Enable SECDED and read data for which ECC is corrupted (either in RDA mode or via IFx registers).

4. single-bit error or double-bit error flag will be set in the diagnostic status register (ECCDIAG STAT) and in the ECC Control and Status register accordingly. A double-bit error or a single-bit error with single-bit error correction disabled also triggers the PER flag.

5. Disable diagnostic mode.

27.16 Debug/Suspend Mode

When the CPU is halted during debug, all DCAN registers are visible and can be inspected and modified by the CPU.

In addition, the Message RAM is directly memory-mapped as described in Table 27-3.

The CAN controller provides two options for entering the debug/suspend state. The options are controlled by the IDS bit in the CAN Control Register (DCAN CTL). By default, when IDS is 0, the DCAN controller completes any active transfers on the CAN bus and waits until the bus is idle before halting. When IDS is 1, the DCAN halts immediately as soon as the CPU is halted.

The InitDbg bit in DCAN CTL register indicates when the DCAN controller has actually entered the debug/suspend state.

NOTE: During Debug/Suspend Mode, the Message RAM cannot be accessed via the IFx register sets.

Writing to control registers in debug/suspend mode may influence the CAN state machine and further message handling.

For debug support, the auto clear functionality of the following DCAN registers is disabled:

- Error and Status Register (clear of status flags by read)
- IF1/IF2 Command Registers (clear of DMAActive flag by read/write)

27.17 DCAN Control Registers

Table 27-6 lists the control registers of the DCAN. After hardware reset, the registers of the DCAN hold the values shown in the register descriptions. The base address for the control registers is FFF7 DC00h for DCAN1, FFF7 DE00h for DCAN2, FFF7 E000h for DCAN3, and FFF7 E200h for DCAN4.

Additionally, the Bus-Off state is reset and the CAN_TX pin is set to recessive (HIGH). The Init bit in the CAN Control Register is set to enable the software initialization. The DCAN will not influence the CAN bus until the CPU resets Init to 0.

Table 27-6. DCAN Control Registers

Offset	Acronym	Register Description	Section
00h	DCAN CTL	CAN Control Register	Section 27.17.1
04h	DCAN ES	Error and Status Register	Section 27.17.2
08h	DCAN ERRC	Error Counter Register	Section 27.17.3
0Ch	DCAN BTR	Bit Timing Register	Section 27.17.4
10h	DCAN INT	Interrupt Register	Section 27.17.5
14h	DCAN TEST	Test Register	Section 27.17.6
1Ch	DCAN PERR	Parity Error Code Register	Section 27.17.7
20h	DCAN REL	Core Release Register	Section 27.17.8
24h	DCAN ECCDIAG	ECC Diagnostic Register	Section 27.17.9
28h	DCAN ECCDIAG STAT	ECC Diagnostic Status Register	Section 27.17.10
2Ch	DCAN ECC CS	ECC Control and Status Register	Section 27.17.11
30h	DCAN ECC SERR	ECC Single-Bit Error Code Register	Section 27.17.12
80h	DCAN ABOTR	Auto-Bus-On Time Register	Section 27.17.13
84h	DCAN TXRQX	Transmission Request X Register	Section 27.17.14
88h	DCAN TXRQ12	Transmission Request 12 Register	Section 27.17.15
8Ch	DCAN TXRQ34	Transmission Request 34 Register	Section 27.17.15
90h	DCAN TXRQ56	Transmission Request 56 Register	Section 27.17.15
94h	DCAN TXRQ78	Transmission Request 78 Register	Section 27.17.15
98h	DCAN NWDATX	New Data X Register	Section 27.17.16

Table 27-6. DCAN Control Registers (continued)

Offset	Acronym	Register Description	Section
9Ch	DCAN NWDAT12	New Data 12 Register	Section 27.17.17
A0h	DCAN NWDAT34	New Data 34 Register	Section 27.17.17
A4h	DCAN NWDAT56	New Data 56 Register	Section 27.17.17
A8h	DCAN NWDAT78	New Data 78 Register	Section 27.17.17
ACh	DCAN INTPNDX	Interrupt Pending X Register	Section 27.17.18
B0h	DCAN INTPND12	Interrupt Pending 12 Register	Section 27.17.19
B4h	DCAN INTPND34	Interrupt Pending 34 Register	Section 27.17.19
B8h	DCAN INTPND56	Interrupt Pending 56 Register	Section 27.17.19
BCh	DCAN INTPND78	Interrupt Pending 78 Register	Section 27.17.19
C0h	DCAN MSGVALX	Message Valid X Register	Section 27.17.20
C4h	DCAN MSGVAL12	Message Valid 12 Register	Section 27.17.21
C8h	DCAN MSGVAL34	Message Valid 34 Register	Section 27.17.21
CCh	DCAN MSGVAL56	Message Valid 56 Register	Section 27.17.21
D0h	DCAN MSGVAL78	Message Valid 78 Register	Section 27.17.21
D8h	DCAN INTMUX12	Interrupt Multiplexer 12 Register	Section 27.17.22
DCh	DCAN INTMUX34	Interrupt Multiplexer 34 Register	Section 27.17.22
E0h	DCAN INTMUX56	Interrupt Multiplexer 56 Register	Section 27.17.22
E4h	DCAN INTMUX78	Interrupt Multiplexer 78 Register	Section 27.17.22
100h	DCAN IF1CMD	IF1Command Register	Section 27.17.23
104h	DCAN IF1MSK	IF1 Mask Register	Section 27.17.24
108h	DCAN IF1ARB	IF1 Arbitration Register	Section 27.17.25
10Ch	DCAN IF1MCTL	IF1 Message Control Register	Section 27.17.26
110h	DCAN IF1DATA	IF1 Data A Register	Section 27.17.27
114h	DCAN IF1DATB	IF1 Data B Register	Section 27.17.27
120h	DCAN IF2CMD	IF2 Command Register	Section 27.17.23
124h	DCAN IF2MSK	IF2 Mask Register	Section 27.17.24
128h	DCAN IF2ARB	IF2 Arbitration Register	Section 27.17.25
12Ch	DCAN IF2MCTL	IF2 Message Control Register	Section 27.17.26
130h	DCAN IF2DATA	IF2 Data A Register	Section 27.17.27
134h	DCAN IF2DATB	IF2 Data B Register	Section 27.17.27
140h	DCAN IF3OBS	IF3 Observation Register	Section 27.17.28
144h	DCAN IF3MSK	IF3 Mask Register	Section 27.17.29
148h	DCAN IF3ARB	IF3 Arbitration Register	Section 27.17.30
14Ch	DCAN IF3MCTL	IF3 Message Control Register	Section 27.17.31
150h	DCAN IF3DATA	IF3 Data A Register	Section 27.17.32
154h	DCAN IF3DATB	IF3 Data B Register	Section 27.17.32
160h	DCAN IF3UPD12	IF3 Update Enable 12 Register	Section 27.17.33
164h	DCAN IF3UPD34	IF3 Update Enable 34 Register	Section 27.17.33
168h	DCAN IF3UPD56	IF3 Update Enable 56 Register	Section 27.17.33
16Ch	DCAN IF3UPD78	IF3 Update Enable 78 Register	Section 27.17.33
1E0h	DCAN TIOC	CAN TX IO Control Register	Section 27.17.34
1E4h	DCAN RIOC	CAN RX IO Control Register	Section 27.17.35

27.17.1 CAN Control Register (DCAN CTL)

NOTE: The Bus-Off recovery sequence (see CAN specification) cannot be shortened by setting or resetting Init bit. If the module goes Bus-Off, it will automatically set the Init bit and stop all bus activities.

When the Init bit is cleared by the application again, the module will then wait for 129 occurrences of Bus Idle (129 × 11 consecutive recessive bits) before resuming normal operation. At the end of the Bus-Off recovery sequence, the error counters will be reset.

After the Init bit is reset, each time when a sequence of 11 recessive bits is monitored, a Bit0 error code is written to the Error and Status Register, enabling the CPU to check whether the CAN bus is stuck at dominant or continuously disturbed, and to monitor the proceeding of the Bus-Off recovery sequence.

Figure 27-20. CAN Control Register (DCAN CTL) [offset = 00h]

31	26	25	24
Reserved		WUBA	PDR
R-0		R/W-0	R/W-0

23	21	20	19	18	17	16
Reserved		DE3	DE2	DE1	IE1	InitDbg
R-0		R/W-0	R/W-0	R/W-0	R/W-0	R-0

15	14	13	10	9	8
SWR	Reserved	PMD		ABO	IDS
R/WP-0	R-0	R/W-5h		R/W-0	R/W-0

7	6	5	4	3	2	1	0
Test	CCE	DAR	Reserved	EIE	SIE	IE0	Init
R/W-0	R/W-0	R/W-0	R-0	R/W-0	R/W-0	R/W-0	R/W-1

LEGEND: R/W = Read/Write; R = Read only; WP = Write protected by Init bit; -*n* = value after reset

Table 27-7. CAN Control Register (DCAN CTL) Field Descriptions

Bit	Field	Value	Description
31-26	Reserved	0	These bits are always read as 0. Writes have no effect.
25	WUBA		Automatic wake up on bus activity when in local power down mode.
		0	No detection of a dominant CAN bus level while in local power down mode.
		1	Detection of a dominant CAN bus level while in local power down mode is enabled. On occurrence of a dominant CAN bus level, the wake up sequence is started. (Additional information can be found in Section 27.12.)
			Note: The CAN message, which Initiates the bus activity, cannot be received. This means that the first message received in power down and automatic wake-up mode, will be lost.
24	PDR		Request for local low power down mode.
		0	No application request for local low power down mode. If the application has cleared this bit while DCAN in local power down mode, also the Init bit has to be cleared.
		1	Local power down mode has been requested by application. The DCAN will acknowledge the local power down mode by setting bit PDA in Error and Status Register. The local clocks will be turned off by DCAN internal logic (Additional information can be found in Section 27.12).
23-21	Reserved	0	These bits are always read as 0. Writes have no effect.
20	DE3		Enable DMA request line for IF3.
		0	Disabled
		1	Enabled
			Note: A pending DMA request for IF3 remains active until first access to one of the IF3 registers.

Table 27-7. CAN Control Register (DCAN CTL) Field Descriptions (continued)

Bit	Field	Value	Description
19	DE2		Enable DMA request line for IF2.
		0	Disabled
		1	Enabled
			Note: A pending DMA request for IF2 remains active until first access to one of the IF2 registers.
18	DE1		Enable DMA request line for IF1.
		0	Disabled
		1	Enabled
			Note: A pending DMA request for IF1 remains active until first access to one of the IF1 registers.
17	IE1		Interrupt line 1 enable.
		0	Disabled. Module Interrupt DCAN1INT is always low.
		1	Enabled. Interrupts will assert line DCAN1INT to one; line remains active until pending interrupts are processed.
16	InitDbg		Internal Init state while debug access.
		0	Not in debug mode, or debug mode requested but is not entered.
		1	Debug mode requested and is internally entered; the DCAN is ready for debug accesses.
15	SWR		SW reset enable.
		0	Normal operation.
		1	Module is forced to reset state. This bit will automatically get cleared after execution of SW reset after one VBUSP clock cycle.
			Note: To execute SW reset the following procedure is necessary: 1. Set Init bit to shut down CAN communication. 2. Set SWR bit additionally to Init bit.
14	Reserved	0	This bit is always read as 0. Writes have no effect.
13-10	PMD		SECDED enable.
		5h	SECDED function is disabled.
		All other values	SECDED function is enabled.
9	ABO		Auto-Bus-On enable.
		0	The Auto-Bus-On feature is disabled.
		1	The Auto-Bus-On feature is enabled.
8	IDS		Interruption debug support enable.
		0	When Debug/Suspend mode is requested, DCAN will wait for a started transmission or reception to be completed before entering Debug/Suspend mode.
		1	When Debug/Suspend mode is requested, DCAN will interrupt any transmission or reception, and enter Debug/Suspend mode immediately.
7	Test		Test mode enable.
		0	Normal operation.
		1	Test mode.
6	CCE		Configuration change enable.
		0	The CPU has no write access to the BTR Config register.
		1	The CPU has write access to the BTR configuration register (when Init bit is set).
5	DAR		Disable automatic retransmission.
		0	Automatic Retransmission of not successful messages is enabled.
		1	Automatic Retransmission is disabled.
4	Reserved	0	This bit is always read as 0. Writes have no effect.

Table 27-7. CAN Control Register (DCAN CTL) Field Descriptions (continued)

Bit	Field	Value	Description
3	EIE		Error interrupt enable.
		0	Disabled. PER, BOff, and EWarn bits cannot generate an interrupt.
		1	Enabled. PER, BOff, and EWarn bits can generate an interrupt at DCAN0INT line and affect the Interrupt Register.
2	SIE		Status change interrupt enable.
		0	Disabled. WakeUpPnd, RxOk, TxOk, and LEC bits cannot generate an interrupt.
		1	Enabled. WakeUpPnd, RxOk, TxOk, and LEC can generate an interrupt at DCAN0INT line and affect the Interrupt Register.
1	IE0		Interrupt line 0 enable.
		0	Disabled. Module Interrupt DCAN0INT is always low.
		1	Enabled. Interrupts will assert line DCAN0INT to one; line remains active until pending interrupts are processed.
0	Init		Initialization
		0	Normal operation.
		1	Initialization mode is entered.

27.17.2 Error and Status Register (DCAN ES)

Interrupts are generated by bits PER, BOff, and EWarn (if EIE bit in CAN Control Register is set) and by bits WakeUpPnd, RxOk, TxOk, and LEC (if SIE bit in CAN Control Register is set). A change of bit EPass will not generate an Interrupt.

NOTE: Reading the Error and Status Register clears the WakeUpPnd, PER, RxOk and TxOk bits and set the LEC to value of 7. Additionally, the Status Interrupt value (8000h) in the Interrupt Register will be replaced by the next lower priority interrupt value.

For debug support, the auto clear functionality of Error and Status Register (clear of status flags by read) is disabled when in Debug/Suspend mode.

Figure 27-21. Error and Status Register (DCAN ES) [offset = 04h]

31-16
Reserved
R-0

15-11	10	9	8
Reserved	PDA	WakeUpPnd	PER
R-0	R-0	RC-0	RC-0

7	6	5	4	3	2-0
BOff	EWarn	EPass	RxOK	TxOK	LEC
R-0	R-0	R-0	RC-0	RC-0	RS-7h

LEGEND: R = Read only; C = Clear on read; S = Set on read; -*n* = value after reset

Table 27-8. Error and Status Register (DCAN ES) Field Descriptions

Bit	Field	Value	Description
31-11	Reserved	0	These bits are always read as 0. Writes have no effect.
10	PDA		Local power down mode acknowledge.
		0	DCAN is not in local power down mode.
		1	Application request for setting DCAN to local power down mode was successful. DCAN is in local power down mode.
9	WakeUp Pnd		Wake Up Pending. This bit can be used by the CPU to identify the DCAN as the source to wake up the system.
		0	No Wake Up is requested by DCAN.
		1	DCAN has initiated a wake up of the system due to dominant CAN bus while module power down. This bit will be reset if Error and Status Register is read.
8	PER		Single-/Double-bit error detected. This bit is set on double-bit errors and additionally on single-bit errors, if single-bit error correction is disabled with the ECCMODE bitfield in the ECC Control and Status register.
		0	No single-/double-bit error has been detected since last read access.
		1	The SECDED mechanism has detected a single-/double-bit error in the Message RAM. This bit will be reset if Error and Status Register is read.
7	BOff		Bus-Off State
		0	The CAN module is not Bus-Off state.
		1	The CAN module is in Bus-Off state.
6	EWarn		Warning State
		0	Both error counters are below the error warning limit of 96.
		1	At least one of the error counters has reached the error warning limit of 96.
5	EPass		Error Passive State
		0	On CAN Bus error, the DCAN could send active error frames.
		1	The CAN Core is in the error passive state as defined in the CAN Specification.

Table 27-8. Error and Status Register (DCAN ES) Field Descriptions (continued)

Bit	Field	Value	Description
4	RxOK		Received a message successfully.
		0	No message has been successfully received since the last time when this bit was read by the CPU. This bit is never reset by DCAN internal events.
		1	A message has been successfully received since the last time when this bit was reset by a read access of the CPU (independent of the result of acceptance filtering).This bit will be reset if Error and Status Register is read.
3	TxOK		Transmitted a message successfully.
		0	No message has been successfully transmitted since the last time when this bit was read by the CPU. This bit is never reset by DCAN internal events.
		1	A message has been successfully transmitted (error free and acknowledged by at least one other node) since the last time when this bit was reset by a read access of the CPU. This bit will be reset if Error and Status Register is read.
2-0	LEC		Last Error Code The LEC field indicates the type of the last error on the CAN bus. This field will be cleared to 0 when a message has been transferred (reception or transmission) without error.
		0	No Error
		1h	Stuff Error: More than five equal bits in a row have been detected in a part of a received message where this is not allowed.
		2h	Form Error: A fixed format part of a received frame has the wrong format.
		3h	Ack Error: The message this CAN Core transmitted was not acknowledged by another node.
		4h	Bit1 Error: During the transmission of a message (with the exception of the arbitration field), the device wanted to send a recessive level (bit of logical value 1), but the monitored bus value was dominant.
		5h	Bit0 Error: During the transmission of a message (or acknowledge bit, or active error flag, or overload flag), the device wanted to send a dominant level (logical value 0), but the monitored bus level was recessive. During Bus-Off recovery, this status is set each time a sequence of 11 recessive bits has been monitored. This enables the CPU to monitor the proceeding of the Bus-Off recovery sequence (indicating the bus is not stuck at dominant or continuously disturbed).
		6h	CRC Error: In a received message, the CRC check sum was incorrect. (CRC received for an incoming message does not match the calculated CRC for the received data).
		7h	No CAN bus event was detected since the last time when CPU has read the Error and Status Register. Any read access to the Error and Status Register reinitializes the LEC bit to 7.

27.17.3 Error Counter Register (DCAN ERRC)

Figure 27-22. Error Counter Register (DCAN ERRC) [offset = 08h]

31	16
Reserved	
R-0	

15	14 8	7 0
RP	REC	TEC
R-0	R-0	R-0

LEGEND: R = Read only; -*n* = value after reset

Table 27-9. Error Counter Register (DCAN ERRC) Field Descriptions

Bit	Field	Value	Description
31-16	Reserved	0	These bits are always read as 0. Writes have no effect.
15	RP		Receive Error Passive
		0	The Receive Error Counter is below the error passive level.
		1	The Receive Error Counter has reached the error passive level as defined in the CAN Specification.
14-8	REC	0-7Fh	Receive Error Counter. Actual state of the Receive Error Counter. (Values from 0 to 127).
7-0	TEC	0-FFh	Transmit Error Counter. Actual state of the Transmit Error Counter. (Values from 0 to 255).

27.17.4 Bit Timing Register (DCAN BTR)

NOTE: This register is only writable if CCE and Init bits in the CAN Control Register are set.

The CAN bit time may be programmed in the range of 8 to 25 time quanta.

The CAN time quantum may be programmed in the range of 1 to 1024 CAN_CLK periods.

With a CAN_CLK of 8 MHz and BRPE = 00, the reset value of 2301h configures the DCAN for a bit rate of 500kBit/s.

For details see Section 27.3.2.1.

Figure 27-23. Bit Timing Register (DCAN BTR) [offset = 0Ch]

31 20	19 16
Reserved	BRPE
R-0	R/WP-0

15	14 12	11 8	7 6	5 0
Rsvd	TSeg2	TSeg1	SJW	BRP
R-0	R/WP-2h	R/WP-3h	R/WP-0	R/WP-1h

LEGEND: R/W = Read/Write; R = Read only; WP = Write Protected by CCE bit; -*n* = value after reset

Table 27-10. Bit Timing Register (DCAN BTR) Field Descriptions

Bit	Field	Value	Description
31-20	Reserved	0	These bits are always read as 0. Writes have no effect.
	BRPE	0-Fh	Baud Rate Prescaler Extension. Valid programmed values are 0 to 15. By programming BRPE the Baud Rate Prescaler can be extended to values up to 1024.
15	Reserved	0	This bit is always read as 0. Writes have no effect.
14-12	TSeg2	0-7h	Time segment after the sample point. Valid programmed values are 0 to 7. The actual TSeg2 value that is interpreted for the Bit Timing will be the programmed TSeg2 value + 1.
11-8	TSeg1	1h-Fh	Time segment before the sample point. Valid programmed values are 1 to 15. The actual TSeg1 value interpreted for the Bit Timing will be the programmed TSeg1 value + 1.
7-6	SJW	0-3h	Synchronization Jump Width Valid programmed values are 0 to 3. The actual SJW value interpreted for the Synchronization will be the programmed SJW value + 1.
5-0	BRP	0-3Fh	Baud Rate Prescaler Value by which the CAN_CLK frequency is divided for generating the bit time quanta. The bit time is built up from a multiple of this quanta. Valid programmed values are 0 to 63. The actual BRP value interpreted for the Bit Timing will be the programmed BRP value + 1.

27.17.5 Interrupt Register (DCAN INT)

Figure 27-24. Interrupt Register (DCAN INT) [offset = 10h]

31 24	23 16
Reserved	Int1ID
R-0	R-0

15 0
Int0ID
R-0

LEGEND: R = Read only; -*n* = value after reset

Table 27-11. Interrupt Register (DCAN INT) Field Descriptions

Bit	Field	Value	Description
31-24	Reserved	0	These bits are always read as 0. Writes have no effect.
23-16	Int1ID		Interrupt 1 Identifier (indicates the message object with the highest pending interrupt).
		0	No interrupt is pending.
		1h-40h	Number of message object that caused the interrupt.
		41h-FFh	Unused
			If several interrupts are pending, the CAN Interrupt Register will point to the pending interrupt with the highest priority. The DCAN1INT interrupt line remains active until Int1ID reaches value 0 (the cause of the interrupt is reset) or until IE1 is cleared. A message interrupt is cleared by clearing the message object's IntPnd bit. Among the message interrupts, the message object's interrupt priority decreases with increasing message number.
15-0	Int0ID		Interrupt Identifier (indicates the source of the interrupt).
		0	No interrupt is pending.
		1h-40h	Number of message object that caused the interrupt.
		41h-7FFFh	Unused
		8000h	Error and Status Register value is not 7h.
		8001h-FFFFh	Unused
			If several interrupts are pending, the CAN Interrupt Register will point to the pending interrupt with the highest priority. The DCAN0INT interrupt line remains active until Int0ID reaches value 0 (the cause of the interrupt is reset) or until IE0 is cleared. The Status Interrupt has the highest priority. Among the message interrupts, the message object's interrupt priority decreases with increasing message number.

27.17.6 Test Register (DCAN TEST)

For all test modes, the Test bit in CAN Control Register needs to be set to one. If Test bit is set, the RDA, EXL, Tx1, Tx0, LBack and Silent bits are writable. Bit Rx monitors the state of pin CAN_RX and therefore is only readable. All Test Register functions are disabled when Test bit is cleared.

NOTE: The Test Register is only writable if Test bit in CAN Control Register is set.

Setting Tx[1:0] other than 00 will disturb message transfer.

When the internal loop back mode is active (bit LBack is set), bit EXL will be ignored.

Figure 27-25. Test Register (DCAN TEST) [offset = 14h]

31	16
Reserved	
R-0	

15	10	9	8
Reserved		RDA	EXL
R-0		R/WP-0	R/WP-0

7	6	5	4	3	2	0
Rx	Tx		LBack	Silent	Reserved	
R-U	R/WP-0		R/WP-0	R/WP-0	R-0	

LEGEND: R/W = Read/Write; R = Read only; WP = Write Protected by Test bit; -*n* = value after reset; U = Undefined

Table 27-12. Test Register (DCAN TEST) Field Descriptions

Bit	Field	Value	Description
31-10	Reserved	0	These bits are always read as 0. Writes have no effect.
9	RDA		RAM direct access enable.
		0	Normal operation.
		1	Direct access to the RAM is enabled while in Test Mode.
8	EXL		External loop back mode.
		0	Disabled
		1	Enabled
7	Rx		Receive Pin. Monitors the actual value of the CAN_RX pin.
		0	The CAN bus is dominant.
		1	The CAN bus is recessive.
6-5	Tx		Control of CAN_TX pin.
		0	Normal operation, CAN_TX is controlled by the CAN Core.
		1h	Sample Point can be monitored at CAN_TX pin.
		2h	CAN_TX pin drives a dominant value.
		3h	CAN_TX pin drives a recessive value.
4	LBack		Loop back mode.
		0	Disabled
		1	Enabled
3	Silent		Silent mode.
		0	Disabled
		1	Enabled
2-0	Reserved	0	These bits are always read as 0. Writes have no effect.

27.17.7 Parity Error Code Register (DCAN PERR)

If a double-bit error is detected, the PER flag will be set in the Error and Status Register. This bit is not reset by the SECDED mechanism; it must be reset by reading the Error and Status Register. In addition to the PER flag, the SECDED Error Code Register will indicate the memory area where the double-bit error has been detected (message number). After a double-bit error has been detected, the register will hold the last error code until power is removed.

Figure 27-26. Parity Error Code Register (DCAN PERR) [offset = 1Ch]

31 16
Reserved
R-0

15 14 11	10 8	7 0
Reserved	Word Number	Message Number
R-0	R-U	R-U

LEGEND: R = Read only; U = value is undefined; -*n* = value after reset

Table 27-13. Parity Error Code Register (DCAN PERR) Field Descriptions

Bit	Field	Value	Description
31-11	Reserved	0	These bits are always read as 0. Writes have no effect.
10-8	Word Number	0	Word Number is reserved and it will always read as 0.
7-0	Message Number	1h-FFh	Message object number where double-bit error has been detected. Only values 1h-40h are valid. Values 41h-FFh are invalid.

27.17.8 Core Release Register (DCAN REL)

Figure 27-27. Core Release Register (DCAN REL) [offset = 20h]

31 28	27 24	23 20	19 16
REL	STEP	SUBSTEP	YEAR
R-Ah	R-3h	R-1h	R-7h

15 8	7 0
MON	DAY
R-5h	R-4h

LEGEND: R = Read only; -*n* = value after reset

Table 27-14. Core Release Register (DCAN REL) Field Descriptions

Bit	Field	Value	Description
31-28	REL	0-9h	Core Release. One digit, BCD-coded.
27-24	STEP	0-9h	Step of Core Release. One digit, BCD-coded.
23-20	SUBSTEP	0-9h	Substep of Core Release. One digit, BCD-coded.
19-16	YEAR	0-9h	Design Time Stamp, Year. One digit, BCD-coded. This field is set by constant parameter on DCAN synthesis.
15-8	MON	0-12h	Design Time Stamp, Month. Two digits, BCD-coded. This field is set by constant parameter on DCAN synthesis.
7-0	DAY	0-31h	Design Time Stamp, Day. Two digits, BCD-coded. This field is set by constant parameter on DCAN synthesis.

27.17.9 ECC Diagnostic Register (DCAN ECCDIAG)

Figure 27-28. ECC Diagnostic Register (DCAN ECCDIAG) [offset = 24h]

31 16
Reserved
R-0

15 4	3 0
Reserved	ECCDIAG
R-0	R/WP-Ah

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privileged mode only; -*n* = value after reset; U = Undefined

Table 27-15. ECC Diagnostic Register (DCAN ECCDIAG) Field Descriptions

Bit	Field	Value	Description
31-4	Reserved	0	These bits are always read as 0. Writes have no effect.
3-0	ECCDIAG		SECDED diagnostic mode enable.
		5h	Diagnostic mode is enabled. Single-bit and double-bit errors are shown in the ECCDIAG STAT and the ECC Control and Status register. A double-bit error (or single-bit error with single-bit error correction disabled) also triggers the parity interrupt flag (PER). Memory mapping of ECC RAM is enabled.
		Ah	Diagnostic mode is disabled, single-bit and double-bit errors are shown only in the ECC Control and Status register.
		All other values	Reserved

27.17.10 ECC Diagnostic Status Register (DCAN ECCDIAG STAT)

Figure 27-29. ECC Diagnostic Status Register (DCAN ECCDIAG STAT) [offset = 28h]

31 16
Reserved
R-0

15 9	8	7 1	0
Reserved	DEFLG_DIAG	Reserved	SEFLG_DIAG
R-0	R/W1C-0	R-0	R/W1C-0

LEGEND: R/W = Read/Write; R = Read only; W1C = Write 1 to clear; -*n* = value after reset; U = Undefined

Table 27-16. ECC Diagnostic Status Register (DCAN ECCDIAG STAT) Field Descriptions

Bit	Field	Value	Description
31-9	Reserved	0	These bits are always read as 0. Writes have no effect.
8	DEFLG_DIAG		Double-bit error flag diagnostic.
		0	Read: No double-bit error is detected. Write: The bit is unchanged.
		1	Read: Double-bit error is detected in diagnostic mode. Write: The bit is cleared to 0.
7-1	Reserved	0	These bits are always read as 0. Writes have no effect.
0	SEFLG_DIAG		Single-bit error flag diagnostic.
		0	Read: No single-bit error is detected. Write: The bit is unchanged.
		1	Read: Single-bit error is detected in diagnostic mode. Write: The bit is cleared to 0.

27.17.11 ECC Control and Status Register (DCAN ECC CS)

Figure 27-30. ECC Control and Status Register (DCAN ECC CS) [offset = 2Ch]

31 28	27 24	23 20	19 16
Reserved	SBE_EVT_EN	Reserved	ECCMODE
R-0	R/WP-5h	R-0	R/WP-Ah

15 9	8	7 1	0
Reserved	DEFLG	Reserved	SEFLG
R-0	R/W1C-0	R-0	R/W1C-0

LEGEND: R/W = Read/Write; R = Read only; W1C = Write 1 to clear; WP = Write in privileged mode only; -*n* = value after reset

Table 27-17. ECC Control and Status Register (DCAN ECC CS) Field Descriptions

Bit	Field	Value	Description
31-9	Reserved	0	These bits are always read as 0. Writes have no effect.
27-24	SBE_EVT_EN		Enable SECDED single-bit error event (CAN_SERR signal).
		5h	SECDED single-bit error event is disabled, single-bit errors are not signaled with a high pulse on DCAN_SERR signal.
		All other values	SECDED single-bit error event is enabled, single-bit errors are signaled with a high pulse on DCAN_SERR signal.
23-20	Reserved	0	These bits are always read as 0. Writes have no effect.
19-16	ECCMODE		Enable SECDED single-bit error correction.
		5h	SECDED single-bit error correction is disabled.
		All other values	SECDED single-bit error correction is enabled.
15-9	Reserved	0	These bits are always read as 0. Writes have no effect.
8	DEFLG		Double-bit error flag.
		0	Read: No double-bit error is detected. Write: The bit is unchanged.
		1	Read: Double-bit error is detected. Write: The bit is cleared to 0.
7-1	Reserved	0	These bits are always read as 0. Writes have no effect.
0	SEFLG		Single-bit error flag.
		0	Read: No single-bit error is detected. Write: The bit is unchanged.
		1	Read: Single-bit error is detected. Write: The bit is cleared to 0.

27.17.12 ECC Single-Bit Error Code Register (DCAN ECC SERR)

If an ECC single-bit error is detected, the SEFLG flag is set in the ECC Control and Status Register. In addition to the SEFLG flag, the ECC Single-Bit Error Code Register indicates the memory area where the single-bit error has been detected (message object number only).

If more than one word with an ECC single-bit error is detected, the highest word number with an ECC single-bit error is displayed.

After an ECC single-bit error is detected, the register holds the last error code until power is removed.

Figure 27-31. ECC Single-Bit Error Code Register (DCAN ECC SERR) [offset = 30h]

31-16
Reserved
R-0

15-8	7-0
Reserved	Message Number
R-0	R-U

LEGEND: R = Read only; -*n* = value after reset; U = value is undefined

Table 27-18. ECC Single-Bit Error Code Register (DCAN ECC SERR) Field Descriptions

Bit	Field	Value	Description
31-8	Reserved	0	These bits are always read as 0. Writes have no effect.
7-0	Message Number	1h-FFh	Message object number where ECC single-bit error has been detected. Only values 1h-40h are valid. Values 41h-FFh are invalid.

27.17.13 Auto-Bus-On Time Register (DCAN ABOTR)

NOTE: On write access to the CAN Control register while Auto-Bus-On timer is running, the Auto-Bus-On procedure will be aborted.

During Debug/Suspend mode, running Auto-Bus-On timer will be paused.

Figure 27-32. Auto-Bus-On Time Register (DCAN ABOTR) [offset = 80h]

31 … 0
ABO_TIME
R/W-0

LEGEND: R/W = Read/Write; -*n* = value after reset

Table 27-19. Auto-Bus-On Time Register (DCAN ABOTR) Field Descriptions

Bit	Field	Description
31-0	ABO_TIME	Number of VBUS clock cycles before a Bus-Off recovery sequence is started by clearing the Init bit. This function has to be enabled by setting bit ABO in CAN Control Register. The Auto-Bus-On timer is realized by a 32-bit counter that starts to count down to 0 when the module goes Bus-Off. The counter will be reloaded with the preload value of the ABO_TIME register after this phase.

27.17.14 Transmission Request X Register (DCAN TXRQ X)

With the Transmission Request X Register, the CPU can detect if one or more bits in the different Transmission Request Registers are set. Each register bit represents a group of eight message objects. If at least one of the TxRqst bits of these message objects are set, the corresponding bit in the Transmission Request X Register will be set.

Figure 27-33. Transmission Request X Register (DCAN TXRQ X) [offset = 84h]

31 … 16
Reserved
R-0

15 14	13 12	11 10	9 8	7 6	5 4	3 2	1 0
TxRqstReg8	TxRqstReg7	TxRqstReg6	TxRqstReg5	TxRqstReg4	TxRqstReg3	TxRqstReg2	TxRqstReg1
R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0

LEGEND: R = Read only; -*n* = value after reset

Example 1

Bit 0 of the Transmission Request X Register represents byte 0 of the Transmission Request 1 Register. If one or more bits in this byte are set, bit 0 of the Transmission Request X Register will be set.

27.17.15 Transmission Request Registers (DCAN TXRQ12 to DCAN TXRQ78)

These registers hold the TxRqst bits of the implemented message objects. By reading out these bits, the CPU can check for pending transmission requests. The TxRqst bit in a specific message object can be set/reset by the CPU via the IF1/IF2 Message Interface Registers, or by the Message Handler after reception of a remote frame or after a successful transmission.

Figure 27-34. Transmission Request 12 Register (DCAN TXRQ12) [offset = 88h]

31 ... 0
TxRqst[32:1]
R-0

LEGEND: R = Read only; -*n* = value after reset

Figure 27-35. Transmission Request 34 Register (DCAN TXRQ34) [offset = 8Ch]

31 ... 0
TxRqst[64:33]
R-0

LEGEND: R = Read only; -*n* = value after reset

Figure 27-36. Transmission Request 56 Register (DCAN TXRQ56) [offset = 90h]

31 ... 0
TxRqst[96:65]
R-0

LEGEND: R = Read only; -*n* = value after reset

Figure 27-37. Transmission Request 78 Register (DCAN TXRQ78) [offset = 94h]

31 ... 0
TxRqst[128:97]
R-0

LEGEND: R = Read only; -*n* = value after reset

Table 27-20. Transmission Request Registers Field Descriptions

Bit	Name	Value	Description
31-0	TxRqst[128:1]		Transmission Request Bits (for all message objects).
		0	No transmission has been requested for this message object.
		1	The transmission of this message object is requested and is not yet done.

27.17.16 New Data X Register (DCAN NWDAT X)

With the New Data X Register, the CPU can detect if one or more bits in the different New Data Registers are set. Each register bit represents a group of eight message objects. If at least on of the NewDat bits of these message objects are set, the corresponding bit in the New Data X Register will be set.

Figure 27-38. New Data X Register (DCAN NWDAT X) [offset = 98h]

31	16
Reserved	
R-0	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NewDatReg8		NewDatReg7		NewDatReg6		NewDatReg5		NewDatReg4		NewDatReg3		NewDatReg2		NewDatReg1	
R-0		R-0		R-0		R-0		R-0		R-0		R-0		R-0	

LEGEND: R = Read only; -*n* = value after reset

Equation 1

Bit 0 of the New Data X Register represents byte 0 of the New Data 1 Register. If one or more bits in this byte are set, bit 0 of the New Data X Register will be set.

27.17.17 New Data Registers (DCAN NWDAT12 to DCAN NWDAT78)

These registers hold the NewDat bits of the implemented message objects. By reading out these bits, the CPU can check for new data in the message objects. The NewDat bit of a specific message object can be set/reset by the CPU via the IF1/IF2 Interface Register sets, or by the Message Handler after reception of a data frame or after a successful transmission.

Figure 27-39. New Data 12 Register (DCAN NWDAT12) [offset = 9Ch]

31 … 0
NewDat[32:1]
R-0

LEGEND: R = Read only; -*n* = value after reset

Figure 27-40. New Data 34 Register (DCAN NWDAT34) [offset = A0h]

31 … 0
NewDat[64:33]
R-0

LEGEND: R = Read only; -*n* = value after reset

Figure 27-41. New Data 56 Register (DCAN NWDAT56) [offset = A4h]

31 … 0
NewDat[96:65]
R-0

LEGEND: R = Read only; -*n* = value after reset

Figure 27-42. New Data 78 Register (DCAN NWDAT78) [offset = A8h]

31 … 0
NewDat[128:97]
R-0

LEGEND: R = Read only; -*n* = value after reset

Table 27-21. New Data Registers Field Descriptions

Bit	Name	Value	Description
31-0	NewDat[128:1]		New Data Bits (for all message objects).
		0	No new data has been written into the data portion of this message object by the Message Handler since the last time when this flag was cleared by the CPU.
		1	The Message Handler or the CPU has written new data into the data portion of this message object.

27.17.18 Interrupt Pending X Register (DCAN INTPND X)

With the Interrupt Pending X Register, the CPU can detect if one or more bits in the different Interrupt Pending Registers are set. Each bit of this register represents a group of eight message objects. If at least one of the IntPnd bits of these message objects are set, the corresponding bit in the Interrupt Pending X Register will be set.

Figure 27-43. Interrupt Pending X Register (DCAN INTPND X) [offset = ACh]

31–16
Reserved
R-0

15–14	13–12	11–10	9–8	7–6	5–4	3–2	1–0
IntPndReg8	IntPndReg7	IntPndReg6	IntPndReg5	IntPndReg4	IntPndReg3	IntPndReg2	IntPndReg1
R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0

LEGEND: R = Read only; -*n* = value after reset

Example 2

Bit 0 of the Interrupt Pending X Register represents byte 0 of the Interrupt Pending 1 Register. If one or more bits in this byte are set, bit 0 of the Interrupt Pending X Register will be set.

27.17.19 Interrupt Pending Registers (DCAN INTPND12 to DCAN INTPND78)

These registers hold the IntPnd bits of the implemented message objects. By reading out these bits, the CPU can check for pending interrupts in the message objects. The IntPnd bit of a specific message object can be set/reset by the CPU via the IF1/IF2 Interface Register sets, or by the Message Handler after a reception or a successful transmission.

Figure 27-44. Interrupt Pending 12 Register (DCAN INTPND12) [offset = B0h]

31	0
IntPnd[32:1]	
R-0	

LEGEND: R = Read only; -*n* = value after reset

Figure 27-45. Interrupt Pending 34 Register (DCAN INTPND34) [offset = B4h]

31	0
IntPnd[64:33]	
R-0	

LEGEND: R = Read only; -*n* = value after reset

Figure 27-46. Interrupt Pending 56 Register (DCAN INTPND56) [offset = B8h]

31	0
IntPnd[96:65]	
R-0	

LEGEND: R = Read only; -*n* = value after reset

Figure 27-47. Interrupt Pending 78 Register (DCAN INTPND78) [offset = BCh]

31	0
IntPnd[128:97]	
R-0	

LEGEND: R = Read only; -*n* = value after reset

Table 27-22. Interrupt Pending Registers Field Descriptions

Bit	Name	Value	Description
31-0	IntPnd[128:1]		Interrupt Pending Bits (for all message objects).
		0	This message object is not the source of an interrupt.
		1	This message object is the source of an interrupt.

27.17.20 Message Valid X Register (DCAN MSGVAL X)

With the Message Valid X Register, the CPU can detect if one or more bits in the different Message Valid Registers are set. Each bit of this register represents a group of eight message objects. If at least one of the MsgVal bits of these message objects are set, the corresponding bit in the Message Valid X Register will be set.

Figure 27-48. Message Valid X Register (DCAN MSGVAL X) [offset = C0h]

31	16
Reserved	
R-0	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MsgValReg8		MsgValReg7		MsgValReg6		MsgValReg5		MsgValReg4		MsgValReg3		MsgValReg2		MsgValReg1	
R-0		R-0		R-0		R-0		R-0		R-0		R-0		R-0	

LEGEND: R = Read only; -*n* = value after reset

Example 3

Bit 0 of the Message Valid X Register represents byte 0 of the Message Valid 1 Register. If one or more bits in this byte are set, bit 0 of the Message Valid X Register will be set.

27.17.21 Message Valid Registers (DCAN MSGVAL12 to DCAN MSGVAL78)

These registers hold the MsgVal bits of the implemented message objects. By reading out these bits, the CPU can check which message objects are valid. The MsgVal bit of a specific message object can be set/reset by the CPU via the IF1/IF2 Interface Register sets, or by the Message Handler after a reception or a successful transmission.

Figure 27-49. Message Valid 12 Register (DCAN MSGVAL12) [offset = C4h]

31	0
MsgVal[32:1]	
R-0	

LEGEND: R = Read only; -*n* = value after reset

Figure 27-50. Message Valid 34 Register (DCAN MSGVAL34) [offset = C8h]

31	0
MsgVal[64:33]	
R-0	

LEGEND: R = Read only; -*n* = value after reset

Figure 27-51. Message Valid 56 Register (DCAN MSGVAL56) [offset = CCh]

31	0
MsgVal[96:65]	
R-0	

LEGEND: R = Read only; -*n* = value after reset

Figure 27-52. Message Valid 78 Register (DCAN MSGVAL78) [offset = D0h]

31	0
MsgVal[128:97]	
R-0	

LEGEND: R = Read only; -*n* = value after reset

Table 27-23. Message Valid Registers Field Descriptions

Bit	Name	Value	Description
31-0	MsgVal[128:1]		Message Valid Bits (for all message objects).
		0	This message object is ignored by the Message Handler.
		1	This message object is configured and will be considered by the Message Handler.

27.17.22 Interrupt Multiplexer Registers (DCAN INTMUX12 to DCAN INTMUX78)

The IntMux flag determines for each message object which of the two interrupt lines (DCAN0INT or DCAN1INT) will be asserted when the IntPnd of this message object is set. Both interrupt lines can be globally enabled or disabled by setting or clearing IE0 and IE1 bits in CAN Control Register.

The IntPnd bit of a specific message object can be set or reset by the CPU via the IF1/IF2 Interface Register sets, or by Message Handler after reception or successful transmission of a frame. This will also affect the Int0ID resp Int1ID flags in the Interrupt Register.

Figure 27-53. Interrupt Multiplexer 12 Register (DCAN INTMUX12) [offset = D8h]

31 ... 0
IntMux[32:1]
R/W-0

LEGEND: R/W = Read/Write; -*n* = value after reset

Figure 27-54. Interrupt Multiplexer 34 Register (DCAN INTMUX34) [offset = DCh]

31 ... 0
IntMux[64:33]
R/W-0

LEGEND: R/W = Read/Write; -*n* = value after reset

Figure 27-55. Interrupt Multiplexer 56 Register (DCAN INTMUX56) [offset = E0h]

31 ... 0
IntMux[96:65]
R/W-0

LEGEND: R/W = Read/Write; -*n* = value after reset

Figure 27-56. Interrupt Multiplexer 78 Register (DCAN INTMUX78) [offset = E4h]

31 ... 0
IntMux[128:97]
R/W-0

LEGEND: R/W = Read/Write; -*n* = value after reset

Table 27-24. Interrupt Multiplexer Registers Field Descriptions

Bit	Name	Value	Description
31-0	IntMux[128:1]		Multiplexes IntPnd value to either DCAN0INT or DCAN1INT interrupt lines. The mapping from the bits to the message objects is as follows: Bit 0 -> last implemented message object. Bit 1 -> message object number 1 Bit 2 -> message object number 2
		0	DCAN0INT line is active if corresponding IntPnd flag is 1.
		1	DCAN1INT line is active if corresponding IntPnd flag is 1.

27.17.23 IF1/IF2 Command Registers (DCAN IF1CMD, DCAN IF2CMD)

The IF1/IF2 Command Register configure and Initiate the transfer between the IF1/IF2 Register sets and the Message RAM. It is configurable which portions of the message object should be transferred.

A transfer is started when the CPU writes the message number to bits [7:0] of the IF1/IF2 Command Register. With this write operation, the Busy bit is automatically set to 1 to indicate that a transfer is in progress.

After 4 to 14 VBUS clock cycles, the transfer between the Interface Register and the Message RAM will be completed and the Busy bit is cleared. The maximum number of cycles is needed when the message transfer concurs with a CAN message transmission, acceptance filtering, or message storage.

If the CPU writes to both IF1/IF2 Command Registers consecutively (request of a second transfer while first transfer is still in progress), the second transfer will start after the first one has been completed.

NOTE: While Busy bit is one, IF1/IF2 Register sets are write protected.

For debug support, the auto clear functionality of the IF1/IF2 Command Registers (clear of DMAactive flag by r/w) is disabled during Debug/Suspend mode.

If an invalid Message Number is written to bits [7:0] of the IF1/IF2 Command Register, the Message Handler may access an implemented (valid) message object instead.

Figure 27-57. IF1 Command Registers (DCAN IF1CMD) [offset = 100h]

31	24
Reserved	
R-0	

23	22	21	20	19	18	17	16
WR/RD	Mask	Arb	Control	ClrIntPnd	TxRqst/NewDat	Data A	Data B
R/WP-0	R/WP-0	R/WP-0	R/WP-0	R/WP-0	R/WP-0	R/WP-0	R/WP-0

15	14	13	8
Busy	DMA Active	Reserved	
R-0	R/WP/C-0	R-0	

7	0
Message Number	
R/WP-1h	

LEGEND: R/W = Read/Write; R = Read; WP = Protected Write (protected by Busy bit); C = Clear by IF1 Access; -*n* = value after reset

Figure 27-58. IF2 Command Registers (DCAN IF2CMD) [offset = 120h]

31	24
Reserved	
R-0	

23	22	21	20	19	18	17	16
WR/RD	Mask	Arb	Control	ClrIntPnd	TxRqst/NewDat	Data A	Data B
R/WP-0	R/WP-0	R/WP-0	R/WP-0	R/WP-0	R/WP-0	R/WP-0	R/WP-0

15	14	13	8
Busy	DMA Active	Reserved	
R-0	R/WP/C-0	R-0	

7	0
Message Number	
R/WP-1h	

LEGEND: R/W = Read/Write; R = Read; WP = Protected Write (protected by Busy bit); C = Clear by IF1 Access; -*n* = value after reset

Table 27-25. IF1/IF2 Command Register Field Descriptions

Bit	Field	Value	Description
31-24	Reserved	0	These bits are always read as 0. Writes have no effect.
23	WR/RD		Write/Read
		0	Direction = Read: Transfer direction is from the message object addressed by Message Number (Bits [7:0]) to the IF1/IF2 register set.
		1	Direction = Write: Transfer direction is from the IF1/IF2 register set to the message object addressed by Message Number (Bits [7:0]).
22	Mask		Access Mask bits.
		0	Mask bits will not be changed.
		1	Direction = Read: The Mask bits (Identifier Mask + MDir + MXtd) will be transferred from the message object addressed by Message Number (Bits [7:0]) to the IF1/IF2 Register set. Direction = Write: The Mask bits (Identifier Mask + MDir + MXtd) will be transferred from the IF1/IF2 Register set to the message object addressed by Message Number (Bits [7:0]).
21	Arb		Access Arbitration bits.
		0	Arbitration bits will not be changed.
		1	Direction = Read: The Arbitration bits (Identifier + Dir + Xtd + MsgVal) will be transferred from the message object addressed by Message Number (Bits [7:0]) to the corresponding IF1/IF2 Register set. Direction = Write: The Arbitration bits (Identifier + Dir + Xtd + MsgVal) will be transferred from the IF1/IF2 Register set to the message object addressed by Message Number (Bits [7:0]).
20	Control		Access Control bits.
		0	Control bits will not be changed.
		1	Direction = Read: The Message Control bits will be transferred from the message object addressed by Message Number (Bits [7:0]) to the corresponding IF1/IF2 Register set. Direction = Write: The Message Control bits will be transferred from the IF1/IF2 Register set to the message object addressed by Message Number (Bits [7:0]). If the TxRqst/NewDat bit in this register (Bit [18]) is set, the TxRqst/NewDat bit in the IF1/IF2 Message Control Register will be ignored.
19	ClrIntPnd		Clear Interrupt Pending bit.
		0	IntPnd bit will not be changed.
		1	Direction = Read: Clears IntPnd bit in the message object. Direction = Write: This bit is ignored. Copying of IntPnd flag from IF1/IF2 Registers to Message RAM can be controlled by only the Control flag (Bit [20]).
18	TxRqst/NewDat		Access Transmission Request bit.
		0	Direction = Read: NewDat bit will not be changed. Direction = Write: TxRqst/NewDat bit will be handled according to the Control bit.
		1	Direction = Read: Clears NewDat bit in the message object. Direction = Write: Sets TxRqst/NewDat in the message object. **Note:** If a CAN transmission is requested by setting TxRqst/NewDat in this register, the TxRqst/NewDat bits in the message object will be set to 1 and independent of the values in IF1/IF2 Message Control Register. A read access to a message object can be combined with the reset of the control bits IntPnd and NewDat. The values of these bits transferred to the IF1/IF2 Message Control Register always reflect the status before resetting them.
17	Data A		Access Data Bytes 0–3.
		0	Data Bytes 0–3 will not be changed.
		1	Direction = Read: The Data Bytes 0–3 will be transferred from the message object addressed by the Message Number (Bits [7:0]) to the corresponding IF1/IF2 Register set. Direction = Write: The Data Bytes 0–3 will be transferred from the IF1/IF2 Register set to the message object addressed by the Message Number (Bits [7:0]). **Note:** The duration of the message transfer is independent of the number of bytes to be transferred.

Table 27-25. IF1/IF2 Command Register Field Descriptions (continued)

Bit	Field	Value	Description
16	Data B		Access Data Bytes 4–7.
		0	Data Bytes 4–7 will not be changed.
		1	Direction = Read: The Data Bytes 4–7 will be transferred from the message object addressed by the Message Number (Bits [7:0]) to the corresponding IF1/IF2 Register set. Direction = Write: The Data Bytes 4–7 will be transferred from the IF1/IF2 Register set to the message object addressed by the Message Number (Bits [7:0]). **Note:** The duration of the message transfer is independent of the number of bytes to be transferred.
15	Busy		Busy flag.
		0	No transfer between IF1/IF2 Register set and Message RAM is in progress.
		1	Transfer between IF1/IF2 Register set and Message RAM is in progress. This bit is set to 1 after the message number has been written to bits [7:0]. IF1/IF2 Register set will be write-protected. The bit is cleared after read/write action has finished.
14	DMA Active		Activation of DMA feature for subsequent internal IF1/IF2 update.
		0	DMA request line is independent of IF1/IF2 activities.
		1	DMA is requested after completed transfer between IF1/IF2 Register set and Message RAM. The DMA request remains active until the first read or write to one of the IF1/IF2 registers. An exception is a write to Message Number (Bits [7:0]) when DMA Active is 1. **Note:** Due to the auto reset feature of the DMA Active bit, this bit has to be separately set for each subsequent DMA cycle.
13-8	Reserved	0	These bits are always read as 0. Writes have no effect.
7-0	Message Number		Number of message object in Message RAM that is used for data transfer.
		0	Invalid message number.
		1h-40h	Valid message numbers.
		41h-FFh	Invalid message numbers. **Note:** When an invalid message number is written to the IF1/IF2 Command Register that is higher than the last implemented message object number, a modulo addressing will occur. For example, when accessing message object 33 in a DCAN module with 32 message objects only, the message object 1 will be accessed instead.

27.17.24 IF1/IF2 Mask Registers (DCAN IF1MSK, DCAN IF2MSK)

The bits of the IF1/IF2 Mask Registers mirror the mask bits of a message object. The function of the relevant message objects bits is described in Section 27.5.1.

NOTE: While Busy bit of IF1/IF2 Command Register is one, IF1/IF2 Register Set is write protected.

Figure 27-59. IF1 Mask Register (DCAN IF1MSK) [offset = 104h]

31	30	29	28 16
MXtd	MDir	Rsvd	Msk[28:16]
R/WP-1	R/WP-1	R-1	R/WP-1FFFh

15 0
Msk[15:0]
R/WP-FFFFh

LEGEND: R/W = Read/Write; R = Read; WP = Protected Write (protected by Busy bit); -*n* = value after reset

Figure 27-60. IF2 Mask Register (DCAN IF2MSK) [offset = 124h]

31	30	29	28 16
MXtd	MDir	Rsvd	Msk[28:16]
R/WP-1	R/WP-1	R-1	R/WP-1FFFh

15 0
Msk[15:0]
R/WP-FFFFh

LEGEND: R/W = Read/Write; R = Read; WP = Protected Write (protected by Busy bit); -*n* = value after reset

Table 27-26. IF1/IF2 Mask Register Field Descriptions

Bit	Field	Value	Description
31	MXtd		Mask extended identifier.
		0	The extended identifier bit (IDE) has no effect on the acceptance filtering.
		1	The extended identifier bit (IDE) is used for acceptance filtering.
			When 11-bit ("standard") identifiers are used for a message object, the identifiers of received Data Frames are written into bits ID[28:18]. For acceptance filtering, only these bits with mask bits Msk[28:18] are considered.
30	MDir		Mask message direction.
		0	The message direction bit (Dir) has no effect on the acceptance filtering.
		1	The message direction bit (Dir) is used for acceptance filtering.
29	Reserved	0	These bits are always read as 1. Writes have no effect.
28-0	Msk[*n*]		Identifier mask.
		0	The corresponding bit in the identifier of the message object is not used for acceptance filtering (don't care).
		1	The corresponding bit in the identifier of the message object is used for acceptance filtering.

27.17.25 IF1/IF2 Arbitration Registers (DCAN IF1ARB, DCAN IF2ARB)

The bits of the IF1/IF2 Arbitration Registers mirror the arbitration bits of a message object. The function of the relevant message objects bits is described in Section 27.5.1.

The Arbitration bits ID, Xtd, and Dir are used to define the identifier and type of outgoing messages and (together with the Mask bits Msk, MXtd, and MDir) for acceptance filtering of incoming messages.

A received message is stored into the valid message object with matching identifier and Direction = receive (Data Frame) or Direction = transmit (Remote Frame).

Extended frames can be stored only in message objects with Xtd = 1, standard frames in message objects with Xtd = 0.

If a received message (Data Frame or Remote Frame) matches more than one valid message objects, it is stored into the one with the lowest message number.

NOTE: While Busy bit of IF1/IF2 Command Register is one, IF1/IF2 Register Set is write protected.

Figure 27-61. IF1 Arbitration Register (DCAN IF1ARB) [offset = 108h]

31	30	29	28 16
MsgVal	Xtd	Dir	ID[28:16]
R/WP-0	R/WP-0	R/WP-0	R/WP-0

15 0
ID[15:0]
R/WP-0

LEGEND: R/W = Read/Write; WP = Protected Write (protected by Busy bit); -*n* = value after reset

Figure 27-62. IF2 Arbitration Register (DCAN IF2ARB) [offset = 128h]

31	30	29	28 16
MsgVal	Xtd	Dir	ID[28:16]
R/WP-0	R/WP-0	R/WP-0	R/WP-0

15 0
ID[15:0]
R/WP-0

LEGEND: R/W = Read/Write; WP = Protected Write (protected by Busy bit); -*n* = value after reset

Table 27-27. IF1/IF2 Arbitration Register Field Descriptions

Bit	Field	Value	Description
31	MsgVal		Message valid
		0	The message object is ignored by the Message Handler.
		1	The message object is used by the Message Handler.
			Note: The CPU should reset the MsgVal bit of all unused Messages Objects during the initialization before it resets bit Init in the CAN Control Register. MsgVal must also be reset if the messages object is no longer used in operation. For reconfiguration of message objects during normal operation, see Section 27.7.6 and Section 27.7.7.
30	Xtd		Extended identifier.
		0	The 11-bit ("standard") identifier is used for this message object.
		1	The 29-bit ("extended") identifier is used for this message object.
29	Dir		Message direction.
		0	Direction = Receive: On TxRqst, a Remote Frame with the identifier of this message object is transmitted. On receiving a Data Frame with a matching identifier, this message is stored in this message object.
		1	Direction = Transmit: On TxRqst, the respective message object is transmitted as a Data Frame. On receiving a Remote Frame with a matching identifier, the TxRqst bit of this message object is set (if RmtEn = 1).
28-0	ID		Message identifier.
		ID[28:0]	29-bit Identifier ("Extended Frame").
		ID[28:18]	11-bit Identifier ("Standard Frame").

27.17.26 IF1/IF2 Message Control Registers (DCAN IF1MCTL, DCAN IF2MCTL)

The bits of the IF1/IF2 Message Control Registers mirror the message control bits of a message object. The function of the relevant message objects bits is described in Section 27.5.1.

NOTE: While Busy bit of IF1/IF2 Command Register is one, IF1/IF2 Register Set is write protected.

Figure 27-63. IF1 Message Control Register (DCAN IF1MCTL) [offset = 10Ch]

31	16
Reserved	
R-0	

15	14	13	12	11	10	9	8
NewDat	MsgLst	IntPnd	UMask	TxIE	RxIE	RmtEn	TxRqst
R/WP-0	R/WP-0	R/WP-0	R/WP-0	R/WP-0	R/WP-0	R/WP-0	R/WP-0

7	6	4	3	0
EoB	Reserved		DLC	
R/WP-0	R-0		R/WP-0	

LEGEND: R/W = Read/Write; R = Read; WP = Protected Write (protected by Busy bit); -*n* = value after reset

Figure 27-64. IF2 Message Control Register (DCAN IF2MCTL) [offset = 12Ch]

31	16
Reserved	
R-0	

15	14	13	12	11	10	9	8
NewDat	MsgLst	IntPnd	UMask	TxIE	RxIE	RmtEn	TxRqst
R/WP-0	R/WP-0	R/WP-0	R/WP-0	R/WP-0	R/WP-0	R/WP-0	R/WP-0

7	6	4	3	0
EoB	Reserved		DLC	
R/WP-0	R-0		R/WP-0	

LEGEND: R/W = Read/Write; R = Read; WP = Protected Write (protected by Busy bit); -*n* = value after reset

Table 27-28. IF1/IF2 Message Control Register Field Descriptions

Bit	Field	Value	Description
31-16	Reserved	0	These bits are always read as 0. Writes have no effect.
15	NewDat		New Data
		0	No new data has been written into the data portion of this message object by the Message Handler since the last time this flag was cleared by the CPU.
		1	The Message Handler or the CPU has written new data into the data portion of this message object.
14	MsgLst		Message Lost (only valid for message objects with direction = receive).
		0	No message lost since the last time when this bit was reset by the CPU.
		1	The Message Handler stored a new message into this object when NewDat was still set, so the previous message has been overwritten.
13	IntPnd		Interrupt Pending
		0	This message object is not the source of an interrupt.
		1	This message object is the source of an interrupt. The interrupt identifier in the interrupt register will point to this message object if there is no other interrupt source with higher priority.
12	UMask		Use Acceptance Mask
		0	Mask is ignored.
		1	Use Mask (Msk[28:0], MXtd, and MDir) for acceptance filtering.
			If the UMask bit is set to 1, the message object's mask bits have to be programmed during initialization of the message object before MsgVal is set to 1.
11	TxIE		Transmit interrupt enable.
		0	IntPnd will not be triggered after the successful transmission of a frame.
		1	IntPnd will be triggered after the successful transmission of a frame.
10	RxIE		Receive interrupt enable.
		0	IntPnd will not be triggered after the successful reception of a frame.
		1	IntPnd will be triggered after the successful reception of a frame.
9	RmtEn		Remote enable.
		0	At the reception of a Remote Frame, TxRqst is not changed.
		1	At the reception of a Remote Frame, TxRqst is set.
8	TxRqst		Transmit request.
		0	This message object is not waiting for a transmission.
		1	The transmission of this message object is requested and not yet done.
7	EoB		End of Block
		0	The message object is part of a FIFO Buffer block and is not the last message object of this FIFO Buffer block.
		1	The message object is a single message object or the last message object in a FIFO Buffer block.
			Note: This bit is used to concatenate multiple message objects to build a FIFO Buffer. For single message objects (not belonging to a FIFO Buffer), this bit must always be set to 1.
6-4	Reserved	0	These bits are always read as 0. Writes have no effect.
3-0	DLC		Data Length Code
		0-8h	Data Frame has 0-8 data bits.
		9h-Fh	Data Frame has 8 data bytes.
			Note: The Data Length Code of a message object must be defined the same as in all the corresponding objects with the same identifier at other nodes. When the message handler stores a data frame, it will write the DLC to the value given by the received message.

27.17.27 IF1/IF2 Data A and Data B Registers (DCAN IF1DATA/DATB, DCAN IF2DATA/DATB)

The data bytes of CAN messages are stored in the IF1/IF2 registers in the following order.

In a CAN Data Frame, Data 0 is the first, and Data 7 is the last byte to be transmitted or received. In CAN's serial bit stream, the MSB of each byte will be transmitted first

Figure 27-65. IF1 Data A Register (DCAN IF1DATA) [offset = 110h]

31	24	23	16
Data 3		Data 2	
R/WP-0		R/WP-0	
15	**8**	**7**	**0**
Data 1		Data 0	
R/WP-0		R/WP-0	

LEGEND: R/W = Read/Write; WP = Protected Write (protected by Busy bit); -*n* = value after reset

Figure 27-66. IF1 Data B Register (DCAN IF1DATB) [offset = 114h]

31	24	23	16
Data 7		Data 6	
R/WP-0		R/WP-0	
15	**8**	**7**	**0**
Data 5		Data 4	
R/WP-0		R/WP-0	

LEGEND: R/W = Read/Write; WP = Protected Write (protected by Busy bit); -*n* = value after reset

Figure 27-67. IF2 Data A Register (DCAN IF2DATA) [offset = 130h]

31	24	23	16
Data 3		Data 2	
R/WP-0		R/WP-0	
15	**8**	**7**	**0**
Data 1		Data 0	
R/WP-0		R/WP-0	

LEGEND: R/W = Read/Write; WP = Protected Write (protected by Busy bit); -*n* = value after reset

Figure 27-68. IF2 Data B Register (DCAN IF2DATB) [offset = 134h]

31	24	23	16
Data 7		Data 6	
R/WP-0		R/WP-0	
15	**8**	**7**	**0**
Data 5		Data 4	
R/WP-0		R/WP-0	

LEGEND: R/W = Read/Write; WP = Protected Write (protected by Busy bit); -*n* = value after reset

27.17.28 IF3 Observation Register (DCAN IF3OBS)

The IF3 register set can automatically be updated with received message objects without the need to Initiate the transfer from Message RAM by CPU (Additional information can be found in Section 27.5.1).

The observation flags (Bits [4:0]) in the IF3 Observation register are used to determine, which data sections of the IF3 Interface Register set have to be read in order to complete a DMA read cycle. After all marked data sections are read, the DCAN is enabled to update the IF3 Interface Register set with new data.

Any access order of single bytes or half-words is supported. When using byte or half-word accesses, a data section is marked as completed, if all bytes are read.

NOTE: If IF3 Update Enable is used and no Observation flag is set, the corresponding message objects will be copied to IF3 without activating the DMA request line and without waiting for DMA read accesses.

A write access to this register aborts a pending DMA cycle by resetting the DMA line and enables updating of IF3 Interface Register set with new data. To avoid data inconsistency, the DMA controller should be disabled before reconfiguring IF3 observation register.

The status of the current read-cycle can be observed via status flags (Bits [12:8]).

An interrupt request may be generated by the IF3Upd flag if the DE3 bit of DCAN CTL register is set. See the device data sheet to find out if this interrupt source is available.

With this, the observation status bits and the IF3Upd bit could be used by the application to realize the notification about new IF3 content in polling or interrupt mode.

Figure 27-69. IF3 Observation Register (DCAN IF3OBS) [offset = 140h]

31 16
Reserved
R-0

15	14 13	12	11	10	9	8	7 5	4	3	2	1	0
IF3Upd	Reserved	IF3SDB	IF3SDA	IF3SC	IF3SA	IF3SM	Reserved	Data B	Data A	Ctrl	Arb	Mask
R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

LEGEND: R/W = Read/Write; R = Read; -*n* = value after reset

Table 27-29. IF3 Observation Register (DCAN IF3OBS) Field Descriptions

Bit	Field	Value	Description
31-16	Reserved	0	These bits are always read as 0. Writes have no effect.
15	IF3 Upd		IF3 Update Data.
		0	No new data has been loaded since IF3 was last read.
		1	New data has been loaded since IF3 was last read.
14-13	Reserved	0	These bits are always read as 0. Writes have no effect
12	IF3 SDB		IF3 Status of Data B read access.
		0	All Data B bytes are already read or are not marked to be read.
		1	Data B section still has data to read.
11	IF3 SDA		IF3 Status of Data A read access.
		0	All Data A bytes are already read or are not marked to be read.
		1	Data A section still has data to read.
10	IF3 SC		IF3 Status of Control bits read access.
		0	All Control section bytes are already read or are not marked to be read.
		1	Control section still has data to read.

Table 27-29. IF3 Observation Register (DCAN IF3OBS) Field Descriptions (continued)

Bit	Field	Value	Description
9	IF3 SA		IF3 Status of Arbitration data read access.
		0	All Arbitration data bytes are already read or are not marked to be read.
		1	Arbitration section still has data to read.
8	IF3 SM		IF3 Status of Mask data read access.
		0	All Mask data bytes are already read or are not marked to be read.
		1	Mask section still has data to read.
7-5	Reserved	0	These bits are always read as 0. Writes have no effect
4	Data B		Data B read observation.
		0	Data B section does not need to be read.
		1	Data B section has to be read to enable next IF3 update.
3	Data A		Data A read observation.
		0	Data A section does not need to be read.
		1	Data A section has to be read to enable next IF3 update.
2	Ctrl		Ctrl read observation.
		0	Ctrl section does not need to be read.
		1	Ctrl section has to be read to enable next IF3 update.
1	Arb		Arbitration data read observation.
		0	Arbitration data does not need to be read.
		1	Arbitration data has to be read to enable next IF3 update.
0	Mask		Mask data read observation.
		0	Mask data does not need to be read.
		1	Mask data has to be read to enable next IF3 update.

27.17.29 IF3 Mask Register (DCAN IF3MSK)

Figure 27-70. IF3 Mask Register (DCAN IF3MSK) [offset = 144h]

31	30	29	28	16
MXtd	MDir	Rsvd	Msk[28:16]	
R-1	R-1	R-1	R-1FFFh	

15	0
Msk[15:0]	
R-FFFFh	

LEGEND: R = Read; -*n* = value after reset

Table 27-30. IF3 Mask Register (DCAN IF3MSK) Field Descriptions

Bit	Field	Value	Description
31	MXtd		Mask extended identifier.
		0	The extended identifier bit (IDE) has no effect on acceptance filtering.
		1	The extended identifier bit (IDE) is used for acceptance filtering.
			Note: When 11-bit ("standard") identifiers are used for a message object, the identifiers of received Data Frames are written into bits ID[28:18]. For acceptance filtering, only these bits, together with mask bits Msk[28:18], are considered.
30	MDir		Mask message direction.
		0	The message direction bit (Dir) has no effect on acceptance filtering.
		1	The message direction bit (Dir) is used for acceptance filtering.
29	Reserved	0	These bits are always read as 0. Writes have no effect.
28-0	Msk[*n*]		Identifier mask.
		0	The corresponding bit in the identifier of the message object is not used for acceptance filtering (don't care).
		1	The corresponding bit in the identifier of the message object is used for acceptance filtering.

27.17.30 IF3 Arbitration Register (DCAN IF3ARB)

Figure 27-71. IF3 Arbitration Register (DCAN IF3ARB) [offset = 148h]

31	30	29	28	16
MsgVal	Xtd	Dir	ID[28:16]	
R-0	R-0	R-0	R-0	

15	0
ID[15:0]	
R-0	

LEGEND: R = Read; -*n* = value after reset

Table 27-31. IF3 Arbitration Register (DCAN IF3ARB) Field Descriptions

Bit	Field	Value	Description
31	MsgVal		Message valid.
		0	The message object is ignored by the Message Handler.
		1	The message object is to be used by the Message Handler.
			Note: The CPU should reset the MsgVal bit of all unused Messages Objects during the initialization before it resets bit Init in the CAN Control Register. MsgVal must also be reset if the messages object is no longer used in operation. For reconfiguration of message objects during normal operation, see Section 27.7.6 and Section 27.7.7.
30	Xtd		Extended identifier.
		0	The 11-bit ("standard") identifier is used for this message object.
		1	The 29-bit ("extended") identifier is used for this message object.
29	Dir		Message direction.
		0	Direction = Receive: On TxRqst, a remote frame with the identifier of this message object is transmitted. On receiving a data frame with a matching identifier, the message is stored in this message object.
		1	Direction = Transmit: On TxRqst, the respective message object is transmitted as a data frame. On receiving a remote frame with a matching identifier, the TxRqst bit of this message object is set (if RmtEn = 1).
28-0	ID		Message identifier.
		ID[28:0]	29-bit Identifier ("Extended Frame").
		ID[28:18]	11-bit Identifier ("Standard Frame").

27.17.31 IF3 Message Control Register (DCAN IF3MCTL)

Figure 27-72. IF3 Message Control Register (DCAN IF3MCTL) [offset = 14Ch]

31 … 16
Reserved
R-0

15	14	13	12	11	10	9	8
NewDat	MsgLst	IntPnd	UMask	TxIE	RxIE	RmtEn	TxRqst
R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0

7	6 … 4	3 … 0
EoB	Reserved	DLC
R-0	R-0	R-0

LEGEND: R = Read; -*n* = value after reset

Table 27-32. IF3 Message Control Register (DCAN IF3MCTL) Field Descriptions

Bit	Field	Value	Description
31-16	Reserved	0	These bits are always read as 0. Writes have no effect.
15	NewDat		New Data
		0	No new data has been written into the data portion of this message object by the Message Handler since the last time this flag was cleared by the CPU.
		1	The Message Handler or the CPU has written new data into the data portion of this message object.
14	MsgLst		Message Lost (only valid for message objects with direction = receive).
		0	No message lost since the last time when this bit was reset by the CPU.
		1	The Message Handler stored a new message into this object when NewDat was still set, so the previous message has been overwritten.
13	IntPnd		Interrupt Pending.
		0	This message object is not the source of an interrupt.
		1	This message object is the source of an interrupt. The interrupt identifier in the interrupt register will point to this message object if there is no other interrupt source with higher priority.
12	UMask		Use Acceptance Mask.
		0	Mask is ignored.
		1	Use Mask (Msk[28:0], MXtd, and MDir) for acceptance filtering.
			If the UMask bit is set to 1, the message object's mask bits have to be programmed during initialization of the message object before MsgVal is set to 1.
11	TxIE		Transmit interrupt enable.
		0	IntPnd will not be triggered after the successful transmission of a frame.
		1	IntPnd will be triggered after the successful transmission of a frame.
10	RxIE		Receive interrupt enable.
		0	IntPnd will not be triggered after the successful transmission of a frame.
		1	IntPnd will be triggered after the successful transmission of a frame.
9	RmtEn		Remote enable.
		0	At the reception of a Remote Frame, TxRqst is not changed.
		1	At the reception of a Remote Frame, TxRqst is set.
8	TxRqst		Transmit request.
		0	This message object is not waiting for a transmission.
		1	The transmission of this message object is requested and not yet done.

Table 27-32. IF3 Message Control Register (DCAN IF3MCTL) Field Descriptions (continued)

Bit	Field	Value	Description
7	EoB		End of Block
		0	The message object is part of a FIFO Buffer block and is not the last message object of the FIFO Buffer block.
		1	The message object is a single message object or the last message object in a FIFO Buffer block.
			Note: This bit is used to concatenate multiple message objects to build a FIFO Buffer. For single message objects (not belonging to a FIFO Buffer), this bit must always be set to 1.
6-4	Reserved	0	These bits are always read as 0. Writes have no effect.
3-0	DLC		Data Length Code
		0-8h	Data Frame has 0-8 data bits.
		9h-Fh	Data Frame has 8 data bytes.
			Note: The Data Length Code of a message object must be defined the same as in all the corresponding objects with the same identifier at other nodes. When the message handler stores a data frame, it will write the DLC to the value given by the received message.

27.17.32 IF3 Data A and Data B Registers (DCAN IF3DATA/DATB)

The data bytes of CAN messages are stored in the IF3 registers in the following order.

In a CAN Data Frame, Data 0 is the first, and Data 7 is the last byte to be transmitted or received. In CAN's serial bit stream, the MSB of each byte will be transmitted first.

Figure 27-73. IF3 Data A Register (DCAN IF3DATA) [offset = 150h]

31-24	23-16
Data 3	Data 2
R-0	R-0

15-8	7-0
Data 1	Data 0
R-0	R-0

LEGEND: R = Read; -*n* = value after reset

Figure 27-74. IF3 Data B Register (DCAN IF3DATB) [offset = 154h]

31-24	23-16
Data 7	Data 6
R/WP-0	R/WP-0

15-8	7-0
Data 5	Data 4
R-0	R-0

LEGEND: R = Read; -*n* = value after reset

27.17.33 IF3 Update Enable Registers (DCAN IF3UPD12 to DCAN IF3UPD78)

The automatic update functionality of the IF3 register set can be configured for each message object. A message object is enabled for automatic IF3 update, if the dedicated IF3UpdEn flag is set. This means that an active NewDat flag of this message object (for example, due to reception of a CAN frame) will trigger an automatic copy of the whole message object to IF3 register set.

NOTE: IF3 Update enable should not be set for transmit objects.

Figure 27-75. IF3 Update Enable 12 Register (DCAN IF3UPD12) [offset = 160h]

31 … 0
IF3UpdEn[32:1]
R/W-0

LEGEND: R/W = Read/Write; -*n* = value after reset

Figure 27-76. IF3 Update Enable 34 Register (DCAN IF3UPD34) [offset = 164h]

31 … 0
IF3UpdEn[64:33]
R/W-0

LEGEND: R/W = Read/Write; -*n* = value after reset

Figure 27-77. IF3 Update Enable 56 Register (DCAN IF3UPD56) [offset = 168h]

31 … 0
IF3UpdEn[96:65]
R/W-0

LEGEND: R/W = Read/Write; -*n* = value after reset

Figure 27-78. IF3 Update Enable 78 Register (DCAN IF3UPD78) [offset = 16Ch]

31 … 0
IF3UpdEn[128:97]
R/W-0

LEGEND: R/W = Read/Write; -*n* = value after reset

Table 27-33. IF3 Update Control Register Field Descriptions

Bit	Name	Value	Description
31-0	IF3UpdEn[128:1]		IF3 Update Enabled (for all message objects).
		0	Automatic IF3 update is disabled for this message object.
		1	Automatic IF3 update is enabled for this message object. A message object is scheduled to be copied to IF3 register set, if NewDat flag of the message object is active.

27.17.34 CAN TX IO Control Register (DCAN TIOC)

The CAN_TX pin of the DCAN module can be used as general-purpose IO pin if CAN function is not needed.

NOTE: The values of the IO Control registers are only writable if Init bit of CAN Control Register is set.

The OD, Func, Dir, and Out bits of the CAN TX IO Control register are forced to certain values when Init bit of CAN Control Register is reset (see bit descriptions).

Figure 27-79. CAN TX IO Control Register (DCAN TIOC) [offset = 1E0h]

31-19	18	17	16
Reserved	PU	PD	OD
R-0	R/W-D	R/W-D	R/WP-0

15-4	3	2	1	0
Reserved	Func	Dir	Out	In
R-0	R/WP-0	R/WP-0	R/WP-0	R-U

LEGEND: R/W = Read/Write; R = Read; WP = Protected Write (protected by Init bit); D = Device-dependent; -*n* = value after reset

Table 27-34. CAN TX IO Control Register (DCAN TIOC) Field Descriptions

Bit	Field	Value	Description
31-19	Reserved	0	These bits are always read as 0. Writes have no effect.
18	PU		CAN_TX Pullup/Pulldown select. This bit is only active when CAN_TX is configured to be an input.
		0	CAN_TX Pulldown is selected, when pull logic is active (PD = 0).
		1	CAN_TX Pullup is selected, when pull logic is active (PD = 0).
17	PD		CAN_TX pull disable. This bit is only active when CAN_TX is configured to be an input.
		0	CAN_TX pull is active.
		1	CAN_TX pull is disabled.
16	OD		CAN_TX open drain enable. This bit is only active when CAN_TX is configured to be in GIO mode (TIOC.Func = 0).
		0	The CAN_TX pin is configured in push/pull mode.
		1	The CAN_TX pin is configured in open drain mode.
			Forced to 0, if Init bit of CAN control register is reset.
15-4	Reserved	0	These bits are always read as 0. Writes have no effect.
3	Func		CAN_TX function. This bit changes the function of the CAN_TX pin.
		0	CAN_TX pin is in GIO mode.
		1	CAN_TX pin is in functional mode (as an output to transmit CAN data).
			Forced to 1, if Init bit of CAN control register is reset.
2	Dir		CAN_TX data direction. This bit controls the direction of the CAN_TX pin when it is configured to be in GIO mode only (TIOC.Func = 0).
		0	The CAN_TX pin is an input.
		1	The CAN_TX pin is an output.
			Forced to 1, if Init bit of CAN control register is reset.
1	Out		CAN_TX data out write. This bit is only active when CAN_TX pin is configured to be in GIO mode (TIOC.Func = 0) and configured to be an output pin (TIOC.Dir = 1). The value of this bit indicates the value to be output to the CAN_TX pin.
		0	The CAN_TX pin is driven to logic low (0).
		1	The CAN_TX pin is at logic high (1).
			Forced to Tx output of the CAN Core, if Init bit of CAN Control register is reset.

Table 27-34. CAN TX IO Control Register (DCAN TIOC) Field Descriptions (continued)

Bit	Field	Value	Description
0	In		CAN_TX data in.
		0	The CAN_TX pin is at logic low (0).
		1	The CAN_TX pin is at logic high (1).
			Note: When CAN_TX pin is connected to a CAN transceiver, an external pullup resistor has to be used to ensure that the CAN bus will not be disturbed (for example, while the DCAN module is reset).

27.17.35 CAN RX IO Control Register (DCAN RIOC)

The CAN_RX pin of the DCAN module can be used as general-purpose IO pin if CAN function is not needed.

NOTE: The values of the IO Control registers are writable only if Init bit of CAN Control Register is set.

The OD, Func, and Dir bits of the CAN RX IO Control register are forced to certain values when Init bit of CAN Control Register is reset, see bit description.

Figure 27-80. CAN RX IO Control Register (DCAN RIOC) [offset = 1E4h]

31-19	18	17	16
Reserved	PU	PD	OD
R-0	R/W-D	R/W-D	R/WP-0

15-4	3	2	1	0
Reserved	Func	Dir	Out	In
R-0	R/WP-0	R/WP-0	R/WP-0	R-U

LEGEND: R/W = Read/Write; R = Read; WP = Protected Write (protected by Init bit); D = value is device-dependent; -*n* = value after reset

Table 27-35. CAN RX IO Control Register (DCAN RIOC) Field Descriptions

Bit	Field	Value	Description
31-19	Reserved	0	These bits are always read as 0. Writes have no effect.
18	PU		CAN_RX Pullup/Pulldown select. This bit is only active when CAN_RX is configured to be an input.
		0	CAN_RX Pulldown is selected, when pull logic is active (PD = 0).
		1	CAN_RX Pullup is selected, when pull logic is active (PD = 0).
17	PD		CAN_RX pull disable. This bit is only active when CAN_RX is configured to be an input.
		0	CAN_RX pull is active.
		1	CAN_RX pull is disabled.
16	OD		CAN_RX open drain enable. This bit is only active when CAN_RX is configured to be in GIO mode (RIOC.Func = 0).
		0	The CAN_RX pin is configured in push/pull mode.
		1	The CAN_RX pin is configured in open drain mode.
			Forced to 0, if Init bit of CAN control register is reset.
15-4	Reserved	0	These bits are always read as 0. Writes have no effect.
3	Func		CAN_RX function. This bit changes the function of the CAN_RX pin.
		0	CAN_RX pin is in GIO mode.
		1	CAN_RX pin is in functional mode (as an input to receive CAN data).
			Forced to 1, if Init bit of CAN control register is reset.

Table 27-35. CAN RX IO Control Register (DCAN RIOC) Field Descriptions (continued)

Bit	Field	Value	Description
2	Dir		CAN_RX data direction. This bit controls the direction of the CAN_RX pin when it is configured to be in GIO mode only (RIOC.Func = 0).
		0	The CAN_RX pin is an input.
		1	The CAN_RX pin is an output.
			Forced to 0, if Init bit of CAN control register is reset.
1	Out		CAN_RX data out write. This bit is only active when CAN_RX pin is configured to be in GIO mode (RIOC.Func = 0) and configured to be an output pin (RIOC.Dir = 1). The value of this bit indicates the value to be output to the CAN_RX pin.
		0	The CAN_RX pin is driven to logic low (0).
		1	The CAN_RX pin is at logic high (1).
0	In		CAN_RX data in.
		0	The CAN_RX pin is at logic low (0).
		1	The CAN_RX pin is at logic high (1).
			Note: When CAN_RX pin is connected to a CAN transceiver, an external pullup resistor has to be used to ensure that the CAN bus will not be disturbed (for example, while the DCAN module is reset).

Multi-Buffered Serial Peripheral Interface Module (MibSPI) with Parallel Pin Option (MibSPIP)

This chapter provides the specifications for a 16-bit configurable synchronous multi-buffered multi-pin serial peripheral interface (MibSPI). This chapter also provides the specifications for MibSPI with Parallel Pin Option (MibSPIP). The MibSPI is a programmable-length shift register used for high-speed communication between external peripherals or other microcontrollers.

Throughout this chapter, all references to SPI also apply to MibSPI/MibSPIP, unless otherwise noted.

NOTE: This chapter describes a superset implementation of the MibSPI/SPI modules that includes features and functionality that may not be available on some devices. Device-specific content that should be determined by referencing the datasheet includes DMA functionality, MibSPI RAM size, number of transfer groups, number of chip selects, parallel mode support, and availability of 5-pin operation (SPInENA).

Topic **Page**

28.1 Overview

28.1.1 Features

The MibSPI/SPI is a high-speed synchronous serial input/output port that allows a serial bit stream of programmed length (two to 16 bits) to be shifted into and out of the device at a programmed bit-transfer rate. The MibSPI/SPI is normally used for communication between the microcontroller and external peripherals or another microcontroller. Typical applications include interface to external I/O or peripheral expansion via devices such as shift registers, display drivers, and analog-to-digital converters. MibSPI is an Extension of SPI. MibSPI works in 2 modes.

- Compatibility Mode
- Multi-buffer Mode

The Compatibility mode of MibSPI makes it behave exactly like that of SPI and ensures full compatibility with the same. Everything described about compatibility mode of MibSPI , in this document, is directly applicable to SPI.

The Multi-buffer mode of operation is specific to MibSPI alone. This feature is not available in SPI.

The MibSPI supports memory fault detection/correction via internal Parity/ECC circuit. MibSPI is configurable to include or not include Memory Parity/ECC logic during circuit synthesis.

The SPI / MibSPI can be configured in three pin, four pin or five pin mode of operation. The SPI / MibSPI allows multiple programmable chip-selects.

The MibSPI has a programmable Multi-buffer array that enables programed transmission to be completed without CPU intervention. The buffers are combined in different transfer groups that could be triggered by external events (Timers, I/O, and so on) or by the internal tick counter. The internal tick counter can support periodic trigger events. Each buffer of the MibSPI can be associated with different DMA channels in different transfer group, allowing the user to move data from/to internal memory to/from external slave with a minimal CPU interaction.

The SPICLK, SPISIMO, and SPISOMI pins are used in all MibSPI pin modes. The $\overline{\text{SPIENA}}$ and $\overline{\text{SPICS}}$ pins are optional and may be used if the pin are present on a given device.

The SPI has the following attributes:

- 16-bit shift register
- Receive buffer register
- 8-bit baud clock generator
- Serial clock (SPICLK) I/O pin
- Up to 8 Slave out, Master in (SPISOMI) I/O pins for faster data transfers
- SPI enable ($\overline{\text{SPIENA}}$) pin (4 or 5-pin mode only)
- Up to 6 slave chip select ($\overline{\text{SPICS}}$) pins (4 or 5-pin mode only)
- SPI pins can be used as functional or digital Input/Output pins (GIOs)

The SPI/MibSPI allows software to program the following options:

- SPISOMI/SPISIMO pin direction configuration
- SPICLK pin source (external/internal)
- MibSPI pins as functional or digital I/O pins. For each Buffer, the following features can be selected from four different combinations of formats using the control fields in the buffer:
  - SPICLK frequency
  - Character length
  - Phase
  - Polarity
  - Enable/Disable parity for transmit and receive
  - Enable/Disable timers for Chip Select Hold and Setup timers
  - Direction of shifting, MSBit first or LSBit first

- Configurable Parallel modes to use multiple SIMO/SOMI pin
- Configurable number of Chip Selects

In Multi-buffer Mode, in addition to the previous features, many other features are configurable:

- Number of buffers for each peripheral (or data source/destination, up to 256 buffers supported) or group (up to 8 groupings)
- Number of DMA controlled buffers and number of DMA request channels (up to 8 for each of transmit and receive)
- Triggers for each groups, trigger types, trigger sources for individual groups(up to 14 external trigger sources and 1 internal trigger source)
- Number of DMA transfers for each buffer (up to 65536 for up to 8 buffers)
- Un-interrupted DMA buffer transfer (NOBREAK buffer)

NOTE: SIMO - Slave In Master Out Pin
SOMI - Slave Out Master In Pin
SPICS - SPI Chip Select Pin
SPIENA - SPI Enable Pin.

28.1.2 Pin Configurations

The SPI supports data connections as shown in Table 28-1.

Table 28-1. Pin Configurations

Pin	Master Mode		Slave Mode	
SPICLK	Drives the clock to external devices		Receives the clock from the external master	
SPISOMI	Receives data from the external slave		Sends data to the external master	
SPISIMO	Transmits data to the external slave		Receives data from the external master	
$\overline{\text{SPIENA}}$	**SPIENA disabled:** GIO	**SPIENA enabled:** Receives ENA signal from the external slave	**SPIENA disabled:** GIO	**SPIENA enabled:** Drives ENA signal from the external master
$\overline{\text{SPICS}}$	**SPICS disabled:** GIO	**SPICS enabled:** Selects one or more slave devices	**SPICS disabled:** GIO	**SPICS enabled:** Receives the CS signal from the external master

NOTE:

1. When the SPICS signals are disabled, the chip-select field in the transmit data is not used.
2. When the SPIENA signal is disabled, the $\overline{\text{SPIENA}}$ pin is ignored in master mode, and not driven as part of the SPI transaction in slave mode.

28.1.3 MibSPI /SPI Configurations

Table 28-2. MibSPI/SPI Configurations

MibSPIx/SPIx	I/Os
MibSPI1	MIBSPI1SIMO[1:0], MIBSPI1SOMI[1:0], MIBSPI1CLK, MIBSPI1nCS[5:0], MIBSPI1nENA
MibSPI2	MIBSPI2SIMO[1:0], MIBSPI2SOMI[1:0], MIBSPI2CLK, MIBSPI2nCS[5:0], MIBSPI2nENA
MibSPI3	MIBSPI3SIMO[1:0], MIBSPI3SOMI[1:0], MIBSPI3CLK, MIBSPI3nCS[5:0], MIBSPI3nENA
MibSPI4	MIBSPI4SIMO[1:0], MIBSPI4SOMI[1:0], MIBSPI4CLK, MIBSPI4nCS[5:0], MIBSPI4nENA
MibSPI5	MIBSPI5SIMO[1:0], MIBSPI5SOMI[1:0], MIBSPI51CLK, MIBSPI5nCS[5:0], MIBSP5nENA
SPI1	SPI1SIMO, ZSPI1SOMI, SPI1CLK, SPI2nCS[1:0], SPI1nENA
SPI2	SPI2SIMO, ZSPI2SOMI, SPI2CLK, SPI2nCS[1:0], SPI2nENA
SPI3	SPI3SIMO, ZSPI3SOMI, SPI3CLK, SPI3nCS[1:0], SPI3nENA

28.2 Basic Operation

This section details the basic operation principle of the SPI mode and the MibSPI mode operation of the device.

28.2.1 SPI Mode

The SPI can be configured via software to operate as either a master or a slave. The MASTER bit (SPIGCR1[0]) selects the configuration of the SPISIMO and SPISOMI pins. CLKMOD bit (SPIGCR1[1]) determines whether an internal or external clock source will be used.

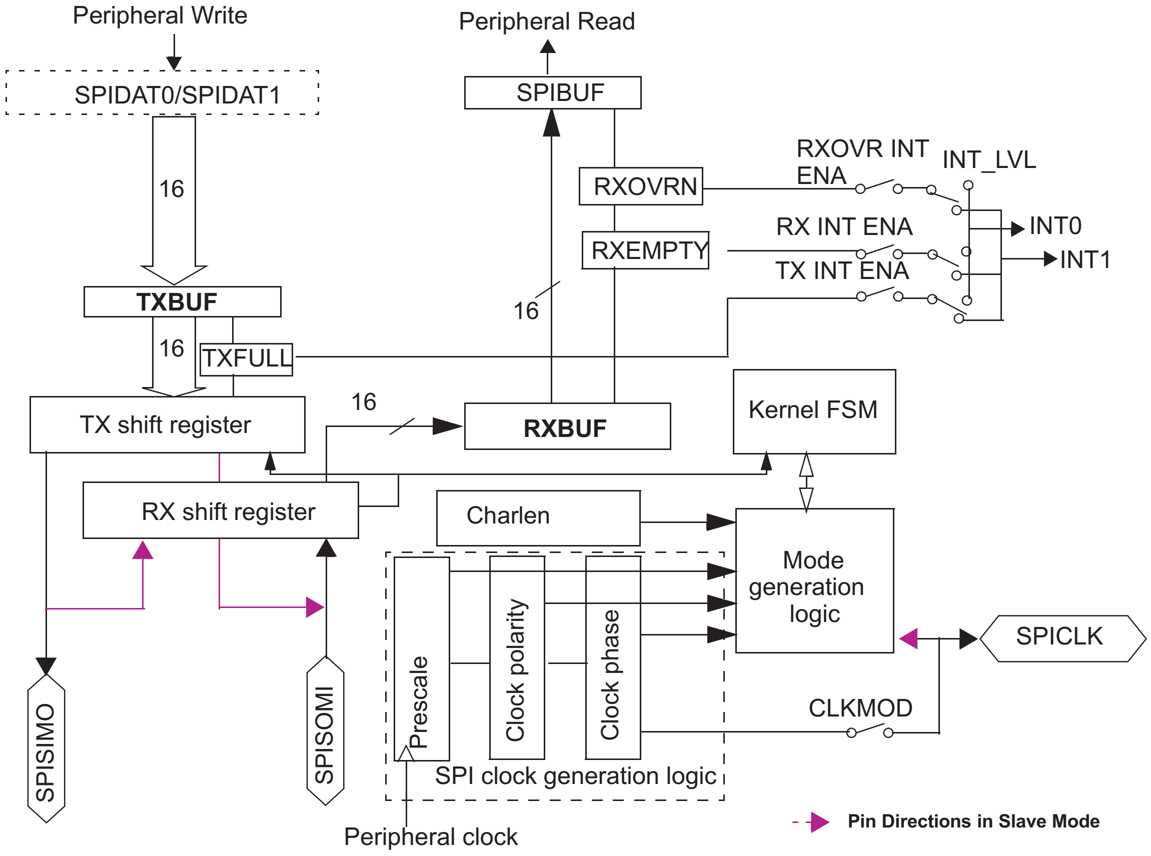
The slave chip select (SPICS) pins are used when communicating with multiple slave devices or, with a single slave, to delimit messages containing a leading register address. When a write occurs to SPIDAT1 in master mode, the SPICS pins are automatically driven to select the specified slave.

Handshaking mechanism, provided by the SPIENA pin, enables a slave SPI to delay the generation of the clock signal supplied by the master if it is not prepared for the next exchange of data.

28.2.1.1 SPI Mode Operation Block Diagram

Figure 28-1 shows the SPI transaction hardware. TXBUF and RXBUF are internal buffers that are intended to improve the overall throughput of data transfer. TXBUF is a transmit buffer, while RXBUF is a receive buffer.

Figure 28-1. SPI Functional Logic Diagram



1 This is a representative diagram, which shows three-pin mode hardware.
2 TXBUF, RXBUF, and SHIFT_REGISTER are user-invisible registers.
3 SPIDAT0 and SPIDAT1 are user-visible, and are physically mapped to the contents of TXBUF.
4 SPISIMO, SPISOMI, SPICLK pin directions depend on the Master or Slave Mode.

28.2.1.2 Data Flow and Handling for TX and RX

28.2.1.2.1 Data Sequencing when SPIDAT0 or SPIDAT1 is Written

- If both the TX shift register and TXBUF are empty, then the data is directly copied to the TX shift register. For devices with DMA, if DMA is enabled, a transmit DMA request (TX_DMA_REQ) is generated to cause the next word to be fetched. If transmit interrupts are enabled, a transmitter-empty interrupt is generated.
- If the TX shift register is already full or is in the process of shifting and if TXBUF is expty then the data written to SPIDAT0 / SPIDAT1 is copied to TXBUF and TXFULL flag is set to 1 at the same time.
- When a shift operation is complete, data from the TXBUF (if it is full) is copied into TX shift register and the TXFULL flag is cleared to 0 to indicate that next data can be fetched. A transmit DMA request (if enabled) or a transmitter-empty interrupt (if enabled) is generated at the same time.

28.2.1.2.2 Data Sequencing when All Bits Shifted into RXSHIFT Register

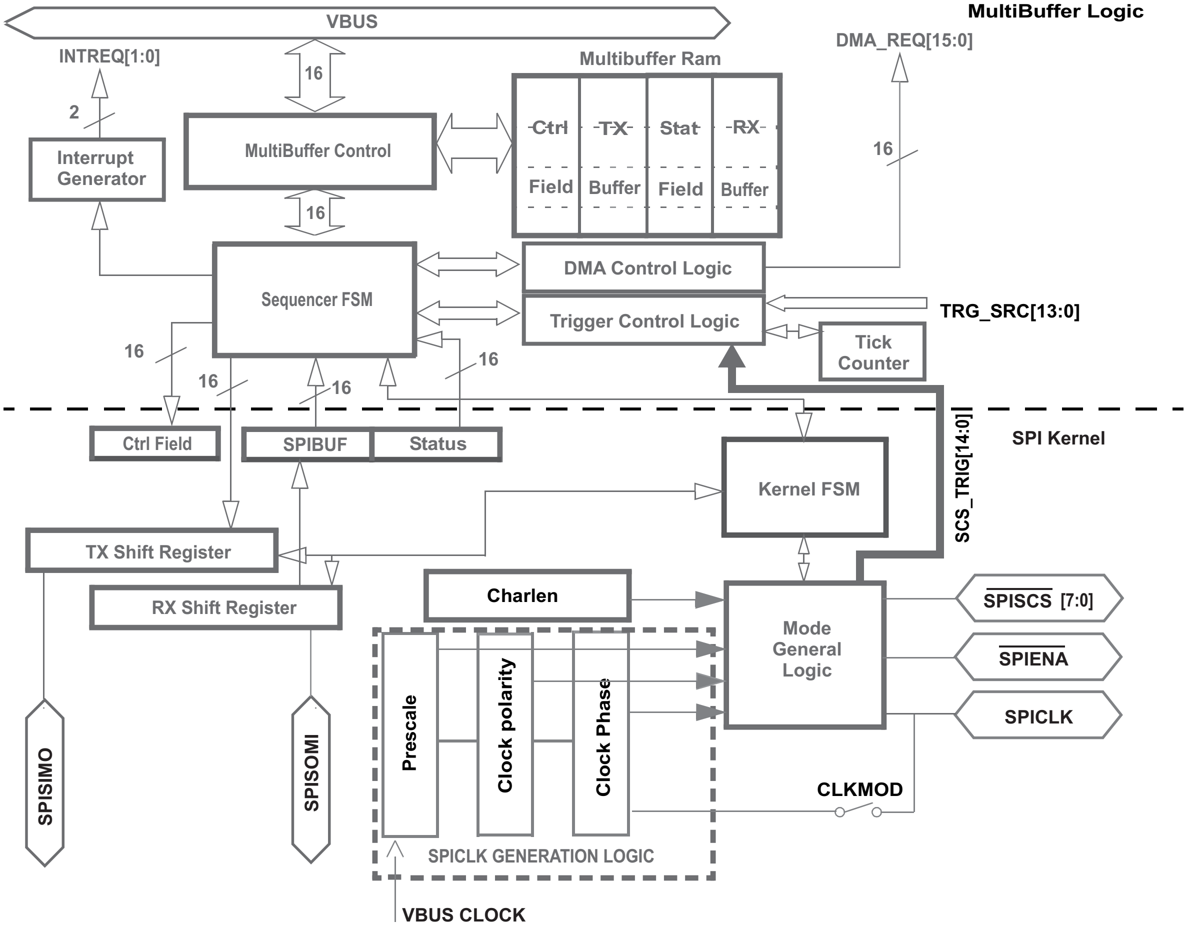
- If both SPIBUF and RXBUF are empty, the received data in RX shift register is directly copied into SPIBUF and the receive DMA request (if enabled) is generated and the receive-interrupt (if enabled) is generated. The RXEMPTY flag in SPIBUF is cleared at the same time.
- If SPIBUF is already full at the end of receive completion, the RX shift register contents is copied to RXBUF. A receive DMA request is generated, if enabled. The receive complete interrupt line remains high.
- If SPIBUF is read by the CPU or DMA and if RXBUF is full, then the contents of RXBUF are copied to SPIBUF as soon as SPIBUF is read. RXEMPTY flag remains cleared, indicating that SPIBUF is still full.
- If both SPIBUF and RXBUF are full, then RXBUF will be overwritten and the RXOVR interrupt flag is set and an interrupt is generated, if enabled.

NOTE: Prefetching is done only in Master mode. In Slave mode, since the TG to be serviced is known only after a valid ChipSelect assertion, no prefetching is done.

28.2.2 MibSPI Mode

Figure 28-2 shows multi-buffered mode operation. In Multi-buffer mode the transmit data has to be written to the TXRAM locations and the receive data has to be read from RXRAM locations of the multi-buffer RAM. A MibSPI supports up to 256 locations each for Transmit and Receive Data.

Figure 28-2. MibSPI Functional Logic Diagram



28.2.2.1 Data Handling for TX and RX Transfer Groups

28.2.2.1.1 Data Sequencing of a Transmit Data

In multi-buffer mode, any buffer that needs to be transmitted over by the SPI, should be associated with a Transfer Group. Each TG (Transfer Group) will have a Trigger Source based on which it'll be triggered. Once a TG is triggered, the buffers belonging to it will be transmitted.

Sequencer (FSM) controls the data flow from the multi-buffer RAM to the Shift Register. The Multi-buffer Control Logic has arbitration logic between VBUS and the Sequencer accessing the multi-buffer RAM. Sequencer picks up a highest priority Transfer Group from among the active TGs to be serviced. For the selected TG the starting buffer to be transferred is obtained from the PSTART of the respective TGxCTRL register.

Sequencer requests for the selected buffer through the Multi-buffer Control Logic, and once it receives the data, it reads the control fields to determine the subsequent action. Once the buffer is determined to be ready for transfer, the data is written to the TX SHIFT REGISTER by the Sequencer. This triggers the Kernel FSM to initiate the SPI transfer.

Once the Sequencer is finished writing to the TX SHIFT REGISTER, it prefetches the next buffer to be transferred from the multi-buffer RAM and stores the Data.

Once the Sequencer is finished writing to the TX SHIFT REGISTER, it prefetches the next buffer to be transferred from the multi-buffer RAM and stores the Data.

Sequencer writes the prefetched Transmit Data to the Shift Register immediately upon request by the Kernel. This way, the throughput of the SPI transfer is increased in Master mode of operation. In case of Slave mode, after the Receive data is copied to the RX RAM, Sequencer waits for the next active Chip Select trigger to fetch the next data.

28.2.2.1.2 Data Sequencing of the Received Data

At the end of a SPI transfer, the received Data is copied to SPIBUF register and then forwarded to the Sequencer. The Sequencer then, requests the Multi-buffer Control Logic to write the received data to the respective RXRAM location. Along with Received Data, the Status fields like Transmission Error Flags and the Last Chip Select Number (LCSNR) are forwarded to be updated in the Status Field of the RXRAM.

Sequencer clears the RXEMPTY bit while writing a new Received Data in the RXRAM. If the RXEMPTY bit is already 0, then the Sequencer sets the RCVR_OVRN bit to 1 to indicate that this particular location has been overwritten in the RXRAM.

28.2.3 DMA Requests

In order to reduce CPU overhead in handling SPI message traffic on a character-by-character basis, SPI can use the DMA controller to transfer the data

28.2.3.1 SPI/MibSPI Compatibility Mode DMA Requests

. The DMA request enable bit (DMA REQ EN) controls the assertion of requests to the DMA controller module. When a character is being transmitted or received, the SPI will signal the DMA via the DMA request signals, TX_DMA_REQ and RX_DMA_REQ. The DMA controller will then perform the required data transfer.

For efficient behavior during DMA operations, the transmitter empty and receive-buffer full interrupts can be disabled. For specific DMA features, see the DMA controller specification.

The SPI generates a request on the TX_DMA_REQ line each time the TX data is copied to the TX shift register either from the TXBUF or from peripheral data bus (when TXBUF is empty).

The first TX_DMA_REQ pulse is generated when either of the following is true:

- DMA REQ EN (SPIINT0[16]) is set to 1 while SPIEN (SPIGCR1[24]) is already 1.
- SPIEN (SPIGCR1[24]) is set to 1 while DMA REQ EN (SPIINT0[16]) is already 1.

The SPI generates a request on the RX_DMA_REQ line each time the received data is copied to the SPIBUF.

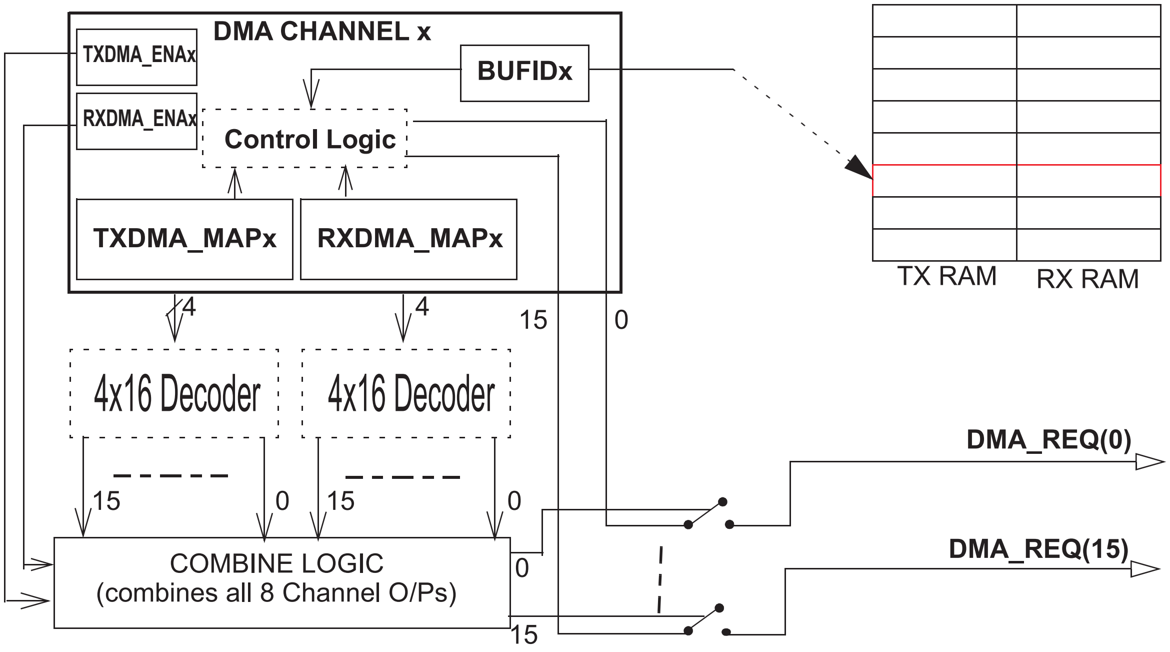
28.2.3.2 DMA in Multi-Buffer Mode

The MibSPI provides sophisticated programmable DMA control logic that completely eliminates the necessity of CPU intervention for data transfers, once programmed. When the multi-buffer mode is used, the DMA enable bit in the SPIINT0 register is ignored. DMA source or destination should be only the multi-buffer RAM and not SPIDAT0 / SPIDAT1 or SPIBUF register as in case of compatibility mode DMA.

The MibSPI offers up to eight DMA channels (for SEND and RECEIVE). All of the DMA channels are programmable individually and can be hooked to any buffer. The MibSPI provides up to 16 DMA request lines, and DMA requests from any channel can be programmed to be routed through any of these 16 lines. A DMA transfer can trigger both transmit and receive.

Each DMA channel has the capability to transfer a block of up to 32 data words without interruption using only one buffer of the array by configuring the DMAxCTRL register. Using the DMAxCOUNT and DMACTNTLEN register, up to 65535 (64K) words of data can be transferred without any interruption using just one buffer of the array. This enables the transfer of memory blocks from or into an external SPI memory.

Figure 28-3. DMA Channel and Request Line (Logical) Structure in Multi-buffer Mode



28.2.4 Interrupts

There are two levels of vectorized interrupts supported by the SPI. These interrupts can be caused under the following circumstances:

- Transmission error
- Receive overrun
- Receive complete (receive buffer full)
- Transmit buffer empty

These interrupts may be enabled or disabled via the SPIINT0 register.

During transmission, if one of the following errors occurs: BITERR, DESYNC, DLENERR, PARITYERR, or TIMEOUT, the corresponding bit in the SPIFLG register is set. If the corresponding enable bit is set, then an interrupt is generated. The level of all the above interrupts is set by the bit fields in the SPILVL register.

The error interrupts are enabled and prioritized independently from each other, but the interrupt generated will be the same if multiple errors are enabled on the same level. The SPIFLG register should be used to determine the actual cause of an error.

NOTE: Since there are two interrupt lines, one each for Level 0 and Level 1, it is possible for a programmer to separate out the interrupts for receive buffer full and transmit buffer empty. By programming one to Level 0 and the other to Level 1, it is possible to avoid a check on whether an interrupt occurred for transmit or for receive. A programmer can also choose to group all of the error interrupts into one interrupt line and both TX-empty and RX-full interrupts into another interrupt line using the LVL control register. In this way, it is possible to separate error-checking from normal data handling.

28.2.4.1 Interrupts in Multi-Buffer Mode

In multi-buffer mode, the SPI can generate interrupts on two levels.

In normal multi-buffer operation, the receive and transmit are not used and therefore the enable bits of SPIINT0 are not used.

The interrupts available in multi-buffer mode are:

- Transmission error interrupt
- Receive overrun interrupt
- TG suspended interrupt
- TG completed interrupt

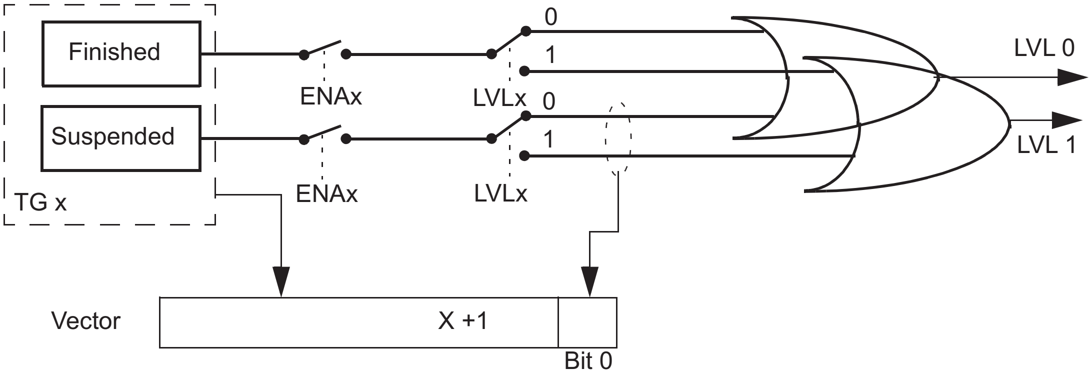
When a TG has finished and the corresponding enable bit in the TGINTENA register is set, a transfer-finished interrupt is generated. The level of priority of the interrupt is determined by the corresponding bit in the TGINTLVL register.

When a TG is suspended by a buffer that has been set as suspend to wait until TXFULL flag or/and RXEMPTY flag are set, and if the corresponding bit in the TGINTENA register is set, an transfer-suspended interrupt is generated. The level of priority of the interrupt is determined by the corresponding bit in the TGINTLVL register.

Figure 28-4 illustrates the TG interrupts.

During transmission, if one of the following errors occurs, BITERR, DESYNC, PARITYERR, TIMEOUT, DLENERR, the corresponding flag in the SPIFLG register is set. If the enable bit is set, then an interrupt is generated. The level of the interrupts could be generated according to the bit field in SPILVL register.

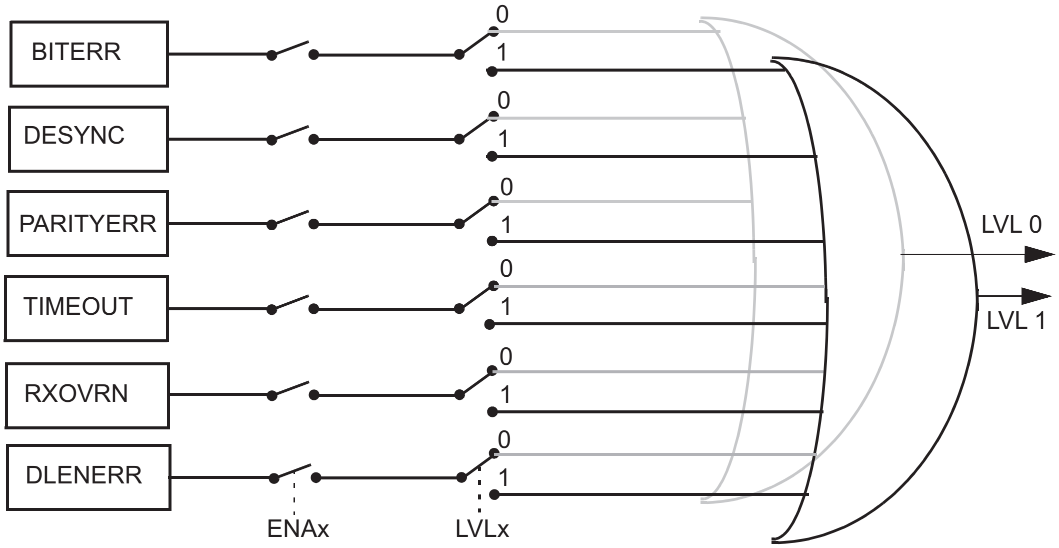
Figure 28-4. TG Interrupt Structure



The RXOVRN interrupt is generated when a buffer in the RXRAM is overwritten by a new received word. While writing newly received data to a RXRAM location, if the RXEMPTY bit of the corresponding location is 0, then the RXOVR bit will be set to 1 during the write operation, so that the buffer starts to indicate an overrun. This RXOVR flag is also reflected in SPIFLG register as RXOVRNINTFLG and the corresponding vector number is updated in TGINTVECT0/TGINTVECT1 register. If an overrun interrupt is enabled, then an interrupt will be generated indicating an overrun condition.

The error interrupts are enabled and prioritized independently from each other, but the vector generated by the SPI will be the same if multiple errors are enabled on the same level.

Figure 28-5. SPIFLG Interrupt Structure



Since the priority of an error interrupt is lower than a completion/suspend interrupt for a TG, the interrupts can be split into two levels. By programming all the error interrupts into Level 0 and TG-complete / TG-suspend interrupts into Level 1, it is possible to get a clear indication of the source of error interrupts. However, when a vector register shows an error interrupt, the actual buffer for which the error has occurred is not readily identifiable. Since each buffer in the multi-buffer RAM is stored along with its individual status flags, each buffer should be read until a buffer with any error flag set is found.

A separate interrupt line is provided to indicate the uncorrectable error condition in the MibSPI. This line is available (and valid) only in the multi-buffer mode of the MibSPI module and if the parity error detection feature for multi-buffer RAM is enabled.

28.2.5 Physical Interface

The SPI can be configured via software to operate as either a master or a slave. The MASTER bit (SPIGCR1[0]) selects the configuration of the SPISIMO and SPISOMI pins. The CLKMOD bit (SPIGCR1[1]) determines whether an internal or external clock source is used.

The slave chip select ($\overline{\text{SPICS}}$) pins, are used when communicating with multiple slave devices. When the a write occurs to SPIDAT1 in master mode, the $\overline{\text{SPICS}}$ pins are automatically driven to select the specified slave.

Handshaking mechanism, provided by the $\overline{\text{SPIENA}}$ pin, enables a slave SPI to delay the generation of the clock signal supplied by the master if it is not prepared for the next exchange of data.

28.2.5.1 Three-Pin Mode

In master mode configuration (MASTER = 1 and CLKMOD = 1), the SPI provides the serial clock on the SPICLK pin. Data is transmitted on the SPISIMO pin and received on the SPISOMI pin (see Figure 28-6).

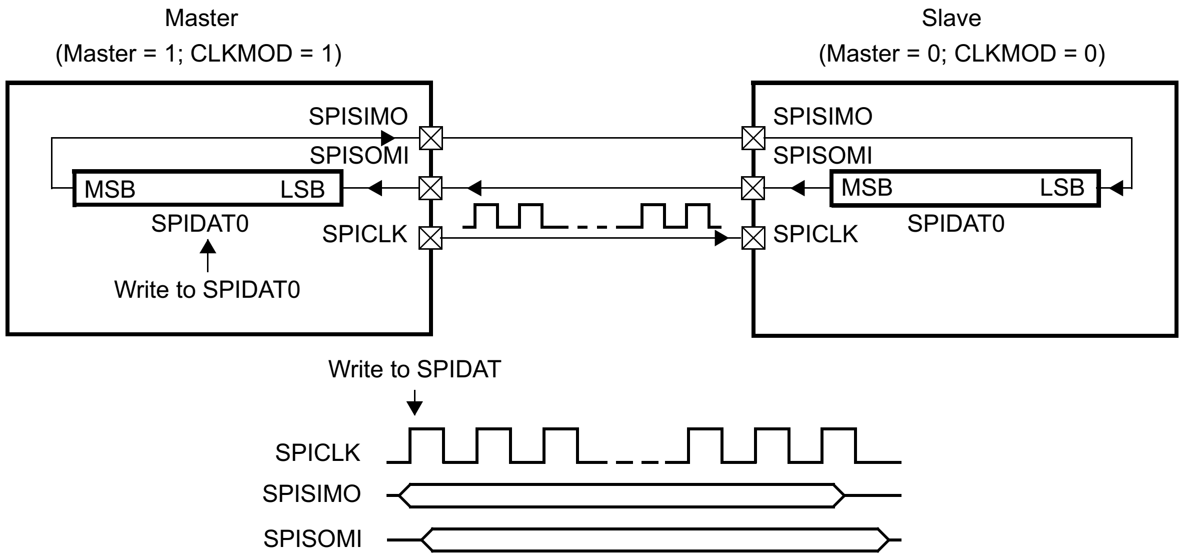
Data written to the shift register (SPIDAT0 / SPIDAT1) initiates data transmission on the SPISIMO pin, MSB first. Simultaneously, received data is shifted through the SPISOMI pin into the LSB of the SPIDAT0 register. When the selected number of bits have been transmitted, the received data in the shift register is transferred to the SPIBUF register for the CPU to read. Data is stored right-justified in SPIBUF.

See Section 28.2.1.2.2 and Section 28.2.2 for details about the data handling for transmit and receive operations.

In slave mode configuration (MASTER = 0 and CLKMOD = 0), data shifts out on the SPISOMI pin and in on the SPISIMO pin. The SPICLK pin is used as the input for the serial shift clock, which is supplied from the external network master. The transfer rate is defined by this clock.

Data written to the SPIDAT0 or SPIDAT1 register is transmitted to the network when the SPICLK signal is received from the network master. To receive data, the SPI waits for the network master to send the SPICLK signal and then shifts data on the SPISIMO pin into the RX shift register. If data is to be transmitted by the slave simultaneously, it must be written to the SPIDAT0 or SPIDAT1register before the beginning of the SPICLK signal.

Figure 28-6. SPI Three-Pin Operation



28.2.5.2 Four-Pin Mode with Chip Select

The three-pin option and the four-pin option of the SPI / MibSPI are identical in the master mode (CLKMOD = 1), except that the four-pin option uses either SPIENA or SPICS pins. The I/O directions of these pins are determined by the CLKMOD control bit as SPI / MibSPI and is not general-purpose I/O.

28.2.5.2.1 Four-Pin Option with SPICS

In master mode, each chip select signal is used to select a specific slave. In slave mode, the chip select signal is used to enable and disable the transfer. Chip-select functionality is enabled by setting one of the SPICS pins as a chip select. It is disabled by setting all SPICS pins as GIOs in SPIPC0.

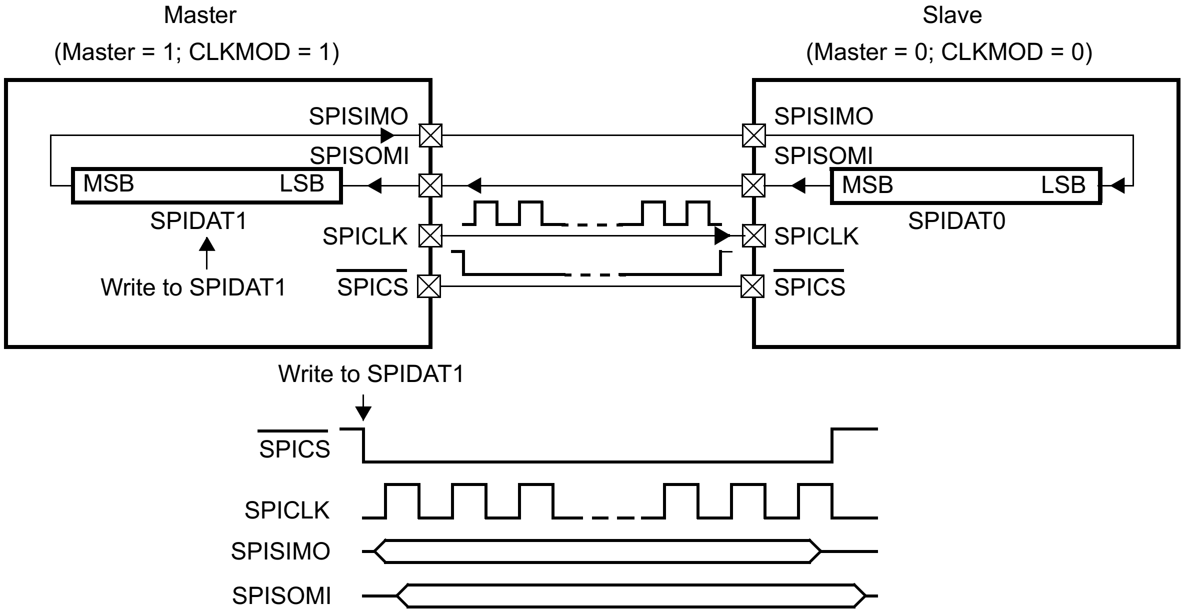
28.2.5.2.1.1 Multiple Chip Selects

The SPICS pins that are used must be configured as functional pins in the SPIPC0 register. The default pattern to be put on the SPICS when all the slaves are deactivated is set in the SPIDEF register. This pattern allows different slaves with different chip-select polarity to be activated by the SP/MibSPI.

The master-mode SPI is capable of driving either 0 or 1 as the active value for any SPICS output pin. The drive state for the SPICS pins is controlled by the CSNR field of SPIDAT1. The pattern that is driven will select the slave to which the transmission is dedicated.

In slave mode, the SPI can only be selected by an active value of 0 on any of its selected SPICS input pins.

Figure 28-7. Operation with SPICS



28.2.5.2.2 Four-Pin Option with SPIENA

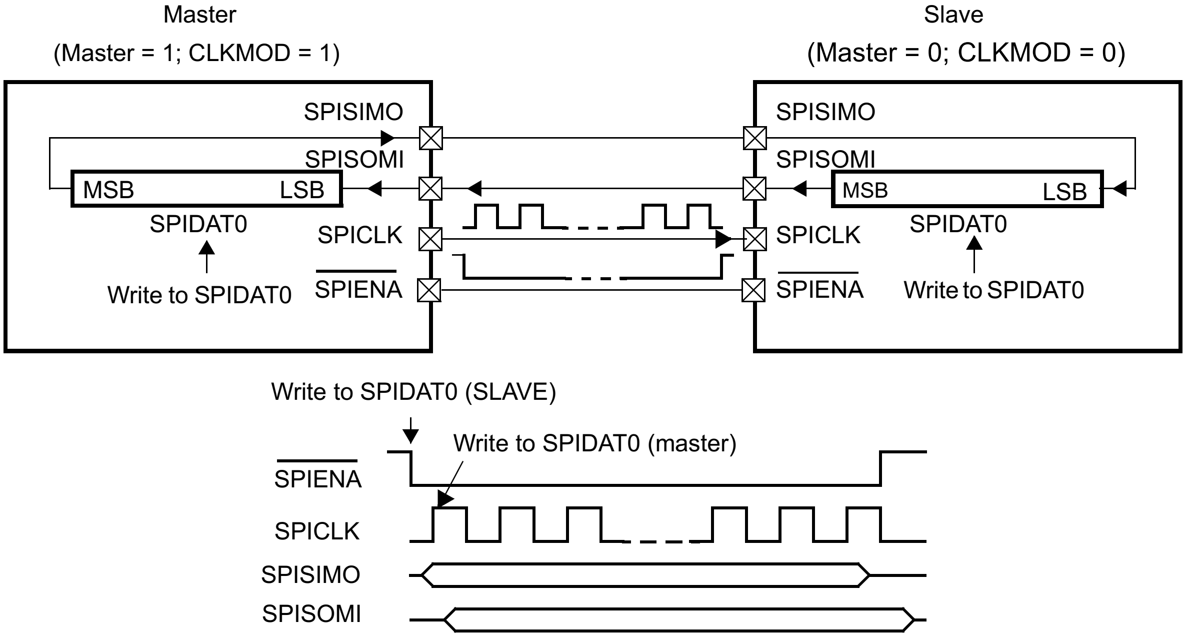
The SPIENA operates as a WAIT signal pin. For both the slave and the master, the SPIENA pin must be configured to be functional (SPIPC0[8] = 1). In this mode, an active-low signal from the slave on the SPIENA pin allows the master SPI to drive the clock pulse stream. A high signal tells the master to hold the clock signal (and delay SPI activity).

If the SPIENA pin is in high-impedance mode (ENABLE_HIGHZ = 1), the slave will put SPIENA into the high-impedance once it completes receiving a new character. If the SPIENA pin is in push-pull mode (ENABLE_HIGHZ = 0), the slave will drive SPIENA to 1 once it completes receiving a new character. The slave will drive SPIENA low again for the next word to transfer, after new data is written to the slave TX shift register.

In master mode (CLKMOD = 1), if the SPIENA pin is configured as functional, then the pin acts as an input pin. If configured as a slave SPI and as functional, the SPIENA pin acts as an output pin.

NOTE: During a transfer, if a slave-mode SPI detects a deassertion of its chip select before its internal character length counter overflows, then it places SPISOMI and SPIENA (if ENABLE_HIGHZ bit is set to 1) in high-impedance mode. Once this condition has occurred, if a SPICLK edge is detected while the chip select is deasserted, then the SPI stops that transfer and sets an DLENERR error flag and generates an interrupt (if enabled).

Figure 28-8. Operation with SPIENA



28.2.5.3 Five-Pin Operation (Hardware Handshaking)

Five-pin operation combines the functionality of three-pin mode, plus the enable pin and one or more chip select pins. The result is full hardware handshaking. To use this mode, both the SPIENA pin and the required number of SPICS pins must be configured as functional pins.

If the SPIENA pin is in high-impedance mode (ENABLE_HIGHZ = 1), the slave SPI will put this signal into the high-impedance state by default. The slave will drive the signal SPIENA low when new data is written to the slave shift register and the slave has been selected by the master (SPICS is low).

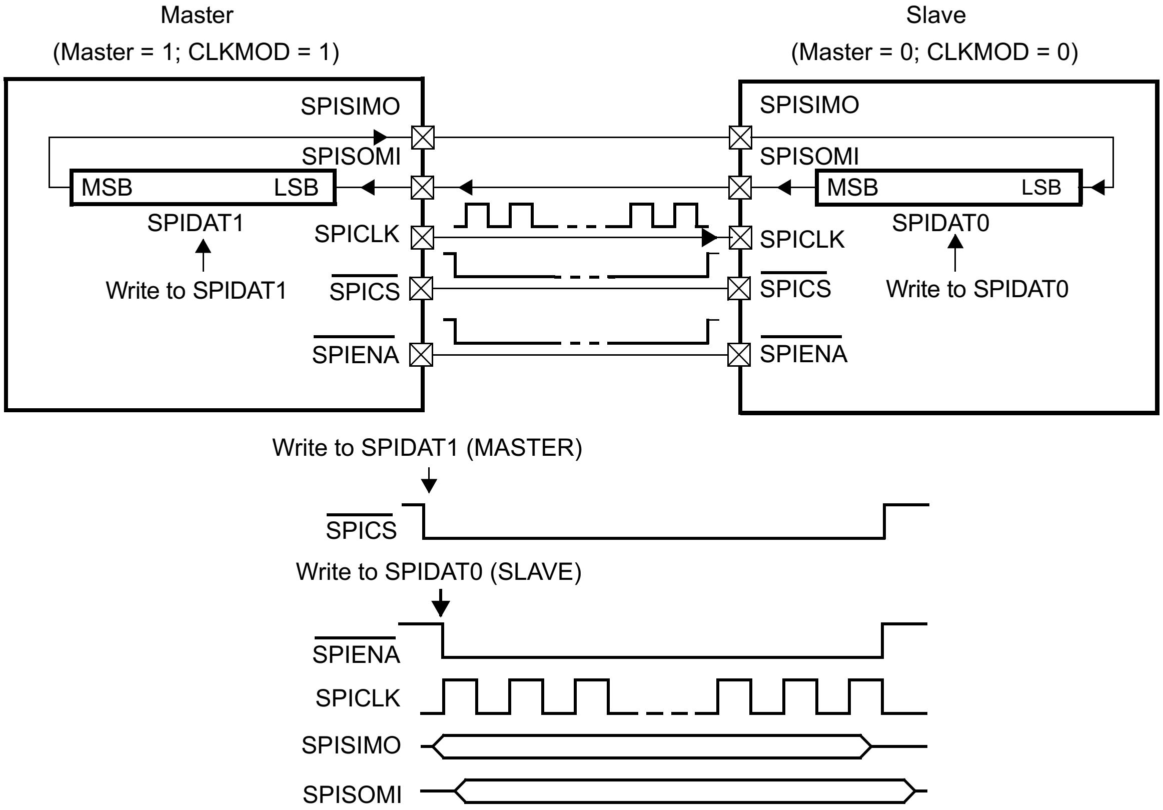
If the SPIENA pin is in push-pull mode (ENABLE_HIGHZ = 0), the slave SPI drives this pin high by default when it is in functional mode. The slave SPI will drive the SPIENA signal low when new data is written to the slave shift register (SPIDAT0/SPIDAT1) and the slave is selected by the master (SPICS is low). If the slave is deselected by the master (SPICS goes high), the slave SPIENA signal is driven high.

NOTE: Push-pull mode of the SPIENA pin can be used only when there is a single slave in the system. When multiple SPI slave devices are connected to the common SPIENA pin, all of the slaves should configure their SPIENA pins in high-impedance mode.

In master mode, if the SPICS pins are configured as functional pins, then the pins will be in output mode. A write to the master's SPIDAT1/SPIDAT0 register will automatically drive the SPICS signals low. The master will drive the SPICS signals high again after completing the transfer of the bits of the data.

In slave mode (CLKMOD = 0), the SPICS pins act as SPI functional inputs.

Figure 28-9. SPI Five-Pin Option with SPIENA and SPICS



28.2.6 Advanced Module Configuration Options

28.2.6.1 Data Formats

To support multiple different types of slaves in one SPI network, four independent data word formats are implemented that allow configuration of individual data word length, polarity, phase, and bit rate. Each word transmitted can select which data format to use via the bits DFSEL[1:0] in its control field from one of the four data word formats. Same data format can be supported on multiple chip selects.

Data formats 0, 1, 2, and 3 can be configured through SPIFMTx control registers.

Each SPI data format includes the standard SPI data format with enhanced features:

- Individually-configurable shift direction can be used to select MSB first or LSB first, whereas the position of the MSB depends on the configured data word length.
- Receive data is automatically right-aligned, independent of shift direction and data word length. Transmit data has to be written right-aligned into the SPI and the internal shift register will transmit according to the selected shift direction and data word length for correct transfer.
- To increase fault detection of data transmission and reception, an odd or even parity bit can be added at the end of a data word. The parity generator can be enabled or disabled individually for each data format. If a received parity bit does not match with the locally calculated parity bit, the parity error flag (PARITYERR) is set and an interrupt is asserted (if enabled).

Since the master-mode SPI can drive two consecutive accesses to the same slave, an 8-bit delay counter is available to satisfy the delay time for data to be refreshed in the accessed slave. The delay counter can be programmed as part of the data format.

CHARLEN[4:0] specifies the number of bits (2 to 16) in the data word. The CHARLEN[4:0] value directs the state control logic to count the number of bits received or transmitted to determine when a complete word is transferred.

Data word length **must** be programmed to the same length for both the **master** and the **slave**. However, when chip selects are used, there may be multiple targets with different lengths in the system.

NOTE: Data must be right-justified when it is written to the SPI for transmission irrespective of its character length or word length.

Figure 28-10 shows how a 12-bit word (0xEC9) needs to be written to the transmit buffer to be transmitted correctly.

Figure 28-10. Format for Transmitting an 12-Bit Word

D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
x	x	x	x	1	1	1	0	1	1	0	0	1	0	0	1

NOTE: The received data is always stored right-justified regardless of the character length or direction of shifting and is padded with leading 0s when the character length is less than 16 bits.

Figure 28-11 shows how a 10-bit word (0x0A2) is stored in the buffer once it is received.

Figure 28-11. Format for Receiving an 10-Bit Word

D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
0	0	0	0	0	0	0	0	1	0	1	0	0	0	1	0

28.2.6.2 Clocking Modes

SPICLK may operate in four different modes, depending on the choice of phase (delay/no delay) and the polarity (rising edge/falling edge) of the clock.

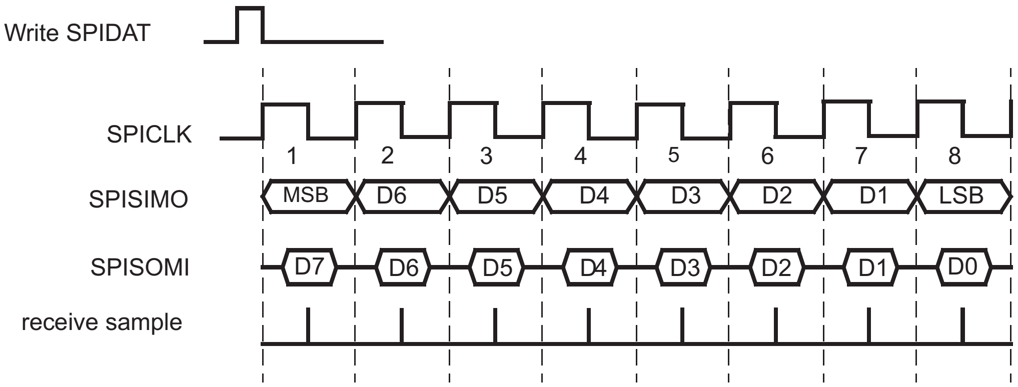
The data input and output edges depend on the values of both POLARITY and PHASE as shown in Table 28-3.

Table 28-3. Clocking Modes

POLARITY	PHASE	Action
0	0	Data is output on the rising edge of SPICLK. Input data is latched on the falling edge.
0	1	Data is output one half-cycle before the first rising edge of SPICLK and on subsequent falling edges. Input data is latched on the rising edge of SPICLK.
1	0	Data is output on the falling edge of SPICLK. Input data is latched on the rising edge.
1	1	Data is output one half-cycle before the first falling edge of SPICLK and on subsequent rising edges. Input data is latched on the falling edge of SPICLK.

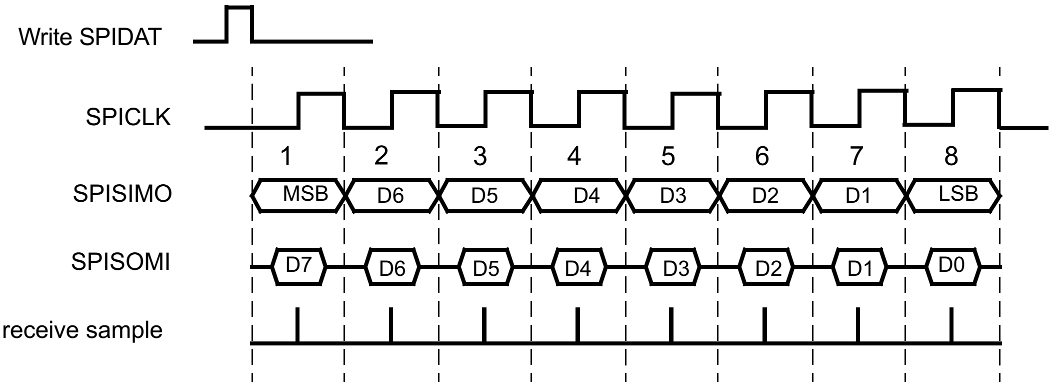
Figure 28-12 to Figure 28-15 illustrate the four possible configurations of **SPICLK** corresponding to each mode. Having four signal options allows the SPI to interface with many different types of serial devices.

Figure 28-12. Clock Mode with Polarity = 0 and Phase = 0



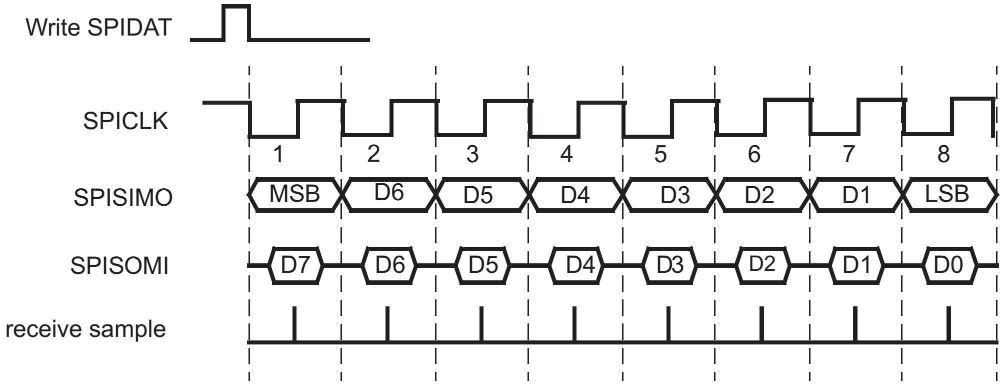
Data is output on the rising edge of SPICLK.
Input data is latched on the falling edge of SPICLK.

Figure 28-13. Clock Mode with Polarity = 0 and Phase = 1



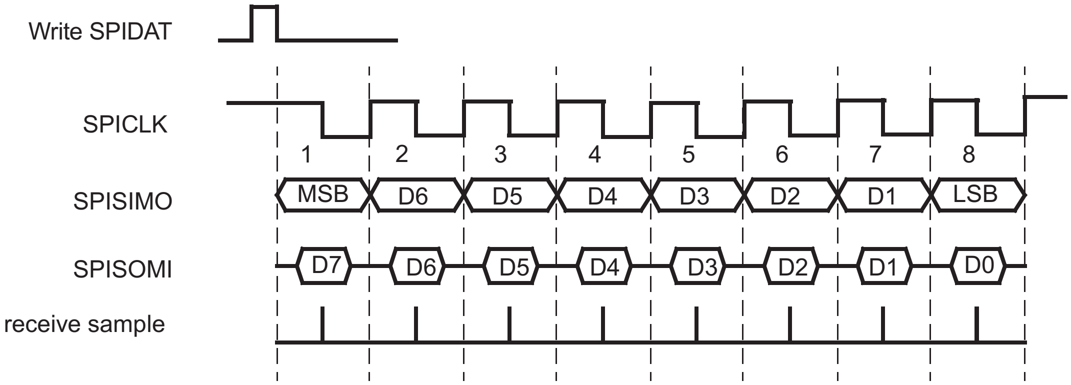
Data is output one-half cycle before the first rising edge of SPICLK and on subsequent falling edges of SPICLK
Input data is latched on the rising edge of SPICLK

Figure 28-14. Clock Mode with Polarity = 1 and Phase = 0



Data is output on the falling edge of SPICLK.
Input data is latched on the rising edge of SPICLK.

Figure 28-15. Clock Mode with Polarity = 1 and Phase = 1

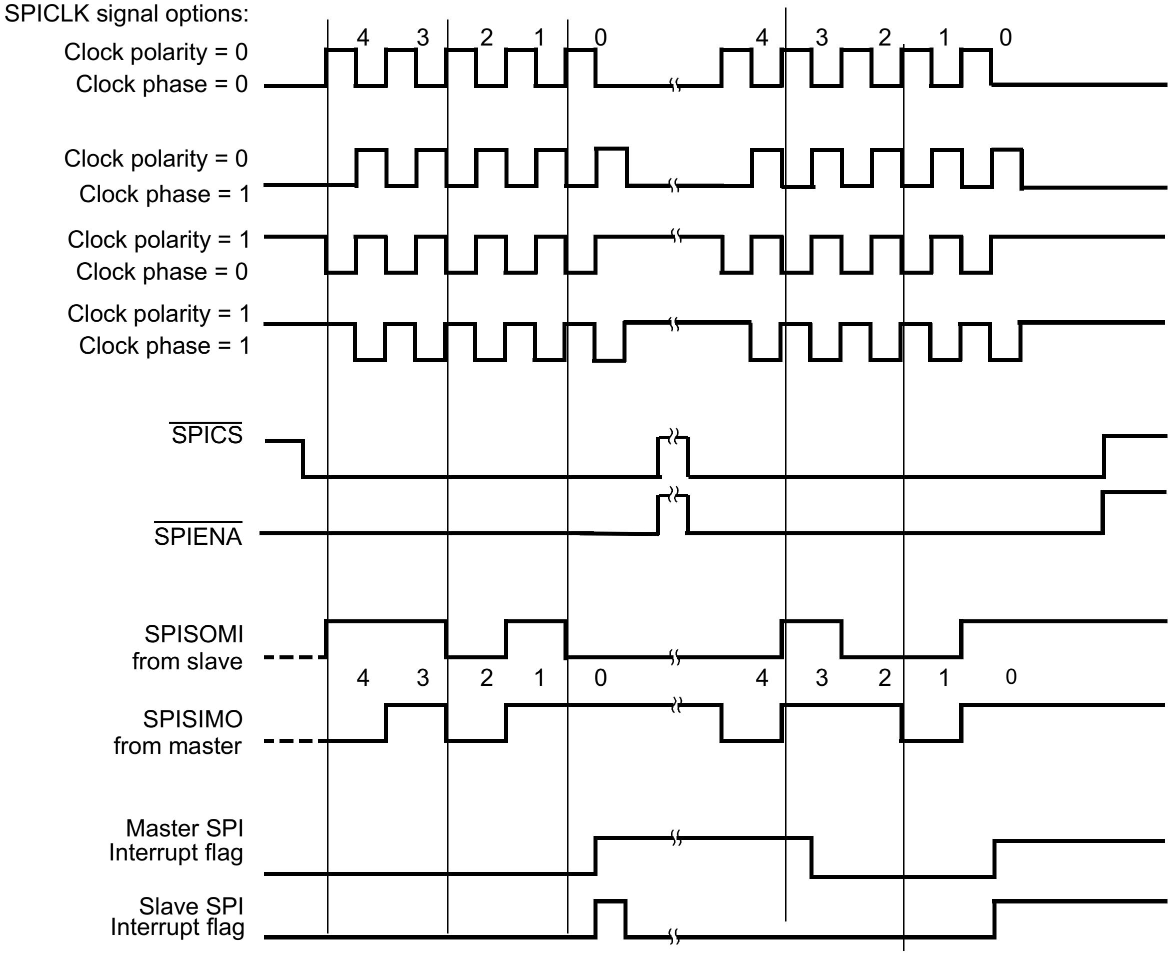


Data is output one-half cycle before the first falling edge of SPICLK and on the subsequent rising edges of SPICLK.
Input data is latched on the falling edge of SPICLK.

28.2.6.2.1 Data Transfer Example

Figure 28-16 illustrates a SPI data transfer between two devices using a character length of five bits.

Figure 28-16. Five Bits per Character (5-Pin Option)



28.2.6.3 Decoded and Encoded Chip Select (Master Only)

In this device, the SPI can connect to up to 6 individual slave devices using chip-selects by routing one wire to each slave. The 6 chip selects in the control field are directly connected to the 6 pins. The default value of each chip select (not active) can be configured via the register CSDEF. During a transmission, the value of the chip select control field (CSNR) of the SPIDAT1 register is driven on the $\overline{\text{SPICS}}$ pins. When the transmission finishes, the default chip-select value (defined by the CSDEF register) is put on the $\overline{\text{SPICS}}$ pins.

The SPI can support more than 6 slaves by using encoded chip selects. To connect the SPI with encoded slaves devices, the CSNR field allows multiple active $\overline{\text{SPICS}}$ pins at the same time, which enables encoded chip selects from 0 to 16. To use encoded chip selects, all 6 chip select lines have to be connected to each slave device and each slave needs to have a unique chip-select address. The CSDEF register is used to provide the address at which slaves devices are all de-selected.

Users can combine decoded and encoded chip selects. For example, *n* $\overline{\text{SPICS}}$ pins can be used for encoding an *n*-bit address and the remaining pins can be connected to decoded-mode slaves.

28.2.6.4 Chip Select Timing Control

This section describes fields of the control register SPIDELAY. This register decides the chip select and timing control for the device.

28.2.6.4.1 Chip-Select-Active-to-Transmit-Start-Delay (C2TDELAY)

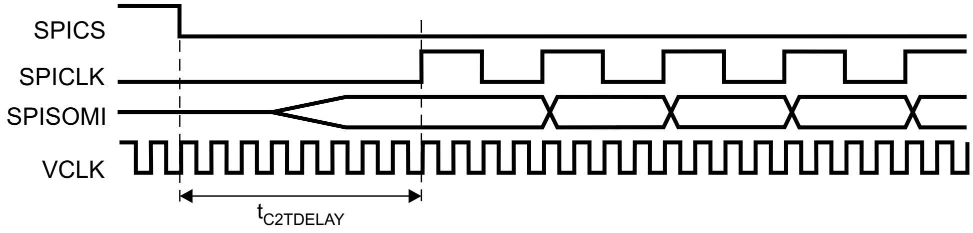
C2TDELAY is used in master mode only. It defines a setup time for the slave device that delays the data transmission from the chip select active edge by a multiple of VCLK cycles. Chip Select-active-to-transmission delays between 2 to 257 VCLK cycles can be achieved.

The setup time value is calculated as:

t_{C2TDELAY}= (C2TDELAY + 2) × VCLK Period

Figure 28-17 is the timing diagram when C2TDELAY of 8 VCLK Cycles.

Figure 28-17. Example: t_{C2TDELAY}= 8 VCLK Cycles



28.2.6.4.2 Transmit-End-to-Chip-Select-Inactive-Delay (T2CDELAY)

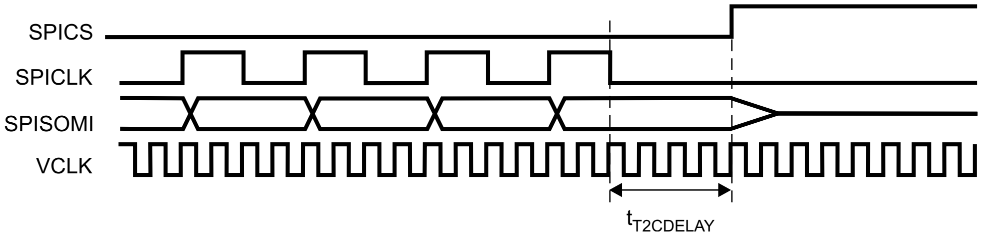
T2CDELAY is used in master mode only. It defines a hold time for the slave device that delays the chip select deactivation by a multiple of VCLK cycles after the last bit is transferred. T2CDELAY can be configured between 2 and 256 VCLK cycles.

The hold time value is calculated as:

$t_{T2CDELAY}$= (T2CDELAY +1) × VCLK Period

Figure 28-18 is the timing diagram when T2CDELAY of 4 VCLK Cycles.

Figure 28-18. Example: $t_{T2CDELAY}$= 4 VCLK Cycles



28.2.6.4.3 Transmit-Data-Finished-to-ENA-Pin-Inactive-Time-Out (T2EDELAY)

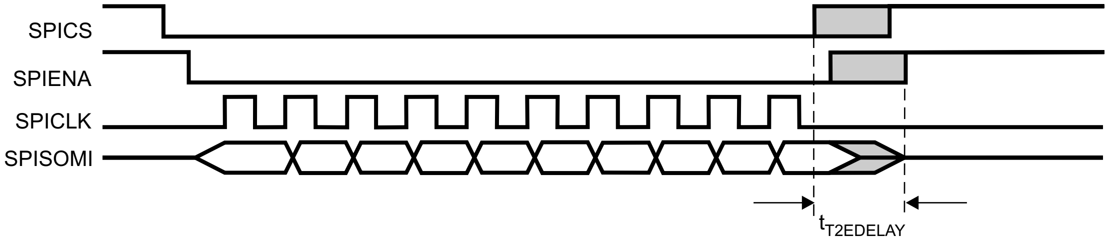
T2EDELAY is used in master mode only. It defines a time-out value as a multiple of SPI clock before the ENAble signal has to become inactive and after the CS becomes inactive. The SPI clock depends on which data format is selected. If the slave device is missing one or more clock edges, it is becoming de-synchronized. Although the master has finished the data transfer the

The T2EDELAY defines a time-out value that triggers the DESYNC flag, if the ENA signal is not deactivated in time. DESYNC flag is set to indicate that the Slave device did not deassert its $\overline{\text{SPIENA}}$ pin in time to acknowledge that it has received all the bits of the sent character.

The timeout value is calculated as:

$t_{T2EDELAY}$ = T2EDELAY/SPIclock

Figure 28-19. Transmit-Data-Finished-to-ENA-Inactive-Timeout



NOTE: If T2CDELAY is programmed a non-zero value, then T2EDELAY will start only after the T2CDELAY completes. This should be taken into consideration to determine an optimum value of T2EDELAY.

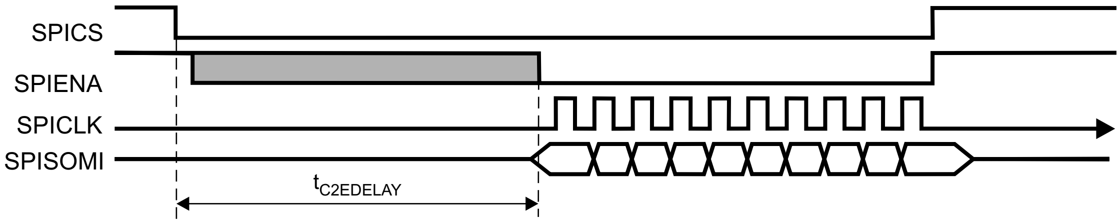
28.2.6.4.4 Chip-Select-Active-to-ENA-Signal-Active-Time-Out (C2EDELAY)

C2EDELAY is used in master mode only and it applies only if the addressed slave generates an ENAble signal as a hardware handshake response. C2EDELAY defines the maximum time between the SPI / MibSPI activating the chip select signal and the addressed slave responding by activating the ENA signal. C2EDELAY defines a time-out value as a multiple of SPI clocks. The SPI clock depends on whether data format 0 or data format 1 is selected.

The timeout value is calculated as:

$t_{C2EDELAY}$ = C2EDELAY/SPIclock

Figure 28-20. Chip-Select-Active-to-ENA-Signal-Active-Timeout



NOTE:

- If the slave device is not responding with the ENA signal before the time-out value is reached, the TIMEOUT flag in SPIFLG register is set and an interrupt is asserted if enabled.
- If a time-out occurs the MibSPI clears the transmit request of the timed-out buffer, sets the TIMEOUT flag for the current buffer and continues with the transfer of the next buffer in the sequence that is enabled.
- If C2TDELAY is programmed a non-zero value, then C2EDELAY will start only after the C2TDELAY completes. This should be taken into consideration to determine an optimum value of C2EDELAY.

28.2.6.5 Multiple Transfers to Same Slave and Variable Chip Select Setup and Hold Timing

This section gives information on the variable chip select setup and it shows how the CSHOLD bit is used and how the multiple transfers to same slave is enabled in the device.

28.2.6.5.1 Variable Chip Select Setup and Hold Timing (Master Only)

In order to support slow slave devices, a delay counter can be configured to delay data transmission after the chip select is activated. A second delay counter can be configured to delay the chip select deactivation after the last data bit is transferred. Both delay counters are clocked with the peripheral clock (VCLK).

If a particular data format specifically does not require these additional set-up or hold times for the chip select pins, then they can be disabled in the corresponding SPIFMTx register.

28.2.6.5.2 Hold Chip-Select Active

Some slave devices require the chip select signal to be held continuously active during several consecutive data word transfers. Other slave devices require the chip select signal to be deactivated between consecutive data word transfers.

CSHOLD is programmable in both master and slave modes of the multi-buffer mode of SPI. However, the meaning of CSHOLD in master mode and slave mode are different.

NOTE: If the CSHOLD bit is set within the current data control field, the programmed hold time and the following programmed set-up time will not be applied between transactions.

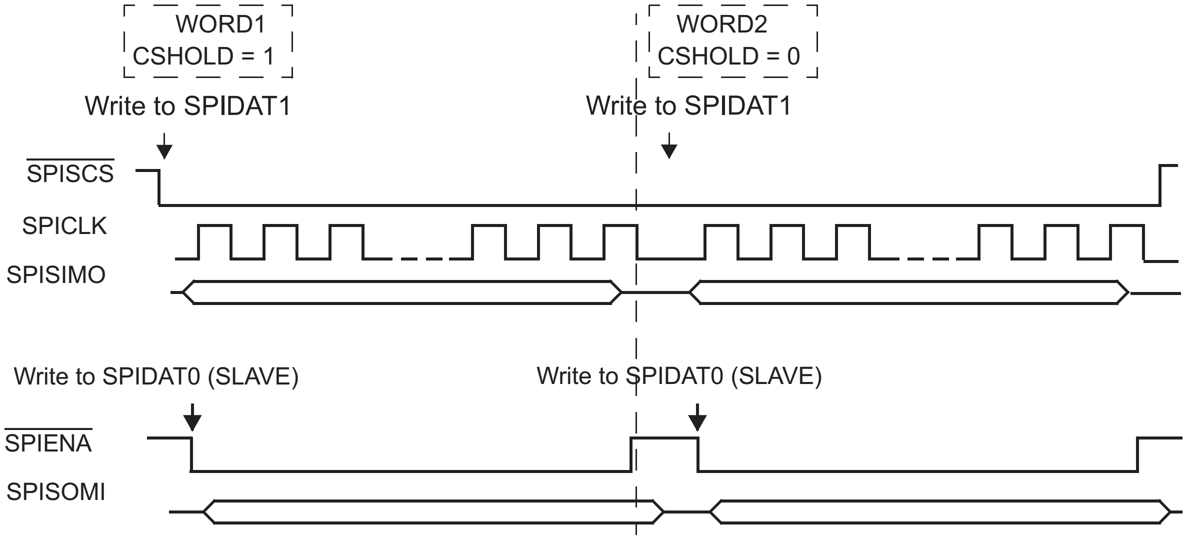
28.2.6.5.2.1 CSHOLD Bit in Master Mode

Each word in a master-mode SPI can be individually initialized for one of the two modes via the CSHOLD bit in its control field.

If the CSHOLD bit is set in the control field of a word, the chip select signal will not be deactivated until the next control field is loaded with new chip select information. Since the chip-select is maintained active between two transfers, the chip-select hold delay (T2CDELAY) is not applied at the end of the current transaction, and the chip-select set-up time delay (C2TDELAY) is not applied as well at the beginning of the following transaction. However, the wait delay (WDELAY) will be still applied between the two transactions, if the WDEL bit is set within the control field.

Figure 28-21 shows the SPI pins when a master-mode SPI transfers a word that has its CSHOLD bit set. The chip-select pins will not be deasserted after the completion of this word. If the next word to transmit has the same chip-select number (CSNR) value, the chip select pins will be maintained until the completion of the second word, regardless of whether the CSHOLD bit is set or not.

Figure 28-21. Typical Diagram when a Buffer in Master is in CSHOLD Mode (SPI-SPI)



28.2.6.5.2.2 CSHOLD Bit in Slave Mode (Multi-buffered Mode)

If the CSHOLD bit in a buffer is set to 1, then the MibSPI does not wait for the $\overline{\text{SPICS}}$ pins to be de-activated at the end of the shift operation to copy the received data to the receive RAM. With this feature, it is possible for a slave in multi-buffer mode to do multiple data transfers without requiring the $\overline{\text{SPICS}}$ pins to be deasserted between two buffer transfers.

If the CSHOLD bit in a buffer is cleared to 0 in a slave MibSPI, even after the shift operation is done, the MibSPI waits until the $\overline{\text{SPICS}}$ pin (if functional) is deasserted to copy the received data to the RXRAM.

If the CSHOLD bit is maintained as 0 across all the buffers, then the slave in multi-buffer mode requires its $\overline{\text{SPICS}}$ pins to be deasserted between any two buffer transfers; otherwise, the Slave SPI will be unable to respond to the next data transfer.

NOTE: In compatibility mode, the slave does not require the $\overline{\text{SPICS}}$ pin to be deasserted between two buffer transfers. The CSHOLD bit of the slave will be ignored in compatibility mode.

28.2.6.6 Parallel Mode (Multiple SIMO/SOMI Support, not available on all devices)

In order to increase throughput, the parallel mode of the SPI enables the module to send data over more than one data line (parallel 2, 4, or 8). When parallel mode is used, the data length must be set as 16 bits. Only module MIBSPIP5 supports Parallel Mode.

This feature increases throughput by 2 for 2 pins, by 4 for 4 pins, or by 8 for 8 pins.

Parallel mode supports the following features:

- Scalable data lines (1, 2, 4, 8) per direction. (SOMI and SIMO lines)
- All clock schemes are supported (clock phase and polarity)
- Parity is supported. The parity bit will be transmitted on bit0 of the SIMO/SOMI lines. The receive parity is expected on bit0 of the SOMI/SIMO pins.

Parallel mode can be programmed using the PMODEx bits of SPIPMCTRL register. See Section 28.3.25 for details about this register.

After reset the parallel mode selection bits are cleared (single SIMO/SOMI lines).

28.2.6.6.1 Parallel Mode Block Diagram

Figure 28-22 and Figure 28-23 show the parallel connections to the SPI shift register.

Figure 28-22. Block Diagram Shift Register, MSB First

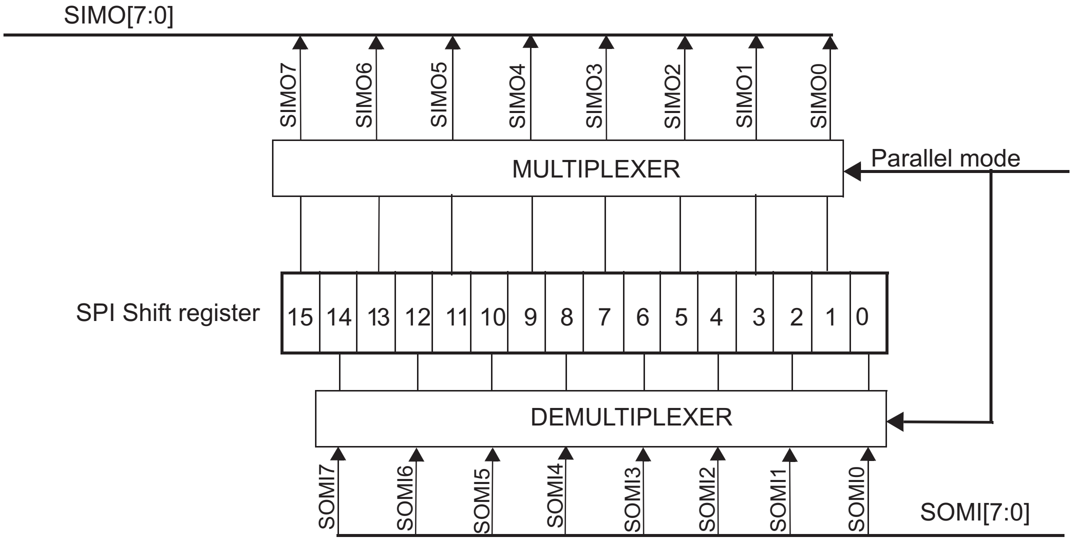
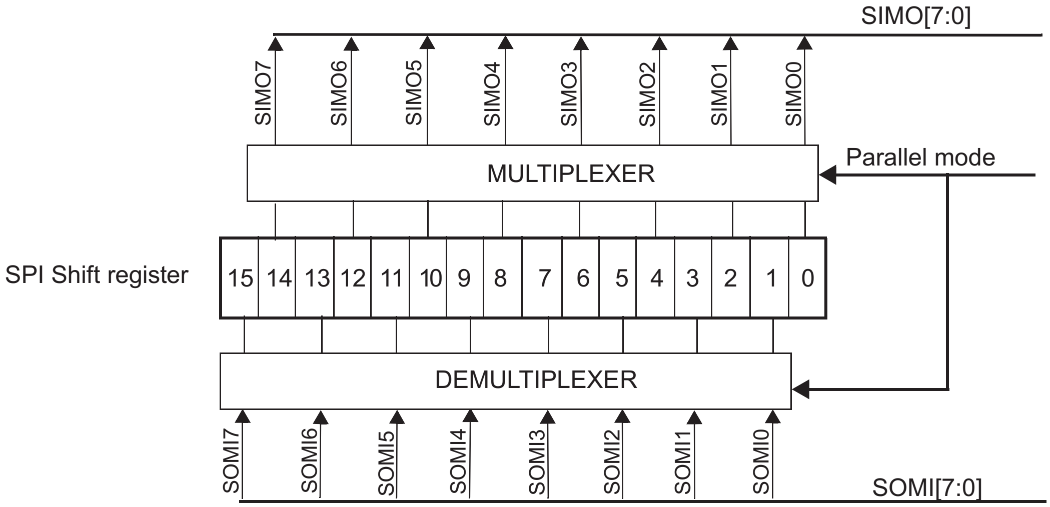


Figure 28-23. Block Diagram Shift Register, LSB First



28.2.6.6.2 Parallel Mode Pin Mapping, MSB First

Table 28-4 and Table 28-5 describe the SOMI and SIMO pin mapping when the SPI is used in parallel mode (1, 2, 4, 8) pin mode, MSB first.

NOTE: MSB-first or LSB-first can be configured using the SHIFTDIRx bit of the SPIFMTx registers.

Table 28-4. Pin Mapping for SIMO Pin with MSB First

Parallel Mode	Shift Register Bit	SIMO[7:0]
1	15	0
2	15	1
	7	0
4	15	3
	11	2
	7	1
	3	0
8	15	7
	13	6
	11	5
	9	4
	7	3
	5	2
	3	1
	1	0

Table 28-5. Pin Mapping for SOMI Pin with MSB First

Parallel Mode	Shift Register Bit	SOMI[7:0]
1	0	0
2	0	0
	8	1
4	0	0
	4	1
	8	2
	12	3
8	0	0
	2	1
	4	2
	6	3
	8	4
	10	5
	12	6
	14	7

28.2.6.6.3 Parallel Mode Pin Mapping, MSB-First, LSB-First

Table 28-6 and Table 28-7 describe the SIMO and SOMI pin mapping when SPI is used in parallel mode (1, 2, 4, 8) pin mode, LSB first.

Table 28-6. Pin Mapping for SIMO Pin with LSB First

Parallel Mode	Shift Register Bit	SIMO[7:0]
1	0	0
2	8	1
	0	0
4	12	3
	8	2
	4	1
	0	0
8	14	7
	12	6
	10	5
	8	4
	6	3
	4	2
	2	1
	0	0

Table 28-7. Pin Mapping for SOMI Pin with LSB First

Parallel Mode	Shift Register Bit	SOMI[7:0]
1	15	0
2	7	0
	15	1
4	3	0
	7	1
	11	2
	15	3
8	1	0
	3	1
	5	2
	7	3
	9	4
	11	5
	13	6
	15	7

28.2.6.6.4 2-Data Line Mode (MSB First, Phase 0, Polarity 0)

In 2-data line mode (master mode) the shift register bits 15 and 7 will be connected to the pins SIMO[1] and SIMO[0], and the shift register bits 8 and 0 will be connected to the pins SOMI[1] and SOMI[0] or vice versa in slave mode. After writing to the SPIDAT0/SPIDAT1 register, the bits 15 and 7 will be output on SIMO[1] and SIMO[0] on the rising edge if SPICLK. With the falling clock edge of the SPICLK, the received data on SOMI[1] and SOMI[0] will be latched to the shift register bits 8 and 0. The subsequent rising edge of SPICLK will shift the data in the shift register by 1 bit to the left. (SIMO[1] will shift the data out from bit 15 to 8, SIMO[0] will shift the data out from bit 7 to 0). After eight SPICLK cycles, when the full data word is transferred, the shift register (16 bits) is copied to the receive buffer, and the RXINT flag will be set. Figure 28-24 shows the clock /data diagram of the 2-data line mode. Figure 28-25 shows the timing of a two-pin parallel transfer.

Figure 28-24. 2-data Line Mode (Phase 0, Polarity 0)

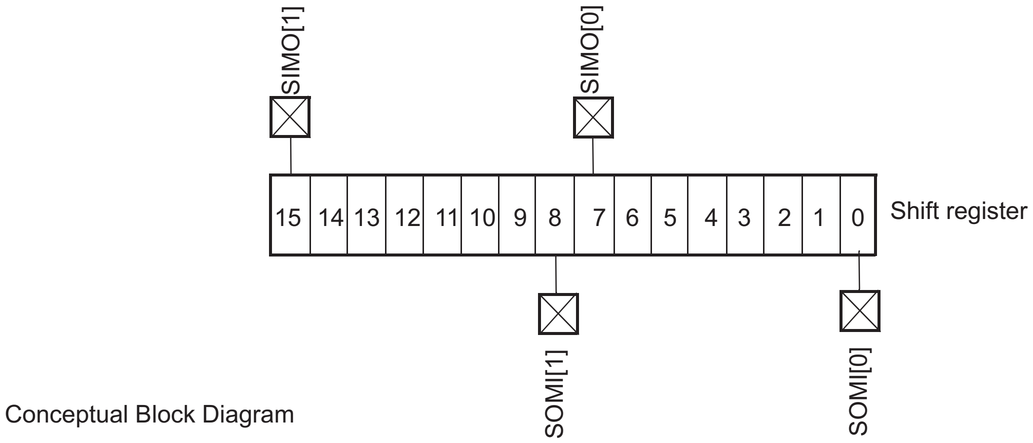
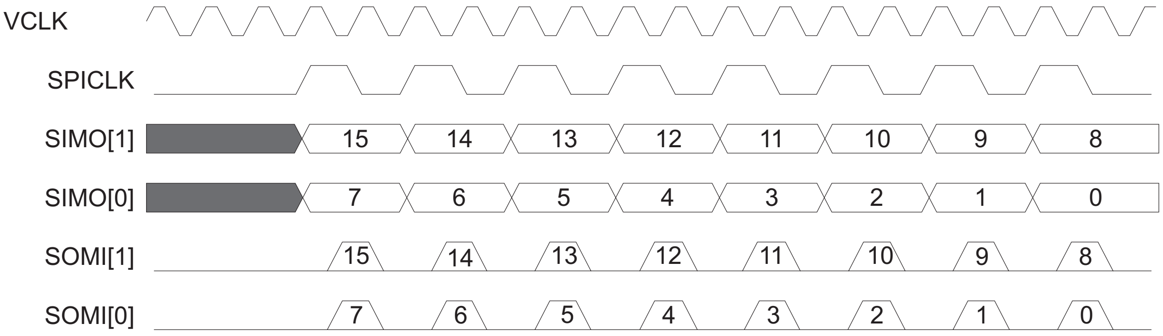


Figure 28-25. Two-Pin Parallel Mode Timing Diagram (Phase 0, Polarity 0)



28.2.6.6.5 4-Data Line Mode (MSB First, Phase 0, Polarity 0)

In 4-data line mode (master mode) the shift register bits 15, 11, 7, and 3 will be connected to the pins SIMO[3], SIMO[2], SIMO[1], and SIMO[0], and the shift register bits 12, 8, 4, and 0 will be connected to the pins SOMI[3], SOMI[2], SOMI[1], and SOMI[0] (or vice versa in slave mode). After writing to SPIDAT1/SPIDAT0, the bits 15, 11, 7, and 3 will be output on SIMO[3], SIMO[2], SIMO[1], and SIMO[0] on the rising edge of SPICLK. With the falling clock edge of the SPICLK, the received data on SOMI[3], SOMI[2], SOMI[1] and SOMI[0] will be latched to shift register bits 12, 8, 4, and 0. The subsequent rising edge of SPICLK will shift data in the shift register by 1 bit to the left (SIMO[3] will shift the data out from bit 15 to 12, SIMO[2] will shift the data out from bit 11 to 8, SIMO[1] will shift the data out from bit 7 to 4, SIMO[0] will shift the data out from bit 3 to 0). After four SPICLK cycles, when the full data word is transferred, the shift register (16 bits) is copied to the receive buffer, and the RXINT flag will be set.

Figure 28-26 shows the clock/data diagram of the four-data line mode. Figure 28-27, shows the timing diagram for four-data line mode.

Figure 28-26. 4-Data Line Mode (Phase 0, Polarity 0)

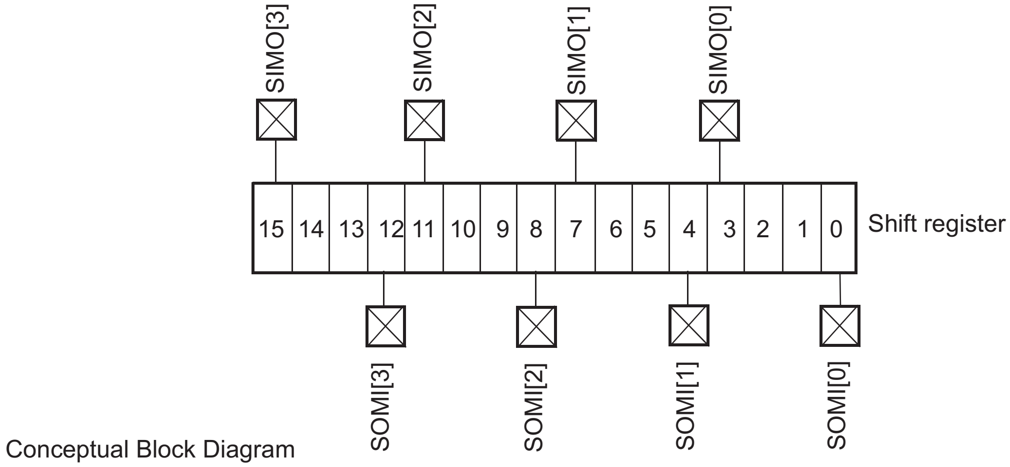
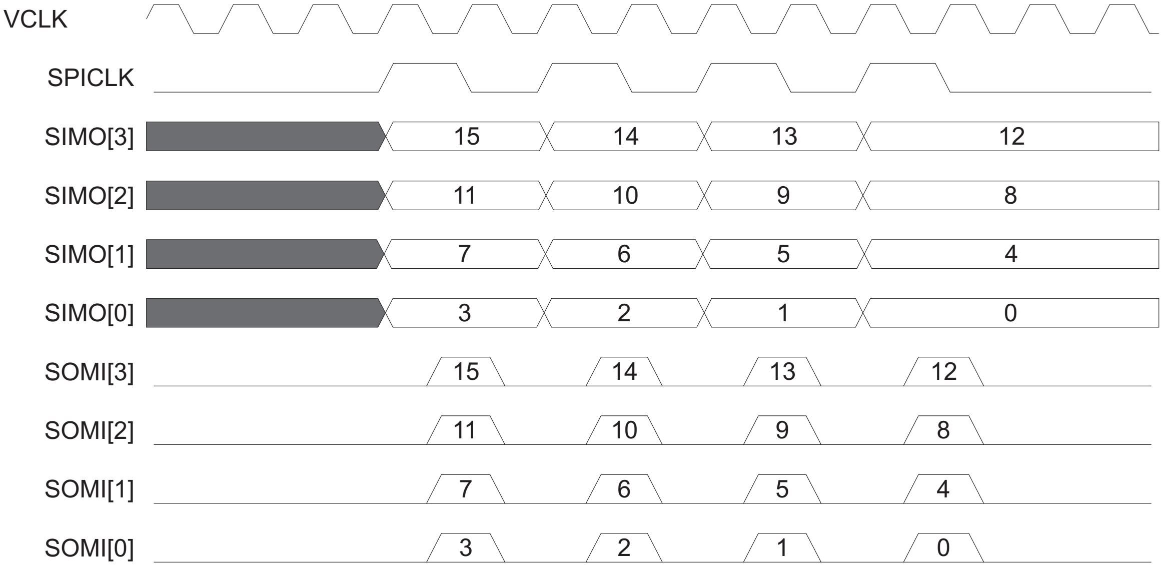


Figure 28-27. 4 Pins Parallel Mode Timing Diagram (Phase 0, Polarity 0)



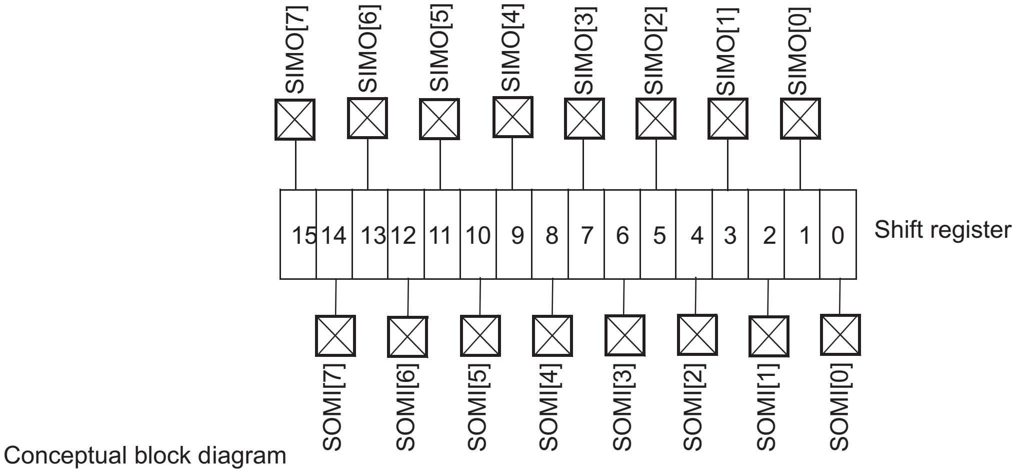
28.2.6.6.6 8-Data Line Mode (MSB First, Phase 0, Polarity 0)

In 8-data line mode (master mode) the shift register bits 15, 13, 11, 9, 7, 5 and 3 will be connected to the pins SIMO[7], SIMO[6], SIMO[5], SIMO[4], SIMO[3], SIMO[2], SIMO[1], and SIMO[0], and the shift-register bits 14, 12, 10, 8, 6, 4, and 0 will be connected to the pins SOMI[7], SOMI[6], SOMI[5], SOMI[4], SOMI[3], SOMI[2], SOMI[1], and SOMI[0] (or vice versa in slave mode).

After writing to SPIDAT0/SPIDAT1, the bits 15, 13, 11, 9, 7, 5, 3, and 1 will be output on SIMO[7], SIMO[6], SIMO[5], SIMO[4], SIMO[3], SIMO[2], SIMO[1], and SIMO[0], on the rising edge of SPICLK. On the falling clock edge of the SPICLK, the received data on SOMI[8], SOMI[7], SOMI[6],SOMI[5], SOMI[4], SOMI[3], SOMI[2], SOMI[1], and SOMI[0] will be latched to the shift register bits 14, 12, 10, 8, 6, 4, 2, and 0.

The subsequent rising edge of SPICLK will shift the data in the shift register by 1 bit to the left. After two SPICLK cycles, when the full data word is transferred the shift register (16 bits) is copied to the receive buffer, and the RXINT flag will be set. Figure 28-28 shows the clock/data diagram of the 8-data line mode. Figure 28-29 shows the pin timings for 8-data line mode.

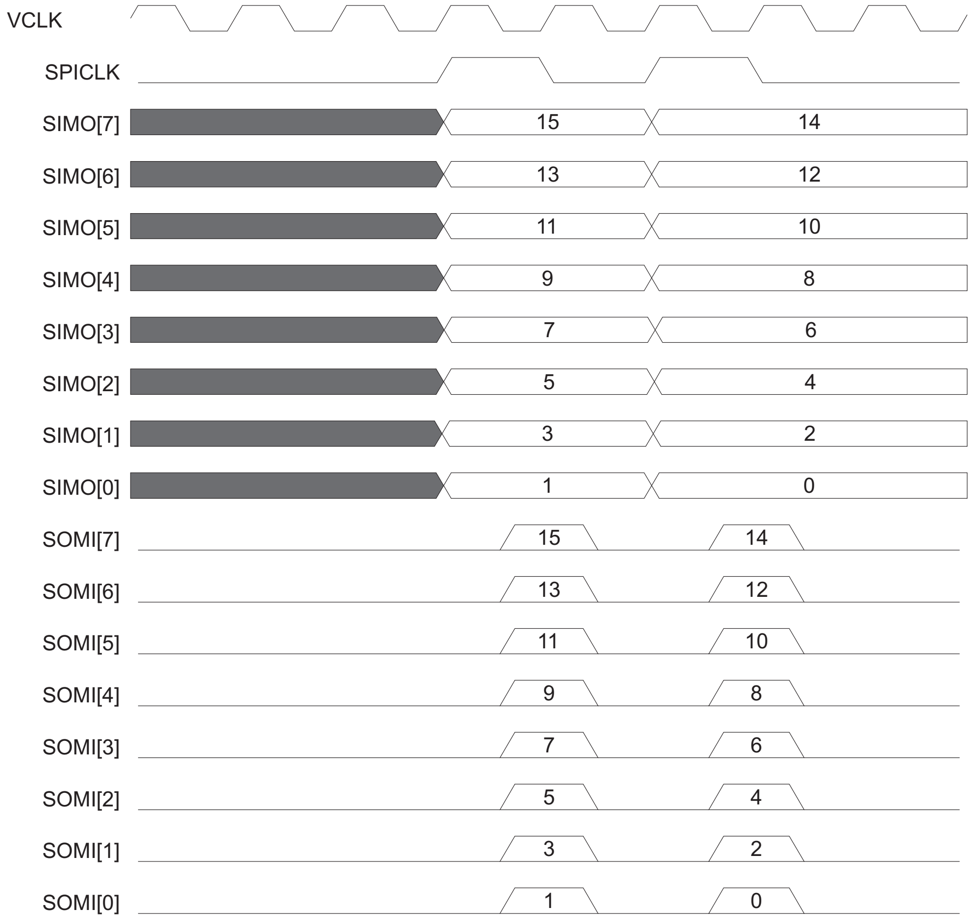
Figure 28-28. 8-data Line Mode (Phase 0, Polarity 0)



NOTE: Parity Support

Using the parity support in parallel mode may seriously affect throughput. For an eight-line mode to transfer 16 bits of data, only two SPICLK pulses are enough. If parity is enabled, one extra SPICLK pulse will be used to transfer and receive the parity bit. Parity will be transmitted and received on the 0th line regardless of 1/2/4/8-line modes. During the parity bit transfer, other data bits are not valid.

Figure 28-29. 8 Pins Parallel Mode Timing Diagram (Phase 0, Polarity 0)



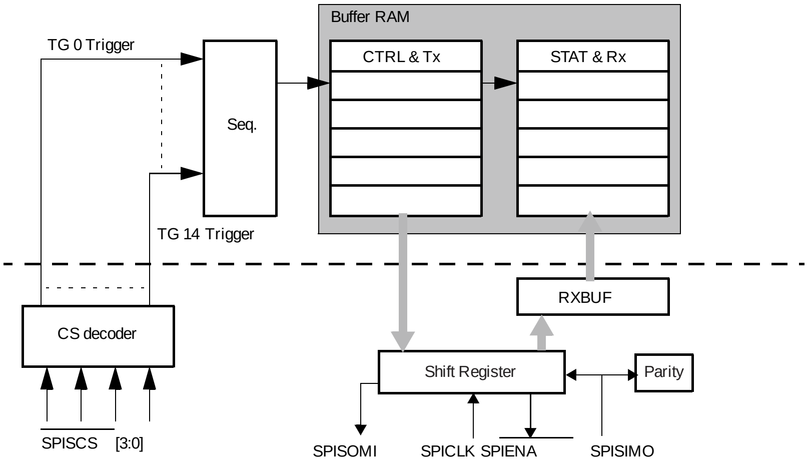
NOTE: Modulo Count Parallel Mode is not supported in this device.

28.2.6.7 MibSPI Slave in Multi-buffer Configuration

When operating in slave mode, the MibSPI uses the chip-select pins 0 to 3 to generate a trigger to the corresponding Transfer Group. For example, putting 0000 on the chip-select pins triggers Transfer Groups 0 and putting 0001 triggers TG1. When the value 1111 is set to the chip-select, the MibSPI is deselected, that is Transfer Group 15 is not available in slave mode. The remaining chip-select pins should stay in GPIO mode. In slave mode, the fields like trigger source and trigger event are not taken into account by the sequencer. Only the $\overline{\text{SPICS}}$ pins can trigger a Transfer Group. The chip-select trigger operates as a level-sensitive trigger. However, when the MibSPI is in 3-pin or 4-pin with $\overline{\text{SPIENA}}$ mode, just one Transfer Group can be triggered and it is restricted to Transfer Group 0 (TG0). In slave mode, the PRST field should be cleared to 0. If the corresponding Transfer Group is enabled, the multi-buffer reads the current buffer of the TG and writes it into SPIDAT1. If Transfer Group is disabled, the multi-buffer does not update the SPIDAT1 register.

NOTE: If the selected Transfer Group is disabled and no update of the SPIDAT1 register has been done, the data to be transferred is meaningless. Even the received data will not be copied to the multi-buffer RAM. However it will be available on SPIBUF register until it is overwritten by the subsequent receive data.

Figure 28-30. Multi-buffer in Slave Mode



When the SPIDAT1 register is updated, the enable signal is released, and the transaction could begin. If the enable signal is not used, the master should wait for 6 VCLK cycles before sending the clock to begin the transaction. This time allows the MibSPI to update the SPIDAT1 register.

Once the transaction is finished, the MibSPI writes back the content of the shift-register into the Rx buffer and updates the status field.

NOTE: If all the Transfer Groups are not needed, the number of $\overline{\text{SPICS}}$ pins that need to be in functional mode could be reduced to 3, 2, or 1 by using the SPIPC0 register. In these cases, the maximum number of Transfer Group accessible are, respectively 7, 3, and 1. The pins that are set in GPIO mode are not decoded.

MibSPI in 3-pin and 4-pin (with $\overline{\text{SPIENA}}$) mode also supports multi-buffer mode. However, it is restricted to having just one transfer group, Transfer Group 0 (TG0). The entire multi-buffer RAM can be configured for TG0 alone. The PSTART field in TG1CTRL register should be used to configure the size of the multi-buffer (end of the buffers) for TG0.

NOTE: The maximum input frequency on the SPICLK pin when in slave mode is VCLK frequency /2. If the Slave is configured in either 3-pin or 4-pin (without $\overline{\text{SPIENA}}$) modes, then, between end of last SPICLK and the start of SPICLK for next buffer, there should be at least 6 VCLK cycles of delay.

28.2.6.8 Transfer Groups

The size of the multi-buffer RAM depends on the implementation. It comprises 0 to 128/256 buffers, whereas 0 buffers considers the special case of no multi-buffer RAM. Each entry in the multi-buffer RAM consists of 4 parts: a 16-bit transmit field, a 16-bit receive field, a 16-bit control field and a 16-bit status field. The multi-buffer RAM can be partitioned into multiple transfer group with variable number of buffers each.

28.2.6.8.1 Configuring Transfer Groups and Trigger Events

Each TG can be configured via one dedicated control register, TGxCTRL. This register even configures the trigger events for the transfer group. The register is described in Section 28.3.34. The actual number of available control registers varies by device.

28.2.6.8.2 Sequencer-Which Handled the Sequencing of Triggered Transfer Groups

Sequencer(FSM) controls the data flow from the multi-buffer RAM to the Shift Register. The Multi-buffer Control Logic has arbitration logic between VBUS and the Sequencer accessing the multi-buffer RAM. Sequencer picks up a highest priority Transfer Group from among the active TGs to be serviced. For the selected TG the starting buffer to be transferred is obtained from the PSTART of the respective TGxCTRL register.

Sequencer requests for the selected buffer through the Multi-buffer Control Logic, and once it receives the data, it reads the control fields to determine the subsequent action. Once the buffer is determined to be ready for transfer, the data is written to the TX SHIFT REGISTER by the Sequencer. This triggers the Kernel FSM to initiate the SPI transfer.

28.2.6.8.3 Inter-group Prioritization and Arbitration

Transfer Group0 (TG0) has the highest priority and TG15 has the lowest priority among the transfer groups TG0 to TG15.Where as under the following conditions under the following conditions a lower priority Transfer Group cannot be interrupted by a higher priority TG.

- When there’s a CSHOLD or LOCK buffer, until the completion of the next buffer transfer which is a non-CSHOLD or non-LOCK buffer, the Transfer Group cannot be interrupted by any higher priority TGs.
- An entire sequence of buffer transfer for NOBRK DMA buffer cannot be interrupted by any higher priority TG.
- Once the last buffer in a Transfer Group is prefetched, a higher priority TG cannot interrupt it until the completion of the Transfer Group.

These prioritizations made among the transfer groups also decide the arbitration logic among the multiple transfer groups which are active

28.2.6.8.4 Transmission Lock Capability

Some slave devices require to have “command” followed by “data”. In this case the SPI transaction should not be interrupted by another group transfer. The LOCK bit within each buffer allows consecutive transfer to happen without being interrupted by another higher priority group transfer.

28.2.7 General-Purpose I/O

All of the SPI pins may be programmed via the SPIPCx control registers to be either functional or general-purpose I/O pins.

If the SPI function is to be used, application software must ensure that at least the SPICLK pin and the SOMI and/or SIMO pins are configured as SPI functional pins, and not as GIO pins, or else the SPI state machine will be held in reset, preventing SPI transactions.

SPI pins support:

- internal pull-up resistors
- internal pull-down resistors
- open-drain or push-pull mode
- input-buffer enabling/disabling (controlled by the PULDIS and PSEL bits)

28.2.8 Low-Power Mode

The SPI can be put into either local or global low-power mode. Global low-power mode is asserted by the system and is not controlled by the SPI. During global low-power mode, all clocks to the SPI are turned off, making the module completely inactive.

Local low-power mode is asserted by setting the POWERDOWN (SPIGCR1[8]) bit; setting this bit stops the clocks to the SPI internal logic and registers. Setting the POWERDOWN bit causes the SPI to enter local low-power mode and clearing the POWERDOWN bit causes SPI to exit from local low-power mode. All registers remain accessible during local power-down mode, since the clock to the SPI registers is temporarily re-enabled for each access. RAM buffers are also accessible during low power mode.

NOTE: Since entering a low-power mode has the effect of suspending all state-machine activities, care must be taken when entering such modes to ensure that a valid state is entered when low-power mode is active. Application software must ensure that a low power mode is not entered during a data transfer.

28.2.9 Safety Features

28.2.9.1 Detection of Slave Desynchronization (Master Only)

When a slave supports generation of an enable signal (ENA), desynchronization can be detected. With the enable signal a slave indicates to the master that it is ready to exchange data. A desynchronization can occur if one or more clock edges are missed by the slave. In this case the slave may block the SOMI line until it detects clock edges corresponding to the next data word. This would corrupt the data word of the desynchronized slave and the consecutive data word. A configurable 8-bit time-out counter (T2EDELAY), which is clocked with SPICLK, is implemented to detect this slave malfunction. After the transmission has finished (end of last bit transferred: either last data bit or parity bit) the counter is started. If the ENA signal generated by the slave does not become inactive before the counter overflows, the DESYNC flag is set and an interrupt is asserted (if enabled).

NOTE: Inconsistency of Desynchronization Flag in Compatibility Mode MibSPI

Because of the nature of this error, under some circumstances it is possible for a desync error detected for the previous buffer to be visible in the current buffer. This is due to the fact that receive completion flag/interrupt will be generated when the buffer transfer is completed. But desync will be detected after the buffer transfer is completed. So, if VBUS master reads the received data quickly when an RXINT is detected, then the status flag may not reflect the correct desynchronization condition. This inconsistency in the desync flag is valid only in compatibility mode of MibSPI. In multi-buffer mode, the desync flag is always assured to be for the current buffer.

28.2.9.2 ENA Signal Time-Out (Master Only)

The SPI in master mode waits for the hardware handshake signal (ENA) coming from the addressed slave before performing a data transfer. To avoid stalling the SPI by a non-responsive slave device, a time-out value can be configured using C2EDELAY. If the time-out counter overflows before an active ENA signal is sampled, the TIMEOUT flag in the status register SPIFLG is set and the TIMEOUT flag in the status field of the corresponding buffer is set.

NOTE: When the chip select signal becomes active, no breaks in transmission are allowed. The next arbitration is performed while waiting for the time-out to occur.

28.2.9.3 Data-Length Error

An SPI can generate an error flag by detecting any mismatch in length of received or transmitted data and the programmed character length under certain conditions.

Data-Length Error in Master Mode: During a data transfer, if the SPI detects a de-assertion of the $\overline{\text{SPIENA}}$ pin (by the slave) while the character counter is not overflowed, then an error flag is set to indicate a data-length error. This can be caused by a slave receiving extra clocks (for example, due to noise on the SPICLK line).

NOTE: In a master mode SPI, the data length error will be generated only if the $\overline{\text{SPIENA}}$ pin is enabled as a functional pin.

Data-Length Error in Slave Mode: During a transfer, if the SPI detects a de-assertion of the $\overline{\text{SPICS}}$ pin before its character length counter overflows, then an error flag is set to indicate a data-length error. This situation can arise If the slave SPI misses one or more SPICLK pulses from the master. This error in slave mode implies that both the transmitted and received data were not complete.

NOTE: In a slave-mode SPI, the data-length error flag will be generated only if at least one of the $\overline{\text{SPICS}}$ pins are configured as functional, and are being used for selecting the slave.

28.2.9.4 Continuous Self-Test (Master/Slave)

During data transfer, the SPI compares its own internal transmit data with its transmit data on the bus. The sample point for the compare is at one-half SPI clock after transmit point. If the data on the bus does not match the expected value, the bit-error (BITERR) flag is set and an interrupt is asserted if enabled.

NOTE: The compare is made from the output pin using its input buffer.

28.2.10 Test Features

28.2.10.1 Internal Loop-Back Test Mode (Master Only)

The internal loop-back self-test mode can be utilized to test the SPI transmit and receive paths, including the shift registers, the SPI buffer registers, and the parity generator. In this mode the transmit signal is internally feedback to the receiver, whereas the SIMO, SOMI, and CLK pin are disconnected; that is, the transmitted data is internally transferred to the corresponding receive buffer while external signals remain unchanged.

This mode allows the CPU to write into the transmit buffer, and check that the receive buffer contains the correct transmit data. If an error occurs the corresponding error is set within the status field.

NOTE: This mode cannot be changed during transmission.

28.2.10.2 Input/Output Loopback Test Mode

Input/Output Loopback Test mode supports the testing of all Input/Output pins without the aid of an external interface. Loopback can be configured as either analog-loopback (loopback through the pin-level input/output buffers) or digital loopback (internal to the SPI module). With Input/Output Loopback, all functional features of the SPI can be tested. Transmit data is fed back through the receive-data line(s). See Figure 28-31 for a diagram of the types of feedback available. The IOLPBKTSTCR register defines all of the available control fields.

In loopback mode, it is also possible to induce various error conditions. See Section 28.3.43 for details of the register field controlling these features.

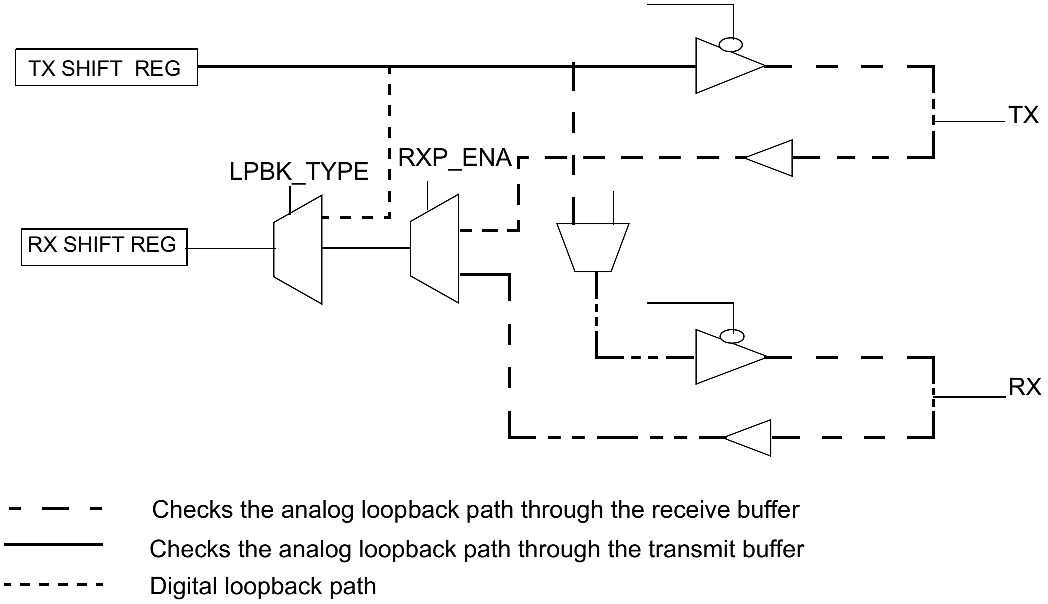
In Input/Output loopback test modes, even when the module is in slave mode, the SPICLK is generated internally. This SPICLK is used for all loopback-mode SPI transactions. Slave-mode features can be tested without the help of another master SPI, using the internally-generated SPICLK. Chip selects are also generated by the slave itself while it is in Input/Output loopback mode.

In Input/Output loopback test modes, if the module is in master mode, the $\overline{\text{ENA}}$ signal is also generated by internal logic so that an external interface is not required.

NOTE: **Usage Guideline for Input/Output Loopback**

Input/Output Loopback mode should be used with caution because, in some configurations, even the receive pins will be driven with transmit data. During testing, it should be ensured that none of the SPI pins are driven by any other device connected to them. Otherwise, if analog loopback is selected in I/O Loopback mode, then testing may damage the device.

Figure 28-31. I/O Paths During I/O Loopback Modes


TX SHIFT REG
RX SHIFT REG
LPBK_TYPE
RXP_ENA
TX
RX
Checks the analog loopback path through the receive buffer
Checks the analog loopback path through the transmit buffer
Digital loopback path

This diagram is intended to illustrate loopback paths and therefore may omit some normal-mode paths.

28.2.10.2.1 Input/Output Loopback Mode Operation in Slave Mode

In multi-buffer slave mode, there are some additional requirements for using I/O loopback mode (IOLPBK). In multi-buffer slave mode, the chip-select pins are the triggers for various TGs. Enabling the IOLPBK mode by writing 0xA to the IOLPBTSTENA bits of the IOLPBKTSTCR register triggers TG0 by driving $\overline{\text{SPICS}}$ to 0. The actual number of chip selects can be programmed to have any or all of the $\overline{\text{SPICS}}$ pins as functional. All other configurations should be completed before enabling the IOLPBK mode in multi-buffer slave mode since it triggers TG0.

After the first buffer transfer is completed, the CSNR field of the current buffer is used to trigger the next buffer. So, if multiple TGs are desired to be tested, then the CSNR field of the final buffer in each TG should hold the number of the next TG to be triggered. As long as TG boundaries are well defined and are enabled, the completion of one TG will trigger the next TG.

To stop the transfer in multi-buffer slave mode in I/O Loopback configuration, either IOLPBK mode can be disabled by writing 0x5 to the IOLPBTSTENA bits or all of the TGs can be disabled.

28.2.11 Module Configuration

MibSPI/MibSPIP can be configured to function as Normal SPI and Multi-buffered SPI. Upon power-up or a system-level reset, each bit in the module registers is set to a default state. The registers are writable only after the RESET bit is set to 1.

28.2.11.1 Compatibility (SPI) Mode Configuration

The following list details the configuration steps that software should perform prior to the transmission or reception of data. As long as the SPIEN bit in the Global Control Register 1 (SPIGCR1) is cleared to 0 the entire time that the SPI is being configured, the order in which the registers are programmed is not important.

- Enable SPI by setting RESET bit.
- Configure the SIMO, SOMI, CLK, and optional $\overline{\text{SPICS}}$ and $\overline{\text{SPIENA}}$ pins for SPI functionality by setting the corresponding bit in SPIPC0 register.
- Configure the module to function as Master or Slave using CLKMOD and MASTER bits.
- Configure the required SPI data format using SPIFMTx register.
- If the module is selected to function as Master, the delay parameters can be configured using SPIDELAY register.
- Enable the Interrupts using SPIINT0 register if required.
- Select the chip select to be used by setting CSNR bits in SPIDAT1 register.
- Configure CSHOLD and WDEL bits in SPIDAT1 register if required.
- Select the Data word format by setting DFSEL bits. Select the Number of the configured SPIFMTx register (0 to 3) to used for the communication.
- Set LOOPBACK bit to connect the transmitter to the receiver internally. (This feature is used to perform a self-test. Do not configure for normal communication to external devices).
- Set SPIEN bit to 1 after the SPI is configured.
- Perform Transmit and receive data, using SPIDAT1 and SPIBUF register.
- You must wait for TXFULL to reset or TXINT before writing next data to SPIDAT1 register.
- You must wait for RXEMPTY to reset or RXINT before reading the data from SPIBUF register.

28.2.11.2 MibSPI Mode Configuration

The following list details the configuration steps that software should perform prior to the transmission or reception of data in MIBSPI mode. As long as the SPIEN bit in the Global Control Register 1 (SPIGCR1) is cleared to 0 the entire time that the SPI is being configured, the order in which the registers are programmed is not important.

- Enable SPI by setting RESET bit.
- Set MSPIENA bit to 1 to get access to multi-buffer mode registers.
- Configure the SIMO, SOMI, CLK, and optional $\overline{\text{SPICS}}$ and $\overline{\text{SPIENA}}$ pins for SPI functionality by setting the corresponding bit in SPIPC0 register.
- Configure the module to function as Master or Slave using CLKMOD and MASTER bits.
- Configure the required SPI data format using SPIFMTx register.
- If the module is selected to function as Master, the delay parameters can be configured using SPIDELAY register.
- Check for BUFINITACTIVE bit to be active before configuring MIBSPI RAM. (From Device Power On it take Number of Buffers × Peripheral clock period to initialize complete RAM.)
- Enable the Transfer Group interrupts using TGITENST register if required.
- Enable error interrupts using SPIINT0 register if required.
- Set SPIEN bit to 1 after the SPI is configured.
- The Trigger Source, Trigger Event, Transfer Group start address for the corresponding Transfer groups can be configured using the corresponding TGxCTRL register.
- Configure LPEND to specify the end address of the last TG.
- Similar to SPIDAT1 register, the 16 bit control fields in every TXRAM buffer in the TG have to be configured.
- Configure one of the eight BUFMODE available for each buffer.
- Fill the data to be transmitted in TXDATA field in TXRAM buffers.
- Configure TGENA bit to enable the required Transfer groups. (In case of Trigger event always setting TGENA will trigger the transfer group).
- At the occurrence of the correct trigger event, the Transfer group will be triggered and data gets transmitted and received one after the other with out any CPU intervention.
- You can poll Transfer group interrupt flag or wait for a transfer-completed interrupt to read and write new data to the buffers.

28.3 Control Registers

This section describes the SPI control, data, and pin registers. The registers support 8-bit, 16-bit and 32-bit writes. The offset is relative to the associated base address of this module in a system. The base address for the control registers is FFF7 F400h for MibSPI1, FFF7 F600h for MibSPI2, FFF7 F800h for MibSPI3, FFF7 FA00h for MibSPI4, and FFF7 FC00h for MibSPI5.

NOTE: TI highly recommends that write values corresponding to the reserved locations of registers be maintained as 0 consistently. This allows future enhancements to use these reserved bits as control bits without affecting the functionality of the module with any older versions of software.

Table 28-8. SPI Registers

Offset	Acronym	Register Description	Section
00h	SPIGCR0	SPI Global Control Register 0	Section 28.3.1
04h	SPIGCR1	SPI Global Control Register 1	Section 28.3.2
08h	SPIINT0	SPI Interrupt Register	Section 28.3.3
0Ch	SPILVL	SPI Interrupt Level Register	Section 28.3.4
10h	SPIFLG	SPI Flag Register	Section 28.3.5
14h	SPIPC0	SPI Pin Control Register 0	Section 28.3.6
18h	SPIPC1	SPI Pin Control Register 1	Section 28.3.7
1Ch	SPIPC2	SPI Pin Control Register 2	Section 28.3.8
20h	SPIPC3	SPI Pin Control Register 3	Section 28.3.9
24h	SPIPC4	SPI Pin Control Register 4	Section 28.3.10
28h	SPIPC5	SPI Pin Control Register 5	Section 28.3.11
2Ch	SPIPC6	SPI Pin Control Register 6	Section 28.3.12
30h	SPIPC7	SPI Pin Control Register 7	Section 28.3.13
34h	SPIPC8	SPI Pin Control Register 8	Section 28.3.14
38h	SPIDAT0	SPI Transmit Data Register 0	Section 28.3.15
3Ch	SPIDAT1	SPI Transmit Data Register 1	Section 28.3.16
40h	SPIBUF	SPI Receive Buffer Register	Section 28.3.17
44h	SPIEMU	SPI Emulation Register	Section 28.3.18
48h	SPIDELAY	SPI Delay Register	Section 28.3.19
4Ch	SPIDEF	SPI Default Chip Select Register	Section 28.3.20
50h-5Ch	SPIFMT0-SPIFMT3	SPI Data Format Registers	Section 28.3.21
60h	INTVECT0	Interrupt Vector 0	Section 28.3.22
64h	INTVECT1	Interrupt Vector 1	Section 28.3.23
68h	SPIPC9[1]	SPI Pin Control Register 9	Section 28.3.24
6Ch	SPIPMCTRL	Parallel/Modulo Mode Control Register	Section 28.3.25
70h	MIBSPIE	Multi-buffer Mode Enable Register	Section 28.3.26
74h	TGITENST	TG Interrupt Enable Set Register	Section 28.3.27
78h	TGITENCR	TG Interrupt Enable Clear Register	Section 28.3.28
7Ch	TGITLVST	Transfer Group Interrupt Level Set Register	Section 28.3.29
80h	TGITLVCR	Transfer Group Interrupt Level Clear Register	Section 28.3.30
84h	TGINTFLG	Transfer Group Interrupt Flag Register	Section 28.3.31
90h	TICKCNT	Tick Count Register	Section 28.3.32
94h	LTGPEND	Last TG End Pointer	Section 28.3.33
98h-D4h	TGxCTRL	TGx Control Registers	Section 28.3.34
D8h-F4h	DMAxCTRL	DMA Channel Control Registers	Section 28.3.35
F8h-114h	ICOUNT	DMAxCOUNT Registers	Section 28.3.36

[1] SPIPC9 only applies to SPI2.

Table 28-8. SPI Registers (continued)

Offset	Acronym	Register Description	Section
118h	DMACNTLEN	DMA Large Count Register	Section 28.3.37
120h	PAR_ECC_CTRL	Parity/ECC Control Register	Section 28.3.38
124h	PAR_ECC_STAT	Parity/ECC Status Register	Section 28.3.39
128h	UERRADDR1	Uncorrectable Parity or Double-Bit ECC Error Address Register - RXRAM	Section 28.3.40
12Ch	UERRADDR0	Uncorrectable Parity or Double-Bit ECC Error Address Register - TXRAM	Section 28.3.41
130h	RXOVRN_BUF_ADDR	RXRAM Overrun Buffer Address Register	Section 28.3.42
134h	IOLPBKTSTCR	I/O Loopback Test Control Register	Section 28.3.43
138h	EXTENDED_PRESCALE1	SPI Extended Prescale Register 1	Section 28.3.44
13Ch	EXTENDED_PRESCALE2	SPI Extended Prescale Register 2	Section 28.3.45
140h	ECCDIAG_CTRL	ECC Diagnostic Control Register	Section 28.3.46
144h	ECCDIAG_STAT	ECC Diagnostic Status Register	Section 28.3.47
148h	SBERRADDR1	Single-Bit Error Address Register - RXRAM	Section 28.3.48
152h	SBERRADDR0	Single-Bit Error Address Register - TXRAM	Section 28.3.49

28.3.1 SPI Global Control Register 0 (SPIGCR0)

Figure 28-32. SPI Global Control Register 0 (SPIGCR0) [offset = 00h]

31	16
Reserved	
R-0	

15	1	0
Reserved		nRESET
R-0		R/WP-0

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privilege mode only; -*n* = value after reset

Table 28-9. SPI Global Control Register 0 (SPIGCR0) Field Descriptions

Bit	Field	Value	Description
31-1	Reserved	0	Reads return 0. Writes have no effect.
0	nRESET		This is the local reset control for the module. This bit needs to be set to 1 before any operation on SPI / MibSPI can be done. Only after setting this bit to 1, the Auto Initialization of Multi-buffer RAM starts. Clearing this bit to 0 will result in all of the control and status register values to return to their default values..
		0	SPI is in the reset state.
		1	SPI is out of the reset state.

28.3.2 SPI Global Control Register 1 (SPIGCR1)

Figure 28-33. SPI Global Control Register 1 (SPIGCR1) [offset = 04h]

31 25	24	23 17	16
Reserved	SPIEN	Reserved	LOOPBACK
R-0	R/W-0	R-0	R/WP-0

15 9	8	7 2	1	0
Reserved	POWERDOWN	Reserved	CLKMOD	MASTER
R-0	R/W-0	R-0	R/W-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privilege mode only; -*n* = value after reset

Table 28-10. SPI Global Control Register 1 (SPIGCR1) Field Descriptions

Bit	Field	Value	Description
31-25	Reserved	0	Reads return 0. Writes have no effect.
24	SPIEN		SPI enable. This bit enables SPI transfers. This bit must be set to 1 after all other SPI configuration bits have been written. When the SPIEN bit is 0 or cleared to 0, the following SPI registers get forced to their default states: • Both TX and RX shift registers • The TXDATA fields of the SPI Transmit Data Register 0 (SPIDAT0) and the SPI Transmit Data Register 1 (SPIDAT1) • All the fields of the SPI Flag Register (SPIFLG) • Contents of SPIBUF and the internal RXBUF registers
		0	The SPI is not activated for transfers.
		1	Activates SPI.
23-17	Reserved	0	Reads return 0. Writes have no effect.
16	LOOPBACK		Internal loop-back test mode. The internal self-test option can be enabled by setting this bit. If the SPISIMO and SPISOMI pins are configured with SPI functionality, then the SPISIMO[7:0] pins are internally connected to the SPISOMI[7:0] pins (transmit data is looped back as receive data). GIO mode for these pins is not supported in loopback mode. Externally, during loop-back operation, the SPICLK pin outputs an inactive value and SPISOMI[7:0] remains in the high-impedance state. If the SPI is initialized in slave mode or a data transfer is ongoing, errors may result. **Note: This loopback mode can only be used in master mode. Master mode must be selected before setting LOOPBACK. When this mode is selected, the CLKMOD bit should be set to 1, meaning that SPICLK is internally generated.**
		0	Internal loop-back test mode is disabled.
		1	Internal loop-back test mode is enabled.
15-9	Reserved	0	Reads return 0. Writes have no effect.
8	POWERDOWN		When active, the SPI state machine enters a power-down state.
		0	The SPI is in active mode.
		1	The SPI is in power-down mode.
7-2	Reserved	0	Reads return 0. Writes have no effect.
1	CLKMOD		Clock mode. This bit selects either an internal or external clock source. This bit also determines the I/O direction of the $\overline{\text{SPIENA}}$ and $\overline{\text{SPICS}}$ pins in functional mode.
		0	Clock is external. • $\overline{\text{SPIENA}}$ is an output. • $\overline{\text{SPICS}}$ are inputs.
		1	Clock is internally-generated. • $\overline{\text{SPIENA}}$ is an input. • $\overline{\text{SPICS}}$ are outputs.

Table 28-10. SPI Global Control Register 1 (SPIGCR1) Field Descriptions (continued)

Bit	Field	Value	Description
0	MASTER		SPISIMO/SPISOMI pin direction determination. Sets the direction of the SPISIMO and SPISOMI pins. **Note: For master-mode operation of the SPI, MASTER bit should be set to 1 and CLKMOD bit can be set either 1 or 0. The master-mode SPI can run on an external clock on SPICLK.** **For slave mode operation, both the MASTER and CLKMOD bits should be cleared to 0. Any other combinations may result in unpredictable behavior of the SPI. In slave mode. SPICLK will not be generated internally in slave mode.**
		0	SPISIMO[7:0] pins are inputs, SPISOMI[7:0] pins are outputs.
		1	SPISOMI[7:0] pins are inputs, SPISIMO[7:0] pins are outputs.

28.3.3 SPI Interrupt Register (SPIINT0)

Figure 28-34. SPI Interrupt Register (SPIINT0) [offset = 08h]

31-25	24
Reserved	ENABLEHIGHZ
R-0	R/W-0

23-17	16
Reserved	DMAREQEN
R-0	R/W-0

15-10	9	8
Reserved	TXINTENA	RXINTENA
R-0	R/W-0	R/W-0

7	6	5	4	3	2	1	0
Reserved	RXOVRNINT ENA	Reserved	BITERR ENA	DESYNC ENA	PARERR ENA	TIMEOUT ENA	DLENERR ENA
R-0	R/W-0	R-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 28-11. SPI Interrupt Register (SPIINT0) Field Descriptions

Bit	Field	Value	Description
31-25	Reserved	0	Reads return 0. Writes have no effect.
24	ENABLEHIGHZ		$\overline{\text{SPIENA}}$ pin high-impedance enable. When active, the $\overline{\text{SPIENA}}$ pin (when it is configured as a WAIT functional output signal in a slave SPI) is forced to high-impedance when not driving a low signal. If inactive, then the pin will output both a high and a low signal.
		0	$\overline{\text{SPIENA}}$ pin is pulled high when not active.
		1	$\overline{\text{SPIENA}}$ pin remains high-impedance when not active.
23-17	Reserved	0	Reads return 0. Writes have no effect.
16	DMAREQEN		DMA request enable. Enables the DMA request signal to be generated for both receive and transmit channels. Enable DMA REQ only after setting the SPIEN bit to 1.
		0	DMA is not used.
		1	DMA requests will be generated. **Note: A DMA request will be generated on the TX DMA REQ line each time a word is copied to the shift register either from TXBUF or directly from SPIDAT0/SPIDAT1 writes.** **Note: A DMA request will be generated on the RX DMA REQ line each time a word is copied to the SPIBUF register either from RXBUF or directly from the shift register.**
15-10	Reserved	0	Reads return 0. Writes have no effect.

Table 28-11. SPI Interrupt Register (SPIINT0) Field Descriptions (continued)

Bit	Field	Value	Description
9	TXINTENA		Causes an interrupt to be generated every time data is written to the shift register, so that the next word can be written to TXBUF. Setting this bit will generate an interrupt if the TXINTFLG bit (SPI Flag Register (SPIFLG)[9]) is set to 1.
		0	No interrupt will be generated upon TXINTFLG being set to 1.
		1	An interrupt will be generated upon TXINTFLG being set to 1.
			The transmitter empty interrupt is valid in compatibility mode of SPI only. In multi-buffered mode, this interrupts will not be generated, even if it is enabled.
			Note: An interrupt request will be generated as soon as this bit is set to 1. By default it will be generated on the INT0 line. The SPILVL register can be programmed to change the interrupt line.
8	RXINTENA		Causes an interrupt to be generated when the RXINTFLAG bit (SPI Flag Register (SPIFLG)[8]) is set by hardware.
		0	Interrupt will not be generated.
		1	Interrupt will be generated.
			The receiver full interrupt is valid in compatibility mode of SPI only. In multi-buffered mode, this interrupts will not be generated, even if it is enabled.
7	Reserved	0	Reads return 0. Writes have no effect.
6	RXOVRNINTENA		Overrun interrupt enable.
		0	Overrun interrupt will not be generated.
		1	Overrun interrupt will be generated.
5	Reserved	0	Reads return 0. Writes have no effect.
4	BITERRENA		Enables interrupt on bit error.
		0	No interrupt asserted upon bit error.
		1	Enables interrupt on bit error.
3	DESYNCENA		Enables interrupt on desynchronized slave. DESYNCENA is used in master mode only.
		0	No interrupt asserted upon desynchronization error.
		1	An interrupt is asserted on desynchronization of the slave (DESYNC = 1).
2	PARERRENA		Enables interrupt-on-parity-error.
		0	No interrupt asserted on parity error.
		1	An interrupt is asserted on a parity error.
1	TIMEOUTENA		Enables interrupt on ENA signal time-out.
		0	No interrupt asserted upon ENA signal time-out.
		1	An interrupt is asserted on a time-out of the ENA signal.
0	DLENERRENA		Data length error interrupt enable. A data length error occurs under the following conditions.
			Master: When $\overline{\text{SPIENA}}$ is used, if the $\overline{\text{SPIENA}}$ pin from the slave is deasserted before the master has completed its transfer, the data length error is set. That is, if the character length counter has not overflowed while $\overline{\text{SPIENA}}$ deassertion is detected, then it means that the slave has neither received full data from the master nor has it transmitted complete data.
			Slave: When $\overline{\text{SPICS}}$ pins are used, if the incoming valid $\overline{\text{SPICS}}$ pin is deactivated before the character length counter overflows, then the data length error is set.
		0	No interrupt is generated upon data length error.
		1	An interrupt is asserted when a data-length error occurs.

28.3.4 SPI Interrupt Level Register (SPILVL)

Figure 28-35. SPI Interrupt Level Register (SPILVL) [offset = 0Ch]

31-16
Reserved
R-0

15-10	9	8
Reserved	TXINTLVL	RXINTLVL
R-0	R/W-0	R/W-0

7	6	5	4	3	2	1	0
Reserved	RXOVRNINTL	Reserved	BITERRLVL	DESYNCLVL	PARERRLVL	TIMEOUTLVL	DLENERRLVL
R-0	R/W-0	R-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 28-12. SPI Interrupt Level Register (SPILVL) Field Descriptions

Bit	Field	Value	Description
31-10	Reserved	0	Reads return 0. Writes have no effect.
9	TXINTLVL		Transmit interrupt level.
		0	Transmit interrupt is mapped to interrupt line INT0.
		1	Transmit interrupt is mapped to interrupt line INT1.
8	RXINTLVL		Receive interrupt level.
		0	Receive interrupt is mapped to interrupt line INT0.
		1	Receive interrupt is mapped to interrupt line INT1.
7	Reserved	0	Reads return 0. Writes have no effect.
6	RXOVRNINTLVL		Receive overrun interrupt level.
		0	Receive overrun interrupt is mapped to interrupt line INT0.
		1	Receive overrun interrupt is mapped to interrupt line INT1.
5	Reserved	0	Reads return 0. Writes have no effect.
4	BITERRLVL		Bit error interrupt level.
		0	Bit error interrupt is mapped to interrupt line INT0.
		1	Bit error interrupt is mapped to interrupt line INT1.
3	DESYNCLVL		Desynchronized slave interrupt level. (master mode only).
		0	An interrupt caused by desynchronization of the slave is mapped to interrupt line INT0.
		1	An interrupt caused by desynchronization of the slave is mapped to interrupt line INT1.
2	PARERRLVL		Parity error interrupt level.
		0	A parity error interrupt is mapped to interrupt line INT0.
		1	A parity error interrupt is mapped to interrupt line INT1.
1	TIMEOUTLVL		$\overline{\text{SPIENA}}$ pin time-out interrupt level.
		0	An interrupt on a time-out of the ENA signal (TIMEOUT = 1) is mapped to interrupt line INT0.
		1	An interrupt on a time-out of the ENA signal (TIMEOUT = 1) is mapped to interrupt line INT1.
0	DLENERRLVL		Data length error interrupt level (line) select.
		0	An interrupt on data length error is mapped to interrupt line INT0.
		1	An interrupt on data length error is mapped to interrupt line INT1.

28.3.5 SPI Flag Register (SPIFLG)

Software must check all flag bits when reading this register.

Figure 28-36. SPI Flag Register (SPIFLG) [offset = 10h]

31 25	24	23 16
Reserved	BUFINIT ACTIVE	Reserved
R-0	R-0	R-0

15 10	9	8
Reserved	TXINTFLG	RXINTFLG
R-0	R-0	R/W1C-0

7	6	5	4	3	2	1	0
Reserved	RXOVRNINT FLG	Reserved	BITERR FLG	DESYNC FLG	PARERR FLG	TIMEOUT FLG	DLENERR FLG
R-0	R/W1C-0	R-0	R/W1C-0	R/W1C-0	R/W1C-0	R/W1C-0	R/W1C-0

LEGEND: R/W = Read/Write; R = Read only; W1C = Write 1 to clear; -*n* = value after reset

Table 28-13. SPI Flag Register (SPIFLG) Field Descriptions

Bit	Field	Value	Description
31-25	Reserved	0	Reads return 0. Writes have no effect.
24	BUFINITACTIVE		Indicates the status of multi-buffer initialization process. Software can poll for this bit to determine if it can proceed with the register configuration of multi-buffer mode registers or buffer handling. **Note: If the SPIFLG register is read while the multi-buffer RAM is being initialized, the BUF INIT ACTIVE bit will be read as 1. If SPIFLG is read after the internal automatic buffer initialization is complete, this bit will be read as 0. This bit will show a value of 1 as long as the nRESET bit is 0, but does not really indicate that buffer initialization is underway. Buffer initialization starts only when the nRESET bit is set to 1.**
		0	Multi-buffer RAM initialization is complete.
		1	Multi-buffer RAM is still being initialized. Do not attempt to write to either multi-buffer RAM or any multi-buffer mode registers.
23-10	Reserved	0	Reads return 0. Writes have no effect.
9	TXINTFLG		Transmitter-empty interrupt flag. Serves as an interrupt flag indicating that the transmit buffer (TXBUF) is empty and a new word can be written to it. This flag is set when a word is copied to the shift register either directly from SPIDAT0/SPIDAT1 or from the TXBUF register. This bit is cleared by one of following methods: • Writing a new data to either SPIDAT0 or SPIDAT1 • Writing a 0 to SPIEN (SPIGCR1[24])
		0	Transmit buffer is now full. No interrupt pending for transmitter empty.
		1	Transmit buffer is empty. An interrupt is pending to fill the transmitter.

Table 28-13. SPI Flag Register (SPIFLG) Field Descriptions (continued)

Bit	Field	Value	Description
8	RXINTFLG		Receiver-full interrupt flag. This flag is set when a word is received and copied into the buffer register (SPIBUF). If RXINTEN is enabled, an interrupt is also generated. This bit is cleared under the following methods: • Reading the SPIBUF register • Reading TGINTVECT0 or TGINTVECT1 register when there is a receive buffer full interrupt • Writing a 1 to this bit • Writing a 0 to SPIEN (SPIGCR1[24]) • System reset During emulation mode, however, a read to the emulation register (SPIEMU) does not clear this flag bit.
		0	No new received data pending. Receive buffer is empty.
		1	A newly received data is ready to be read. Receive buffer is full. **Note: Clearing RXINTFLG bit by writing a 1 before reading the SPIBUF sets the RXEMPTY bit of the SPIBUF register too. In this way, one can ignore a received word. However, if the internal RXBUF is already full, the data from RXBUF will be copied to SPIBUF and the RXEMPTY bit will be cleared again. The SPIBUF contents should be read first if this situation needs to be avoided.**
7	Reserved	0	Reads return 0. Writes have no effect.
6	RXOVRNINTFLG		Receiver overrun flag. The SPI hardware sets this bit when a receive operation completes before the previous character has been read from the receive buffer. The bit indicates that the last received character has been overwritten and therefore lost. The SPI will generate an interrupt request if this bit is set and the RXOVRN INTEN bit (SPIINT0.6) is set high. This bit is cleared under the following conditions in compatibility mode of MibSPI: • Reading TGINTVECT0 or TGINTVECT1 register when there is a receive-buffer-overrun interrupt • Writing a 1 to RXOVRNINTFLG in the SPI Flag Register (SPIFLG) itself • Writing a 0 to SPIEN • Reading the data field of the SPIBUF register **Note: Reading the SPIBUF register does not clear this RXOVRNINTFLG bit. If an RXOVRN interrupt is detected, then the SPIBUF may need to be read twice to get to the overrun buffer. This is due to the fact that the overrun will always occur to the internal RXBUF. Each read to the SPIBUF will result in RXBUF contents (if it is full) getting copied to SPIBUF.** **Note: There is a special condition under which the RXOVRNINTFLG flag gets set. If both SPIBUF and RXBUF are already full and while another reception is underway, if any errors (TIMEOUT, BITERR, and DLEN_ERR) occur, then RXOVR in RXBUF and RXOVRNINTFLG in SPIFLG registers will be set to indicate that the status flags are getting overwritten by the new transfer. This overrun should be treated like a receive overrun.** In multi-buffer mode of MibSPI, this bit is cleared under the following conditions: • Reading the RXOVRN_BUF_ADDR register • Writing a 1 to RXOVRNINTFLG in the SPI Flag Register (SPIFLG) itself In multi-buffer mode, if RXOVRNINTFLG is set, then the address of the buffer which experienced the overrun is available in RXOVRN_BUF_ADDR.
		0	Overrun condition did not occur.
		1	Overrun condition has occurred.
5	Reserved	0	Reads return 0. Writes have no effect.
4	BITERRFLG		Mismatch of internal transmit data and transmitted data. This flag can be cleared by one of the following methods: • Write a 1 to this bit. • Clear the SPIEN bit to 0.
		0	No bit error occurred.
		1	A bit error occurred. The SPI samples the signal of the transmit pin (master: SIMO, slave: SOMI) at the receive point (half clock cycle after transmit point). If the sampled value differs from the transmitted value a bit error is detected and the flag BITERRFLG is set. If BITERRENA is set an interrupt is asserted. Possible reasons for a bit error can be an excessively high bit rate, capacitive load, or another master/slave trying to transmit at the same time.

Table 28-13. SPI Flag Register (SPIFLG) Field Descriptions (continued)

Bit	Field	Value	Description
3	DESYNCFLG		Desynchronization of slave device. Desynchronization monitor is active in master mode only. This flag can be cleared by one of the following methods: • Write a 1 to this bit. • Clear the SPIEN bit to 0.
		0	No slave desynchronization detected.
		1	A slave device is desynchronized. The master monitors the ENAble signal coming from the slave device and sets the DESYNC flag after the last bit is transmitted plus $t_{T2EDELAY}$. If DESYNCENA is set an interrupt is asserted. Desynchronization can occur if a slave device misses a clock edge coming from the master.
2	PARERRFLG		Calculated parity differs from received parity bit. If the parity generator is enabled (can be selected individually for each buffer) an even or odd parity bit is added at the end of a data word. During reception of the data word the parity generator calculates the reference parity and compares it to the received parity bit. In the event of a mismatch the PARITYERR flag is set and an interrupt is asserted if PARERRENA is set. This flag can be cleared by one of the following methods: • Write a 1 to this bit. • Clear the SPIEN bit to 0.
		0	No parity error detected.
		1	A parity error occurred.
1	TIMEOUTFLG		Time-out caused by nonactivation of ENA signal. This flag can be cleared by one of the following methods: • Write a 1 to this bit. • Clear the SPIEN bit to 0.
		0	No ENA-signal time-out occurred.
		1	An ENA signal time-out occurred. The SPI generates a time-out because the slave has not responded in time by activating the ENA signal after the chip select signal has been activated. If a time-out condition is detected the corresponding chip select is deactivated immediately and the TIMEOUT flag is set. In addition the TIMEOUT flag in the status field of the corresponding buffer is set. The transmit request of the concerned buffer is cleared, that is, the SPI does not re-start a data transfer from this buffer.
0	DLENERRFLG		Data-length error flag. This flag can be cleared by one of the following methods: • Write a 1 to this bit. • Clear the SPIEN bit to 0. **Note: Whenever any transmission errors (TIMEOUT, BITERR, DLEN_ERR, PARITY_ERR, DESYNC) are detected and the error flags are cleared by writing to the error bit in the SPIFLG register, the corresponding error flag in SPIBUF does not get cleared. Software needs to read the SPIBUF until it becomes empty before proceeding. This ensures that all of the old status bits in SPIBUF are cleared before starting the next transfer.**
		0	No data length error has occurred.
		1	A data length error has occurred.

28.3.6 SPI Pin Control Register 0 (SPIPC0)

NOTE: Register bits vary by device

Register bits 31:24 and 23:16 of SPIPC0 to SPIPC9 reflect the number of SIMO/SOMI data lines per device. On devices with 8 data-line support, all of bits 31 to 16 are implemented. On devices with less than 8 data lines, only a subset of these bits are available. Unimplemented bits return 0 upon read and are not writable.

Figure 28-37. SPI Pin Control Register 0 (SPIPC0) [offset = 14h]

31 24	23 16
SOMIFUN	SIMOFUN
R/W-0	R/W-0

15 12	11	10	9	8
Reserved	SOMIFUN0	SIMOFUN0	CLKFUN	ENAFUN
R-0	R/W-0	R/W-0	R/W-0	R/W-0

7 0
SCSFUN
R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 28-14. SPI Pin Control (SPIPC0) Field Descriptions

Bit	Field	Value	Description
31-24	SOMIFUN		Slave out, master in function. Determines whether SPISOMI[x] is to be used as a general-purpose I/O pin or as a SPI functional pin.
			Note: Duplicate Control Bits for SPISOMI[0]. Bit 24 is not physically implemented. It is a mirror of Bit 11. Any write to bit 24 will be reflected on bit 11. When bit 24 and bit 11 are simultaneously written, the value of bit 11 will control the SPISOMI[0] pin. The read value of Bit 24 always reflects the value of bit 11.
		0	SPISOMI[x] pin is a GIO pin.
		1	SPISOMI[x] pin is a SPI functional pin.
23-16	SIMOFUN		Slave in, master out function. Determines whether SPISIMO[x] is to be used as a general-purpose I/O pin or as a SPI functional pin.
			Note: Duplicate Control Bits for SPISIMO[x]. Bit 16 is not physically implemented. It is a mirror of Bit 10. Any write to bit 16 will be reflected on bit 10. When bit 16 and bit 10 are simultaneously written, the value of bit 10 will control the SPISOMI[x] pin. The read value of Bit 16 always reflects the value of bit 10.
		0	SPISIMOx pin is a GIO pin.
		1	SPISIMOx pin is a SPI functional pin.
15-12	Reserved	0	Reads return 0. Writes have no effect.
11	SOMIFUN0		Slave out, master in function. This bit determines whether the SPISOMI0 pin is to be used as a general-purpose I/O pin or as a SPI functional pin.
		0	SPISOMI0 pin is a GIO pin.
		1	SPISOMI0 pin is a SPI functional pin.
			Note: Regardless of the number of parallel pins used, the SPISOMI0 pin will always have to be programmed as functional pins for any SPI transfers.
10	SIMOFUN0		Slave in, master out function. This bits determine whether each SPISIMO0 pin is to be used as a general-purpose I/O pin or as a SPI functional pin.
		0	SPISIMO0 pin is a GIO pin.
		1	SPISIMO0 pin is a SPI functional pin.
			Note: Regardless of the number of parallel pins used, the SPISIMO0 pin will always have to be programmed as functional pins for any SPI transfers.

Table 28-14. SPI Pin Control (SPIPC0) Field Descriptions (continued)

Bit	Field	Value	Description
9	CLKFUN		SPI clock function. This bit determines whether the SPICLK pin is to be used as a general-purpose I/O pin, or as a SPI functional pin.
		0	SPICLK pin is a GIO pin.
		1	SPICLK pin is a SPI functional pin.
8	ENAFUN		$\overline{\text{SPIENA}}$ function. This bit determines whether the $\overline{\text{SPIENA}}$ pin is to be used as a general-purpose I/O pin or as a SPI functional pin.
		0	$\overline{\text{SPIENA}}$ pin is a GIO pin.
		1	$\overline{\text{SPIENA}}$ pin is a SPI functional pin.
7-0	SCSFUN		$\overline{\text{SPICS}}$ function. Determines whether each $\overline{\text{SPICS}}$ pin is to be used as a general-purpose I/O pin or as a SPI functional pin. If the slave $\overline{\text{SPICS}}$ pins are in functional mode and receive an inactive-high signal, the slave SPI will place its output in high-impedance and disable shifting.
		0	$\overline{\text{SPICS}}$ pin is a GIO pin.
		1	$\overline{\text{SPICS}}$ pin is a SPI functional pin.

28.3.7 SPI Pin Control Register 1 (SPIPC1)

NOTE: Register bits vary by device

Register bits 31:24 and 23:16 of this register reflect the number of SIMO/SOMI data lines per device. On devices with 8 data-line support, all of bits 31 to 16 are implemented. On devices with less than 8 data lines, only a subset of these bits are available. Unimplemented bits return 0 upon read and are not writable.

Figure 28-38. SPI Pin Control Register 1 (SPIPC1) [offset = 18h]

31-24	23-16
SOMIDIR	SIMODIR
R/W-0	R/W-0

15-12	11	10	9	8
Reserved	SOMIDIR0	SIMODIR0	CLKDIR	ENADIR
R-0	R/W-0	R/W-0	R/W-0	R/W-0

7-0
SCSDIR
R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 28-15. SPI Pin Control Register (SPIPC1) Field Descriptions

Bit	Field	Value	Description
31-24	SOMIDIR		SPISOMIx direction. Controls the direction of SPISOMIx when used for general-purpose I/O. If SPISOMIx pin is used as a SPI functional pin, the I/O direction is determined by the MASTER bit in the SPIGCR1 register. **Note: Duplicate Control Bits for SPISOMI0. Bit 24 is not physically implemented. It is a mirror of Bit 11. Any write to bit 24 will be reflected on bit 11. When bit 24 and bit 11 are simultaneously written, the value of bit 11 will control the SPISOMI pin. The read value of Bit 24 always reflects the value of bit 11.**
		0	SPISOMIx pin is an input.
		1	SPISOMIx pin is an output.

Table 28-15. SPI Pin Control Register (SPIPC1) Field Descriptions (continued)

Bit	Field	Value	Description
23-16	SIMODIR		SPISIMOx direction. Controls the direction of SPISIMOx when used for general-purpose I/O. If SPISIMOx pin is used as a SPI functional pin, the I/O direction is determined by the MASTER bit in the SPIGCR1 register. **Note: Duplicate Control Bits for SPISIMO. Bit 16 is not physically implemented. It is a mirror of Bit 10. Any write to bit 16 will be reflected on bit 10. When bit 16 and bit 10 are simultaneously written, the value of bit 10 will control the SPISOMI pin. The read value of Bit 16 always reflects the value of bit 10.**
		0	SPISOMIOx pin is an input.
		1	SPISOMIOx pin is an output.
15-12	Reserved	0	Reads return 0. Writes have no effect.
11	SOMIDIR0		SPISOMI0 direction. This bit controls the direction of the SPISOMI0 pin when it is used as a general-purpose I/O pin. If the SPISOMI0 pin is used as a SPI functional pin, the I/O direction is determined by the MASTER bit in the SPIGCR1 register.
		0	SPISOMI0 pin is an input.
		1	SPISOMI0 pin is an output.
10	SIMODIR0		SPISIMO0 direction. This bit controls the direction of the SPISIMO0 pin when it is used as a general-purpose I/O pin. If the SPISIMO0 pin is used as a SPI functional pin, the I/O direction is determined by the MASTER bit in the SPIGCR1 register.
		0	SPISIMO0 pin is an input.
		1	SPISIMO0 pin is an output.
9	CLKDIR		SPICLK direction. This bit controls the direction of the SPICLK pin when it is used as a general-purpose I/O pin. In functional mode, the I/O direction is determined by the CLKMOD bit.
		0	SPICLK pin is an input.
		1	SPICLK pin is an output.
8	ENADIR		$\overline{\text{SPIENA}}$ direction. This bit controls the direction of the $\overline{\text{SPIENA}}$ pin when it is used as a general-purpose I/O. If the $\overline{\text{SPIENA}}$ pin is used as a functional pin, then the I/O direction is determined by the CLKMOD bit (SPIGCR1[1]).
		0	$\overline{\text{SPIENA}}$ pin is an input.
		1	$\overline{\text{SPIENA}}$ pin is an output.
7-0	SCSDIR		$\overline{\text{SPICS}}$ direction. These bits control the direction of each $\overline{\text{SPICS}}$ pin when it is used as a general-purpose I/O pin. Each pin could be configured independently from the others if the $\overline{\text{SPICS}}$ is used as a SPI functional pin. The I/O direction is determined by the CLKMOD bit (SPIGCR1[1]).
		0	$\overline{\text{SPICS}}$ pin is an input.
		1	$\overline{\text{SPICS}}$ pin is an output.

28.3.8 SPI Pin Control Register 2 (SPIPC2)

NOTE: Register bits vary by device

Register bits 31:24 and 23:16 of this register reflect the number of SIMO/SOMI data lines per device. On devices with 8 data-line support, all of bits 31 to 16 are implemented. On devices with less than 8 data lines, only a subset of these bits are available. Unimplemented bits return 0 upon read and are not writable.

Figure 28-39. SPI Pin Control Register 2 (SPIPC2) [offset = 1Ch]

31 24	23 16
SOMIDIN	SIMODIN
R/W-U	R/W-U

15 12	11	10	9	8
Reserved	SOMIDIN0	SIMODIN0	CLKDIN	ENADIN
R-0	R-U	R-U	R-U	R-U

7 0
SCSDIN
R/W-U

LEGEND: R/W = Read/Write; R = Read only; U = value is undefined; -*n* = value after reset

Table 28-16. SPI Pin Control Register 2 (SPIPC2) Field Descriptions

Bit	Field	Value	Description
31-24	SOMIDIN		SPISOMIx data in. The value of the SPISOMIx pins.
		0	SPISOMIx pin is logic 0.
		1	SPISOMIx pin is logic 1.
23-16	SIMODIN		SPISIMOx data in. The value of the SPISIMOx pins.
		0	SPISIMOx pin is logic 0.
		1	SPISIMOx pin is logic 1.
15-12	Reserved	0	Reads return 0. Writes have no effect.
11	SOMIDIN0		SPISOMI0 data in. The value of the SPISOMI0 pin.
		0	SPISOMI0 pin is logic 0.
		1	SPISOMI0 pin is logic 1.
10	SIMODIN0		SPISIMO0 data in. The value of the SPISIMO0 pin.
		0	SPISIMO0 pin is logic 0.
		1	SPISIMO0 pin is logic 1.
9	CLKDIN		Clock data in. The value of the SPICLK pin.
		0	SPICLK pin is logic 0.
		1	SPICLK pin is logic 1.
8	ENADIN		$\overline{\text{SPIENA}}$ data in. The value of the $\overline{\text{SPIENA}}$ pin.
		0	$\overline{\text{SPIENA}}$ pin is logic 0.
		1	$\overline{\text{SPIENA}}$ pin is logic 1.
7-0	SCSDIN		$\overline{\text{SPICS}}$ data in. The value of each $\overline{\text{SPICS}}$ pin.
		0	$\overline{\text{SPICS}}$ pin is logic 0.
		1	$\overline{\text{SPICS}}$ pin is logic 1.

28.3.9 SPI Pin Control Register 3 (SPIPC3)

NOTE: Register bits vary by device

Register bits 31:24 and 23:16 of this register reflect the number of SIMO/SOMI data lines per device. On devices with 8 data-line support, all of bits 31 to 16 are implemented. On devices with less than 8 data lines, only a subset of these bits are available. Unimplemented bits return 0 upon read and are not writable.

Figure 28-40. SPI Pin Control Register 3 (SPIPC3) [offset = 20h]

31 24	23 16
SOMIDOUT	SIMODOUT
R/W-U	R/W-U

15 12	11	10	9	8
Reserved	SOMIDOUT0	SIMODOUT0	CLKDOUT	ENADOUT
R-0	R/W-U	R/W-U	R/W-U	R/W-U

7 0
SCSDOUT
R/W-U

LEGEND: R/W = Read/Write; R = Read only; U = value is undefined; -*n* = value after reset

Table 28-17. SPI Pin Control Register 3 (SPIPC3) Field Descriptions

Bit	Field	Value	Description
31-24	SOMIDOUT		SPISOMIx data out write. This bit is only active when the SPISOMIx pin is configured as a general-purpose I/O pin and configured as an output pin. The value of this bit indicates the value sent to the pin. **Bit 11 or bit 24 can be used to set the direction for pin SPISOMI0. If a 32-bit write is performed, bit 11 will have priority over bit 24.**
		0	Current value on SPISOMIx pin is logic 0.
		1	Current value on SPISOMIx pin is logic 1
23-16	SIMODOUT		SPISIMOx data out write. This bit is only active when the SPISIMOx pin is configured as a general-purpose I/O pin and configured as an output pin. The value of this bit indicates the value sent to the pin. **Bit 10 or bit 16 can be used to set the direction for pin SPISOMI0. If a 32-bit write is performed, bit 10 will have priority over bit 16.**
		0	Current value on SPISIMOx pin is logic 0.
		1	Current value on SPISIMOx pin is logic 1.
15-12	Reserved	0	Reads return 0. Writes have no effect.
11	SOMIDOUT0		SPISOMI0 data out write. This bit is only active when the SPISOMI0 pin is configured as a general-purpose I/O pin and configured as an output pin. The value of this bit indicates the value sent to the pin.
		0	Current value on SPISOMI0 pin is logic 0.
		1	Current value on SPISOMI0 pin is logic 1.
10	SIMODOUT0		SPISIMO0 data out write. This bit is only active when the SPISIMO0 pin is configured as a general-purpose I/O pin and configured as an output pin. The value of this bit indicates the value sent to the pin.
		0	Current value on SPISIMO0 pin is logic 0.
		1	Current value on SPISIMO0 pin is logic 1.
9	CLKDOUT		SPICLK data out write. This bit is only active when the SPICLK pin is configured as a general-purpose I/O pin and configured as an output pin. The value of this bit indicates the value sent to the pin.
		0	Current value on SPICLK pin is logic 0.
		1	Current value on SPICLK pin is logic 1.

Table 28-17. SPI Pin Control Register 3 (SPIPC3) Field Descriptions (continued)

Bit	Field	Value	Description
8	ENADOUT		$\overline{\text{SPIENA}}$ data out write. Only active when the $\overline{\text{SPIENA}}$ pin is configured as a general-purpose I/O pin and configured as an output pin. The value of this bit indicates the value sent to the pin.
		0	Current value on $\overline{\text{SPIENA}}$ pin is logic 0.
		1	Current value on $\overline{\text{SPIENA}}$ pin is logic 1.
7-0	SCSDOUT		$\overline{\text{SPICS}}$ data out write. Only active when the $\overline{\text{SPICS}}$ pins are configured as a general-purpose I/O pins and configured as output pins. The value of these bits indicates the value sent to the pins.
		0	Current value on $\overline{\text{SPICS}}$ pin is logic 0.
		1	Current value on $\overline{\text{SPICS}}$ pin is logic 1.

28.3.10 SPI Pin Control Register 4 (SPIPC4)

NOTE: Register bits vary by device

Register bits 31:24 and 23:16 of this register reflect the number of SIMO/SOMI data lines per device. On devices with 8 data-line support, all of bits 31 to 16 are implemented. On devices with less than 8 data lines, only a subset of these bits are available. Unimplemented bits return 0 upon read and are not writable.

Figure 28-41. SPI Pin Control Register 4 (SPIPC4) [offset = 24h]

31 – 24	23 – 16
SOMISET	SIMOSET
R/W-U	R/W-U

15 – 12	11	10	9	8
Reserved	SOMISET0	SIMOSET0	CLKSET	ENASET
R-0	R/W-U	R/W-U	R/W-U	R/W-U

7 – 0
SCSSET
R/W-U

LEGEND: R/W = Read/Write; R = Read only; U = value is undefined; -*n* = value after reset

Table 28-18. SPI Pin Control Register 4 (SPIPC4) Field Descriptions

Bit	Field	Value	Description
31-24	SOMISET		SPISOMIx data out set. This pin is only active when the SPISOMIx pin is configured as a general-purpose output pin. **Bit 11 or bit 24 can be used to set the SOMI0 pin. If a 32-bit write is performed, bit 11 will have priority over bit 24.**
		0	Read: SPISIMOx is logic 0. Write: No effect.
		1	Read: SPISOMIx is logic 1. Write: Logic 1 is placed on SPISOMIx pin, if it is in general-purpose output mode.
23-16	SIMOSET		SPISIMOx data out set. This bit is only active when the SPISIMOx pin is configured as a general-purpose output pin. **Bit 10 or bit 16 can be used to set the SOMI0 pin. If a 32-bit write is performed, bit 10 will have priority over bit 16.**
		0	Read: SPISIMIx is logic 0. Write: No effect.
		1	Read: SPISIMIx is logic 1. Write: Logic 1 is placed on SPISIMIx pin, if it is in general-purpose output mode.

Table 28-18. SPI Pin Control Register 4 (SPIPC4) Field Descriptions (continued)

Bit	Field	Value	Description
15-12	Reserved	0	Reads return 0. Writes have no effect.
11	SOMISET0	0	SPISOMI0 data out set. This pin is only active when the SPISOMI0 pin is configured as a general-purpose output pin. Read: SPISOMI0 is logic 0. Write: No effect.
		1	Read: SPISOMI0 is logic 1. Write: Logic 1 is placed on SPISOMI0 pin, if it is in general-purpose output mode.
10	SIMOSET0	0	SPISIMO0 data out set. This pin is only active when the SPISIMO0 pin is configured as a general-purpose output pin. Read: SPISIMO0 is logic 0. Write: No effect.
		1	Read: SPISIMO0 is logic 1. Write: Logic 1 is placed on SPISIMO0 pin, if it is in general-purpose output mode.
9	CLKSET	0	SPICLK data out set. This bit is only active when the SPICLK pin is configured as a general-purpose output pin. Read: SPICLK is logic 0. Write: No effect.
		1	Read: SPICLK is logic 1. Write: Logic 1 is placed on SPICLK pin, if it is in general-purpose output mode.
8	ENASET	0	$\overline{\text{SPIENA}}$ data out set. This bit is only active when the $\overline{\text{SPIENA}}$ pin is configured as a general-purpose output pin. Read: $\overline{\text{SPIENA}}$ is logic 0. Write: No effect.
		1	Read: $\overline{\text{SPIENA}}$ is logic 1. Write: Logic 1 is placed on $\overline{\text{SPIENA}}$ pin, if it is in general-purpose O/P mode.
7-0	SCSSET	0	$\overline{\text{SPICS}}$ data out set. This bit is only active when the $\overline{\text{SPICS}}$ pin is configured as a general-purpose output pin. A value of 1 written to this bit sets the corresponding SCSDOUT bit to 1. Read: $\overline{\text{SPICS}}$ is logic 0. Write: No effect.
		1	Read: $\overline{\text{SPICS}}$ is logic 1. Write: Logic 1 is placed on $\overline{\text{SPICS}}$ pin, if it is in general-purpose output mode.

28.3.11 SPI Pin Control Register 5 (SPIPC5)

NOTE: Register bits vary by device

Register bits 31:24 and 23:16 of this register reflect the number of SIMO/SOMI data lines per device. On devices with 8 data-line support, all of bits 31 to 16 are implemented. On devices with less than 8 data lines, only a subset of these bits are available. Unimplemented bits return 0 upon read and are not writable.

Figure 28-42. SPI Pin Control Register 5 (SPIPC5) [offset = 28h]

31 - 24	23 - 16
SOMICLR	SIMOCLR
R/W-U	R/W-U

15 - 12	11	10	9	8
Reserved	SOMICLR0	SIMOCLR0	CLKCLR	ENACLR
R-0	R/W-U	R/W-U	R/W-U	R/W-U

7 - 0
SCSCLR
R/W-U

LEGEND: R/W = Read/Write; R = Read only; U = value is undefined; -*n* = value after reset

Table 28-19. SPI Pin Control Register 5 (SPIPC5) Field Descriptions

Bit	Field	Value	Description
31-24	SOMICLR		SPISOMIx data out clear. This pin is only active when the SPISOMIx pin is configured as a general-purpose output pin. **Bit 11 or bit 24 can be used to set the SOMI0 pin. If a 32-bit write is performed, bit 11 will have priority over bit 24.**
		0	Read: The current value on SOMIDOUTx is 0. Write: No effect.
		1	Read: The current value on SOMIDOUTx is 1. Write: Logic 0 is placed on SPISOMIx pin, if it is in general-purpose output mode.
23-16	SIMOCLR		SPISIMOx data out clear. This bit is only active when the SPISIMOx pin is configured as a general-purpose output pin. **Bit 10 or bit 16 can be used to set the SOMI0 pin. If a 32-bit write is performed, bit 10 will have priority over bit 16.**
		0	Read: The current value on SOMODOUTx is 0. Write: No effect.
		1	Read: The current value on SOMODOUTx is 1. Write: Logic 0 is placed on SPISIMIx pin, if it is in general-purpose output mode.
15-12	Reserved	0	Reads return 0. Writes have no effect.
11	SOMICLR0		SPISOMI0 data out cleart. This pin is only active when the SPISOMI0 pin is configured as a general-purpose output pin.
		0	Read: The current value on SPISOMI0 is 0. Write: No effect.
		1	Read: The current value on SPISOMI0 is 1. Write: Logic 0 is placed on SPISOMI0 pin, if it is in general-purpose output mode.
10	SIMOCLR0		SPISIMO0 data out clear. This pin is only active when the SPISIMO0 pin is configured as a general-purpose output pin.
		0	Read: The current value on SPISIMO0 is 0. Write: No effect.
		1	Read: The current value on SPISIMO0 is 1. Write: Logic 0 is placed on SPISIMO0 pin, if it is in general-purpose output mode.

Table 28-19. SPI Pin Control Register 5 (SPIPC5) Field Descriptions (continued)

Bit	Field	Value	Description
9	CLKCLR	0	SPICLK data out clear. This bit is only active when the SPICLK pin is configured as a general-purpose output pin. Read: The current value on SPICLK is 0. Write: No effect.
		1	Read: The current value on SPICLK is 1. Write: Logic 0 is placed on SPICLK pin, if it is in general-purpose output mode.
8	ENACLR	0	$\overline{\text{SPIENA}}$ data out clear. This bit is only active when the $\overline{\text{SPIENA}}$ pin is configured as a general-purpose output pin. A value of 1 written to this bit clears the corresponding ENABLEDOUT bit to 0. Read: The current value on $\overline{\text{SPIENA}}$ is 0. Write: No effect.
		1	Read: The current value on $\overline{\text{SPIENA}}$ is 1. Write: Logic 0 is placed on $\overline{\text{SPIENA}}$ pin, if it is in general-purpose output mode.
7-0	SCSCLR	0	$\overline{\text{SPICS}}$ data out clear. This bit is only active when the $\overline{\text{SPICS}}$ pin is configured as a general-purpose output pin. Read: The current value on SCSDOUT is 0. Write: No effect.
		1	Read: The current value on SCSDOUT is 1. Write: Logic 0 is placed on $\overline{\text{SPICS}}$ pin, if it is in general-purpose output mode.

28.3.12 SPI Pin Control Register 6 (SPIPC6)

NOTE: Register bits vary by device

Register bits 31:24 and 23:16 of SPIPC0 to SPIPC9 reflect the number of SIMO/SOMI data lines per device. On devices with 8 data-line support, all of bits 31 to 16 are implemented. On devices with less than 8 data lines, only a subset of these bits are available. Unimplemented bits return 0 upon read and are not writable.

Figure 28-43. SPI Pin Control Register 6 (SPIPC6) [offset = 2Ch]

31-24	23-16
SOMIPDR	SIMOPDR
R/W-0	R/W-0

15-12	11	10	9	8
Reserved	SOMIPDR0	SIMOPDR0	CLKPDR	ENAPDR
R-0	R/W-0	R/W-0	R/W-0	R/W-0

7-0
SCSPDR
R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 28-20. SPI Pin Control Register 6 (SPIPC6) Field Descriptions

Bit	Field	Value	Description
31-24	SOMIPDR		SPISOMIx open drain enable. This bit enables open drain capability for the SPISOMIx pin if the following conditions are met: • SOMIDIRx = 1 (SPISOMIx pin configured in GIO mode as an output) • SOMIDOUTx = 1 **Bit 11 or bit 24 can both be used to enable open-drain for SOMI0. If a 32-bit write is performed, bit 11 will have priority over bit 24.**
		0	The output value on the SPISOMIx pin is logic 1.
		1	Output pin SPISOMIx is in a high-impedance state.
23-16	SIMOPDR		SPISIMOx open drain enable. This bit enables open drain capability for the SPISIMOx pin if the following conditions are met: • SIMODIRx = 1 (SPISIMOx pin configured in GIO mode as an output) • SIMODOUTx = 1 **Bit 10 or bit 16 can both be used to enable open-drain for SIMO0. If a 32-bit write is performed, bit 10 will have priority over bit 16.**
		0	The output value on SPISIMOx pin is logic 1.
		1	Output pin SPISIMOx is in a high-impedance state.
15-12	Reserved	0	Reads return 0. Writes have no effect.
11	SOMIPDR0		SOMI0 open-drain enable. This bit enables open-drain capability for SOMI0 if the following conditions are met. • SOMI0 pin configured in GIO mode as output pin • Output value on SPISOMI0 pin is logic 1.
		0	Output value 1 of SPISOMI0 pin is logic 1.
		1	Output value 1 of SPISOMI0 is high-impedance.
10	SIMOPDR0		SPISIMO0 open-drain enable. This bit enables open -drain capability for the SPISIMO0 pin if the following conditions are met. • SIMO0 pin configured in GIO mode as output pin • Output value on SPISIMO0 pin is logic 1.
		0	Output value 1 of SPISIMO0 pin is logic 1.
		1	Output value 1 of SPISIMO0 is high-impedance.
9	CLKPDR		CLK open drain enable. This bit enables open drain capability for the pin CLK if the following conditions are met: • SPICLK pin configured in GIO mode as an output pin • SPICLKDOUT = 1
		0	Output value on CLK pin is logic 1.
		1	Output pin CLK is in a high-impedance state.
8	ENAPDR		$\overline{\text{SPIENA}}$ open drain enable. This bit enables open drain capability for the $\overline{\text{SPIENA}}$ pin, if the following conditions are met: • $\overline{\text{SPIENA}}$ pin configured in GIO mode as an output pin • SPIENADOUT = 1
		0	Output value on the $\overline{\text{SPIENA}}$ pin is logic 1.
		1	Output pin $\overline{\text{SPIENA}}$ is in a high-impedance state.
7-0	SCSPDR		$\overline{\text{SPICS}}$ open drain enable. This bit enables open drain capability for the $\overline{\text{SPICS}}$ pin, if the following conditions are met: • $\overline{\text{SPICS}}$ pin configured in GIO mode as an output pin • SCSDOUT = 1
		0	Output value on the $\overline{\text{SPICS}}$ pin is logic 1.
		1	Output pin $\overline{\text{SPICS}}$ is in a high-impedance state.

28.3.13 SPI Pin Control Register 7 (SPIPC7)

NOTE: Register bits vary by device

Register bits 31:24 and 23:16 of this register reflect the number of SIMO/SOMI data lines per device. On devices with 8 data-line support, all of bits 31 to 16 are implemented. On devices with less than 8 data lines, only a subset of these bits are available. Unimplemented bits return 0 upon read and are not writable.

NOTE: Default Register Value

The default values of these register bits vary by device. See your device datasheet for information about default pin states, which correspond to the register reset values (see the pin-list table).

Figure 28-44. SPI Pin Control Register 7 (SPIPC7) [offset = 30h]

31-24	23-16
SOMIDIS	SIMODIS
R/W-x	R/W-x

15-12	11	10	9	8
Reserved	SOMIPDIS0	SIMOPDIS0	CLKPDIS	ENAPDIS
R-0	R/W-x	R/W-x	R/W-x	R/W-x

7-0
SCSPDIS
R/W-x

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset; -x = value varies by device

Table 28-21. SPI Pin Control Register 7 (SPIPC7) Field Descriptions

Bit	Field	Value	Description
31-24	SOMIDIS		SOMIx pull control disable. This bit disables pull control capability for each SOMIx pin if it is in input mode, regardless of whether it is in functional or GIO mode. **Note: Bit 11 or bit 24 can be used to set pull-disable for SOMIO. If a 32-bit write is performed, bit 11 will have priority over bit 24.**
		0	Pull control on the SPISOMIx pin is enabled.
		1	Pull control on the SPISOMIx pin is disabled.
23-16	SIMODIS		SIMOx pull control disable. This bit disables pull control capability for each SIMOx pin if it is in input mode, regardless of whether it is in functional or GIO mode. **Note: Bit 10 or bit 16 can be used to set pull-disable for SIMO0. If a 32-bit write is performed, bit 10 will have priority over bit 16.**
		0	Pull control on SPISIMOx pin is enabled.
		1	Pull control on SPISIMOx pin is disabled.
15-12	Reserved	0	Reads return 0. Writes have no effect.
11	SOMIPDIS0		SPISOMI0 pull control disable. This bit disables pull control capability for the SPISOMI0 pin if it is in input mode, regardless of whether it is in functional or GIO mode.
		0	Pull control on the SPISOMI0 pin is enabled.
		1	Pull control on the SPISOMI0 pin is disabled.
10	SIMOPDIS0		SPISIMO0 pull control disable. This bit disables pull control capability for the SPISIMO0 pin if it is in input mode, regardless of whether it is in functional or GIO mode.
		0	Pull control on the SPISIMO0 pin is enabled.
		1	Pull control on the SPISIMO0 pin is disabled.

Table 28-21. SPI Pin Control Register 7 (SPIPC7) Field Descriptions (continued)

Bit	Field	Value	Description
9	CLKPDIS		CLK pull control disable. This bit disables pull control capability for the SPICLK pin if it is in input mode, regardless of whether it is in functional or GIO mode.
		0	Pull control on the CLK pin is enabled.
		1	Pull control on the CLK pin is disabled.
8	ENAPDIS		$\overline{\text{SPIENA}}$ pull control disable. This bit disables pull control capability for the $\overline{\text{SPIENA}}$ pin if it is in input mode, regardless of whether it is in functional or GIO mode.
		0	Pull control on the $\overline{\text{SPIENA}}$ pin is enabled.
		1	Pull control on the $\overline{\text{SPIENA}}$ pin is disabled.
7-0	SCSPDIS		$\overline{\text{SPICS}}$ pull control disable. This bit disables pull control capability for the $\overline{\text{SPICS}}$ pin if it is in input mode, regardless of whether it is in functional or GIO mode.
		0	Pull control on the $\overline{\text{SPICS}}$ pin is enabled.
		1	Pull control on the $\overline{\text{SPICS}}$ pin is disabled.

28.3.14 SPI Pin Control Register 8 (SPIPC8)

NOTE: Register bits vary by device

Register bits 31:24 and 23:16 of this register reflect the number of SIMO/SOMI data lines per device. On devices with 8 data-line support, all of bits 31 to 16 are implemented. On devices with less than 8 data lines, only a subset of these bits are available. Unimplemented bits return 0 upon read and are not writable.

NOTE: Default Register Value

The default values of these register bits vary by device. See your device datasheet for information about default pin states, which correspond to the register reset values (see the pin-list table).

Figure 28-45. SPI Pin Control Register 8 (SPIPC8) [offset = 34h]

31-24	23-16
SOMIPSEL	SIMOPSEL
R/W-x	R/W-x

15-12	11	10	9	8
Reserved	SOMIPSEL0	SIMOPSEL0	CLKPSEL	ENAPSEL
R-0	R/W-x	R/W-x	R/W-x	R/W-x

7-0
SCSPSEL
R/W-x

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset; -x = value varies by device

Table 28-22. SPI Pin Control Register 8 (SPIPC8) Field Descriptions

Bit	Field	Value	Description
31-24	SOMIPSEL		SPISOMIx pull select. This bit selects the type of pull logic at the SOMIx pin. **Note: Bit 11 or bit 24 can be used to set pull-select for SPISOMI0. If a 32-bit write is performed, bit 11 will have priority over bit 24.**
		0	Pull down on the SOMIx pin.
		1	Pull up on the SOMIx pin.

Table 28-22. SPI Pin Control Register 8 (SPIPC8) Field Descriptions (continued)

Bit	Field	Value	Description
23-16	SIMOPSEL		SPISIMOx pull select. This bit selects the type of pull logic at the SPISIMOx pin. **Note: Bit 10 or bit 16 can be used to set pull-select for SPISOMI0. If a 32-bit write is performed, bit 10 will have priority over bit 16.**
		0	Pull down on the SPISIMOx pin.
		1	Pull up on the SPISIMOx pin.
15-12	Reserved	0	Reads return 0. Writes have no effect.
11	SOMIPSEL0		SOMI pull select. This bit selects the type of pull logic at the SOMI pin.
		0	Pull down on the SPISOMI pin.
		1	Pull up on the SPISOMI pin.
10	SIMOPSEL0		SPISIMO pull select. This bit selects the type of pull logic at the SPISIMO pin.
		0	Pull down on the SPISIMO pin.
		1	Pull up on the SPISIMO pin.
9	CLKPSEL		SPICLK pull select. This bit selects the type of pull logic at the SPICLK pin.
		0	Pull down on the SPICLK pin.
		1	Pull up on the SPICLK pin.
8	ENAPSEL		$\overline{\text{SPIENA}}$ pull select. This bit selects the type of pull logic at the $\overline{\text{SPIENA}}$ pin.
		0	Pull down on the $\overline{\text{SPIENA}}$ pin.
		1	Pull up on the $\overline{\text{SPIENA}}$ pin.
7-0	SCSPSEL		$\overline{\text{SPICS}}$ pull select. This bit selects the type of pull logic at the $\overline{\text{SPICS}}$ pin.
		0	Pull down on the $\overline{\text{SPICS}}$ pin.
		1	Pull up on the $\overline{\text{SPICS}}$ pin.

28.3.15 SPI Transmit Data Register 0 (SPIDAT0)

Figure 28-46. SPI Transmit Data Register 0 (SPIDAT0) [offset = 38h]

31	16
Reserved	
R-0	

15	0
TXDATA	
R/W-0	

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 28-23. SPI Transmit Data Register 0 (SPIDAT0) Field Descriptions

Bit	Field	Value	Description
31-16	Reserved	0	Reads return 0. Writes have no effect.
15-0	TXDATA	0-FFFFh	SPI transmit data. When written, these bits will be copied to the shift register if it is empty. If the shift register is not empty, TXBUF holds the written data. SPIEN (SPICGR1[24]) must be set to 1 before this register can be written to. Writing a 0 to the SPIEN register forces the lower 16 bits of the SPIDAT0 to 0x00. **Note: When this register is read, the contents TXBUF, which holds the latest written data, will be returned.** **Note: Regardless of character length, the transmit word should be right-justified before writing to the SPIDAT1 register.** **Note: The default data format control register for SPIDAT0 is SPIFMT0. However, it is possible to reprogram the DFSEL[1:0] fields of SPIDAT1 before using SPIDAT0, to select a different SPIFMTx register.** **Note: It is highly recommended to use SPIDAT1 register, SPIDAT0 is supported for compatibility reasons.**

28.3.16 SPI Transmit Data Register 1 (SPIDAT1)

NOTE: Writing to only the control fields, bits 28 through 16, does not initiate any SPI transfer in master mode. This feature can be used to set up SPICLK phase or polarity before actually starting the transfer by only updating the DFSEL bit field to select the required phase and polarity combination.

Figure 28-47. SPI Transmit Data Register 1 (SPIDAT1) [offset = 3Ch]

31 29	28	27	26	25 24	23 16
Reserved	CSHOLD	Rsvd	WDEL	DFSEL	CSNR
R-0	R/W-0	R-0	R/W-0	R/W-0	R/W-0

15 0
TXDATA
R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 28-24. SPI Transmit Data Register 1 (SPIDAT1) Field Descriptions

Bit	Field	Value	Description
31-29	Reserved	0	Reads return 0. Writes have no effect.
28	CSHOLD		Chip select hold mode. The CSHOLD bit is supported in master mode only in compatibility-mode of SPI, (it is ignored in slave mode). CSHOLD defines the behavior of the chip select line at the end of a data transfer.
		0	The chip select signal is deactivated at the end of a transfer after the T2CDELAY time has passed. If two consecutive transfers are dedicated to the same chip select this chip select signal will be deactivated for at least 2VCLK cycles before it is activated again.
		1	The chip select signal is held active at the end of a transfer until a control field with new data and control information is loaded into SPIDAT1. If the new chip select number equals the previous one, the active chip select signal is extended until the end of transfer with CSHOLD cleared, or until the chip-select number changes.
27	Reserved	0	Reads return 0. Writes have no effect.
26	WDEL		Enable the delay counter at the end of the current transaction. **Note: The WDEL bit is supported in master mode only. In slave mode, this bit will be ignored.**
		0	No delay will be inserted. However, the $\overline{\text{SPICS}}$ pins will still be de-activated for at least for 2VCLK cycles if CSHOLD = 0. **Note: The duration for which the $\overline{\text{SPICS}}$ pin remains deactivated depends upon the time taken to supply a new word after completing the shift operation. If TXBUF is already full, then the $\overline{\text{SPICS}}$ pin will be deasserted for at least two VCLK cycles (if WDEL = 0).**
		1	After a transaction, WDELAY of the corresponding data format will be loaded into the delay counter. No transaction will be performed until the WDELAY counter overflows. The $\overline{\text{SPICS}}$ pins will be de-activated for at least (WDELAY + 2) × VCLK_Period duration.
25-24	DFSEL		Data word format select.
		0	Data word format 0 is selected.
		1h	Data word format 1 is selected.
		2h	Data word format 2 is selected.
		3h	Data word format 3 is selected.
23-16	CSNR	0-FFh	Chip select (CS) number. CSNR defines the chip select pins that will be activated during the data transfer. CSNR is a bit-mask that controls all chip select pins. See Table 28-25. **Note: If your MibSPI has less than 8 chip select pins, all unused upper bits will be 0. For example, MiBSPI3 has 6 chip select pins, if you write FFh to CSNR, the actual number stored in CSNR is 3Fh.**

Table 28-24. SPI Transmit Data Register 1 (SPIDAT1) Field Descriptions (continued)

Bit	Field	Value	Description
15-0	TXDATA	0-FFFFh	Transfer data. When written, these bits are copied to the shift register if it is empty. If the shift register is not empty, then they are held in TXBUF. SPIEN must be set to 1 before this register can be written to. Writing a 0 to SPIEN forces the lower 16 bits of SPIDAT1 to 0x0000. A write to this register (or to the TXDATA field only) drives the contents of the CSNR field on the $\overline{\text{SPICS}}$ pins, if the pins are configured as functional pins (automatic chip select, see Section 28.2.1). When this register is read, the contents of TXBUF, which holds the latest data written, will be returned. **Note: Regardless of the character length, the transmit data should be right-justified before writing to the SPIDAT1 register.**

Table 28-25. Chip Select Number Active

CSNR Value	Chip Select Active:					
	CS[5][1]	CS[4][1]	CS[3][1]	CS[2][1]	CS[1][1]	CS[0]
0h	No chip select pin is active.					
1h						x
2h					x	
3h					x	x
4h				x		
5h				x		x
6h				x	x	
7h				x	x	x
8h			x			
9h			x			x
Ah			x		x	
Bh			x		x	x
Ch			x	x		
Dh			x	x		x
Eh			x	x	x	
Fh			x	x	x	x
10h		x				
11h		x				x
12h		x			x	
13h		x			x	x
14h		x		x		
15h		x		x		x
16h		x		x	x	
17h		x		x	x	x
18h		x	x			
19h		x	x			x
1Ah		x	x		x	
1Bh		x	x		x	x
1Ch		x	x	x		
1Dh		x	x	x		x
1Eh		x	x	x	x	
1Fh		x	x	x	x	x
20h	x					
21h	x					x
22h	x				x	
23h	x				x	x
24h	x			x		
25h	x			x		x
26h	x			x	x	
27h	x			x	x	x
28h	x		x			
29h	x		x			x
2Ah	x		x		x	
2Bh	x		x		x	x
2Ch	x		x	x		
2Dh	x		x	x		x
2Eh	x		x	x	x	
2Fh	x		x	x	x	x
30h	x	x				
31h	x	x				x
32h	x	x			x	
33h	x	x			x	x
34h	x	x		x		
35h	x	x		x		x
36h	x	x		x	x	
37h	x	x		x	x	x
38h	x	x	x			
39h	x	x	x			x
3Ah	x	x	x		x	
3Bh	x	x	x		x	x
3Ch	x	x	x	x		
3Dh	x	x	x	x		x
3Eh	x	x	x	x	x	
3Fh	x	x	x	x	x	x

[1] If your MibSPI does not have this chip select pin, this bit is 0.

28.3.17 SPI Receive Buffer Register (SPIBUF)

Figure 28-48. SPI Receive Buffer Register (SPIBUF) [offset = 40h]

31	30	29	28	27	26	25	24
RXEMPTY	RXOVR	TXFULL	BITERR	DESYNC	PARITYERR	TIMEOUT	DLENERR
R-1	R-0	R-0	R-0	R-0	R-0	R-0	R-0

23	16
LCSNR	
R-0	

15	0
RXDATA	
R-0	

LEGEND: R = Read only; -*n* = value after reset

Table 28-26. SPI Receive Buffer Register (SPIBUF) Field Descriptions

Bit	Field	Value	Description
31	RXEMPTY		Receive data buffer empty. When the host reads the RXDATA field or the entire SPIBUF register, it automatically sets the RXEMPTY flag. When a data transfer is completed, the received data is copied into RXDATA and the RXEMPTY flag is cleared.
		0	New data has been received and copied into RXDATA.
		1	No data has been received since the last read of RXDATA.
			This flag gets set to 1 under the following conditions: • Reading the RXDATA field of the SPIBUF register. • Writing a 1 to clear the RXINTFLG bit in the SPI Flag Register (SPIFLG). Write-clearing the RXINTFLG bit before reading the SPIBUF indicates the received data is being ignored. Conversely, RXINTFLG can be cleared by reading the RXDATA field of SPIBUF (or the entire register).
30	RXOVR		Receive data buffer overrun. When a data transfer is completed and the received data is copied into RXBUF while it is already full, RXOVR is set. Overruns always occur to RXBUF, not to SPIBUF; the contents of SPIBUF are overwritten only after it is read by the Peripheral(VBUSP) master (CPU, DMA, or other host processor). If enabled, the RXOVRN interrupt is generated when RXBUF is overwritten, and reading either SPI Flag Register (SPIFLG) or SPIVECTx shows the RXOVRN condition. Two read operations from the SPIBUF register are required to reach the overwritten buffer word (one to read SPIBUF, which then transfers RXDATA into SPIBUF for the second read). **Note: This flag is cleared to 0 when the RXDATA field of the SPIBUF register is read.** **Note: A special condition under which RXOVR flag gets set. If both SPIBUF and RXBUF are already full and while another buffer receive is underway, if any errors such as TIMEOUT, BITERR and DLEN_ERR occur, then RXOVR in RXBUF and SPI Flag Register (SPIFLG) will be set to indicate that the status flags are getting overwritten by the new transfer. This overrun should be treated like a normal receive overrun.**
		0	No receive data overrun condition occurred since last read of the data field.
		1	A receive data overrun condition occurred since last read of the data field.
29	TXFULL		Transmit data buffer full. This flag is a read-only flag. Writing into the SPIDAT0 or SPIDAT1 field while the TX shift register is full will automatically set the TXFULL flag. Once the word is copied to the shift register, the TXFULL flag will be cleared. Writing to SPIDAT0 or SPIDAT1 when both TXBUF and the TX shift register are empty does not set the TXFULL flag.
		0	The transmit buffer is empty; SPIDAT0/SPIDAT1 is ready to accept a new data.
		1	The transmit buffer is full; SPIDAT0/SPIDAT1 is not ready to accept new data.

Table 28-26. SPI Receive Buffer Register (SPIBUF) Field Descriptions (continued)

Bit	Field	Value	Description
28	BITERR		Bit error. There was a mismatch of internal transmit data and transmitted data. **Note: This flag is cleared to 0 when the RXDATA field of the SPIBUF register is read.**
		0	No bit error occurred.
		1	A bit error occurred. The SPI samples the signal of the transmit pins (master: SIMOx, slave: SOMIx) at the receive point (one-half clock cycle after the transmit point). If the sampled value differs from the transmitted value, a bit error is detected and the BITERR flag is set. Possible reasons for a bit error include noise, an excessively high bit rate, capacitive load, or another master/slave trying to transmit at the same time.
27	DESYNC		Desynchronization of slave device. This bit is valid in master mode only. The master monitors the ENA signal coming from the slave device and sets the DESYNC flag if ENA is deactivated before the last reception point or after the last bit is transmitted plus $t_{T2EDELAY}$. If DESYNCENA is set, an interrupt is asserted. Desynchronization can occur if a slave device misses a clock edge coming from the master. **Note: In the Compatibility Mode MibSPI, under some circumstances it is possible for a desync error detected for the previous buffer to be visible in the current buffer. This is because the receive completion flag/interrupt is generated when the buffer transfer is completed. But desynchronization is detected after the buffer transfer is completed. So, if the VBUS master reads the received data quickly when an RXINT is detected, then the status flag may not reflect the correct desync condition. In multi-buffer mode, the desync flag is always guaranteed to be for the current buffer.** **Note: This flag is cleared to 0 when the RXDATA field of the SPIBUF register is read.**
		0	No slave desynchronization is detected.
		1	A slave device is desynchronized.
26	PARITYERR		Parity error. The calculated parity differs from the received parity bit. If the parity generator is enabled (selected individually for each buffer) an even or odd parity bit is added at the end of a data word. During reception of the data word, the parity generator calculates the reference parity and compares it to the received parity bit. If a mismatch is detected, the PARITYERR flag is set. **Note: This flag is cleared to 0 when the RXDATA field of the SPIBUF register is read.**
		0	No parity error is detected.
		1	A parity error occurred.
25	TIMEOUT		Time-out because of non-activation of $\overline{\text{SPIENA}}$ pin. The SPI generates a time-out when the slave does not respond in time by activating the ENA signal after the chip select signal has been activated. If a time-out condition is detected, the corresponding chip select is deactivated immediately and the TIMEOUT flag is set. In addition, the TIMEOUT flag in the status field of the corresponding buffer and in the SPI Flag Register (SPIFLG) is set. **This bit is valid only in master mode.** **Note: This flag is cleared to 0 when the RXDATA field of the SPIBUF register is read.**
		0	No $\overline{\text{SPIENA}}$ pin time-out occurred.
		1	An $\overline{\text{SPIENA}}$ signal time-out occurred.
24	DLENERR		Data length error flag. **Note: This flag is cleared to 0 when the RXDATA field of the SPIBUF register is read.**
		0	No data-length error occurred.
		1	A data length error occurred.
23-16	LCSNR	0-FFh	Last chip select number. LCSNR in the status field is a copy of CSNR in the corresponding control field. It contains the chip select number that was activated during the last word transfer.
15-0	RXDATA	0-FFFFh	SPI receive data. This is the received word, transferred from the receive shift-register at the end of a transfer. Regardless of the programmed character length and the direction of shifting, the received data is stored right-justified in the register.

28.3.18 SPI Emulation Register (SPIEMU)

Figure 28-49. SPI Emulation Register (SPIEMU) [offset = 44h]

31	16
Reserved	
R-8000h	
15	**0**
EMU_RXDATA	
R-0	

LEGEND: R = Read only; -*n* = value after reset

Table 28-27. SPI Emulation Register (SPIEMU) Field Descriptions

Bit	Field	Value	Description
31-16	Reserved	8000h	Reserved
15-0	EMU_RXDATA	0-FFFFh	SPI receive data. The SPI emulation register is a mirror of the SPIBUF register. The only difference between SPIEMU and SPIBUF is that a read from SPIEMU does not clear any of the status flags.

28.3.19 SPI Delay Register (SPIDELAY)

Figure 28-50. SPI Delay Register (SPIDELAY) [offset = 48h]

31 - 24	23 - 16
C2TDELAY	T2CDELAY
R/W-0	R/W-0
15 - 8	**7 - 0**
T2EDELAY	C2EDELAY
R/W-0	R/W-0

LEGEND: R/W = Read/Write; -*n* = value after reset

Table 28-28. SPI Delay Register (SPIDELAY) Field Descriptions

Bit	Field	Value	Description
31-24	C2TDELAY	0-FFh	Chip-select-active to transmit-start delay. See Figure 28-51 for an example. C2TDELAY is used only in master mode. It defines a setup time (for the slave device) that delays the data transmission from the chip select active edge by a multiple of VCLK cycles. The setup time value is calculated as follows. $t_{C2TDELAY}$ = (C2TDELAY + 2) × VCLK Period Example: VCLK = 25 MHz -> VCLK Period = 40ns; C2TDELAY = 07h; > $t_{C2TDELAY}$ = 360 ns When the chip select signal becomes active, the slave has to prepare data transfer within 360 ns. **Note: If phase = 1, the delay between SPICS falling edge to the first edge of SPICLK will have an additional 0.5 SPICLK period delay. This delay is as per the SPI protocol.**

Table 28-28. SPI Delay Register (SPIDELAY) Field Descriptions (continued)

Bit	Field	Value	Description
23-16	T2CDELAY	0-FFh	Transmit-end-to-chip-select-inactive-delay. See Figure 28-52 for an example. T2CDELAY is used only in master mode. It defines a hold time for the slave device that delays the chip select deactivation by a multiple of VCLK cycles after the last bit is transferred. The hold time value is calculated as follows: $t_{T2CDELAY}$ = (T2CDELAY +1) × VCLK Period Example: VCLK = 25 MHz -> VCLK Period = 40ns; T2CDELAY = 03h; > $t_{T2CDELAY}$ = 160 ns After the last data bit (or parity bit) is being transferred the chip select signal is held active for 160 ns. **Note: If phase = 0, then between the last edge of SPICLK and rise-edge of SPICS there will be an additional delay of 0.5 SPICLK period. This is as per the SPI protocol.** Both C2TDELAY and T2CDELAY counters do not have any dependency on the $\overline{\text{SPIENA}}$ pin value. Even if the $\overline{\text{SPIENA}}$ pin is asserted by the slave, the master will continue to delay the start of SPICLK until the C2TDELAY counter overflows. Similarly, even if the $\overline{\text{SPIENA}}$ pin is deasserted by the slave, the master will continue to hold the $\overline{\text{SPICS}}$ pins active until the T2CDELAY counter overflows. In this way, it is guaranteed that the setup and hold times of the $\overline{\text{SPICS}}$ pins are determined by the delay timers alone. To achieve better throughput, it should be ensured that these two timers are kept at the minimum possible values.
15-8	T2EDELAY	0-FFh	Transmit-data-finished to ENA-pin-inactive time-out. T2EDELAY is used in master mode only. It defines a time-out value as a multiple of SPI clock before $\overline{\text{SPIENA}}$ signal has to become inactive and after $\overline{\text{SPICS}}$ becomes inactive. SPICLK depends on which data format is selected. If the slave device is missing one or more clock edges, it becomes de-synchronized. In this case, although the master has finished the data transfer, the slave is still waiting for the missed clock pulses and the ENA signal is not disabled. The T2EDELAY defines a time-out value that triggers the DESYNC flag, if the $\overline{\text{SPIENA}}$ signal is not deactivated in time. The DESYNC flag is set to indicate that the slave device did not de-assert its $\overline{\text{SPIENA}}$ pin in time to acknowledge that it received all bits of the sent word. See Figure 28-53 for an example of this condition. **Note: DESYNC is also set if the SPI detects a de-assertion of $\overline{\text{SPIENA}}$ before the end of the transmission. The time-out value is calculated as follows:** $t_{T2EDELAY}$ = T2EDELAY/SPIclock Example: SPIclock = 8 Mbit/s; T2EDELAY = 10h; > $t_{T2EDELAY}$ = 2 µs The slave device has to disable the ENA signal within 2, otherwise DESYNC is set and an interrupt is asserted (if enabled).
7-0	C2EDELAY	0-FFh	Chip-select-active to ENA-signal-active time-out. C2EDELAY is used only in master mode and it applies only if the addressed slave generates an ENA signal as a hardware handshake response. C2EDELAY defines the maximum time between when the SPI activates the chip-select signal and the addressed slave has to respond by activating the ENA signal. C2EDELAY defines a time-out value as a multiple of SPI clocks. The SPI clock depends on whether data format 0 or data format 1 is selected. See Figure 28-54 for an example of this condition. **Note: If the slave device does not respond with the ENA signal before the time-out value is reached, the TIMEOUT flag in the SPIFLG register is set and a interrupt is asserted (if enabled).** If a time-out occurs, the SPI clears the transmit request of the timed-out buffer, sets the TIMEOUT flag for the current buffer, and continues with the transfer of the next buffer in the sequence that is enabled. The timeout value is calculated as follows: $t_{C2EDELAY}$ = C2EDELAY/SPIclock Example: SPIclock = 8 Mbit/s; C2EDELAY = 30 h; > $t_{C2EDELAY}$ = 6 ms The slave device has to activate the ENA signal within 6 ms after the SPI has activated the chip select signal ($\overline{\text{SPICS}}$), otherwise the TIMEOUT flag is set and an interrupt is asserted (if enabled).

Figure 28-51. Example: $t_{C2TDELAY}$= 8 VCLK Cycles

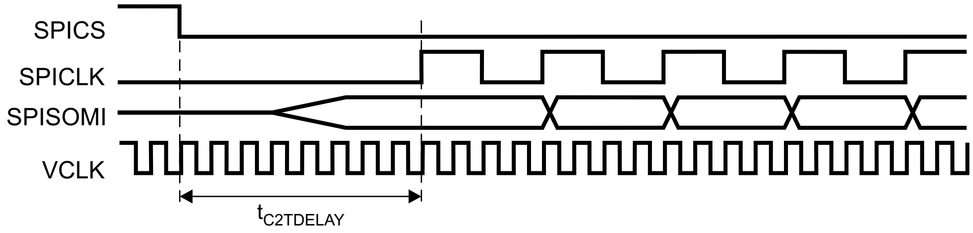


Figure 28-52. Example: $t_{T2CDELAY}$= 4 VCLK Cycles

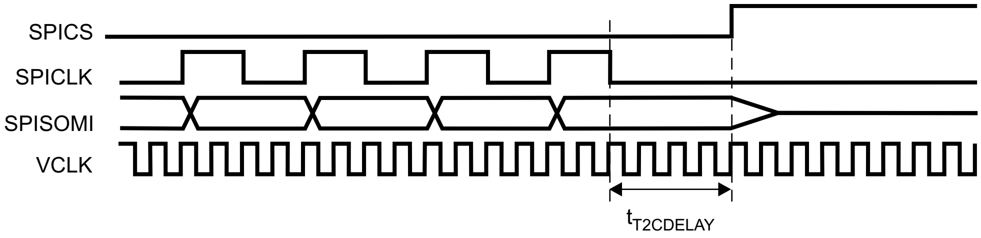


Figure 28-53. Transmit-Data-Finished-to-ENA-Inactive-Timeout

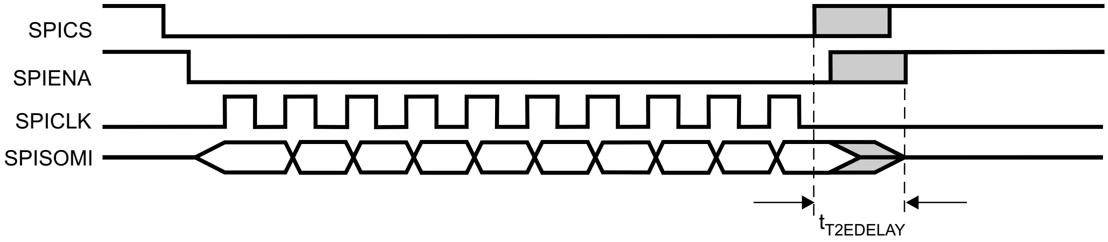
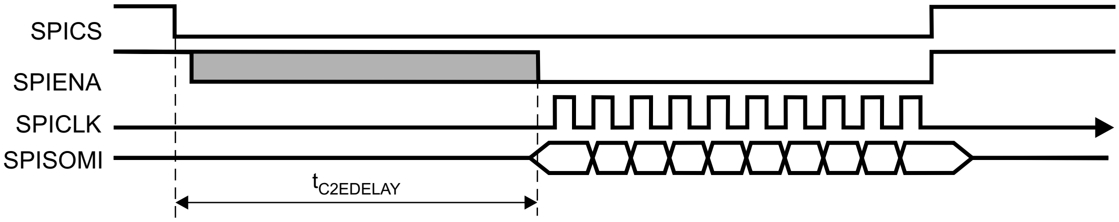


Figure 28-54. Chip-Select-Active-to-ENA-Signal-Active-Timeout



28.3.20 SPI Default Chip Select Register (SPIDEF)

Figure 28-55. SPI Default Chip Select Register (SPIDEF) [offset = 4Ch]

31 … 16
Reserved
R-0

15 … 8	7 … 0
Reserved	CSDEF
R-0	R/W-FFh

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 28-29. SPI Default Chip Select Register (SPIDEF) Field Descriptions

Bit	Field	Value	Description
31-8	Reserved	0	Reads return 0. Writes have no effect.
7-0	CDEF		Chip select default pattern. Master-mode only. The CSDEF bits are output to the $\overline{\text{SPICS}}$ pins when no transmission is being performed. It allows the user to set a programmable chip-select pattern that deselects all of the SPI slaves.
		0	$\overline{\text{SPICS}}$ is cleared to 0 when no transfer is active.
		1	$\overline{\text{SPICS}}$ is set to 1 when no transfer is active.

28.3.21 SPI Data Format Registers (SPIFMT[3:0])

Figure 28-56. SPI Data Format Registers (SPIFMTn) [offset = 5Ch-50h]

31-24
WDELAY
R/WP-0

23	22	21	20	19	18	17	16
PARPOL	PARITYENA	WAITENA	SHIFTDIR	HDUPLEX_ ENAx	DIS_CS_ TIMERS	POLARITY	PHASE
R/WP-0	R/WP-0	R/WP-0	R/WP-0	R/WP-0	R/WP-0	R/WP-0	R/WP-0

15-8	7-5	4-0
PRESCALE	Reserved	CHARLEN
R/WP-0	R-0	R/WP-0

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privilege mode only; -*n* = value after reset

Table 28-30. SPI Data Format Registers (SPIFMTn) Field Descriptions

Bit	Field	Value	Description
31-24	WDELAY	0-FFh	Delay in between transmissions for data format x (x= 0,1,2,3).Idle time that will be applied at the end of the current transmission if the bit WDEL is set in the current buffer. The delay to be applied is equal to: WDELAY × P_{VCLK} + 2 × P_{VCLK} P_{VCLK} -> Period of VCLK.
23	PARPOL		Parity polarity: even or odd. PARPOLx can be modified in privilege mode only. It can be used for data format x (x= 0,1,2,3).
		0	An even parity flag is added at the end of the transmit data stream.
		1	An odd parity flag is added at the end of the transmit data stream.
22	PARITYENA		Parity enable for data format x. No parity generation/ verification is performed for this data format.
		0	A parity bit is transmitted at the end of each transmitted word. At the end of a transfer the parity generator compares the received parity bit with the locally-calculated parity flag. If the parity bits do not match the RXERR flag is set in the corresponding control field. The parity type (even or odd) can be selected via the PARPOL bit.
		1	**Note: If an uncorrectable error flag is set in a slave-mode SPI, then the wrong parity bit will be transmitted to indicate to the master that there has been some issue with the data parity. The SOMI pins will be forced to transmit all 0s, and the parity bit will be transmitted as 1 if even parity is selected and as 0 if odd parity is selected (using the PARPOLx bit of this register). This behavior occurs regardless of an uncorrectable parity error on either TXRAM or RXRAM.**
21	WAITENA		The master waits for the ENA signal from slave for data format x. WAITENA is valid in master mode only. WAITENA enables a flexible SPI network where slaves with ENA signal and slaves without ENA signal can be mixed. WAITENA defines, for each transferred word, whether the addressed slave generates the ENA signal or not.
		0	The SPI does not wait for the ENA signal from the slave and directly starts the transfer.
		1	Before the SPI starts the data transfer it waits for the ENA signal to become low. If the ENA signal is not pulled down by the addressed slave before the internal time-out counter (C2EDELAY) overflows, then the master aborts the transfer and sets the TIMEOUT error flag.
20	SHIFTDIR		Shift direction for data format x. With bit SHIFTDIRx, the shift direction for data format x (x=0,1,2,3) can be selected.
		0	MSB is shifted out first.
		1	LSB is shifted out first.

Table 28-30. SPI Data Format Registers (SPIFMTn) Field Descriptions (continued)

Bit	Field	Value	Description
19	HDUPLEX_ENAx		Half Duplex transfer mode enable for Data Format x. This bit controls the I/O function of SOMI/SIMO lines for a specific requirement where in the case of Master mode, TX pin - SIMO will act as an RX pin, and in the case of Slave mode, RX pin - SIMO will act as a TX pin..
		0	Normal Full Duplex transfer.
		1	If MASTER = 1, SIMO pin will act as an RX pin (No TX possible) If MASTER = 0, SIMO pin will act as a TX pin (No RX possible).
			For all normal operations, HDUPLEX_ENAx bits should always remain 0. It is intended for the usage when the SIMO pin is used for both TX & RX operations at different times.
18	DIS_CS_TIMERS		Disable chip-select timers for this format. The C2TDELAY and T2CDELAY timers are by default enabled for all the data format registers. Using this bit, these timers can be disabled for a particular data format, if they are not required. When a master is handling multiple slaves, with varied set-up hold requirement, the application can selectively choose to include or not include the chip-select delay timers for any slaves.
		0	Both C2TDELAY and T2CDELAY counts are inserted for the chip selects.
		1	No C2TDELAY or T2CDELAY is inserted in the chip select timings.
17	POLARITY		SPI data format x clock polarity. POLARITYx defines the clock polarity of data format x. The following restrictions apply when switching clock phase and/or polarity: • In 3-pin/4-pin with nENA pin configuration of a slave SPI, the clock phase and polarity cannot be changed on-the-fly between two transfers. The slave should be reset and reconfigured if clock phase/polarity needs to be switched. In summary, SPI format switching is not fully supported in slave mode. • Even while using chip select pins, the polarity of SPICLK can be switched only while the slave is not selected by a valid chip select. The master SPI should ensure that while switching SPICLK polarity, it has deselected all of its slaves. Otherwise, the switching of SPICLK polarity may be incorrectly treated as a clock edge by some slaves.
		0	If POLARITYx is cleared to 0, the SPI clock signal is low-inactive, that is, before and after data transfer the clock signal is low.
		1	If POLARITYx is set to 1, the SPI clock signal is high-inactive, that is, before and after data transfer the clock signal is high.
16	PHASE		SPI data format x clock delay. PHASEx defines the clock delay of data format x.
		0	If PHASEx is cleared to 0, the SPI clock signal is not delayed versus the transmit/receive data stream. The first data bit is transmitted with the first clock edge and the first bit is received with the second (inverse) clock edge.
		1	If PHASEx is set to 1, the SPI clock signal is delayed by a half SPI clock cycle versus the transmit/receive data stream. The first transmit bit has to output prior to the first clock edge. The master and slave receive the first bit with the first edge.
15-8	PRESCALE		SPI data format x prescaler. PRESCALEx determines the bit transfer rate of data format x if the SPI is the network master. PRESCALEx is use to derive SPICLK from VCLK. If the SPI is configured as slave, PRESCALEx **does not need** to be configured. The clock rate for data format x can be calculated as: $BR_{Formatx}$ = VCLK / (PRESCALEx + 1) **Note: When PRESCALEx is cleared to 0, the SPI clock rate defaults to VCLK/2.**
7-5	Reserved	0	Reads return 0. Writes have no effect.
4-0	CHARLEN	0-1Fh	SPI data format x data-word length. CHARLENx defines the word length of data format x. Legal values are 0x02 (data word length = 2 bit) to 10h (data word length = 16). Illegal values, such as 00 or 1Fh are not allowed; their effect is indeterminate.

28.3.22 Interrupt Vector 0 (INTVECT0)

NOTE: The TG interrupt is not available in MibSPI in compatibility mode. Therefore, there is no possibility to access this register in compatibility mode.

Figure 28-57. Interrupt Vector 0 (NTVECT0) [offset = 60h]

31-16
Reserved
R-0

15-6	5-1	0
Reserved	INTVECT0	SUSPEND0
R-0	R-0	R-0

LEGEND: R = Read only; -*n* = value after reset

Table 28-31. Transfer Group Interrupt Vector 0 (INTVECT0)

Bit	Field	Value	Description
31-6	Reserved	0	Reads return 0. Writes have no effect.
5-1	INTVECT0		INTVECT0. Interrupt vector for interrupt line INT0. Returns the vector of the pending interrupt at interrupt line INT0. If more than one interrupt is pending, INTVECT0 always references the highest prior interrupt source first. **Note: This field reflects the status of the SPIFLG register in vector format. Any updates to the SPIFLG register will automatically cause updates to this field.**
		0	There is no pending interrupt.
		1h ÷ x	Transfer group x (x=0,..,15) has a pending interrupt. SUSPEND0 reflects the type of interrupt (*suspended* or *finished*).
		11h	Error Interrupt pending. The lower half of SPIFLG contains more details about the type of error.
		13h	The pending interrupt is a Receive Buffer Overrun interrupt.
		12h	**SPI mode:** The pending interrupt is a Receive Buffer Full interrupt. **Mib mode:** Reserved. This bit combination should not occur.
		14h	**SPI mode:** The pending interrupt is a Transmit Buffer Empty interrupt. **Mib mode:** Reserved. This bit combination should not occur.
		All Other Combinations	**SPI mode:** Reserved. These bit combinations should not occur.
0	SUSPEND0		Transfer suspended / Transfer finished interrupt flag. Every time INTVECT0 is read by the host, the corresponding interrupt flag of the referenced transfer group is cleared and INTVECT0 is updated with the vector coming next in the priority chain.
		0	The interrupt type is a transfer finished interrupt. In other words, the buffer array referenced by INTVECT0 has asserted an interrupt because all of data from the transfer group has been transferred.
		1	The interrupt type is a transfer suspended interrupt. In other words, the transfer group referenced by INTVECT0 has asserted an interrupt because the buffer to be transferred next is in suspend-to-wait mode.

NOTE: Reading from the INTVECT0 register when Transmit Empty is indicated does not clear the TXINTFLG flag in the SPI Flag Register (SPIFLG). Writing a new word to the SPIDATx register clears the Transmit Empty interrupt.

NOTE: In multi-buffer mode, INTVECT0 contains the interrupt for the highest priority transfer group. A read from INTVECT0 automatically causes the next-highest priority transfer group's interrupt status to get loaded into INTVECT0 and its corresponding SUSPEND flag to get loaded into SUSPEND0. The transfer group with the lowest number has the highest priority, and the transfer group with the highest number has the lowest priority.

Reading the INTVECT0 register when the RXOVRN interrupt is indicated in multi-buffer mode does not clear the RXOVRN flag and hence does not clear the vector. The RXOVRN interrupt vector may be cleared in multi-buffer mode either by write-clearing the RXOVRN flag in the SPI Flag Register (SPIFLG) or by reading the RXRAM Overrun Buffer Address Register (RXOVRN_BUF_ADDR).

28.3.23 Interrupt Vector 1 (INTVECT1)

NOTE: The TG interrupt is not available in SPI in compatibility mode compatibility mode. Therefore, there is no possibility to access this register in compatibility mode.

Figure 28-58. Interrupt Vector 1 (INTVECT1) [offset = 64h]

31-16
Reserved
R-0

15-6	5-1	0
Reserved	INTVECT1	SUSPEND1
R-0	R-0	R-0

LEGEND: R = Read only; -*n* = value after reset

Table 28-32. Transfer Group Interrupt Vector 1 (INTVECT1)

Bit	Field	Value	Description
31-6	Reserved	0	Reads return 0. Writes have no effect.
5-1	INTVECT1		INTVECT1. Interrupt vector for interrupt line INT1. Returns the vector of the pending interrupt at interrupt line INT1. If more than one interrupt is pending, INTVECT1 always references the highest prior interrupt source first. **Note: This field reflects the status of the SPIFLG register in vector format. Any updates to the SPIFLG register will automatically cause updates to this field.**
		0	There is no pending interrupt. **SPI mode only.**
		11h	Error Interrupt pending. The lower half of SPIINT1 contains more details about the type of error. **SPI mode only.**
		13h	The pending interrupt is a Receive Buffer Overrun interrupt. **SPI mode only.**
		12h	The pending interrupt is a Receive Buffer Full interrupt. **SPI mode only.**
		14h	The pending interrupt is a Transmit Buffer Empty interrupt. **SPI mode only.**
		All Other Combinations	Reserved. These bit combinations should not occur. **SPI mode only.**
0	SUSPEND1		Transfer suspended / Transfer finished interrupt flag. Every time INTVECT1 is read by the host, the corresponding interrupt flag of the referenced transfer group is cleared and INTVECT1 is updated with the vector coming next in the priority chain.
		0	The interrupt type is a transfer finished interrupt. In other words, the buffer array referenced by INTVECT1 has asserted an interrupt because all of data from the transfer group has been transferred.
		1	The interrupt type is a transfer suspended interrupt. In other words, the transfer group referenced by INTVECT1 has asserted an interrupt because the buffer to be transferred next is in suspend-to-wait mode.

NOTE: Reading from the INTVECT1 register when Transmit Empty is indicated does not clear the TXINTFLG flag in the SPI Flag Register (SPIFLG). Writing a new word to the SPIDATx register clears the Transmit Empty interrupt.

NOTE: In multi-buffer mode, INTVECT1 contains the interrupt for the highest priority transfer group. A read from INTVECT1 automatically causes the next-highest priority transfer group's interrupt status to get loaded into INTVECT1 and its corresponding SUSPEND flag to get loaded into SUSPEND1. The transfer group with the lowest number has the highest priority, and the transfer group with the highest number has the lowest priority.

Reading the INTVECT1 register when the RXOVRN interrupt is indicated in multi-buffer mode does not clear the RXOVRN flag and hence does not clear the vector. The RXOVRN interrupt vector may be cleared in multi-buffer mode either by write-clearing the RXOVRN flag in the SPI Flag Register (SPIFLG) or by reading the RXRAM Overrun Buffer Address Register (RXOVRN_BUF_ADDR).

28.3.24 SPI Pin Control Register 9 (SPIPC9)

SPIPC9 only applies to SPI2.

Figure 28-59. SPI Pin Control Register 9 (SPIPC9) [offset = 68h]

31 25	24	23 17	16
Reserved	SOMISRS0	Reserved	SIMOSRS0
R-0	R/W-0	R-0	R/W-0

15 12	11	10	9	8 0
Reserved	SOMISRS0	SIMOSRS0	CLKSRS	Reserved
R-0	R/W-0	R/W-0	R/W-0	R-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 28-33. SPI Pin Control Register 9 (SPIPC9) Field Descriptions

Bit	Field	Value	Description
31-25	Reserved	0	Reads return the value that was last written. Writes have no effect.
24	SOMISRS0		SPI2 SOMI[0] slew control. This bit controls between the fast or slow slew mode. **Note: Duplicate Control Bits for SPI2 SOMI[0]. Bit 24 is not physically implemented. It is a mirror of bit 11. Any write to bit 24 will be reflected on bit 11. When bit 24 and bit 11 are simultaneously written, the value of bit 11 will control the SPI2 SOMI[0] pin. The read value of bit 24 always reflects the value of bit 11.**
		0	Fast mode is enabled; the normal output buffer is used for this pin.
		1	Slow mode is enabled; slew rate control is used for this pin.
23-17	Reserved	0	Reads return the value that was last written. Writes have no effect.
16	SIMOSRS0		SPI2 SPISIMO[0] slew control. This bit controls between the fast or slow slew mode. **Note: Duplicate Control Bits for SPI2 SIMO[0]. Bit 16 is not physically implemented. It is a mirror of bit 10. Any write to bit 16 will be reflected on bit 10. When bit 16 and bit 10 are simultaneously written, the value of bit 10 will control the SPI2 SOMI[0] pin. The read value of bit 16 always reflects the value of bit 10.**
		0	Fast mode is enabled; the normal output buffer is used for this pin.
		1	Slow mode is enabled; slew rate control is used for this pin.
15-12	Reserved	0	Reads return 0. Writes have no effect.
11	SOMISRS0		SPI2 SOMI[0] slew control. This bit controls between the fast or slow slew mode.
		0	Fast mode is enabled; the normal output buffer is used for this pin.
		1	Slow mode is enabled; slew rate control is used for this pin.
10	SIMOSRS0		SPI2 SPISIMO[0] slew control. This bit controls between the fast or slow slew mode.
		0	Fast mode is enabled; the normal output buffer is used for this pin.
		1	Slow mode is enabled; slew rate control is used for this pin.
9	CLKSRS		SPI2 CLK slew control. This bit controls between the fast or slow slew mode.
		0	Fast mode is enabled; the normal output buffer is used for this pin.
		1	Slow mode is enabled; slew rate control is used for this pin.
8-0	Reserved	0	Reads return the value that was last written. Writes have no effect.

28.3.25 Parallel/Modulo Mode Control Register (SPIPMCTRL)

NOTE: Do not configure MODCLKPOLx and MMODEx bits since this device does not support modulo mode.

NOTE: The bits of this register are used in conjunction with the SPIFMTx registers. Each byte of this register corresponds to one of the SPIFMTx registers.

1. Byte0 (Bits 7:0) are used when SPIFMT0 register is selected by DFSEL[1:0] = 00 in the control field of a buffer.

2. Byte1 (Bits 15:8) are used when SPIFMT1 register is selected by DFSEL[1:0] = 01 in the control field of a buffer.

3. Byte2 (Bits 23:16) are used when SPIFMT2 register is selected by DFSEL[1:0] = 10 in the control field of a buffer.

4. Byte3 (Bits31:24) are used when SPIFMT3 register is selected by DFSEL[1:0] = 11 in the control field of a buffer.

Figure 28-60. Parallel/Modulo Mode Control Register (SPIPMCTRL) [offset = 6Ch]

31 30	29	28 26	25 24
Reserved	MODCLKPOL3	MMODE3	PMODE3
R-0	R/WP-0	R/WP-0	R/WP-0

23 22	21	20 18	17 16
Reserved	MODCLKPOL2	MMODE2	PMODE2
R-0	R/WP-0	R/WP-0	R/WP-0

15 14	13	12 10	9 8
Reserved	MODCLKPOL1	MMODE1	PMODE1
R-0	R/WP-0	R/WP-0	R/WP-0

7 6	5	4 2	1 0
Reserved	MODCLKPOL0	MMODE0	PMODE0
R-0	R/WP-0	R/WP-0	R/WP-0

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privilege mode only; -*n* = value after reset

Table 28-34. SPI Parallel/Modulo Mode Control Register (SPIPMCTRL) Field Descriptions

Bit	Field	Value	Description
31-30	Reserved	0	Reads return 0. Writes have no effect.
29	MODCLKPOL3		Modulo mode SPICLK polarity. This bit determines the polarity of the SPICLK in modulo mode only. If the MMODE3 bits are 000, this bit will be ignored.
		0	Normal SPICLK in all the modes.
		1	Polarity of the SPICLK will be inverted if Modulo mode is selected.
28-26	MMODE3		These bits determine whether the SPI/MibSPI operates with 1, 2, 4, 5, or 6 data lines (if modulo option is supported by the module).
		0	Normal single data line mode (default). (PMODE3 should be set to 00).
		1h	2-data line mode (PMODE3 should be set to 00).
		2h	3-data line mode (PMODE3 should be set to 00).
		3h	4-data line mode (PMODE3 should be set to 00).
		4h	5-data line mode (PMODE3 should be set to 00).
		5h	6-data line mode (PMODE3 should be set to 01).
		6h-7h	Reserved

Table 28-34. SPI Parallel/Modulo Mode Control Register (SPIPMCTRL) Field Descriptions (continued)

Bit	Field	Value	Description
25-24	PMODE3		Parallel mode bits determine whether the SPI/MibSPI operates with 1, 2, 4, or 8 data lines.
		0	Normal operation/1-data line (MMODE3 should be set to 000).
		1h	2-data line mode (MMODE3 should be set to 000).
		2h	4-data line mode (MMODE3 should be set to 000).
		3h	8-data line mode (MMODE3 should be set to 000).
23-22	Reserved	0	Reads return 0. Writes have no effect.
21	MODCLKPOL2		Modulo mode SPICLK polarity. This bit determines the polarity of the SPICLK in modulo mode only. If the MMODE2 bits are 000, this bit will be ignored.
		0	Normal SPICLK in all the modes.
		1	Polarity of the SPICLK will be inverted if Modulo mode is selected.
20-18	MMODE2		These bits determine whether the SPI/MibSPI operates with 1, 2, 4, 5, or 6 data lines (if modulo option is supported by the module).
		0	1-data line mode (default). (PMODE2 should be set to 00).
		1h	2-data line mode (PMODE2 should be set to 00).
		2h	3-data line mode (PMODE2 should be set to 00).
		3h	4-data line mode (PMODE2 should be set to 00).
		4h	5-data line mode (PMODE2 should be set to 00).
		5h	6-data line mode (PMODE2 should be set to 01).
		6h-7h	Reserved
17-16	PMODE2		Parallel mode bits determine whether the SPI/MibSPI operates with 1, 2, 4, or 8 data lines.
		0	Normal operation/1-data line (MMODE2 should be set to 000).
		1h	2-data line mode (MMODE2 should be set to 000).
		2h	4-data line mode (MMODE2 should be set to 000).
		3h	8-data line mode (MMODE2 should be set to 000).
15-12	Reserved	0	Reads return 0. Writes have no effect.
13	MODCLKPOL1		Modulo mode SPICLK polarity. This bit determines the polarity of the SPICLK in modulo mode only. If the MMODE1 bits are 000, this bit will be ignored.
		0	Normal SPICLK in all the modes.
		1	Polarity of the SPICLK will be inverted if Modulo mode is selected.
12-10	MMODE1		These bits determine whether the SPI/MibSPI operates with 1, 2, 4, 5, or 6 data lines (if modulo option is supported by the module).
		0	1-data line mode (default). (PMODE1 should be set to 00).
		1h	2-data line mode (PMODE1 should be set to 00).
		2h	3-data line mode (PMODE1 should be set to 00).
		3h	4-data line mode (PMODE1 should be set to 00).
		4h	5-data line mode (PMODE1 should be set to 00).
		5h	6-data line mode (PMODE1 should be set to 01).
		6h-7h	Reserved
9-8	PMODE1		Parallel mode bits determine whether the SPI/MibSPI operates with 1, 2, 4, or 8 data lines.
		0	Normal operation/1-data line (MMODE1 should be set to 000).
		1h	2-data line mode (MMODE1 should be set to 000).
		2h	4-data line mode (MMODE1 should be set to 000).
		3h	8-data line mode (MMODE1 should be set to 000).
7-6	Reserved	0	Reads return 0. Writes have no effect.
5	MODCLKPOL0		Modulo mode SPICLK polarity. This bit determines the polarity of the SPICLK in modulo mode only. If the MMODE0 bits are 000, this bit will be ignored.
		0	Normal SPICLK in all the modes.
		1	Polarity of the SPICLK will be inverted if Modulo mode is selected.

Table 28-34. SPI Parallel/Modulo Mode Control Register (SPIPMCTRL) Field Descriptions (continued)

Bit	Field	Value	Description
4-2	MMODE0		These bits determine whether the SPI/MibSPI operates with 1, 2, 4, 5, or 6 data lines (if modulo option is supported by the module).
		0	1-data line mode (default). (PMODE0 should be set to 00).
		1h	2-data line mode (PMODE0 should be set to 00).
		2h	3-data line mode (PMODE0 should be set to 00).
		3h	4-data line mode (PMODE0 should be set to 00).
		4h	5-data line mode (PMODE0 should be set to 00).
		5h	6-data line mode (PMODE0 should be set to 01).
		6h-7h	Reserved
1-0	PMODE0		Parallel mode bits determine whether the SPI/MibSPI operates with 1, 2, 4, or 8 data lines.
		0	Normal operation/1-data line (MMODE0 should be set to 000).
		1h	2-data line mode (MMODE0 should be set to 000).
		2h	4-data line mode (MMODE0 should be set to 000).
		3h	8-data line mode (MMODE0 should be set to 000).

NOTE: Accessibility of Registers

Registers from this offset address onwards are not accessible in SPI compatibility mode. They are accessible only in the multi-buffer mode.

28.3.26 Multi-buffer Mode Enable Register (MIBSPIE)

NOTE: Accessibility of Multi-Buffer RAM

The multi-buffer RAM is not accessible unless the MSPIENA bit is set to 1. The only exception to this is in test mode, where, by setting RXRAMACCESS to 1, the multi-buffer RAM can be fully accessed for both read and write.

Figure 28-61. Multi-buffer Mode Enable Register (MIBSPIE) [offset = 70h]

31-17	16
Reserved	RXRAM_ACCESS
R-0	R/WP-0

15-12	11-8	7-1	0
Reserved	EXTENDED_BUF_ENA	Reserved	MSPIENA
R-0	R/WP-5h		R/WP-0

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privilege mode only; -*n* = value after reset

Table 28-35. Multi-buffer Mode Enable Register (MIBSPIE) Field Descriptions

Bit	Field	Value	Description
31-17	Reserved	0	Reads return 0. Writes have no effect.
16	RXRAM_ACCESS		Receive-RAM access control. During normal operating mode of SPI, the receive data/status portion of multi-buffer RAM is read-only. To enable testing of receive RAM, direct read/write access is enabled by setting this bit.
		0	The RX portion of multi-buffer RAM is not writable by the CPU.
		1	The whole of multi-buffer RAM is fully accessible for read/write by the CPU.
			Note: The RX RAM ACCESS bit remains 0 after reset and it should remain set to 0 at all times, except when testing the RAM. SPI should be given a local reset by using the nRESET (SPIGCR0[0]) bit after RAM testing is performed so that the multi-buffer RAM gets re-initialized.
15-12	Reserved	0	Reads return 0. Writes have no effect.
11-8	EXTENDED_BUF_ENA		Enables the support for 256 buffers. By default MibSPI supports up to 128 buffers for both TX and RX. Refer to the device specific datasheet if 256 buffer extension is implemented for the specific MibSPI instance in the device.
		5h	Write: Disables the Extended Buffer mode - MibSPI supports only 128 buffers (default).
		Ah	Write: Enables the Extended Buffer mode - up to 256 buffers can be used.
		all others	All other values - writes are ignored and the values are not updated into this field. The state of the feature remains unchanged.
			Read: Returns the current value of this field.
7-1	Reserved	0	Reads return 0. Writes have no effect.
0	MSPIENA		Multi-buffer mode enable. After power-up or reset, MSPIENA remains cleared, which means that the SPI runs in compatibility mode by default. If multi-buffer mode is desired, this register should be configured first after configuring the SPIGCR0 register. If MSPIENA is not set to 1, the multi-buffer mode registers are not writable.
		0	The SPI runs in compatibility mode, that is, in this mode the MibSPI is fully code-compliant to the standard device SPI. No multi-buffered-mode features are supported.
		1	The SPI is configured to run in multi-buffer mode.

28.3.27 TG Interrupt Enable Set Register (TGITENST)

The register TGITENST contains the TG interrupt enable flags for transfer-finished and for transfer-suspended events. Each of the enable bits in the higher half-word and the lower half-word of TGITENST belongs to one TG.

The register map shown in Figure 28-62 and Table 28-36 represents a super-set device with the maximum number of TGs (16) assumed. The actual number of bits available varies per device.

Figure 28-62. TG Interrupt Enable Set Register (TGITENST) [offset = 74h]

31	16
SETINTENRDY[15:0]	
R/W-0	

15	0
SETINTENSUS[15:0]	
R/W-0	

LEGEND: R/W = Read/Write; -*n* = value after reset

Table 28-36. TG Interrupt Enable Set Register (TGITENST) Field Descriptions

Bit	Field	Value	Description
31-16	SETINTENRDY[*n*]		TG interrupt set (enable) when transfer finished. Bit 16 corresponds to TG0, bit 17 corresponds to TG1, and so on.
		0	Read: The TGx-completed interrupt is disabled. This interrupt does not get generated when TGx completes. Write: A write of 0 to this bit has no effect.
		1	Read: The TGx-completed interrupt is enabled. The interrupt gets generated when TGx completes. Write: Enable the TGx-completed interrupt. The interrupt gets generated when TGx completes.
15-0	SETINTENSUS[*n*]		TG interrupt set (enabled) when transfer suspended. Bit 0 corresponds to TG0, bit 1 corresponds to TG1, and so on.
		0	Read: The TGx-completed interrupt is disabled. This interrupt does not get generated when TGx is suspended. Write: A write of 0 to this bit has no effect.
		1	Read: The TGx-completed interrupt is enabled. The interrupt gets generated when TGx is suspended. Write: Enable the TGx-completed interrupt. The interrupt gets generated when TGx is suspended.

28.3.28 TG Interrupt Enable Clear Register (TGITENCR)

The register TGITENCR is used to clear the interrupt enables for the TG-completed interrupt and the TG-suspended interrupts.

The register map shown in Figure 28-63 and Table 28-37 represents a super-set device with the maximum number of TGs (16) assumed. The actual number of bits available varies per device.

Figure 28-63. TG Interrupt Enable Clear Register (TGITENCR) [offset = 78h]

31	16
CLRINTENRDY[15:0]	
R/W-0	
15	0
CLRINTENSUS[15:0]	
R/W-0	

LEGEND: R/W = Read/Write; -*n* = value after reset

Table 28-37. TG Interrupt Enable Clear Register (TGITENCR) Field Descriptions

Bit	Field	Value	Description
31-16	CLRINTENRDY[*n*]		TG interrupt clear (disabled) when transfer finished. Bit 16 corresponds to TG0, bit 17 corresponds to TG1, and so on.
		0	Read: The TGx-completed interrupt is disabled. This interrupt does not get generated when TGx completes. Write: A write of 0 to this bit has no effect.
		1	Read: The TGx-completed interrupt is enabled. The interrupt gets generated when TGx completes. Write: Disable the TGx-completed interrupt. The interrupt does not get generated when TGx completes.
15-0	CLRINTENSUS[*n*]		TG interrupt clear (disabled) when transfer suspended. Bit 0 corresponds to TG0, bit 1 corresponds to TG1, and so on.
		0	Read: The TGx-completed interrupt is disabled. This interrupt does not get generated when TGx is suspended. Write: A write of 0 to this bit has no effect.
		1	Read: The TGx-completed interrupt is enabled. The interrupt gets generated when TGx is suspended. Write: Disable the TGx-completed interrupt. The interrupt does not get generated when TGx is suspended.

28.3.29 Transfer Group Interrupt Level Set Register (TGITLVST)

The register TGITLVST sets the level of interrupts for transfer completed interrupt and for transfer suspended interrupt to level 1.

The register map shown in Figure 28-64 andTable 28-38 represents a super-set device with the maximum number of TGs (16) assumed. The actual number of bits available varies per device.

Figure 28-64. Transfer Group Interrupt Level Set Register (TGITLVST) [offset = 7Ch]

31	16
SETINTLVLRDY[15:0]	
R/W-0	

15	0
SETINTLVLSUS[15:0]	
R/W-0	

LEGEND: R/W = Read/Write; -*n* = value after reset

Table 28-38. Transfer Group Interrupt Level Set Register (TGITLVST) Field Descriptions

Bit	Field	Value	Description
31-16	SETINTLVLRDY[*n*]		Transfer-group completed interrupt level set. Bit 16 corresponds to TG0, bit 17 corresponds to TG1, and so on.
		0	Read: The TGx-completed interrupt is set to INT0. Write: A write of 0 to this bit has no effect.
		1	Read: The TGx-completed interrupt is set to INT1. Write: Set the TGx-completed interrupt to INT1.
15-0	SETINTLVLSUS[*n*]		Transfer-group suspended interrupt level set. Bit 0 corresponds to TG0, bit 1 corresponds to TG1, and so on.
		0	Read: The TGx-suspended interrupt is set to INT0. Write: A write of 0 to this bit has no effect.
		1	Read: The TGx-suspended interrupt is set to INT1. Write: Set the TG-x suspended interrupt to INT1.

28.3.30 Transfer Group Interrupt Level Clear Register (TGITLVCR)

The register TGITLVCR clears the level of interrupts for transfer completed interrupt and for transfer suspended interrupt to level 0.

The register map shown in Figure 28-65 and Table 28-39 represents a super-set device with the maximum number of TGs (16) assumed. The actual number of bits available varies per device.

Figure 28-65. Transfer Group Interrupt Level Clear Register (TGITLVCR) [offset = 80h]

31	16
CLRINTLVLRDY[15:0]	
R/W-0	

15	0
CLRINTLVLSUS[15:0]	
R/W-0	

LEGEND: R/W = Read/Write; -*n* = value after reset

Table 28-39. Transfer Group Interrupt Level Clear Register (TGITLVCR) Field Descriptions

Bit	Field	Value	Description
31-16	CLRINTLVLRDY[*n*]		Transfer-group completed interrupt level clear. Bit 16 corresponds to TG0, bit 17 corresponds to TG1, and so on.
		0	Read: The TGx-completed interrupt is set to INT0. Write: A write of 0 to this bit has no effect.
		1	Read: The TGx-completed interrupt is set to INT1. Write: Clear the TGx-completed interrupt to INT0.
15-0	CLRINTLVLSUS[*n*]		Transfer group suspended interrupt level clear. Bit 0 corresponds to TG0, bit 1 corresponds to TG1, and so on.
		0	Read: TGx-suspended interrupt is set to INT0. Write: A write of 0 to this bit has no effect.
		1	Read: The TGx-suspended interrupt is set to INT1. Write: Clear the TG-x suspended interrupt to INT0.

28.3.31 Transfer Group Interrupt Flag Register (TGINTFLAG)

The TGINTFLAG register comprises the transfer group interrupt flags for transfer-completed interrupts (INTFLGRDYx) and for transfer-suspended interrupts (INTFLGSUSx). Each of the interrupt flags in the higher half-word and the lower half-word of TGINTFLAG belongs to one TG.

The register map shown in Figure 28-66 and Table 28-40 represents a super-set device with the maximum number of TGs (16) assumed. The actual number of bits available varies per device.

Figure 28-66. Transfer Group Interrupt Flag Register (TGINTFLAG) [offset = 84h]

31	16
INTFLGRDY[15:0]	
R/W1C-0	

15	0
INTFLGSUS[15:0]	
R/W1C-0	

LEGEND: R/W = Read/Write; W1C = Write 1 to clear; -*n* = value after reset

Table 28-40. Transfer Group Interrupt Level Clear Register (TGITLVCR) Field Descriptions

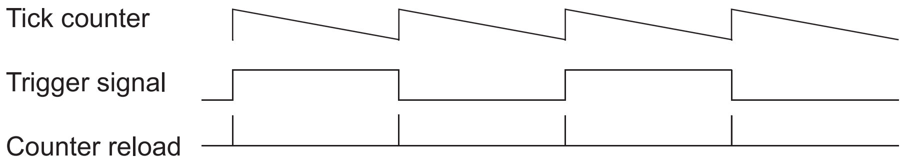
Bit	Field	Value	Description
31-16	INTFLGRDY[*n*]		Transfer-group interrupt flag for a transfer-completed interrupt. Bit 16 corresponds to TG0, bit 17 corresponds to TG1, and so on. **Note: Read Clear Behavior. Reading the interrupt vector registers TGINTVECT0 or TGINTVECT1 automatically clears the interrupt flag bit INTFLGRDYx referenced by the vector number given by INTVECT0/INTVECT1 bits, if the SUSPEND[0:1] bit in the vector registers is 0.**
		0	Read: No transfer-completed interrupt occurred since last clearing of the INTFLGRDYx flag. Write: A write of 0 to this bit has no effect.
		1	Read: A transfer finished interrupt from transfer group x occurred. No matter whether the interrupt is enabled or disabled (INTENRDYx = don't care) or whether the interrupt is mapped to INT0 or INT1, INTFLGRDYx is set right after the transfer from TGx is finished. Write: The corresponding bit flag is cleared.
15-0	INTFLGSUS[*n*]		Transfer-group interrupt flag for a transfer-suspend interrupt. Bit 0 corresponds to TG0, bit 1 corresponds to TG1, and so on. **Note: Read Clear Behavior. Reading the interrupt vector registers TGINTVECT0 or TGINTVECT1 automatically clears the interrupt flag bit INTFLGSUSx referenced by the vector number given by INTVECT0/INTVECT1 bits, if the SUSPEND[0:1] bit in the corresponding vector registers is 1.**
		0	Read: No transfer-suspended interrupt occurred since the last clearing of the INTFLGSUSx flag. Write: A write of 0 to this bit has no effect.
		1	Read: A transfer-suspended interrupt from TGx occurred. No matter whether the interrupt is enabled or disabled (INTENSUSx = don't care) or whether the interrupt is mapped to INT0 or INT1, INTFLGSUSx is set right after the transfer from transfer group x is suspended. Write: The corresponding bit flag is cleared.

28.3.32 Tick Count Register (TICKCNT)

One of the trigger sources for TGs is an internal periodic time trigger. This time trigger is called a tick counter and is basically a down-counter with a preload/reload value. Every time the tick counter detects an underflow it reloads the initial value and toggles the trigger signal provided to the TGs.

The trigger signal, shown in Figure 28-67 as a square wave, illustrates the different trigger event types for the TGs (for example, rising edge, falling edge, and both edges).

Figure 28-67. Tick Counter Operation



This register is shown in Figure 28-68 and described in Table 28-41.

Figure 28-68. Tick Count Register (TICKCNT) [offset = 90h]

31	30	29-28	27-16
TICKENA	RELOAD	CLKCTRL	Reserved
R/W-0	R/S-0	R/W-0	R-0

15-0
TICKVALUE
R/W-0

LEGEND: R = Read only; R/W = Read/Write; S = Set; -*n* = value after reset

Table 28-41. Tick Count Register (TICKCNT) Field Descriptions

Bit	Field	Value	Description
31	TICKENA		Tick counter enable.
		0	The internal tick counter is disabled. The counter value remains unchanged. **Note: When the tick counter is disabled, the trigger signal is forced low.**
		1	The internal tick counter is enabled and is clocked by the clock source selected by CLKCTRL. When TICKENA goes from 0 to 1, the tick counter is automatically loaded with the contents of TICKVALUE.
30	RELOAD		Pre-load the tick counter. RELOAD is a set-only bit; writing a 1 reloads the tick counter with the value stored in TICKVALUE. Reading RELOAD always returns a 0. **Note: When the tick counter is reloaded by the RELOAD bit, the trigger signal is not toggled.**
29-28	CLKCTRL		Tick counter clock source control. CLKCTRL defines the clock source that is used to clock the internal tick counter.
		0	SPICLK of data word format 0 is selected as the clock source of the tick counter.
		1h	SPICLK of data word format 1 is selected as the clock source of the tick counter.
		2h	SPICLK of data word format 2 is selected as the clock source of the tick counter.
		3h	SPICLK of data word format 3 is selected as the clock source of the tick counter.
27-16	Reserved	0	Reads return 0. Writes have no effect.
15-0	TICKVALUE	0-FFFFh	Initial value for the tick counter. TICKVALUE stores the initial value for the tick counter. The tick counter is loaded with the contents of TICKVALUE every time an underflow condition occurs and every time the RELOAD flag is set by the host.

28.3.33 Last TG End Pointer (LTGPEND)

Figure 28-69. Last TG End Pointer (LTGPEND) [offset = 94h]

31 29	28 24	23 16
Reserved	TG IN SERVICE	Reserved
R-0	R-0	R-0

15 8	7 0
LPEND	Reserved
R/W-0	R-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 28-42. Last TG End Pointer (LTGPEND) Field Descriptions

Bit	Field	Value	Description
31-29	Reserved	0	Reads return 0. Writes have no effect.
28-24	TG IN SERVICE		The TG number currently being serviced by the sequencer. These bits indicate the current TG that is being serviced. This field can generally be used for code debugging.
		0	No TG is being serviced by the sequencer.
		1h	TG0 is being serviced by the sequencer.
		:	:
		10h	TG15 is being serviced by the sequencer. **Note: The number of transfer groups varies by device.**
		11h-1Fh	Invalid values.
23-16	Reserved	0	Reads return 0. Writes have no effect.
15-8	LPEND	0-FFh	Last TG end pointer. Usually the TG end address (PEND) is inherently defined by the start value of the starting pointer of the subsequent TG (PSTART). The TG ends one word before the next TG starts (PEND[x] = PSTART[x+1] - 1). For a full configuration of MibSPI, the last TG has no subsequent TG, that is, no end address is defined. Therefore, LPEND has to be programmed to specify explicitly the end address of the last TG. **Note: For MibSPI1 that supports 256 buffers (values from 0-FFh), bit 15 is used. For MibSPI2-5 that support 128 buffers (values from 0-7Fh), bit 15 is reserved.** **Note: When using all 8 transfer groups, program the LPEND bits to define the end of the last transfer group. When using less than 8 transfer groups, leave the LPEND bits programmed to point to the end of the buffer and create a dummy transfer group that defines the end of your last intentional transfer group and occupies all the remaining buffer space.**
7-0	Reserved	0	Reads return 0. Writes have no effect.

28.3.34 TGx Control Registers (TGxCTRL)

Each TG can be configured via one dedicated control register. The register description shows one control register (x) that is identical for all TGs. For example, the control register for TG2 is named TG2CTRL and is located at *base address + 98h + 4 × 2*. The actual number of available control registers varies by device.

Figure 28-70. MibSPI TG Control Registers (TGxCTRL) [offsets = 98h-D4h]

31	30	29	28	27-24	23-20	19-16
TGENA	ONESHOT	PRST	TGTD	Reserved	TRIGEVT	TRIGSRC
R/W-0	R/W-0	R/W-0	R-0	R-0	R/W-0	R/W-0

15-8	7-0
PSTART	PCURRENT
R/W-0	R-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 28-43. TG Control Registers (TGxCTRL) Field Descriptions

Bit	Field	Value	Description
31	TGENA		TGx enable.
			If the correct event (TRIGEVTx) occurs at the selected source (TRIGSRCx) a group transfer is initiated if no higher priority TG is in active transfer mode or if one or more higher-priority TGs are in transfer-suspend mode.
			Disabling a TG while a transfer is ongoing will finish the ongoing word transfer but not the whole group transfer.
		0	TGx is disabled.
		1	TGx is enabled.
30	ONESHOTx		Single transfer for TGx.
		0	TGx initiates a transfer every time a trigger event occurs and TGENA is set.
		1	A transfer from TGx will be performed only once (one shot) after a valid trigger event at the selected trigger source. After the transfer is finished the TGENAx control bit will be cleared and therefore no additional transfer can be triggered before the host enables the TG again. This one shot mode ensures that after one group transfer the host has enough time to read the received data and to provide new transmit data.
29	PRSTx		TGx pointer reset mode. Configures the way to resolve trigger events during an ongoing transfer. This bit is meaningful only for level-triggered TGs. Edge-triggered TGs cannot be restarted before their completion by another edge. The PRST bit will have no effect on this behavior.
			Note: When the PRST bit is set, if the buffer being transferred at the time of a new trigger event is a LOCK, CSHOLD or NOBRK buffer, then only after finishing those transfers, the TG will be restarted. This means that even if the TG is retriggered, the TG will only be restarted after finishing the transfer of the first non-LOCK or non-CSHOLD buffer. In the case of the NOBRK buffer, after completing the ICOUNT number of transfers, the TG will be restarted from its PSTART.
			This means that TX control fields such as LOCK and CSHOLD, and DMA control fields such as NOBRK have higher priority over anything else. They have the capability to delay the restart of the TG even if it is retriggered when PRST is 1.
		0	If a trigger event occurs during a transfer from TGx, the event is ignored and is not stored internally. The TGx transfer has priority over additional trigger events.
		1	The TGx pointer (PCURRENTx) will be reset to the start address (PSTARTx) when a valid trigger event occurs at the selected trigger source while a transfer from the same TG is ongoing. Every trigger event resets PCURRENTx no matter whether the concerned TG is in transfer mode or not. The trigger events have priority over the ongoing transfer.
28	TGTDx		TG triggered.
		0	TGx has not been triggered or is no longer waiting for service.
		1	TGx has been triggered and is either currently being serviced or waiting for servicing.
27-24	Reserved	0	Reads return 0. Writes have no effect.

Table 28-43. TG Control Registers (TGxCTRL) Field Descriptions (continued)

Bit	Field	Value	Description	
23-20	TRIGEVTx		Type of trigger event. A level-triggered TG can be stopped by de-activating the level trigger. However, the following restrictions apply. • Deactivating the level trigger for a TG during a NOBRK transfer does not stop the transfers until all of the ICOUNT number of buffers are transferred for the NOBRK buffer. Once a NOBRK buffer is prefetched, the trigger event loses control over the TG until the NOBRK buffer transfer is completed. • Once the transfer of a buffer with CSHOLD or LOCK bit set starts, deactivating the trigger level does not stop the transfer until the sequencer completes the transfer of the next non-CSHOLD or non-LOCK buffer in the same TG. • Once the last buffer in a TG is pre-fetched, de-activating the trigger level does not stop the transfer group until the last buffer transfer is completed. This means even if the trigger level is deactivated at the beginning of the penultimate (one-before-last) buffer transfer, the sequencer continues with the same TG until it is completed.	
		0	never	Never trigger TGx. This is the default value after reset.
		1h	rising edge	A rising edge (0 to 1) at the selected trigger source (TRIGSRCx) initiates a transfer for TGx.
		2h	falling edge	A falling edge (1 to 0) at the selected trigger source (TRIGSRCx) initiates a transfer for TGx.
		3h	both edges	Rising and falling edges at the selected trigger source (TRIGSRCx) initiates a transfer for TGx.
		4h	Rsvd	Reserved
		5h	high-active	While the selected trigger source (TRIGSRCx) is at a logic high level (1) the group transfer is continued and at the end of one group transfer restarted at the beginning. If the logic level changes to low (0) during an ongoing group transfer, the whole group transfer will be stopped. **Note: If ONESHOTx is set the transfer is performed only once.**
		6h	low-active	While the selected trigger source (TRIGSRCx) is at a logic low level (0) the group transfer is continued and at the end of one restarted at the beginning. If the logic level changes to high (1) during an ongoing group transfer, the whole group transfer will be stopped. **Note: If ONESHOTx is set the transfer is performed only once.**
		7h	always	A repetitive group transfer will be performed. **Note: By setting the TRIGSRC to 0, the TRIGEVT to 7h (ALWAYS), and the ONESHOTx bit to 1, software can trigger this TG. Upon setting the TGENA bit, the TG is immediately triggered.** **Note: If ONESHOTx is set the transfer is performed only once.**
		1xxx	Rsvd	Reserved

Table 28-43. TG Control Registers (TGxCTRL) Field Descriptions (continued)

Bit	Field	Value	Description	
19-16	TRIGSRCx		Trigger source. After reset, the trigger sources of all TGs are disabled.	
		0	Disabled	
		1h	EXT0	External trigger source 0. The actual source varies per device (for example, HET I/O channel, event pin).
		2h	EXT1	External trigger source 1. The actual source varies per device (for example, HET I/O channel, event pin).
		3h	EXT2	External trigger source 2. The actual source varies per device (for example, HET I/O channel, event pin).
		4h	EXT3	External trigger source 3. The actual source varies per device (for example, HET I/O channel, event pin).
		5h	EXT4	External trigger source 4. The actual source varies per device (for example, HET I/O channel, event pin).
		6h	EXT5	External trigger source 5. The actual source varies per device (for example, HET I/O channel, event pin).
		7h	EXT6	External trigger source 6. The actual source varies per device (for example, HET I/O channel, event pin).
		8h	EXT7	External trigger source 7. The actual source varies per device (for example, HET I/O channel, event pin).
		9h	EXT8	External trigger source 8. The actual source varies per device (for example, HET I/O channel, event pin).
		Ah	EXT9	External trigger source 9. The actual source varies per device (for example, HET I/O channel, event pin).
		Bh	EXT10	External trigger source 10. The actual source varies per device (for example, HET I/O channel, event pin).
		Ch	EXT11	External trigger source 11. The actual source varies per device (for example, HET I/O channel, event pin).
		Dh	EXT12	External trigger source 12. The actual source varies per device (for example, HET I/O channel, event pin).
		Eh	EXT13	External trigger source 13. The actual source varies per device (for example, HET I/O channel, event pin).
		Fh	TICK	Internal periodic event trigger. The tick counter can initiate periodic group transfers.
15-8	PSTARTx	0-FFh	TG start address. PSTARTx stores the start address of the corresponding TG. The corresponding end address is inherently defined by the subsequent TG start address minus 1 (PENDx[TGx] = PSTARTx[TGx+1]-1). PSTARTx is copied into PCURRENTx when: • The TG is enabled. • The end of the TG is reached during a transfer. • A trigger event occurs while PRST is set to 1. **Note: For MibSPI1 that supports 256 buffers (values from 0-FFh), bit 15 is used. For MibSPI2-5 that support 128 buffers (values from 0-7Fh), bit 15 is reserved.**	
7-0	PCURRENTx	0-FFh	Pointer to current buffer. PCURRENT is read-only. PCURRENTx stores the address of the buffer that corresponds to this TG. If the TG switches from active transfer mode to suspend to wait, PCURRENTx contains the address of the currently suspended word. After the TG resumes from suspend to wait mode, the next buffer will be transferred; that is, no buffer data is transferred because of suspend to wait mode. **Note: For MibSPI1 that supports 256 buffers (values from 0-FFh), bit 7 is used. For MibSPI2-5 that support 128 buffers (values from 0-7Fh), bit 7 is reserved.**	

NOTE: Register bits vary by device

TG0 has the highest priority and TG15 has the lowest priority. Under the following conditions a lower priority TG cannot be interrupted by a higher priority TG.

1. When there is a CSHOLD or LOCK buffer, until the completion of the next buffer transfer which is a non-CSHOLD or non-LOCK buffer.
2. An entire sequence of words transferred for a NOBRK DMA buffer.
3. Once the last word in a TG is pre-fetched.

28.3.35 DMA Channel Control Register (DMAxCTRL)

Each DMA channel can be configured via one dedicated control register. The register description below shows one exemplary control register that is identical for all DMA channels; for example, the control register for DMA channel 0 is named DMA0CTRL. The MibSPI supports up to 8 bidirectional DMA channels.

The number of bidirectional DMA channels varies by device. The number of DMA channels and hence the number of DMA channel control registers may vary.

Figure 28-71. DMA Channel Control Register (DMAxCTRL) [offset = D8h-F4h]

31	30 24	23 20	19 16
ONESHOT	BUFID	RXDMA_MAP	TXDMA_MAP
R/W-0	R/W-0	R/W-0	R/W-0

15	14	13	12 8
RXDMAENA	TXDMAENA	NOBRK	ICOUNT
R/W-0	R/W-0	R/W-0	R/W-0

7	6	5 0
Reserved	COUNT BIT17	COUNT
R-0	R-0	R-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 28-44. DMA Channel Control Register (DMAxCTRL) Field Descriptions

Bit	Field	Value	Description
31	ONESHOT		Auto-disable of DMA channel after ICOUNT+1 transfers. **Note: This ONESHOT applies to the DMA channel identified by x and will autodisable based on ICOUNTx.**
		0	The length of the block transfer is fully controlled by the DMA controller. The enable bits RXDMAENAx and TXDMAENAx are not modified by the MibSPI.
		1	ONESHOT allows a block transfer of defined length (ICOUNTx+1), mainly controlled by the MibSPI and not by the DMA controller. After ICOUNTx +1 transfers, the enable bits RXDMAENAx and TXDMAENAx are automatically cleared by the MibSPI, hence no more DMA requests are generated. In conjunction with NOBRKx, a burst transfer can be initiated without any other transfer through another buffer.
30-24	BUFIDx	0-7Fh	Buffer utilized for DMA transfer. BUFIDx defines the buffer that is utilized for the DMA transfer. In order to synchronize the transfer with the DMA controller with the NOBRK condition the "suspend to wait until..." modes must be used.
23-20	RXDMA_MAPx	0-Fh	Each MibSPI DMA channel can be linked to two physical DMA Request lines of the DMA controller. One request line for receive data and the other for request line for transmit data. RXDMA_MAPx defines the number of the physical DMA Request line that is connected to the receive path of the MibSPI DMA channel. If RXDMAENAx and TXDMAENAx are both set to 1, then RXDMA_MAPx shall differ from TXDMA_MAPx and shall differ from any other used physical DMA Request line. Otherwise unexpected interference may occur.
19-16	TXDMA_MAPx	0-Fh	Each MibSPI DMA channel can be linked to two physical DMA Request lines of the DMA controller. One request line for receive data and the other for request line for transmit data. TXDMA_MAPx defines the number of the physical DMA Request line that is connected to the transmit path of the MibSPI DMA channel. If RXDMAENAx and TXDMAENAx are both set then TXDMA_MAPx shall differ from RXDMA_MAPx and shall differ from any other used physical DMA Request line. Otherwise unexpected interference may occur.
15	RXDMAENAx		Receive data DMA channel enable.
		0	No DMA request upon new receive data.
		1	The physical DMA channel for the receive path is enabled. The first DMA request pulse is generated after the first transfer from the referenced buffer (BUFIDx) is finished. The buffer should be configured in as "skip until RXEMPTY is set" or "suspend to wait until RXEMPTY is set" in order to ensure synchronization between the DMA controller and the MibSPI sequencer.

Table 28-44. DMA Channel Control Register (DMAxCTRL) Field Descriptions (continued)

Bit	Field	Value	Description
14	TXDMAENAx		Transmit data DMA channel enable.
		0	No DMA request upon new transmit data.
		1	The physical DMA channel for the transmit path is enabled. The first DMA request pulse is generated right after setting TXDMAENAx to load the first transmit data. The buffer should be configured in the as "skip until TXFULL is set" or "suspend to wait until TXFULL is set" in order to ensure synchronization between the DMA controller and the MibSPI sequencer.
13	NOBRKx		Non-interleaved DMA block transfer. This bit is available in master mode only. **Note: Special Conditions during a NOBRK Buffer Transfer. If a NOBRK DMA buffer is currently being serviced by the sequencer, then it is not allowed to be disabled prematurely.** During a NOBRK transfer, the following operations are not allowed: • Clearing the NOBRKx bit to 0 • Clearing the RXDMAENAx to 0 (if it is already 1) • Clearing the TXDMAENAx to 0 (if it is already 1) • Clearing the BUFMODE[2:0] bits to 000 **Note: Any attempts to perform these actions during a NOBRK transfer will produce unpredictable results.**
		0	DMA transfers through the buffer referenced by BUFIDx are interleaved by data transfers from other active buffers or TGs. Every time the sequencer checks the DMA buffer, it performs one transfer and then steps to the next buffer.
		1	NOBRKx ensures that ICOUNTx + 1 data transfers are performed from the buffer referenced by BUFIDx without a data transfer from any other buffer. The sequencer remains at the DMA buffer until ICOUNTx + 1 transfers have been processed.For example, this can be used to generate a burst transfer to one device without disabling the chip select signal in-between (the concerned buffer has to be configured with CSHOLD = 1). Another example would be to have a defined block data transfer in slave mode, synchronous to the master SPI. **Note: Triggering of higher priority TGs or enabling of higher priority DMA channels will not interrupt a NOBRK block transfer.**
12-8	ICOUNTx	0-1Fh	Initial count of DMA transfers. ICOUNTx is used to preset the transfer counter COUNTx. Every time COUNTx hits 0, it is reloaded with ICOUNTx. The real number of transfers equals ICOUNTx plus 1. If ONESHOTx is set, ICOUNTx defines the number of DMA transfers that are performed before the MibSPI automatically disables the DMA channels. If NOBRKx is set, ICOUNTx defines the number of DMA transfers that are performed in one sequence without a transfer from any other buffer. If ONESHOTx and NOBRKx are not set, ICOUNTx should be 0. **Note: See Section 28.3.36 (ICOUNT) and Section 28.3.37 (DMACNTLEN) about how to increase the ICOUNT to a 16-bit value. With this extended capability, MibSPI can transfer a block of up to 65535 (65K) words without interleaving (if NOBRK is used) or without deasserting the chip select between the buffers (if CSHOLD is used).**
7	Reserved	0	Reads return 0. Writes have no effect.
6	COUNT BIT17x		The 17th bit of the COUNT field of DMAxCOUNT register.
5-0	COUNTx	0-3Fh	Actual number of remaining DMA transfers. This field contains the actual number of DMA transfers that remain, until the DMA channel is disabled, if ONESHOTx is set. **Note: If the TX and RX DMA requests are enabled, the COUNT register will be decremented when the RX has been serviced.**

28.3.36 DMAxCOUNT Register (ICOUNT)

NOTE: These registers are used only if the LARGE COUNT bit in the DMACNTLEN register is set.

The number of bidirectional DMA channels varies by device. The number of DMA channels and hence the number of DMA registers varies by device.

Figure 28-72. DMAxCOUNT Register (ICOUNT) [offset = F8h-114h]

31	16
ICOUNTx	
R/W-0	

15	0
COUNTx	
R-0	

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 28-45. MibSPI DMAxCOUNT Register (ICOUNT) Field Descriptions

Bit	Field	Value	Description
31-16	ICOUNTx	0-FFFFh	Initial number of DMA transfers. ICOUNTx is used to preset the transfer counter COUNTx. Every time COUNTx hits 0, it is reloaded with ICOUNTx. The real number of transfer equals ICOUNTx plus 1. If ONESHOTx is set, ICOUNTx defines the number of DMA transfers that are performed before the MibSPI automatically disables the corresponding DMA channel. If NOBRKx is set, ICOUNTx defines the number of DMA transfers that are performed in one sequence without a transfer from any other buffer
15-0	COUNTx	0-FFFFh	Actual number of remaining DMA transfers. COUNTx Contains the actual number of DMA transfers that remain, until the DMA channel is disabled, if ONESHOTx is set. Since the real counter value is always ICOUNTx + 1, the 17th bit of COUNTx is available on DMACTRLx[6] bit. **Note: Usage Tip for Block Transfer Using a Single DMA Request. It is possible to use the multi-buffer RAM to transfer chunks of data to/from an external SPI. A DMA Controller can be used to handle the data in bursts. Suppose a chunk of 64 bytes of data needs to be transferred and a single DMA request needs to be generated at the end of transferring the 64 bytes. This can be easily achieved by configuring a TG register for the 64 buffer locations and using the DMAxCTRL/DMAxCOUNT registers to configure the last buffer (64th) of the TG as the BUFID and enable RXDMA (NOBRK = 0). At the end of the transfer of the 64th buffer, a DMA request will be generated on the selected DMA request channel. The DMA controller can do a burst read of all 64 bytes from RXRAM and/or then do a burst write to all 64 bytes to the TXRAM for the next chunk.**

28.3.37 DMA Large Count (DMACNTLEN)

Figure 28-73. DMA Large Count Register (DMACNTLEN) [offset = 118h]

31 – 16
Reserved
R-0

15 – 1	0
Reserved	LARGE COUNT
R-0	R/WP-0

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privilege mode only; -*n* = value after reset

Table 28-46. MibSPI DMA Large Count Register (DMACNTLEN) Field Descriptions

Bit	Field	Value	Description
31-1	Reserved	0	Reads return 0. Writes have no effect.
0	LARGE COUNT		Select either the 16-bit DMAxCOUNT counters or the smaller counters in DMAxCTRL.
		0	Select the DMAxCTRL counters. Writes to the DMAxCTRL register will modify the ICOUNT value. Reading ICOUNT and COUNT can be done from the DMAxCTRL register. The DMAxCOUNT register should not be used since any write to this register will be overwritten by a subsequent write to the DMAxCTRL register to set the TXDMAENA or RXDMAENA bits.
		1	Select the DMAxCOUNT counters. Writes to the DMAxCTRL register will not modify the ICOUNT value. The ICOUNT value must be written to in the DMAxCOUNT register before the RXDMAENA or TXDMAENA bits are set in the DMAxCTRL register. The DMAxCOUNT register should be used for reading COUNT or ICOUNT.

28.3.38 Parity/ECC Control Register (PAR_ECC_CTRL)

Figure 28-74. Parity/ECC Control Register (PAR_ECC_CTRL) [offset = 120]

31-28	27-24	23-20	19-16
Reserved	SBE_EVT_EN	Reserved	EDAC_MODE
R-0	R/W-5h	R-0	R/WP-Ah

15-9	8	7-4	3-0
Reserved	PTESTEN	Reserved	EDEN
R-0	R/WP-0	R-0	R/W-5h

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privilege mode only; -*n* = value after reset

Table 28-47. MibSPI Parity/ECC Control Register (PAR_ECC_CTRL) Field Descriptions

Bit	Field	Value	Description
31-28	Reserved	0	Reads return 0. Writes have no effect.
27-24	SBE_EVT_EN		Single-Bit Error Event Enable This bit controls the generation of error signaling (on MIBSPI_SBERR port) whenever a Single-Bit Error (SBE) is detected on TXRAM/RXRAM. This signal can be used to generate interrupt if required.
		5h	Write: Disable Error Event indication upon detection of SBE on TXRAM/RXRAM.
		Ah	Write: Enable Error Event upon detection of SBE on TXRAM/RXRAM.
			All other values - writes are ignored and the values are not updated into this field. The state of the feature remains unchanged.
			Read: Returns the current value of the field.
23-20	Reserved	0	Reads return 0. Writes have no effect.
19-16	EDAC_MODE		These bits determine whether Single-Bit Errors (SBE) detected by the SECDED block will be corrected or not.
		5h	Write: Disable correction of SBE detected by the SECDED block.
		Ah	Write: Enable correction of SBE detected by the SECDED block.
			All other values - writes are ignored and the values are not updated into this field. The state of the feature remains unchanged.
			Read: Returns the current value of the field.
15-9	Reserved	0	Reads return 0. Writes have no effect.
8	PTESTEN		Parity/ECC memory test enable. This bit, maps the parity/ECC bits corresponding to multi-buffer RAM locations into the peripheral RAM frame to make them accessible by the CPU.
			User and privilege mode (read):
		0	Parity/ECC bits are not memory-mapped.
		1	Parity/ECC bits are memory-mapped.
			Privilege mode (write):
		0	Disable memory-mapping of Parity/ECC locations.
		1	Enable memory-mapping of Parity/ECC locations.
7-4	Reserved	0	Reads return 0. Writes have no effect.
3-0	EDEN		Error Detection Enable These bits enable Parity/ECC error detection.
		5h	Write: Disable Parity/ECC error detection logic (default).
		All other values	Write: Enable Parity/ECC error detection logic.
			Read: Returns the current value of this field.

28.3.39 Parity/ECC Status Register (PAR_ECC_STAT)

Figure 28-75. Parity/ECC Status Register (PAR_ECC_STAT) [offset = 124]

31	16
Reserved	
R-0	

15	10	9	8
Reserved		SBE_FLG1	SBE_FLG0
R-0		R/W1C-0	R/W1C-0

7	2	1	0
Reserved		UERR_ FLG1	UERR_ FLG0
R-0		R/W1C-0	R/W1C-0

LEGEND: R/W = Read/Write; R = Read only; W1C = Write 1 to clear; -*n* = value after reset

Table 28-48. Parity/ECC Status Register (PAR_ECC_STAT) Field Descriptions

Bit	Field	Value	Description
31-10	Reserved	0	Reads return 0. Writes have no effect.
9	SBE_FLG1	0	Single-Bit Error in RXRAM. This flag indicates if a single-bit ECC error occurred on reading RXRAM. Read: No error occurred. Write: No effect.
		1	Read: Single-bit error is detected in RXRAM and the address is captured in SBERRADDR1 register. Write: Clears the bit.
8	SBE_FLG0	0	Single-Bit Error in TXRAM. This flag indicates if a single-bit ECC error occurred on reading TXRAM.
			Read: No error occurred. Write: No effect.
		1	Read: Single-bit error is detected in TXRAM and the address is captured in SBERRADDR0 register Write: Clears the bit .
7-2	Reserved	0	Reads return 0. Writes have no effect.
1	UERR_FLG1	0	Uncorrectable Parity or double-bit ECC error detection flag. This flag indicates if a Parity or double-bit ECC error occurred on reading RXRAM Read: No error occurred. Write: No effect.
		1	Read: Error detected and the address is captured in UERRADDR1 register. Write: Clears the bit.
0	UERR_FLG0	0	Uncorrectable Parity or double-bit ECC error detection flag. This flag indicates if a Parity or double-bit ECC error occurred on reading TXRAM Read: No error occurred. Write: No effect.
		1	Read: Error detected and the address is captured in UERRADDR0 register. Write: Clears the bit.

28.3.40 Uncorrectable Parity or Double-Bit ECC Error Address Register - RXRAM (UERRADDR1)

Figure 28-76. Uncorrectable Parity or Double-Bit ECC Error Address Register - RXRAM (UERRADDR1) [offset = 128h]

31 … 16
Reserved
R-0

15 … 11	10 … 0
Reserved	UERRADDR1
R-0	RC-x

LEGEND: R/W = Read/Write; R = Read only; RC = Read to clear; -*n* = value after reset

Table 28-49. Uncorrectable Parity or Double-Bit ECC Error Address Register - RXRAM (UERRADDR1) Field Descriptions

Bit	Field	Value	Description
31-11	Reserved	0	Reads return 0. Writes have no effect.
10-0	UERRADDR1		Uncorrectable Parity or double-bit ECC error address This register holds the address of the RAM location, if a parity or double-bit ECC error is detected when reading the MibSPI (Receive) RXRAM. The address captured is byte aligned when RAM Parity Check is supported. This error address is frozen from being updated until it is read by the VBUS host. Reading this register clears its contents to the default value. The default value is 400h if Extended Buffer feature is enabled; else, it is 200h. Writes to this register are ignored.

NOTE: UERRADDR1 values

The offset address of RXRAM can vary from 000h-1FFh, if EXTENDED_BUF mode is disabled. If the EXTENDED_BUF mode is enabled, the offset address can vary from 000h-3FFh.

The register does not clear its contents during and after any of the module-level resets, System-level resets, or even Power-on Reset.

NOTE: A read to UERRADDR1 register will clear the UERR_FLG1 in PAR_ECC_STAT register. However, in emulation mode (VBUSP_EMUDBG = 1), the read to UERRADDR1 register does not clear the corresponding UERR_FLG1.

After a power-on reset the contents of this register will be unpredictable. So, a read operation can be performed after power-up to clear its contents if required. Contents of this register are meaningful only when UERR_FLG1 is set to 1.

If ECC feature is implemented, the Sequencer FSM clearing the TXFULL flag (after a TXRAM location read out and written to the shift register for transfer) will trigger read-modify-write operation to the RXRAM. Similarly, each time FSM reads a TXRAM to transfer it out, the corresponding RXRAM location is also automatically read to determine the status of the buffer. A double-bit error could be detected during these FSM read operations and result in error address and flags getting captured.

NOTE: Clearing of UERR status and address registers

After completing a memory test sequence, specifically where parity or ECC features are tested, user must read back the status flags in PAR_ECC_STAT and UERRADDRx registers and ensure that they are in normal clear state by reading/writing appropriately. This can be performed before the start of a normal multi-buffer mode transactions as well.

If RAM Parity Check is supported, UERRADDR1[1:0] values will reflect the byte positions of failed byte based on the following scheme to take care of Endianness of memory organization.

Table 28-50. Effect of BIG_ENDIAN Port on UERRADDR1[1:0] Bits

	Endianness		Fault Location is Among the RAM Bits
	1 (Big Endian)	0 (Little Endian)	
UERRADDR1[1:0]	00	11	7:0
	01	10	15:8
	10	01	23:16
	11	00	31:24

NOTE: When ECC is supported, UERRADDR0 will indicate only word address. UERRADDR0[1:0] will always be 00.

28.3.41 Uncorrectable Parity or Double-Bit ECC Error Address Register - TXRAM (UERRADDR0)

Figure 28-77. Uncorrectable Parity or Double-Bit ECC Error Address Register - TXRAM (UERRADDR0) [offset = 12Ch]

31 … 16
Reserved
R-0

15 … 11	10 … 0
Reserved	UERRADDR0
R-0	RC-x

LEGEND: R/W = Read/Write; R = Read only; RC = Read to clear; -*n* = value after reset

Table 28-51. Uncorrectable Parity or Double-Bit ECC Error Address Register - TXRAM (UERRADDR0) Field Descriptions

Bit	Field	Value	Description
31-11	Reserved	0	Reads return 0. Writes have no effect.
10-0	UERRADDR0		Uncorrectable Parity or double-bit ECC error address. This register holds the address when a parity error is generated while reading the MibSPI (Transmit) TXRAM. The TXRAM can be read either by CPU or by the MibSPI Sequencer FSM logic for transmission. The address captured is byte aligned. This error address is frozen from being updated until it is read by the VBUSP host. Reading this register clears its contents to the default value of 000. Writes to this register are ignored.

NOTE: UERRADDR0 values

The offset address of TXRAM can vary from 200h-3FFh, if EXTENDED_BUF mode is disabled. If the EXTENDED_BUF mode is enabled, the offset address can vary from 400h-7FFh.

The register does not clear its contents during and after any of the module-level resets, System-level resets, or even Power-on Reset.

NOTE: A Read to UERRADDR0 register will clear the UERR_FLG0 in PAR_ECC_STAT register. However, in emulation mode (VBUSP_EMUDBG = 1), the read to UERRADDR0 register does not clear the corresponding UERR_FLG0.

After a power-on reset the contents will be unpredictable. A read operation can be performed after power-up to keep the register at its default value if required. Contents of this register are meaningful only when UERR_FLG0 is set to 1.

If ECC feature is implemented, the Sequencer FSM clearing the TXFULL flag (after a TXRAM location read out and written to the shift register for transfer) will trigger read-modify-write operation to the RXRAM. Similarly, each time FSM reads a TXRAM to transfer it out, the corresponding RXRAM location is also automatically read to determine the status of the buffer. A double-bit error could be detected during these FSM read operations and result in error address and flags getting captured.

NOTE: Clearing of UERR status and address registers

After completing a memory test sequence, specifically where parity or ECC features are tested, user must read back the status flags in PAR_ECC_STAT and UERRADDRx registers and ensure that they are in normal clear state by reading/writing appropriately. This can be performed before the start of a normal multi-buffer mode transactions as well.

If RAM Parity Check is supported, UERRADDR0[1:0] values will reflect the byte positions of failed byte based on the following scheme to take care of Endianness of memory organization.

Table 28-52. Effect of BIG_ENDIAN Port on UERRADDR0[1:0] Bits

	Endianness		Fault Location is Among the RAM Bits
	1 (Big Endian)	0 (Little Endian)	
UERRADDR0[1:0]	00	11	7:0
	01	10	15:8
	10	01	23:16
	11	00	31:24

NOTE: When ECC is supported, UERRADDR0 will indicate only word address. UERRADDR0[1:0] will always be 00.

28.3.42 RXRAM Overrun Buffer Address Register (RXOVRN_BUF_ADDR)

In multi-buffer mode, if a particular RXRAM location is written by the MibSPI sequencer logic after the completion of a new transfer when that location already contains valid data, the RX_OVR bit will be set to 1 while the data is being written. The RXOVRN_BUF_ADDR register captures the address of the RXRAM location for which a receiver overrun condition occurred.

Figure 28-78. RXRAM Overrun Buffer Address Register (RXOVRN_BUF_ADDR) [offset = 130h]

31	16
Reserved	
R-0	

15	10	9	0
Reserved		RXOVRN_BUF_ADDR	
R-0		R-200h	

LEGEND: R = Read only; -*n* = value after reset

Table 28-53. RXRAM Overrun Buffer Address Register (RXOVRN_BUF_ADDR) Field Descriptions

Bit	Field	Value	Description
31-10	Reserved	0	Reads return 0. Writes have no effect.
9-0	RXOVRN_BUF_ADDR	200h-3FCh	Address in RXRAM at which an overwrite occurred. This address value will show only the offset address of the RAM location in the multi-buffer RAM address space. Refer to the device-specific data sheet for the actual absolute address of RXRAM. This word-aligned address can vary from 200h-3FCh. Contents of this register are valid only when any of the INTVECT0 or INTVECT1 and SPIFLG registers show an RXOVRN error vector while in multi-buffer mode. If there are multiple overrun errors, then this register holds the address of first overrun address until it is read. **Note: Reading this register clears the RXOVRN interrupt flag in the SPIFLG register and the TGINTVECTx.** **Note: Receiver overrun errors in multi-buffer mode can be completely avoided by using the SUSPEND until RXEMPTY feature, which can be programmed into each buffer of any TG. However, using the SUSPEND until RXEMPTY feature will make the sequencer wait until the current RXRAM location is read by the VBUS master before it can start the transfer for the same buffer location again. This may affect the overall throughput of the SPI transfer. By enabling the interrupt on RXOVRN in multi-buffer mode, the user can rely on interrupts to know if a receiver overrun has occurred. The address of the overrun in RXRAM is indicated in this RXOVRN_BUF_ADDR register**.

28.3.43 I/O-Loopback Test Control Register (IOLPBKTSTCR)

This register controls test mode for I/O pins. It also controls whether loop-back should be digital or analog. In addition, it contains control bits to induce error conditions into the module. These are to be used only for module testing.

All of the control/status bits in this register are valid only when the IOLPBKTSTENA field is set to Ah.

Figure 28-79. I/O-Loopback Test Control Register (IOLPBKTSTCR) [offset = 134h]

31	25	24
Reserved		SCS FAIL FLG
R-0		R/W1C-0

23	21	20	19	18	17	16
Reserved		CTRL BITERR	CTRL DESYNC	CTRL PARERR	CTRL TIMEOUT	CTRL DLENERR
R-0		R/WP-0	R/WP-0	R/WP-0	R/WP-0	R/WP-0

15	12	11	8
Reserved		IOLPBKTSTENA	
R-0		R/WP-0	

7	6	5	3	2	1	0
Reserved		ERR SCS PIN		CTRL SCS PINERR	LPBKTYPE	RXPENA
R-0		R/WP-0		R/WP-0	R/WP-0	R/WP-0

LEGEND: R/W = Read/Write; R = Read only; W1C = Write 1 to clear; WP = Write in privilege mode only; -*n* = value after reset

Table 28-54. I/O-Loopback Test Control Register (IOLPBKTSTCR) Field Descriptions

Bit	Field	Value	Description
31-25	Reserved	0	Reads return 0. Writes have no effect.
24	SCS FAIL FLG		Bit indicating a failure on $\overline{\text{SPICS}}$ pin compare during analog loopback.
		0	Read: No miscompares occurred on any of the eight chip select pins (vs. the internal chip select number CSNR during transfers). Write: No effect.
		1	Read: A comparison between the internal CSNR field and the analog looped-back value of one or more of the $\overline{\text{SPICS}}$ pins failed. A stuck-at fault is detected on one of the $\overline{\text{SPICS}}$ pins. Comparison is done only on the pins that are configured as functional and during transfer operation. Write: This flag bit is cleared.
23-21	Reserved	0	Reads return 0. Writes have no effect.
20	CTRL BITERR		Controls inducing of BITERR during I/O loopback test mode.
		0	Do not interfere with looped-back data.
		1	Induces bit errors by inverting the value of the incoming data during loopback.
19	CTRL DESYNC		Controls inducing of the desync error during I/O loopback test mode.
		0	Do not cause a desync error.
		1	Induce a desync error by forcing the incoming $\overline{\text{SPIENA}}$ pin (if functional) to remain 0 even after the transfer is complete. This forcing will be retained until the kernel reaches the idle state.
18	CTRL PARERR		Controls inducing of the parity errors during I/O loopback test mode.
		0	Do not cause a parity error.
		1	Induce a parity error by inverting the polarity of the parity bit.
17	CTRL TIMEOUT		Controls inducing of the timeout error during I/O loopback test mode.
		0	Do not cause a timeout error.
		1	Induce a timeout error by forcing the incoming $\overline{\text{SPIENA}}$ pin (if functional) to remain 1 when transmission is initiated. The forcing will be retained until the kernel reaches the idle state.

Table 28-54. I/O-Loopback Test Control Register (IOLPBKTSTCR) Field Descriptions (continued)

Bit	Field	Value	Description
16	CTRL DLENERR		Controls inducing of the data length error during I/O loopback test mode.
		0	Do not cause a data-length error.
		1	Induce a data-length error. *Master mode:* The SPIENA pin (if functional) is forced to 1 when the module starts shifting data. *Slave mode:* The incoming SPICS pin (if functional) is forced to 1 when the module starts shifting data.
15-12	Reserved	0	Reads return 0. Writes have no effect.
11-8	IOLPBKSTENA		Module I/O loopback test enable key.
		Ah	Enable I/O loopback test mode.
		All Other Values	Disable I/O loopback test mode.
7-6	Reserved	0	Reads return 0. Writes have no effect.
5-3	ERR SCS PIN		Inject error on chip-select pin number x. The value in this field is decoded as the number of the chip select pin on which to inject an error. During analog loopback, if CTRL SCS PIN ERR bit is set to 1, then the chip select pin selected by this field is forced to the opposite of its value in the CSNR.
		0	Select SPICS[0] for injecting error.
		1h	Select SPICS[1] for injecting error.
		:	:
		7h	Select SPICS[7] for injecting error.
2	CTRL SCS PINERR		Enable the injection of an error on the SPICS pins. The individual SPICS pins can be chosen using the ERR SCS PIN field.
		0	Disable the SPICS error-inducing logic.
		1	Enable the SPICS error-inducing logic.
1	LPBK TYPE		Module I/O loopback type (analog/digital). See Figure 28-31 for the different types of loopback modes.
		0	Enable Digital loopback when IOLPBKTSTENA = 1010.
		1	Enable Analog loopback when IOLPBKTSTENA = 1010.
0	RXP ENA		Enable analog loopback through the receive pin. **Note: This bit is valid only when LPBK TYPE = 1, which chooses analog loopback mode.**
		0	Analog loopback is through the transmit pin.
		1	Analog loopback is through the receive pin.

28.3.44 SPI Extended Prescale Register 1 (EXTENDED_PRESCALE1 for SPIFMT0 and SPIFMT1)

This register provides an extended Prescale values for SPICLK generation to be able to interface with much slower SPI Slaves. This is an extension of SPIFMT0 and SPIFMT1 registers. For example, EPRESCALE_FMT1(7:0) of EXTENDED_PRESCALE1 and PRESCALE1(7:0) of SPIFMT1 register will always reflect the same contents. Similarly EPRESCALE_FMT0(7:0) and PRESCALE0(7:0) of SPIFMT0 reflect the same contents.

Figure 28-80. SPI Extended Prescale Register 1 (EXTENDED_PRESCALE1 for SPIFMT0 and SPIFMT1) [offset = 138h]

31 - 27	26 - 16
Reserved	EPRESCALE_FMT1
R-0	R/WP-0

15 - 11	10 - 0
Reserved	EPRESCALE_FMT0
R-0	R/WP-0

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privilege mode only; -*n* = value after reset

Table 28-55. SPI Extended Prescale Register 1 (EXTENDED_PRESCALE1) Field Descriptions

Bit	Field	Value	Description
31-27	Reserved	0	Reads return 0. Writes have no effect.
26-16	EPRESCALE_FMT1	0-7FFh	EPRESCALE_FMT1. Extended Prescale value for SPIFMT1. EPRESCALE_FMT1 determines the bit transfer rate of data format 1 if the SPI/MibSPI is the network master. EPRESCALE_FMT1 is use to derive SPICLK from VCLK. If the SPI is configured as slave, EPRESCALE_FMT1 **does not need** to be configured. These EPRESCALE_FMT1(7:0) bits and PRESCALE1 bits of SPIFMT1 register will point to the same physically implemented register. The clock rate for data format 1 can be calculated as: $BR_{Format1}$ = VCLK / (EPRESCALE_FMT1 + 1) Write: This register field should be written if a SPICLK prescaler of more VCLK/256 is required. This field provides a prescaler of up to VCLK/2048 for SPICLK. Writing to this register field will also get reflected in SPIFMT1(15:8). Read: Reading this field will reflect the PRESCALE value based on the last written register field, that is, EXTENDED_PRESCALE1(26:16) or SPIFMT1(15:8) register. **Note: If Extended Prescaler is required, it should be ensured that EXTENDED_PRESCALE1 register is programmed after SPIFMT1 register is programmed. This is to ensure that the final SPICLK prescale value is controlled by EXTENDED_PRESCALE1 register when a prescale of more 256 is intended on SPICLK. Writing to PRESCALE1 field of SPIFMT1 will automatically clear EPRESCALE_FMT1(10:8) bits to 000 so that the integrity of PRESCALE value is maintained.**
15-11	Reserved	0	Reads return 0. Writes have no effect.

Table 28-55. SPI Extended Prescale Register 1 (EXTENDED_PRESCALE1) Field Descriptions (continued)

Bit	Field	Value	Description
10-0	EPRESCALE_FMT0	0-7FFh	EPRESCALE_FMT0. Extended Prescale value for SPIFMT0. EPRESCALE_FMT0 determines the bit transfer rate of data format 0 if the SPI/MibSPI is the network master. EPRESCALE_FMT0 is use to derive SPICLK from VCLK. If the SPI is configured as slave, EPRESCALE_FMT0 **does not need** to be configured. These EPRESCALE_FMT0(7:0) bits and PRESCALE0 bits of SPIFMT0 register will point to the same physically implemented register. The clock rate for data format 0 can be calculated as: $BR_{Format0}$ = VCLK / (EPRESCALE_FMT0 + 1) Write: This register field should be written if a SPICLK prescaler of more VCLK/256 is required. This field provides a prescaler of up to VCLK/2048 for SPICLK. Writing to this register field will also get reflected in SPIFMT0(15:8). Read: Reading this field will reflect the PRESCALE value based on the last written register field, that is, EXTENDED_PRESCALE0(10:0) or SPIFMT0(15:8) register. **Note: If Extended Prescaler is required, it should be ensured that EXTENDED_PRESCALE1 register is programmed after SPIFMT0 register is programmed. This is to ensure that the final SPICLK prescale value is controlled by EXTENDED_PRESCALE1 register when a prescale of more 256 is intended on SPICLK. Writing to PRESCALE0 field of SPIFMT0 will automatically clear EPRESCALE_FMT0(10:8) bits to 000 so that the integrity of PRESCALE value is maintained.**

28.3.45 SPI Extended Prescale Register 2 (EXTENDED_PRESCALE2 for SPIFMT2 and SPIFMT3)

This register provides an extended Prescale values for SPICLK generation to be able to interface with much slower SPI Slaves. This is an extension of SPIFMT2 and SPIFMT3 registers. For example, EPRESCALE_FMT3(7:0) of EXTENDED_PRESCALE2 and PRESCALE3(7:0) of SPIFMT3 register will always reflect the same contents. Similarly EPRESCALE_FMT2(7:0) and PRESCALE2(7:0) of SPIFMT2 reflect the same contents.

Figure 28-81. SPI Extended Prescale Register 2 (EXTENDED_PRESCALE2 for SPIFMT2 and SPIFMT3) [offset = 13Ch]

31 27	26 16
Reserved	EPRESCALE_FMT3
R-0	R/WP-0

15 11	10 0
Reserved	EPRESCALE_FMT2
R-0	R/WP-0

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privilege mode only; -*n* = value after reset

Table 28-56. SPI Extended Prescale Register 2 (EXTENDED_PRESCALE2) Field Descriptions

Bit	Field	Value	Description
31-27	Reserved	0	Reads return 0. Writes have no effect.
26-16	EPRESCALE_FMT3	0-7FFh	EPRESCALE_FMT3. Extended Prescale value for SPIFMT3. EPRESCALE_FMT3 determines the bit transfer rate of data format 3, if the SPI/MibSPI is the network master. EPRESCALE_FMT3 is use to derive SPICLK from VCLK. If the SPI is configured as slave, EPRESCALE_FMT3 **does not need** to be configured. These EPRESCALE_FMT3(7:0) bits and PRESCALE3 bits of SPIFMT3 register will point to the same physically implemented register. The clock rate for data format 3 can be calculated as: $BR_{Format3}$ = VCLK / (EPRESCALE_FMT3 + 1) Write: This register field should be written if a SPICLK prescaler of more VCLK/256 is required. This field provides a prescaler of up to VCLK/2048 for SPICLK. Writing to this register field will also get reflected in SPIFMT3(15:8). Read: Reading this field will reflect the PRESCALE value based on the last written register field, that is, EXTENDED_PRESCALE3(26:16) or SPIFMT3(15:8) register. **Note: If Extended Prescaler is required, it should be ensured that EXTENDED_PRESCALE2 register is programmed after SPIFMT3 register is programmed. This is to ensure that the final SPICLK prescale value is controlled by EXTENDED_PRESCALE2 register when a prescale of more 256 is intended on SPICLK. Writing to PRESCALE3 field of SPIFMT3 will automatically clear EPRESCALE_FMT3(10:8) bits to 000 so that the integrity of PRESCALE value is maintained.**
15-11	Reserved	0	Reads return 0. Writes have no effect.

Table 28-56. SPI Extended Prescale Register 2 (EXTENDED_PRESCALE2) Field Descriptions (continued)

Bit	Field	Value	Description
10-0	EPRESCALE_FMT2	0-7FFh	EPRESCALE_FMT2. Extended Prescale value for SPIFMT2. EPRESCALE_FMT2 determines the bit transfer rate of data format 2, if the SPI/MibSPI is the network master. EPRESCALE_FMT2 is use to derive SPICLK from VCLK. If the SPI is configured as slave, EPRESCALE_FMT2 **does not need** to be configured. These EPRESCALE_FMT2(7:0) bits and PRESCALE2 bits of SPIFMT2 register will point to the same physically implemented register. The clock rate for data format 2 can be calculated as: $BR_{Format2}$ = VCLK / (EPRESCALE_FMT2 + 1) Write: This register field should be written if a SPICLK prescaler of more VCLK/256 is required. This field provides a prescaler of up to VCLK/2048 for SPICLK. Writing to this register field will also get reflected in SPIFMT2(15:8). Read: Reading this field will reflect the PRESCALE value based on the last written register field, that is, EXTENDED_PRESCALE2(10:0) or SPIFMT2(15:8) register. **Note: If Extended Prescaler is required, it should be ensured that EXTENDED_PRESCALE2 register is programmed after SPIFMT2 register is programmed. This is to ensure that the final SPICLK prescale value is controlled by EXTENDED_PRESCALE2 register when a prescale of more 256 is intended on SPICLK. Writing to PRESCALE2 field of SPIFMT2 will automatically clear EPRESCALE_FMT2(10:8) bits to 000 so that the integrity of PRESCALE value is maintained.**

28.3.46 ECC Diagnostic Control Register (ECCDIAG_CTRL)

Figure 28-82. ECC Diagnostic Control Register (ECCDIAG_CTRL) [offset = 140h]

<table>
<tr><td>31</td><td>16</td></tr>
<tr><td colspan="2">Reserved</td></tr>
<tr><td colspan="2">R-0</td></tr>
</table>

<table>
<tr><td>15</td><td>4</td><td>3</td><td>0</td></tr>
<tr><td colspan="2">Reserved</td><td colspan="2">ECCDIAG_EN</td></tr>
<tr><td colspan="2">R-0</td><td colspan="2">R/WP-Ah</td></tr>
</table>

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privilege mode only; -*n* = value after reset

Table 28-57. ECC Diagnostic Control Register (ECCDIAG_CTRL) Field Descriptions

Bit	Field	Value	Description
31-4	Reserved		Reads return 0. Writes have no effect.
3-0	ECCDIAG_EN		ECC Diagnostic mode Enable Key bits.
		5h	Write: Diagnostic mode is enabled. Writes and reads from ECC bits allowed from the ECC address space.
		All other values	Write: Diagnostic mode is disabled. No writes to ECC bits are ignored.
			Read: Returns 0.

28.3.47 ECC Diagnostic Status Register (ECCDIAG_STAT)

NOTE: ECCDIAG_STAT Validity

Both SEFLG and DEFLG are valid only during Diagnostic Mode (when ECCDIAG_EN = 5h). This status register should be write-cleared after coming out of Diagnostic Mode.

Figure 28-83. ECC Diagnostic Status Register (ECCDIAG_STAT) [offset = 144h]

31	18	17	16
Reserved		DEFLG	
R-0		R/W1C-0	

15	2	1	0
Reserved		SEFLG	
R-0		R/W1C-0	

LEGEND: R/W = Read/Write; R = Read only; W1C = Write 1 to clear; -*n* = value after reset

Table 28-58. ECC Diagnostic Status Register (ECCDIAG_STAT) Field Descriptions

Bit	Field	Value	Description
31-18	Reserved	0	Reads return 0. Writes have no effect.
17	DEFLG[1]	0	Double-bit error flag. Read: No error. Write: No effect.
		1	Read: A double-bit error is detected for RXRAM bank during diagnostic mode tests. Write: Clears the bit.
16	DEFLG[0]	0	Double-bit error flag. Read: No error. Write: No effect.
		1	Read: A double-bit error is detected for TXRAM bank during diagnostic mode tests. Write: Clears the bit.
15-2	Reserved	0	Reads return 0. Writes have no effect.
1	SEFLG[1]	0	Single-bit error flag. Read: No error. Write: No effect.
		1	Read: A single-bit error is detected for RXRAM bank during diagnostic mode tests. Write: Clears the bit.
0	SEFLG[0]	0	Single-bit error flag. Read: No error. Write: No effect.
		1 1	Read: A single-bit error is detected for TXRAM bank during diagnostic mode tests. Write: Clears the bit.

28.3.48 Single-Bit Error Address Register - RXRAM (SBERRADDR1)

Figure 28-84. Single-Bit Error Address Register - RXRAM (SBERRADDR1) [offset = 148h]

31 – 16
Reserved
R-0

15 – 11	10 – 0
Reserved	SBERRADDR1
R-0	RC-0

LEGEND: R = Read only; RC = Read to clear; -*n* = value after reset

Table 28-59. Single-Bit Error Address Register - RXRAM (SBERRADDR1) Field Descriptions

Bit	Field	Value	Description
31-11	Reserved	0	Reads return 0. Writes have no effect.
10-0	SBERRADDR1		This register holds the address of the RAM location when a single-bit error is generated by SECDED block while reading the MibSPI (Receive) RXRAM. This error address is frozen from being updated until it is read by the VBUS host. Reading this register clears its contents to the default value. The default value is 400h if Extended Buffer feature is enabled; else, it is 200h. Writes to this register are ignored.

NOTE: SBERRADDR1 values

The offset address of RXRAM can vary from 200h-3FFh, if EXTENDED_BUF mode is disabled. If the EXTENDED_BUF mode is enabled, the offset address can vary from 400h-7FFh.

The register does not clear its contents during and after any of the module-level resets, System-level resets, or even Power-on Reset.

NOTE: A Read to SBERRADDR1 Register will clear the SBE_FLG1 in PAR_ECC_STAT register. However, in emulation mode (VBUSP_EMUDBG = 1), the read to SBERRADDR1 register does not clear the corresponding SBE_FLG1.

After a power-on reset the contents will be unpredictable. A read operation can be performed after power-up to keep the register at its default value if required. Contents of this register are meaningful only when SBE_FLG1 is set to 1.

If ECC feature is implemented, the Sequencer FSM clearing the TXFULL flag (after a TXRAM location read out and written to the shift register for transfer) will trigger read-modify-write operation to the RXRAM. Similarly, each time FSM reads a TXRAM to transfer it out, the corresponding RXRAM location is also automatically read to determine the status of the buffer. A single-bit error could be detected during these FSM read operations and result in error address and flags getting captured.

NOTE: Clearing of SBERR status and address registers

After completing a memory test sequence, specifically where ECC features are tested, user must read back the status flags in ECC_STAT and SBERRADDRx registers and ensure that they are in normal clear state by reading/writing appropriately. This can be performed before the start of a normal multi-buffer mode transactions as well.

NOTE: When ECC is supported, SBERRADDR1 will indicate only word address. SBERRADDR1[1:0] will always be 00.

28.3.49 Single-Bit Error Address Register - TXRAM (SBERRADDR0)

Figure 28-85. Single-Bit Error Address Register - TXRAM (SBERRADDR0) [offset = 14Ch]

31 – 16
Reserved
R-0

15 – 11	10 – 0
Reserved	SBERRADDR0
R-0	RC-0

LEGEND: R = Read only; RC = Read to clear; -*n* = value after reset

Table 28-60. Single-Bit Error Address Register - TXRAM (SBERRADDR0) Field Descriptions

Bit	Field	Value	Description
31-11	Reserved	0	Reads return 0. Writes have no effect.
10-0	SBERRADDR0		This register holds the address when a single-bit error is generated from SECDED block while reading the MibSPI (Transmit) TXRAM. The TXRAM can be read either by CPU or by the MibSPI Sequencer logic for transmission. This error address is frozen from being updated until it is read by the VBUSP host. Reading this register clears its contents to the default value of 0x000. Writes to this register are ignored.

NOTE: SBERRADDR0 values

The offset address of TXRAM can vary from 000h-1FFh, if EXTENDED_BUF mode is disabled. If the EXTENDED_BUF mode is enabled, the offset address can vary from 000h-3FFh.

The register does not clear its contents during and after any of the module-level resets, System-level resets, or even Power-on Reset. A Read operation to this register clears its contents to all 0s.

NOTE: A read to SBERRADDR0 register will clear the SBE_FLG0 in PAR_ECC_STAT register. However, in emulation mode (VBUSP_EMUDBG = 1), the read to SBERRADDR0 register does not clear the corresponding SBE_FLG0.

After a power-on reset the contents of this register will be unpredictable. So, a read operation can be performed after power-up to clear its contents if required. Contents of this register are meaningful only when SBE_FLG0 is set to 1.

NOTE: Clearing of SBERR status and address registers

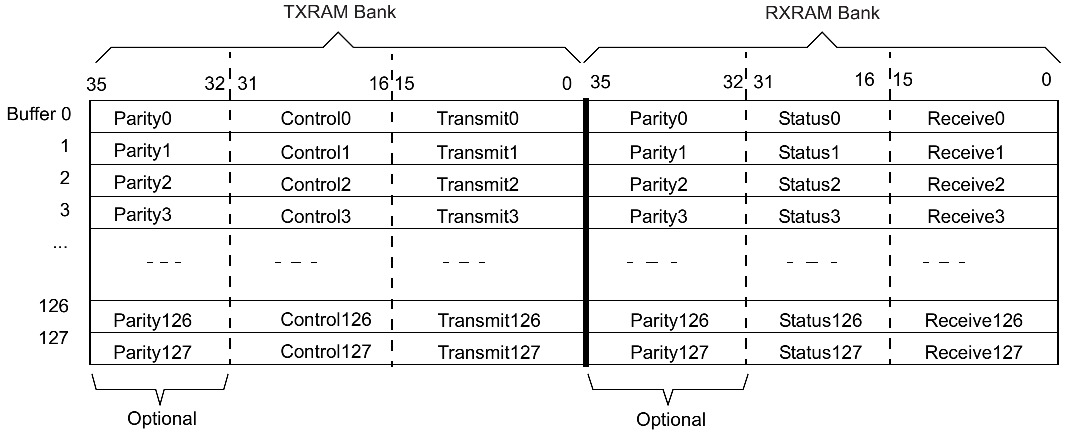
After completing a memory test sequence, specifically where ECC features are tested, user must read back the status flags in ECC_STAT and SBERRADDRx registers to ensure that they are in normal clear state by reading/writing appropriately. This can be performed before the start of a normal multi-buffer mode transactions as well.

NOTE: When ECC is supported, SBERRADDR0 will indicate only word address. SBERRADDR0[1:0] will always be 00.

28.4 Multi-buffer RAM

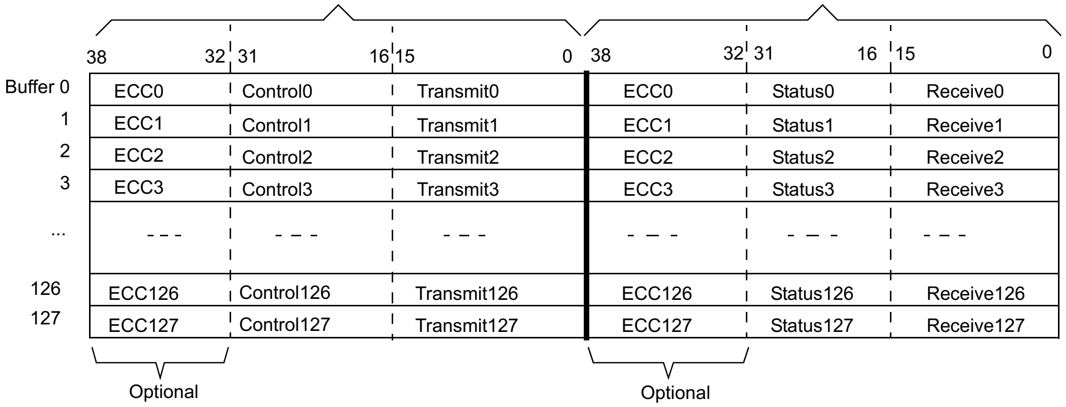
The multi-buffer RAM comprises of all buffers, which can be configured identically. The multi-buffer RAM contains two banks of up to128/256 words of 32 bits for a maximum configuration, one each for TXRAM (replicating the SPIDAT1 register) and RXRAM (replicating the SPIBUF register). The buffers can be partitioned into multiple transfer groups, each containing a variable number of buffers. Each of the buffers can be sub-divided into a 16-bit transmit field, a 16-bit receive field, a 16-bit control field, and a 16-bit status field. A 4-bit parity field per word is also included in each RAM bank, as shown in Figure 28-86. If ECC support is implemented for RAM fault detection, then a 7-bit ECC field per word is also included in each RAM bank, as shown in Figure 28-87.

Figure 28-86. Multi-buffer RAM Configuration When Parity Check is Supported



Depth will be up to 256 buffers, if EXTENDED_BUF feature is implemented.

Figure 28-87. Multi-buffer RAM Configuration When ECC Check is Supported



Depth will be up to 256 buffers, if EXTENDED_BUF feature is implemented.

All fields can be read and written with 8-bit, 16-bit, or 32-bit accesses.

The transmit fields can be written and read in the address range 000h to 1FFh. The transmit words contain data and control fields.

The receive RAM fields are read-only and can be accessed through the address range 200h to 3FCh. The receive words contain data and status fields.

The chip select number bit field CSNR[7:0] of the control field for a given word is mirrored into the corresponding receive-buffer status field after transmission.

The Parity is automatically calculated and copied to Parity location

NOTE: Refer to the specific device datasheet for the actual number of transmit and receive buffers.

Write to unimplemented buffer is overwriting the corresponding implemented buffer. In MIBSPI, if the RAM SIZE specified is 32 buffers, write to 33rd buffer overwrites 1st buffer, write to 34th buffer overwrites 2nd buffer, and so on.

28.4.1 Multi-buffer RAM Auto Initialization

When the MIBSPI is out of reset mode, auto initialization of multi-buffer RAM starts. The application code must check for BUFINITACTIVE bit to be 0 (multi-buffer RAM initialization is complete) before configuring multi-buffer RAM.

Besides the default auto initialization after reset, the auto-initialization sequence can also be done by:

1. Enable the global hardware memory initialization key by programming a value of 1010b to the bits [3:0] of the MINITGCR register of the System module.
2. Set the control bit for the multi-buffer RAM in the MSINENA System module register. This bit is device-specific for each memory that support auto-initialization. Please refer to the device datasheet to identify the control bit for the multi-buffer RAM. This starts the initialization process. The BUFINITACTIVE bit will get set to reflect that the initialization is ongoing.
3. When the memory initialization is completed, the corresponding status bit in the MINISTAT register will be set. Also, the BUFINITACTIVE bit will get cleared.
4. Disable the global hardware memory initialization key by programming a value of 0101 to the bits [3:0] of the MINITGCR register of the System module.

Please refer to the Architecture User Guide for more details on the memory auto-initialization process.

NOTE: During Auto Initialization process, all the multi-buffer mode registers (except MIBSPIE) will be reset to their default values. So, it should be ensured that Auto Initialization is completed before configuring the multi-buffer mode register.

28.4.2 Multi-buffer RAM Register Summary

This section describes the multi-buffer RAM control and transmit-data fields of each word of TXRAM, and the status and receive-data fields of each word of RXRAM. The base address for multi-buffer RAM is FF0E 0000h for MibSPI1 RAM, FF08 0000h for MibSPI2 RAM, FF0C 000h for MibSPI3 RAM, FF06 0000h for MibSPI4 RAM, and FF0A 0000h for MibSPI5 RAM.

Table 28-61. Multi-buffer RAM Register

Offset	Acronym	Register Description	Section
Base + 0h-1FFh	TXRAM	Multi-buffer RAM Transmit Data Register	Section 28.4.3
Base + 200h-3FFh	RXRAM	Multi-buffer RAM Receive Buffer Register	Section 28.4.4

28.4.3 Multi-buffer RAM Transmit Data Register (TXRAM)

Each word of TXRAM is a transmit-buffer register.

NOTE: Writing to only the control fields, bits 28 through 16, does not initiate any SPI transfer in master mode. This feature can be used to set up SPICLK phase or polarity before actually starting the transfer by only updating the DFSEL bit field to select the required phase and polarity combination.

Figure 28-88. Multi-buffer RAM Transmit Data Register (TXRAM) [offset = Base + 000-1FFh]

31 29	28	27	26	25 24	23 16
BUFMODE	CSHOLD	LOCK	WDEL	DFSEL	CSNR
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

15 0
TXDATA
R/W-0

LEGEND: R/W = Read/Write; -*n* = value after reset

Table 28-62. Multi-buffer RAM Transmit Data Register (TXRAM) Field Descriptions

Bit	Field	Value	Description
31-29	BUFMODE		Specify conditions that are recognized by the sequencer to initiate transfers of each buffer word. When one of the "skip" modes is selected, the sequencer checks the buffer status every time it reads from this buffer. If the current buffer status (TXFULL, RXEMPTY) does not match, the buffer is skipped without a data transfer. When one of the "suspend" modes is selected, the sequencer checks the buffer status when it reads from this buffer. If TXFULL and/or RXEMPTY do not match, the sequencer waits until a match occurs. No data transfer is initiated until the status condition of this buffer changes.
		0	**disabled.** The buffer is disabled.
		1h	**skip single-transfer mode.** Skip this buffer until the corresponding TXFULL flag is set (new transmit data is available).
		2h	**skip overwrite-protect mode.** Skip this buffer until the corresponding RXEMPTY flag is set (new receive data can be stored in RXDATA without data loss).
		3h	**skip single-transfer overwrite-protect mode**. Skip this buffer until both of the corresponding TXFULL and RXEMPTY flags are set. (new transmit data available and previous data received by the host).
		4h	**continuous mode.** Initiate a transfer each time the sequencer checks this buffer. Data words are retransmitted if the buffer has not been updated. Receive data is overwritten, even if it has not been read.
		5h	**suspend single-transfer mode.** Suspend-to-wait until the corresponding TXFULL flag is set (the sequencer stops at the current buffer until new transmit data is written in the TXDATA field).
		6h	**suspend overwrite-protect mode.** Suspend-to-wait until the corresponding RXEMPTY flag is set (the sequencer stops at the current buffer until the previously-received data is read by the host.
		7h	**suspend single-transfer overwrite-protect mode.** Suspend-to-wait until the corresponding TXFULL and RXEMPTY flags are set (the sequencer stops at the current buffer until new transmit data is written into the TXDATA field and the previously-received data is read by the host).
28	CSHOLD		Chip select hold mode. The CSHOLD bit is supported in master mode only, it is ignored in slave mode. CSHOLD defines the behavior of the chip select line at the end of a data transfer.
		0	The chip select signal is deactivated at the end of a transfer after the T2CDELAY time has passed. If two consecutive transfers are dedicated to the same chip select this chip select signal will be deactivated for at least 2VCLK cycles before it is activated again.
		1	The chip select signal is held active at the end of a transfer until a control field with new data and control information is loaded into SPIDAT1. If the new chip select number equals the previous one, the active chip select signal is extended until the end of transfer with CSHOLD cleared, or until the chip-select number changes.

Table 28-62. Multi-buffer RAM Transmit Data Register (TXRAM) Field Descriptions (continued)

Bit	Field	Value	Description
27	LOCK		Lock two consecutive buffer words. Do not allow interruption by TG's with higher priority.
		0	Any higher-priority TG can begin at the end of the current transaction.
		1	A higher-priority TG cannot occur until after the next unlocked buffer word is transferred.
26	WDEL		Enable the delay counter at the end of the current transaction. **Note: The WDEL bit is supported in master mode only. In slave mode, this bit will be ignored.**
		0	No delay will be inserted. However, the $\overline{\text{SPICS}}$ pins will still be de-activated for at least for 2VCLK cycles if CSHOLD = 0. **Note: The duration for which the $\overline{\text{SPICS}}$ pin remains deactivated also depends upon the time taken to supply a new word after completing the shift operation (in compatibility mode). If TXBUF is already full, then the $\overline{\text{SPICS}}$ pin will be deasserted for at least two VCLK cycles (if WDEL = 0).**
		1	After a transaction, WDELAY of the corresponding data format will be loaded into the delay counter. No transaction will be performed until the WDELAY counter overflows. The $\overline{\text{SPICS}}$ pins will be de-activated for at least (WDELAY + 2) × VCLK_Period duration.
25-24	DFSEL		Data word format select.
		0	Data word format 0 is selected.
		1h	Data word format 1 is selected.
		2h	Data word format 2 is selected.
		3h	Data word format 3 is selected.
23-16	CSNR	0-FFh	Chip select (CS) number. CSNR defines the chip select pins that will be activated during the data transfer. CSNR is a bit-mask that controls all chip select pins. See Table 28-63. **Note: If your MibSPI has less than 8 chip select pins, all unused upper bits will be 0. For example, MiBSPI3 has 6 chip select pins, if you write FFh to CSNR, the actual number stored in CSNR is 3Fh.**
15-0	TXDATA	0-7FFFh	Transfer data. When written, these bits are copied to the shift register if it is empty. If the shift register is not empty, then they are held in TXBUF. SPIEN must be set to 1 before this register can be written to. Writing a 0 to SPIEN forces the lower 16 bits of TXDATA to 0. A write to this register (or to the TXDATA field only) drives the contents of the CSNR field on the $\overline{\text{SPICS}}$ pins, if the pins are configured as functional pins (automatic chip select, see Section 28.2.1). When this register is read, the contents of TXBUF, which holds the latest data written, will be returned. **Note: Regardless of the character length, the transmit data should be right-justified before writing to the SPIDAT1 register.**

Table 28-63. Chip Select Number Active

CSNR Value	Chip Select Active:							CSNR Value	Chip Select Active:					
	CS[5][1]	CS[4][1]	CS[3][1]	CS[2][1]	CS[1][1]	CS[0]			CS[5][1]	CS[4][1]	CS[3][1]	CS[2][1]	CS[1][1]	CS[0]
0h	No chip select pin is active.							20h	x					
1h						x		21h	x					x
2h					x			22h	x				x	
3h					x	x		23h	x				x	x
4h				x				24h	x			x		
5h				x		x		25h	x			x		x
6h				x	x			26h	x			x	x	
7h				x	x	x		27h	x			x	x	x
8h			x					28h	x		x			
9h			x			x		29h	x		x			x
Ah			x		x			2Ah	x		x		x	
Bh			x		x	x		2Bh	x		x		x	x
Ch			x	x				2Ch	x		x	x		
Dh			x	x		x		2Dh	x		x	x		x
Eh			x	x	x			2Eh	x		x	x	x	
Fh			x	x	x	x		2Fh	x		x	x	x	x
10h		x						30h	x	x				
11h		x				x		31h	x	x				x
12h		x			x			32h	x	x			x	
13h		x			x	x		33h	x	x			x	x
14h		x		x				34h	x	x		x		
15h		x		x		x		35h	x	x		x		x
16h		x		x	x			36h	x	x		x	x	
17h		x		x	x	x		37h	x	x		x	x	x
18h		x	x					38h	x	x	x			
19h		x	x			x		39h	x	x	x			x
1Ah		x	x		x			3Ah	x	x	x		x	
1Bh		x	x		x	x		3Bh	x	x	x		x	x
1Ch		x	x	x				3Ch	x	x	x	x		
1Dh		x	x	x		x		3Dh	x	x	x	x		x
1Eh		x	x	x	x			3Eh	x	x	x	x	x	
1Fh		x	x	x	x	x		3Fh	x	x	x	x	x	x

[1] If your MibSPI does not have this chip select pin, this bit is 0.

28.4.4 Multi-buffer RAM Receive Buffer Register (RXRAM)

Each word of RXRAM is a receive-buffer register.

Figure 28-89. Multi-buffer RAM Receive Buffer Register (RXRAM) [offset = RAM Base + 200-3FFh]

31	30	29	28	27	26	25	24
RXEMPTY	RXOVR	TXFULL	BITERR	DESYNC	PARITYERR	TIMEOUT	DLENERR
RS-1	RC-0	R-0	RC-0	RC-0	RC-0	RC-0	RC-0

23	16
LCSNR	
R-0	

15	0
RXDATA	
R/W-0	

LEGEND: R = Read only; R/W = Read/Write; C = Clear; S = Set; -*n* = value after reset

Table 28-64. Multi-buffer Receive Buffer Register (RXRAM) Field Descriptions

Bit	Field	Value	Description
31	RXEMPTY		Receive data buffer empty. When the host reads the RXDATA field or the entire RXRAM register, it automatically sets the RXEMPTY flag. When a data transfer is completed, the received data is copied into RXDATA, and the RXEMPTY flag is cleared.
		0	New data has been received and copied into RXDATA.
		1	No data has been received since the last read of RXDATA.
			This flag gets set to 1 under the following conditions: • Reading the RXDATA field of the RXRAM register. • Writing a 1 to clear the RXINTFLG bit in the SPI Flag Register (SPIFLG). Write-clearing the RXINTFLG bit before reading RXDATA indicates the received data is being ignored. Conversely, RXINTFLG can be cleared by reading the RXDATA field of RXRAM (or the entire register).
30	RXOVR		Receive data buffer overrun. When a data transfer is completed and the received data is copied into RXBUF while it is already full, RXOVR is set. Overruns always occur to RXBUF, not to RXRAM; the contents of RXRAM are overwritten only after it is read by the Peripheral (VBUSP) master (CPU, DMA, or other host processor). If enabled, the RXOVRN interrupt is generated when RXBUF is overwritten, and reading either SPI Flag Register (SPIFLG) or SPIVECTx shows the RXOVRN condition. Two read operations from the RXRAM register are required to reach the overwritten buffer word (one to read RXRAM, which then transfers RXDATA into RXRAM for the second read). **Note: This flag is cleared to 0 when the RXDATA field of the RXRAM register is read.** **Note: A special condition under which RXOVR flag gets set.If both RXRAM and RXBUF are already full and while another buffer receive is underway, if any errors such as TIMEOUT, BITERR and DLEN_ERR occur, then RXOVR in RXBUF and SPI Flag Register (SPIFLG) will be set to indicate that the status flags are getting overwritten by the new transfer. This overrun should be treated like a normal receive overrun.**
		0	No receive data overrun condition occurred since last read of the data field.
		1	A receive data overrun condition occurred since last read of the data field.
29	TXFULL		Transmit data buffer full. This flag is a read-only flag. Writing into the SPIDAT0 or SPIDAT1 field while the TX shift register is full will automatically set the TXFULL flag. Once the word is copied to the shift register, the TXFULL flag will be cleared. Writing to SPIDAT0 or SPIDAT1 when both TXBUF and the TX shift register are empty does not set the TXFULL flag.
		0	The transmit buffer is empty; SPIDAT0/SPIDAT1 is ready to accept a new data.
		1	The transmit buffer is full; SPIDAT0/SPIDAT1 is not ready to accept new data.

Table 28-64. Multi-buffer Receive Buffer Register (RXRAM) Field Descriptions (continued)

Bit	Field	Value	Description
28	BITERR		Bit error. There was a mismatch of internal transmit data and transmitted data.
			Note: This flag is cleared to 0 when the RXDATA field of the RXRAM register is read.
		0	No bit error occurred.
		1	A bit error occurred. The SPI samples the signal of the transmit pins (master: SIMOx, slave: SOMIx) at the receive point (one-half clock cycle after the transmit point). If the sampled value differs from the transmitted value, a bit error is detected and the BITERR flag is set. Possible reasons for a bit error include noise, an excessively high bit rate, capacitive load, or another master/slave trying to transmit at the same time.
27	DESYNC		Desynchronization of slave device. This bit is valid in master mode only.
			The master monitors the ENA signal coming from the slave device and sets the DESYNC flag if ENA is deactivated before the last reception point or after the last bit is transmitted plus $t_{T2EDELAY}$. If DESYNCENA is set, an interrupt is asserted. Desynchronization can occur if a slave device misses a clock edge coming from the master.
			Note: In the Compatibility Mode MibSPI, under some circumstances it is possible for a desync error detected for the previous buffer to be visible in the current buffer. This is because the receive completion flag/interrupt is generated when the buffer transfer is completed. But desynchronization is detected after the buffer transfer is completed. So, if the VBUS master reads the received data quickly when an RXINT is detected, then the status flag may not reflect the correct desync condition. In multi-buffer mode, the desync flag is always guaranteed to be for the current buffer.
			Note: This flag is cleared to 0 when the RXDATA field of the RXRAM register is read.
		0	No slave desynchronization is detected.
		1	A slave device is desynchronized.
26	PARITYERR		Parity error. The calculated parity differs from the received parity bit.
			If the parity generator is enabled (selected individually for each buffer) an even or odd parity bit is added at the end of a data word. During reception of the data word, the parity generator calculates the reference parity and compares it to the received parity bit. If a mismatch is detected, the PARITYERR flag is set.
			Note: This flag is cleared to 0 when the RXDATA field of the RXRAM register is read.
		0	No parity error is detected.
		1	A parity error occurred.
25	TIMEOUT		Time-out because of non-activation of $\overline{\text{SPIENA}}$ pin.
			The SPI generates a time-out when the slave does not respond in time by activating the ENA signal after the chip select signal has been activated. If a time-out condition is detected, the corresponding chip select is deactivated immediately and the TIMEOUT flag is set. In addition, the TIMEOUT flag in the status field of the corresponding buffer and in the SPI Flag Register (SPIFLG) is set.
			This bit is valid only in master mode.
			Note: This flag is cleared to 0 when the RXDATA field of the RXRAM register is read.
		0	No $\overline{\text{SPIENA}}$ pin time-out occurred.
		1	An $\overline{\text{SPIENA}}$ signal time-out occurred.
24	DLENERR		Data length error flag.
			Note: This flag is cleared to 0 when the RXDATA field of the RXRAM register is read.
		0	No data-length error occurred.
		1	A data length error occurred.
23-16	LCSNR	0-FFh	Last chip select number. LCSNR in the status field is a copy of CSNR in the corresponding control field. It contains the chip select number that was activated during the last word transfer.
15-0	RXDATA	0-FFFFh	SPI receive data. This is the received word, transferred from the receive shift-register at the end of a transfer. Regardless of the programmed character length and the direction of shifting, the received data is stored right-justified in the register.

28.5 Parity\ECC Memory

Parity/ECC portion of multi-buffer RAM is not accessible by the CPU during normal operating modes. However each read or write operation to the Control/Data/Status portion of the multi-buffer RAM causes reads/writes to the parity/ECC portion as well.

- Each write to the multi-buffer RAM (either from the VBUS interface or by the MibSPI itself) causes a write operation to the Parity/ECC portion of RAM simultaneously to update the equivalent parity/ECC bits.
- Each read operation from the multi-buffer RAM (either from the VBUS interface or by the MibSPI itself) causes a read operation from the Parity/ECC portion of the RAM for parity/ECC comparison purpose.
- Reads/Writes to multi-buffer RAM could either be caused by any CPU/DMA accesses or by the Sequencer logic of MibSPI itself.

For testing the Parity/ECC portion of the multi-buffer RAM that is a 4-bit or 7-bit field per word address, a separate parity/ECC memory test mode is available. The parity memory test mode can be enabled and disabled by the PTESTEN bit in PAR_ECC_CTRL register and the ECCDIAG_EN bit in ECCDIAG_CTRL register.

During the parity test mode, the parity locations are addressable at the address between RAM_BASE_ADDR + 0x400h and RAM_BASE_ADDR + 0x7FFh. Each location corresponds, sequentially, to each TXRAM word, then to each RXRAM word. See Figure 28-90 for a diagram of the memory map of parity memory during normal operating mode and during parity test mode while EXTENDED_BUF mode is disabled or the feature is not implemented. See Figure 28-91 for a diagram of the memory map of parity memory during normal operating mode and during parity test mode while EXTENDED_BUF mode is enabled.

During Parity/ECC test mode, after writing the Data/Control portion of the RAM, the Parity/ECC locations can be written with wrong parity/ECC bits to intentionally cause Parity/ECC Errors.

See the device-specific data sheet to get the actual base address of the multi-buffer RAM.

NOTE: The RX_RAM_ACCESS bit can also be set to 1 during the Parity/ECC Test mode to be able to write to RXRAM locations for test purpose. Both Parity/ECC bits testing and RXRAM testing can be done together.

There are 4 bits of parity corresponding to each of the 32-bit multi-buffer locations. Individual bits in the parity memory are byte-addressable in parity test mode. See the example in Section 28.5.1 for further details.

If ECC is enabled, there are 7 ECC-bits corresponding to each of the 32-bit multi-buffer locations. See the example in Section 28.5.1 for further details.

NOTE: Polarity of the parity (odd/even) varies by device. In some devices, a control register in the system module can be used to select odd or even parity.

Figure 28-90. Memory-Map for Parity Locations During Normal and Test Mode While EXTENDED_BUF Mode is Disabled or the Feature is Not Implemented

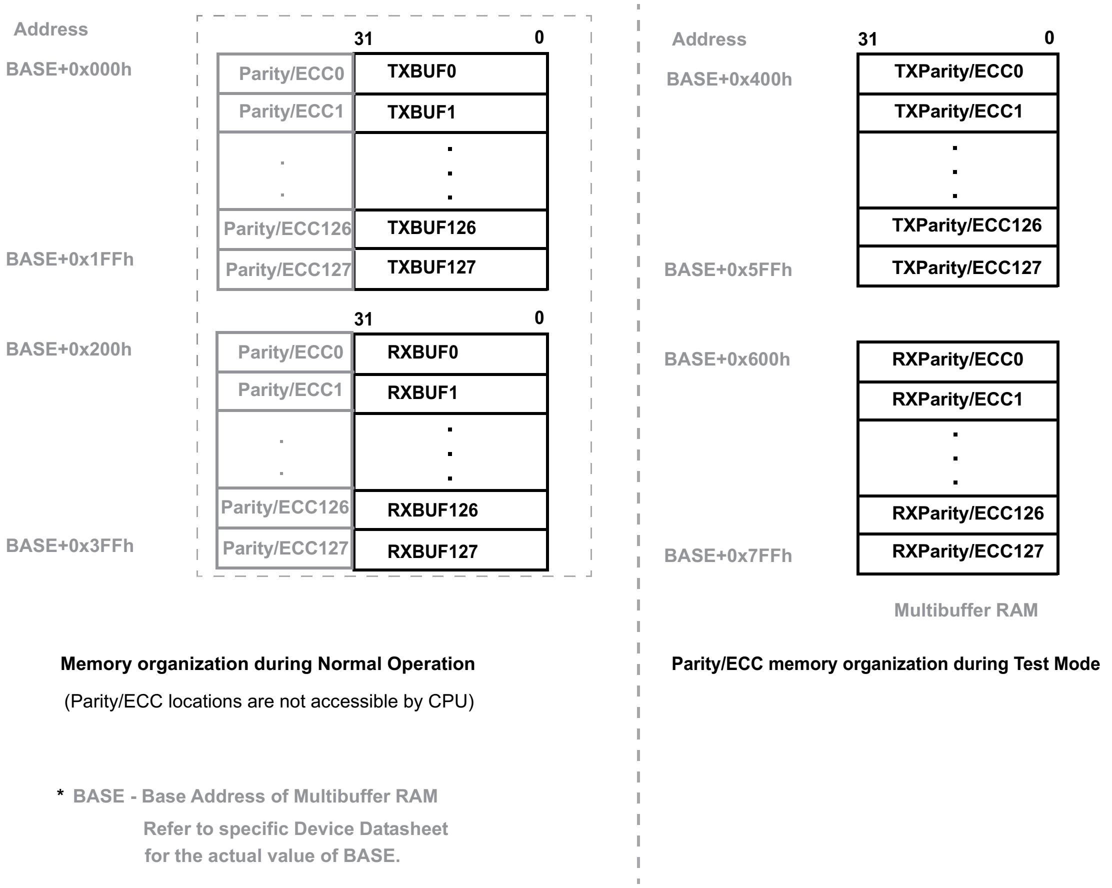
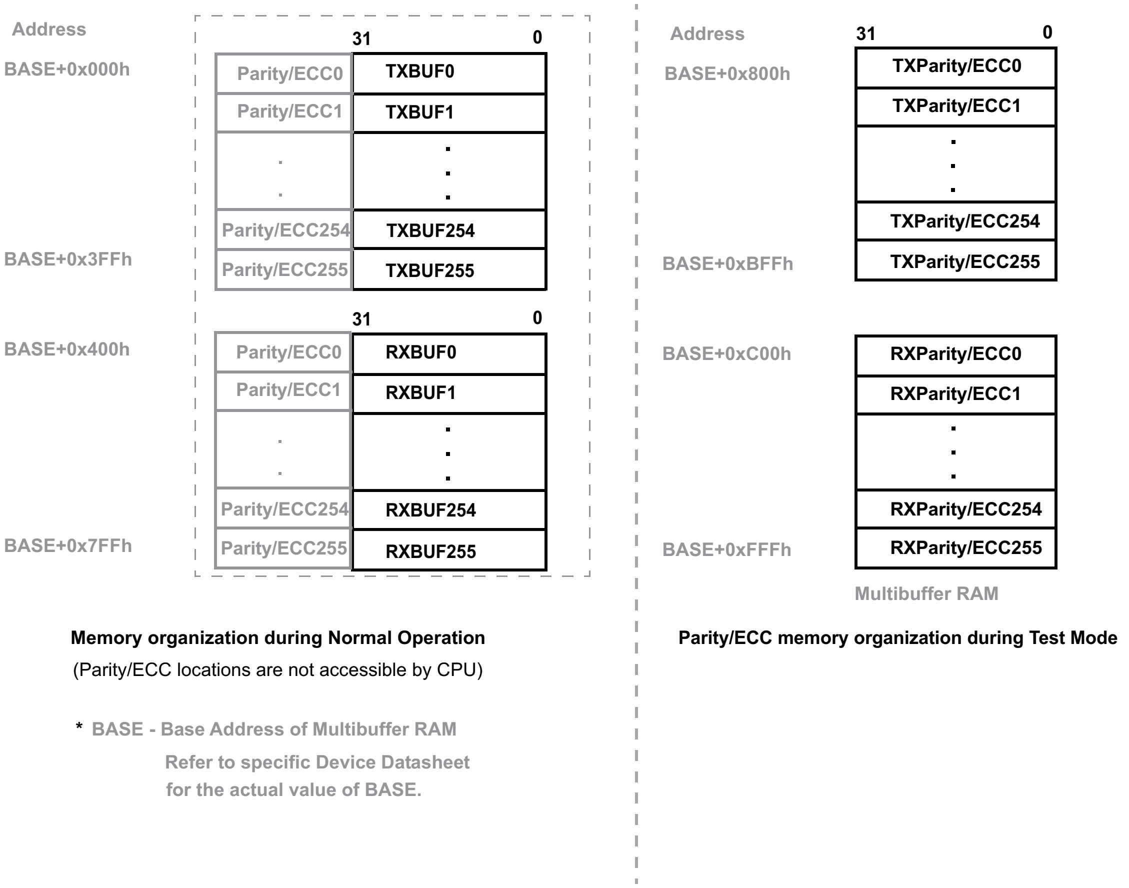


Figure 28-91. Memory-Map for Parity Locations During Normal and Test Mode While EXTENDED_BUF Mode is Enabled

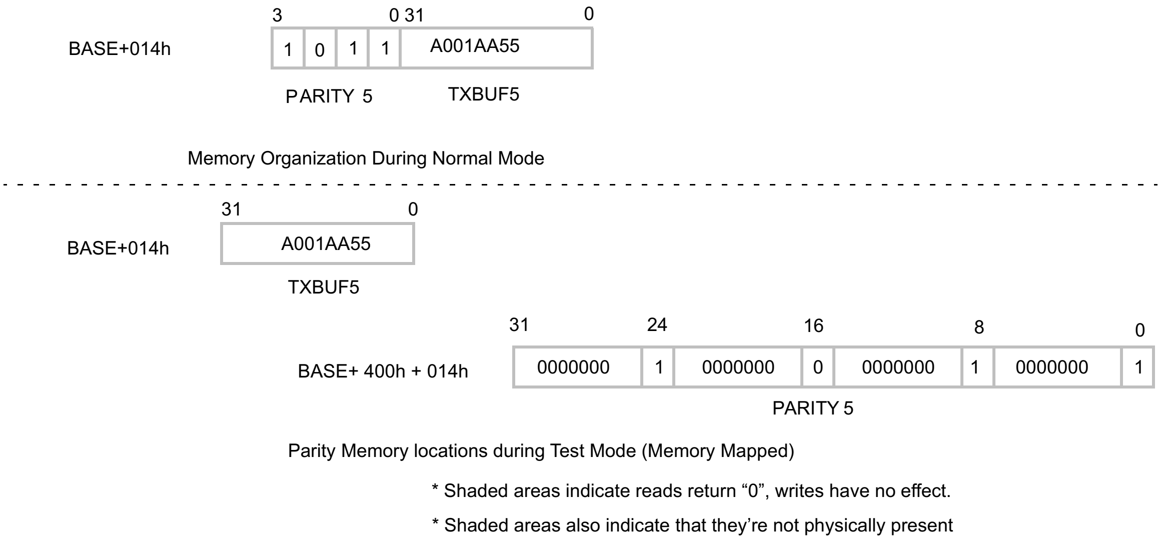


28.5.1 Example of Parity Memory Organization

Suppose TXBUF5 (6th location in TXRAM) in the multi-buffer RAM is written with a value of A001_AA55. If the polarity of the parity is set to odd, the corresponding parity location parity5 will get updated with equivalent parity of 1011 in its field.

During parity-memory test mode, these bits can be individually byte addressed. The return data will be a byte adjusted with actual parity bit in the LSB of the byte. If a word is read from the word-boundary address of parity locations, then each bit of the 4-bit parity is byte-adjusted and a 32-bit word is returned. 0s will be padded into the parity bits to get each byte. See Figure 28-92 for a diagram.

Figure 28-92. Example of Memory-Mapped Parity Locations During Test Mode


3
0 31
0
BASE+014h
1
0
1
1
A001AA55
PARITY 5
TXBUF5
Memory Organization During Normal Mode
31
0
BASE+014h
A001AA55
TXBUF5
31
24
16
8
0
BASE+ 400h + 014h
0000000
1
0000000
0
0000000
1
0000000
1
PARITY 5
Parity Memory locations during Test Mode (Memory Mapped)
* Shaded areas indicate reads return “0”, writes have no effect.
* Shaded areas also indicate that they’re not physically present

NOTE: Read Access to Parity Memory Locations

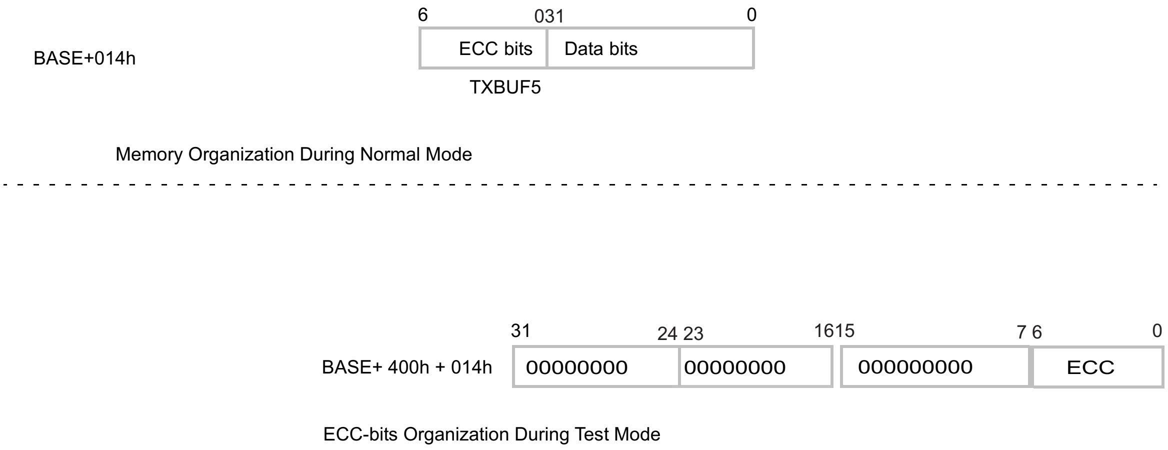
Parity memory locations can be read even without entering into parity memory test mode. Their address remains as in memory test mode. It is only to enter parity-memory test mode to enable write access to the parity memory locations.

28.5.2 Example of ECC Memory Organization

Suppose TXBUF5 (6th location in TXRAM portion) in the multi-buffer RAM is written with a value of A001_AA55, then the corresponding ECC-bits will be updated in ECC location.

The ECC bits can be accessed by user, when Memory Test mode is enabled and additionally diagnostic mode is also enabled. The actual ECC bits will be aligned as shown in Figure 28-93.

Figure 28-93. Example of ECC Bit Locations During Test Mode



NOTE: Access to ECC locations

ECC locations can be read/write only when Parity Memory Test mode and diagnostic mode is enabled

28.6 MibSPI Pin Timing Parameters

The pin timings of SPI can be classified based on its mode of operation. In each mode, different configurations like Phase & Polarity affect the pin timings.

The pin directions are based on the mode of operation.

Master mode SPI:

- SPICLK (SPI Clock) - Output
- SPISIMO (SPI Slave In Master Out) - Output
- $\overline{\text{SPICS}}$ (SPI Slave Chip Selects) - Output
- SPISOMI (SPI Slave Out Master In) - Input
- $\overline{\text{SPIENA}}$ (SPI slave ready Enable) - Input

Slave mode SPI:

- SPICLK - Input
- SPISIMO - Input
- $\overline{\text{SPICS}}$ - Input
- SPISOMI - Output
- $\overline{\text{SPIENA}}$ - Output

NOTE: All the following timing diagrams are with Phase = 0 and Polarity = 0, unless explicitly stated otherwise.

28.6.1 Master Mode Timings for SPI/MibSPI

Figure 28-94. SPI/MibSPI Pins During Master Mode 3-Pin Configuration

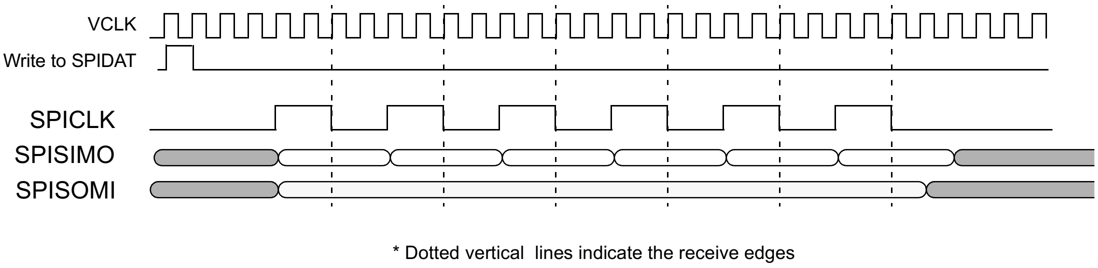


Figure 28-95. SPI/MibSPI Pins During Master Mode 4-Pin with $\overline{\text{SPICS}}$ Configuation

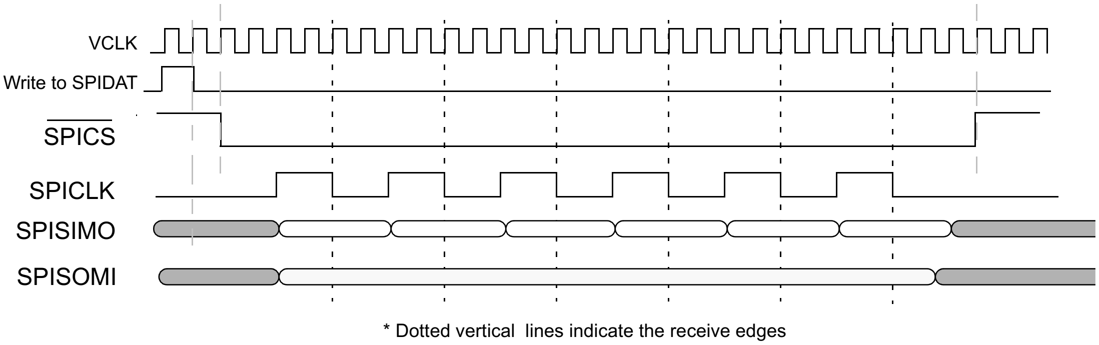
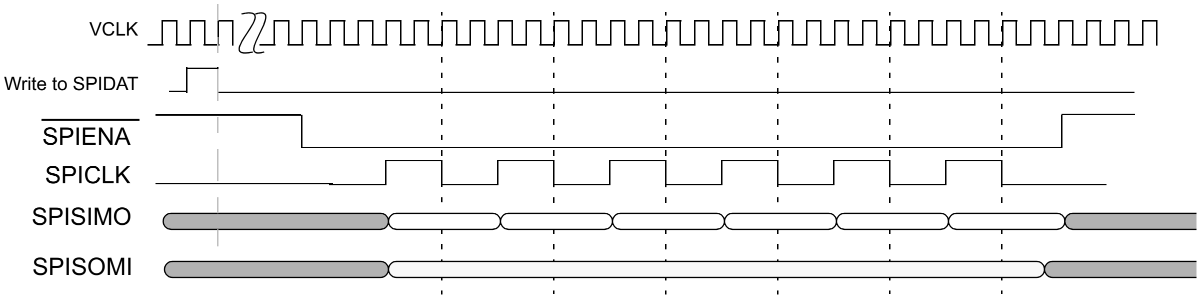


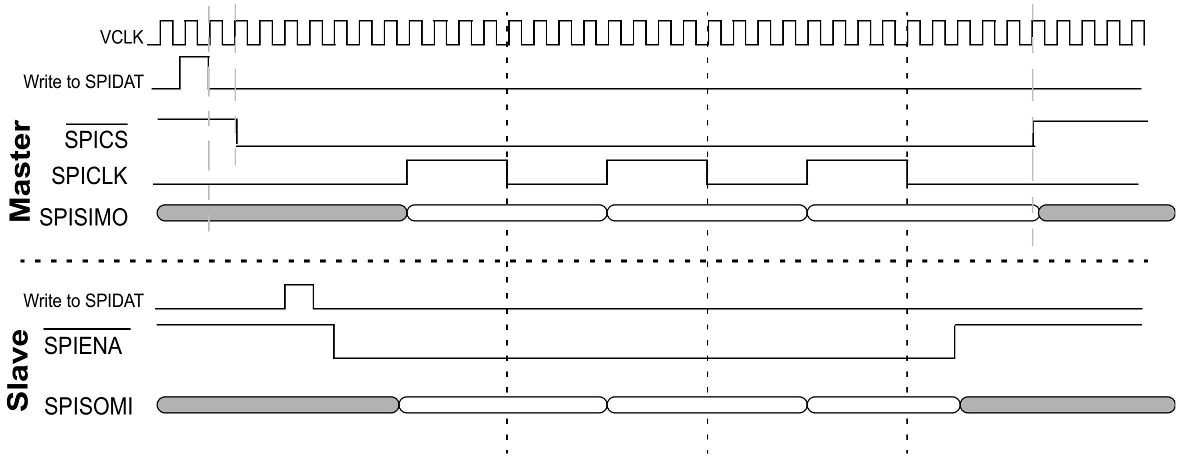
Figure 28-96. SPI/MibSPI Pins During Master Mode in 4-Pin with $\overline{\text{SPIENA}}$ Configuration



* De-activation of $\overline{\text{SPIENA}}$ pin is controlled by the Slave.

* Dotted vertical lines indicate the receive edges

Figure 28-97. SPI/MibSPI Pins During Master/Slave Mode with 5-Pin Configuration

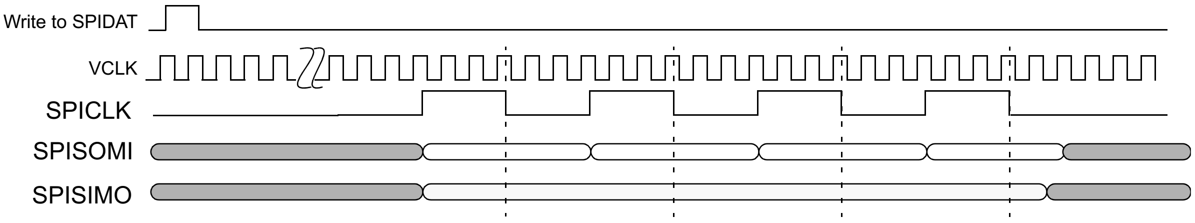


* Dotted vertical lines indicate the receive edges for the Master

* ENABLE_HIGHZ is cleared to 0 in Slave SPI

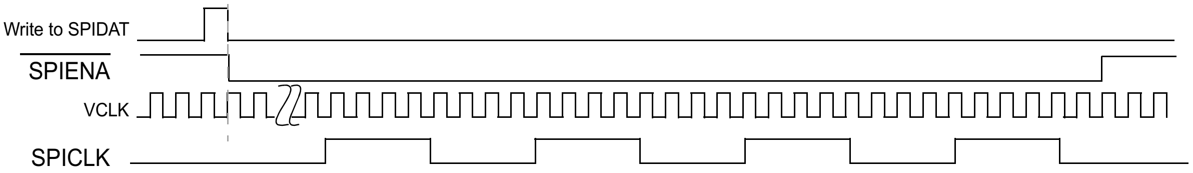
28.6.2 Slave Mode Timings for SPI/MibSPI

Figure 28-98. SPI/MibSPI Pins During Slave Mode 3-Pin Configuration



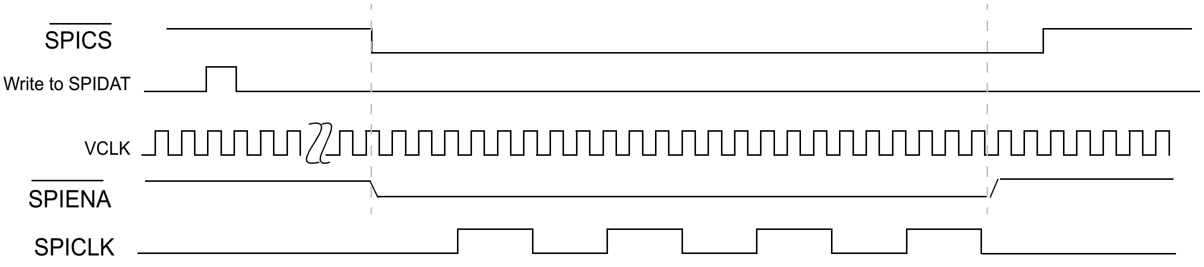
* Dotted vertical lines indicate the receive edges

Figure 28-99. SPI/MibSPI Pins During Slave Mode in 4-Pin with SPIENA Configuration



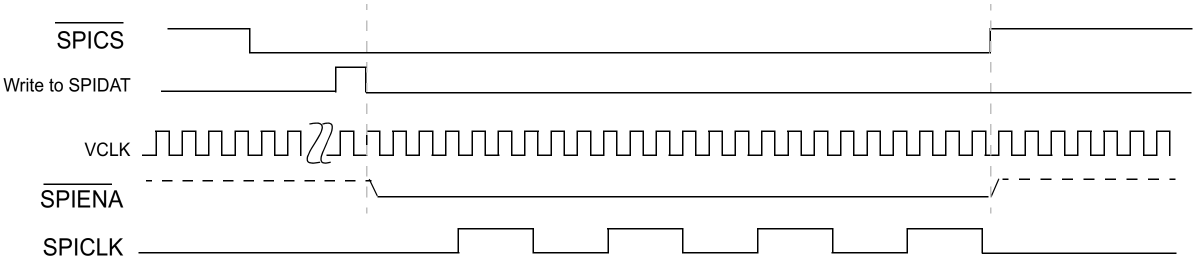
* Diagram shows a relationship between the SPIENA from Slave and SPICLK from Master

Figure 28-100. SPI/MibSPI Pins During Slave Mode in 5-Pin Configuration (Single Slave)



* ENABLE_HIGHZ is cleared to 0 in Slave SPI
* Diagram shows relationship between the SPICS from a Master to SPIENA from Slave SPI when SPIENA is configured in Push-Pull mode

Figure 28-101. SPI/MibSPI Pins During Slave Mode in 5-Pin Configuration (Single/Multi-Slave)



* ENABLE_HIGHZ is set to 1 in Slave SPI
* Diagram shows relationship between the SPICS from a Master to SPIENA from Slave SPI when SPIENA is configured in High-Impedance mode

28.6.3 Master Mode Timing Parameter Details

In case of Master, the module drives out SPICLK. It also drives out the Transmit data on SPISIMO with respect to its internal SPICLK. In case of Master mode, the RX data on the SPISOMI pin is registered with respect to SPICLK received through the input buffer from the I/O pad.

If the chip select pin is functional, then the Master will drive out the $\overline{\text{SPICS}}$ pins before starting the SPICLK. If the $\overline{\text{SPIENA}}$ pin is functional, then the Master will wait for an active low from the Slave on the input pin to start the SPICLK.

28.6.4 Slave Mode Timing Parameter Details

In case of Slave mode, the module will drive only the SPISOMI and $\overline{\text{SPIENA}}$ pins. All other pins are inputs to it. The RX data on the SPISIMO pin will be registered with respect to the SPICLK pin. The Slave will use the $\overline{\text{SPICS}}$ pin to drive out the $\overline{\text{SPIENA}}$ pin if both are functional. If 4-pin with $\overline{\text{SPIENA}}$ is configured, then the Slave will drive out an active-low signal on the $\overline{\text{SPIENA}}$ pin when new data is written to the TX Shift Register. Irrespective of 4-pin with $\overline{\text{SPIENA}}$ or 5-pin configuration, the Slave will deassert the $\overline{\text{SPIENA}}$ pin after the last bit is received. If ENABLE_HIGHZ (SPIINT0.24) bit is 0, the de-asserted value of the $\overline{\text{SPIENA}}$ pin will be 1. Otherwise, it will depend upon the internal pull up or pull down resistor (if implemented) depending upon the Specification of the Chip.

Chapter 29

SPNU563A – March 2018

Serial Communication Interface (SCI)/ Local Interconnect Network (LIN) Module

This chapter describes the serial communication interface (SCI) / local interconnect network (LIN) module. The SCI/LIN is compliant to the LIN 2.1 protocol specified in the *LIN Specification Package*. This module can be configured to operate in either SCI (UART) or LIN mode.

NOTE: This chapter describes a superset implementation of the LIN/SCI module that includes features and functionality that require DMA. Since not all devices have DMA capability, consult your device-specific datasheet to determine applicability of these features and functions to your device being used.

Topic **Page**

29.1 Introduction and Features

The SCI/LIN module can be programmed to work either as an SCI or as a LIN. The core of the module is an SCI. The SCI's hardware features are augmented to achieve LIN compatibility.

The SCI module is a universal asynchronous receiver-transmitter that implements the standard nonreturn to zero format. The SCI can be used to communicate, for example, through an RS-232 port or over a K-line.

The LIN standard is based on the SCI (UART) serial data link format. The communication concept is single-master/multiple-slave with a message identification for multi-cast transmission between any network nodes.

Throughout the chapter Compatibility Mode refers to SCI Mode functionary of SCI/LIN Module. Section 29.2 explains about the SCI functionality and Section 29.3 explains about the LIN functionality. Though the registers are common for LIN and SCI, the register descriptions has notes to identify the register/bit usage in different modes.

29.1.1 SCI Features

The following are the features of the SCI module:

- Standard universal asynchronous receiver-transmitter (UART) communication
- Supports full- or half-duplex operation
- Standard nonreturn to zero (NRZ) format
- Double-buffered receive and transmit functions in compatibility mode
- Supports two individually enabled interrupt lines: level 0 and level 1
- Configurable frame format of 3 to 13 bits per character based on the following:
  - Data word length programmable from one to eight bits
  - Additional address bit in address-bit mode
  - Parity programmable for zero or one parity bit, odd or even parity
  - Stop programmable for one or two stop bits
- Asynchronous or isosynchronous communication modes
- Two multiprocessor communication formats allow communication between more than two devices
- Sleep mode is available to free CPU resources during multiprocessor communication and then wake up to receive an incoming message
- The 24-bit programmable baud rate supports 2^{24} different baud rates provide high accuracy baud rate selection
- At 100-MHz peripheral clock, 3.125 Mbits/s is the Max Baud Rate achievable
- Capability to use Direct Memory Access (DMA) for transmit and receive data
- Five error flags and Seven status flags provide detailed information regarding SCI events
- Two external pins: LINRX and LINTX
- Multi-buffered receive and transmit units

NOTE: SCI/LIN module does not support UART hardware flow control. This feature can be implemented in software using a general purpose I/O pin.

29.1.2 LIN Features

The following are the features of the LIN module:

- Compatibility with LIN 1.3, 2.0, and 2.1protocols
- Configurable Baud Rate up to 20 Kbits/s
- Two external pins: LINRX and LINTX.
- Multi-buffered receive and transmit units
- Identification masks for message filtering
- Automatic master header generation
  - Programmable synchronization break field
  - Synchronization field
  - Identifier field
- Slave automatic synchronization
  - Synchronization break detection
  - Optional baud rate update
  - Synchronization validation
- 2^{31} programmable transmission rates with 7 fractional bits
- Wakeup on LINRX dominant level from transceiver
- Automatic wakeup support
  - Wakeup signal generation
  - Expiration times on wakeup signals
- Automatic bus idle detection
- Error detection
  - Bit error
  - Bus error
  - No-response error
  - Checksum error
  - Synchronization field error
  - Parity error
- Capability to use Direct Memory Access (DMA) for transmit and receive data.
- 2 Interrupt lines with priority encoding for:
  - Receive
  - Transmit
  - ID, error, and status
- Support for LIN 2.0 checksum
- Enhanced synchronizer finite state machine (FSM) support for frame processing
- Enhanced handling of extended frames
- Enhanced baud rate generator
- Update wakeup/go to sleep

29.1.3 Block Diagram

The SCI/LIN module contains core SCI block with added sub-blocks to support LIN protocol.

Three Major components of the SCI Module are:

- Transmitter
- Baud Clock Generator
- Receiver

Transmitter (TX) contains two major registers to perform the double- buffering:

- The transmitter data buffer register (SCITD) contains data loaded by the CPU to be transferred to the shift register for transmission.
- The transmitter shift register (SCITXSHF) loads data from the data buffer (SCITD) and shifts data onto the LINTX pin, one bit at a time.

Baud Clock Generator

- A programmable baud generator produces either a baud clock scaled from VBUSP CLK.

Receiver (RX) contains two major registers to perform the double- buffering:

- The receiver shift register (SCIRXSHF) shifts data in from the LINRX pin one bit at a time and transfers completed data into the receive data buffer.
- The receiver data buffer register (SCIRD) contains received data transferred from the receiver shift register

The SCI receiver and transmitter are double-buffered, and each has its own separate enable and interrupt bits. The receiver and transmitter may each be operated independently or simultaneously in full duplex mode.

To ensure data integrity, the SCI checks the data it receives for breaks, parity, overrun, and framing errors. The bit rate (baud) is programmable to over 16 million different rates through a 24-bit baud-select register. Figure 29-1 shows the detailed SCI block diagram.

The SCI/LIN module is based on the standalone SCI with the addition of an error detector (parity calculator, checksum calculator, and bit monitor), a mask filter, a synchronizer, and a multi-buffered receiver and transmitter. The SCI interface, the DMA control subblocks and the baud generator are modified as part of the hardware enhancements for LIN compatibility. Figure 29-2 shows the SCI/LIN block diagram.

Figure 29-1. SCI Block Diagram

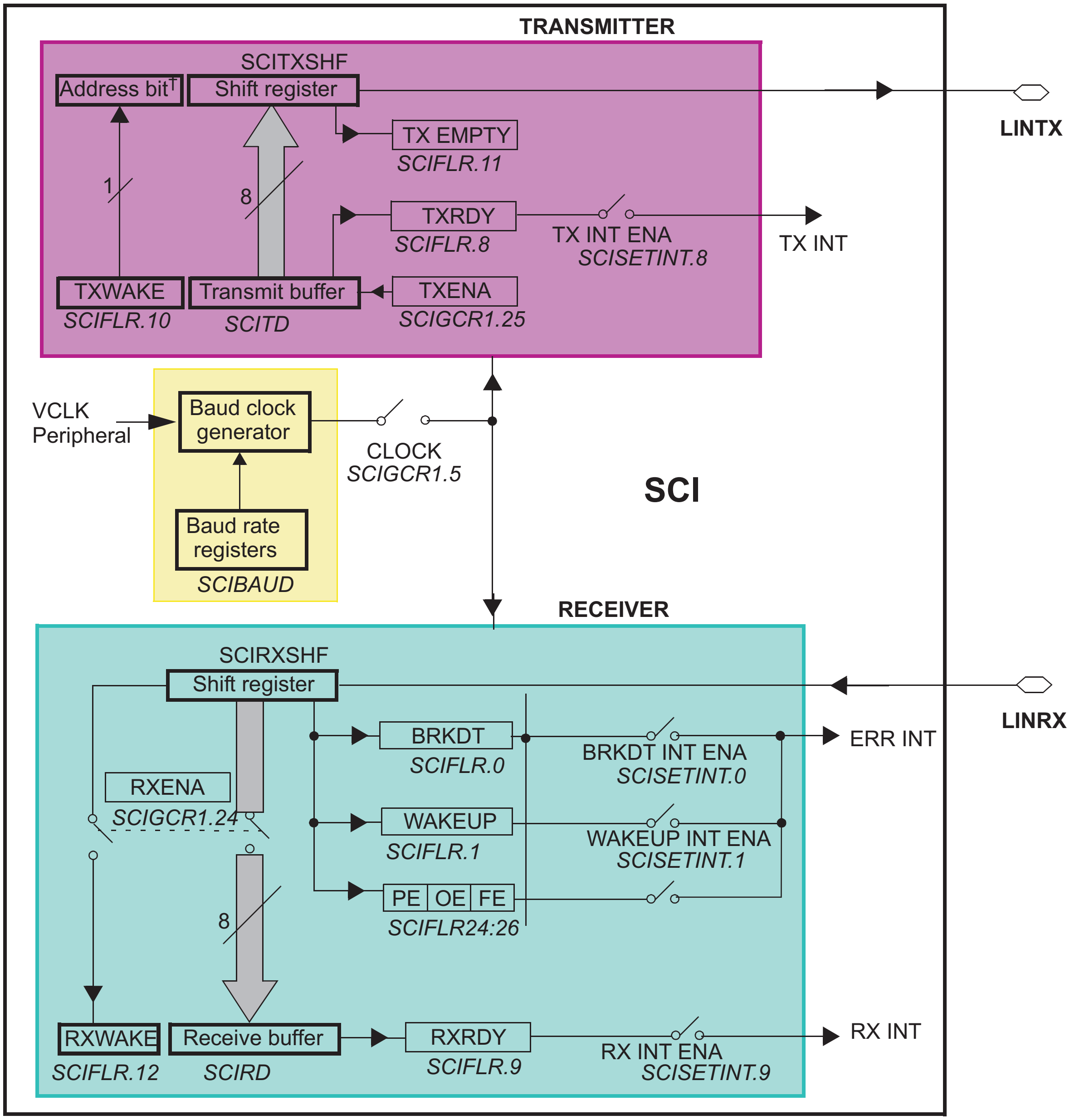
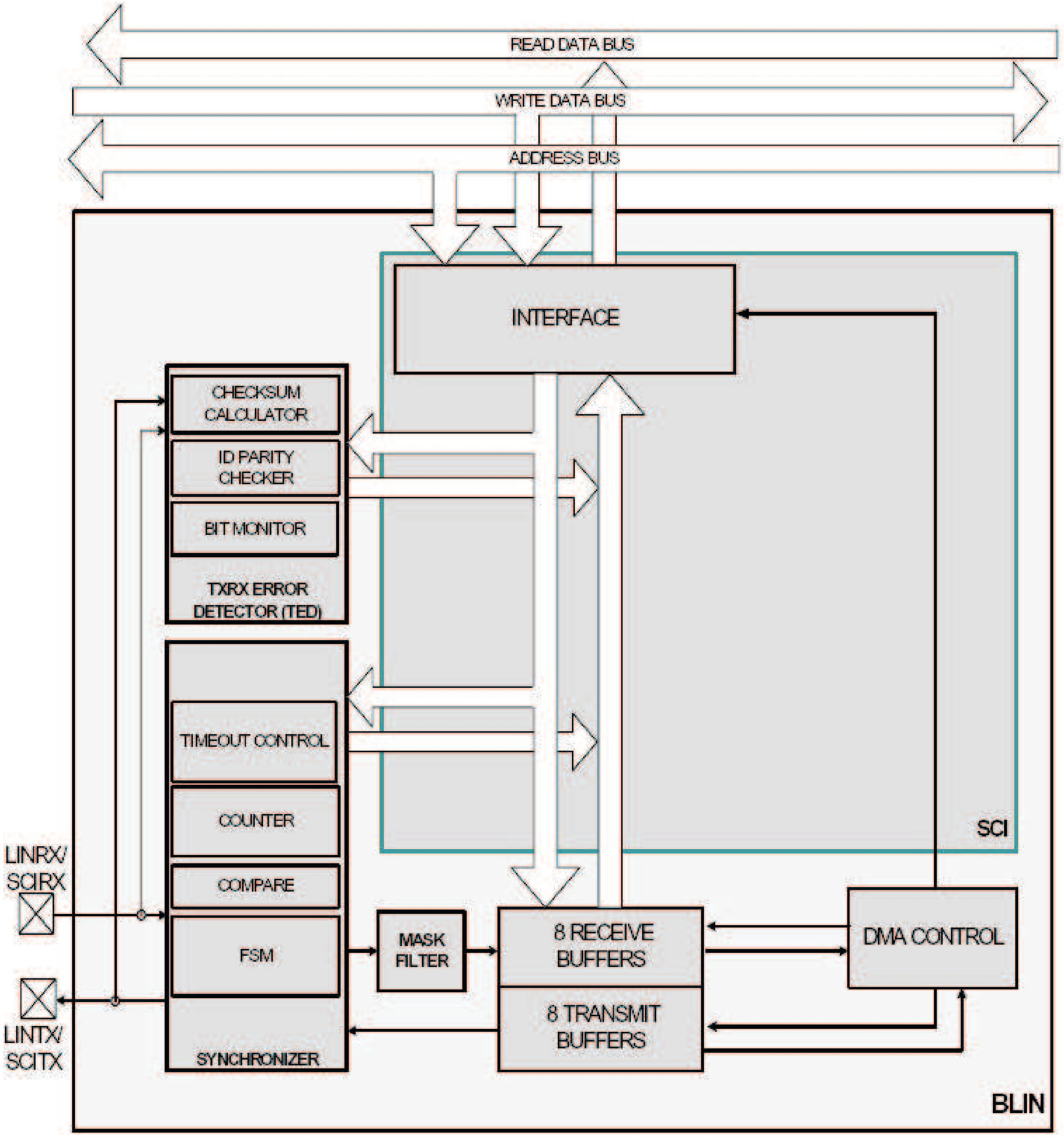


Figure 29-2. SCI/LIN Block Diagram



29.2 SCI

29.2.1 SCI Communication Formats

The SCI module can be configured to meet the requirements of many applications. Because communication formats vary depending on the specific application, many attributes of the SCI/LIN are user configurable. The list below describes these configuration options:

- SCI Frame format
- SCI Timing modes
- SCI Baud rate
- SCI Multiprocessor modes

29.2.1.1 SCI Frame Formats

The SCI uses a programmable frame format. All frames consist of the following:

- One start bit
- One to eight data bits
- Zero or one address bit
- Zero or one parity bit
- One or two stop bits

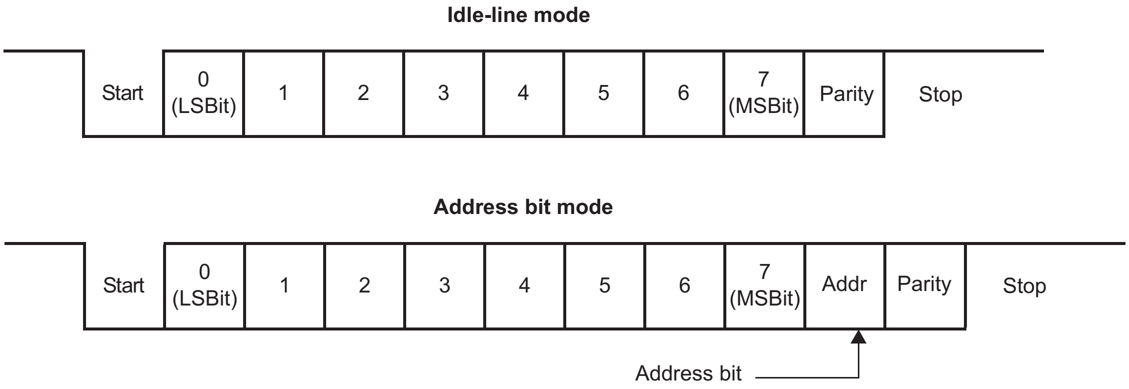
The frame format for both the transmitter and receiver is programmable through the bits in the SCIGCR1 register. Both receive and transmit data is in nonreturn to zero (NRZ) format, which means that the transmit and receive lines are at logic high when idle. Each frame transmission begins with a start bit, in which the transmitter pulls the SCI line low (logic low). Following the start bit, the frame data is sent and received least significant bit first (LSB).

An address bit is present in each frame if the SCI is configured to be in address-bit mode but is not present in any frame if the SCI is configured for idle-line mode. The format of frames with and without the address bit is illustrated in Figure 29-3.

A parity bit is present in every frame when the PARITY ENA bit is set. The value of the parity bit depends on the number of one bits in the frame and whether odd or even parity has been selected via the PARITY ENA bit. Both examples in Figure 29-3 have parity enabled.

All frames include one stop bit, which is always a high level. This high level at the end of each frame is used to indicate the end of a frame to ensure synchronization between communicating devices. Two stop bits are transmitted if the STOP bit in SCIGCR1 register is set. The examples shown in Figure 29-3 use one stop bit per frame.

Figure 29-3. Typical SCI Data Frame Formats



29.2.1.2 SCI Timing Mode

The SCI can be configured to use asynchronous or isosynchronous timing using TIMING MODE bit in SCIGCR1 register.

29.2.1.2.1 Asynchronous Timing Mode

The asynchronous timing mode uses only the receive and transmit data lines to interface with devices using the standard universal asynchronous receiver- transmitter (UART) protocol.

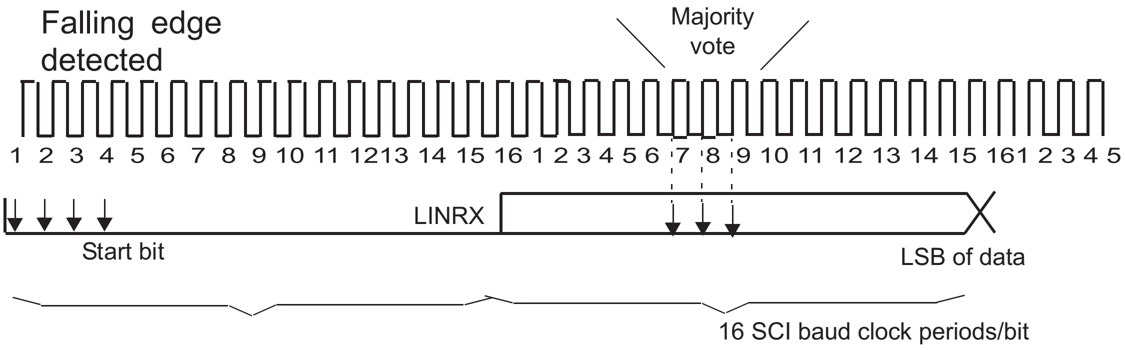
In the asynchronous timing mode, each bit in a frame has a duration of 16 SCI baud clock periods. Each bit therefore consists of 16 samples (one for each clock period). When the SCI is using asynchronous mode, the baud rates of all communicating devices must match as closely as possible. Receive errors result from devices communicating at different baud rates.

With the receiver in the asynchronous timing mode, the SCI detects a valid start bit if the first four samples after a falling edge on the LINRX pin are of logic level 0. As soon as a falling edge is detected on LINRX, the SCI assumes that a frame is being received and synchronizes itself to the bus.

To prevent interpreting noise as Start bit SCI expects LINRX line to be low for at least four contiguous SCI baud clock periods to detect a valid start bit. The bus is considered idle if this condition is not met. When a valid start bit is detected, the SCI determines the value of each bit by sampling the LINRX line value during the seventh, eighth, and ninth SCI baud clock periods. A majority vote of these three samples is used to determine the value stored in the SCI receiver shift register. By sampling in the middle of the bit, the SCI reduces errors caused by propagation delays and rise and fall times and data line noises. Figure 29-4 illustrates how the receiver samples a start bit and a data bit in asynchronous timing mode.

The transmitter transmits each bit for a duration of 16 SCI baud clock periods. During the first clock period for a bit, the transmitter shifts the value of that bit onto the LINTX pin. The transmitter then holds the current bit value on LINTX for 16 SCI baud clock periods.

Figure 29-4. Asynchronous Communication Bit Timing



29.2.1.2.2 Isosynchronous Timing Mode

In isosynchronous timing mode, each bit in a frame has a duration of exactly 1 baud clock period and therefore consists of a single sample. With this timing configuration, the transmitter and receiver are required to make use of the SCICLK pin to synchronize communication with other SCI. **This mode is not supported on this device because SCICLK pin is not available.**

29.2.1.3 SCI Baud Rate

The SCI/LIN has an internally generated serial clock determined by the peripheral VCLK and the prescalers P and M in this register. The SCI uses the 24-bit integer prescaler P value in the BRS register to select the required baud rates. The additional 4-bit fractional divider M refines the baud rate selection.

In asynchronous timing mode, the SCI generates a baud clock according to the following formula:

$$\textit{SCICLK Frequency} = \frac{\text{VCLK Frequency}}{\text{P} + 1 + \frac{\text{M}}{16}}$$

$$\textit{Asynchronous baud value} = \frac{\text{SCICLK Frequency}}{16}$$

For P = 0,

$$\textit{Asynchronous baud value} = \frac{\text{VCLK Frequency}}{32} \tag{42}$$

29.2.1.3.1 Superfractional Divider, SCI Asynchronous Mode

The superfractional divider is available in SCI asynchronous mode (idle-line and address-bit mode). Building on the 4-bit fractional divider M (BRS[27:24]), the superfractional divider uses an additional 3-bit modulating value (see Table 29-2). The bits with a 1 in the table will have an additional VCLK period added to their T_{bit}. If the character length is more than 10, then the modulation table will be a rolled-over version of the original table (Table 29-1), as shown in Table 29-2.

The baud rate will vary over a data field to average according to the BRS[30:28] value by a "**d**" fraction of the peripheral internal clock: **0<d<1.** See Figure 29-5 for a simple Average "d' calculation based on "U" value (BRS[30:28]).

The instantaneous bit time is expressed in terms of T_{VCLK} as follows:

For all P other than 0, and all M and d (0 or 1),

$$T^{i}bit = \left[16\left(P+1+\frac{M}{16}\right)+d\right]T_{VCLK} \tag{43}$$

For P = 0 $T_{bit} = 32T_{VCLK}$

The averaged bit time is expressed in terms of T_{VCLK} as follows:

For all P other than 0, and all M and d (0<d<1),

$$T^{a}bit = \left[16\left(P+1+\frac{M}{16}\right)+d\right]T_{VCLK} \tag{44}$$

For P = 0 $T_{bit} = 32T_{VCLK}$

Table 29-1. Superfractional Bit Modulation for SCI Mode (Normal Configuration) [1]

BRS[30:28]	Start Bit	D[0]	D[1]	D[2]	D[3]	D[4]	D[5]	D[6]	D[7]	Stop Bit
0h	0	0	0	0	0	0	0	0	0	0
1h	1	0	0	0	0	0	0	0	1	0
2h	1	0	0	0	1	0	0	0	1	0
3h	1	0	1	0	1	0	0	0	1	0
4h	1	0	1	0	1	0	1	0	1	0
5h	1	1	1	0	1	0	1	0	1	1
6h	1	1	1	0	1	1	1	0	1	1
7h	1	1	1	1	1	1	1	0	1	1

[1] Normal configuration = Start + 8 Data Bits + Stop Bit

Table 29-2. Superfractional Bit Modulation for SCI Mode (Maximum Configuration) [1]

BRS[30:28]	Start Bit	D[0]	D[1]	D[2]	D[3]	D[4]	D[5]	D[6]	D[7]	Addr	Parity	Stop0	Stop1
0h	0	0	0	0	0	0	0	0	0	0	0	0	0
1h	1	0	0	0	0	0	0	0	1	0	0	0	0
2h	1	0	0	0	1	0	0	0	1	0	0	0	1
3h	1	0	1	0	1	0	0	0	1	0	1	0	1
4h	1	0	1	0	1	0	1	0	1	0	1	0	1
5h	1	1	1	0	1	0	1	0	1	1	1	0	1
6h	1	1	1	0	1	1	1	0	1	1	1	0	1
7h	1	1	1	1	1	1	1	0	1	1	1	1	1

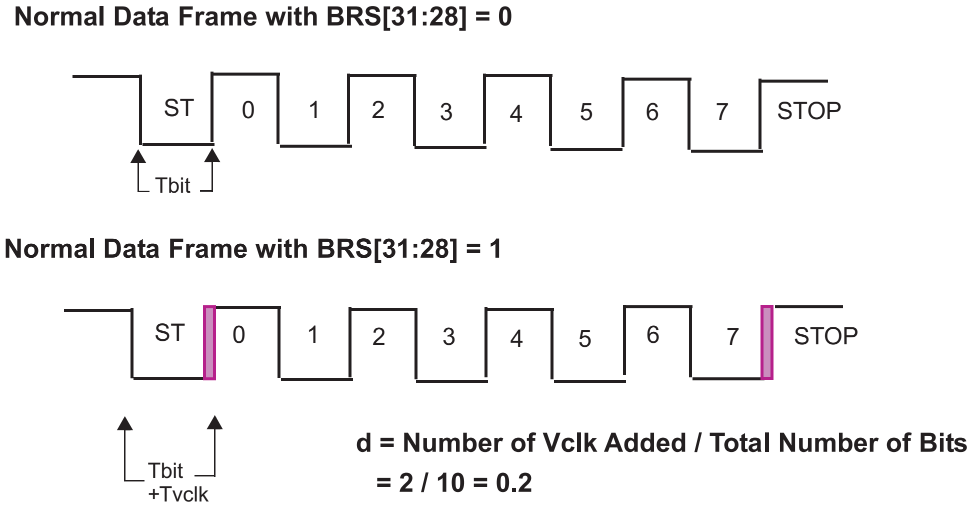
[1] Maximum configuration = Start + 8 Data Bits + Addr Bit + Parity Bit + Stop Bit 0 + Stop Bit 1

Table 29-3. SCI Mode (Minimum Configuration) [1]

BRS[30:28]	Start Bit	D[0]	Stop
0h	0	0	0
1h	1	0	0
2h	1	0	0
3h	1	0	1
4h	1	0	1
5h	1	1	1
6h	1	1	1
7h	1	1	1

[1] Minimum configuration = Start + 1 Data Bits + Stop Bit

Figure 29-5. Superfractional Divider Example


Normal Data Frame with BRS[31:28] = 0
ST
0
1
2
3
4
5
6
7
STOP
Tbit
Normal Data Frame with BRS[31:28] = 1
ST
0
1
2
3
4
5
6
7
STOP
Tbit
+Tvclk
d = Number of Vclk Added / Total Number of Bits
= 2 / 10 = 0.2

29.2.1.4 SCI Multiprocessor Communication Modes

In some applications, the SCI may be connected to more than one serial communication device. In such a multiprocessor configuration, several frames of data may be sent to all connected devices or to an individual device. In the case of data sent to an individual device, the receiving devices must determine when they are being addressed. When a message is not intended for them, the devices can ignore the following data. When only two devices make up the SCI network, addressing is not needed, so multiprocessor communication schemes are not required.

SCI supports two multiprocessor communication modes which can be selected using COMM MODE bit:

- Idle-Line Mode
- Address Bit Mode

When the SCI is not used in a multiprocessor environment, software can consider all frames as data frames. In this case, the only distinction between the idle-line and address-bit modes is the presence of an extra bit (the address bit) in each frame sent with the address-bit protocol.

The SCI allows full-duplex communication where data can be sent and received via the transmit and receive pins simultaneously. However, the protocol used by the SCI assumes that only one device transmits data on the same bus line at any one time. No arbitration is done by the SCI.

29.2.1.4.1 Idle-Line Multiprocessor Modes

In idle-line multiprocessor mode, a frame that is preceded by an idle period (10 or more idle bits) is an address frame. A frame that is preceded by fewer than 10 idle bits is a data frame. Figure 29-6 illustrates the format of several blocks and frames with idle-line mode.

There are two ways to transmit an address frame using idle-line mode:

Method 1: In software, deliberately leave an idle period between the transmission of the last data frame of the previous block and the address frame of the new block.

Method 2: Configure the SCI to automatically send an idle period between the last data frame of the previous block and the address frame of the new block.

Although Method 1 is only accomplished by a delay loop in software, Method 2 can be implemented by using the transmit buffer and the TXWAKE bit in the following manner:

Step 1: Write a 1 to the TXWAKE bit.

Step 2: Write a dummy data value to the SCITD register. This triggers the SCI to begin the idle period as soon as the transmitter shift register is empty.

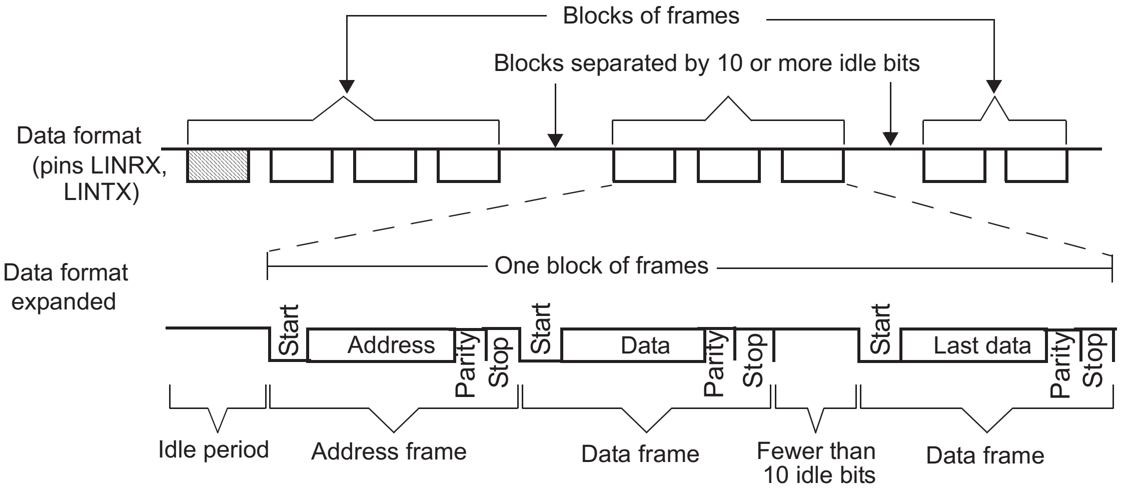
Step 3: Wait for the SCI to clear the TXWAKE flag.

Step 4: Write the address value to SCITD.

As indicated by Step 3, software should wait for the SCI to clear the TXWAKE bit. However, the SCI clears the TXWAKE bit at the same time it sets TXRDY (that is, transfers data from SCITD into SCITXSHF). Therefore, if the TX INT ENA bit is set, the transfer of data from SCITD to SCITXSHF causes an interrupt to be generated at the same time that the SCI clears the TXWAKE bit. If this interrupt method is used, software is not required to poll the TXWAKE bit waiting for the SCI to clear it.

When idle-line multiprocessor communications are used, software must ensure that the idle time exceeds 10 bit periods before addresses (using one of the methods mentioned above), and software must also ensure that data frames are written to the transmitter quickly enough to be sent without a delay of 10 bit periods between frames. Failure to comply with these conditions will result in data interpretation errors by other devices receiving the transmission.

Figure 29-6. Idle-Line Multiprocessor Communication Format



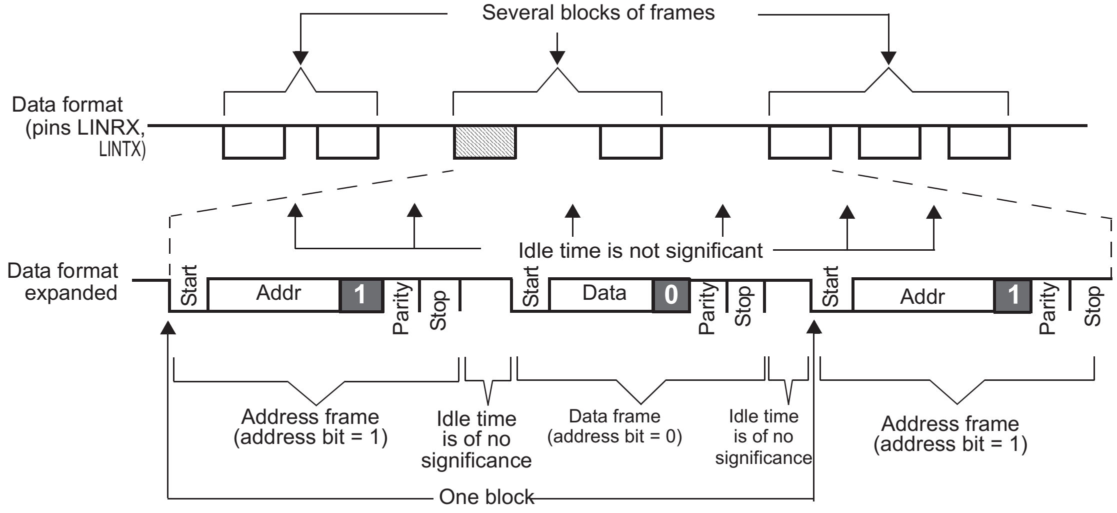
29.2.1.4.2 Address-Bit Multiprocessor Mode

In the address-bit protocol, each frame has an extra bit immediately following the data field called an address bit. A frame with the address bit set to 1 is an address frame; a frame with the address bit set to 0 is a data frame. The idle period timing is irrelevant in this mode. Figure 29-7 illustrates the format of several blocks and frames with the address-bit mode.

When address-bit mode is used, the value of the TXWAKE bit is the value sent as the address bit. To send an address frame, software must set the TXWAKE bit. This bit is cleared as the contents of the SCITD are shifted from the TXWAKE register so that all frames sent are data except when the TXWAKE bit is written as a 1.

No dummy write to SCITD is required before an address frame is sent in address-bit mode. The first byte written to SCITD after the TXWAKE bit is written to 1 is transmitted with the address bit set when address-bit mode is used.

Figure 29-7. Address-Bit Multiprocessor Communication Format



29.2.1.5 SCI Multi Buffered Mode

To reduce CPU load when Receiving or Transmitting data in interrupt mode or DMA mode, the SCI/LIN module has eight separate Receive and transmit buffers. Multi buffered mode is enabled by setting the MBUF MODE bit.

The multi-buffer 3-bit counter counts the data bytes transferred from the SCIRXSHF register to the RDy receive buffers and TDy transmit buffers register to SCITXSHF register. The 3-bit compare register contains the number of data bytes expected to be received or transmitted. the LENGTH value in SCIFORMAT register indicates the expected length and is used to load the 3-bit compare register.

A receive interrupt (RX interrupt; see the SCIINTVECT0 and SCIINTVECT1registers), and a receive ready RXRDY flag set in SCIFLR register, as well as a DMA request (RXDMA) could occur after receiving a response if there are no response receive errors for the frame (such as, there is, frame error, and overrun error).

A transmit interrupt (TX interrupt), and a transmit ready flag (TXRDY flag in SCIFLR register), and a DMA request (TXDMA) could occur after transmitting a response.

Figure 29-8 and Figure 29-9 shows the receive and transmit multi-buffer functional block diagram.

Figure 29-8. Receive Buffers

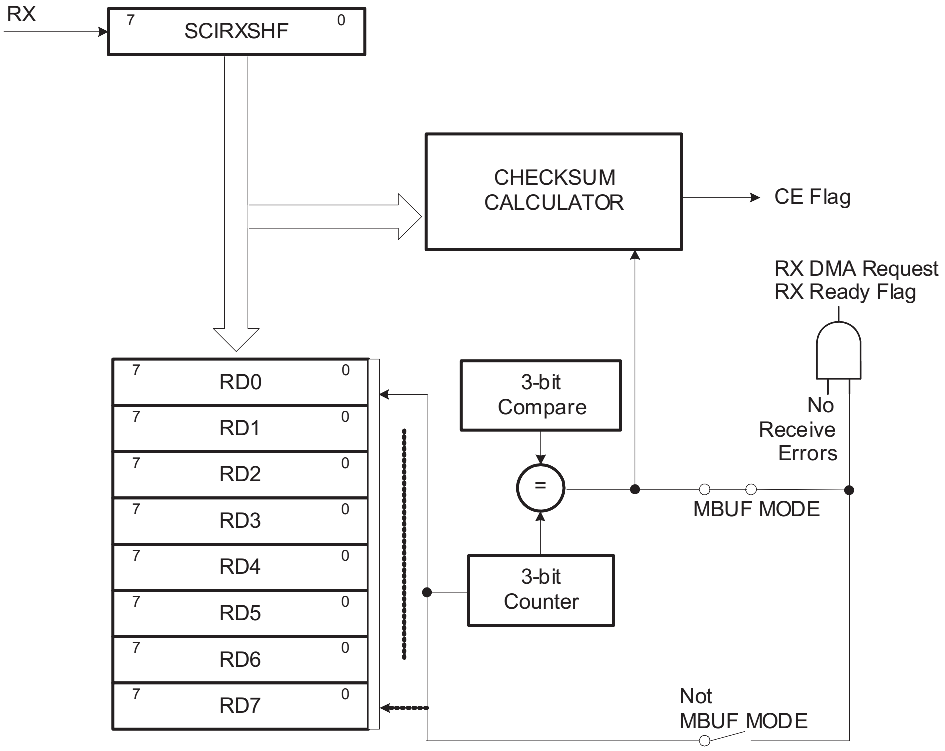
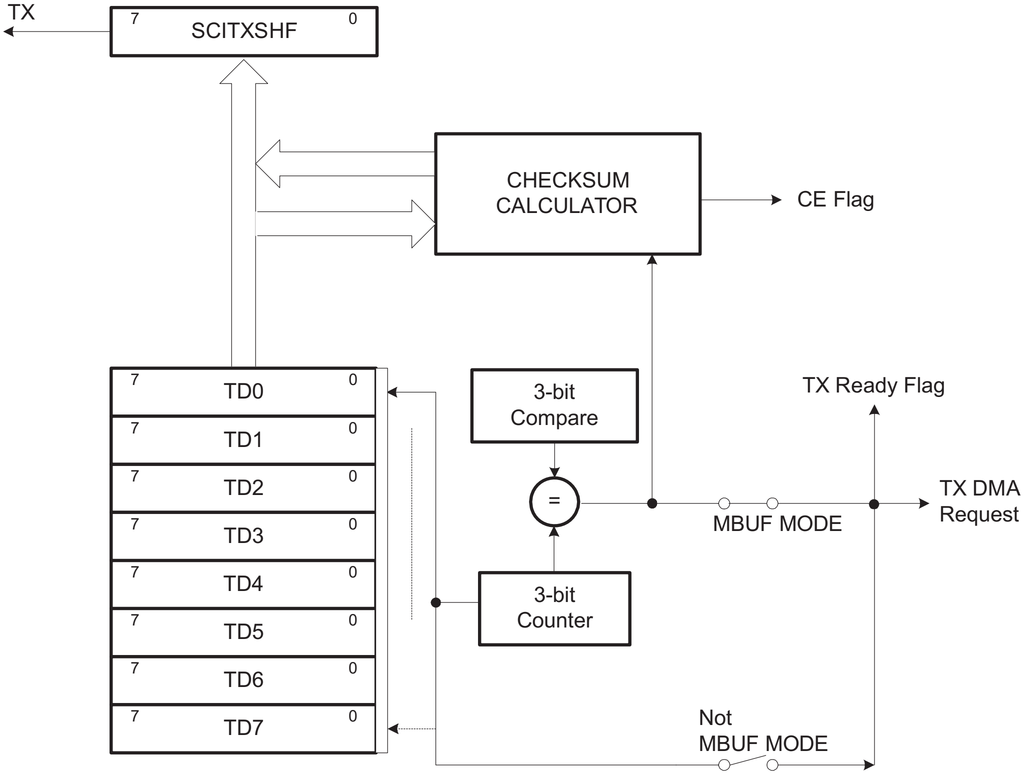


Figure 29-9. Transmit Buffers



29.2.2 SCI Interrupts

The SCI/LIN module has two interrupt lines, level 0 and level 1, to the vectored interrupt manager (VIM) module (see Figure 29-10). Two offset registers SCIINTVECT0 and SCIINTVECT1 determine which flag triggered the interrupt according to the respective priority encoders. Each interrupt condition has a bit to enable/disable the interrupt in the SCISETINT and SCICLRINT registers respectively.

Each interrupt also has a bit that can be set as interrupt level 0(INT0) or as interrupt level 1(INT1). By default, interrupts are in interrupt level 0. SCISETINTLVL sets a given interrupt to level1. SCICLEARINTLVL resets a given interrupt level to the default level 0.

The interrupt vector registers SCIINTVECT0 and SCIINTVECT1 return the vector of the pending interrupt line INT0 or INT1. If more than one interrupt is pending, the interrupt vector register holds the highest priority interrupt.

Figure 29-10. General Interrupt Scheme

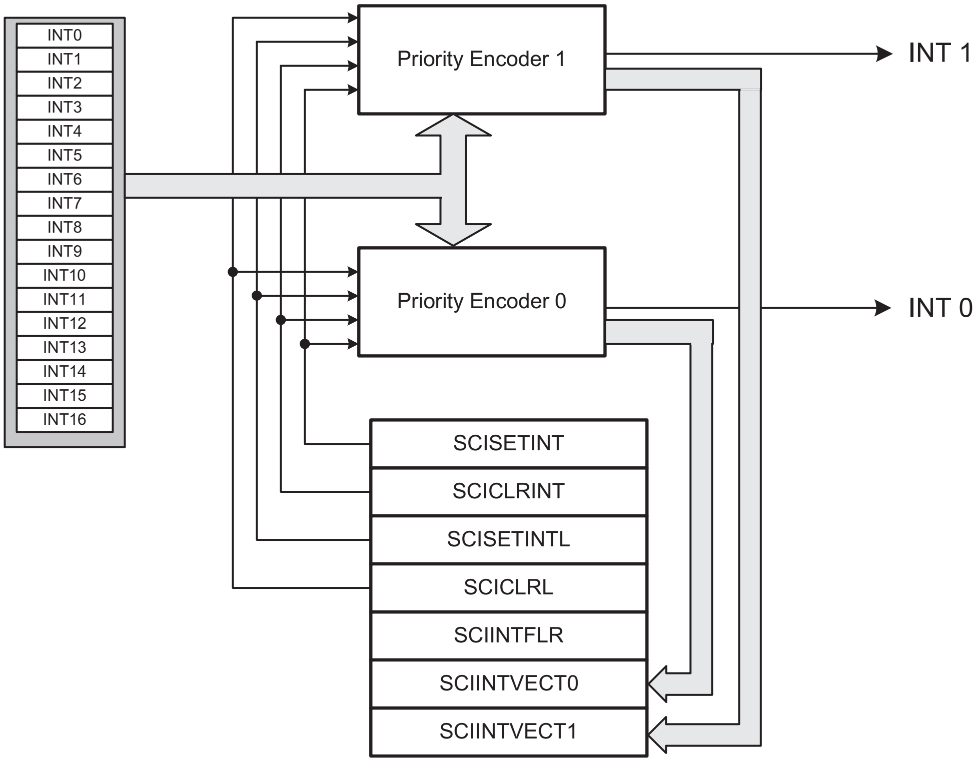
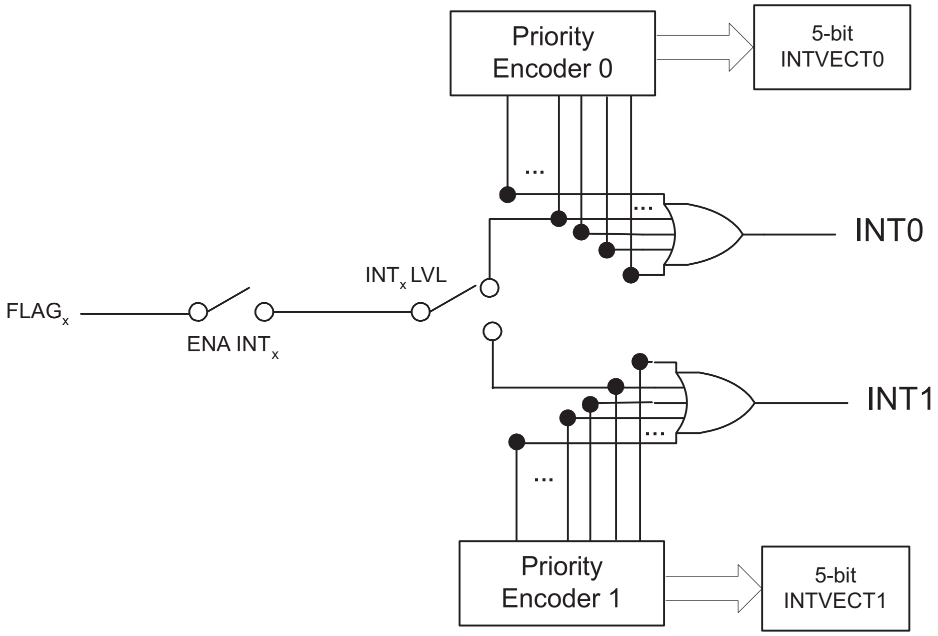


Figure 29-11. Interrupt Generation for Given Flags



29.2.2.1 Transmit Interrupt

To use transmit interrupt functionality, SET TX INT bit must be enabled and SET TX DMA bit must be cleared. The transmit ready (TXRDY) flag is set when the SCI transfers the contents of SCITD to the shift register, SCITXSHF. The TXRDY flag indicates that SCITD is ready to be loaded with more data. In addition, the SCI sets the TX EMPTY bit if both the SCITD and SCITXSHF registers are empty. If the SET TX INT bit is set, then a transmit interrupt is generated when the TXRDY flag goes high. Transmit Interrupt is not generated immediately after setting the SET TX INT bit unlike transmit DMA request. Transmit Interrupt is generated only after the first transfer from SCITD to SCITXSHF, that is first data has to be written to SCITD by you before any interrupt gets generated. To transmit further data, you can write data to SCITD in the transmit Interrupt service routine.

Writing data to the SCITD register clears the TXRDY bit. When this data has been moved to the SCITXSHF register, the TXRDY bit is set again. The interrupt request can be suspended by setting the CLR TX INT bit; however, when the SET TX INT bit is again set to 1, the TXRDY interrupt is asserted again. The transmit interrupt request can be eliminated until the next series of values is written to SCITD, by disabling the transmitter via the TXENA bit, by a software reset SWnRST, or by a device hardware reset.

29.2.2.2 Receive Interrupt

The receive ready (RXRDY) flag is set when the SCI transfers newly received data from SCIRXSHF to SCIRD. The RXRDY flag therefore indicates that the SCI has new data to be read. Receive interrupts are enabled by the SET RX INT bit. If the SET RX INT is set when the SCI sets the RXRDY flag, then a receive interrupt is generated. The received data can be read in the Interrupt Service routine.

On a device with both SCI and a DMA controller, the bits SET RX DMA ALL and SET RX DMA must be cleared to select interrupt functionality.

29.2.2.3 WakeUp Interrupt

SCI sets the WAKEUP flag if bus activity on the RX line either prevents power-down mode from being entered, or RX line activity causes an exit from power-down mode. If enabled (SET WAKEUP INT), wakeup interrupt is triggered once WAKEUP flag is set.

29.2.2.4 Error Interrupts

The following error detection are supported with Interrupt by the SCI module:

- Parity errors (PE)
- Frame errors (FE)
- Break Detect errors (BRKDT)
- Overrun errors (OE)
- Bit errors (BE)

If all of these errors (PE, FE, BRKDT, OE, BE) are flagged, an interrupt for the flagged errors will be generated if enabled. A message is valid for both the transmitter and the receiver if there is no error detected until the end of the frame. Each of these flags is located in the receiver status (SCIFLR) register.

There are 16 interrupt sources in the SCI/LIN module, In SCI mode 8 interrupts are supported, as seen in Table 29-4.

Table 29-4. SCI/LIN Interrupts

Offset [1]	Interrupt	Applicable to SCI	Applicable to LIN
0	No interrupt		
1	Wakeup	Yes	Yes
2	Inconsistent-synch-field error	No	Yes
3	Parity error	Yes	Yes
4	ID	No	Yes
5	Physical bus error	No	Yes
6	Frame error	Yes	Yes
7	Break detect	Yes	No
8	Checksum error	No	Yes
9	Overrun error	Yes	Yes
10	Bit error	Yes	Yes
11	Receive	Yes	Yes
12	Transmit	Yes	Yes
13	No-response error	No	Yes
14	Timeout after wakeup signal (150 ms)	No	Yes
15	Timeout after three wakeup signals (1.5 s)	No	Yes
16	Timeout (Bus Idle, 4s)	No	Yes

[1] Offset 1 is the highest priority. Offset 16 is the lowest priority.

29.2.3 SCI DMA Interface

DMA requests for receive (RXDMA request) and transmit (TXDMA request) are available for the SCI/LIN module. The DMA transfers depending on whether multi-buffer mode bit (MBUF MODE) is enabled or not enabled. See Chapter 30 for more information.

29.2.3.1 Receive DMA Requests

This DMA functionality is enabled/disabled by the CPU using the SET RX DMA/CLR RX DMA bits, respectively.

In Multi-Buffered SCI mode with DMA enabled, the receiver loads the RDy buffers for each received character. RX DMA request is triggered once the last character of the programmed number of characters (LENGTH) are received and copied to the corresponding RDy buffer successfully.

If the multi-buffer option is disabled, then DMA requests will be generated on a byte-per-byte basis.

In multiprocessor mode, the SCI can generate receiver interrupts for address frames and DMA requests for data frames or DMA requests for both. This is controlled by the SET RX DMA ALL bit.

In multiprocessor mode with the SLEEP bit set, no DMA is generated for received data frames. The software must clear the SLEEP bit before data frames can be received.

29.2.3.2 Transmit DMA Requests

DMA functionality is enabled/disabled by the CPU with SET TX DMA/CLR TX DMA bits, respectively.

In Multi-Buffered SCI mode once TXRDY bit is set or after a transmission of programmed number of characters (LENGTH) (up to eight data bytes stored in the transmit buffer(s) TDy in the LINTD0 and LINTD1 registers), a DMA request is generated in order to reload the transmit buffer for the next transmission. If the multi-buffer option is disabled, then DMA requests will be generated on a byte-per-byte basis.

29.2.4 SCI Configurations

Before the SCI sends or receives data, its registers should be properly configured. Upon power-up or a system-level reset, each bit in the SCI registers is set to a default state. The registers are writable only after the RESET bit in the SCIGCR0 register is set to 1. Of particular importance is the SWnRST bit in the SCIGCR1 register. The SWnRST is an active-low bit initialized to 0 and keeps the SCI in a reset state until it is programmed to 1. Therefore, all SCI configuration should be completed before a 1 is written to the SWnRST bit.

The following list details the configuration steps that software should perform prior to the transmission or reception of data. As long as the SWnRST bit is cleared to 0 the entire time that the SCI is being configured, the order in which the registers are programmed is not important.

- Enable SCI by setting the RESET bit to 1.
- Clear the SWnRST bit to 0 before SCI is configured.
- Select the desired frame format by programming the SCIGCR1 register.
- Set both the RX FUNC and TX FUNC bits in SCIPIO0 to 1 to configure the LINRX and LINTX pins for SCI functionality.
- Select the baud rate to be used for communication by programming the BRS register.
- Set the CLOCK bit in SCIGCR1 to 1 to select the internal clock.
- Set the CONT bit in SCIGCR1 to 1 to make SCI not halt for an emulation breakpoint until its current reception or transmission is complete (this bit is used only in an emulation environment).
- Set the LOOP BACK bit in SCIGCR1 to 1 to connect the transmitter to the receiver internally (this feature is used to perform a self-test).
- Set the RXENA bit in SCIGCR1 to 1, if data is to be received.
- Set the TXENA bitin SCIGCR1 to 1, if data is to be transmitted.
- Set the SWnRST bit to 1 after SCI is configured.
- Perform receiving or transmitting data (see Section 29.2.4.1 or Section 29.2.4.2).

29.2.4.1 Receiving Data

The SCI receiver is enabled to receive messages if both the RX FUNC bit and the RXENA bit are set to 1. If the RX FUNC bit is not set, the LINRX pin functions as a general-purpose I/O pin rather than as an SCI function pin.

SCI module can receive data in one of the following modes:

- Single-Buffer (Normal) Mode
- Multi-Buffer Mode

After a valid idle period is detected, data is automatically received as it arrives on the LINRX pin.

29.2.4.1.1 Receiving Data in Single-Buffer Mode

Single-buffer mode is selected when the MBUF MODE bit in SCIGCR1 is cleared to 0. In this mode, SCI sets the RXRDY bit when it transfers newly received data from SCIRXSHF to SCIRD. The SCI clears the RXRDY bit after the new data in SCIRD has been read. Also, as data is transferred from SCIRXSHF to SCIRD, the SCI sets the FE, OE, or PE flags if any of these error conditions were detected in the received data. These error conditions are supported with configurable interrupt capability. The wake-up and break-detect status bits are also set if one of these errors occurs, but they do not necessarily occur at the same time that new data is being loaded into SCIRD.

You can receive data by:

1. Polling Receive Ready Flag
2. Receive Interrupt
3. DMA

In polling method, software can poll for the RXRDY bit and read the data from the SCIRD register once the RXRDY bit is set high. The CPU is unnecessarily overloaded by selecting the polling method. To avoid this, you can use either the interrupt or DMA method. To use the interrupt method, the SET RX INT bit is set. To use the DMA method, the SET RX DMA bit is set. Either an interrupt or a DMA request is generated the moment the RXRDY bit is set.

29.2.4.1.2 Receiving Data in Multi-Buffer Mode

Multi-buffer mode is selected when the MBUF MODE bit in SCIGCR1 is set to 1. In this mode, SCI sets the RXRDY bit after receiving the programmed number of data in the receive buffer, the complete frame. The error condition detection logic is similar to the single-buffer mode, except that it monitors for the complete frame. Like single-buffer mode, you can use the polling, interrupt, or DMA method to read the data. The SCI clears the RXRDY bit after the new data in SCIRD has been read.

29.2.4.2 Transmitting Data

The SCI transmitter is enabled if both the TX FUNC bit and the TXENA bit are set to 1. If the TX FUNC bit is not set, the LINTX pin functions as a general-purpose I/O pin rather than as an SCI function pin. Any value written to the SCITD before TXENA is set to 1 is not transmitted. Both of these control bits allow for the SCI transmitter to be held inactive independently of the receiver.

SCI module can transmit data in one of the following modes:

- Single-Buffer (Normal) Mode
- Multi-Buffered or Buffered SCI Mode

29.2.4.2.1 Transmitting Data in Single-Buffer Mode

Single-buffer mode is selected when the MBUF MODE bit in SCIGCR1 is cleared to 0. In this mode, SCI waits for data to be written to SCITD, transfers it to SCITXSHF, and transmits the data. The TXRDY and TX EMPTY bits indicate the status of the transmit buffers. That is, when the transmitter is ready for data to be written to SCITD, the TXRDY bit is set. Additionally, if both SCITD and SCITXSHF are empty, then the TX EMPTY bit is also set.

You can transmit data by:

1. Polling Transmit Ready Flag
2. Transmit Interrupt
3. DMA

In polling method, software can poll for the TXRDY bit to go high before writing the data to the SCITD register. The CPU is unnecessarily overloaded by selecting the polling method. To avoid this, you can use either the interrupt or DMA method. To use the interrupt method, the SET TX INT bit is set. To use the DMA method, the SET TX DMA bit is set. Either an interrupt or a DMA request is generated the moment the TXRDY bit is set. When the SCI has completed transmission of all pending frames, the SCITXSHF register and SCITD are empty, the TXRDY bit is set, and an interrupt/DMA request is generated, if enabled. Because all data has been transmitted, the interrupt/DMA request should be halted. This can either be done by disabling the transmit interrupt (CLR TX INT) / DMA request (CLR TX DMA bit) or by disabling the transmitter (clear TXENA bit).

NOTE: The TXRDY flag cannot be cleared by reading the corresponding interrupt offset in the SCIINTVECT0 or SCIINTVECT1 register.

29.2.4.2.2 Transmitting Data in Multi-Buffer Mode

Multi-buffer mode is selected when the MBUF MODE bit in SCIGCR1 is set to 1. Like single-buffer mode, you can use the polling, interrupt, or DMA method to write the data to be transmitted. The transmitted data has to be written to the SCITD registers. SCI waits for data to be written to the SCITD register and transfers the programmed number of bytes to SCITXSHF to transmit one by one automatically.

29.2.5 SCI Low-Power Mode

The SCI/LIN can be put in either local or global low-power mode. Global low-power mode is asserted by the system and is not controlled by the SCI/LIN. During global low-power mode, all clocks to the SCI/LIN are turned off so the module is completely inactive.

Local low-power mode is asserted by setting the POWERDOWN bit; setting this bit stops the clocks to the SCI/LIN internal logic and the module registers. Setting the POWERDOWN bit causes the SCI to enter local low-power mode and clearing the POWERDOWN bit causes SCI/LIN to exit from local low-power mode. All the registers are accessible during local power-down mode as any register access enables the clock to SCI for that particular access alone.

The wake-up interrupt is used to allow the SCI to exit low-power mode automatically when a low level is detected on the LINRX pin and also this clears the POWERDOWN bit. If wake-up interrupt is disabled, then the SCI/LIN immediately enters low-power mode whenever it is requested and also any activity on the LINRX pin does not cause the SCI to exit low-power mode.

NOTE: Enabling Local Low-Power Mode During Receive and Transmit

If the wake-up interrupt is enabled and low-power mode is requested while the receiver is receiving data, then the SCI immediately generates a wake-up interrupt to clear the powerdown bit and prevents the SCI from entering low-power mode and thus completes the current reception. Otherwise, if the wake-up interrupt is disabled, then the SCI completes the current reception and then enters the low-power mode.

29.2.5.1 Sleep Mode for Multiprocessor Communication

When the SCI receives data and transfers that data from SCIRXSHF to SCIRD, the RXRDY bit is set and if RX INT ENA is set, the SCI also generates an interrupt. The interrupt triggers the CPU to read the newly received frame before another one is received. In multiprocessor communication modes, this default behavior may be enhanced to provide selective indication of new data. When SCI receives an address frame that does not match its address, the device can ignore the data following this non-matching address until the next address frame by using sleep mode. Sleep mode can be used with both idle-line and address-bit multiprocessor modes.

If sleep mode is enabled by the SLEEP bit, then the SCI transfers data from SCIRXSHF to SCIRD only for address frames. Therefore, in sleep mode, all data frames are assembled in the SCIRXSHF register without being shifted into the SCIRD and without initiating a receive interrupt or DMA request. Upon reception of an address frame, the contents of the SCIRXSHF are moved into SCIRD, and the software must read SCIRD and determine if the SCI is being addressed by comparing the received address against the address previously set in the software and stored somewhere in memory (the SCI does not have hardware available for address comparison). If the SCI is being addressed, the software must clear the SLEEP bit so that the SCI will load SCIRD with the data of the data frames that follow the address frame.

When the SCI has been addressed and sleep mode has been disabled (in software) to allow the receipt of data, the SCI should check the RXWAKE bit (SCIFLR.12) to determine when the next address has been received. This bit is set to 1 if the current value in SCIRD is an address and set to 0 if SCIRD contains data. If the RXWAKE bit is set, then software should check the address in SCIRD against its own address. If it is still being addressed, then sleep mode should remain disabled. Otherwise, the SLEEP bit should be set again.

Following is a sequence of events typical of sleep mode operation:

- The SCI is configured and both sleep mode and receive actions are enabled.
- An address frame is received and a receive interrupt is generated.
- Software compares the received address frame against that set by software and determines that the SCI is not being addressed, so the value of the SLEEP bit is not changed.
- Several data frames are shifted into SCIRXSHF, but no data is moved to SCIRD and no receive interrupts are generated.
- A new address frame is received and a receive interrupt is generated.
- Software compares the received address frame against that set by software and determines that the SCI is being addressed and clears the SLEEP bit.
- Data shifted into SCIRXSHF is transferred to SCIRD, and a receive interrupt is generated after each data frame is received.
- In each interrupt routine, software checks RXWAKE to determine if the current frame is an address frame.
- Another address frame is received, RXWAKE is set, software determines that the SCI is not being addressed and sets the SLEEP bit back to 1. No receive interrupts are generated for the data frames following this address frame.

By ignoring data frames that are not intended for the device, fewer interrupts are generated. These interrupts would otherwise require CPU intervention to read data that is of no significance to this specific device. Using sleep mode can help free some CPU resources.

Except for the RXRDY flag, the SCI continues to update the receiver status flags (see Table 29-13) while sleep mode is active. In this way, if an error occurs on the receive line, an application can immediately respond to the error and take the appropriate corrective action.

Because the RXRDY bit is not updated for data frames when sleep mode is enabled, the SCI can enable sleep mode and use a polling algorithm if desired. In this case, when RXRDY is set, software knows that a new address has been received. If the SCI is not being addressed, then the software should not change the value of the SLEEP bit and should continue to poll RXRDY.

29.3 LIN

29.3.1 LIN Communication Formats

The SCI/LIN module can be used in LIN mode or SCI mode. The enhancements for baud generation, DMA controls and additional receive/transmit buffers necessary for LIN mode operation are also part of the enhanced buffered SCI module. LIN mode is selected by enabling LIN MODE bit in SCIGCR1 register.

NOTE: The SCI/LIN is built around the SCI platform and uses a similar sampling scheme: 16 samples for each bit with majority vote on samples 8, 9, and 10.

The SCI/LIN control registers are located at the SCI/LIN base address. For a detailed description of each register, see Section 29.7.

29.3.1.1 LIN Standards

For compatibility with LIN2.0 standard the following additional features are implemented over LIN1.3:

i. Support for LIN 2.0 checksum
ii. Enhanced synchronizer FSM support for frame processing
iii. Enhanced handling of extended frames
iv. Enhanced baud rate generator
v. Update wakeup/go to sleep

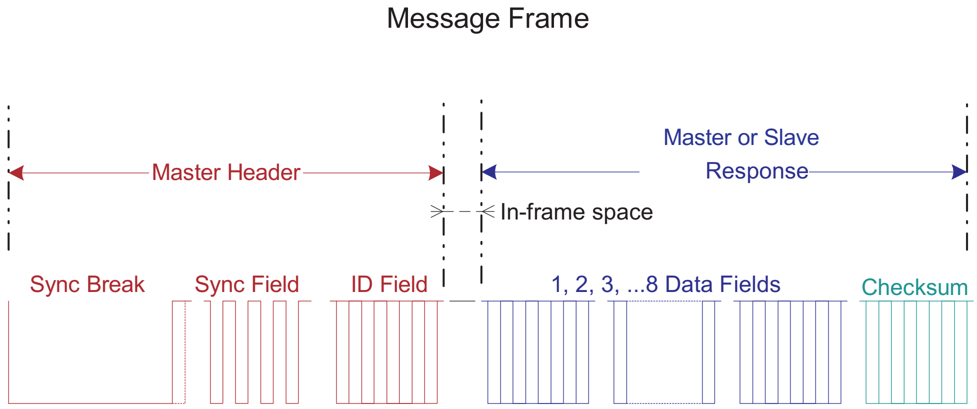
The LIN module covers the CPU performance-consuming features, defined in the *LIN Specification Package* Revision 1.3 and 2.0 by hardware.

The Master Mode of LIN module is compatible with LIN 2.1 standard.

29.3.1.2 Message Frame

The LIN protocol defines a message frame format, illustrated in Figure 29-12. Each frame includes one master header, one response, one in-frame response space, and inter-byte spaces. In-frame-response and inter-byte spaces may be 0.

Figure 29-12. LIN Protocol Message Frame Format: Master Header and Slave Response



There is no arbitration in the definition of the LIN protocol; therefore, multiple slave nodes responding to a header might be detected as an error.

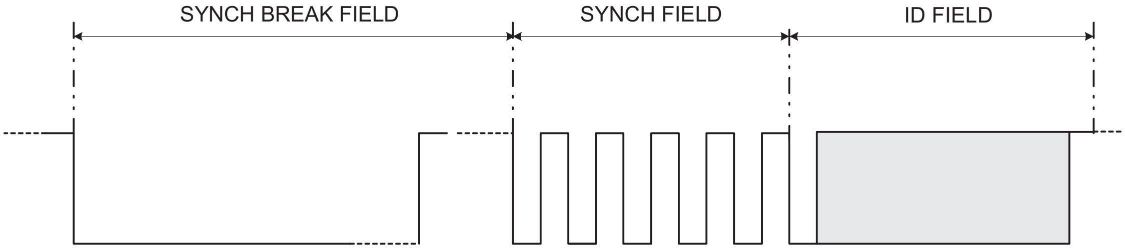
The LIN bus is a single channel wired-AND. The bus has a binary level: either dominant for a value of 0, or recessive for a value of 1.

29.3.1.2.1 Message Header

The header of a message is initiated by a master (see Figure 29-13) and consists of a three field-sequence:

- The synch break field signaling the beginning of a message
- The synch field conveying bit rate information of the LIN bus
- The ID field denoting the content of a message

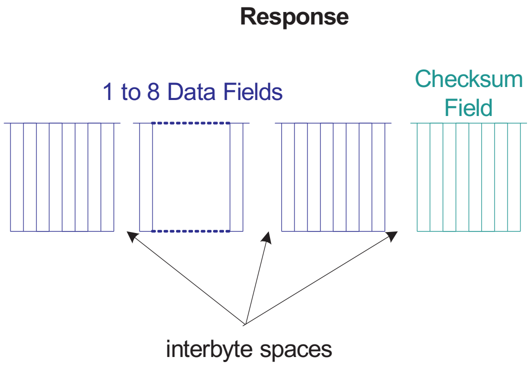
Figure 29-13. Header 3 Fields: Synch Break, Synch, and ID



29.3.1.2.2 Response

The format of the response is as illustrated in Figure 29-14. There are two types of fields in a response: data and checksum. The data field consists of exactly one data byte, one start bit, and one stop bit, for a total of 10 bits. The LSB is transmitted first. The checksum field consists of one checksum byte, one start bit and one stop bit. The checksum byte is the inverted modulo-256 sum over all data bytes in the data fields of the response.

Figure 29-14. Response Format of LIN Message Frame



The format of the response is a stream of N data fields and one checksum field. Typically N is from 1 to 8, with the exception of the extended command frames (Section 29.3.1.6). The length N of the response is indicated either with the optional length control bits of the ID Field (this is used in standards earlier than LIN 1.x); see Table 29-5, or by LENGTH value in SCIFORMAT[18:16] register; see Table 29-6. The SCI/LIN module supports response lengths from 1 to 8 bytes in compliance with LIN 2.0.

Table 29-5. Response Length Info Using IDBYTE Field Bits [5:4] for LIN Standards Earlier than 1.3

ID5	ID4	Number of Data bytes
0	0	2
0	1	2
1	0	4
1	1	8

Table 29-6. Response Length with SCIFORMAT[18:16] Programming

SCIFORMAT[18:16]	No. of Bytes
000	1
001	2
010	3
011	4
100	5
101	6
110	7
111	8

29.3.1.3 Synchronizer

The synchronizer has three major functions in the messaging between master and slave nodes. It generates the master header data stream, it synchronizes to the LIN bus for responding, and it locally detects timeouts. A bit rate is programmed using the prescalers in the BRS register to match the indicated LIN_speed value in the LIN description file.

The LIN synchronizer will perform the following functions: master header signal generation, slave detection and synchronization to message header with optional baud rate adjustment, response transmission timing and timeout control.

The LIN synchronizer is capable of detecting an incoming break and initializing communication at all times.

29.3.1.4 Baud Rate

The transmission baud rate of any node is configured by the CPU at the beginning; this defines the bit time T_{bit}. The bit time is derived from the fields P and M in the baud rate selection register (BRS). There is an additional 3-bit fractional divider value, field U in the BRS register, which further fine-tunes the data field baud rate.

The ranges for the prescaler values in the BRS register are:

$P = 0, 1, 2, 3, . . . , 2^{24} - 1$

$M = 0, 1, 2, . . . , 15$

$U = 0, 1, 2, 3, 4, 5, 6, 7$

The P, M, and U values in the BRS register are user programmable. The P and M dividers could be used for both SCI mode and LIN mode to select a baud rate. The U value is an additional 3-bit value determining that “**a TVCLK**“ (with **a** = 0,1) is added to each T_{bit} as explained in Section 29.3.1.4.2. If the ADAPT bit is set and the LIN slave is in adaptive baud rate mode, then all these divider values are automatically obtained during header reception when the synchronization field is measured.

The LIN protocol defines baud rate boundaries as follows:

$1kHz \le F_{LINCLK} \le 20kHz$

All transmitted bits are shifted in and out at T_{bit} periods.

29.3.1.4.1 Fractional Divider

The M field of the BRS register modifies the integer prescaler P for fine tuning of the baud rate. The M value adds in increments of 1/16 of the P value.

The bit time, T_{bit} is expressed in terms of the VCLK period T_{VCLK} as follows:

For all P other than 0, and all M,

$$T_{bit} = 16\left(P + 1 + \frac{M}{16}\right)T_{VCLK} \tag{45}$$

For P= 0 : $T_{bit} = 32T_{VCLK}$

Therefore, the LINCLK frequency is given by:

$$F_{LINCLK} = \frac{F_{VCLK}}{16(P + 1 + \frac{M}{16})} \quad \text{For all P other than zero}$$

$$F_{LINCLK} = \frac{F_{VCLK}}{32} \quad \text{For P = 0} \tag{46}$$

29.3.1.4.2 Superfractional Divider

The superfractional divider scheme applies to the following modes:

- LIN master mode (synch field + identifier field + response field + checksum field)
- LIN slave mode (response field + checksum field)

29.3.1.4.3 Superfractional Divider In LIN Mode

Building on the 4-bit fractional divider M (BRS[27:24], the superfractional divider uses an additional 3-bit modulating value, illustrated in Table 29-7. The sync field (0x55), the identifier field and the response field can all be seen as 8-bit data bytes flanked by a start bit and a stop bit. The bits with a 1 in the table will have an additional VCLK period added to their T_{bit}.

Table 29-7. Superfractional Bit Modulation for LIN Master Mode and Slave Mode [1]

BRS[30:28]	Start Bit	D[0]	D[1]	D[2]	D[3]	D[4]	D[5]	D[6]	D[7]	Stop Bit
0h	0	0	0	0	0	0	0	0	0	0
1h	1	0	0	0	0	0	0	0	1	0
2h	1	0	0	0	1	0	0	0	1	0
3h	1	0	1	0	1	0	0	0	1	0
4h	1	0	1	0	1	0	1	0	1	0
5h	1	1	1	0	1	0	1	0	1	1
6h	1	1	1	0	1	1	1	0	1	1
7h	1	1	1	1	1	1	1	0	1	1

[1]

1. In LIN master mode bit modulation applies to synch field + identifier field + response field
2. In LIN slave mode bit modulation applies to identifier field + response field

The baud rate will vary over a LIN data field to average according to the BRS[30:28] value by a *d* fraction of the peripheral internal clock: 0<d<1.

The instantaneous bit time is expressed in terms of T_{VCLK} *as follows:*

For all P other than 0, and all M and d (0 or 1),

$$T^{i}bit = \left[16\left(P+1+\frac{M}{16}\right)+d\right]T_{VCLK} \tag{47}$$

For P = 0 $T_{bit} = 32T_{VCLK}$

The averaged bit time is expressed in terms of T_{VCLK} as follows*:*

For all P other than 0, and all M and d (0<d<1),

$$T^{a}bit = \left[16\left(P+1+\frac{M}{16}\right)+d\right]T_{VCLK} \tag{48}$$

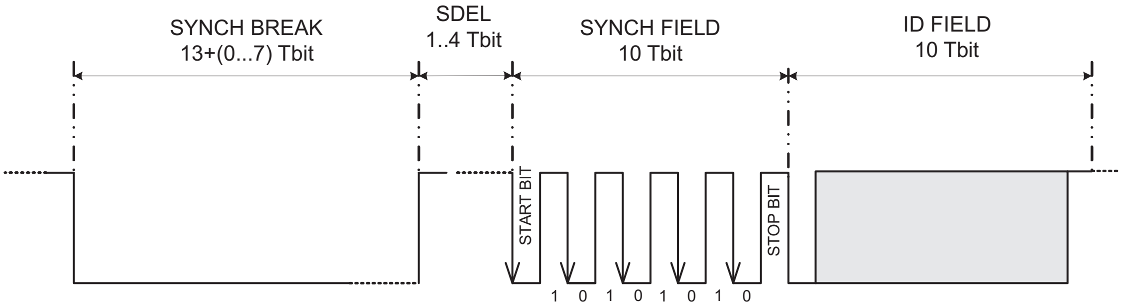
For P = 0 $T_{bit} = 32T_{VCLK}$

With the superfractional divider, a LIN baud rate of 20 kbps is achievable with an internal clock VCLK of 726 kHz. Furthermore, a rate of 400 kbps is achievable with an VCLK of 14.6 MHz.

29.3.1.5 Header Generation

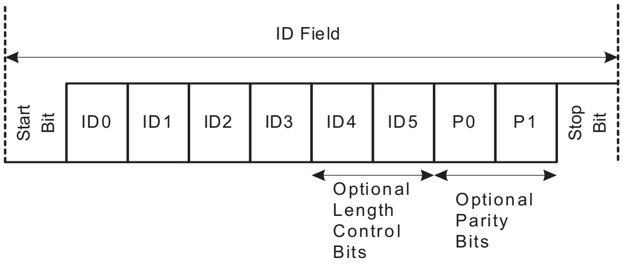
Automatic generation of the LIN protocol header data stream is supported without CPU interaction. The CPU pr the DMA will trigger a message header generation and the LIN state machine will handle the generation itself. A master node initiates header generation on CPU or DMA writes to the IDBYTE in the LINID register. The header is always sent by the master to initiate a LIN communication and consists of three fields: break field, synchronization field, and identification field, as seen in Figure 29-15.

Figure 29-15. Message Header in Terms of T_{bit}



- The break field consists of two components:
  - The synchronization break (SYNCH BREAK) consists of a minimum of 13 (dominant) low bits to a maximum of 20 dominant bits. The synch break length may be extended from the minimum with the 3-bit SBREAK value in the LINCOMP register.
  - The synchronization break delimiter (SDEL) consists of a minimum of 1 (recessive) high bit to a maximum of 4 recessive bits. The delimiter marks the end of the synchronization break field. The synch break delimiter length depends on the 2-bit SDEL value in the LINCOMP register.
- The synchronization field (SYNCH FIELD) consists of one start bit, byte 0x55, and a stop bit. It is used to convey T_{bit} information and resynchronize LIN bus nodes.
- The identifier field’s ID byte may use six bits as an identifier, with optional length control (see *Note:Optional Control Length Bits*), and two optional bits as parity of the identifier. The identifier parity is used and checked if the PARITY ENA bit is set. If length control bits are not used, then there can be a total of 64 identifiers plus parity. If neither length control or parity are used there can be up to 256 identifiers. See Figure 29-16 for an illustration of the ID field.

Figure 29-16. ID Field



NOTE: Optional Control Length Bits

The control length bits only apply to LIN standards prior to LIN 1.3. IDBYTE field conveys response length information if compliant to standards earlier than LIN1.3. The SCIFORMAT register stores the length of the response for later versions of the LIN protocol.

NOTE: If the BLIN module, configured as Slave in multi-buffer mode, is in the process of transmitting data while a new header comes in, the module might end up in responding with the data from the previous interrupted response (not the data corresponding to the new ID). To avoid this scenario the following procedure could be used:

1. Check for the Bit Error (BE) during the response transmission. If the BE flag is set, this indicates that a collision has happened on the LIN bus (here because of the new Synch Break).
2. In the Bit Error ISR, configure the TD0 and TD1 registers with the next set of data to be transmitted on a TX Match for the incoming ID. Before writing to TD0/TD1 make sure that there was not already an update because of a Bit Error; otherwise TD0/TD1 might be written twice for one ID.
3. Once the complete ID is received, based on the match, the newly configured data will be transmitted by the node.

29.3.1.5.1 Event Triggered Frame Handling Proposal

The LIN 2.0 protocol uses event-triggered frames that may occasionally cause collisions. Event-triggered frames have to be handled in software.

If no slave answers to an event triggered frame header, the master node will set the NRE flag, and a NRE interrupt will occur if enabled. If a collision occurs, a frame error and checksum error may arise before the NRE error. Those errors are flagged and the appropriate interrupts will occur, if enabled.

Frame errors and checksum errors depend on the behavior and synchronization of the responding slaves. If the slaves are totally synchronized and stop transmission once the collision occurred, it is possible that only the NRE error is flagged despite the occurrence of a collision. To detect if there has been a reception of one byte before the NRE error is flagged, the BUS BUSY flag can be used as an indicator.

The bus busy flag is set on the reception of the first bit of the header and remains set until the header reception is complete, and again is set on the reception of the first bit of the response. In the case of a collision the flag is cleared in the same cycle as the NRE flag is set.

Software could implement the following sequence:

- Once the reception of the header is done (poll for RXID flag), wait for the bus busy flag to get set or NRE flag to get set.
- If bus busy flag is not set before NRE flag, then it is a true no response case (no data has been transmitted onto the bus).
- If bus busy flag gets set, then wait for NRE flag to get set or for successful reception. If NRE flag is set, then in this case a collision has occurred on the bus.

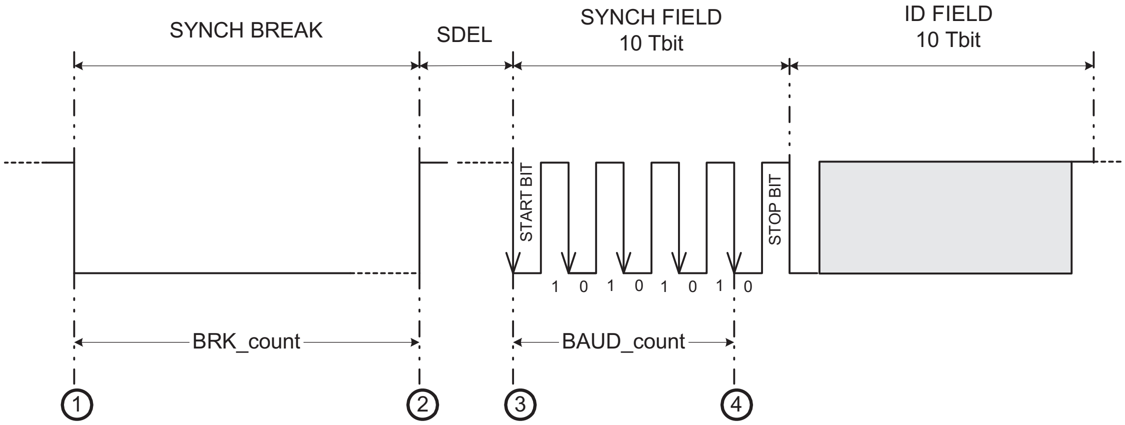
Even in the case of a collision, the received (corrupted) data is accessible in the RX buffers; registers LINRD0 and LINRD1.

29.3.1.5.2 Header Reception and Adaptive Baud Rate

A slave node baud rate can optionally be adjusted to the detected bit rate as an option to the LIN module. The adaptive baud rate option is enabled by setting the ADAPT bit. During header reception, a slave measures the baud rate during detection of the synch field. If ADAPT bit is set, then the measured baud rate is compared to the slave node's programmed baud rate and adjusted to the LIN bus baud rate if necessary.

The LIN synchronizer determines two measurements: BRK_count and BAUD_count (Figure 29-17). These values are always calculated during the Header reception for synch field validation (Figure 29-18).

Figure 29-17. Measurements for Synchronization



By measuring the values BRK_count and BAUD_count, a valid synch break sequence can be detected as described in Figure 29-18. The four numbered events in Figure 29-17 signal the start/stop of the synchronizer counter. The synchronizer counter uses VCLK as the time base.

The synchronizer counter is used to measure the synch break relative to the detecting node T_{bit}. For a slave node receiving the synch break, a threshold of 11 T_{bit} is used as required by the LIN protocol. For detection of the dominant data stream of the synch break, the synchronizer counter is started on a falling edge and stopped on a rising edge of the LINRX. On detection of the synch break delimiter, the synchronizer counter value is saved and then reset.

On detection of five consecutive falling edges, the BAUD_count is measured. Bit timing calculation and consistency to required accuracy is implemented following the recommendations of LIN revision 2.0. A slave node can calculate a single T_{bit} time by division of BAUD_count by 8. In addition, for consistency between the detected edges the following is evaluated:

BAUD_count + BAUD_count » 2 + BAUD_count » 3 ≤ BRK_count

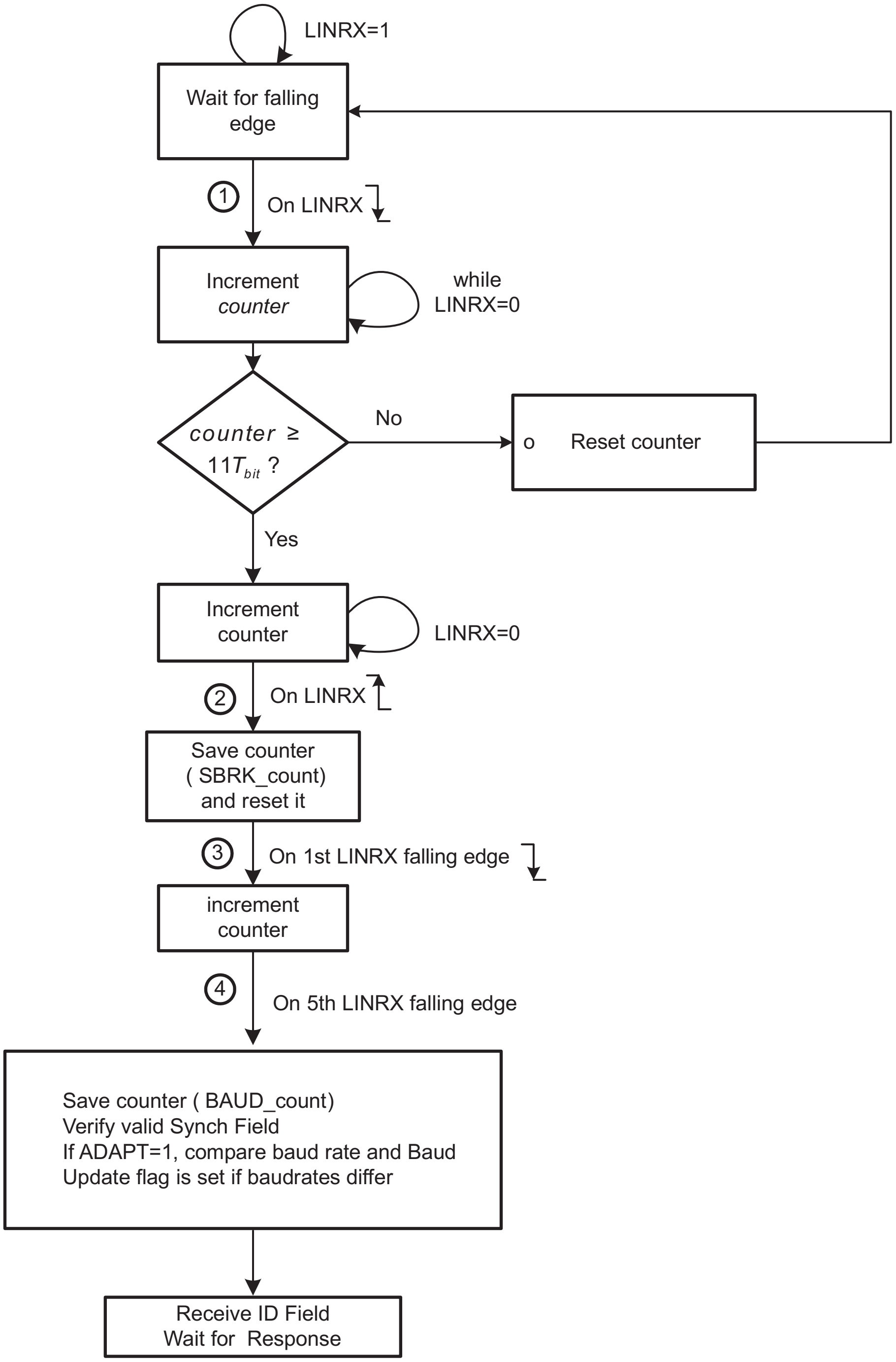
The BAUD_count value is shifted 3 times to the right and rounded using the first insignificant bit to obtain a T_{bit} unit. If the ADAPT bit is set, then the detected baud rate is compared to the programmed baud rate.

During the header reception processing as illustrated in Figure 29-18, if the measured BRK_count value is less than 11 T_{bit}, the synch break is not valid according to the protocol for a fixed rate. If the ADAPT bit is set, then the MBRS register is used for measuring BRK_count and BAUD_count values and automatically adjusts to any allowed LIN bus rate (refer to *LIN Specification Package 2.0*).

NOTE: In adaptive mode the MBRS divider should be set to allow a maximum baud rate that is not more than 10% above the expected operating baud rate in the LIN network. Otherwise, a 0x00 data byte could mistakenly be detected as a synch break.

The break-threshold relative to the slave node is 11 T_{bit}. The break is 13 T_{bit} as specified in LIN version 1.3.

Figure 29-18. Synchronization Validation Process and Baud Rate Adjustment



If the synch field is not detected within the given tolerances, the inconsistent-synch-field-error (ISFE) flag will be set. An ISFE interrupt will be generated, if enabled by its respective bit in the SCISETINT register. The ID byte should be received after the synch field validation was successful. Any time a valid break (larger than 11 T_{bit}) is detected, the receiver's state machine should reset to reception of this new frame. This reset condition is only valid during response state, not if an additional synch break occurs during header reception.

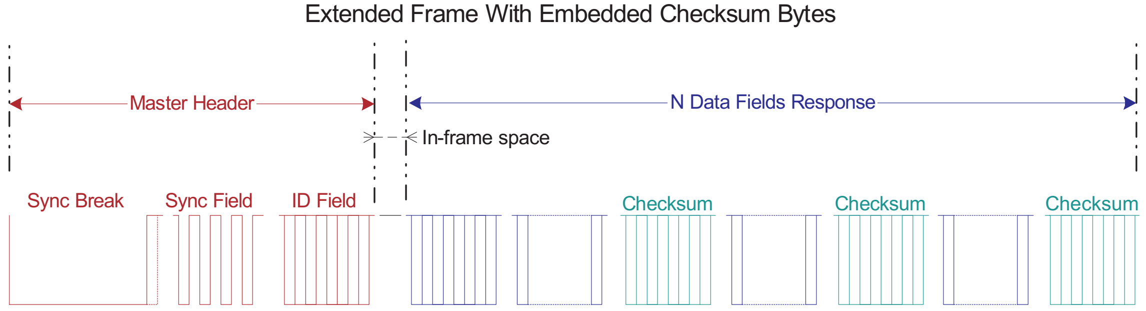
NOTE: When an inconsistent synch field (ISFE) error occurs, suggested action for the application is to Reset the SWnRST bit and set the SWnRST bit to make sure that the internal state machines are back to their normal states

29.3.1.6 Extended Frames Handling

The LIN protocol 2.0 and prior includes two extended frames with identifiers 62 (user-defined) and 63 (reserved extended). The response data length of the user-defined frame (ID 62, or 0x3E) is unlimited. The length for this identifier will be set at network configuration time to be shared with the LIN bus nodes.

Extended frame communication is triggered on reception of a header with identifier 0x3E; see Figure 29-19. Once the extended frame communication is triggered, unlike normal frames, this communication needs to be stopped before issuing another header. To stop the extended frame communication the STOP EXT FRAME bit must be set.

Figure 29-19. Optional Embedded Checksum in Response for Extended Frames



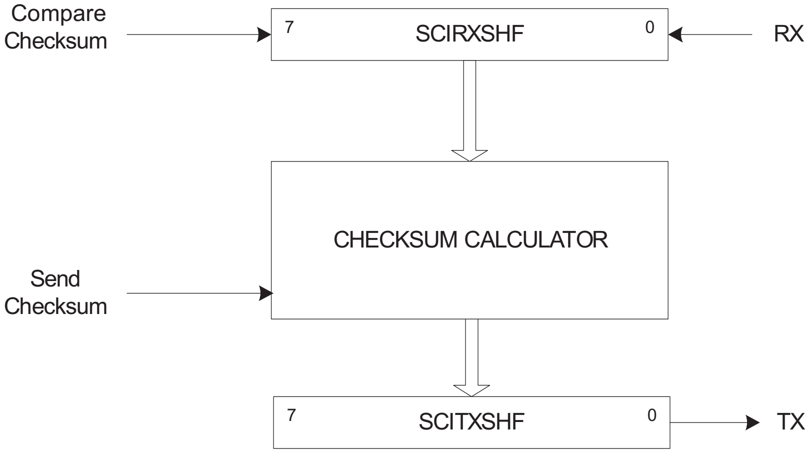
An ID interrupt will be generated (if enabled and there is a match) on reception of ID 62 (0x3E). This interrupt allows the CPU using a software counter to keep track of the bytes that are being sent out and decides when to calculate and insert a checksum byte (recommended at periodic rates). To handle this procedure, SC bit is used. A write to the send checksum bit SC will initiate an automatic send of the checksum byte. The last data field should always be a checksum in compliance with the LIN protocol.

The periodicity of the checksum insertion, defined at network configuration time, is used by the receiving node to evaluate the checksum of the ongoing message, and has the benefit of enhanced reliability.

For the sending node, the checksum is automatically embedded each time the send checksum bit SC is set. For the receiving node, the checksum is compared each time the compare checksum bit CC is set; see Figure 29-20.

NOTE: The LIN 2.0 enhanced checksum does not apply to the reserved identifiers. The reserved identifiers always use the classic checksum.

Figure 29-20. Checksum Compare and Send for Extended Frames



29.3.1.7 Timeout Control

Any LIN node listening to the bus and expecting a response initiated from a master node could flag a no-response error timeout event. The LIN protocol defines four types of timeout events, which are all handled by the hardware of the LIN module. The four LIN protocol events are:

- No-response timeout error
- Bus idle detection
- Timeout after wakeup signal
- Timeout after three wakeup signals

29.3.1.7.1 No-Response Error (NRE)

The no-response error will occur when any node expecting a response waits for T_{FRAME_MAX} time and the message frame is not fully completed within the maximum length allowed, T_{FRAME_MAX}. After this time a no-response error (NRE) is flagged in the NRE bit of the SCIFLR register. An interrupt is triggered if enabled.

As specified in the LIN 1.3 standard, the minimum time to transmit a frame is:

$T_{FRAME\_MIN} = T_{HEADER\_MIN} + T_{DATA\_FIELD} + T_{CHECKSUM\_FIELD}$

$= 44 + 10N$

where N = number of data fields.

And the maximum time frame is given by:

$T_{FRAME\_MAX} = T_{FRAME\_MIN} * 1.4$

$= (44 + 10N) * 1.4$

The timeout value T_{FRAME_MAX} is derived from the *N* number of data fields value. The *N* value is either embedded in the header's ID field for messages or is part of the description file. In the latter case, the 3-bit CHAR value in SCIFORMAT register, will indicate the value for *N*.

NOTE: The length coding of the ID field does not apply to two extended frame identifiers, ID fields of 0x3E (62) and 0x3F (63). In these cases, the ID field can be followed by an arbitrary number of data byte fields. Also, the LIN 2.0 protocol specification mentions that ID field 0x3F (63) cannot be used. For these two cases, the NRE will not be handled by the LIN controller hardware.

Table 29-8. Timeout Values in T_{bit} Units

N	T_{DATA_FIELD}	T_{FRAME_MIN}	T_{FRAME_MAX}
1	10	54	76
2	20	64	90
3	30	74	104
4	40	84	118
5	50	94	132
6	60	104	146
7	70	114	160
8	80	124	174

29.3.1.7.2 Bus Idle Detection

The second type of timeout can occur when a node detects an inactive LIN bus: no transitions between recessive and dominant values are detected on the bus. This happens after a minimum of 4 s (this is 80,000 F_{LINCLK} cycles with the fastest bus rate of 20 kbps). If a node detects no activity in the bus as the TIMEOUT bit is set, then it can be assumed that the LIN bus is in sleep mode. Application software can use the Timeout flag to determine when the LIN bus is inactive and put the LIN into sleep mode by writing the POWERDOWN bit.

NOTE: After the timeout was flagged, a SW nRESET should be asserted before entering Low-Power Mode. This is required to reset the receiver in case that an incomplete frame was on the bus before the idle period.

29.3.1.7.3 Timeout after Wakeup Signal and Timeout after Three Wakeup Signals

The third and fourth types of timeout are related to the wakeup signal. A node initiating a wakeup should expect a header from the master within a defined amount of time: timeout after wakeup signal. See Section 29.4.3 for more details.

29.3.1.8 TXRX Error Detector (TED)

The following sources of error are detected by the TXRX error detector logic (TED). The TED logic consists of a bit monitor, an ID parity checker, and a checksum error. The following errors are detected:

- Bit errors (BE)
- Physical bus errors (PBE)
- Identifier parity errors (PE)
- Checksum errors (CE)

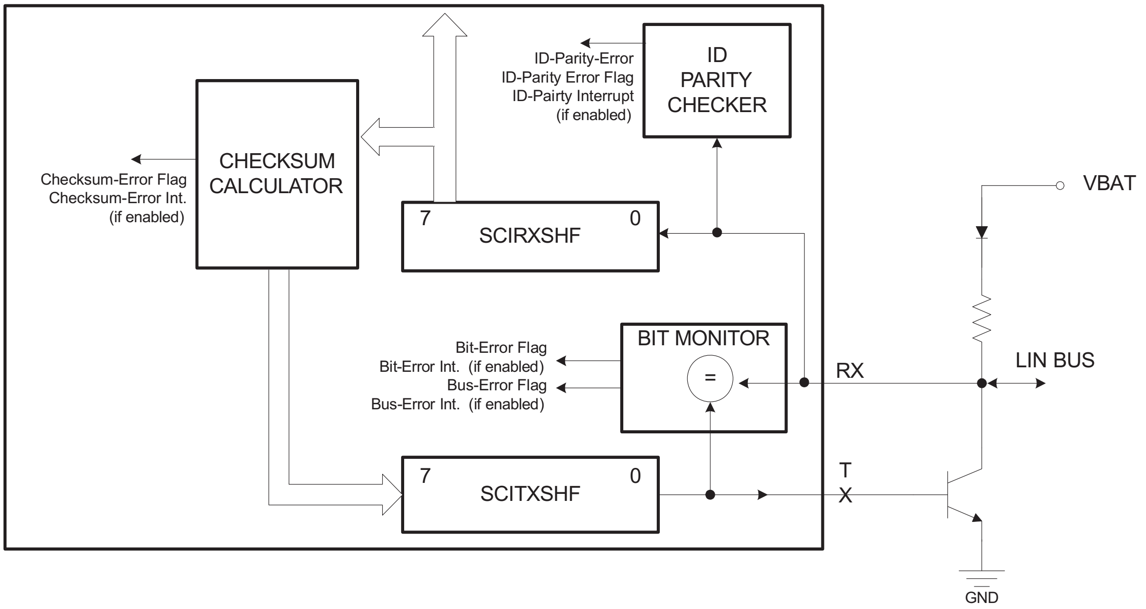
All of these errors (BE, PBE, PE, CE) are flagged. An interrupt for the flagged errors will be generated if enabled. A message is valid for both the transmitter and the receiver if there is no error detected until the end of the frame.

29.3.1.8.1 Bit Errors

A bit error (BE) is detected at the bit time when the bit value that is monitored is different from the bit value that is sent. A bit error is indicated by the BE flag in SCIFLR. After signaling a BE, the transmission is aborted no later than the next byte. The bit monitor ensures that the transmitted bit in LINTX is the correct value on the LIN bus by reading back on the LINRX pin as shown in Figure 29-21.

NOTE: If BE Occurs due to New Header reception during a Slave Response, NRE/TIMEOUT flag will not be set for the new Frame.

Figure 29-21. TXRX Error Detector



29.3.1.8.2 Physical Bus Errors

A Physical Bus Error (PBE) has to be detected by a master if no valid message can be generated on the bus (Bus shorted to GND or VBAT). The bit monitor detects a PBE during the header transmission, if no Synch Break can be generated (for example, because of a bus shortage to VBAT) or if no Synch break Delimiter can be generated (for example, because of a bus shortage to GND). Once the Synch Break Delimiter was validated, all other deviations between the monitored and the sent bit value are flagged as Bit Errors (BE) for this frame.

29.3.1.8.3 ID Parity Errors

If parity is enabled, an ID parity error (PE) is detected if any of the two parity bits of the sent ID byte are not equal to the calculated parity on the receiver node. The two parity bits are generated using the following mixed parity algorithm.

$$P0 = ID0 \oplus ID1 \oplus ID2 \oplus ID4 \ (even \ Parity)$$
$$P1 = ID1 \oplus ID3 \oplus ID4 \oplus ID5 \ (odd \ Parity) \qquad (49)$$

If an ID-parity error is detected, the ID-parity error is flagged, and the received ID is not valid. See Section 29.3.1.9 for details.

29.3.1.8.4 Checksum Errors

A checksum error (CE) is detected and flagged at the receiving end if the calculated modulo-256 sum over all received data bytes (including the ID byte if it is the enhanced checksum type) plus the checksum byte does not result in 0xFF. The modulo-256 sum is calculated over each byte by adding with carry, where the carry bit of each addition is added to the LSB of its resulting sum.

For the transmitting node, the checksum byte sent at the end of a message is the inverted sum of all the data bytes (see Figure 29-22) for classic checksum implementation. The checksum byte is the inverted sum of the identifier byte and all the data bytes (see Figure 29-23) for the LIN 2.0 compliant enhanced checksum implementation. The classic checksum implementation should always be used for reserved identifiers 60 to 63; therefore, the CTYPE bit will be overridden in this case. For signal-carrying-frame identifiers (0 to 59) the type of checksum used depends on the CTYPE bit.

Figure 29-22. Classic Checksum Generation at Transmitting Node

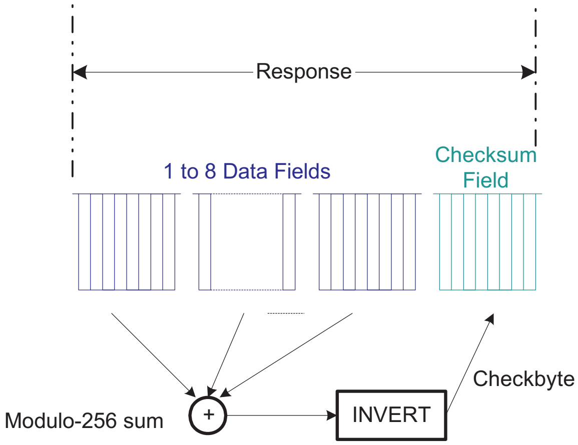
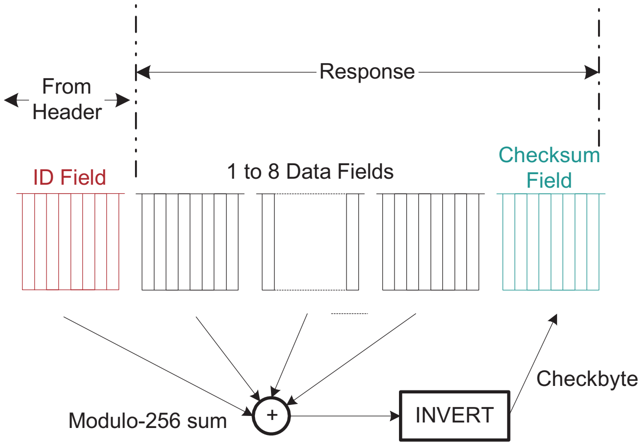


Figure 29-23. LIN 2.0-Compliant Checksum Generation at Transmitting Node



29.3.1.9 Message Filtering and Validation

Message filtering uses the entire identifier to determine which nodes will participate in a response, either receiving or transmitting a response. Therefore, two acceptance masks are used as shown in Figure 29-24. During header reception, all nodes filter the ID-Field (ID-Field is the part of the header explained in Figure 29-16) to determine whether they transmit a response or receive a response for the current message. There are two masks for message ID filtering: one to accept a response reception, the other to initiate a response transmission. See Figure 29-24. All nodes compare the received ID to the identifier stored in the ID-SlaveTask BYTE of the LINID register and use the RX ID MASK and the TX ID MASK fields in the LINMASK register to filter the bits of the identifier that should not be compared.

If there is an RX match with no parity error and the RXENA bit is set, there will be an ID RX flag and an interrupt will be triggered if enabled. If there is a TX match with no parity error and the TXENA bit is set, there will be an ID TX flag and an interrupt will be triggered if enabled in the SCISETINT register.

The masked bits become don’t cares for the comparison. To build a mask for a set of identifiers, an XOR function could be used.

For example, to build a mask to accept IDs 0x26 and 0x25 using LINID[7:0] = 0x20; that is, compare 5 most significant bits (MSBs) and filter 3 least significant bits (LSBs), the acceptance mask could be:

$$(0x26 + 0x25) \oplus 0x20 = 0x07 \tag{50}$$

A mask of all zeros will compare all bits of the received identifier in the shift register with the ID-BYTE in LINID[7:0]. If HGEN CTRL is set to 1, a mask of 0xFF will always cause a match. A mask of all 1s will filter all bits of the received identifier, and thus there will be an ID match regardless of the content of the ID-SlaveTask BYTE field in the LINID register.

NOTE: When the HGEN CTRL bit = 0, the LIN nodes compare the received ID to the ID-BYTE field in the LINID register, and use the RX ID MASK and the TX ID MASK in the LINMASK register to filter the bits of the identifier that should not be compared.

If there is an RX match with no parity error and the RXENA bit is set, there will be an ID RX flag and an interrupt will be triggered if enabled. A mask of all 0s will compare all bits of the received identifier in the shift register with the ID-BYTE field in LINID[7:0]. A mask of all 1s will filter all bits of the received identifier and there will be no match.

During header reception, the received identifier is copied to the Received ID field LINID[23:16]. If there is no parity error and there is either a TX match or an RX match, then the corresponding TX or RX ID flag is set. If the ID interrupt is enabled, then an ID interrupt is generated.

After the ID interrupt is generated, the CPU may read the Received ID field LINID[23:16] and determine what response to load into the transmit buffers.

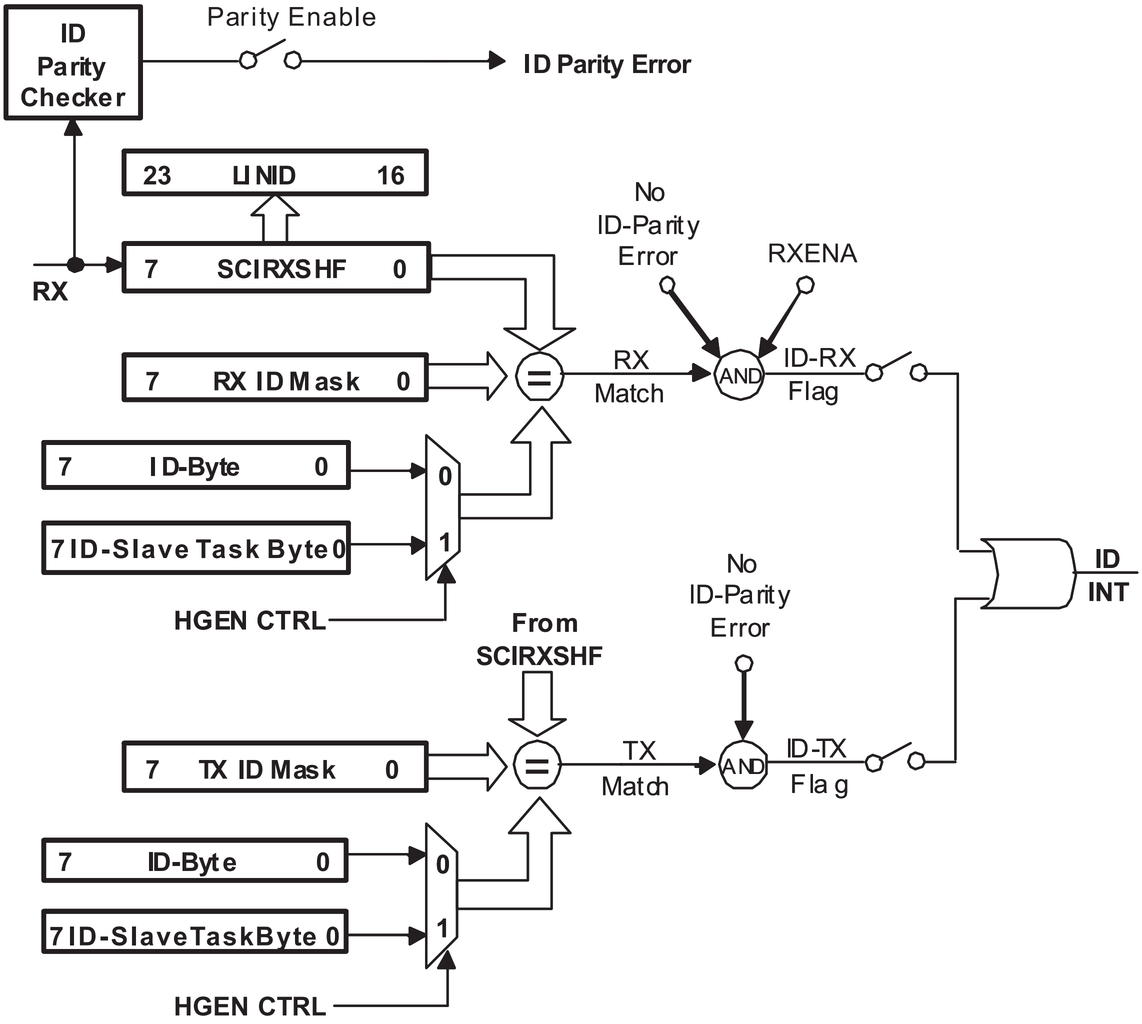
NOTE: When byte 0 is written to TD0 (LINTD0[31:24]), the response transmission is automatically generated.

In multi-buffer mode, the TXRDY flag will be set when all the response data bytes and checksum byte are copied to the shift register SCITXSHF. In non-multi-buffer mode, the TXRDY flag is set each time a byte is copied to the SCITXSHF register, and also for the last byte of the frame after the checksum byte is copied to the SCITXSHF register.

In multi-buffer mode, the TXEMPTY flag is set when both the transmit buffer(s) TDy and the SCITXSHF shift register are emptied and the checksum has been sent. In nonmulti-buffer mode, TXEMPTY is set each time TD0 and SCITXSHF are emptied, except for the last byte of the frame where the checksum byte must also be transmitted.

If parity is enabled, all slave receiving nodes will validate the identifier using all eight bits of the received ID byte. The SCI/LIN will flag a corrupted identifier if an ID-parity error is detected.

Figure 29-24. ID Reception, Filtering and Validation



29.3.1.10 Receive Buffers

To reduce CPU load when receiving a LIN N-byte (with N = 1–8) response in interrupt mode or DMA mode, the SCI/LIN module has eight receive buffers. These buffers can store an entire LIN response in the RDy receive buffers. Figure 29-8 illustrates the receive buffers.

The checksum byte following the data bytes is validated by the internal checksum calculator. The checksum error (CE) flag indicates a checksum error and a CE interrupt will be generated if enabled in the SCISETINT register.

The multi-buffer 3-bit counter counts the data bytes transferred from the SCIRXSHF register to the RDy receive buffers if multi-buffer mode is enabled, or to RD0 if multi-buffer mode is disabled. The 3-bit compare register contains the number of data bytes expected to be received. In cases where the ID BYTE field does not convey message length (see *Note:Optional Control Length Bits* in Section 29.3.1.5), the LENGTH value, indicates the expected length and is used to load the 3-bit compare register. Whether the length control field or the LENGTH value is used is selectable with the COMM MODE bit.

A receive interrupt, and a receive ready RXRDY flag set as well as a DMA request (RXDMA) could occur after receiving a response if there are no response receive errors for the frame (such as, there is no checksum error, frame error, and overrun error). The checksum byte will be compared before acknowledging a reception. A DMA request can be generated for each received byte or for the entire response depending on whether the multi-buffer mode is enabled or not (MBUF MODE bit).

NOTE: In multi-buffer mode following are the scenarios associated with clearing the "RXRDY" flag bit:

1. The RXRDY flag cannot be cleared by reading the corresponding interrupt offset in the SCIINTVECT0/1 register.
2. For LENGTH less than or equal to 4, Read to RD0 register will clear the “RXRDY” flag.
3. For LENGTH greater than 4, Read to RD1 register will clear the “RXRDY” flag.

29.3.1.11 Transmit Buffers

To reduce the CPU load when transmitting a LIN N-byte (with N = 1–8) response in interrupt mode or DMA mode, the SCI/LIN module has eight transmit buffers, TD0–TD7 in LINTD0 and LINTD1. With these transmit buffers, an entire LIN response field can be preloaded in the TXy transmit buffers. Optionally, a DMA transfer could be done on a byte-per-byte basis when multi-buffer mode is not enabled (MBUF MODE bit). Figure 29-9 illustrates the transmit buffers.

The multi-buffer 3-bit counter counts the data bytes transferred from the TDy transmit buffers register if multi-buffer mode is enabled, or from TD0 to SCITXSHF if multi-buffer mode is disabled. The 3-bit compare register contains the number of data bytes expected to be transmitted. If the ID field is not used to convey message length (see *Note:Optional Control Length Bits* in Section 29.3.1.5), the LENGTH value indicates the expected length and is used instead to load the 3-bit compare register. Whether the length control field or the LENGTH value is used is selectable with the COMM MODE bit.

A transmit interrupt (TX interrupt), and a transmit ready flag (TXRDY flag), and a DMA request (TXDMA) could occur after transmitting a response. A DMA request can be generated for each transmitted byte or for the entire response depending on whether multi-buffer mode is enabled or not (MBUF MODE bit).

The checksum byte will be automatically generated by the checksum calculator and sent after the data-fields transmission is finished. The multi-buffer 3-bit counter counts the data bytes transferred from the TDy buffers into the SCITXSHF register.

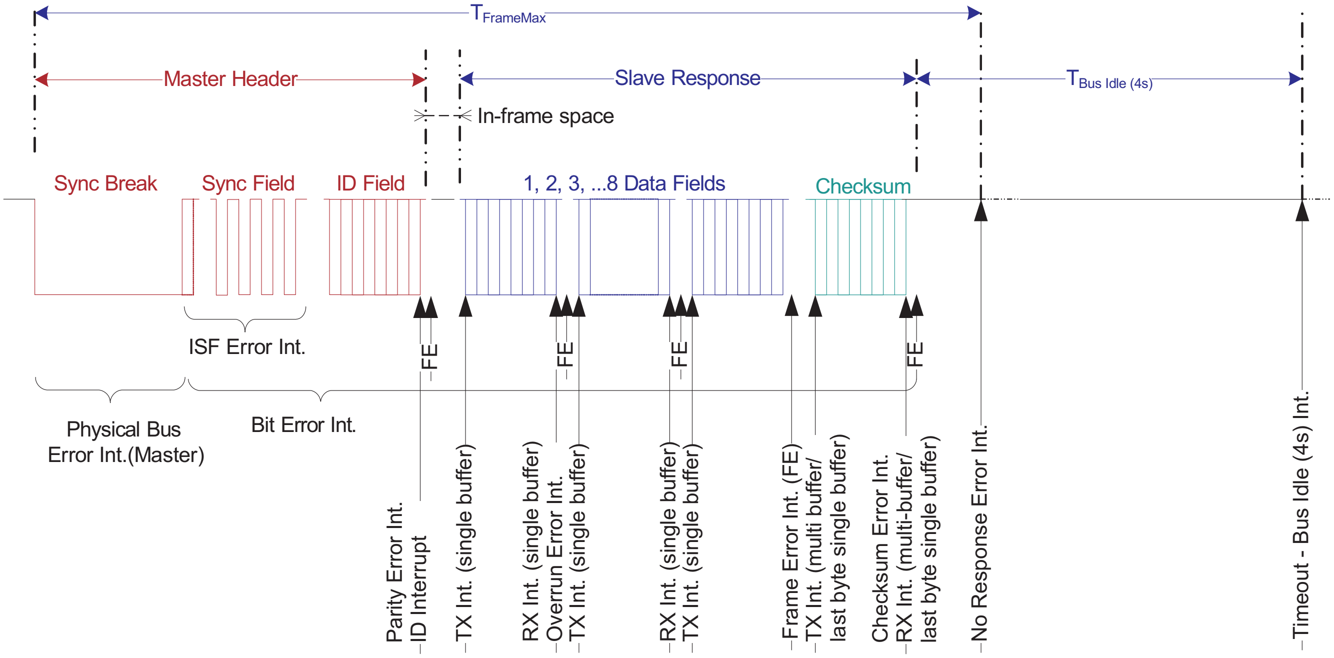
NOTE: The transmit interrupt request can be eliminated until the next series of data is written into the transmit buffers LINTD0 and LINTD1, by disabling the corresponding interrupt via the SCICLRINT register or by disabling the transmitter via the TXENA bit.

29.3.2 LIN Interrupts

LIN and SCI mode have a common Interrupt block as explained in Section 29.2.2. There are 16 interrupt sources in the SCI/LIN module, with 8 of them being LIN mode only, as seen in Table 29-4.

A LIN message frame indicating the timing and sequence of the LIN interrupts that could occur is shown in Figure 29-25.

Figure 29-25. LIN Message Frame Showing LIN Interrupt Timing and Sequence



29.3.3 LIN DMA Interface

LIN DMA Interface uses the SCI DMA interface logic. DMA requests for receive (RXDMA request) and transmit (TXDMA request) are available for the SCI/LIN module. There are two modes for DMA transfers depending on whether multi-buffer mode is enabled or not via the multi-buffer enable control bit (MBUF MODE).

29.3.3.1 LIN Receive DMA Requests

In LIN mode, when the multi-buffer option is enabled, if a received response (up to eight data bytes) is transferred to the receive buffers (RDy), then a DMA request is generated. If the multi-buffer option is disabled, then DMA requests will be generated on a byte-per-byte basis until all the expected response data fields are received. This DMA functionality is enabled and disabled using the SET RX DMA and CLR RX DMA bits, respectively.

29.3.3.2 LIN Transmit DMA Requests

In LIN mode with the multi-buffer option enabled, after a transmission (up to eight data bytes stored in the transmit buffer(s) TDy in the LINTD0 and LINTD1 registers), a DMA request is generated in order to reload the transmit buffer for the next transmission. If the multi-buffer option is disabled, then DMA requests will be generated on a byte-per-byte basis until all bytes are transferred. This DMA functionality is enabled and disabled using the SET TX DMA and CLR TX DMA bits, respectively.

29.3.4 LIN Configurations

The following list details the configuration steps that software should perform prior to the transmission or reception of data in LIN mode. As long as the SWnRST bit in the SCIGCR1 register is cleared to 0 the entire time that the LIN is being configured, the order in which the registers are programmed is not important.

- Enable LIN by setting the RESET bit in SCIGCR0 to 1.
- Clear the SWnRST bit to 0 before LIN is configured.
- Select the desired frame format by programming the SCIGCR1 register.
- Set both the RX FUNC and TX FUNC bits in SCIPIO0 to 1 to configure the LINRX and LINTX pins for LIN functionality.
- Set the LIN MODE bit in SCIGCR1 to 1 to enable LIN mode.
- Select Master or Slave mode by programming the CLOCK bit in SCIGCR1.
- Set the MBUF MODE bit in SCIGCR1 to 1 to select multi-buffer mode.
- Select the baud rate to be used for communication by programming the BRS register.
- Set the maximum baud rate to be used for communication by programming the BMRS register.
- Set the CONT bit in SCIGCR1 to 1 to make LIN not halt for an emulation breakpoint until its current reception or transmission is complete (this bit is used only in an emulation environment).
- Set the LOOP BACK bit in SCIGCR1 to 1 to connect the transmitter to the receiver internally (this feature is used to perform a self-test).
- Set the RXENA bit in SCIGCR1 to 1, if data is to be received.
- Set the TXENA bit in SCIGCR1 to 1, if data is to be transmitted.
- Select the RX ID MASK and the TX ID MASK fields in the LINMASK register.
- Set the SWnRST bit to 1 after LIN is configured.
- Perform receiving or transmitting data (see Section 29.3.4.1 or Section 29.3.4.2).

29.3.4.1 Receiving Data

The LIN receiver is enabled to receive messages if both the RX FUNC bit and the RXENA bit are set to 1. If the RX FUNC bit is not set, the LINRX pin functions as a general-purpose I/O pin rather than as a LIN function pin.

The ID RX FLAG is set after a valid LIN ID is received with RX Match. An ID interrupt is generated, if enabled.

29.3.4.1.1 Receiving Data in Single-Buffer Mode

Single-buffer mode is selected when the MBUF MODE bit is cleared to 0. In this mode, LIN sets the RXRDY bit when it transfers newly received data from SCIRXSHF to RD0. The SCI clears the RXRDY bit after the new data in RD0 has been read. Also, as data is transferred from SCIRXSHF to RD0, the LIN sets the FE, OE, or PE flags if any of these error conditions were detected in the received data. These error conditions are supported with configurable interrupt capability.

You can receive data by:

1. Polling Receive Ready Flag
2. Receive Interrupt
3. DMA

In polling method, software can poll for the RXRDY bit and read the data from RD0 byte of the LINRD0 register once the RXRDY bit is set high. The CPU is unnecessarily overloaded by selecting the polling method. To avoid this, you can use either the interrupt or DMA method. To use the interrupt method, the SET RX INT bit is set. To use the DMA method, the SET RX DMA bit is set. Either an interrupt or a DMA request is generated the moment the RXRDY bit is set. If the checksum scheme is enabled by setting the Compare Checksum (CC) bit to 1, the checksum will be compared on the byte that is currently being received, which is expected to be the checksum byte. The CC bit will be cleared once the checksum is received. A CE will immediately be flagged if there is a checksum error.

29.3.4.1.2 Receiving Data in Multi-Buffer Mode

Multi-buffer mode is selected when the MBUF MODE bit is set to 1. In this mode, LIN sets the RXRDY bit after receiving the programmed number of data in the receive buffer and the checksum field, the complete frame. The error condition detection logic is similar to the single-buffer mode, except that it monitors for the complete frame. Like single-buffer mode, you can use the polling, interrupt, or DMA method to read the data. The received data has to be read from the LINRD0 and LINRD1 registers, based on the number of bytes. For a LENGTH less than or equal to 4, a read from the LINRD0 register clears the RXRDY flag. For a LENGTH greater than 4, a read from the LINRD1 register clears the RXRDY flag. If the checksum scheme is enabled by setting the Compare Checksum (CC) bit to 1 during the reception of the data, then the byte that is received after the reception of the programmed number of data bytes indicated by the LENGTH field is treated as a checksum byte. The CC bit will be cleared once the checksum is received and compared.

29.3.4.2 Transmitting Data

The LIN transmitter is enabled if both the TX FUNC bit and the TXENA bit are set to 1. If the TX FUNC bit is not set, the LINTX pin functions as a general-purpose I/O pin rather than as a LIN function pin. Any value written to the TD0 before the TXENA bit is set to 1 is not transmitted. Both of these control bits allow for the LIN transmitter to be held inactive independently of the receiver.

The ID TX flag is set after a valid LIN ID is received with TX Match. An ID interrupt is generated, if enabled.

29.3.4.2.1 Transmitting Data in Single-Buffer Mode

Single-buffer mode is selected when the MBUF MODE bit is cleared to 0. In this mode, LIN waits for data to be written to TD0, transfers it to SCITXSHF, and transmits the data. The TXRDY and TX EMPTY bits indicate the status of the transmit buffers. That is, when the transmitter is ready for data to be written to TD0, the TXRDY bit is set. Additionally, if both TD0 and SCITXSHF are empty, then the TX EMPTY bit is also set.

You can transmit data by:

1. Polling Transmit Ready Flag
2. Transmit Interrupt
3. DMA

In polling method, software can poll for the TXRDY bit to go high before writing the data to the TD0. The CPU is unnecessarily overloaded by selecting the polling method. To avoid this, you can use either the interrupt or DMA method. To use the interrupt method, the SET TX INT bit is set. To use the DMA method, the SET TX DMA bit is set. Either an interrupt or a DMA request is generated the moment the TXRDY bit is set. When the LIN has completed transmission of all pending frames, the SCITXSHF register and the TD0 are empty, the TXRDY bit is set, and an interrupt/DMA request is generated, if enabled. Because all data has been transmitted, the interrupt/DMA request should be halted. This can either be done by disabling the transmit interrupt (CLR TX INT) / DMA request (CLR TX DMA bit) or by disabling the transmitter (clear TXENA bit). If the checksum scheme is enabled by setting the Send Checksum (SC) bit to 1, the checksum byte will be sent after the current byte transmission. The SC bit will be cleared after the checksum byte has been transmitted.

NOTE: The TXRDY flag cannot be cleared by reading the corresponding interrupt offset in the SCIINTVECT0 or SCIINTVECT1 register.

29.3.4.2.2 Transmitting Data in Multi-Buffer Mode

Multi-buffer mode is selected when the MBUF MODE bit is set to 1. Like single-buffer mode, you can use the polling, interrupt, or DMA method to write the data to be transmitted. The transmitted data has to be written to the LINTD0 and LINTD1 registers, based on the number of bytes. LIN waits for data to be written to Byte 0 (TD0) of the LINTD0 register and transfers the programmed number of bytes to SCITXSHF to transmit one by one automatically. If the checksum scheme is enabled by setting the Send Checksum (SC) bit to 1, the checksum will be sent after transmission of the last byte of the programmed number of data bytes, indicated by the LENGTH field. The SC bit will be cleared after the checksum byte has been transmitted.

29.4 Low-Power Mode

The SCI/LIN module can be put in either local or global low-power mode. Global low-power mode is asserted by the system and is not controlled by the SCI/LIN module. During global low-power mode, all clocks to the SCI/LIN are turned off so the module is completely inactive. If global low-power mode is requested while the receiver is receiving data, then the SCI/BLIN completes the current reception and then enters the low-power mode, that is, module enters low-power mode only when Busy bit (SCIFLR.3) is cleared.

The BLIN module may enter low-power mode either when there was no activity on the LINRX pin for more than 4s (this can be either a constant recessive or dominant level) or when a Sleep Command frame was received. Once the Timeout flag (SCIFLR.4) was set or once a Sleep Command was received, the POWERDOWN bit (SCIGCR2.0) must be set by the application software to make the module enter local low-power mode. A wakeup signal will terminate the sleep mode of the LIN bus.

NOTE: Enabling Local Low-Power Mode During Receive and Transmit

If the wakeup interrupt is enabled and low-power mode is requested while the receiver is receiving data, then the SCI/LIN immediately generates a wake-up interrupt to clear the powerdown bit. Thus, the SCI/LIN is prevented from entering low-power mode and completes the current reception. Otherwise, if the wakeup interrupt is disabled, the SCI/LIN completes the current reception and then enters the low-power mode.

29.4.1 Entering Sleep Mode

In LIN protocol, a sleep command is used to broadcast the sleep mode to all nodes. The sleep command consists of a diagnostic master request frame with identifier 0x3C (60), with the first data field as 0x00. There should be no activity in the bus once all nodes receive the sleep command: the bus is in sleep mode.

Local low-power mode is asserted by setting the POWERDOWN bit; setting this bit stops the clocks to the SCI/LIN internal logic and registers. Clearing the POWERDOWN bit causes SCI/LIN to exit from local low-power mode. All the registers are accessible during local power-down mode. If a register is accessed in low-power mode, this access results in enabling the clock to the module for that particular access alone.

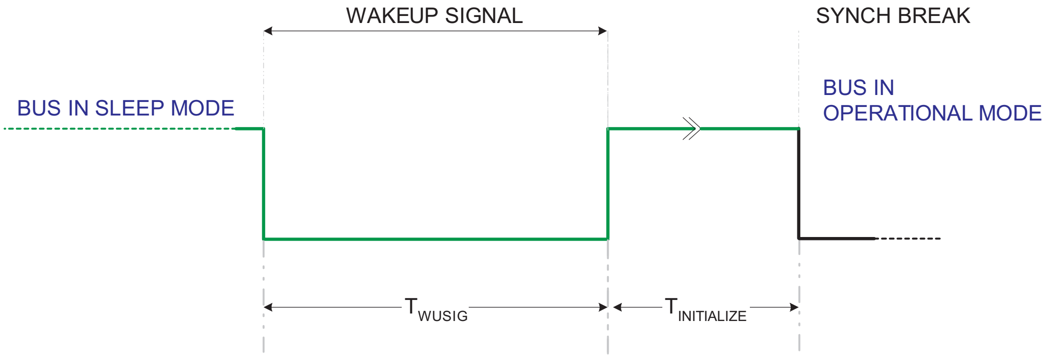
29.4.2 Wakeup

The wakeup interrupt is used to allow the SCI/LIN module to automatically exit low-power mode. A SCI/LIN wakeup is triggered when a low level is detected on the receive RX pin, and this clears the POWERDOWN bit.

NOTE: If the wakeup interrupt is disabled then the SCI/LIN enters low-power mode whenever it is requested to do so, but a low level on the receive RX pin does NOT cause the SCI/LIN to exit low-power mode.

In LIN mode, any node can terminate sleep mode by sending a wakeup signal; see Figure 29-26. A slave node that detects the bus in sleep mode, and with a wakeup request pending, will send a wakeup signal. The wakeup signal is a dominant value on the LIN bus for T_{WUSIG}; this is at least 5 T_{bits} for the LIN bus baud rates. The wakeup signal is generated by sending an 0xF0 byte containing 5 dominant T_{bits} and 5 recessive T_{bits}.

Figure 29-26. Wakeup Signal Generation



$$0.25ms \leq T_{WUSIG} \leq 5ms \tag{51}$$

Assuming a perfect bus with no noise or loading effects, a write of 0xF0 to TD0 will load the transmitter to meet the wakeup signal timing requirement for T_{WUSIG}. Then, setting the GENWU bit will transmit the preloaded value in TD0 for a wakeup signal transmission.

NOTE: The GENWU bit can be set/reset only when SWnRST is set to'1' and the node is in power down mode. The bit will be cleared on a valid synch break detection. A master sending a wakeup request, will exit power down mode upon reception of the wakeup pulse. The bit will be cleared on a SWnRST. This can be used to stop a master from sending further wakeup requests.

The TI TPIC1021 LIN transceiver, upon receiving a wakeup signal, will translate it to the microcontroller for wakeup with a dominant level on the RX pin, or a signal to the voltage regulator. While the POWERDOWN bit is set, if the LIN module detects a recessive-to-dominant edge (falling edge) on the RX pin, it will generate a wakeup interrupt if enabled in the SCISETINT register.

According to LIN protocol 2.0, the TI TPIC1021 LIN transceiver detecting a dominant level on the bus longer than 150 ms will detect it as a wakeup request. The LIN controller's slave is ready to listen to the bus in less than 100 ms ($T_{INITIALIZE}$<100ms) after a dominant-to-recessive edge (end-of-wakeup signal).

29.4.3 *Wakeup Timeouts*

The LIN protocol defines the following timeouts for a wakeup sequence. After a wakeup signal has been sent to the bus, all nodes wait for the master to send a header. If no synch field is detected before 150 ms (3,000 cycles at 20 kHz) after wakeup signal is transmitted, a new wakeup is sent by the same node that requested the first wakeup. This sequence is not repeated more than two times. After three attempts to wake up the LIN bus, wakeup signal generation is suspended for a 1.5 s (30,000 cycles at 20 kHz) period after three breaks.

NOTE: To achieve compatibility to LIN1.3 timeout conditions, the MBRS register must be set to assure that the LIN 2.0 (real-time-based) timings meet the LIN 1.3 bit time base. A node triggering the wakeup should set the MBRS register accordingly to meet the targeted time as 128 Tbits × programmed prescaler.

The LIN controller handles the wakeup expiration times defined by the LIN protocol with a hardware implementation.

29.5 Emulation Mode

In emulation mode, the CONT bit determines how the SCI/LIN operates when the program is suspended. The SCI/LIN counters are affected by this bit during debug mode. when set, the counters are not stopped and when cleared, the counters are stopped debug mode.

Any reads in emulation mode to a SCI/LIN register will not have any effect on the flags in the SCIFLR register.

NOTE: When emulation mode is entered during the Frame transmission or reception of the frame and CONT bit is not set, Communication is not expected to be successful. The suggested usage is to set CONT bit during emulation mode for successful communication.

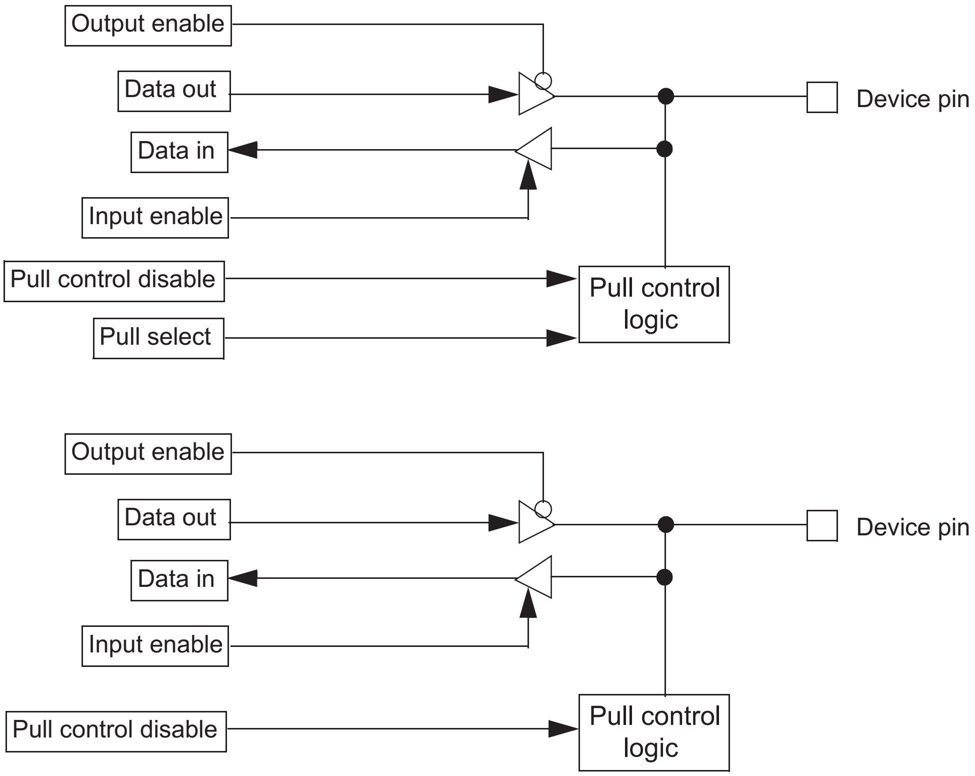
29.6 GPIO Functionality

The following section applies to all device pins that can be configured as functional or general-purpose I/O pins.

29.6.1 GPIO Functionality

Figure 29-27 illustrates the GPIO functionality.

Figure 29-27. GPIO Functionality



29.6.2 Under Reset

The following apply if a device is under reset:

- Pull control. The reset pull control on the pins is enabled.
- Input buffer. The input buffer is enabled.
- Output buffer. The output buffer is disabled.

29.6.3 Out of Reset

The following apply if the device is out of reset:

- Pull control. The pull control is enabled by clearing the PD (pull control disable) bit in the SCIPIO7 register (Section 29.7.21). In this case, if the PSL (pull select) bit in the SCIPIO8 register (Section 29.7.22) is set, the pin will have a pull-up. If the PSL bit is cleared, the pin will have a pull-down. If the PD bit is set in the control register, there is no pull-up or pull-down on the pin.
- Input buffer. The input buffer is permanently enabled in this device.

NOTE: The pull-disable logic depends on the pin direction. It is independent of whether the device is in I/O or functional mode. If the pin is configured as output or transmit, then the pulls are disabled automatically.

- Output buffer. A pin can be driven as an output pin if the TX DIR bit is set in the pin direction control register (SCIPIO1; Section 29.7.15) AND the open-drain feature is not enabled in the SCIPIO6 register (Section 29.7.20).

29.6.4 Open-Drain Feature Enabled on a Pin

The following apply if the open-drain feature is enabled on a pin:

- The output buffer is enabled if a low signal is being driven on to the pin.
- The output buffer is disabled (the direction control signal DIR is internally forced low) if a high signal is being driven on to the pin.

NOTE: The open-drain feature is available only in I/O mode (SCIPIO0; Section 29.7.14).

29.6.5 Summary

The behavior of the input buffer, output buffer, and the pull control is summarized in Table 29-9.

Table 29-9. Input Buffer, Output Buffer, and Pull Control Behavior as GPIO Pins [1]

Device under Reset?	Pin Direction (DIR) [2]	Pull Disable (PULDIS) [3]	Pull Select (PULSEL) [4]	Pull Control	Output Buffer	Input Buffer
Yes	X	X	X	Enabled	Disabled	Enabled
No	0	0	0	Pull down	Disabled	Enabled
No	0	0	1	Pull up	Disabled	Enabled
No	0	1	0	Disabled	Disabled	Enabled
No	0	1	1	Disabled	Disabled	Enabled
No	1	X	X	Disabled	Enabled	Enabled

[1] X = Don’t care

[2] DIR = 0 for input, = 1 for output

[3] PULDIS = 0 for enabling pull control, = 1 for disabling pull control

[4] PULSEL= 0 for pull-down functionality, = 1 for pull-up functionality

29.7 SCI/LIN Control Registers

The SCI/LIN module registers are based on the SCI registers, with added functionality registers enabled by the LIN MODE bit in the SCIGCR1 register.

These registers are accessible in 8-, 16-, and 32-bit reads or writes. The SCI/LIN is controlled and accessed through the registers listed in Table 29-10. Among the features that can be programmed are the LIN protocol mode, communication and timing modes, baud rate value, frame format, DMA requests, and interrupt configuration. The base address for the control registers is FFF7 E400h for LIN1/SCI1 and FFF7 E600h for LIN2/SCI2.

Table 29-10. SCI/LIN Control Registers

Offset	Acronym	Register Description	Section
00h	SCIGCR0	SCI Global Control Register 0	Section 29.7.1
04h	SCIGCR1	SCI Global Control Register 1	Section 29.7.2
08h	SCIGCR2	SCI Global Control Register 2	Section 29.7.3
0Ch	SCISETINT	SCI Set Interrupt Register	Section 29.7.4
10h	SCICLEARINT	SCI Clear Interrupt Register	Section 29.7.5
14h	SCISETINTLVL	SCI Set Interrupt Level Register	Section 29.7.6
18h	SCICLEARINTLVL	SCI Clear Interrupt Level Register	Section 29.7.7
1Ch	SCIFLR	SCI Flags Register	Section 29.7.8
20h	SCIINTVECT0	SCI Interrupt Vector Offset 0	Section 29.7.9
24h	SCIINTVECT1	SCI Interrupt Vector Offset 1	Section 29.7.10
28h	SCIFORMAT	SCI Format Control Register	Section 29.7.11
2Ch	BRS	Baud Rate Selection Register	Section 29.7.12
30h	SCIED	Receiver Emulation Data Buffer	Section 29.7.13.1
34h	SCIRD	Receiver Data Buffer	Section 29.7.13.2
38h	SCITD	Transmit Data Buffer	Section 29.7.13.3
3Ch	SCIPIO0	SCI Pin I/O Control Register 0	Section 29.7.14
40h	SCIPIO1	SCI Pin I/O Control Register 1	Section 29.7.15
44h	SCIPIO2	SCI Pin I/O Control Register 2	Section 29.7.16
48h	SCIPIO3	SCI Pin I/O Control Register 3	Section 29.7.17
4Ch	SCIPIO4	SCI Pin I/O Control Register 4	Section 29.7.18
50h	SCIPIO5	SCI Pin I/O Control Register 5	Section 29.7.19
54h	SCIPIO6	SCI Pin I/O Control Register 6	Section 29.7.20
58h	SCIPIO7	SCI Pin I/O Control Register 7	Section 29.7.21
5Ch	SCIPIO8	SCI Pin I/O Control Register 8	Section 29.7.22
60h	LINCOMPARE	LIN Compare Register	Section 29.7.23
64h	LINRD0	LIN Receive Buffer 0 Register	Section 29.7.24
68h	LINRD1	LIN Receive Buffer 1 Register	Section 29.7.25
6Ch	LINMASK	LIN Mask Register	Section 29.7.26
70h	LINID	LIN Identification Register	Section 29.7.27
74h	LINTD0	LIN Transmit Buffer 0	Section 29.7.28
78h	LINTD1	LIN Transmit Buffer 1	Section 29.7.29
7Ch	MBRS	Maximum Baud Rate Selection Register	Section 29.7.30
90h	IODFTCTRL	Input/Output Error Enable Register	Section 29.7.31

29.7.1 SCI Global Control Register 0 (SCIGCR0)

The SCIGCR0 register defines the module reset. Figure 29-28 and Table 29-11 illustrate this register.

Figure 29-28. SCI Global Control Register 0 (SCIGCR0) (offset = 00)

31 ... 16
Reserved
R-0

15 ... 1	0
Reserved	RESET
R-0	R/WP-0

LEGEND: R = Read only; R/WP = Read/Write in privileged mode only; -*n* = value after reset

Table 29-11. SCI Global Control Register 0 (SCIGCR0) Field Descriptions

Bit	Field	Value	Description
31-1	Reserved	0	Reads return 0. Writes have no effect.
0	RESET		This bit resets the SCI/LIN module. This bit is effective in SCI and LIN mode.
		0	SCI/LIN module is in reset.
		1	SCI/LIN module is out of reset.
			Note: Read/Write in privileged mode only.

29.7.2 SCI Global Control Register 1 (SCIGCR1)

The SCIGCR1 register defines the frame format, protocol, and communication mode used by the SCI. Figure 29-29 and Table 29-12 illustrate this register.

Figure 29-29. SCI Global Control Register 1 (SCIGCR1) (offset = 04h)

31					26	25	24
Reserved						TXENA	RXENA
R-0						R/W-0	R/W-0

23					18	17	16
Reserved						CONT	LOOP BACK
R-0						R/W-0	R/W-0

15	14	13	12	11	10	9	8
Reserved		STOP EXT FRAME	HGEN CTRL	CTYPE	MBUF MODE	ADAPT	SLEEP
R-0		R/WL-0	R/WL-0	R/WL-0	R/W-0	R/WL-0	R/W-0

7	6	5	4	3	2	1	0
SWnRST	LIN MODE	CLOCK	STOP	PARITY	PARITY ENA	TIMING MODE	COMM MODE
R/W-0	R/W-0	R/W-0	R/WC-0	R/WC-0	R/W-0	R/WC-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; WL = Write in LIN mode only; WC = Write in SCI-compatible mode only; -*n* = value after reset

Table 29-12. SCI Global Control Register 1 (SCIGCR1) Field Descriptions

Bit	Field	Value	Description
31-26	Reserved	0	Reads return 0. Writes have no effect.
25	TXENA		Transmit enable. This bit is effective in LIN and SCI modes. Data is transferred from SCITD, or the TDy (with y=0, 1,...7) buffers in LIN mode to the SCITXSHF shift out register only when the TXENA bit is set.
		0	Transfers from SCITD or TDy to SCITXSHF are disabled.
		1	Transfers from SCITD or TDy to SCITXSHF are enabled.
			Note: Data written to SCITD or the transmit multi-buffer before TXENA is set is not transmitted. If TXENA is cleared while transmission is ongoing, the data previously written to SCITD is sent (including the checksum byte in LIN mode).
24	RXENA		Receive enable. This bit is effective in LIN and SCI modes. RXENA allows or prevents the transfer of data from SCIRXSHF to SCIRD or the receive multi-buffers.
		0	The receiver will not transfer data from the shift buffer to the receive buffer or multi-buffers.
		1	The receiver will transfer data from the shift buffer to the receive buffer or multi-buffers.
			Note: Clearing RXENA stops received characters from being transferred into the receive buffer or multi-buffers, prevents the RX status flags from being updated by receive data, and inhibits both receive and error interrupts. However, the shift register continues to assemble data regardless of the state of RXENA. **Note: If RXENA is cleared before a frame is completely received, the data from the frame is not transferred into the receive buffer.** **Note: If RXENA is set before a frame is completely received, the data from the frame is transferred into the receive buffer. If RXENA is set while SCIRXSHF is in the process of assembling a frame, the status flags are not guaranteed to be accurate for that frame. To ensure that the status flags correctly reflect what was detected on the bus during a particular frame, RXENA should be set before the detection of that frame.**
23-18	Reserved	0	Reads return 0. Writes have no effect.
17	CONT		Continue on suspend. This bit is effective in LIN and SCI modes. This bit has an effect only when a program is being debugged with an emulator, and it determines how the SCI/LIN operates when the program is suspended. The SCI/LIN counters are affected by this bit: when the bit is set the counters are not stopped, when the bit is cleared the counters are stopped during debug mode.
		0	When debug mode is entered, the SCI/LIN state machine is frozen. Transmissions and LIN counters are halted and resume when debug mode is exited.
		1	When debug mode is entered, the SCI/LIN continues to operate until the current transmit and receive functions are complete.

Table 29-12. SCI Global Control Register 1 (SCIGCR1) Field Descriptions (continued)

Bit	Field	Value	Description
16	LOOP BACK		Loopback bit. This bit is effective in LIN and SCI modes. The self-checking option for the SCI/LIN can be selected with this bit. If the LINITX and LINRX pins are configured with SCI/LIN functionality, then the LINTX pin is internally connected to the LINRX pin. Externally, during loop back operation, the LINTX pin outputs a high value and the LINRX pin is in a high-impedance state. If this bit value is changed while the SCI/LIN is transmitting or receiving data, errors may result.
		0	Loop back mode is disabled.
		1	Loop back mode is enabled.
15-14	Reserved	0	Reads return 0. Writes have no effect.
13	STOP EXT FRAME		Stop extended frame communication. This bit is effective in LIN mode only. This bit can be written only during extended frame communication. When the extended frame communication is stopped, this bit is cleared automatically.
		0	This bit has no effect.
		1	Extended frame communication will be stopped when current frame transmission/reception is completed.
12	HGEN CTRL		HGEN control. This bit is effective in LIN mode only. This bit controls the type of mask filtering comparison.
		0	ID filtering using the ID-BYTE field in LIN Identification Register (LINID) occurs. Mask of FFh in LIN Mask Register (LINMASK) register will result in no match.
		1	ID filtering uses ID-SlaveTask BYTE (recommended). Mask of FFh in LIN Mask Register (LINMASK) register will result in ALWAYS match. **Note: For software compatibility with future LIN modules the HGEN CTRL bit must be set to 1, the RX ID MASK must be set to FFh and the TX ID MASK must be set to FFh.**
11	CTYPE		Checksum type. This bit is effective in LIN mode only. This bit controls the type of checksum to be used: classic or enhanced.
		0	Classic checksum is used.
		1	Enhanced checksum is used.
10	MBUF MODE		Multi-buffer mode. This bit is effective in LIN and SCI modes. This bit controls receive/transmit buffer usage, that is, whether the RX/TX multi-buffers are used or a single register, RD0/TD0, is used.
		0	The multi-buffer mode is disabled.
		1	The multi-buffer mode is enabled.
9	ADAPT		Adapt. This mode is effective in LIN mode only. This bit has an effect during the detection of the synch field. Two LIN protocol bit rate modes could be enabled with this bit according to the node capability file definition: automatic or select. The software and network configuration will decide which of these two modes are enabled. When this bit is cleared, the LIN 2.0 protocol fixed bit rate should be used. If the ADAPT bit is set, a SCI/LIN slave node detecting the baud rate will compare it to the prescalers in BRS register and update it if they are different. The BRS register will be updated with the new value. If this bit is not set there will be no adjustment to the BRS register.
		0	Automatic baud rate adjustment is disabled.
		1	Automatic baud rate adjustment is enabled.
8	SLEEP		SCI sleep. This bit is effective in SCI mode only. In a multiprocessor configuration, this bit controls the receive sleep function. Clearing this bit brings the SCI/LIN out of sleep mode.
		0	Sleep mode is disabled.
		1	Sleep mode is enabled. **Note: The receiver still operates when the SLEEP bit is set; however, RXRDY is updated and SCIRD is loaded with new data only when an address frame is detected. The remaining receiver status flags are updated and an error interrupt is requested if the corresponding interrupt enable bit is set, regardless of the value of the SLEEP bit. In this way, if an error is detected on the receive data line while the SCI is asleep, software can promptly deal with the error condition.** **Note: The SLEEP bit is *not* automatically cleared when an address byte is detected.** See Section 29.2.5.1 for more information on using the SLEEP bit for multiprocessor communication.

Table 29-12. SCI Global Control Register 1 (SCIGCR1) Field Descriptions (continued)

Bit	Field	Value	Description
7	SWnRST		Software reset (active low). This bit is effective in LIN and SCI modes.
		0	The SCI/LIN is in its reset state; no data will be transmitted or received. Writing a 0 to this bit initializes the SCI/LIN state machines and operating flags as defined in Table 29-13 and Table 29-14. All affected logic is held in the reset state until a 1 is written to this bit.
		1	The SCI/LIN is in its ready state; transmission and reception can be done. After this bit is set to 1, the configuration of the module should not change. **Note: The SCI/LIN should only be configured while SWnRST = 0.**
6	LIN MODE		LIN mode. This bit is effective in LIN and SCI mode. This bit controls the module mode of operation.
		0	LIN mode is disabled; SCI mode is enabled.
		1	LIN mode is enabled; SCI mode is disabled.
5	CLOCK		SCI internal clock enable. The CLOCK bit determines the source of the module clock on the SCICLK pin. It also determines whether a LIN node is a slave or master. **SCI mode:**
		0	The external SCICLK is the clock source. **Note: If an external clock is selected, then the internal baud rate generator and baud rate registers are bypassed. The maximum frequency allowed for an externally sourced SCI clock is VCLK/16.**
		1	The internal SCICLK is the clock source.
			LIN mode:
		0	The node is in slave mode.
		1	The node is in master mode.
4	STOP		SCI number of stop bits per frame. This bit is effective in SCI mode only.
		0	One stop bit is used.
		1	Two stop bits are used. **Note: The receiver checks for only one stop bit. However in idle-line mode, the receiver waits until the end of the second stop bit (if STOP = 1) to begin checking for an idle period**.
3	PARITY		SCI parity odd/even selection. This bit is effective in SCI mode only. If the PARITY ENA bit is set, PARITY designates odd or even parity.
		0	Odd parity is used.
		1	Even parity is used. **The parity bit is calculated based on the data bits in each frame and the address bit (in address-bit mode). The start and stop fields in the frame are not included in the parity calculation.** **For odd parity, the SCI transmits and expects to receive a value in the parity bit that makes odd the total number of bits in the frame with the value of 1.** **For even parity, the SCI transmits and expects to receive a value in the parity bit that makes even the total number of bits in the frame with the value of 1.**
2	PARITY ENA		Parity enable. This bit enables or disables the parity function. **SCI or buffered SCI mode:**
		0	Parity is disabled. No parity bit is generated during transmission or is expected during reception.
		1	Parity is enabled. A parity bit is generated during transmission and is expected during reception.
			LIN mode:
		0	ID field parity verification is disabled.
		1	ID field parity verification is enabled.
1	TIMING MODE		SCI timing mode bit. This bit is effective in SCI mode only. it selects the SCI timing mode.
		0	Synchronous timing is used.
		1	Asynchronous timing is used.

Table 29-12. SCI Global Control Register 1 (SCIGCR1) Field Descriptions (continued)

Bit	Field	Value	Description
0	COMM MODE		SCI/LIN communication mode bit. In compatibility mode it selects the SCI communication mode. In LIN mode it selects length control option for ID-field bits ID4 and ID5.
			SCI mode:
		0	Idle-line mode is used.
		1	Address-bit mode is used.
			LIN mode:
		0	ID4 and ID5 are not used for length control.
		1	ID4 and ID5 are used for length control.

Table 29-13. SCI Receiver Status Flags

SCI Flag	Register	Bit	Value After Reset[1]
CE	SCIFLR	29	0
ISFE	SCIFLR	28	0
NRE	SCIFLR	27	0
FE	SCIFLR	26	0
OE	SCIFLR	25	0
PE	SCIFLR	24	0
RXWAKE	SCIFLR	12	0
RXRDY	SCIFLR	9	0
BUSY	SCIFLR	3	0
IDLE	SCIFLR	2	0
WAKE UP	SCIFLR	1	0
BRKDT	SCIFLR	0	0

[1] The flags are frozen with their reset value while SWnRST = 0.

Table 29-14. SCI Transmitter Status Flags

SCI Flag	Register	Bit	Value After Reset[1]
BE	SCIFLR	31	0
PBE	SCIFLR	30	0
TX WAKE	SCIFLR	10	0
TX EMPTY	SCIFLR	11	1
TXRDY	SCIFLR	8	1

[1] The flags are frozen with their reset value while SWnRST = 0.

29.7.3 SCI Global Control Register 2 (SCIGCR2)

The SCIGCR2 register is used to send or compare a checksum byte during extended frames, to generate a wakeup and for low-power mode control of the LIN module. Figure 29-30 and Table 29-15 illustrate this register.

Figure 29-30. SCI Global Control Register 2 (SCIGCR2) (offset = 08h)

31 18	17	16
Reserved	CC	SC
R-0	R/WL-0	R/WL-0

15 9	8	7 1	0
Reserved	GEN WU	Reserved	POWERDOWN
R-0	R/W-0	R-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; WL = Write in LIN mode only; -*n* = value after reset

Table 29-15. SCI Global Control Register 2 (SCIGCR2) Field Descriptions

Bit	Field	Value	Description
31-18	Reserved	0	Reads return 0. Writes have no effect.
17	CC		Compare checksum. LIN mode only. This bit is used by the receiver for extended frames to trigger a checksum compare. You initiate this transaction by writing a one to this bit. CC bit has to be set only after RX_RDY flag is set for the last received data. In non-multi-buffer mode, when the CC bit is set, the checksum will be compared on the byte that is expected to be the checksum byte. During multi-buffer mode, the following scenarios are associated with the CC bit: a) If the CC bit is set during the reception of the data, then the byte that is received after the reception of the programmed number of data bytes as indicated by SCIFORMAT[18:16] is treated as a checksum byte. b) If the CC bit is set during the idle period (that is, during the inter-frame space), then the immediate next byte will be treated as a checksum byte. c) CC bit will be auto cleared after the checkbyte has been received and compared. Checksum reception is not guaranteed if CC bit is write cleared by software during the checksum reception. See Section 29.3.1.6 for more details.
		0	No checksum compare will occur.
		1	Compare checksum on expected checksum byte.
16	SC		Send checksum byte. This bit is effective in LIN mode only. This bit is used by the transmitter with extended frames to send a checksum byte. In non-multi-buffer mode, the checksum byte will be sent after the current byte transmission. In multi-buffer mode, the checksum byte will be sent after the last byte count, indicated by the SCIFORMAT[18:16]). See Section 29.3.1.6 for more details. This byte will be cleared after the checksum byte has been transmitted. In non-multi-buffer mode, the checksum byte will be sent after the current byte transmission. During multi-buffer mode, the following scenarios are associated with the SC bit: a) The checkbyte will be sent after the last byte count, indicated by the SCIFORMAT[18:16]). b) Checksum will not be sent if SC is set before transmitting the very first byte(that is, during interframe space). c) SC bit will be auto cleared after the checkbyte has been transmitted. Checksum transmission is not guaranteed if SC bit is write cleared by software during the checksum transmission. See Section 29.3.1.6 for more details.
		0	No checksum byte will be sent.
		1	A checksum byte will be sent.
15-9	Reserved	0	Reads return 0. Writes have no effect.
8	GEN WU		Generate wakeup signal. This bit is effective in LIN mode only. This bit controls the generation of a wakeup signal, by transmitting the TDO buffer value. The LIN protocol specifies that this signal should be a dominant for T_{WUSIG}. This bit is cleared on reception of a valid synch break.
		0	No wakeup signal will be generated.
		1	The TDO buffer value will be transmitted for a wakeup signal. The bit will be cleared on a SWnRST.
7-1	Reserved	0	Reads return 0. Writes have no effect.

Table 29-15. SCI Global Control Register 2 (SCIGCR2) Field Descriptions (continued)

Bit	Field	Value	Description
0	POWERDOWN		Power down. This bit is effective in LIN or SCI mode. When this bit is set, the SCI/LIN module attempts to enter local low-power mode. If the POWERDOWN bit is set while the receiver is actively receiving data and the wakeup interrupt is disabled, then the SCI/LIN will delay entering low-power mode until the reception is completed. In LIN mode, you may set the POWERDOWN bit after receiving a sleep command or on idle bus detection (more than 4 seconds). See Section 29.4 for more information on low-power mode.
		0	The SCI/LIN module is in normal operation.
		1	The SCI/LIN module enters local low-power mode.

29.7.4 SCI Set Interrupt Register (SCISETINT)

Figure 29-31 and Table 29-16 illustrate this register. Refer to Figure 29-31 for details on when different interrupt flags get set in a frame during LIN Mode.

Figure 29-31. SCI Set Interrupt Register (SCISETINT) (offset = 0Ch)

31	30	29	28	27	26	25	24
SETBE INT	SET PBE INT	SET CE INT	SET ISFE INT	SET NRE INT	SET FE INT	SET OE INT	SET PE INT
R/WL-0	R/WL-0	R/WL-0	R/WL-0	R/WL-0	R/W-0	R/W-0	R/W-0

23				19	18	17	16
Reserved					SET RX DMA ALL	SET RX DMA	SET TX DMA
R-0					R/WC-0	R/W-0	R/W-0

15	14	13	12		10	9	8
Reserved		SET ID INT	Reserved			SET RX INT	SET TX INT
R-0		R/WL-0	R-0			R/W-0	R/W-0

7	6	5	4	3	2	1	0
SET TOA3WUS INT	SET TOAWUS INT	Reserved	SET TIMEOUT INT	Reserved		SET WAKEUP INT	SET BRKDT INT
R/WL-0	R/WL-0	R-0	R/WL-0	R-0		R/W-0	R/WC-0

LEGEND: R/W = Read/Write; R = Read only; WL = Write in LIN mode only; WC = Write in SCI-compatible mode only; -*n* = value after reset

Table 29-16. SCI Set Interrupt Register (SCISETINT) Field Descriptions

Bit	Field	Value	Description
31	SET BE INT		Set bit error interrupt. This bit is effective in LIN mode only. Setting this bit enables the SCI/LIN module to generate an interrupt when there is a bit error.
		0	*Read:* The interrupt is disabled. *Write:* No effect.
		1	*Read or write:* The interrupt is enabled.
30	SET PBE INT		Set physical bus error interrupt. This bit is effective in LIN mode only. Setting this bit enables the SCI/LIN module to generate an interrupt when a physical bus error occurs.
		0	*Read:* The interrupt is disabled. *Write:* No effect.
		1	*Read or write:* The interrupt is enabled.
29	SET CE INT		Set checksum-error interrupt. This bit is effective in LIN mode only. Setting this bit enables the SCI/LIN module to generate an interrupt when there is a checksum error.
		0	*Read:* The interrupt is disabled. *Write:* No effect.
		1	*Read or write:* The interrupt is enabled.
28	SET ISFE INT		Set inconsistent-synch-field-error interrupt. This bit is effective in LIN mode only. Setting this bit enables the SCI/LIN module to generate an interrupt when there is an inconsistent synch field error.
		0	*Read:* The interrupt is disabled. *Write:* No effect.
		1	*Read or write:* The interrupt is enabled.
27	SET NRE INT		Set no-response-error interrupt. This bit is effective in LIN mode only. Setting this bit enables the SCI/LIN module to generate an interrupt when a no-response error occurs.
		0	*Read:* The interrupt is disabled. *Write:* No effect.
		1	*Read or write:* The interrupt is enabled.

Table 29-16. SCI Set Interrupt Register (SCISETINT) Field Descriptions (continued)

Bit	Field	Value	Description
26	SET FE INT		Set framing-error interrupt. This bit is effective in LIN or SCI-compatible mode. Setting this bit enables the SCI/LIN module to generate an interrupt when a framing error occurs.
		0	*Read:* The interrupt is disabled. *Write:* No effect.
		1	*Read or write:* The interrupt is enabled.
25	SET OE INT		Set overrun-error interrupt. This bit is effective in LIN or SCI-compatible mode. Setting this bit enables the SCI/LIN module to generate an interrupt when an overrun error occurs.
		0	*Read:* The interrupt is disabled. *Write:* No effect.
		1	*Read or write:* The interrupt is enabled.
24	SET PE INT		Set parity interrupt. This bit is effective in LIN or SCI-compatible mode. Setting this bit enables the SCI/LIN module to generate an interrupt when a parity error occurs.
		0	*Read:* The interrupt is disabled. *Write:* No effect.
		1	*Read or write:* The interrupt is enabled.
23-19	Reserved	0	Reads return 0. Writes have no effect.
18	SET RX DMA ALL		Set receive DMA all. This bit is effective in SCI-compatible mode only. This bit determines if a separate interrupt is generated for the address frames sent in multiprocessor communications. When this bit is 0, RX interrupt requests are generated for address frames and DMA requests are generated for data frames. When this bit is 1, RX DMA requests are generated for both address and data frames.
		0	*Read:* The DMA request is disabled for address frames (the receive interrupt request is enabled for address frames). *Write:* No effect.
		1	*Read and write:* The DMA request is enabled for address and data frames.
17	SET RX DMA		Set receiver DMA. This bit is effective in LIN or SCI-compatible mode. To enable receiver DMA requests, this bit must be set. If it is cleared, interrupt requests are generated depending on SET RX INT bit (SCISETINT).
		0	*Read:* The DMA request is disabled. *Write:* No effect.
		1	*Read and write:* The DMA request is enabled for address and data frames.
16	SET TX DMA		Set transmit DMA. This bit is effective in LIN or SCI-compatible mode. To enable DMA requests for the transmitter, this bit must be set. If it is cleared, interrupt requests are generated depending on SET TX INT bit (SCISETINT).
		0	*Read:* Transmit DMA request is disabled. *Write:* No effect.
		1	*Read and write:* Transmit DMA request is enabled.
15-14	Reserved	0	Reads return 0. Writes have no effect.
13	SET ID INT		Set identification interrupt. This bit is effective in LIN mode only. This bit is set to enable an interrupt when a valid matching identifier is received. See Section 29.3.1.9 for more details.
		0	*Read:* The interrupt is disabled. *Write:* No effect.
		1	*Read or write:* The interrupt is enabled.
12-10	Reserved	0	Reads return 0. Writes have no effect.
9	SET RX INT		Receiver interrupt enable. This bit is effective in LIN or SCI-compatible mode. Setting this bit enables the SCI/LIN to generate a receive interrupt after a frame has been completely received and the data is being transferred from SCIRXSHF to SCIRD.
		0	*Read:* The interrupt is disabled. *Write:* No effect.
		1	*Read or write:* The interrupt is enabled.

Table 29-16. SCI Set Interrupt Register (SCISETINT) Field Descriptions (continued)

Bit	Field	Value	Description
8	SET TX INT	0	Set transmitter interrupt. This bit is effective in LIN or SCI-compatible mode. Setting this bit enables the SCI/LIN to generate a transmit interrupt as data is being transferred from SCITD to SCITXSHF and the TXRDY bit is being set. *Read:* The interrupt is disabled. *Write:* No effect.
		1	*Read or write:* The interrupt is enabled.
7	SET TOA3WUS INT	0	Set timeout after three wakeup signals interrupt. This bit is effective in LIN mode only. Setting this bit enables the SCI/LIN to generate an interrupt when a timeout occurs after three wakeup signals have been sent. *Read:* The interrupt is disabled. *Write:* No effect.
		1	*Read or write:* The interrupt is enabled.
6	SET TOAWUS INT	0	Set timeout after wakeup signal interrupt. This bit is effective in LIN mode only. Setting this bit enables the SCI/LIN to generate an interrupt when a timeout occurs after one wakeup signal has been sent. *Read:* The interrupt is disabled. *Write:* No effect.
		1	*Read or write:* The interrupt is enabled.
5	Reserved	0	Reads return 0. Writes have no effect.
4	SET TIMEOUT INT	0	Set timeout interrupt. This bit is effective in LIN mode only. Setting this bit enables the SCI/LIN to generate an interrupt when there is no LIN bus activity (bus idle) for at least four seconds. *Read:* The interrupt is disabled. *Write:* No effect.
		1	*Read or write:* The interrupt is enabled.
3-2	Reserved	0	Reads return 0. Writes have no effect.
1	SET WAKEUP INT	0	Set wake-up interrupt. This bit is effective in LIN or SCI-compatible mode. Setting this bit enables the SCI/LIN to generate a wakeup interrupt and thereby exit low-power mode. If enabled, the wakeup interrupt is asserted when local low-power mode is requested while the receiver is busy or if a low level is detected on the LINRX pin during low-power mode. *Read:* The interrupt is disabled. *Write:* No effect.
		1	*Read or write:* The interrupt is enabled.
0	SET BRKDT INT	0	Set break-detect interrupt. This bit is effective in SCI-compatible mode only. Setting this bit enables the SCI/LIN to generate an error interrupt if a break condition is detected on the LINRX pin. *Read:* The interrupt is disabled. *Write:* No effect.
		1	*Read or write:* The interrupt is enabled.

29.7.5 SCI Clear Interrupt Register (SCICLEARINT)

Figure 29-32 and Table 29-17 illustrate this register. SCICLEARINT register is used to clear the enabled interrupts without accessing SCISETINT register.

Figure 29-32. SCI Clear Interrupt Register (SCICLEARINT) (offset = 10h)

31	30	29	28	27	26	25	24
CLR BE INT	CLR PBE INT	CLR CE INT	CLR ISFE INT	CLR RE INT	CLR FE INT	CLR OE INT	CLR PE INT
R/WL-0	R/WL-0	R/WL-0	R/WL-0	R/WL-0	R/W-0	R/W-0	R/W-0

23				19	18	17	16
Reserved					CLR RX DMA ALL	CLR RX DMA	CLR TX DMA
R-0					R/WC-0	R/W-0	R/W-0

15	14	13	12		10	9	8
Reserved		CLR ID INT	Reserved			CLR RX INT	CLR TX INT
R-0		R/WL-0	R-0			R/W-0	R/W-0

7	6	5	4	3	2	1	0
CLR TOA3WUS INT	CLR TOAWUS INT	Reserved	CLR TIMEOUT INT	Reserved		CLR WAKEUP INT	CLR BRKDT INT
R/WL-0	R/WL-0	R-0	R/WL-0	R-0		R/W-0	R/WC-0

LEGEND: R/W = Read/Write; R = Read only; WL = Write in LIN mode only; WC = Write in SCI-compatible mode only; -*n* = value after reset

Table 29-17. SCI Clear Interrupt Register (SCICLEARINT) Field Descriptions

Bit	Field	Value	Description
31	CLR BE INT		Clear bit error interrupt. This bit is effective in LIN mode only. This bit disables the bit error interrupt when set.
		0	*Read:* The interrupt is disabled. *Write:* No effect.
		1	*Read:* The interrupt is enabled. *Write:* The interrupt is disabled.
30	CLR PBE INT		Clear physical bus error interrupt. This bit is effective in LIN mode only. This bit disables the physical-bus error interrupt when set.
		0	*Read:* The interrupt is disabled. *Write:* No effect.
		1	*Read:* The interrupt is enabled. *Write:* The interrupt is disabled.
29	CLR CE INT		Clear checksum-error interrupt. This bit is effective in LIN mode only. This bit disables the checksum interrupt when set.
		0	*Read:* The interrupt is disabled. *Write:* No effect.
		1	*Read:* The interrupt is enabled. *Write:* The interrupt is disabled.
28	CLR ISFE INT		Clear inconsistent-synch-field-error (ISFE) interrupt. This bit is effective in LIN mode only. This bit disables the ISFE interrupt when set.
		0	*Read:* The interrupt is disabled. *Write:* No effect.
		1	*Read:* The interrupt is enabled. *Write:* The interrupt is disabled.

Table 29-17. SCI Clear Interrupt Register (SCICLEARINT) Field Descriptions (continued)

Bit	Field	Value	Description
27	CLR NRE INT		Clear no-response-error interrupt. This bit is effective in LIN mode only. This bit disables the NRE interrupt when set.
		0	*Read:* The interrupt is disabled. *Write:* No effect.
		1	*Read:* The interrupt is enabled. *Write:* The interrupt is disabled.
26	CLR FE INT		Clear framing-error interrupt. This bit is effective in LIN or SCI mode. This bit disables the framing-error interrupt when set.
		0	*Read:* The interrupt is disabled. *Write:* No effect.
		1	*Read:* The interrupt is enabled. *Write:* The interrupt is disabled.
25	CLR OE INT		Clear overrun-error interrupt. This bit is effective in LIN or SCI mode. This bit disables the SCI/LIN overrun error interrupt when set.
		0	*Read:* The interrupt is disabled. *Write:* No effect.
		1	*Read:* The interrupt is enabled. *Write:* The interrupt is disabled.
24	CLR PE INT		Clear parity interrupt. This bit is effective in LIN or SCI mode. This bit disables the parity error interrupt when set.
		0	*Read:* The interrupt is disabled. *Write:* No effect.
		1	*Read:* The interrupt is enabled. *Write:* The interrupt is disabled.
23-19	Reserved	0	Reads return 0. Writes have no effect.
18	CLR RX DMA ALL		Clear receive DMA all. This bit is effective in SCI mode only. This bit clears the receive DMA request for address frames when set. Only receive data frames generate a DMA request.
		0	*Read:* The interrupt is disabled. *Write:* No effect.
		1	*Read:* The receive DMA request for address and data frames is enabled. *Write:* The receive DMA request for address and data frames is disabled.
17	CLR RX DMA		Clear receive DMA request. This bit is effective in LIN or SCI mode. This bit disables the receive DMA request when set.
		0	*Read:* The receive DMA request is disabled. *Write:* No effect.
		1	*Read:* The receive DMA request is enabled. *Write:* The receive DMA request for is disabled.
16	CLR TX DMA		Clear transmit DMA request. This bit is effective in LIN or SCI mode. This bit disables the transmit DMA request when set.
		0	*Read:* The transmit DMA request is disabled. *Write:* No effect.
		1	*Read:* The transmit DMA request is enabled. *Write:* The transmit DMA request for is disabled.
15-14	Reserved	0	Reads return 0. Writes have no effect.
13	CLR ID INT		Clear ID interrupt. This bit is effective in LIN mode only. This bit disables the ID interrupt when set.
		0	*Read:* The interrupt is disabled. *Write:* No effect.
		1	*Read:* The interrupt is enabled. *Write:* The interrupt is disabled.

Table 29-17. SCI Clear Interrupt Register (SCICLEARINT) Field Descriptions (continued)

Bit	Field	Value	Description
12-10	Reserved	0	Reads return 0. Writes have no effect.
9	CLR RX INT		Clear receiver interrupt. This bit is effective in LIN or SCI mode. This bit disables the receiver interrupt when set.
		0	*Read:* The interrupt is disabled. *Write:* No effect.
		1	*Read:* The interrupt is enabled. *Write:* The interrupt is disabled.
8	CLR TX INT		Clear transmitter interrupt. This bit is effective in LIN or SCI mode. This bit disables the transmitter interrupt when set.
		0	*Read:* The interrupt is disabled. *Write:* No effect.
		1	*Read:* The interrupt is enabled. *Write:* The interrupt is disabled.
7	CLR TOA3WUS INT		Clear timeout after three wakeup signals interrupt. This bit is effective in LIN mode only. This bit disables the timeout after three wakeup signals interrupt when set.
		0	*Read:* The interrupt is disabled. *Write:* No effect.
		1	*Read:* The interrupt is enabled. *Write:* The interrupt is disabled.
6	CLR TOAWUS INT		Clear timeout after wakeup signal interrupt. This bit is effective in LIN mode only. This bit disables the timeout after one wakeup signal interrupt when set.
		0	*Read:* The interrupt is disabled. *Write:* No effect.
		1	*Read:* The interrupt is enabled. *Write:* The interrupt is disabled.
5	Reserved	0	Reads return 0. Writes have no effect.
4	CLR TIMEOUT INT		Clear timeout interrupt. This bit is effective in LIN mode only. This bit disables the timeout (LIN bus idle) interrupt when set.
		0	*Read:* The interrupt is disabled. *Write:* No effect.
		1	*Read:* The interrupt is enabled. *Write:* The interrupt is disabled.
3-2	Reserved	0	Reads return 0. Writes have no effect.
1	CLR WAKEUP INT		Clear wake-up interrupt. This bit is effective in LIN or SCI-compatible mode. This bit disables the wakeup interrupt when set.
		0	*Read:* The interrupt is disabled. *Write:* No effect.
		1	*Read:* The interrupt is enabled. *Write:* The interrupt is disabled.
0	CLR BRKDT INT		Clear break-detect interrupt. This bit is effective in SCI-compatible mode only. This bit disables the break-detect interrupt when set.
		0	*Read:* The interrupt is disabled. *Write:* No effect.
		1	*Read:* The interrupt is enabled. *Write:* The interrupt is disabled.

29.7.6 SCI Set Interrupt Level Register (SCISETINTLVL)

Figure 29-33 and Table 29-18 illustrate this register.

Figure 29-33. SCI Set Interrupt Level Register (SCISETINTLVL) (offset = 14h)

31	30	29	28	27	26	25	24
SET BE INT LVL	SET PBE INT LVL	SET CE INT LVL	SET ISFE INT LVL	SET NRE INT LVL	SET FE INT LVL	SET OE INT LVL	SET PE INT LVL
R/WL-0	R/WL-0	R/WL-0	R/WL-0	R/WL-0	R/W-0	R/W-0	R/W-0

23				19	18	17	16
Reserved					SET RX DMA ALL INT LVL	Reserved	
R-0					R/WC-0	R-0	

15	14	13	12		10	9	8
Reserved		SET ID INT LVL	Reserved			SET RX INT LVL	SET TX INT LVL
R-0		R/WL-0	R-0			R/W-0	R/W-0

7	6	5	4	3	2	1	0
SET TOA3WUS INT LVL	SET TOAWUS INT LVL	Reserved	SET TIMEOUT INT LVL	Reserved		SET WAKEUP INT LVL	SET BRKDT INT LVL
R/WL-0	R/WL-0	R-0	R/WL-0	R-0		R/W-0	R/WC-0

LEGEND: R/W = Read/Write; R = Read only; WL = Write in LIN mode only; WC = Write in SCI-compatible mode only; -*n* = value after reset

Table 29-18. SCI Set Interrupt Level Register (SCISETINTLVL) Field Descriptions

Bit	Field	Value	Description
31	SET BE INT LVL		Set bit error interrupt level. This bit is effective in LIN mode only.
		0	*Read:* The interrupt level is mapped to the INT0 line. *Write:* No effect.
		1	*Read or write:* The interrupt level is mapped to the INT1 line.
30	SET PBE INT LVL		Set physical bus error interrupt level. This bit is effective in LIN mode only.
		0	*Read:* The interrupt level is mapped to the INT0 line. *Write:* No effect.
		1	*Read or write:* The interrupt level is mapped to the INT1 line.
29	SET CE INT LVL		Set checksum-error interrupt level. This bit is effective in LIN mode only.
		0	*Read:* The interrupt level is mapped to the INT0 line. *Write:* No effect.
		1	*Read or write:* The interrupt level is mapped to the INT1 line.
28	SET ISFE INT LVL		Set inconsistent-synch-field-error interrupt level. This bit is effective in LIN mode only.
		0	*Read:* The interrupt level is mapped to the INT0 line. *Write:* No effect.
		1	*Read or write:* The interrupt level is mapped to the INT1 line.
27	SET NRE INT LVL		Set no-response-error interrupt level. This bit is effective in LIN mode only.
		0	*Read:* The interrupt level is mapped to the INT0 line. *Write:* No effect.
		1	*Read or write:* The interrupt level is mapped to the INT1 line.
26	SET FE INT LVL		Set framing-error interrupt level. This bit is effective in LIN or SCI-compatible mode.
		0	*Read:* The interrupt level is mapped to the INT0 line. *Write:* No effect.
		1	*Read or write:* The interrupt level is mapped to the INT1 line.

Table 29-18. SCI Set Interrupt Level Register (SCISETINTLVL) Field Descriptions (continued)

Bit	Field	Value	Description
25	SET OE INT LVL		Set overrun-error interrupt level. This bit is effective in LIN or SCI-compatible mode.
		0	*Read:* The interrupt level is mapped to the INT0 line. *Write:* No effect.
		1	*Read or write:* The interrupt level is mapped to the INT1 line.
24	SET PE INT LVL		Set parity error interrupt level. This bit is effective in LIN or SCI-compatible mode.
		0	*Read:* The interrupt level is mapped to the INT0 line. *Write:* No effect.
		1	*Read or write:* The interrupt level is mapped to the INT1 line.
23-19	Reserved	0	Reads return 0. Writes have no effect.
18	SET RX DMA ALL LVL		Set receive DMA all interrupt levels. This bit is effective in SCI mode only.
		0	*Read:*The receive interrupt request for address frames is mapped to the INT0 line. *Write:* No effect.
		1	*Read or write:* The receive interrupt request for address frames is mapped to the INT1 line.
17-14	Reserved	0	Reads return 0. Writes have no effect.
13	SET ID INT LVL		Set ID interrupt level. This bit is effective in LIN mode only.
		0	*Read:* The interrupt level is mapped to the INT0 line. *Write:* No effect.
		1	*Read or write:* The interrupt level is mapped to the INT1 line.
12-10	Reserved	0	Reads return 0. Writes have no effect.
9	SET RX INT LVL		Set receiver interrupt level. This bit is effective in LIN or SCI-compatible mode.
		0	*Read:* The interrupt level is mapped to the INT0 line. *Write:* No effect.
		1	*Read or write:* The interrupt level is mapped to the INT1 line.
8	SET TX INT LVL		Set transmitter interrupt level. This bit is effective in LIN or SCI-compatible mode.
		0	*Read:* The interrupt level is mapped to the INT0 line. *Write:* No effect.
		1	*Read or write:* The interrupt level is mapped to the INT1 line.
7	SET TOA3WUS INT LVL		Set timeout after three wakeup signals interrupt level. This bit is effective in LIN mode only.
		0	*Read:* The interrupt level is mapped to the INT0 line. *Write:* No effect.
		1	*Read or write:* The interrupt level is mapped to the INT1 line.
6	SET TOAWUS INT LVL		Set timeout after wakeup signal interrupt level. This bit is effective in LIN mode only.
		0	*Read:* The interrupt level is mapped to the INT0 line. *Write:* No effect.
		1	*Read or write:* The interrupt level is mapped to the INT1 line.
5	Reserved	0	Reads return 0. Writes have no effect.
4	SET TIMEOUT INT LVL		Set timeout interrupt level. This bit is effective in LIN mode only.
		0	*Read:* The interrupt level is mapped to the INT0 line. *Write:* No effect.
		1	*Read or write:* The interrupt level is mapped to the INT1 line.
3-2	Reserved	0	Reads return 0. Writes have no effect.
1	SET WAKEUP INT LVL		Set wake-up interrupt level. This bit is effective in LIN or SCI-compatible mode.
		0	*Read:* The interrupt level is mapped to the INT0 line. *Write:* No effect.
		1	*Read or write:* The interrupt level is mapped to the INT1 line.

Table 29-18. SCI Set Interrupt Level Register (SCISETINTLVL) Field Descriptions (continued)

Bit	Field	Value	Description
0	SET BRKDT INT LVL		Set break-detect interrupt level. This bit is effective in SCI-compatible mode only.
		0	*Read:* The interrupt level is mapped to the INT0 line. *Write:* No effect.
		1	*Read or write:* The interrupt level is mapped to the INT1 line.

29.7.7 SCI Clear Interrupt Level Register (SCICLEARINTLVL)

Figure 29-34 and Table 29-19 illustrate this register.

Figure 29-34. SCI Clear Interrupt Level Register (SCICLEARINTLVL) (offset = 18h)

31	30	29	28	27	26	25	24
CLR BE INT LVL	CLR PBE INT LVL	CLR CE INT LVL	CLR ISFE INT LVL	CLR NRE INT LVL	CLR FE INT LVL	CLR OE INT LVL	CLR PE INT LVL
R/WL-0	R/WL-0	R/WL-0	R/WL-0	R/WL-0	R/W-0	R/W-0	R/W-0

23				19	18	17	16
Reserved					CLR RX DMA ALL INT LVL	Reserved	
R-0					R/WC-0	R-0	

15	14	13	12		10	9	8
Reserved		CLR ID INT LVL	Reserved			CLR RX INT LVL	CLR TX INT LVL
R-0		R/WL-0	R-0			R/W-0	R/W-0

7	6	5	4	3	2	1	0
CLR TOA3WUS INT LVL	CLR TOAWUS INT LVL	Reserved	CLR TIMEOUT INT LVL	Reserved		CLR WAKEUP INT LVL	CLR BRKDT INT LVL
R/WL-0	R/WL-0	R-0	R/WL-0	R-0		R/W-0	R/WC-0

LEGEND: R/W = Read/Write; R = Read only; WL = Write in LIN mode only; WC = Write in SCI-compatible mode only; -*n* = value after reset

Table 29-19. SCI Clear Interrupt Level Register (SCICLEARINTLVL) Field Descriptions

Bit	Field	Value	Description
31	CLR BE INT LVL		Clear bit error interrupt. This bit is effective in LIN mode only.
		0	*Read:* The interrupt level is mapped to the INT0 line. *Write:* No effect.
		1	*Read:* The interrupt level is mapped to the INT1 line. *Write:* The interrupt level is mapped to the INT0 line.
30	CLR PBE INT LVL		Clear physical bus error interrupt. This bit is effective in LIN mode only.
		0	*Read:* The interrupt level is mapped to the INT0 line. *Write:* No effect.
		1	*Read:* The interrupt level is mapped to the INT1 line. *Write:* The interrupt level is mapped to the INT0 line.
29	CLR CE INT LVL		Clear checksum-error interrupt. This bit is effective in LIN mode only.
		0	*Read:* The interrupt level is mapped to the INT0 line. *Write:* No effect.
		1	*Read:* The interrupt level is mapped to the INT1 line. *Write:* The interrupt level is mapped to the INT0 line.
28	CLR ISFE INT LVL		Clear inconsistent-synch-field-error (ISFE) interrupt. This bit is effective in LIN mode only.
		0	*Read:* The interrupt level is mapped to the INT0 line. *Write:* No effect.
		1	*Read:* The interrupt level is mapped to the INT1 line. *Write:* The interrupt level is mapped to the INT0 line.
27	CLR NRE INT LVL		Clear no-response-error interrupt. This bit is effective in LIN mode only.
		0	*Read:* The interrupt level is mapped to the INT0 line. *Write:* No effect.
		1	*Read:* The interrupt level is mapped to the INT1 line. *Write:* The interrupt level is mapped to the INT0 line.

Table 29-19. SCI Clear Interrupt Level Register (SCICLEARINTLVL) Field Descriptions (continued)

Bit	Field	Value	Description
26	CLR FE INT LVL	0	Clear framing-error interrupt. This bit is effective in LIN or SCI-compatible mode. *Read:* The interrupt level is mapped to the INT0 line. *Write:* No effect.
		1	*Read:* The interrupt level is mapped to the INT1 line. *Write:* The interrupt level is mapped to the INT0 line.
25	CLR OE INT LVL	0	Clear overrun-error interrupt. This bit is effective in LIN or SCI-compatible mode. *Read:* The interrupt level is mapped to the INT0 line. *Write:* No effect.
		1	*Read:* The interrupt level is mapped to the INT1 line. *Write:* The interrupt level is mapped to the INT0 line.
24	CLR PE INT LVL	0	Clear parity interrupt. This bit is effective in LIN or SCI-compatible mode. *Read:* The interrupt level is mapped to the INT0 line. *Write:* No effect.
		1	*Read:* The interrupt level is mapped to the INT1 line. *Write:* The interrupt level is mapped to the INT0 line.
23-19	Reserved	0	Reads return 0. Writes have no effect.
18	CLR RX DMA ALL LVL	0	Clear receive DMA interrupt level. This bit is effective in SCI-compatible mode only. *Read:* The receive interrupt request for address frames is mapped to the INT0 line. *Write:* No effect.
		1	*Read:* The receive interrupt request for address frames is mapped to the INT1 line. *Write:* The receive interrupt request for address frames is mapped to the INT0 line.
17-14	Reserved	0	Reads return 0. Writes have no effect.
13	CLR ID INT LVL	0	Clear ID interrupt. This bit is effective in LIN mode only. *Read:* The interrupt level is mapped to the INT0 line. *Write:* No effect.
		1	*Read:* The interrupt level is mapped to the INT1 line. *Write:* The interrupt level is mapped to the INT0 line.
12-10	Reserved	0	Reads return 0. Writes have no effect.
9	CLR RX INT LVL	0	Clear receiver interrupt. This bit is effective in LIN or SCI-compatible mode. *Read:* The interrupt level is mapped to the INT0 line. *Write:* No effect.
		1	*Read:* The interrupt level is mapped to the INT1 line. *Write:* The interrupt level is mapped to the INT0 line.
8	CLR TX INT LVL	0	Clear transmitter interrupt. This bit is effective in LIN or SCI-compatible mode. *Read:* The interrupt level is mapped to the INT0 line. *Write:* No effect.
		1	*Read:* The interrupt level is mapped to the INT1 line. *Write:* The interrupt level is mapped to the INT0 line.
7	CLR TOA3WUS INT LVL	0	Clear timeout after three wakeup signals interrupt. This bit is effective in LIN mode only. *Read:* The interrupt level is mapped to the INT0 line. *Write:* No effect.
		1	*Read:* The interrupt level is mapped to the INT1 line. *Write:* The interrupt level is mapped to the INT0 line.

Table 29-19. SCI Clear Interrupt Level Register (SCICLEARINTLVL) Field Descriptions (continued)

Bit	Field	Value	Description
6	CLR TOAWUS INT LVL		Clear timeout after wakeup signal interrupt. This bit is effective in LIN mode only.
		0	*Read:* The interrupt level is mapped to the INT0 line. *Write:* No effect.
		1	*Read:* The interrupt level is mapped to the INT1 line. *Write:* The interrupt level is mapped to the INT0 line.
5	Reserved	0	Reads return 0. Writes have no effect.
4	CLR TIMEOUT INT LVL		Clear timeout interrupt. This bit is effective in LIN mode only.
		0	*Read:* The interrupt level is mapped to the INT0 line. *Write:* No effect.
		1	*Read:* The interrupt level is mapped to the INT1 line. *Write:* The interrupt level is mapped to the INT0 line.
3-2	Reserved	0	Reads return 0. Writes have no effect.
1	CLR WAKEUP INT LVL		Clear wake-up interrupt. This bit is effective in LIN or SCI-compatible mode.
		0	*Read:* The interrupt level is mapped to the INT0 line. *Write:* No effect.
		1	*Read:* The interrupt level is mapped to the INT1 line. *Write:* The interrupt level is mapped to the INT0 line.
0	CLR BRKDT INT LVL		Clear break-detect interrupt. This bit is effective in SCI-compatible mode only.
		0	*Read:* The interrupt level is mapped to the INT0 line. *Write:* No effect.
		1	*Read:* The interrupt level is mapped to the INT1 line. *Write:* The interrupt level is mapped to the INT0 line.

29.7.8 SCI Flags Register (SCIFLR)

Figure 29-35 and Table 29-20 illustrate this register.

Figure 29-35. SCI Flags Register (SCIFLR) (offset = 1Ch)

31	30	29	28	27	26	25	24
BE	PBE	CE	ISFE	NRE	FE	OE	PE
R/WL-0	R/WL-0	R/WL-0	R/WL-0	R/WL-0	R/W-0	R/W-0	R/W-0

23	16
Reserved	
R-0	

15	14	13	12	11	10	9	8
Reserved	ID RX	ID TX	RX WAKE	TX EMPTY	TX WAKE	RX RDY	TX RDY
R-0	R/WL-0	R/WL-0	R/WC-0	R/W-1	R/WC-0	R/W-0	R/W-1

7	6	5	4	3	2	1	0
TOA3WUS	TOAWUS	Reserved	TIMEOUT	BUSY	IDLE	WAKE UP	BRKDT
R/WL-0	R/WL-0	R-0	R/WL-0	R/W-0	R-0	R/WL-0	R/WC-0

LEGEND: R/W = Read/Write; R = Read only; WC = Write in SCI-compatible mode only; WL = Write in LIN mode only; -*n* = value after reset

Table 29-20. SCI Flags Register (SCIFLR) Field Descriptions

Bit	Field	Value	Description
31	BE		Bit error flag. This bit is effective in LIN mode only. This bit is set when a bit error has occurred. This is detected by the internal bit monitor. See Section 29.3.1.8 for more information. The bit error flag is cleared by any of the following: • Setting of the SWnRST bit • Setting of the RESET bit • A system reset • Writing a 1 to this bit • On reception of a new synch break • Reading the corresponding interrupt offset in SCIINTVECT0/1
		0	*Read:* No error has been detected since this bit was last cleared. *Write:* No effect.
		1	*Read:* An error has been detected since this bit was last cleared. *Write:* The bit is cleared to 0.
30	PBE		Physical bus error flag. This bit is effective in LIN mode only. This bit is set when a physical bus error has been detected by the bit monitor in TED. See Section 29.3.1.8 for more information. The physical bus error flag is cleared by the following: • Setting of the SWnRST bit • Setting of the RESET bit • A system reset • Writing a 1 to this bit • On reception of a new synch break • Reading the corresponding interrupt offset in SCIINTVECT0/1 **Note: The PBE will only be flagged, if no Synch Break can be generated (for example, because of a bus shortage to VBAT) or if no Synch Break Delimiter can be generated (for example, because of a bus shortage to GND).**
		0	*Read:* No error has been detected since this bit was last cleared. *Write:* No effect.
		1	*Read:* An error has been detected since this bit was last cleared. *Write:* The bit is cleared to 0.

Table 29-20. SCI Flags Register (SCIFLR) Field Descriptions (continued)

Bit	Field	Value	Description
29	CE		Checksum error flag. This bit is effective in LIN mode only. This bit is set when a checksum error has been detected by a receiving node. This error is detected by the TED logic. See Section 29.3.1.8 for more information. The type of checksum to be used depends on the CTYPE bit in SCIGCR1. The checksum error flag is cleared by the following: • Setting of the SWnRST bit • Setting of the RESET bit • A system reset • Writing a 1 to this bit • Reception of a new synch break • Reading the corresponding interrupt offset in SCIINTVECT0/1
		0	*Read:* No error has been detected since this bit was last cleared. *Write:* No effect.
		1	*Read:* An error has been detected since this bit was last cleared. *Write:* The bit is cleared to 0.
28	ISFE		Inconsistent synch field error flag. This bit is effective in LIN mode only. This bit is set when an inconsistent synch field error has been detected by the synchronizer during header reception. See Section 29.3.1.5.2 for more information. The inconsistent synch field error flag is cleared by the following: • Setting of the SWnRST bit • Setting of the RESET bit • A system reset • Writing a 1 to this bit • Reception of a new synch break • Reading the corresponding interrupt offset in SCIINTVECT0/1
		0	*Read:* No inconsistent synch field error has been detected. *Write:* No effect.
		1	*Read:* An inconsistent synch field error has been detected. *Write:* The bit is cleared to 0.
27	NRE		No-response error flag. This bit is effective in LIN mode only. This bit is set when there is no response to a master's header completed within TFRAME_MAX. This timeout period is applied for message frames of known length (identifiers 0 to 61). This error is detected by the synchronizer. See Section 29.3.1.7 for more information. The no-response error flag is cleared by the following: • Setting of the SWnRST bit • Setting of the RESET bit • A system reset • Writing a 1 to this bit • Reception of a new synch break • Reading the corresponding interrupt offset in SCIINTVECT0/1
		0	*Read:* No no-response error has been detected since the last clear. *Write:* No effect.
		1	*Read:* A no-response error has been detected. *Write:* The bit is cleared to 0.

Table 29-20. SCI Flags Register (SCIFLR) Field Descriptions (continued)

Bit	Field	Value	Description
26	FE		Framing error flag. This bit is effective in LIN or SCI-compatible mode. This bit is set when an expected stop bit is not found. In SCI compatibility mode, only the first stop bit is checked. The missing stop bit indicates that synchronization with the start bit has been lost and that the character is incorrectly framed. Detection of a framing error causes the SCI/LIN to generate an error interrupt if the SET FE INT bit is set in the register SCISETINT. The framing error flag is cleared by the following: • Setting of the SWnRST bit • Setting of the RESET bit • A system reset • Writing a 1 to this bit • Reading the corresponding interrupt offset in SCIINTVECT0/1 • Reception of a new character/frame, depending on whether the module is in SCI compatible or LIN mode In multi-buffer mode the frame is defined in the SCIFORMAT register.
		0	*Read:* No framing error has been detected since the last clear. *Write:* No effect.
		1	*Read:* A framing error has been detected since the last clear. *Write:* The bit is cleared to 0.
25	OE		Overrun error flag. This bit is effective in LIN or SCI-compatible mode. This bit is set when the transfer of data from SCIRXSHF to SCIRD overwrites unread data already in SCIRD or the RDy buffers in LINRD0 and LINRD1. Detection of an overrun error causes the LIN to generate an error interrupt if the SET OE INT bit = 1. The OE flag is reset by the following: • Setting of the SWnRST bit • Setting of the RESET bit • A system reset • Writing a 1 to this bit • Reading the corresponding interrupt offset in SCIINTVECT0/1
		0	*Read:* No overrun error has been detected since the last clear. *Write:* No effect.
		1	*Read:* An overrun error has been detected. *Write:* The bit is cleared to 0.
24	PE		Parity error flag. This bit is effective in LIN or SCI-compatible mode. This bit is set when a parity error is detected in the received data. In SCI address-bit mode, the parity is calculated on the data and address bit fields of the received frame. In idle-line mode, only the data is used to calculate parity. An error is generated when a character is received with a mismatch between the number of 1s and its parity bit. If the parity function is disabled (SCIGCR[2] = 0), the PE flag is disabled and read as 0. Detection of a parity error causes the LIN to generate an error interrupt if the SET PE INT bit = 1. The PE bit is reset by the following: • Setting of the SWnRST bit • Setting of the RESET bit • A system reset • Writing a 1 to this bit • Reception of a new character or frame, depending on whether the module is in SCI compatible or LIN mode, respectively. • Reading the corresponding interrupt offset in SCIINTVECT0/1
		0	*Read:* No parity error has been detected since the last clear. *Write:* No effect.
		1	*Read:* A parity error has been detected. *Write:* The bit is cleared to 0.
23-15	Reserved	0	Reads return 0. Writes have no effect.

Table 29-20. SCI Flags Register (SCIFLR) Field Descriptions (continued)

Bit	Field	Value	Description
14	ID RX		Identifier on receive flag. This bit is effective in LIN mode only. This flag is set once an identifier is received with an receive match and no ID-parity error. See Section 29.3.1.9 for more details. This flag indicates that a new valid identifier has been received on an RX match. This bit is cleared by the following: • Setting of the SWnRST bit • Setting of the RESET bit • A system reset • Writing a 1 to this bit • Reading the LINID register • Reception of a new synch break • Reading the corresponding interrupt offset in SCIINTVECT0/1
		0	*Read:* No valid ID has been received since the last clear. *Write:* No effect.
		1	*Read:* A valid ID RX has been received in LINID[23:16] on an RX match. *Write:* The bit is cleared to 0.
13	ID TX		Identifier on transmit flag. This bit is effective in LIN mode only. This flag is set when an identifier is received with a transmit match and no ID-parity error. See Section 29.3.1.9 for more details. This flag indicates that a new valid identifier has been received on a TX match. This bit is cleared by the following: • Setting the SWnRST bit • Setting of the RESET bit • A system reset • Writing a 1 to this bit • Reading the LINID register • Receiving a new synch break • Reading the corresponding interrupt offset in SCIINTVECT0/1
		0	*Read:* No valid ID has been received since the last clear. *Write:* No effect.
		1	*Read:* A valid ID TX has been received in LINID[23:16] on an TX match. *Write:* The bit is cleared to 0.
12	RX WAKE		Receiver wakeup detect flag. This bit is effective in SCI-compatible mode only. The SCI sets this bit to indicate that the data currently in SCIRD is an address. RX WAKE is cleared by the following: • Setting of the SWnRST bit • Setting of the RESET bit • A system reset • Upon receipt of a data frame.
		0	The data in SCIRD is not an address.
		1	The data in SCIRD is an address.
11	TX EMPTY		Transmitter empty flag. This flag indicates the transmitter's buffer register(s) (SCITD/TDy) and shift register (SCITXSHF) are empty. In multi-buffer mode, this flag indicates the TDx registers and shift register (SCITXSHF) are empty. In non-multi-buffer mode, this flag indicates the LINTD0 byte and the shift register (SCITXSHF) are empty. **Note: The RESET bit, an active SWnRST (SCIGCR1[7]) or a system reset sets this bit. This bit does not cause an interrupt request.** *SCI mode or LIN non-multi-buffer mode:*
		0	Transmitter buffer or shift register (or both) are loaded with data.
		1	Transmitter buffer and shift registers are both empty.
			In LIN mode using multi-buffer mode:
		0	Multi-buffer or shift register (or all) are loaded with data
		1	Multi-buffer and shift registers are all empty.

Table 29-20. SCI Flags Register (SCIFLR) Field Descriptions (continued)

Bit	Field	Value	Description
10	TX WAKE		Transmitter wakeup method select. This bit is effective in SCI mode only. The TX WAKE bit controls whether the data in SCITD should be sent as an address or data frame using multiprocessor communication format. This bit is set to 1 or 0 by software before a byte is written to SCITD and is cleared by the SCI when data is transferred from SCITD to SCITXSHF or by a system reset. **Note: TX WAKE is not cleared by the SWnRST bit.** *Address-bit mode*:
		0	Frame to be transmitted will be data (address bit = 0).
		1	Frame to be transmitted will be an address (address bit = 1).
			Idle-line mode:
		0	The frame to be transmitted will be data.
		1	The following frame to be transmitted will be an address (writing a 1 to this bit followed by writing dummy data to the SCITD will result in a idle period of 11 bit periods before the next frame is transmitted).
9	RX RDY		Receiver ready flag. In SCI-compatible mode, the receiver sets this bit to indicate that the SCIRD contains new data and is ready to be read by the CPU. In *LIN mode,* RX RDY is set once a valid frame is received in multi-buffer mode, a valid frame being a message frame received with no errors. In *non-multi-buffer mode,* RX RDY is set for each received byte and will be set for the last byte of the frame if there are no errors. The SCI/LIN generates a receive interrupt when RX RDY flag bit is set if the SET RX INT bit is set (SCISETINT[9]); RX RDY is cleared by the following: • Setting of the SWnRST bit • Setting of the RESET bit • A system reset • Writing a 1 to this bit • Reading the SCIRD register in compatibility mode • Reading the last data byte RDy of the response in LIN mode **Note: The RXRDY flag cannot be cleared by reading the corresponding interrupt offset in the SCIINTVECT0/1 register.**
		0	*Read:* No new data is in SCIRD. *Write:* No effect.
		1	*Read:* New data is ready to be read from SCIRD. *Write:* The bit is cleared to 0.
8	TX RDY		Transmitter buffer register ready flag. When set, this bit indicates that the transmit buffer(s) register(s) (SCITD in compatibility mode and LINTD0/LINTD1 in multi-buffer mode) are ready to get another character from a CPU write. *In SCI,* writing data to SCITD automatically clears this bit. In *LIN mode,* this bit is cleared once byte 0 (TD0) is written to LINTD0. This bit is set after the data of the TX buffer is shifted into the SCITXSHF register. This event can trigger a transmit interrupt after data is copied to the TX shift register SCITXSHF, if the SET TX INT is set. **Note:** **1) TXRDY is also set to 1 either by setting of the RESET bit, enabling SWnRST or by a system reset.** **2) The TXRDY flag cannot be cleared by reading the corresponding interrupt offset in the SCIINTVECT0/1 register.** **3) The transmit interrupt request can be eliminated until the next series of data written into the transmit buffers LINTD0 and LINTD1, by disabling the corresponding interrupt via the SCICLEARINT register or by disabling the transmitter via the TXENA bit.** *SCI mode:*
		0	SCITD is full.
		1	SCITD is ready to receive the next character.
			LIN mode:
		0	Multi-buffers are full.
		1	Multi-buffers are ready to receive the next character. For more information on transmit interrupt handling, see the SCI document for compatibility mode and Section 29.3.1.9 for LIN mode.

Table 29-20. SCI Flags Register (SCIFLR) Field Descriptions (continued)

Bit	Field	Value	Description
7	TOA3WUS		Timeout after three wakeup signals flag. This bit is effective in LIN mode only. This flag is set if there is no synch break received after three wakeup signals and a period of 1.5 seconds has passed. Such expiration time is used before issuing another round of wakeup signals. This bit is cleared by the following: • Setting of the SWnRST bit • Setting of the RESET bit • A system reset • Writing a 1 to this bit • Reading the corresponding interrupt offset in SCIINTVECT0/1 See Section 29.4.3 for more information.
		0	*Read:* No timeout occurred after three wakeup signals. *Write:* No effect.
		1	*Read:* Timeout occurred after three wakeup signals and 1.5 seconds time. *Write:* The bit is cleared to 0.
6	TOAWUS		Timeout after wakeup signal flag. This bit is effective in LIN mode only. This bit is set if there is no synch break received after a wakeup signal has been sent. A minimum of 150 ms expiration time is used before issuing another wakeup signal. This bit is cleared by the following: • Setting the SWnRST bit • Setting of the RESET bit • A system reset occurring • Writing a 1 to this bit • Reading the corresponding interrupt offset in SCIINTVECT0/1 See Section 29.4.3 for more information.
		0	*Read:* No timeout occurred after one wakeup signal (150 ms). *Write:* No effect.
		1	*Read:* Timeout occurred after one wakeup signal. *Write:* The bit is cleared to 0.
5	Reserved	0	Reads return 0. Writes have no effect.
4	TIMEOUT		LIN bus idle timeout flag. This bit is effective in LIN mode only. This bit is set earliest after at least four seconds of bus inactivity. Bus inactivity is defined as no transactions between recessive and dominant (and vice versa). This bit is cleared by the following: • Setting the SWnRST bit • Setting of the RESET bit • A system reset occurring • Writing a 1 to this bit • Reading the corresponding interrupt offset in SCIINTVECT0/1 See Section 29.3.1.7 for more information.
		0	*Read:* No bus idle has been detected since this bit was last cleared. *Write:* No effect.
		1	*Read:* A LIN bus idle has been detected. *Write:* The bit is cleared to 0.
3	BUSY		Bus busy flag. This bit is effective in LIN mode and SCI-compatible mode. This bit indicates whether the receiver is in the process of receiving a frame. As soon as the receiver detects the beginning of a start bit, the BUSY bit is set to 1. When the reception of a frame is complete, the SCI/LIN clears the BUSY bit. If SET WAKEUP INT is set and power down is requested while this bit is set, the SCI/LIN automatically prevents low-power mode from being entered and generates wakeup interrupt. The BUSY bit is controlled directly by the SCI/LIN receiver, but this bit can also be cleared by the following: • Setting the SWnRST bit • Setting of the RESET bit • A system reset occurring
		0	The receiver is not currently receiving a frame.
		1	The receiver is currently receiving a frame.

Table 29-20. SCI Flags Register (SCIFLR) Field Descriptions (continued)

Bit	Field	Value	Description
2	IDLE		SCI receiver in idle state. This bit is effective in SCI-compatible mode only. While this bit is set, the SCI looks for an idle period to resynchronize itself with the bit stream. The receiver does not receive any data while the bit is set. The bus must be idle for 11 bit periods to clear this bit. The SCI enters the idle state if one of the following events occurs: • A system reset • An SCI software reset • A power down • The RX pin is configured as a general I/O pin
		0	The idle period has been detected; the SCI is ready to receive.
		1	The idle period has not been detected; the SCI will not receive any data.
1	WAKEUP		Wakeup flag. This bit is effective in LIN mode only. This bit is set by the SCI/LIN when receiver or transmitter activity has taken the module out of power-down mode. An interrupt is generated if the SET WAKEUP INT bit (SCISETINT[2]) is set. It is cleared by the following: • Setting of the SWnRST bit • Setting of the RESET bit • A system reset • Writing a 1 to this bit • Reading the corresponding interrupt offset in SCIINTVECT0/1 For compatibility mode, see the SCI document for more information on low-power mode.
		0	*Read:* The module will not wake up from power-down mode. *Write:* No effect.
		1	*Read:* Wake up from power-down mode. *Write:* The bit is cleared to 0.
0	BRKDT		SCI break-detect flag. This bit is effective in SCI-compatible mode only. This bit is set when the SCI detects a break condition on the LINRX pin. A break condition occurs when the LINRX pin remains continuously low for at least 10 bits after a missing first stop bit, that is, after a framing error. Detection of a break condition causes the SCI to generate an error interrupt if the SET BRKDT INT bit is set. The BRKDT bit is reset by the following: • Setting of the SWnRST bit • Setting of the RESET bit • A system reset • Writing a 1 to this bit • Reading the corresponding interrupt offset in SCIINTVECT0/1
		0	*Read:* No break condition has been detected since the last clear. *Write:* No effect.
		1	*Read:* A break condition has been detected. *Write:* The bit is cleared to 0.

29.7.9 SCI Interrupt Vector Offset 0 (SCIINTVECT0)

Figure 29-36 and Table 29-21 illustrate this register.

Figure 29-36. SCI Interrupt Vector Offset 0 (SCIINTVECT0) (offset = 20h)

31	16
Reserved	
R-0	

15	5	4	0
Reserved		INTVECT0	
R-0		R-0	

LEGEND: R = Read only; -*n* = value after reset

Table 29-21. SCI Interrupt Vector Offset 0 (SCIINTVECT0) Field Descriptions

Bit	Field	Value	Description
31-5	Reserved	0	Reads return 0. Writes have no effect.
4-0	INVECT0	0-1Fh	Interrupt vector offset for INT0. This register indicates the offset for interrupt line INT0. A read to this register updates its value to the next highest priority pending interrupt in SCIFLR and clears the flag in SCIFLR corresponding to the offset that was read. See Table 29-4 for a list of the interrupts. **Note: The flags for the receive (SCIFLR[9]) and the transmit (SCIFLR[8]) interrupt cannot be cleared by reading the corresponding offset vector in this register (see detailed description in SCIFLR register).**

29.7.10 SCI Interrupt Vector Offset 1 (SCIINTVECT1)

Figure 29-37 and Table 29-22 illustrate this register.

Figure 29-37. SCI Interrupt Vector Offset 1 (SCIINTVECT1) (offset = 24h)

31	16
Reserved	
R-0	

15	5	4	0
Reserved		INTVECT1	
R-0		R-0	

LEGEND: R = Read only; -*n* = value after reset

Table 29-22. SCI Interrupt Vector Offset 1 (SCIINTVECT1) Field Descriptions

Bit	Field	Value	Description
31-5	Reserved	0	Reads return 0. Writes have no effect.
4-0	INVECT1	0-1Fh	Interrupt vector offset for INT1. This register indicates the offset for interrupt line INT1. A read to this register updates its value to the next highest priority pending interrupt in SCIFLR and clears the flag in SCIFLR corresponding to the offset that was read. See Table 29-4 for list of interrupts. **Note: The flags for the receive (SCIFLR[9]) and the transmit (SCIFLR[8]) interrupt cannot be cleared by reading the corresponding offset vector in this register (see detailed description in SCIFLR register).**

29.7.11 SCI Format Control Register (SCIFORMAT)

Figure 29-38 and Table 29-23 illustrate this register.

Figure 29-38. SCI Format Control Register (SCIFORMAT) (offset = 28h)

31 19	18 16
Reserved	LENGTH
R-0	R/W-0

15 3	2 0
Reserved	CHAR
R-0	R/WC-0

LEGEND: R/W = Read/Write; R = Read only; WC = Write in SCI-compatible mode only; -*n* = value after reset

Table 29-23. SCI Format Control Register (SCIFORMAT) Field Descriptions

Bit	Field	Value	Description
31-19	Reserved	0	Reads return 0. Writes have no effect.
18-16	LENGTH	0-3h	Frame length control bits. In *LIN mode*, these bits indicate the number of bytes in the response field from 1 to 8 bytes. In *buffered SCI mode*, these bits indicate the number of characters, with the number of bits per character specified in CHAR (SCIFORMAT[2:0]). When these bits are used to indicate LIN response length (SCIGCR1[0] = 1), then when there is an ID RX match, this value should be updated with the expected length of the response. In buffered SCI mode, these bits indicate the transmitter/receiver format for the number of characters: 1 to 8. There can be up to eight characters with eight bits each.
		0	The response field has 1 byte/character.
		1h	The response field has 2 bytes/characters.
		2h	The response field has 3 bytes/characters.
		3h	The response field has 4 bytes/characters.
		4h	The response field has 5 bytes/characters.
		5h	The response field has 6 bytes/characters.
		6h	The response field has 7 bytes/characters.
		7h	The response field has 8 bytes/characters.
15-3	Reserved	0	Reads return 0. Writes have no effect.
2-0	CHAR	0-7h	Character length control bits. These bits are effective in non-buffered SCI and buffered SCI modes only. These bits set the SCI character length from 1 to 8 bits. **In non-buffered SCI and buffered SCI mode, when data of fewer than eight bits in length is received, it is left justified in SCIRD/RDy and padded with trailing zeros. Data read from the SCIRD should be shifted by software to make the received data right justified.** **Data written to the SCITD should be right justified but does not need to be padded with leading zeros.**
		0	The character is 1 bit long.
		1h	The character is 2 bits long.
		2h	The character is 3 bits long.
		3h	The character is 4 bits long.
		4h	The character is 5 bits long.
		5h	The character is 6 bits long.
		6h	The character is 7 bits long.
		7h	The character is 8 bits long.

29.7.12 Baud Rate Selection Register (BRS)

This section describes the baud rate selection register. Figure 29-39 and Table 29-24 illustrate this register.

Figure 29-39. Baud Rate Selection Register (BRS) (offset = 2Ch)

31	30 - 28	27 - 24	23 - 16
Rsvd	U	M	PRESCALER P
R-0	R/W-0	R/W-0	R/W-0

15 - 0
PRESCALER P
R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 29-24. Baud Rate Selection Register (BRS) Field Descriptions

Bit	Field	Value	Description
31	Reserved	0	Reads return 0. Writes have no effect.
30-28	U	0-2h	SCI/LIN super fractional divider selection. These bits are effective in LIN or SCI asynchronous mode. These bits are an additional fractional part for the baud rate specification. These bits allow a super-fine tuning of the fractional baud rate with seven more intermediate values for each of the M fractional divider values. See Section 29.3.1.4.1 for more details.
27-24	M	0-3h	SCI/LIN 4-bit fractional divider selection. These bits are effective in LIN or SCI asynchronous mode. These bits are used to select a baud rate for the SCI/LIN module, and they are a fractional part for the baud rate specification. The M divider allows fine-tuning of the baud rate over the P prescaler with 15 additional intermediate values for each of the P integer values. See Section 29.3.1.4.1 for more details.
23-0	PRESCALER P	0-FF FFFFh	These bits are used to select a baud rate for the SCI/LIN module. These bits are effective in LIN mode and SCI compatibility. The SCI/LIN has an internally generated serial clock determined by the VCLK and the prescalers P and M in this register. The LIN uses the 24-bit integer prescaler P value of this register to select one of over 16,700,000. The additional 4-bit fractional divider M refines the baud rate selection PRESCALER[27:24]. **NOTE: In LIN mode, ONLY the asynchronous mode and baud rate values are used.** The baud rate can be calculated using the following formulas: $\textit{Asynchronous baud value} = \left(\frac{\text{VCLK Frequency}}{16\left(P + 1 + \frac{M}{16}\right)}\right)$ (52) $\textit{Isosynchronous baud value} = \left(\frac{\text{VCLK Frequency}}{P + 1}\right)$ (53) For P = 0, $\textit{Asynchronous baud value} = \left(\frac{\text{VCLK Frequency}}{32}\right)$ (54) $\textit{Isosynchronous baud value} = \left(\frac{\text{VCLK Frequency}}{2}\right)$ (55) Table 29-25 contains comparative baud values for different P values, with VCLK = 50 MHz, for asynchronous mode.

Table 29-25. Comparative Baud Values for Different P Values, Asynchronous Mode[1][2]

24-Bit Register Value		Baud Selected		Percent Error
Decimal	Hex	Ideal	Actual	
26	00001A	115200	115740	0.47
53	000035	57600	57870	0.47
80	000050	38400	38580	0.47
162	0000A2	19200	19172	-0.15
299	00012B	10400	10417	0.16
325	000145	9600	9586	-0.15
399	00018F	7812.5	7812.5	0.00
650	00028A	4800	4800	0.00
15624	003BA0	200	200	0.00
624999	098967	5	5	0.00

[1] VCLK = 50 MHz

[2] Values are in decimal except for column 2.

29.7.13 SCI Data Buffers (SCIED, SCIRD, SCITD)

The SCI has three addressable registers in which transmit and receive data is stored. These three registers are available in SCI mode only.

29.7.13.1 Receiver Emulation Data Buffer (SCIED)

The SCIED register is addressed at a location different from SCIRD, but is physically the same register. Figure 29-40 and Table 29-26 illustrate this register.

Figure 29-40. Receiver Emulation Data Buffer (SCIED) (offset = 30h)

31 16
Reserved
R-0

15 8	7 0
Reserved	ED
R-0	R-0

LEGEND: R = Read only; -*n* = value after reset

Table 29-26. Receiver Emulation Data Buffer (SCIED) Field Descriptions

Bit	Field	Value	Description
31-8	Reserved	0	Reads return 0. Writes have no effect.
7-0	ED	0-FFh	Emulator data. This bit is effective in SCI-compatible mode only. Reading SCIED[7:0] does not clear the RXRDY flag, unlike reading SCIRD. This register should be used only by an emulator that must continually read the data buffer without affecting the RXRDY flag.

29.7.13.2 Receiver Data Buffer (SCIRD)

This register provides a location for the receiver data. Figure 29-41 and Table 29-27 illustrate this register.

Figure 29-41. Receiver Data Buffer (SCIRD) (offset = 34h)

31 16
Reserved
R-0

15 8	7 0
Reserved	RD
R-0	R-0

LEGEND: R = Read only; -*n* = value after reset

Table 29-27. Receiver Data Buffer (SCIRD) Field Descriptions

Bit	Field	Value	Description
31-8	Reserved	0	Reads return 0. Writes have no effect.
7-0	RD	0-FFh	Receiver data. This bit is effective in SCI-compatible mode only. When a frame has been completely received, the data in the frame is transferred from the receiver shift register SCIRXSHF to this register. As this transfer occurs, the RXRDY flag is set and a receive interrupt is generated if SET RX INT is set. **Note: When the data is read from SCIRD, the RXRDY flag is automatically cleared.**

NOTE: When the SCI receives data that is fewer than eight bits in length, it loads the data into this register in a left justified format padded with trailing zeros. Therefore, your software should perform a logical shift on the data by the correct number of positions to make it right justified.

29.7.13.3 Transmit Data Buffer Register (SCITD)

Data to be transmitted is written to the SCITD register. Figure 29-42 and Table 29-28 illustrate this register.

Figure 29-42. Transmit Data Buffer Register (SCITD) (offset = 38h)

31–16
Reserved
R-0

15–8	7–0
Reserved	TD
R-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 29-28. Transmit Data Buffer Register (SCITD) Field Descriptions

Bit	Field	Value	Description
31-8	Reserved	0	Reads return 0. Writes have no effect.
7-0	TD	0-FFh	Transmit data. This bit is effective in SCI-compatible mode only. Data to be transmitted is written to the SCITD register. The transfer of data from this register to the transmit shift register SCITXSHF sets the TXRDY flag, which indicates that SCITD is ready to be loaded with another byte of data. **Note: If SET TX INT is set, this data transfer also causes an interrupt.**

NOTE: Data written to the SCITD register that is fewer than eight bits long must be right justified, but it does not need to be padded with leading zeros.

29.7.14 SCI Pin I/O Control Register 0 (SCIPIO0)

Figure 29-43 and Table 29-29 illustrate this register.

Figure 29-43. SCI Pin I/O Control Register 0 (SCIPIO0) (offset = 3Ch)

31–8
Reserved
R-0

7–3	2	1	0
Reserved	TX FUNC	RX FUNC	Reserved
R-0	R/W-0	R/W-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 29-29. SCI Pin I/O Control Register 0 (SCIPIO0) Field Descriptions

Bit	Field	Value	Description
31-3	Reserved	0	Reads return 0. Writes have no effect.
2	TX FUNC		Transfer function. This bit is effective in LIN or SCI mode. This bit defines the function of pin LINTX.
		0	LINTX is a general-purpose digital I/O pin.
		1	LINTX is the SCI/LIN transmit pin.
1	RX FUNC		Receive function. This bit is effective in LIN or SCI mode. This bit defines the function of pin LINRX.
		0	LINRX is a general-purpose digital I/O pin.
		1	LINRX is the SCI/LIN receive pin.
0	Reserved	0	Writes have no effect.

29.7.15 SCI Pin I/O Control Register 1 (SCIPIO1)

Figure 29-44 and Table 29-30 illustrate this register.

Figure 29-44. SCI Pin I/O Control Register 1 (SCIPIO1) (offset = 40h)

31	8
Reserved	
R-0	

7	3	2	1	0
Reserved		TX DIR	RX DIR	Reserved
R-0		R/W-0	R/W-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 29-30. SCI Pin I/O Control Register 1 (SCIPIO1) Field Descriptions

Bit	Field	Value	Description
31-3	Reserved	0	Reads return 0. Writes have no effect.
2	TX DIR		Transmit pin direction. This bit is effective in LIN or SCI mode. This bit determines the data direction on the LINTX pin if it is configured with general-purpose I/O functionality (TX FUNC = 0). See Table 29-31 for the LINTX pin control with this bit and others.
		0	LINTX is a general-purpose input pin.
		1	LINTX is a general-purpose output pin.
1	RX DIR		Receive pin direction. This bit is effective in LIN or SCI mode. This bit determines the data direction on the LINRX pin if it is configured with general-purpose I/O functionality (RX FUNC = 0). See Table 29-32 for the LINRX pin control with this bit and others.
		0	LINRX is a general-purpose input pin.
		1	LINRX is a general-purpose output pin.
0	Reserved	0	Writes have no effect.

Table 29-31. LINTX Pin Control

Function	TX IN [1]	TX OUT	TX FUNC	TX DIR
LINTX	X	X	1	X
General purpose input	X	X	0	0
General purpose output, high	X	1	0	1
General purpose output, low	X	0	0	1

[1] TX IN is a read-only bit. Its value always reflects the level of the SCITX pin.

Table 29-32. LINRX Pin Control

Function	RX IN [1]	RX OUT	RX FUNC	RX DIR
LINRX	X	X	1	X
General purpose input	X	X	0	0
General purpose output, high	X	1	0	1
General purpose output, low	X	0	0	1

[1] RX IN is a read-only bit. Its value always reflects the level of the SCIRX pin.

29.7.16 SCI Pin I/O Control Register 2 (SCIPIO2)

Figure 29-45 and Table 29-33 illustrate this register.

Figure 29-45. SCI Pin I/O Control Register 2 (SCIPIO2) (offset = 44h)

31 – 8
Reserved
R-0

7 – 3	2	1	0
Reserved	TX IN	RX IN	Reserved
R-0	R-X	R-X	R-X

LEGEND: R = Read only; X= value is indeterminate; -*n* = value after reset

Table 29-33. SCI Pin I/O Control Register 2 (SCIPIO2) Field Descriptions

Bit	Field	Value	Description
31-3	Reserved	0	Reads return 0. Writes have no effect.
2	TX IN		Transmit pin in. This bit is effective in LIN or SCI mode. This bit contains the current value on the LINTX pin.
		0	The LINTX pin is at logic low (0).
		1	The LINTX pin is at logic high (1).
1	RX IN		Receive pin in. This bit is effective in LIN or SCI mode. This bit contains the current value on the LINRX pin.
		0	The LINRX pin is at logic low (0).
		1	The LINRX pin is at logic high (1).
0	Reserved	0	Writes have no effect.

29.7.17 SCI Pin I/O Control Register 3 (SCIPIO3)

Figure 29-46 and Table 29-34 illustrate this register.

Figure 29-46. SCI Pin I/O Control Register 3 (SCIPIO3) (offset = 48h)

31 – 8
Reserved
R-0

7 – 3	2	1	0
Reserved	TX OUT	RX OUT	Reserved
R-0	R/W-0	R/W-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 29-34. SCI Pin I/O Control Register 3 (SCIPIO3) Field Descriptions

Bit	Field	Value	Description
31-3	Reserved	0	Reads return 0. Writes have no effect.
2	TX OUT		Transmit pin out. This bit is effective in LIN or SCI mode. This pin specifies the logic to be output on pin LINTX if the following conditions are met: • TX FUNC = 0 (LINTX pin is a general-purpose I/O.) • TX DIR = 1 (LINTX pin is a general-purpose output.) See Table 29-31 for an explanation of this bit's effect in combination with other bits.
		0	The output on the LINTX is at logic low (0).
		1	The output on the LINTX pin is at logic high (1). (Output voltage is V_{OH} or higher if TXPDR = 0 and output is in high impedance state if TXPDR = 1.)
1	RX OUT		Receive pin out. This bit is effective in LIN or SCI mode. This bit specifies the logic to be output on pin LINRX if the following conditions are met: • RX FUNC = 0 (LINRX pin is a general-purpose I/O.) • RX DIR = 1 (LINRX pin is a general-purpose output.) See Table 29-32 for an explanation of this bit's effect in combination with the other bits.
		0	The output on the LINRX pin is at logic low (0).
		1	The output on the LINRX pin is at logic high (1). (Output voltage is V_{OH} or higher if RXPDR = 0, and output is in high impedance state if RXPDR = 1.)
0	Reserved	0	Writes have no effect.

29.7.18 SCI Pin I/O Control Register 4 (SCIPIO4)

Figure 29-47 and Table 29-35 illustrate this register.

Figure 29-47. SCI Pin I/O Control Register 4 (SCIPIO4) (offset = 4Ch)

31-8
Reserved
R-0

7-3	2	1	0
Reserved	TX SET	RX SET	Reserved
R-0	R/W-0	R/W-0	R-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 29-35. SCI Pin I/O Control Register 4 (SCIPIO4) Field Descriptions

Bit	Field	Value	Description
31-3	Reserved	0	Reads return 0. Writes have no effect.
2	TX SET		Transmit pin set. This bit is effective in LIN or SCI mode. This bit sets the logic to be output on pin LINTX if the following conditions are met: • TX FUNC = 0 (LINTX pin is a general-purpose I/O.) • TX DIR = 1 (LINTX pin is a general-purpose output.) See Table 29-31 for an explanation of this bit's effect in combination with other bits.
		0	*Read:* The output on LINTX is at logic low (0). *Write:* No effect.
		1	*Read and write:* The output on LINTX is at logic high (1).
1	RX SET		Receive pin set. This bit is effective in LIN or SCI mode. This bit sets the data to be output on pin LINRX if the following conditions are met: • RX FUNC = 0 (LINRX pin is a general-purpose I/O.) • RX DIR = 1 (LINRX pin is a general-purpose output.) See Table 29-32 for an explanation of this bit's effect in combination with the other bits.
		0	*Read:* The output on LINRX is at logic low (0). *Write:* No effect.
		1	*Read and write:* The output on LINRX is at logic high (1).
0	Reserved	0	Reads return 0. Writes have no effect.

29.7.19 SCI Pin I/O Control Register 5 (SCIPIO5)

Figure 29-48 and Table 29-36 illustrate this register.

Figure 29-48. SCI Pin I/O Control Register 5 (SCIPIO5) (offset = 50h)

31 … 8
Reserved
R-0

7 … 3	2	1	0
Reserved	TX CLR	RX CLR	Reserved
R-0	R/W-0	R/W-0	R-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 29-36. SCI Pin I/O Control Register 5 (SCIPIO5) Field Descriptions

Bit	Field	Value	Description
31-3	Reserved	0	Reads return 0. Writes have no effect.
2	TX CLR		Transmit pin clear. This bit is effective in LIN or SCI mode. This bit clears the logic to be output on pin LINTX if the following conditions are met: • TX FUNC = 0 (LINTX pin is a general-purpose I/O.) • TX DIR = 1 (LINTX pin is a general-purpose output.)
		0	*Read:* The output on LINTX is at logic low (0). *Write:* No effect.
		1	*Read:* The output on LINTX is at logic high (1). *Write:* The output on LINTX is at logic low (0).
1	RX CLR		Receive pin clear. This bit is effective in LIN or SCI mode. This bit clears the logic to be output on pin LINRX if the following conditions are met: • RX FUNC = 0 (LINRX pin is a general-purpose I/O.) • RX DIR = 1 (LINRX pin is a general-purpose output.)
		0	*Read:* The output on LINRX is at logic low (0). *Write:* No effect.
		1	*Read:* The output on LINRX is at logic high (1). *Write:* The output on LINRX is at logic low (0).
0	Reserved	0	Reads return 0. Writes have no effect.

29.7.20 SCI Pin I/O Control Register 6 (SCIPIO6)

Figure 29-49 and Table 29-37 illustrate this register.

Figure 29-49. SCI Pin I/O Control Register 6 (SCIPIO6) (offset = 54h)

31	8
Reserved	
R-0	

3	2	1	0
Reserved	TX PDR	RX PDR	Reserved
R-0	R/W-0	R/W-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 29-37. SCI Pin I/O Control Register 6 (SCIPIO6) Field Descriptions

Bit	Field	Value	Description
31-3	Reserved	0	Reads return 0. Writes have no effect.
2	TX PDR		Transmit pin open drain enable. This bit is effective in LIN or SCI mode. This bit enables open-drain capability in the output pin LINTX, if the following conditions are met: • TX FUNC = 0 (LINTX pin is a general-purpose I/O.) • TX DIR = 1 (LINTX pin is a general-purpose output.)
		0	Open-drain functionality is disabled; the output voltage is V_{OL} or lower if TXOUT = 0, and is V_{OH} or higher if TXOUT = 1.
		1	Open-drain functionality is enabled; the output voltage is V_{OL} or lower if TXOUT = 0, and is high-impedance if TXOUT = 1.
1	RX PDR		Receive pin open drain enable. This bit is effective in LIN or SCI mode. This bit enables open-drain capability in the output pin LINRX, if the following conditions are met: • RX FUNC = 0 (LINRX pin is a general-purpose I/O.) • RX DIR = 1 (LINRX pin is a general-purpose output.)
		0	Open-drain functionality is disabled; the output voltage is V_{OL} or lower if RXOUT = 0, and is V_{OH} or higher if RXOUT = 1.
		1	Open-drain functionality is enabled; the output voltage is V_{OL} or lower if RXOUT = 0, and is high-impedance if RXOUT = 1.
0	Reserved	0	Writes have no effect.

29.7.21 SCI Pin I/O Control Register 7 (SCIPIO7)

Figure 29-50 and Table 29-38 illustrate this register.

Figure 29-50. SCI Pin I/O Control Register 7 (SCIPIO7) (offset = 58h)

31 - 8
Reserved
R-0

7 - 3	2	1	0
Reserved	TX PD	RX PD	Reserved
R-0	R/W-n	R/W-n	R/W-n

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset, Refer to the Terminal Functions in the device datasheet for default pin settings.

Table 29-38. SCI Pin I/O Control Register 7 (SCIPIO7) Field Descriptions

Bit	Field	Value	Description
31-3	Reserved	0	Reads return 0. Writes have no effect.
2	TX PD		Transmit pin pull control disable. This bit is effective in LIN or SCI mode. This bit disables pull control capability on the input pin LINTX.
		0	The pull control on the LINTX pin is enabled.
		1	The pull control on the LINTX pin is disabled.
1	RX PD		Receive pin pull control disable. This bit is effective in LIN or SCI mode. This bit disables pull control capability on the input pin LINRX.
		0	Pull control on the LINRX pin is enabled.
		1	Pull control on the LINRX pin is disabled.
0	Reserved	0	Writes have no effect.

29.7.22 SCI Pin I/O Control Register 8 (SCIPIO8)

Figure 29-51 and Table 29-39 illustrate this register.

Figure 29-51. SCI Pin I/O Control Register 8 (SCIPIO8) (offset = 5Ch)

<table>
<tr><td>31</td><td>8</td></tr>
<tr><td colspan="2">Reserved</td></tr>
<tr><td colspan="2">R-0</td></tr>
</table>

<table>
<tr><td>7</td><td>3</td><td>2</td><td>1</td><td>0</td></tr>
<tr><td colspan="2">Reserved</td><td>TX PSL</td><td>RX PSL</td><td>Reserved</td></tr>
<tr><td colspan="2">R-0</td><td>R/W-n</td><td>R/W-n</td><td>R/W-n</td></tr>
</table>

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset, Refer to the Terminal Functions in the device datasheet for default pin settings.

Table 29-39. SCI Pin I/O Control Register 8 (SCIPIO8) Field Descriptions

Bit	Field	Value	Description
31-3	Reserved	0	Reads return 0. Writes have no effect.
2	TX PSL		TX pin pull select. This bit is effective in LIN or SCI mode. This bit selects pull type in the input pin LINTX.
		0	The LINTX pin is a pull down.
		1	The LINTX pin is a pull up.
1	RX PSL		RX pin pull select. This bit is effective in LIN or SCI mode. This bit selects pull type in the input pin LINRX.
		0	The LINRX pin is a pull down.
		1	The LINRX pin is a pull up.
0	Reserved	0	Writes have no effect.

29.7.23 LIN Compare Register (LINCOMPARE)

Figure 29-52 and Table 29-40 illustrate this register.

Figure 29-52. LIN Compare Register (LINCOMPARE) (offset = 60h)

31	16
Reserved	
R-0	

15 10	9 8	7 3	2 0
Reserved	SDEL	Reserved	SBREAK
R-0	R/WP-0	R-0	R/WP-0

LEGEND: R/W = Read/Write; R = Read only; R/WP = Read/Write in privileged mode only; -*n* = value after reset

Table 29-40. LIN Compare Register (LINCOMPARE) Field Descriptions

Bit	Field	Value	Description
31-19	Reserved	0	Reads return 0. Writes have no effect.
18-16	SDEL		2-bit synch delimiter compare. These bits are effective in LIN mode only. These bits are used to configure the number of T_{bit} for the synch delimiter in the synch field. The default value is 0.
			The formula to program the value (in T_{bits}) for the synchronization delimiter is:
			T_{SDEL} = (SDEL + 1) T_{bit}
		0	The synch delimiter has 1 T_{bit}.
		1h	The synch delimiter has 2 T_{bit}.
		2h	The synch delimiter has 3 T_{bit}.
		3h	The synch delimiter has 4 T_{bit}.
15-3	Reserved	0	Reads return 0. Writes have no effect.
2-0	SBREAK		Synch break extend. These bits are effective in LIN mode only. These bits are used to configure the number of T_{bit} for the synch break to extend the minimum 13 T_{bit} break field to a maximum of 20 T_{bit} long.
			Note: The default value is 0, which adds nothing to the automatically generated SYNCH BREAK.
			The formula to program the value (in T $_{bits}$) for the SYNCH BREAK is:
			T_{SYNBRK} = 13T_{bit} + (SBREAK × T_{bit})
		0	The synch break has no additional T_{bit}.
		1h	The synch break has 1 additional T_{bit}.
		2h	The synch break has 2 additional T_{bit}.
		3h	The synch break has 3 additional T_{bit}.
		4h	The synch break has 4 additional T_{bit}.
		5h	The synch break has 5 additional T_{bit}.
		6h	The synch break has 6 additional T_{bit}.
		7h	The synch break has 7 additional T_{bit}.

29.7.24 LIN Receive Buffer 0 Register (LINRD0)

Figure 29-53 and Table 29-41 illustrate this register.

Figure 29-53. LIN Receive Buffer 0 Register (LINRD0) (offset = 64h)

31 — 24	23 — 16
RD0	RD1
R-0	R-0

15 — 8	7 — 0
RD2	RD3
R-0	R-0

LEGEND: R = Read only; -*n* = value after reset

Table 29-41. LIN Receive Buffer 0 Register (LINRD0) Field Descriptions

Bit	Field	Value	Description
31-24	RD0	0-FFh	Receive buffer 0. Byte 0 of the response data byte. Each response data-byte that is received in the SCIRXSHFT register is transferred to the corresponding RDy bit field according to the number of bytes received. A read of this byte clears the RXDY byte. **Note: RD<x-1> is equivalent to data byte <x> of the LIN frame.**
23-16	RD1	0-FFh	Receive buffer 1. Byte 1 of the response data byte. Each response data-byte that is received in the SCIRXSHFT register is transferred to the corresponding RDy register according to the number of bytes received.
15-8	RD2	0-FFh	Receive buffer 2. Byte 2 of the response data byte. Each response data-byte that is received in the SCIRXSHFT register is transferred to the corresponding RDy register according to the number of bytes received.
7-0	RD3	0-FFh	Receive buffer 3. Byte 3 of the response data byte. Each response data-byte that is received in the SCIRXSHFT register is transferred to the corresponding RDy register according to the number of bytes received.

29.7.25 LIN Receive Buffer 1 Register (LINRD1)

Figure 29-54 and Table 29-42 illustrate this register.

Figure 29-54. LIN Receive Buffer 1 Register (RD1) (offset = 68h)

31 24	23 16
RD4	RD5
R-0	R-0

15 8	7 0
RD6	RD7
R-0	R-0

LEGEND: R = Read only; -*n* = value after reset

Table 29-42. LIN Receive Buffer 1 Register (RD1) Field Descriptions

Bit	Field	Value	Description
31-24	RD4	0-FFh	Receive buffer 4. Byte 4 of the response data byte. Each response data-byte that is received in the SCIRXSHFT register is transferred to the corresponding register according to the number of bytes received. **Note: RD<x-1> is equivalent to data byte <x> of the LIN frame.**
23-16	RD5	0-FFh	Receive buffer 5. Byte 5 of the response data byte. Each response data-byte that is received in the SCIRXSHFT register is transferred to the corresponding register according to the number of bytes received.
15-8	RD6	0-FFh	Receive buffer 6. Byte 6 of the response data byte. Each response data-byte that is received in the SCIRXSHFT register is transferred to the corresponding register according to the number of bytes received.
7-0	RD7	0-FFh	Receive buffer 7. Byte 7 of the response data byte. Each response data-byte that is received in the SCIRXSHFT register is transferred to the corresponding register according to the number of bytes received.

29.7.26 LIN Mask Register (LINMASK)

Figure 29-55 and Table 29-43 illustrate this register.

Figure 29-55. LIN Mask Register (LINMASK) (offset = 6Ch)

31 24	23 16
Reserved	RX ID MASK
R-0	R/WL-0

15 8	7 0
Reserved	TX ID MASK
R-0	R/WL-0

LEGEND: R/W = Read/Write; R = Read only; WL = Write in LIN mode only; -*n* = value after reset

Table 29-43. LIN Mask Register (LINMASK) Field Descriptions

Bit	Field	Value	Description
31-24	Reserved	0	Reads return 0. Writes have no effect.
23-16	RX ID MASK	0-FFh	Receive ID mask. These bits are effective in LIN mode only. This 8-bit mask is used for filtering an incoming ID message and comparing it to the ID-byte. A compare match of the received ID with the RX ID MASK will set the ID RX flag and trigger an ID interrupt if enabled (SET ID INT in SCISETINT). A 0 bit in the mask indicates that bit is compared to the ID-byte. A 1 bit in the mask indicates that bit is filtered and therefore is not used in the compare.
15-8	Reserved	0	Reads return 0. Writes have no effect.
7-0	TX ID MASK	0-FFh	Transmit ID mask. These bits are effective in LIN mode only. This 8-bit mask is used for filtering an incoming ID message and comparing it to the ID-byte. A compare match of the received ID with the TX ID MASK will set the ID TX flag and trigger an ID interrupt if enabled (SET ID INT in SCISETINT). A 0 bit in the mask indicates that bit is compared to the ID-byte. A 1 bit in the mask indicates that bit is filtered and therefore is not used for the compare.

29.7.27 LIN Identification Register (LINID)

Figure 29-56 and Table 29-44 illustrate this register.

Figure 29-56. LIN Identification Register (LINID) (offset = 70h)

31 24	23 16
Reserved	RECEIVED ID
R-0	R-0

15 8	7 0
ID-SlaveTask BYTE	ID BYTE
R/WL-0	R/WL-0

LEGEND: R/W = Read/Write; R = Read only; WL = Write in LIN mode only; -*n* = value after reset

Table 29-44. LIN Identification Register (LINID) Field Descriptions

Bit	Field	Value	Description
31-24	Reserved	0	Reads return 0. Writes have no effect.
23-16	RECEIVED ID	0-FFh	Received identification. These bits are effective in LIN mode only. This byte contains the current message identifier. During header reception the received ID is copied from the SCIRXSHF register to this byte if there is no ID-parity error and there has been an RX/TX match.
15-8	ID-SLAVETASK BYTE	0-FFh	ID-SlaveTask Byte. These bits are effective in LIN mode only. This field contains the identifier to which the received ID of an incoming header will be compared to decide whether a receive response, a transmit response, or no action needs to be performed by the LIN node when a header with that particular ID is received.
7-0	ID BYTE	0-FFh	ID byte. This field is effective in LIN mode only. This byte is the LIN mode message ID. On a master node, a write to this register by the CPU initiates a header transmission. For a slave task, this byte is used for message filtering when HGEN CTRL = 0.

NOTE: For software compatibility with future LIN modules, the HGEN CTRL bit must be set to 1, the RX ID MASK field must be set to FFh, and the TX ID MASK field must be set to FFh.

29.7.28 LIN Transmit Buffer 0 Register (LINTD0)

Figure 29-57 and Table 29-45 illustrate this register.

Figure 29-57. LIN Transmit Buffer 0 Register (LINTD0) (offset = 74h)

31 24	23 16
TD0	TD1
R/W-0	R/W-0

15 8	7 0
TD2	TD3
R/W-0	R/W-0

LEGEND: R/W = Read/Write; -*n* = value after reset

Table 29-45. LIN Transmit Buffer 0 Register (LINTD0) Field Descriptions

Bit	Field	Value	Description
31-24	TD0	0-FFh	8-Bit transmit buffer 0. Byte 0 to be transmitted is written into this register and then copied to SCITXSHF for transmission. Once byte 0 is written in TD0 buffer, transmission will be initiated. **Note: TD<x-1> is equivalent to data byte <x> of the LIN frame.**
23-16	TD1	0-FFh	8-Bit transmit buffer 1. Byte 1 to be transmitted is written into this register and then copied to SCITXSHF for transmission.
15-8	TD2	0-FFh	8-Bit transmit buffer 2. Byte 2 to be transmitted is written into this register and then copied to SCITXSHF for transmission.
7-0	TD3	0-FFh	8-Bit transmit buffer 3. Byte 3 to be transmitted is written into this register and then copied to SCITXSHF for transmission.

29.7.29 LIN Transmit Buffer 1 Register (LINTD1)

Figure 29-58 and Table 29-46 illustrate this register.

Figure 29-58. LIN Transmit Buffer 1 Register (LINTD1) (offset = 78h)

31 24	23 16
TD4	TD5
R/W-0	R/W-0

15 8	7 0
TD6	TD7
R/W-0	R/W-0

LEGEND: R/W = Read/Write; -*n* = value after reset

Table 29-46. LIN Transmit Buffer 1 Register (LINTD1) Field Descriptions

Bit	Field	Value	Description
31-24	TD4	0-FFh	8-Bit transmit buffer 4. Byte 4 to be transmitted is written into this register and then copied to SCITXSHF for transmission. **Note: TD<x-1> is equivalent to data byte <x> of the LIN frame.**
23-16	TD5	0-FFh	8-Bit transmit buffer 5. Byte 5 to be transmitted is written into this register and then copied to SCITXSHF for transmission.
15-8	TD6	0-FFh	8-Bit transmit buffer 6. Byte 6 to be transmitted is written into this register and then copied to SCITXSHF for transmission.
7-0	TD7	0-FFh	8-Bit transmit buffer 7. Byte 7 to be transmitted is written into this register and then copied to SCITXSHF for transmission.

29.7.30 Maximum Baud Rate Selection Register (MBRS)

Figure 29-59 and Table 29-47 illustrate this register.

Figure 29-59. Maximum Baud Rate Selection Register (MBRS) (offset = 7Ch)

31	16
Reserved	
R-0	

15	13	12	0
Reserved		MBR	
R-0		R/WL-DACh	

LEGEND: R/W = Read/Write; R = Read only; WL = Write in LIN mode only; -*n* = value after reset

Table 29-47. Maximum Baud Rate Selection Register (MBRS) Field Descriptions

Bit	Field	Value	Description
31-13	Reserved	0	Reads return 0. Writes have no effect.
12-0	MBR	0-1FFFh	Maximum baud rate prescaler. This bit is effective in LIN mode only. This 13-bit prescaler is used during the synchronization phase (see Section 29.3.1.5.2) of a slave module if the ADAPT bit is set. In this way, a SCI/LIN slave using an automatic or select bit rate modes detects any LIN bus legal rate automatically. The MBR value should be programmed to allow a maximum baud rate that is not more than 10% above the expected operating baud rate in the LIN network. Otherwise, a 00h data byte could mistakenly be detected as a sync break. The default value for a 70-MHz VCLK is DACh. This MBR prescaler is used by the wake-up and idle time counters for a constant expiration time relative to a 20-kHz rate. $\text{MBR} = \frac{0.9\text{x VCLK}}{\text{maxbaudrate}} \quad (56)$

29.7.31 Input/Output Error Enable (IODFTCTRL) Register

Figure 29-60 and Table 29-48 illustrate this register. After the basic SCI/LIN module configuration, enable the required Error mode to be created followed by IODFT Key enable.

NOTE: 1) All the bits are used in IODFT mode only.

2) Each IODFT are expected to be checked individually.

3) ISFE Error will not be Flagged during IODFT mode.

Figure 29-60. Input/Output Error Enable Register (IODFTCTRL) (offset = 90h)

31	30	29	28	27	26	25	24
BEN	PBEN	CEN	ISFE	Reserved	FEN	PEN	BRKDT ENA
R/WL-0	R/WL-0	R/WL-0	R/WL-0	R-0	R/W-0	R/WC-0	R/WC-0
23		21	20	19	18		16
Reserved			PIN SAMPLE MASK		TX SHIFT		
R-0			R/W-0		R/W-0		
15			12	11			8
Reserved				IODFTENA			
R-0				R/WP-5h			
7					2	1	0
Reserved						LPB ENA	RXP ENA
R-0						R/WP-0	R/WP-0

LEGEND: R/W = Read/Write; R = Read only; WL = Write in LIN mode only; WC = Write in SCI-compatible mode only; WP = Write in privilege mode only; -*n* = value after reset

Table 29-48. Input/Output Error Enable Register (IODFTCTRL) Field Descriptions

Bit	Field	Value	Description
31	BEN		Bit error enable. This bit is effective in LIN mode only. This bit is used to create a bit error.
		0	No bit error is created.
		1	The bit received is ORed with 1 and passed to the bit monitor circuitry.
30	PBEN		Physical bus error enable. This bit is effective in LIN mode only. This bit is used to create a physical bus error.
		0	No error is created.
		1	The bit received during synch break field transmission is ORed with 1 and passed to the bit monitor circuitry.
29	CEN		Checksum error enable. This bit is effective in LIN mode only. This bit is used to create a checksum error.
		0	No error is created.
		1	The polarity of the CTYPE (checksum type) in the receive checksum calculator is changed so that a checksum error is occurred.
28	ISFE		Inconsistent synch field (ISF) error enable. This bit is effective in LIN mode only. This bit is used to create an ISF error.
		0	No error is created.
		1	The bit widths in the synch field are varied so that the ISF check fails and the error flag is set.
27	Reserved	0	Reads return 0. Writes have no effect.
26	FEN		Frame error enable. This bit is used to create a frame error.
		0	No error is created.
		1	The stop bit received is ANDed with 0 and passed to the stop bit check circuitry.

Table 29-48. Input/Output Error Enable Register (IODFTCTRL) Field Descriptions (continued)

Bit	Field	Value	Description
25	PEN		Parity error enable. This bit is effective in SCI-compatible mode only. This bit is used to create a parity error.
		0	No parity error occurs.
		1	The parity bit received is toggled so that a parity error occurs.
24	BRKDT ENA		Break detect error enable. This bit is effective in SCI-compatible mode only. This bit is used to create a BRKDT error.
		0	No error is created.
		1	The stop bit of the frame is ANDed with 0 and passed to the RSM so that a frame error occurs. Then the RX pin is forced to continuous low for 10 T_{bit} so that a BRKDT error occurs.
32-21	Reserved	0	Reads return 0. Writes have no effect.
20-19	PIN SAMPLE MASK		Pin sample mask. These bits define the sample number at which the TX pin value that is being transmitted will be inverted to verify the receive pin samples majority detection circuitry. **Note: In IODFT mode testing for pin_sample mask must be done with prescalar P programmed greater than 2 (P > 2).**
		0	No mask is used.
		1h	Invert the TX Pin value at TBIT_CENTER.
		2h	Invert the TX Pin value at TBIT_CENTER + SCLK.
		3h	Invert the TX Pin value at TBIT_CENTER + 2 SCLK.
18-16	TX SHIFT		Transmit shift. These bits define the amount by which the value on TX pin is delayed so that the value on the RX pin is asynchronous. This feature is not applicable to the start bit.
		0	No delay occurs.
		1h	The value is delayed by 1 SCLK.
		2h	The value is delayed by 2 SCLK.
		3h	The value is delayed by 3 SCLK.
		4h	The value is delayed by 4 SCLK.
		5h	The value is delayed by 5 SCLK.
		6h	The value is delayed by 6 SCLK.
		7h	The value is delayed by 7 SCLK.
15-12	Reserved	0	Reads return 0. Writes have no effect.
11-8	IODFTENA		IODFT enable key. Write access permitted in Privilege mode only.
		Ah	IODFT is enabled.
		All other values	IODFT is disabled.
7-2	Reserved	0	Reads return 0. Writes have no effect.
1	LPB ENA		Module loopback enable. Write access permitted in Privilege mode only. **Note: In analog loopback mode the complete communication path through the I/Os can be tested, whereas in digital loopback mode the I/O buffers are excluded from this path.**
		0	Digital loopback is enabled.
		1	Analog loopback is enabled in module I/O DFT mode when IODFTENA = 1010.
0	RXP ENA		Module analog loopback through receive pin enable. Write access permitted in Privilege mode only. This bit defines whether the I/O buffers for the transmit or the receive pin are included in the communication path (in analog loopback mode).
		0	Analog loopback through the transmit pin is enabled.
		1	Analog loopback through the receive pin is enabled.

Serial Communication Interface (SCI) Module

This chapter contains the description of the serial communication interface (SCI) module.

Topic **Page**

30.1 Introduction

The SCI module is a universal asynchronous receiver-transmitter that implements the standard nonreturn to zero format. The SCI can be used to communicate, for example, through an RS-232 port or over a K-line.

30.1.1 SCI Features

The following are the features of the SCI module:

- Standard universal asynchronous receiver-transmitter (UART) communication
- Supports full- or half-duplex operation
- Standard nonreturn to zero (NRZ) format
- Double-buffered receive and transmit functions
- Supports two individually enabled interrupt lines: level 0 and level 1
- Configurable frame format of 3 to 13 bits per character based on the following:
  - Data word length programmable from one to eight bits
  - Additional address bit in address-bit mode
  - Parity programmable for zero or one parity bit, odd or even parity
  - Stop programmable for one or two stop bits
- Asynchronous or isosynchronous communication modes with no CLK pin
- Two multiprocessor communication formats allow communication between more than two devices
- Sleep mode is available to free CPU resources during multiprocessor communication and then wake up to receive an incoming message
- The 24-bit programmable baud rate supports 2^{24} different baud rates provide high accuracy baud rate selection
- Capability to use Direct Memory Access (DMA) for transmit and receive data
- Four error flags and Five status flags provide detailed information regarding SCI events
- Two external pins: SCIRX and SCITX

NOTE: SCI module does not support UART Hardware Flow Control. This feature can be implemented in Software using a General Purpose I/O pin.

30.1.2 Block Diagram

Three Major components of the SCI Module are:

- Transmitter
- Baud Clock Generator
- Receiver

Transmitter (TX) contains two major registers to perform double buffering:

- The transmitter data buffer register (SCITD) contains data loaded by the CPU to be transferred to the shift register for transmission.
- The transmitter shift register (SCITXSHF) loads data from the data buffer (SCITD) and shifts data onto the SCITX pin, one bit at a time.

Baud Clock Generator

- A programmable baud generator produces a baud clock scaled from VCLK.

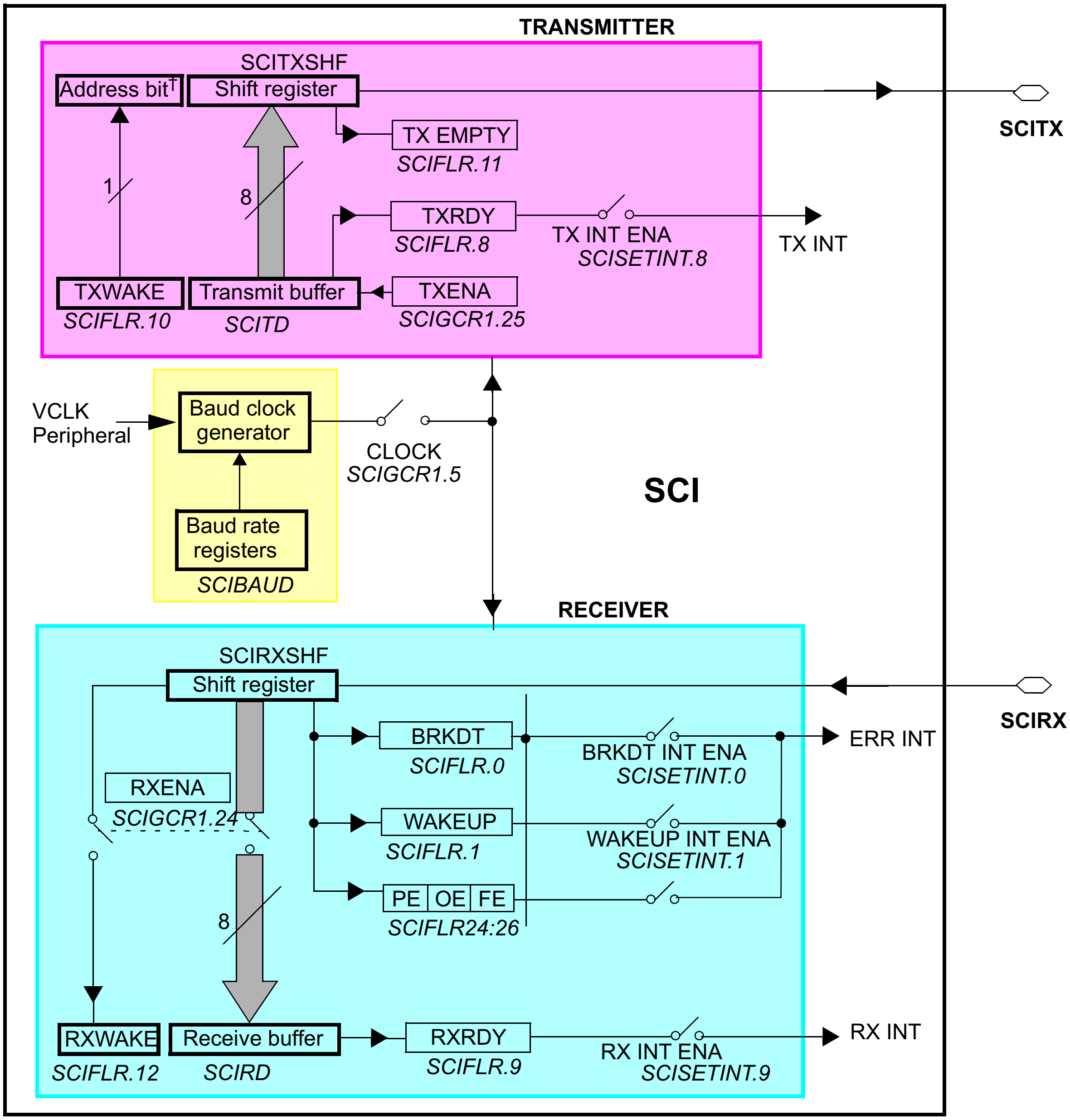
Receiver (RX) contains two major registers to perform double buffering:

- The receiver shift register (SCIRXSHF) shifts data in from the SCIRX pin one bit at a time and transfers completed data into the receive data buffer.
- The receiver data buffer register (SCIRD) contains received data transferred from the receiver shift register

The SCI receiver and transmitter are double-buffered, and each has its own separate enable and interrupt bits. The receiver and transmitter may each be operated independently or simultaneously in full duplex mode.

To ensure data integrity, the SCI checks the data it receives for breaks, parity, overrun, and framing errors. The bit rate (baud) is programmable to over 16 million different rates through a 24-bit baud-select register. Figure 30-1 shows the detailed SCI block diagram.

Figure 30-1. Detailed SCI Block Diagram



30.2 SCI Communication Formats

The SCI module can be configured to meet the requirements of many applications. Because communication formats vary depending on the specific application, many attributes of the SCI are user configurable. The list below describes these configuration options:

- SCI Frame format
- SCI Timing modes
- SCI Baud rate
- SCI Multiprocessor modes

30.2.1 SCI Frame Formats

The SCI uses a programmable frame format. All frames consist of the following:

- One start bit
- One to eight data bits
- Zero or one address bit
- Zero or one parity bit
- One or two stop bits

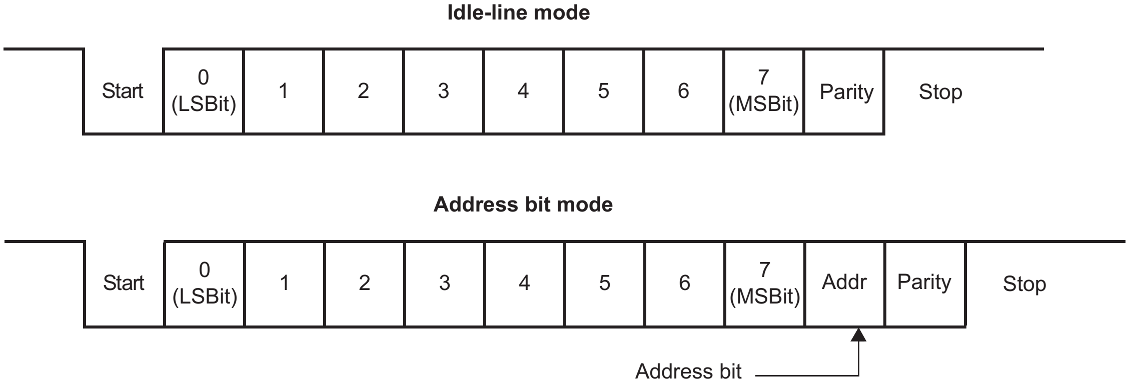
The frame format for both the transmitter and receiver is programmable through the bits in the SCIGCR1 register. Both receive and transmit data is in nonreturn to zero (NRZ) format, which means that the transmit and receive lines are at logic high when idle. Each frame transmission begins with a start bit, in which the transmitter pulls the SCI line low (logic low). Following the start bit, the frame data is sent and received least significant bit first (LSB).

An address bit is present in each frame if the SCI is configured to be in address-bit mode but is not present in any frame if the SCI is configured for idle-line mode. The format of frames with and without the address bit is illustrated in Figure 30-2.

A parity bit is present in every frame when the PARITY ENA bit is set. The value of the parity bit depends on the number of one bits in the frame and whether odd or even parity has been selected via the PARITY ENA bit. Both examples in Figure 30-2 have parity enabled.

All frames include one stop bit, which is always a high level. This high level at the end of each frame is used to indicate the end of a frame to ensure synchronization between communicating devices. Two stop bits are transmitted if the STOP bit in SCIGCR1 register is set. The examples shown in Figure 30-2 use one stop bit per frame.

Figure 30-2. Typical SCI Data Frame Formats



30.2.2 SCI Timing Mode

The SCI can be configured to use asynchronous or isosynchronous timing using TIMING MODE bit in SCIGCR1 register.

30.2.2.1 Asynchronous Timing Mode

The asynchronous timing mode uses only the receive and transmit data lines to interface with devices using the standard universal asynchronous receiver- transmitter (UART) protocol.

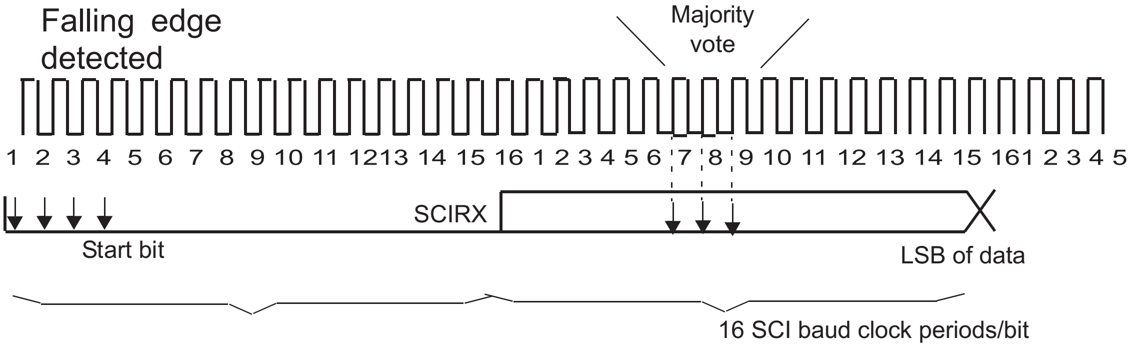
In the asynchronous timing mode, each bit in a frame has a duration of 16 SCI baud clock periods. Each bit therefore consists of 16 samples (one for each clock period). When the SCI is using asynchronous mode, the baud rates of all communicating devices must match as closely as possible. Receive errors result from devices communicating at different baud rates.

With the receiver in the asynchronous timing mode, the SCI detects a valid start bit if the first four samples after a falling edge on the SCIRX pin are of logic level 0. As soon as a falling edge is detected on SCIRX, the SCI assumes that a frame is being received and synchronizes itself to the bus.

To prevent interpreting noise as Start bit SCI expects SCIRX line to be low for at least four contiguous SCI baud clock periods to detect a valid start bit. The bus is considered idle if this condition is not met. When a valid start bit is detected, the SCI determines the value of each bit by sampling the SCIRX line value during the seventh, eighth, and ninth SCI baud clock periods. A majority vote of these three samples is used to determine the value stored in the SCI receiver shift register. By sampling in the middle of the bit, the SCI reduces errors caused by propagation delays and rise and fall times and data line noises. Figure 30-3 illustrates how the receiver samples a start bit and a data bit in asynchronous timing mode.

The transmitter transmits each bit for a duration of 16 SCI baud clock periods. During the first clock period for a bit, the transmitter shifts the value of that bit onto the SCITX pin. The transmitter then holds the current bit value on SCITX for 16 SCI baud clock periods.

Figure 30-3. Asynchronous Communication Bit Timing



30.2.2.2 Isosynchronous Timing Mode

In isosynchronous timing mode, each bit in a frame has a duration of exactly 1 baud clock period and therefore consists of a single sample. With this timing configuration, the transmitter and receiver are required to make use of the SCICLK pin to synchronize communication with other SCI. **This mode is not fully supported on this device because SCICLK pin is not available.**

30.2.3 SCI Baud Rate

The SCI has an internally generated serial clock determined by the peripheral VCLK and the prescalers BAUD. The SCI uses the 24-bit integer prescaler BAUD value in the BRS register to select the required baud rates.

In asynchronous timing mode, the SCI generates a baud clock according to the following formula:

$$\textit{Asynchronous baud value} = \frac{\text{VCLK Frequency}}{16 * (\text{BAUD} + 1)}$$

For BAUD = 0,

$$\textit{Asynchronous baud value} = \frac{\text{VCLK Frequency}}{32} \tag{57}$$

In isosynchronous timing mode, the SCI generates a baud clock according to the following formula:

$$\textit{Isosynchronous baud value} = \frac{\text{VCLK Frequency}}{\text{BAUD} + 1}$$

For BAUD = 0,

$$\textit{Isosynchronous baud value} = \frac{\text{VCLK Frequency}}{32} \tag{58}$$

30.2.4 SCI Multiprocessor Communication Modes

In some applications, the SCI may be connected to more than one serial communication device. In such a multiprocessor configuration, several frames of data may be sent to all connected devices or to an individual device. In the case of data sent to an individual device, the receiving devices must determine when they are being addressed. When a message is not intended for them, the devices can ignore the following data. When only two devices make up the SCI network, addressing is not needed, so multiprocessor communication schemes are not required.

SCI supports two multiprocessor Communication Modes which can be selected using COMM MODE bit:

- Idle-Line Mode
- Address Bit Mode

When the SCI is not used in a multiprocessor environment, software can consider all frames as data frames. In this case, the only distinction between the idle-line and address-bit modes is the presence of an extra bit (the address bit) in each frame sent with the address-bit protocol.

The SCI allows full-duplex communication where data can be sent and received via the transmit and receive pins simultaneously. However, the protocol used by the SCI assumes that only one device transmits data on the same bus line at any one time. No arbitration is done by the SCI.

NOTE: Avoid Transmitting Simultaneously on the Same Serial Bus

The system designer must ensure that devices connected to the same serial bus line do not attempt to transmit simultaneously. If two devices are transmitting different data, the resulting bus conflict could damage the device..

30.2.4.1 Idle-Line Multiprocessor Modes

In idle-line multiprocessor mode, a frame that is preceded by an idle period (10 or more idle bits) is an address frame. A frame that is preceded by fewer than 10 idle bits is a data frame. Figure 30-4 illustrates the format of several blocks and frames with idle-line mode.

There are two ways to transmit an address frame using idle-line mode:

Method 1: In software, deliberately leave an idle period between the transmission of the last data frame of the previous block and the address frame of the new block.

Method 2: Configure the SCI to automatically send an idle period between the last data frame of the previous block and the address frame of the new block.

Although Method 1 is only accomplished by a delay loop in software, Method 2 can be implemented by using the transmit buffer and the TXWAKE bit in the following manner:

Step1 : Write a 1 to the TXWAKE bit.

Step2 : Write a dummy data value to the SCITD register. This triggers the SCI to begin the idle period as soon as the transmitter shift register is empty.

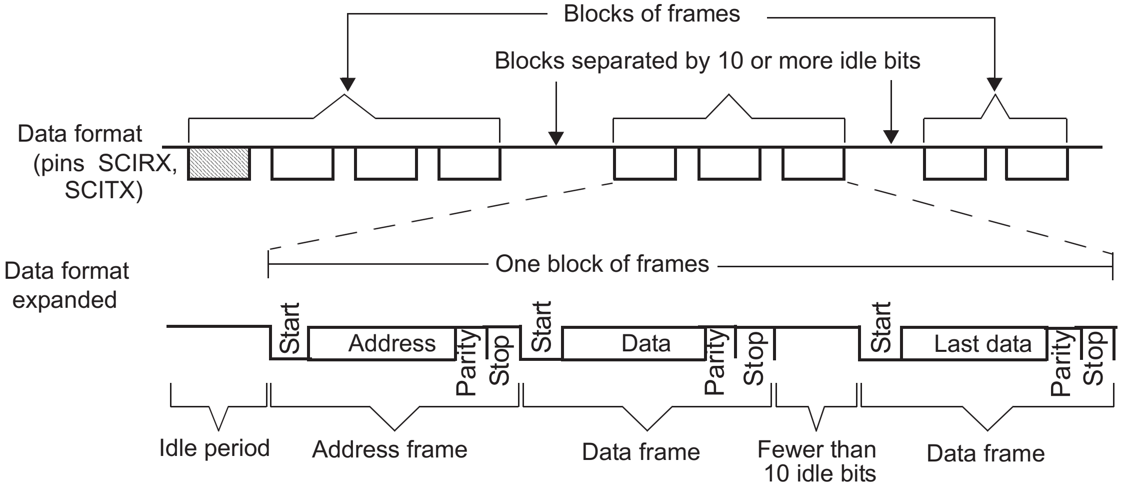
Step3 : Wait for the SCI to clear the TXWAKE flag.

Step4 : Write the address value to SCITD.

As indicated by Step 3, software should wait for the SCI to clear the TXWAKE bit. However, the SCI clears the TXWAKE bit at the same time it sets TXRDY (that is, transfers data from SCITD into SCITXSHF). Therefore, if the TX INT ENA bit is set, the transfer of data from SCITD to SCITXSHF causes an interrupt to be generated at the same time that the SCI clears the TXWAKE bit. If this interrupt method is used, software is not required to poll the TXWAKE bit waiting for the SCI to clear it.

When idle-line multiprocessor communications are used, software must ensure that the idle time exceeds 10 bit periods before addresses (using one of the methods mentioned above), and software must also ensure that data frames are written to the transmitter quickly enough to be sent without a delay of 10 bit periods between frames. Failure to comply with these conditions will result in data interpretation errors by other devices receiving the transmission.

Figure 30-4. Idle-Line Multiprocessor Communication Format



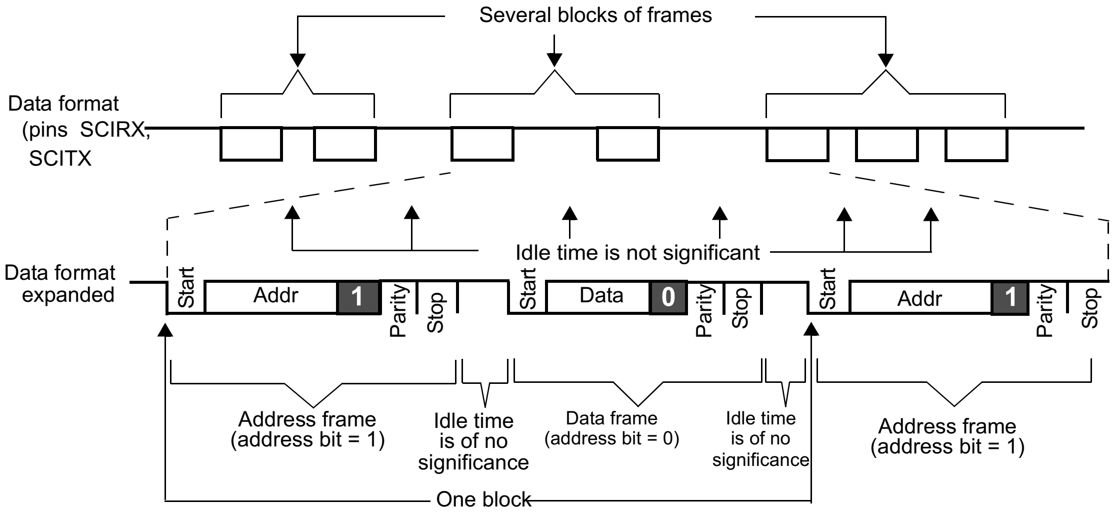
30.2.4.2 Address-Bit Multiprocessor Mode

In the address-bit protocol, each frame has an extra bit immediately following the data field called an address bit. A frame with the address bit set to 1 is an address frame; a frame with the address bit set to 0 is a data frame. The idle period timing is irrelevant in this mode. Figure 30-5 illustrates the format of several blocks and frames with the address-bit mode.

When address-bit mode is used, the value of the TXWAKE bit is the value sent as the address bit. To send an address frame, software must set the TXWAKE bit. This bit is cleared as the contents of the SCITD are shifted from the TXWAKE register so that all frames sent are data except when the TXWAKE bit is written as a 1.

No dummy write to SCITD is required before an address frame is sent in address-bit mode. The first byte written to SCITD after the TXWAKE bit is written to 1 is transmitted with the address bit set when address-bit mode is used.

Figure 30-5. Address-Bit Multiprocessor Communication Format



30.3 SCI Interrupts

The SCI module has two interrupt lines, level 0 and level 1, to the vectored interrupt manager (VIM) module (see Figure 30-6). Two offset registers SCIINTVECT0 and SCIINTVECT1 determine which flag triggered the interrupt according to the respective priority encoders. Each interrupt condition has a bit to enable/disable the interrupt in the SCISETINT and SCICLRINT registers, respectively.

Each interrupt also has a bit that can be set as interrupt level 0 (INT0) or as interrupt level 1 (INT1). By default, interrupts are in interrupt level 0. SCISETINTLVL sets a given interrupt to level1. SCICLEARINTLVL resets a given interrupt level to the default level 0.

The interrupt vector registers SCIINTVECT0 and SCIINTVECT1 return the vector of the pending interrupt line INT0 or INT1. If more than one interrupt is pending, the interrupt vector register holds the highest priority interrupt.

Figure 30-6. General Interrupt Scheme

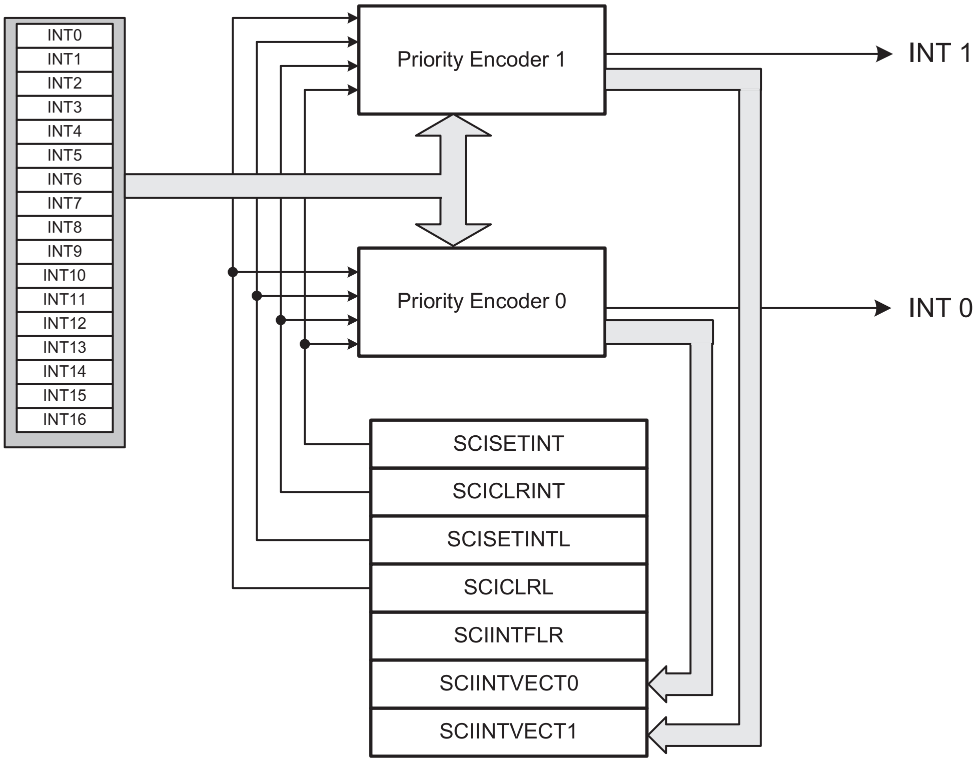
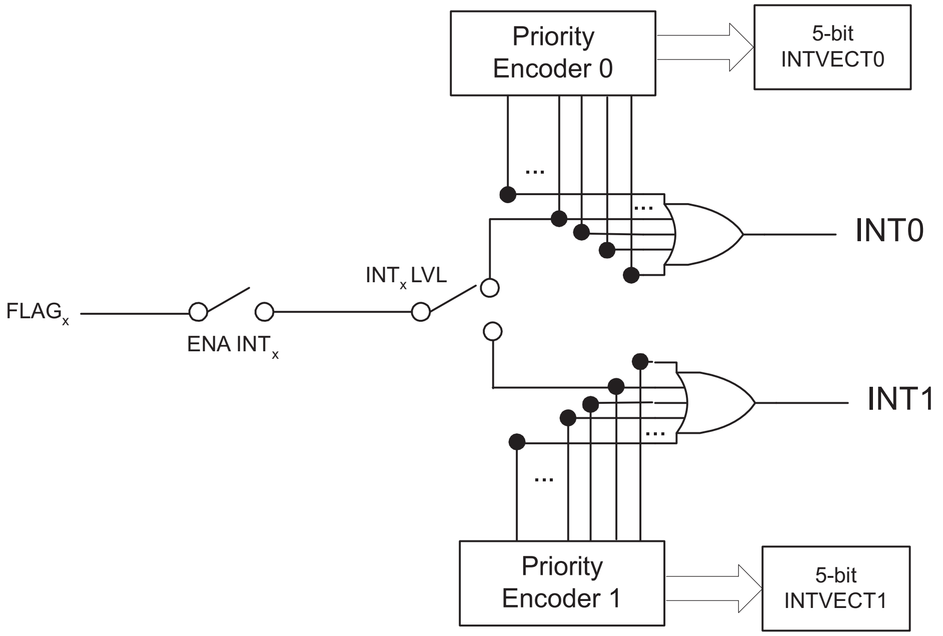


Figure 30-7. Interrupt Generation for Given Flags



30.3.1 Transmit Interrupt

To use transmit interrupt functionality, SET TX INT bit must be enabled and SET TX DMA bit must be cleared. The transmit ready (TXRDY) flag is set when the SCI transfers the contents of SCITD to the shift register, SCITXSHF. The TXRDY flag indicates that SCITD is ready to be loaded with more data. In addition, the SCI sets the TX EMPTY bit if both the SCITD and SCITXSHF registers are empty. If the SET TX INT bit is set, then a transmit interrupt is generated when the TXRDY flag goes high. Transmit Interrupt is not generated immediately after setting the SET TX INT bit unlike transmit DMA request. Transmit Interrupt is generated only after the first transfer from SCITD to SCITXSHF, that is first data has to be written to SCITD by the User before any interrupt gets generated. To transmit further data the user can write data to SCITD in the transmit Interrupt service routine.

Writing data to the SCITD register clears the TXRDY bit. When this data has been moved to the SCITXSHF register, the TXRDY bit is set again. The interrupt request can be suspended by setting the CLR TX INT bit; however, when the SET TX INT bit is again set to 1, the TXRDY interrupt is asserted again. The transmit interrupt request can be eliminated until the next series of values is written to SCITD, by disabling the transmitter via the TXENA bit, by a software reset SWnRST, or by a device hardware reset.

30.3.2 Receive Interrupt

The receive ready (RXRDY) flag is set when the SCI transfers newly received data from SCIRXSHF to SCIRD. The RXRDY flag therefore indicates that the SCI has new data to be read. Receive interrupts are enabled by the SET RX INT bit. If the SET RX INT is set when the SCI sets the RXRDY flag, then a receive interrupt is generated. The received data can be read in the Interrupt Service routine.

On a device with both SCI and a DMA controller, the bits SET RX DMA ALL and SET RX DMA must be cleared to select interrupt functionality.

30.3.3 WakeUp Interrupt

SCI sets the WAKEUP flag if bus activity on the RX line either prevents power-down mode from being entered, or RX line activity causes an exit from power-down mode. If enabled (SET WAKEUP INT), wakeup interupt is triggered once WAKEUP flag is set.

30.3.4 Error Interrupts

The following error detection features are supported with Interrupt by the SCI module:

- Parity errors (PE)
- Frame errors (FE)
- Break Detect errors (BRKDT)
- Overrun errors (OE)

If any of these errors (PE, FE, BRKDT, OE) is flagged, an interrupt for the flagged errors will be generated if enabled. A message is valid for both the transmitter and the receiver if there is no error detected until the end of the frame. Each of these flags is located in the receiver status (SCIFLR) register. Further details on these flags are explained in SCIFLR register description.

The SCI module supports following 7 interrupts as seen in Table 30-1.

Table 30-1. SCI Interrupts

Offset [1]	Interrupt
0	Reserved
1	Wakeup
2	Reserved
3	Parity error
4	Reserved
5	Reserved
6	Frame error
7	Break detect error
8	Reserved
9	Overrun error
10	Reserved
11	Receive
12	Transmit
13 - 15	Reserved

[1] Offset 1 is the highest priority. Offset 16 is the lowest priority.

30.4 SCI DMA Interface

DMA requests for receive (RXDMA request) and transmit (TXDMA request) are available for the SCI module. Refer to the DMA module chapter for DMA module configurations.

30.4.1 Receive DMA Requests

This DMA functionality is enabled/disabled by the CPU using the SET RX DMA/CLR RX DMA bits, respectively.

The receiver DMA request is set when a frame is received successfully and DMA functionality has been previously enabled. The RXRDY flag is set when the SCI transfers newly received data from the SCIRXSHF register to the SCIRD buffer. The RXRDY flag therefore indicates that the SCI has new data to be read. Receive DMA requests are enabled by the SET RX INT bit.

Parity, overrun, break detect, wake-up, and framing errors generate an error interrupt request immediately upon detection, if enabled, even if the device is in the process of a DMA data transfer. The DMA transfer is postponed until the error interrupt is served. The error interrupt can delete this particular DMA request by reading the receive buffer.

In multiprocessor mode, the SCI can generate receiver interrupts for address frames and DMA requests for data frames. This is controlled by an extra select bit SET RX DMA ALL.

If the SET RX DMA ALL bit is set and the SET RX DMA bit is set when the SCI sets the RXRDY flag, then a receive DMA request is generated for address and data frames.

If the SET RX DMA ALL bit is cleared and the SET RX DMA bit is set when the SCI sets the RXRDY flag upon receipt of a data frame, then a receive DMA request is generated. Receive interrupt requests are generated for address frames.

In multiprocessor mode with the SLEEP bit set, no DMA is generated for received data frames. The software must clear the SLEEP bit before data frames can be received. Table 30-2 specifies the bit values for DMA requests in multiprocessor modes.

In multiprocessor mode, the SCI can generate receiver interrupts for address frames and DMA requests for data frames or DMA requests for both. This is controlled by the SET RX DMA ALL bit.

In multiprocessor mode with the SLEEP bit set, no DMA is generated for received data frames. The software must clear the SLEEP bit before data frames can be received.

Table 30-2. DMA and Interrupt Requests in Multiprocessor Modes

SET RX INT	SET RX DMA	SET RX DMA ALL	ADDR FRAME INT	ADDR FRAME DMA	DATA FRAME INT	DATA FRAME DMA
0	0	x	N	N	N	N
0	1	0	Y	N	N	Y
0	1	1	N	Y	N	Y
1	0	x	Y	N	Y	N
1	1	0	Y	N	Y	Y
1	1	1	Y	Y	Y	Y

30.4.2 Transmit DMA Requests

DMA functionality is enabled/disabled by the CPU with SET TX DMA/CLR TX DMA bits, respectively.

The TXRDY flag is set when the SCI transfers the contents of SCITD to SCITXSHF. The TXRDY flag indicates that SCITD is ready to be loaded with more data. In addition, the SCI sets the TX EMPTY bit if both the SCITD and SCITXSHF registers are empty.

Transmit DMA requests are enabled by the setting SET TX DMA and SET TX INT bits. If the SET TX DMA bit is set, then a TX DMA request is sent to the DMA when data is written to SCITD and TXRDY is set. In other words, CPU needs to write the first data to start a DMA block transfer. For example, we want to transmit a data buffer of 20 bytes. DMA will be set up to transmit 19 bytes. The first data for DMA to transfer is the second byte in the buffer. CPU will have to write the first byte in the buffer to the SCITD register to start the transfer.

.

30.5 SCI Configurations

Before the SCI sends or receives data, its registers should be properly configured. Upon power-up or a system-level reset, each bit in the SCI registers is set to a default state. The registers are writable only after the RESET bit in the SCIGCR0 register is set to 1. Of particular importance is the SWnRST bit in the SCIGCR1 register. The SWnRST is an active-low bit initialized to 0 and keeps the SCI in a reset state until it is programmed to 1. Therefore, all SCI configuration should be completed before a 1 is written to the SWnRST bit.

The following list details the configuration steps that software should perform prior to the transmission or reception of data. As long as the SWnRST bit is cleared to 0 the entire time that the SCI is being configured, the order in which the registers are programmed is not important.

- Enable SCI by setting the RESET bit to 1.
- Clear the SWnRST bit to 0 before SCI is configured.
- Select the desired frame format by programming the SCIGCR1 register.
- Set both the RX FUNC and TX FUNC bits in SCIPIO0 to 1 to configure the SCIRX and SCITX pins for SCI functionality.
- Select the baud rate to be used for communication by programming the BRS register.
- Set the CLOCK bit in SCIGCR1 to 1 to select the internal clock.
- Set the CONT bit in SCIGCR1 to 1 to make SCI not halt for an emulation breakpoint until its current reception or transmission is complete (this bit is used only in an emulation environment).
- Set LOOP BACK bit in SCIGCR1 to 1 to connect the transmitter to the receiver internally (this feature is used to perform a self-test).
- Set the RXENA bit in SCIGCR1 to 1, if data is to be received.
- Set the TXENA bitin SCIGCR1 to 1, if data is to be transmitted.
- Set the SWnRST bit to 1 after SCI is configured.
- Perform receiving or transmitting data (see Section 30.5.1 or Section 30.5.2).

30.5.1 Receiving Data

The SCI receiver is enabled to receive messages if both the RX FUNC bit and the RXENA bit are set to 1. If the RX FUNC bit is not set, the SCIRX pin functions as a general-purpose I/O pin rather than as an SCI function pin. After a valid idle period is detected, data is automatically received as it arrives on the SCIRX pin.

SCI sets the RXRDY bit when it transfers newly received data from SCIRXSHF to SCIRD. The SCI clears the RXRDY bit after the new data in SCIRD has been read. Also, as data is transferred from SCIRXSHF to SCIRD, the SCI sets the FE, OE, or PE flags if any of these error conditions were detected in the received data. These error conditions are supported with configurable interrupt capability. The wake-up and break-detect status bits are also set if one of these errors occurs, but they do not necessarily occur at the same time that new data is being loaded into SCIRD.

You can receive data by:

1. Polling Receive Ready Flag
2. Receive Interrupt
3. DMA

In polling method, software can poll for the RXRDY bit and read the data from SCIRD register once RXRDY is set high. The CPU is unnecessarily overloaded by selecting the polling method. To avoid this, you can use either the interrupt or DMA method. To use the interrupt method, the SET RX INT bit is set. To use the DMA method, the SET RX DMA bit is set. Either an interrupt or a DMA request is generated the moment the RXRDY bit is set.

30.5.2 Transmitting Data

The SCI transmitter is enabled if both the TX FUNC bit and the TXENA bit are set to 1. If the TX FUNC bit is not set, the SCITX pin functions as a general-purpose I/O pin rather than as an SCI function pin. Any value written to the SCITD before TXENA is set to 1 is not transmitted. Both of these control bits allow for the SCI transmitter to be held inactive independently of the receiver.

SCI waits for data to be written to SCITD, transfers it to SCITXSHF, and transmits the data. The TXRDY and TX EMPTY bits indicate the status of the transmit buffers. That is, when the transmitter is ready for data to be written to SCITD, the TXRDY bit is set. Additionally, if both SCITD and SCITXSHF are empty, then the TX EMPTY bit is also set.

You can transmit data by:

1. Polling Transmit Ready Flag
2. Transmit Interrupt
3. DMA

In polling method, software can poll for the TXRDY bit to go high before writing the data to the SCITD register. The CPU is unnecessarily overloaded by selecting the polling method. To avoid this, you can use either the interrupt or DMA method. To use the interrupt method, the SET TX INT bit is set. To use the DMA method, the SET TX DMA bit is set. Either an interrupt or a DMA request is generated the moment the TXRDY bit is set. When the SCI has completed transmission of all pending frames, the SCITXSHF register and SCITD are empty, the TXRDY bit is set, and an interrupt/DMA request is generated, if enabled. Because all data has been transmitted, the interrupt/DMA request should be halted. This can either be done by disabling the transmit interrupt (CLR TX INT) / DMA request (CLR TX DMA bit) or by disabling the transmitter (clear TXENA bit).

NOTE: The TXRDY flag cannot be cleared by reading the corresponding interrupt offset in the SCIINTVECT0 or SCIINTVECT1 register.

30.6 SCI Low-Power Mode

The SCI can be put in either local or global low-power mode. Global low-power mode is asserted by the system and is not controlled by the SCI. During global low-power mode, all clocks to the SCI are turned off so the module is completely inactive.

Local low-power mode is asserted by setting the POWERDOWN bit; setting this bit stops the clocks to the SCI internal logic and the module registers. Setting the POWERDOWN bit causes the SCI to enter local low-power mode and clearing the POWERDOWN bit causes SCI to exit from local low-power mode. All the registers are accessible during local power-down mode as any register access enables the clock to SCI for that particular access alone.

The wake-up interrupt is used to allow the SCI to exit low-power mode automatically when a low level is detected on the SCIRX pin and also this clears the POWERDOWN bit. If wake-up interrupt is disabled, then the SCI immediately enters low-power mode whenever it is requested and also any activity on the SCIRX pin does not cause the SCI to exit low-power mode.

NOTE: Enabling Local Low-Power Mode During Receive and Transmit

If the wake-up interrupt is enabled and low-power mode is requested while the receiver is receiving data, then the SCI immediately generates a wake-up interrupt to clear the powerdown bit and prevents the SCI from entering low-power mode and thus completes the current reception. Otherwise, if the wake-up interrupt is disabled, then the SCI completes the current reception and then enters the low-power mode.

30.6.1 Sleep Mode for Multiprocessor Communication

When the SCI receives data and transfers that data from SCIRXSHF to SCIRD, the RXRDY bit is set and if RX INT ENA is set, the SCI also generates an interrupt. The interrupt triggers the CPU to read the newly received frame before another one is received. In multiprocessor communication modes, this default behavior may be enhanced to provide selective indication of new data. When SCI receives an address frame that does not match its address, the device can ignore the data following this non-matching address until the next address frame by using sleep mode. Sleep mode can be used with both idle-line and address-bit multiprocessor modes.

If sleep mode is enabled by the SLEEP bit, then the SCI transfers data from SCIRXSHF to SCIRD only for address frames. Therefore, in sleep mode, all data frames are assembled in the SCIRXSHF register without being shifted into the SCIRD and without initiating a receive interrupt or DMA request. Upon reception of an address frame, the contents of the SCIRXSHF are moved into SCIRD, and the software must read SCIRD and determine if the SCI is being addressed by comparing the received address against the address previously set in the software and stored somewhere in memory (the SCI does not have hardware available for address comparison). If the SCI is being addressed, the software must clear the SLEEP bit so that the SCI will load SCIRD with the data of the data frames that follow the address frame.

When the SCI has been addressed and sleep mode has been disabled (in software) to allow the receipt of data, the SCI should check the RXWAKE bit (SCIFLR.12) to determine when the next address has been received. This bit is set to 1 if the current value in SCIRD is an address and set to 0 if SCIRD contains data. If the RXWAKE bit is set, then software should check the address in SCIRD against its own address. If it is still being addressed, then sleep mode should remain disabled. Otherwise, the SLEEP bit should be set again.

Following is a sequence of events typical of sleep mode operation:

- The SCI is configured and both sleep mode and receive actions are enabled.
- An address frame is received and a receive interrupt is generated.
- Software compares the received address frame against that set by software and determines that the SCI is not being addressed, so the value of the SLEEP bit is not changed.
- Several data frames are shifted into SCIRXSHF, but no data is moved to SCIRD and no receive interrupts are generated.
- A new address frame is received and a receive interrupt is generated.
- Software compares the received address frame against that set by software and determines that the SCI is being addressed and clears the SLEEP bit.
- Data shifted into SCIRXSHF is transferred to SCIRD, and a receive interrupt is generated after each data frame is received.
- In each interrupt routine, software checks RXWAKE to determine if the current frame is an address frame.
- Another address frame is received, RXWAKE is set, software determines that the SCI is not being addressed and sets the SLEEP bit back to 1. No receive interrupts are generated for the data frames following this address frame.

By ignoring data frames that are not intended for the device, fewer interrupts are generated. These interrupts would otherwise require CPU intervention to read data that is of no significance to this specific device. Using sleep mode can help free some CPU resources.

Except for the RXRDY flag, the SCI continues to update the receiver status flags (see Table 30-11) while sleep mode is active. In this way, if an error occurs on the receive line, an application can immediately respond to the error and take the appropriate corrective action.

Because the RXRDY bit is not updated for data frames when sleep mode is enabled, the SCI can enable sleep mode and use a polling algorithm if desired. In this case, when RXRDY is set, software knows that a new address has been received. If the SCI is not being addressed, then the software should not change the value of the SLEEP bit and should continue to poll RXRDY.

30.7 SCI Control Registers

These registers are accessible in 8-, 16-, and 32-bit reads or writes. The SCI is controlled and accessed through the registers listed in Table 30-3. Among the features that can be programmed are the SCI communication and timing modes, baud rate value, frame format, DMA requests, and interrupt configuration. The base address for the control registers is FFF7 E500h for SCI3 and FFF7 E700h for SCI4.

Table 30-3. SCI Control Registers Summary

Offset	Acronym	Register Description	Section
00h	SCIGCR0	SCI Global Control Register 0	Section 30.7.1
04h	SCIGCR1	SCI Global Control Register 1	Section 30.7.2
0Ch	SCISETINT	SCI Set Interrupt Register	Section 30.7.3
10h	SCICLEARINT	SCI Clear Interrupt Register	Section 30.7.4
14h	SCISETINTLVL	SCI Set Interrupt Level Register	Section 30.7.5
18h	SCICLEARINTLVL	SCI Clear Interrupt Level Register	Section 30.7.6
1Ch	SCIFLR	SCI Flags Register	Section 30.7.7
20h	SCIINTVECT0	SCI Interrupt Vector Offset 0	Section 30.7.8
24h	SCIINTVECT1	SCI Interrupt Vector Offset 1	Section 30.7.9
28h	SCIFORMAT	SCI Format Control Register	Section 30.7.10
2Ch	BRS	Baud Rate Selection Register	Section 30.7.11
30h	SCIED	Receiver Emulation Data Buffer	Section 30.7.12.1
34h	SCIRD	Receiver Data Buffer	Section 30.7.12.2
38h	SCITD	Transmit Data Buffer	Section 30.7.12.3
3Ch	SCIPIO0	SCI Pin I/O Control Register 0	Section 30.7.13
40h	SCIPIO1	SCI Pin I/O Control Register 1	Section 30.7.14
44h	SCIPIO2	SCI Pin I/O Control Register 2	Section 30.7.15
48h	SCIPIO3	SCI Pin I/O Control Register 3	Section 30.7.16
4Ch	SCIPIO4	SCI Pin I/O Control Register 4	Section 30.7.17
50h	SCIPIO5	SCI Pin I/O Control Register 5	Section 30.7.18
54h	SCIPIO6	SCI Pin I/O Control Register 6	Section 30.7.19
58h	SCIPIO7	SCI Pin I/O Control Register 7	Section 30.7.20
5Ch	SCIPIO8	SCI Pin I/O Control Register 8	Section 30.7.21
90h	IODFTCTRL	Input/Output Error Enable Register	Section 30.7.22

30.7.1 SCI Global Control Register 0 (SCIGCR0)

The SCIGCR0 register defines the module reset. Figure 30-8 and Table 30-4 illustrate this register.

Figure 30-8. SCI Global Control Register 0 (SCIGCR0) [offset = 00]

31–16
Reserved
R-0

15–1	0
Reserved	RESET
R-0	R/WP-0

LEGEND: R/W = Read/Write; R = Read only; R/WP = Read/Write in privileged mode only; -*n* = value after reset

Table 30-4. SCI Global Control Register 0 (SCIGCR0) Fied Descriptions

Bit	Field	Value	Description
31-1	Reserved	0	Reads return 0. Writes have no effect.
0	RESET		This bit resets the SCI module.
		0	SCI module is in reset.
		1	SCI module is out of reset.
			Note: Read/Write in privileged mode only.

30.7.2 SCI Global Control Register 1 (SCIGCR1)

The SCIGCR1 register defines the frame format, protocol, and communication mode used by the SCI. Figure 30-9 and Table 30-5 illustrate this register.

Figure 30-9. SCI Global Control Register 1 (SCIGCR1) [offset = 04h]

31–26	25	24
Reserved	TXENA	RXENA
R-0	R/W-0	R/W-0

23–18	17	16
Reserved	CONT	LOOP BACK
R-0	R/W-0	R/W-0

15–10	9	8
Reserved	POWERDOWN	SLEEP
R-0	R/WP-0	R/W-0

7	6	5	4	3	2	1	0
SWnRST	Reserved	CLOCK	STOP	PARITY	PARITY ENA	TIMING MODE	COMM MODE
R/W-0	R-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privileged mode only; -*n* = value after reset

NOTE: The SCIGCR1 Control Register Bits should not be changed during Frame Transmission or Reception.

Table 30-5. SCI Global Control Register 1 (SCIGCR1) Field Descriptions

Bit	Field	Value	Description
31-26	Reserved	0	Reads return 0. Writes have no effect.
25	TXENA		Transmit enable. Data is transferred from SCITD to the SCITXSHF shift out register only when the TXENA bit is set.
		0	Disable transfers from SCITD to SCITXSHF.
		1	Enable SCI to transfer data from SCITD to SCITXSHF.
			Note: Data written to SCITD or the transmit multi-buffer before TXENA is set is not transmitted. If TXENA is cleared while transmission is ongoing, the data previously written to SCITD is sent.
24	RXENA		Receive enable. RXENA allows or prevents the transfer of data from SCIRXSHF to SCIRD.
		0	The receiver will not transfer data from the shift buffer to the receive buffer.
		1	The receiver will transfer data from the shift buffer to the receive buffer.
			Note: Clearing RXENA stops received characters from being transferred into the receive buffer or multi-buffers, prevents the RX status flags from being updated by receive data, and inhibits both receive and error interrupts. However, the shift register continues to assemble data regardless of the state of RXENA. **Note: If RXENA is cleared before a frame is completely received, the data from the frame is not transferred into the receive buffer.** **Note: If RXENA is set before a frame is completely received, the data from the frame is transferred into the receive buffer. If RXENA is set while SCIRXSHF is in the process of assembling a frame, the status flags are not assured to be accurate for that frame. To ensure that the status flags correctly reflect what was detected on the bus during a particular frame, RXENA should be set before the detection of that frame.**
23-18	Reserved	0	Reads return 0. Writes have no effect.
17	CONT		Continue on suspend. This bit has an effect only when a program is being debugged with an emulator, and it determines how the SCI operates when the program is suspended. The
		0	When debug mode is entered, the SCI state machine is frozen. Transmissions are halted and resume when debug mode is exited.
		1	When debug mode is entered, the SCI continues to operate until the current transmit and receive functions are complete.

Table 30-5. SCI Global Control Register 1 (SCIGCR1) Field Descriptions (continued)

Bit	Field	Value	Description
16	LOOP BACK		Loopback bit. The self-checking option for the SCI can be selected with this bit. If the SCITX and SCIRX pins are configured with SCI functionality, then the SCITX pin is internally connected to the SCIRX pin. Externally, during loop back operation, the SCITX pin outputs a high value and the SCIRX pin is in a high-impedance state. If this bit value is changed while the SCI is transmitting or receiving data, errors may result.
		0	Loop back mode is disabled.
		1	Loop back mode is enabled.
15-10	Reserved	0	Reads return 0. Writes have no effect.
9	POWERDOWN		Power down. When the POWERDOWN bit is set, the SCI attempts to enter local low-power mode. If the POWERDOWN bit is set while the receiver is actively receiving data and the wake-up interrupt is enabled, then the SCI immediately asserts an error interrupt to prevent low-power mode from being entered. Only Privilege mode writes allowed.
		0	Normal operation.
		1	Low-power mode is enabled.
8	SLEEP		SCI sleep. In a multiprocessor configuration, this bit controls the receive sleep function. Clearing this bit brings the SCI out of sleep mode.
		0	Sleep mode is disabled.
		1	Sleep mode is enabled.
			Note: The receiver still operates when the SLEEP bit is set; however, RXRDY is updated and SCIRD is loaded with new data only when an address frame is detected. The remaining receiver status flags are updated and an error interrupt is requested if the corresponding interrupt enable bit is set, regardless of the value of the SLEEP bit. In this way, if an error is detected on the receive data line while the SCI is asleep, software can promptly deal with the error condition.
			Note: The SLEEP bit is not automatically cleared when an address byte is detected.
			See Section 30.6.1 for more information on using the SLEEP bit for multiprocessor communication.
7	SWnRST		Software reset (active low). This bit is effective in LIN and SCI modes.
		0	The SCI is in its reset state; no data will be transmitted or received. Writing a 0 to this bit initializes the SCI state machines and operating flags as defined in Table 30-11 and Table 30-12. All affected logic is held in the reset state until a 1 is written to this bit.
		1	The SCI is in its ready state; transmission and reception can be done. After this bit is set to 1, the configuration of the module should not change.
			Note: The SCI should only be configured while SWnRST = 0.
6	Reserved	0	Reads return 0. Writes have no effect.
5	CLOCK		SCI internal clock enable. The CLOCK bit determines the source of the module clock on the SCICLK pin.
		0	The external SCICLK is the clock source.
		1	The internal SCICLK is the clock source.
			Note: If an external clock is selected, then the internal baud rate generator and baud rate registers are bypassed. The maximum frequency allowed for an externally sourced SCI clock is VCLK/16.
4	STOP		SCI number of stop bits per frame.
		0	One stop bit is used.
		1	Two stop bits are used.
			Note: The receiver checks for only one stop bit. However in idle-line mode, the receiver waits until the end of the second stop bit (if STOP = 1) to begin checking for an idle period.

Table 30-5. SCI Global Control Register 1 (SCIGCR1) Field Descriptions (continued)

Bit	Field	Value	Description
3	PARITY		SCI parity odd/even selection. If the PARITY ENA bit is set, PARITY designates odd or even parity.
		0	Odd parity is used.
		1	Even parity is used.
			The parity bit is calculated based on the data bits in each frame and the address bit (in address-bit mode). The start and stop fields in the frame are not included in the parity calculation.
			For odd parity, the SCI transmits and expects to receive a value in the parity bit that makes odd the total number of bits in the frame with the value of 1.
			For even parity, the SCI transmits and expects to receive a value in the parity bit that makes even the total number of bits in the frame with the value of 1.
2	PARITY ENA		Parity enable. This bit enables or disables the parity function.
		0	Parity is disabled; no parity bit is generated during transmission or is expected during reception.
		1	Parity is enabled. A parity bit is generated during transmission and is expected during reception.
1	TIMING MODE		SCI timing mode bit.
		0	Synchronous timing is used.
		1	Asynchronous timing is used.
0	COMM MODE		SCI communication mode bit.
		0	Idle-line mode is used.
		1	Address-bit mode is used.

30.7.3 SCI Set Interrupt Register (SCISETINT)

Figure 30-10 and Table 30-6 illustrate this register. SCISETINT register is used to enable the required interrupts supported by the module.

Figure 30-10. SCI Set Interrupt Register (SCISETINT) [offset = 0Ch]

31-27	26	25	24
Reserved	SET FE INT	SET OE INT	SET PE INT
R-0	R/W-0	R/W-0	R/W-0

23-19	18	17	16
Reserved	SET RX DMA ALL	SET RX DMA	SET TX DMA
R-0	R/W-0	R/W-0	R/W-0

15-10	9	8
Reserved	SET RX INT	SET TX INT
R-0	R/W-0	R/W-0

7-2	1	0
Reserved	SET WAKEUP INT	SET BRKDT INT
R-0	R/W-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 30-6. SCI Set Interrupt Register (SCISETINT) Field Descriptions

Bit	Field	Value	Description
31-27	Reserved	0	Reads return 0. Writes have no effect.
26	SET FE INT		Set framing-error interrupt. Setting this bit enables the SCI module to generate an interrupt when a framing error occurs.
		0	*Read:* The interrupt is disabled. *Write:* No effect.
		1	*Read or write:* The interrupt is enabled.
25	SET OE INT		Set overrun-error interrupt. Setting this bit enables the SCI module to generate an interrupt when an overrun error occurs.
		0	*Read:* The interrupt is disabled. *Write:* No effect.
		1	*Read or write:* The interrupt is enabled.
24	SET PE INT		Set parity interrupt. Setting this bit enables the SCI module to generate an interrupt when a parity error occurs.
		0	*Read:* The interrupt is disabled. *Write:* No effect.
		1	*Read or write:* The interrupt is enabled.
23-19	Reserved	0	Reads return 0. Writes have no effect.
18	SET RX DMA ALL		Set receive DMA all. This bit determines if a separate interrupt is generated for the address frames sent in multiprocessor communications. When this bit is 0, RX interrupt requests are generated for address frames and DMA requests are generated for data frames. When this bit is 1, RX DMA requests are generated for both address and data frames.
		0	*Read:* The DMA request is disabled for address frames (the receive interrupt request is enabled for address frames). *Write:* No effect.
		1	*Read or write:* The DMA request is enabled for address and data frames
17	SET RX DMA		Set receiver DMA. To enable receiver DMA requests, this bit must be set. If it is cleared, interrupt requests are generated depending on bit SCISETINT.
		0	*Read:* The DMA request is disabled. *Write:* No effect.
		1	*Read or write:* The DMA request is enabled for address and data frames.

Table 30-6. SCI Set Interrupt Register (SCISETINT) Field Descriptions (continued)

Bit	Field	Value	Description
16	SET TX DMA		Set transmit DMA. To enable DMA requests for the transmitter, this bit must be set. If it is cleared, interrupt requests are generated depending on SET TX INT bit (SCISETINT).
		0	*Read:* Transmit DMA request is disabled. *Write:* No effect.
		1	*Read or write:* Transmit DMA request is enabled.
15-10	Reserved	0	Reads return 0. Writes have no effect.
9	SET RX INT		Receiver interrupt enable. Setting this bit enables the SCI to generate a receive interrupt after a frame has been completely received and the data is being transferred from SCIRXSHF to SCIRD.
		0	*Read:* The interrupt is disabled. *Write:* No effect.
		1	*Read or write:* The interrupt is enabled.
8	SET TX INT		Set transmitter interrupt. Setting this bit enables the SCI to generate a transmit interrupt as data is being transferred from SCITD to SCITXSHF and the TXRDY bit is being set.
		0	*Read:* The interrupt is disabled. *Write:* No effect.
		1	*Read or write:* The interrupt is enabled.
7-2	Reserved	0	Reads return 0. Writes have no effect.
1	SET WAKEUP INT		Set wakeup interrupt. Setting this bit enables the SCI to generate a wakeup interrupt and thereby exit low-power mode. If enabled, the wakeup interrupt is asserted when local low-power mode is requested while the receiver is busy or if a low level is detected on the SCIRX pin during low-power mode.
		0	*Read:* The interrupt is disabled. *Write:* No effect.
		1	*Read or write:* The interrupt is enabled.
0	SET BRKDT INT		Set breakdetect interrupt. Setting this bit enables the SCI to generate an error interrupt if a break condition is detected on the SCIRX pin.
		0	*Read:* The interrupt is disabled. *Write:* No effect.
		1	*Read or write:* The interrupt is enabled.

30.7.4 SCI Clear Interrupt Register (SCICLEARINT)

Figure 30-11 and Table 30-7 illustrate this register. SCICLEARINT register is used to clear the selected enabled interrupts with out accessing SCISETINT register.

Figure 30-11. SCI Clear Interrupt Register (SCICLEARINT) [offset = 10h]

31	27	26	25	24
Reserved		CLR FE INT	CLR OE INT	CLR PE INT
R-0		R/W-0	R/W-0	R/W-0

23	19	18	17	16
Reserved		CLR RX DMA ALL	CLR RX DMA	CLR TX DMA
R-0		R/W-0	R/W-0	R/W-0

15	10	9	8
Reserved		CLR RX INT	CLR TX INT
R-0		R/W-0	R/W-0

7	2	1	0
Reserved		CLR WAKEUP INT	CLR BRKDT INT
R-0		R/W-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 30-7. SCI Clear Interrupt Register (SCICLEARINT) Field Descriptions

Bit	Field	Value	Description
31-27	Reserved	0	Reads return 0. Writes have no effect.
26	CLR FE INT		Clear framing-error interrupt. This bit disables the framing-error interrupt when set.
		0	*Read:* The interrupt is disabled. *Write:* No effect.
		1	*Read:* The interrupt is enabled. *Write:* The interrupt is disabled.
25	CLR CE INT		Clear overrun-error interrupt. This bit disables the SCI overrun error interrupt when set.
		0	*Read:* The interrupt is disabled. *Write:* No effect.
		1	*Read:* The interrupt is enabled. *Write:* The interrupt is disabled.
24	CLR PE INT		Clear parity interrupt. This bit disables the parity error interrupt when set.
		0	*Read:* The interrupt is disabled. *Write:* No effect.
		1	*Read:* The interrupt is enabled. *Write:* The interrupt is disabled.
23-19	Reserved	0	Reads return 0. Writes have no effect.
18	CLR RX DMA ALL		Clear receive DMA all. This bit clears the receive DMA request for address frames when set. Only receive data frames generate a DMA request.
		0	*Read:* Receive DMA request for address frames is disabled; Instead, RX interrupt requests are enabled for address frames. Receive DMA requests are still enabled for data frames. *Write:* No effect.
		1	*Read:* The receive DMA request for address and data frames is enabled. *Write:* The receive DMA request for address and data frames is disabled.

Table 30-7. SCI Clear Interrupt Register (SCICLEARINT) Field Descriptions (continued)

Bit	Field	Value	Description
17	CLR RX DMA		Clear receive DMA request. This bit disables the receive DMA request when set.
		0	*Read:* The DMA request is disabled. *Write:* No effect.
		1	*Read:* The receive DMA request is enabled. *Write:* The receive DMA request for is disabled.
16	CLR TX DMA		Clear transmit DMA request. This bit disables the transmit DMA request when set.
		0	*Read:* Transmit DMA request is disabled. *Write:* No effect.
		1	*Read:* The transmit DMA request is enabled. *Write:* The transmit DMA request for is disabled.
15-10	Reserved	0	Reads return 0. Writes have no effect.
9	CLR RX INT		Clear receiver interrupt. This bit disables the receiver interrupt when set.
		0	*Read:* The interrupt is disabled. *Write:* No effect.
		1	*Read:* The interrupt is enabled. *Write:* The interrupt is disabled.
8	CLR TX INT		Clear transmitter interrupt. This bit disables the transmitter interrupt when set.
		0	*Read:* The interrupt is disabled. *Write:* No effect.
		1	*Read:* The interrupt is enabled. *Write:* The interrupt is disabled.
7-2	Reserved	0	Reads return 0. Writes have no effect.
1	CLR WAKEUP INT		Clear wakeup interrupt. This bit disables the wakeup interrupt when set.
		0	*Read:* The interrupt is disabled. *Write:* No effect.
		1	*Read:* The interrupt is enabled. *Write:* The interrupt is disabled.
0	CLR BRKDT INT		Clear breakdetect interrupt. This bit disables the break-detect interrupt when set.
		0	*Read:* The interrupt is disabled. *Write:* No effect.
		1	*Read:* The interrupt is enabled. *Write:* The interrupt is disabled.

30.7.5 SCI Set Interrupt Level Register (SCISETINTLVL)

Figure 30-12 and Table 30-8 illustrate this register. This register is used to set the interrupt level for the supported interrupts.

Figure 30-12. SCI Set Interrupt Level Register (SCISETINTLVL) [offset = 14h]

31 27	26	25	24
Reserved	SET FE INT LVL	SET OE INT LVL	SET PE INT LVL
R-0	R/W-0	R/W-0	R/W-0

23 19	18	17 16
Reserved	SET RX DMA ALL INT LVL	Reserved
R-0	R/W-0	R-0

15 10	9	8
Reserved	SET RX INT LVL	SET TX INT LVL
R-0	R/W-0	R/W-0

7 2	1	0
Reserved	SET WAKEUP INT LVL	SET BRKDT INT LVL
R-0	R/W-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 30-8. SCI Set Interrupt Level Register (SCISETINTLVL) Field Descriptions

Bit	Field	Value	Description
31-27	Reserved	0	Reads return 0. Writes have no effect.
26	SET FE INT LVL		Set framing-error interrupt level.
		0	*Read:* The interrupt level is mapped to the INT0 line. *Write:* No effect.
		1	*Read or write:* The interrupt level is mapped to the INT1 line.
25	SET CE INT LVL		Set overrun-error interrupt level.
		0	*Read:* The interrupt level is mapped to the INT0 line. *Write:* No effect.
		1	*Read or write:* The interrupt level is mapped to the INT1 line.
24	SET PE INT LVL		Set parity error interrupt level.
		0	*Read:* The interrupt level is mapped to the INT0 line. *Write:* No effect.
		1	*Read or write:* The interrupt level is mapped to the INT1 line.
23-19	Reserved	0	Reads return 0. Writes have no effect.
18	SET RX DMA ALL LVL		Set receive DMA all interrupt levels.
		0	*Read:* The receive interrupt request for address frames is mapped to the INT0 line. *Write:* No effect.
		1	*Read or write:* The receive interrupt request for address frames is mapped to the INT1 line.
17-10	Reserved	0	Reads return 0. Writes have no effect.
9	SET RX INT LVL		Set receiver interrupt level.
		0	*Read:* The interrupt level is mapped to the INT0 line. *Write:* No effect.
		1	*Read or write:* The interrupt level is mapped to the INT1 line.

Table 30-8. SCI Set Interrupt Level Register (SCISETINTLVL) Field Descriptions (continued)

Bit	Field	Value	Description
8	SET TX INT LVL		Set transmitter interrupt level.
		0	*Read:* The interrupt level is mapped to the INT0 line. *Write:* No effect.
		1	*Read or write:* The interrupt level is mapped to the INT1 line.
7-2	Reserved	0	Reads return 0. Writes have no effect.
1	SET WAKEUP INT LVL		Set wakeup interrupt level.
		0	*Read:* The interrupt level is mapped to the INT0 line. *Write:* No effect.
		1	*Read or write:* The interrupt level is mapped to the INT1 line.
0	SET BRKDT INT LVL		Set breakdetect interrupt level.
		0	*Read:* The interrupt level is mapped to the INT0 line. *Write:* No effect.
		1	*Read or write:* The interrupt level is mapped to the INT1 line.

30.7.6 SCI Clear Interrupt Level Register (SCICLEARINTLVL)

Figure 30-13 and Table 30-9 illustrate this register.

Figure 30-13. SCI Clear Interrupt Level Register (SCICLEARINTLVL) [offset = 18h]

31-27	26	25	24
Reserved	CLR FE INT LVL	CLR OE INT LVL	CLR PE INT LVL
R-0	R/W-0	R/W-0	R/W-0

23-19	18	17-16
Reserved	CLR RX DMA ALL INT LVL	Reserved
R-0	R/W-0	R-0

15-10	9	8
Reserved	CLR RX INT LVL	CLR TX INT LVL
R-0	R/W-0	R/W-0

7-2	1	0
Reserved	CLR WAKEUP INT LVL	CLR BRKDT INT LVL
R-0	R/W-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 30-9. SCI Clear Interrupt Level Register (SCICLEARINTLVL) Field Descriptions

Bit	Field	Value	Description
31-27	Reserved	0	Reads return 0. Writes have no effect.
26	CLR FE INT LVL		Clear framing-error interrupt.
		0	*Read:* The interrupt level is mapped to the INT0 line. *Write:* No effect.
		1	*Read:* The interrupt level is mapped to the INT1 line. *Write:* The interrupt level is mapped to the INT0 line.

Table 30-9. SCI Clear Interrupt Level Register (SCICLEARINTLVL) Field Descriptions (continued)

Bit	Field	Value	Description
25	CLR CE INT LVL	0	Clear overrun-error interrupt. *Read:* The interrupt level is mapped to the INT0 line. *Write:* No effect.
		1	*Read:* The interrupt level is mapped to the INT1 line. *Write:* The interrupt level is mapped to the INT0 line.
24	CLR PE INT LVL	0	Clear parity interrupt. *Read:* The interrupt level is mapped to the INT0 line. *Write:* No effect.
		1	*Read:* The interrupt level is mapped to the INT1 line. *Write:* The interrupt level is mapped to the INT0 line.
23-19	Reserved	0	Reads return 0. Writes have no effect.
18	CLR RX DMA ALL LVL	0	Clear receive DMA interrupt level. *Read:* The receive interrupt request for address frames is mapped to the INT0 line. *Write:* No effect.
		1	*Read:* The receive interrupt request for address frames is mapped to the INT1 line. *Write:* The receive interrupt request for address frames is mapped to the INT0 line.
17-10	Reserved	0	Reads return 0. Writes have no effect.
9	CLR RX INT LVL	0	Clear receiver interrupt. *Read:* The interrupt level is mapped to the INT0 line. *Write:* No effect.
		1	*Read:* The interrupt level is mapped to the INT1 line. *Write:* The interrupt level is mapped to the INT0 line.
8	CLR TX INT LVL	0	Clear transmitter interrupt. *Read:* The interrupt level is mapped to the INT0 line. *Write:* No effect.
		1	*Read:* The interrupt level is mapped to the INT1 line. *Write:* The interrupt level is mapped to the INT0 line.
7-2	Reserved	0	Reads return 0. Writes have no effect.
1	CLR WAKEUP INT LVL	0	Clear wakeup interrupt. *Read:* The interrupt level is mapped to the INT0 line. *Write:* No effect.
		1	*Read:* The interrupt level is mapped to the INT1 line. *Write:* The interrupt level is mapped to the INT0 line.
0	CLR BRKDT INT LVL	0	Clear breakdetect interrupt. *Read:* The interrupt level is mapped to the INT0 line. *Write:* No effect.
		1	*Read:* The interrupt level is mapped to the INT1 line. *Write:* The interrupt level is mapped to the INT0 line.

30.7.7 SCI Flags Register (SCIFLR)

Figure 30-14 and Table 30-10 illustrate this register.

Figure 30-14. SCI Flags Register (SCIFLR) [offset = 1Ch]

31 27	26	25	24
Reserved	FE	OE	PE
R-0	R/W-0	R/W-0	R/W-0

23 16
Reserved
R-0

15 13	12	11	10	9	8
Reserved	RX WAKE	TX EMPTY	TX WAKE	RX RDY	TX RDY
R-0	R/W-0	R/W-1	R/W-0	R/W-0	R/W-1

7 4	3	2	1	0
Reserved	BUSY	IDLE	WAKE UP	BRKDT
R-0	R/W-0	R-0	R/WL-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 30-10. SCI Flags Register (SCIFLR) Field Descriptions

Bit	Field	Value	Description
31-27	Reserved		Reads return 0. Writes have no effect.
26	FE		Framing error flag. This bit is effective in LIN or SCI-compatible mode. This bit is set when an expected stop bit is not found. In SCI compatibility mode, only the first stop bit is checked. The missing stop bit indicates that synchronization with the start bit has been lost and that the character is incorrectly framed. Detection of a framing error causes the SCI/LIN to generate an error interrupt if the SET FE INT bit (SCISETINT[26]). The framing error flag is cleared by the following: • Setting of the SWnRST bit • Setting of the RESET bit • A system reset • Writing a 1 to this bit • Reading the corresponding interrupt offset in SCIINTVECT0/1 • Reception of a new character/frame, depending on whether the module is in SCI compatible or LIN mode In multi-buffer mode the frame is defined in the SCIFORMAT register.
		0	*Read:* No framing error has been detected since the last clear. *Write:* No effect.
		1	*Read:* A framing error has been detected since the last clear. *Write:* The bit is cleared to 0.
25	OE		Overrun error flag. This bit is set when the transfer of data from SCIRXSHF to SCIRD overwrites unread data already in SCIRD. Detection of an overrun error causes the LIN to generate an error interrupt if the SET OE INT bit (SCISETINT[25]) is set. The OE flag is reset by the following: • Setting of the SWnRST bit • Setting of the RESET bit • A system reset • Writing a 1 to this bit • Reading the corresponding interrupt offset in SCIINTVECT0/1
		0	*Read:* No overrun error has been detected since the last clear. *Write:* No effect.
		1	*Read:* An overrun error has been detected since the last clear. *Write:* The bit is cleared to 0.

Table 30-10. SCI Flags Register (SCIFLR) Field Descriptions (continued)

Bit	Field	Value	Description
24	PE		Parity error flag. This bit is set when a parity error is detected in the received data. In SCI address-bit mode, the parity is calculated on the data and address bit fields of the received frame. In idle-line mode, only the data is used to calculate parity. An error is generated when a character is received with a mismatch between the number of 1s and its parity bit. If the parity function is disabled (SCIGCR[2] = 0), the PE flag is disabled and read as 0. Detection of a parity error causes the LIN to generate an error interrupt if the SET PE INT bit (SCISETINT[24]) is set. The PE bit is reset by the following: • Setting of the SWnRST bit • Setting of the RESET bit • A system reset • Writing a 1 to this bit • Reception of a new character or frame, depending on whether the module is in SCI compatible or LIN mode, respectively • Reading the corresponding interrupt offset in SCIINTVECT0/1
		0	*Read:* No parity error has been detected since the last clear. *Write:* No effect.
		1	*Read:* A parity error has been detected since the last clear. *Write:* The bit is cleared to 0.
23-13	Reserved	0	Reads return 0. Writes have no effect.
12	RXWAKE		Receiver wakeup detect flag. The SCI sets this bit to indicate that the data currently in SCIRD is an address. RXWAKE is cleared by the following: • Setting of the SWnRST bit • Setting of the RESET bit • A system reset • Upon receipt of a data frame
		0	The data in SCIRD is not an address.
		1	The data in SCIRD is an address.
11	TX EMPTY		Transmitter empty flag. This flag indicates the transmitter's buffer register(s) (SCITD/TDy) and shift register (SCITXSHF) are empty. **Note: The RESET bit, an active SWnRST (SCIGCR1[7]), or a system reset sets this bit. This bit does not cause an interrupt request.**
		0	Transmitter buffer or shift register (or both) are loaded with data.
		1	Transmitter buffer and shift registers are both empty.
10	TXWAKE		Transmitter wakeup method select. The TXWAKE bit controls whether the data in SCITD should be sent as an address or data frame using multiprocessor communication format. This bit is set to 1 or 0 by software before a byte is written to SCITD and is cleared by the SCI when data is transferred from SCITD to SCITXSHF or by a system reset. **Note: TXWAKE is not cleared by the SWnRST bit.** *Address-bit mode*
		0	Frame to be transmitted will be data (address bit = 0).
		1	Frame to be transmitted will be an address (address bit = 1).
			Idle-line mode
		0	The frame to be transmitted will be data.
		1	The following frame to be transmitted will be an address (writing a 1 to this bit followed by writing dummy data to the SCITD will result in a idle period of 11 bit periods before the next frame is transmitted).

Table 30-10. SCI Flags Register (SCIFLR) Field Descriptions (continued)

Bit	Field	Value	Description
9	RXRDY		Receiver ready flag. The receiver sets this bit to indicate that the SCIRD contains new data and is ready to be read by the CPU or DMA. The SCI generates a receive interrupt when RXRDY flag bit is set if the SET RX INT bit (SCISETINT[9]) is set. RXRDY is cleared by the following: • Setting of the SWnRST bit • Setting of the RESET bit • A system reset • Writing a 1 to this bit • Reading the SCIRD register in compatibility mode • Reading the last data byte RDy of the response in LIN mode **Note: The RXRDY flag cannot be cleared by reading the corresponding interrupt offset in the SCIINTVECT0/1 register.**
		0	*Read:* No new data is in SCIRD. *Write:* No effect.
		1	*Read:* New data is ready to be read from SCIRD. *Write:* The bit is cleared to 0.
8	TXRDY		Transmitter buffer register ready flag. When set, this bit indicates that the transmit buffer is ready to get another character from a CPU or DMA write. Writing data to SCITD automatically clears this bit. This bit is set after the data of the TX buffer is shifted into the SCITXSHF register. This event can trigger a transmit interrupt after data is copied to the TX shift register SCITXSHF, if the interrupt enable bit TXINT is set. **Note: 1) TXRDY is also set to 1 by setting of the RESET bit, enabling SWnRST, or by a system reset.** **2) The TXRDY flag cannot be cleared by reading the corresponding interrupt offset in the SCIINTVECT0/1 register.** **3) The transmit interrupt request can be eliminated until the next series of data written into the transmit buffers LINTD0 and LINTD1, by disabling the corresponding interrupt via the SCICLEARINT register or by disabling the transmitter via the TXENA bit (SCIGCR1[25]).**
		0	SCITD is full.
		1	SCITD is ready to receive the next character.
7-4	Reserved	0	Reads return 0. Writes have no effect.
3	BUSY		Bus busy flag. TThis bit indicates whether the receiver is in the process of receiving a frame. As soon as the receiver detects the beginning of a start bit, the BUSY bit is set to 1. When the reception of a frame is complete, the SCI clears the BUSY bit. If SET WAKEUP INT bit (SCISETINT[2]) is set and power down is requested while this bit is set, the SCI automatically prevents low-power mode from being entered and generates wakeup interrupt. The BUSY bit is controlled directly by the SCI receiver, but this bit can also be cleared by the following: • Setting the SWnRST bit • Setting of the RESET bit • A system reset occurring
		0	The receiver is not currently receiving a frame.
		1	The receiver is currently receiving a frame.
2	IDLE		SCI receiver in idle state. While this bit is set, the SCI looks for an idle period to resynchronize itself with the bit stream. The receiver does not receive any data while the bit is set. The bus must be idle for 11 bit periods to clear this bit. The SCI enters the idle state if one of the following events occurs: • A system reset • An SCI software reset • A power down • The RX pin is configured as a general I/O pin
		0	The idle period has been detected; the SCI is ready to receive.
		1	The idle period has not been detected; the SCI will not receive any data.

Table 30-10. SCI Flags Register (SCIFLR) Field Descriptions (continued)

Bit	Field	Value	Description
1	WAKEUP		Wakeup flag. This bit is set by the SCI when receiver or transmitter activity has taken the module out of power-down mode. An interrupt is generated if the SET WAKEUP INT bit (SCISETINT[2]) is set. It is cleared by the following: • Setting of the SWnRST bit • Setting of the RESET bit • A system reset • Writing a 1 to this bit • Reading the corresponding interrupt offset in SCIINTVECT0/1 For compatibility mode, see the SCI document for more information on low-power mode.
		0	*Read:* The module will not wake up from power-down mode. *Write:* No effect.
		1	*Read:* Wake up from power-down mode. *Write:* The bit is cleared to 0.
0	BRKDT		SCI break-detect flag. This bit is set when the SCI detects a break condition on the LINRX pin. A break condition occurs when the SCIRX pin remains continuously low for at least 10 bits after a missing first stop bit, that is, after a framing error. Detection of a break condition causes the SCI to generate an error interrupt if the SET BRKDT INT bit (SCISETINT[0]) is set. The BRKDT bit is reset by the following: • Setting of the SWnRST bit • Setting of the RESET bit • A system reset • Writing a 1 to this bit • Reading the corresponding interrupt offset in SCIINTVECT0/1
		0	*Read:* No break condition has been detected since the last clear. *Write:* No effect.
		1	*Read:* A break condition has been detected. *Write:* The bit is cleared to 0.

Table 30-11. SCI Receiver Status Flags

SCI Flag	Register	Bit	Value After Reset [1]
FE	SCIFLR	26	0
OE	SCIFLR	25	0
PE	SCIFLR	24	0
RXWAKE	SCIFLR	12	0
RXRDY	SCIFLR	9	0
BRKDT	SCIFLR	0	0

[1] The flags are frozen with their reset value while SWnRST = 0.

Table 30-12. SCI Transmitter Status Flags

SCI Flag	Register	Bit	Value After Reset [1]
TX EMPTY	SCIFLR	11	1
TXRDY	SCIFLR	8	1

[1] The flags are frozen with their reset value while SWnRST = 0.

30.7.8 SCI Interrupt Vector Offset 0 (SCIINTVECT0)

Figure 30-15 and Table 30-13 illustrate this register.

Figure 30-15. SCI Interrupt Vector Offset 0 (SCIINTVECT0) [offset = 20h]

31–16
Reserved
R-0

15–4	3–0
Reserved	INTVECT0
R-0	R-0

LEGEND: R = Read only; -*n* = value after reset

Table 30-13. SCI Interrupt Vector Offset 0 (SCIINTVECT0) Field Descriptions

Bit	Field	Value	Description
31-4	Reserved	0	Reads return 0. Writes have no effect.
3-0	INVECT0	0-Fh	Interrupt vector offset for INT0. This register indicates the offset for interrupt line INT0. A read to this register updates its value to the next highest priority pending interrupt in SCIFLR and clears the flag in SCIFLR corresponding to the offset that was read. See Table 30-1 for a list of the interrupts. **Note: The flags for the receive (SCIFLR[9]) and the transmit (SCIFLR[8]) interrupt cannot be cleared by reading the corresponding offset vector in this register (see detailed description in SCIFLR register).**

30.7.9 SCI Interrupt Vector Offset 1 (SCIINTVECT1)

Figure 30-16 and Table 30-14 illustrate this register.

Figure 30-16. SCI Interrupt Vector Offset 1 (SCIINTVECT1) [offset = 24h]

31–16
Reserved
R-0

15–4	3–0
Reserved	INTVECT1
R-0	R-0

LEGEND: R = Read only; -*n* = value after reset

Table 30-14. SCI Interrupt Vector Offset 1 (SCIINTVECT1) Field Descriptions

Bit	Field	Value	Description
31-4	Reserved	0	Reads return 0. Writes have no effect.
3-0	INVECT1	0-Fh	Interrupt vector offset for INT1. This register indicates the offset for interrupt line INT1. A read to this register updates its value to the next highest priority pending interrupt in SCIFLR and clears the flag in SCIFLR corresponding to the offset that was read. See Table 30-1 for list of interrupts. **Note: The flags for the receive (SCIFLR[9]) and the transmit (SCIFLR[8]) interrupt cannot be cleared by reading the corresponding offset vector in this register (see detailed description in SCIFLR register).**

30.7.10 SCI Format Control Register (SCIFORMAT)

Figure 30-17 and Table 30-15 illustrate this register.

Figure 30-17. SCI Format Control Register (SCIFORMAT) [offset = 28h]

31	16
Reserved	
R-0	

15	3	2	0
Reserved		CHAR	
R-0		R/W-0	

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 30-15. SCI Format Control Register (SCIFORMAT) Field Descriptions

Bit	Field	Value	Description
31-3	Reserved	0	Reads return 0. Writes have no effect.
2-0	CHAR		Character length control bits. These bits set the SCI character length from 1 to 8 bits. **When data of fewer than eight bits in length is received, it is left-justified in SCIRD and padded with trailing zeros.** **Data read from the SCIRD should be shifted by software to make the received data right-justified.** **Data written to the SCITD should be right-justified but does not need to be padded with leading zeros.**
		0	The character is 1 bit long.
		1h	The character is 2 bits long.
		2h	The character is 3 bits long.
		3h	The character is 4 bits long.
		4h	The character is 5 bits long.
		5h	The character is 6 bits long.
		6h	The character is 7 bits long.
		7h	The character is 8 bits long.

30.7.11 Baud Rate Selection Register (BRS)

This section describes the baud rate selection register. Figure 30-18 and Table 30-16 illustrate this register.

Figure 30-18. Baud Rate Selection Register (BRS) [offset = 2Ch]

31 – 24	23 – 16
Reserved	BAUD
R-0	R/W-0

15 – 0
BAUD
R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 30-16. Baud Rate Selection Register (BRS) Field Descriptions

Bit	Field	Value	Description
31-24	Reserved	0	Reads return 0. Writes have no effect.
23-0	BAUD	0-FF FFFFh	SCI 24-bit baud selection. The SCI has an internally generated serial clock determined by the VCLK and the prescalers BAUD in this register. The SCI uses the 24-bit integer prescaler BAUD value of this register to select one of over 16,700,000. The baud rate can be calculated using the following formulas: $\textit{Asynchronous baud value} = \left(\frac{\text{VCLK Frequency}}{16(\text{Baud} + 1)}\right)$ (59) $\textit{Isosynchronous baud value} = \left(\frac{\text{VCLK Frequency}}{\text{Baud} + 1}\right)$ (60) For BAUD = 0, $\textit{Asynchronous baud value} = \left(\frac{\text{VCLK Frequency}}{32}\right)$ (61) $\textit{Isosynchronous baud value} = \left(\frac{\text{VCLK Frequency}}{2}\right)$ (62) Table 30-17 contains comparative baud values for different P values, with VCLK = 50 MHz, for asynchronous mode..

Table 30-17. Comparative Baud Values for Different P Values, Asynchronous Mode [(1)][(2)]

24-Bit Register Value		Baud Selected		Percent Error
Decimal	Hex	Ideal	Actual	
26	00001A	115200	115740	0.47
53	000035	57600	57870	0.47
80	000050	38400	38580	0.47
162	0000A2	19200	19172	-0.15
299	00012B	10400	10417	0.16
325	000145	9600	9586	-0.15
399	00018F	7812.5	7812.5	0.00
650	00028A	4800	4800	0.00
15624	003BA0	200	200	0.00
624999	098967	5	5	0.00

[(1)] VCLK = 50 MHz

[(2)] Values are in decimal except for column 2.

30.7.12 SCI Data Buffers (SCIED, SCIRD, SCITD)

The SCI has three addressable registers in which transmit and receive data is stored.

30.7.12.1 Receiver Emulation Data Buffer (SCIED)

The SCIED register is addressed at a location different from SCIRD, but is physically the same register. Figure 30-19 and Table 30-18 illustrate this register.

Figure 30-19. Receiver Emulation Data Buffer (SCIED) [offset = 30h]

31-16
Reserved
R-0

15-8	7-0
Reserved	ED
R-0	R-0

LEGEND: R = Read only; -*n* = value after reset

Table 30-18. Receiver Emulation Data Buffer (SCIED) Field Descriptions

Bit	Field	Value	Description
31-8	Reserved	0	Reads return 0. Writes have no effect.
7-0	ED	0-FFh	Emulator data. Reading SCIED[7:0] does not clear the RXRDY flag (SCIFLR[9]), unlike reading SCIRD. This register should be used only by an emulator that must continually read the data buffer without affecting the RXRDY flag.

30.7.12.2 Receiver Data Buffer (SCIRD)

This register provides a location for the receiver data. Figure 30-20 and Table 30-19 illustrate this register.

Figure 30-20. Receiver Data Buffer (SCIRD) [offset = 34h]

31-16
Reserved
R-0

15-8	7-0
Reserved	RD
R-0	R-0

LEGEND: R = Read only; -*n* = value after reset

Table 30-19. Receiver Data Buffer (SCIRD) Field Descriptions

Bit	Field	Value	Description
31-8	Reserved	0	Reads return 0. Writes have no effect.
7-0	RD	0-FFh	Receiver data. When a frame has been completely received, the data in the frame is transferred from the receiver shift register SCIRXSHF to this register. As this transfer occurs, the RXRDY flag (SCIFLR[9]) is set and a receive interrupt is generated if SET RX INT bit (SCISETINT[9]) is set. **Note: When the data is read from SCIRD, the RXRDY flag (SCIFLR[9]) is automatically cleared.**

NOTE: When the SCI receives data that is fewer than eight bits in length, it loads the data into this register in a left-justified format padded with trailing zeros. Therefore, the user software should perform a logical shift on the data by the correct number of positions to make it right justified.

30.7.12.3 Transmit Data Buffer Register (SCITD)

Data to be transmitted is written to the SCITD register. Figure 30-21 and Table 30-20 illustrate this register.

Figure 30-21. Transmit Data Buffer Register (SCITD) [offset = 38h]

31–16
Reserved
R-0

15–8	7–0
Reserved	TD
R-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 30-20. Transmit Data Buffer Register (SCITD) Field Descriptions

Bit	Field	Value	Description
31-8	Reserved	0	Reads return 0. Writes have no effect.
7-0	TD	0-FFh	Transmit data. Data to be transmitted is written to the SCITD register. The transfer of data from this register to the transmit shift register SCITXSHF sets the TXRDY flag (SCIFLR[8]), which indicates that SCITD is ready to be loaded with another byte of data. **Note: If SET TX INT bit (SCISETINT[8] is set, this data transfer also causes an interrupt.**

NOTE: Data written to the SCITD register that is fewer than eight bits long must be right-justified, but it does not need to be padded with leading zeros.

30.7.13 SCI Pin I/O Control Register 0 (SCIPIO0)

Figure 30-22 and Table 30-21 illustrate this register.

Figure 30-22. SCI Pin I/O Control Register 0 (SCIPIO0) [offset = 3Ch]

31–8
Reserved
R-0

7–3	2	1	0
Reserved	TX FUNC	RX FUNC	Reserved
R-0	R/W-0	R/W-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 30-21. SCI Pin I/O Control Register 0 (SCIPIO0) Field Descriptions

Bit	Field	Value	Description
31-3	Reserved	0	Reads return 0. Writes have no effect.
2	TX FUNC		Transfer function. This bit defines the function of pin SCITX.
		0	SCITX is a general-purpose digital I/O pin.
		1	SCITX is the SCI transmit pin.
1	RX FUNC		Receive function. This bit defines the function of pin SCIRX.
		0	SCIRX is a general-purpose digital I/O pin.
		1	SCIRX is the SCI receive pin.
0	Reserved	0	Writes have no effect.

30.7.14 SCI Pin I/O Control Register 1 (SCIPIO1)

Figure 30-23 and Table 30-22 illustrate this register.

Figure 30-23. SCI Pin I/O Control Register 1 (SCIPIO1) [offset = 40h]

31 - 8
Reserved
R-0

7 - 3	2	1	0
Reserved	TX DIR	RX DIR	Reserved
R-0	R/W-0	R/W-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 30-22. SCI Pin I/O Control Register 1 (SCIPIO1) Field Descriptions

Bit	Field	Value	Description
31-3	Reserved	0	Reads return 0. Writes have no effect.
2	TX DIR		Transmit pin direction. This bit determines the data direction on the SCITX pin if it is configured with general-purpose I/O functionality (TX FUNC = 0). See Table 30-23 for the SCITX pin control with this bit and others.
		0	SCITX is a general-purpose input pin.
		1	SCITX is a general-purpose output pin.
1	RX DIR		Receive pin direction. This bit determines the data direction on the SCIRX pin if it is configured with general-purpose I/O functionality (RX FUNC = 0). See Table 30-24 for the SCIRX pin control with this bit and others.
		0	SCIRX is a general-purpose input pin.
		1	SCIRX is a general-purpose output pin.
0	Reserved	0	Writes have no effect.

Table 30-23. SCITX Pin Control

Function	TX IN [(1)]	TX OUT	TX FUNC	TX DIR
SCITX	X	X	1	X
General-purpose input	X	X	0	0
General-purpose output, high	X	1	0	1
General-purpose output, low	X	0	0	1

[(1)] TX IN is a read-only bit. Its value always reflects the level of the SCITX pin.

Table 30-24. SCIRX Pin Control

Function	RX IN [(1)]	RX OUT	RX FUNC	RX DIR
SCIRX	X	X	1	X
General-purpose input	X	X	0	0
General-purpose output, high	X	1	0	1
General-purpose output, low	X	0	0	1

[(1)] RX IN is a read-only bit. Its value always reflects the level of the SCIRX pin.

30.7.15 SCI Pin I/O Control Register 2 (SCIPIO2)

Figure 30-24 and Table 30-25 illustrate this register.

Figure 30-24. SCI Pin I/O Control Register 2 (SCIPIO2) [offset = 44h]

31–8
Reserved
R-0

7–3	2	1	0
Reserved	TX IN	RX IN	Reserved
R-0	R-X	R-X	R-X

LEGEND: R = Read only; -*n* = value after reset; -*X* = value is indeterminate

Table 30-25. SCI Pin I/O Control Register 2 (SCIPIO2) Field Descriptions

Bit	Field	Value	Description
31-3	Reserved	0	Reads return 0. Writes have no effect.
2	TX IN		Transmit pin in. This bit contains the current value on the SCITX pin.
		0	The SCITX pin is at logic low (0).
		1	The SCITX pin is at logic high (1).
1	RX IN		Receive pin in. This bit contains the current value on the SCIRX pin.
		0	The SCIRX pin is at logic low (0).
		1	The SCIRX pin is at logic high (1).
0	Reserved	0	Writes have no effect.

30.7.16 SCI Pin I/O Control Register 3 (SCIPIO3)

Figure 30-25 and Table 30-26 illustrate this register.

Figure 30-25. SCI Pin I/O Control Register 3 (SCIPIO3) [offset = 48h]

31 ... 8
Reserved
R-0

7 ... 3	2	1	0
Reserved	TX OUT	RX OUT	Reserved
R-0	R/W-0	R/W-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 30-26. SCI Pin I/O Control Register 3 (SCIPIO3) Field Descriptions

Bit	Field	Value	Description
31-3	Reserved	0	Reads return 0. Writes have no effect.
2	TX OUT		Transmit pin out. This pin specifies the logic to be output on pin SCITX, if the following conditions are met: • TX FUNC = 0 (SCITX pin is a general-purpose I/O.) • TX DIR = 1 (SCITX pin is a general-purpose output.) See Table 30-23 for an explanation of this bit's effect in combination with other bits.
		0	The output on the SCITX is at logic low (0).
		1	The output on the SCITX pin is at logic high (1). (Output voltage is V_{OH} or higher if TXPDR = 0 and output is in high impedance state if TXPDR = 1.)
1	RX OUT		Receive pin out. This bit specifies the logic to be output on pin SCIRX, if the following conditions are met: • RX FUNC = 0 (SCIRX pin is a general-purpose I/O.) • RX DIR = 1 (SCIRX pin is a general-purpose output.) See Table 30-24 for an explanation of this bit's effect in combination with the other bits.
		0	The output on the SCIRX pin is at logic low (0).
		1	The output on the SCIRX pin is at logic high (1). (Output voltage is V_{OH} or higher if RXPDR = 0, and output is in high impedance state if RXPDR = 1.)
0	Reserved	0	Writes have no effect.

30.7.17 SCI Pin I/O Control Register 4 (SCIPIO4)

Figure 30-26 and Table 30-27 illustrate this register.

Figure 30-26. SCI Pin I/O Control Register 4 (SCIPIO4) [offset = 4Ch]

31–8
Reserved
R-0

7–3	2	1	0
Reserved	TX SET	RX SET	Reserved
R-0	R/W-0	R/W-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 30-27. SCI Pin I/O Control Register 4 (SCIPIO4) Field Descriptions

Bit	Field	Value	Description
31-3	Reserved	0	Reads return 0. Writes have no effect.
2	TX SET		Transmit pin set. This bit sets the logic to be output on pin SCITX, if the following conditions are met: • TX FUNC = 0 (SCITX pin is a general-purpose I/O.) • TX DIR = 1 (SCITX pin is a general-purpose output.) See Table 30-23 for an explanation of this bit's effect in combination with other bits.
		0	*Read:* The output on SCITX is at logic low (0). *Write:* No effect.
		1	*Read or write:* The output on SCITX is at logic high (1).
1	RX SET		Receive pin set. This bit sets the data to be output on pin SCIRX, if the following conditions are met: • RX FUNC = 0 (SCIRX pin is a general-purpose I/O.) • RX DIR = 1 (SCIRX pin is a general-purpose output.) See Table 30-24 for an explanation of this bit's effect in combination with the other bits.
		0	*Read:* The output on SCIRX is at logic low (0). *Write:* No effect.
		1	*Read or write:* The output on SCIRX is at logic high (1).
0	Reserved	0	Writes have no effect.

30.7.18 SCI Pin I/O Control Register 5 (SCIPIO5)

Figure 30-27 and Table 30-28 illustrate this register.

Figure 30-27. SCI Pin I/O Control Register 5 (SCIPIO5) [offset = 50h]

31 … 8
Reserved
R-0

7 … 3	2	1	0
Reserved	TX CLR	RX CLR	Reserved
R-0	R/W-0	R/W-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 30-28. SCI Pin I/O Control Register 5 (SCIPIO5) Field Descriptions

Bit	Field	Value	Description
31-3	Reserved	0	Reads return 0. Writes have no effect.
2	TX CLR		Transmit pin clear. This bit clears the logic to be output on pin SCITX, if the following conditions are met: • TX FUNC = 0 (SCITX pin is a general-purpose I/O.) • TX DIR = 1 (SCITX pin is a general-purpose output.)
		0	*Read:* The output on SCITX is at logic low (0). *Write:* No effect.
		1	*Read:* The output on SCITX is at logic high (1). *Write:* The output on SCITX is at logic low (0).
1	RX CLR		Receive pin clear. This bit clears the logic to be output on pin SCIRX, if the following conditions are met: • RX FUNC = 0 (SCIRX pin is a general-purpose I/O.) • RX DIR = 1 (SCIRX pin is a general-purpose output.)
		0	*Read:* The output on SCIRX is at logic low (0). *Write:* No effect.
		1	*Read:* The output on SCIRX is at logic high (1). *Write:* The output on SCIRX is at logic low (0).
0	Reserved	0	Writes have no effect.

30.7.19 SCI Pin I/O Control Register 6 (SCIPIO6)

Figure 30-28 and Table 30-29 illustrate this register.

Figure 30-28. SCI Pin I/O Control Register 6 (SCIPIO6) [offset = 54h]

31 – 8
Reserved
R-0

7 – 3	2	1	0
Reserved	TX PDR	RX PDR	Reserved
R-0	R/W-0	R/W-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 30-29. SCI Pin I/O Control Register 6 (SCIPIO6) Field Descriptions

Bit	Field	Value	Description
31-3	Reserved	0	Reads return 0. Writes have no effect.
2	TX PDR		Transmit pin open drain enable. This bit enables open-drain capability in the output pin SCITX, if the following conditions are met: • TX FUNC = 0 (SCITX pin is a general-purpose I/O.) • TX DIR = 1 (SCITX pin is a general-purpose output.)
		0	Open drain functionality is disabled; the output voltage is V_{OL} or lower if TXOUT = 0 and V_{OH} or higher if TXOUT = 1.
		1	Open drain functionality is enabled; the output voltage is V_{OL} or lower if TXOUT = 0 and high impedance if TXOUT = 1.
1	RX PDR		Receive pin open drain enable. This bit enables open-drain capability in the output pin SCIRX, if the following conditions are met: • RX FUNC = 0 (SCIRX pin is a general-purpose I/O.) • RX DIR = 1 (SCIRX pin is a general-purpose output.)
		0	Open drain functionality is disabled; the output voltage is V_{OL} or lower if RXOUT = 0 and V_{OH} or higher if RXOUT = 1.
		1	Open drain functionality is enabled; the output voltage is V_{OL} or lower if RXOUT = 0 and high impedance if RXOUT = 1.
0	Reserved	0	Writes have no effect.

30.7.20 SCI Pin I/O Control Register 7 (SCIPIO7)

Figure 30-29 and Table 30-30 illustrate this register.

Figure 30-29. SCI Pin I/O Control Register 7 (SCIPIO7) [offset = 58h]

31-8
Reserved
R-0

7-3	2	1	0
Reserved	TX PD	RX PD	Reserved
R-0	R/W-n	R/W-n	R/W-n

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset, Refer to the Terminal Functions in the device datasheet for default pin settings.

Table 30-30. SCI Pin I/O Control Register 7 (SCIPIO7) Field Descriptions

Bit	Field	Value	Description
31-3	Reserved	0	Reads return 0. Writes have no effect.
2	TX PD		Transmit pin pull control disable. This bit disables pull control capability on the input pin SCITX.
		0	Pull control on the SCITX pin is enabled.
		1	Pull control on the SCITX pin is disabled.
1	RX PD		Receive pin pull control disable. This bit disables pull control capability on the input pin SCIRX.
		0	Pull control on the SCIRX pin is enabled.
		1	Pull control on the SCIRX pin is disabled.
0	Reserved	0	Writes have no effect.

30.7.21 SCI Pin I/O Control Register 8 (SCIPIO8)

Figure 30-30 and Table 30-31 illustrate this register.

Figure 30-30. SCI Pin I/O Control Register 8 (SCIPIO8) [offset = 5Ch]

31-8
Reserved
R-0

7-3	2	1	0
Reserved	TX PSL	RX PSL	Reserved
R-0	R/W-n	R/W-n	R/W-n

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset, Refer to the Terminal Functions in the device datasheet for default pin settings.

Table 30-31. SCI Pin I/O Control Register 8 (SCIPIO8) Field Descriptions

Bit	Field	Value	Description
31-3	Reserved	0	Reads return 0. Writes have no effect.
2	TX PSL		TX pin pull select. This bit selects pull type in the input pin SCITX.
		0	The SCITX pin is a pull down.
		1	The SCITX pin is a pull up.
1	RX PSL		RX pin pull select. This bit selects pull type in the input pin SCIRX.
		0	The SCIRX pin is a pull down.
		1	The SCIRX pin is a pull up.
0	Reserved	0	Writes have no effect.

30.7.22 Input/Output Error Enable (IODFTCTRL) Register

Figure 30-31 and Table 30-32 illustrate this register. After the basic SCI module configuration, enable the required Error mode to be created followed by IODFT Key enable.

NOTE:

1. All the bits are used in IODFT mode only.
2. Each IODFT are expected to be checked individually.

Figure 30-31. Input/Output Error Enable Register (IODFTCTRL) [offset = 90h]

31				27	26	25	24
Reserved					FEN	PEN	BRKDTENA
R-0					R/W-0	R/W-0	R/W-0

23		21	20	19	18		16
Reserved			PIN SAMPLE MASK		TX SHIFT		
R-0			R/W-0		R/W-0		

15			12	11			8
Reserved				IODFTENA			
R-0				R/WP-0	R/WP-1	R/WP-0	R/WP-1

7					2	1	0
Reserved						LPB ENA	RXPENA
R-0						R/WP-0	R/WP-0

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privilege mode only; -*n* = value after reset

Table 30-32. Input/Output Error Enable Register (IODFTCTRL) Field Descriptions

Bit	Field	Value	Description
31-27	Reserved	0	Reads return 0. Writes have no effect.
26	FEN		Frame error enable. This bit is used to create a frame error.
		0	No error is created.
		1	The stop bit received is ANDed with 0 and passed to the stop bit check circuitry.
25	PEN		Parity error enable. This bit is used to create a parity error.
		0	No parity error occurs.
		1	The parity bit received is toggled so that a parity error occurs.
24	BRKD TENA		Break detect error enable. This bit is used to create a BRKDT error.
		0	No error is created.
		1	The stop bit of the frame is ANDed with 0 and passed to the RSM so that a frame error occurs. Then the RX pin is forced to continuous low for 10 T_{BITS} so that a BRKDT error occurs.
32-21	Reserved	0	Reads return 0. Writes have no effect.
20-19	PIN SAMPLE MASK		Pin sample mask. These bits define the sample number at which the TX pin value that is being transmitted will be inverted to verify the receive pin samples majority detection circuitry.
		0	No mask is used.
		1h	Invert the TX Pin value at 7th SCLK.
		2h	Invert the TX Pin value at 8th SCLK.
		3h	Invert the TX Pin value at 9th SCLK.

Table 30-32. Input/Output Error Enable Register (IODFTCTRL) Field Descriptions (continued)

Bit	Field	Value	Description
18-16	TX SHIFT		Transmit shift. These bits define the amount by which the value on TX pin is delayed so that the value on the RX pin is asynchronous. This feature is not applicable to the start bit.
		0	No delay occurs.
		1h	The value is delayed by 1 SCLK.
		2h	The value is delayed by 2 SCLK.
		3h	The value is delayed by 3 SCLK.
		4h	The value is delayed by 4 SCLK.
		5h	The value is delayed by 5 SCLK.
		6h	The value is delayed by 6 SCLK.
		7h	No delay occurs.
15-12	Reserved	0	Reads return 0. Writes have no effect.
11-8	IODFTENA		IODFT enable key. Write access permitted in Privilege mode only.
		Ah	IODFT is enabled.
		All Others	IODFT is disabled.
7-2	Reserved	0	Reads return 0. Writes have no effect.
1	LPBENA		Module loopback enable. Write access permitted in Privilege mode only. **Note: In analog loopback mode the complete communication path through the I/Os can be tested, whereas in digital loopback mode the I/O buffers are excluded from this path.**
		0	Digital loopback is enabled.
		1	Analog loopback is enabled in module I/O DFT mode when IODFTENA = 1010.
0	RXPENA		Module analog loopback through receive pin enable. Write access permitted in Privilege mode only. This bit defines whether the I/O buffers for the transmit or the receive pin are included in the communication path (in analog loopback mode).
		0	Analog loopback through the transmit pin is enabled.
		1	Analog loopback through the receive pin is enabled.

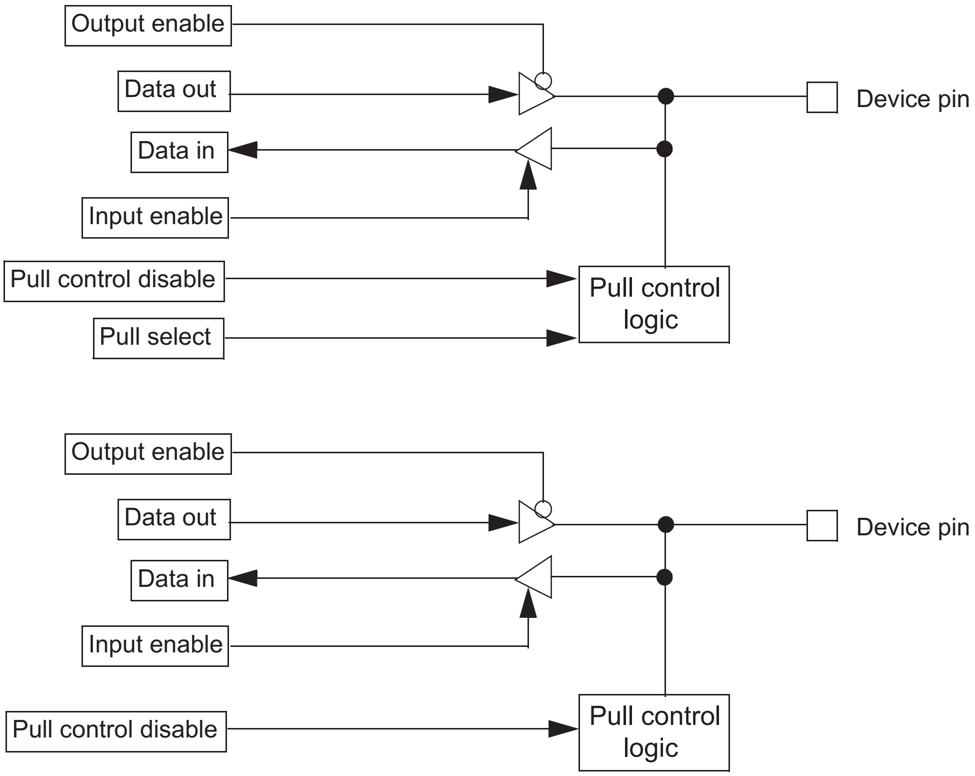
30.8 GPIO Functionality

The following sections apply to all device pins that can be configured as functional or general-purpose I/O pins.

30.8.1 GPIO Functionality

Figure 30-32 illustrates the GPIO functionality.

Figure 30-32. GPIO Functionality



30.8.2 Under Reset

The following apply if a device is under reset:

- Pull control. The reset pull control on the pins is enabled.
- Input buffer. The input buffer is enabled.
- Output buffer. The output buffer is disabled.

30.8.3 Out of Reset

The following apply if the device is out of reset:

- Pull control. The pull control is enabled by clearing the PD (pull control disable) bit in the SCIPIO7 register (Section 30.7.20). In this case, if the PSL (pull select) bit in the SCIPIO8 register (Section 30.7.21) is set, the pin will have a pull-up. If the PSL bit is cleared, the pin will have a pull-down. If the PD bit is set in the control register, there is no pull-up or pull-down on the pin.
- Input buffer. The input buffer is always enabled in functional mode.

NOTE: The pull-disable logic depends on the pin direction. It is independent of whether the device is in I/O or functional mode. If the pin is configured as output or transmit, then the pulls are disabled automatically. If the pin is configured as input or receive, the pulls are enabled or disabled depending on bit PD in the pull disable register SCIPIO7 (Section 30.7.20).

- Output buffer. A pin can be driven as an output pin if the TX DIR bit is set in the pin direction control register (SCIPIO1; Section 30.7.14) AND the open-drain feature is not enabled in the SCIPIO6 register (Section 30.7.19).

30.8.4 Open-Drain Feature Enabled on a Pin

The following apply if the open-drain feature is enabled on a pin:

- The output buffer is enabled if a low signal is being driven on to the pin.
- The output buffer is disabled (the direction control signal DIR is internally forced low) if a high signal is being driven on to the pin.

NOTE: The open-drain feature is available only in I/O mode (SCIPIO0; Section 30.7.13).

30.8.5 Summary

The behavior of the input buffer, output buffer, and the pull control is summarized in Table 30-33.

Table 30-33. Input Buffer, Output Buffer, and Pull Control Behavior as GPIO Pins

Device under Reset?	Pin Direction (DIR) [(1)][(2)]	Pull Disable (PULDIS) [(1)][(3)]	Pull Select (PULSEL) [(1)][(4)]	Pull Control	Output Buffer	Input Buffer
Yes	X	X	X	Enabled	Disabled	Enabled
No	0	0	0	Pull down	Disabled	Enabled
No	0	0	1	Pull up	Disabled	Enabled
No	0	1	0	Disabled	Disabled	Enabled
No	0	1	1	Disabled	Disabled	Enabled
No	1	X	X	Disabled	Enabled	Enabled

[(1)] X = Don't care

[(2)] DIR = 0 for input, = 1 for output

[(3)] PULDIS = 0 for enabling pull control
= 1 for disabling pull control

[(4)] PULSEL= 0 for pull-down functionality
= 1 for pull-up functionality

Inter-Integrated Circuit (I2C) Module

This chapter describes the inter-integrated circuit (I2C or I^{2}C) module. The I2C is a multi-master communication module providing an interface between the Texas Instruments (TI) microcontroller and devices compliant with Philips Semiconductor I^{2}C-bus specification version 2.1 and connected by an I2C-bus. This module will support any slave or master I2C compatible device.

Topic **Page**

31.1 Overview

The I2C has the following features:

- Compliance to the Philips I^{2}C bus specification, v2.1 (*The I2C Specification,* Philips document number 9398 393 40011)
  - Bit/Byte format transfer
  - 7-bit and 10-bit device addressing modes
  - General call
  - START byte
  - Multi-master transmitter/ slave receiver mode
  - Multi-master receiver/ slave transmitter mode
  - Combined master transmit/receive and receive/transmit mode
  - Transfer rates of 10 kbps up to 400 kbps (Phillips fast-mode rate)
- Free data format
- Two DMA events (transmit and receive)
- DMA event enable/disable capability
- Seven interrupts that can be used by the CPU
- Operates with VBUS frequency from 6.7 MHz up
- Operates with module frequency between 6.7 MHz to 13.3 MHz
- Module enable/disable capability
- The SDA and SCL are optionally configurable as general purpose I/O
- Slew rate control of the outputs
- Open drain control of the outputs
- Programmable pullup/pulldown capability on the inputs
- Supports Ignore NACK mode

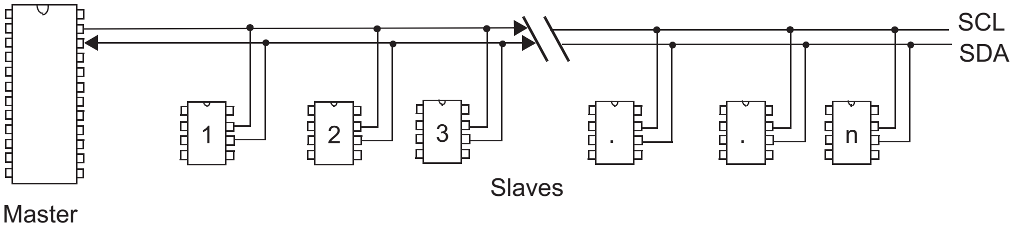
NOTE: This I2C module does **not** support:

- High-speed (HS) mode
- C-bus compatibility mode
- The combined format in 10-bit address mode (the I2C sends the slave address second byte every time it sends the slave address first byte)

31.1.1 Introduction to the I2C Module

The I2C module supports any slave or master I2C-compatible device. Figure 31-1 shows an example of multiple I2C serial ports connected for a two-way transfer from one device to another device.

Figure 31-1. Multiple I2C Modules Connection Diagram



31.1.2 Functional Overview

The I2C module is a serial bus that supports multiple master devices. In multimaster mode, one or more devices can be connected to the same bus and are capable of controlling the bus. Each I2C device on the bus is recognized by a unique address and can operate as either a transmitter or a receiver, depending on the function of the device. In addition to being a transmitter or receiver, a device connected to the I2C bus can also be considered a master or a slave when performing data transfers.

NOTE: A master device is the device that initiates the data transfer on a bus and generates the clock signal that permits the transfer. During the transmission, any device addressed by the master is considered the slave.

Data is communicated to devices interfacing to the I2C module using the serial data pin (SDA) and the serial clock pin (SCL) as shown in Figure 31-2. These two wires carry information between the device and the other devices connected to the I2C bus. Both SDA and SCL pins on the device are bidirectional. They must be connected to a positive supply voltage through a pull-up resistor. When the bus is free, both pins are high. The driver of these two pins has an open-drain configuration to perform the wired-AND function.

The device has a special mode that can be entered to ignore a NACK generated from non-compliant I2C devices that are incapable of generating an ACK.

The I2C module consists of the following primary blocks:

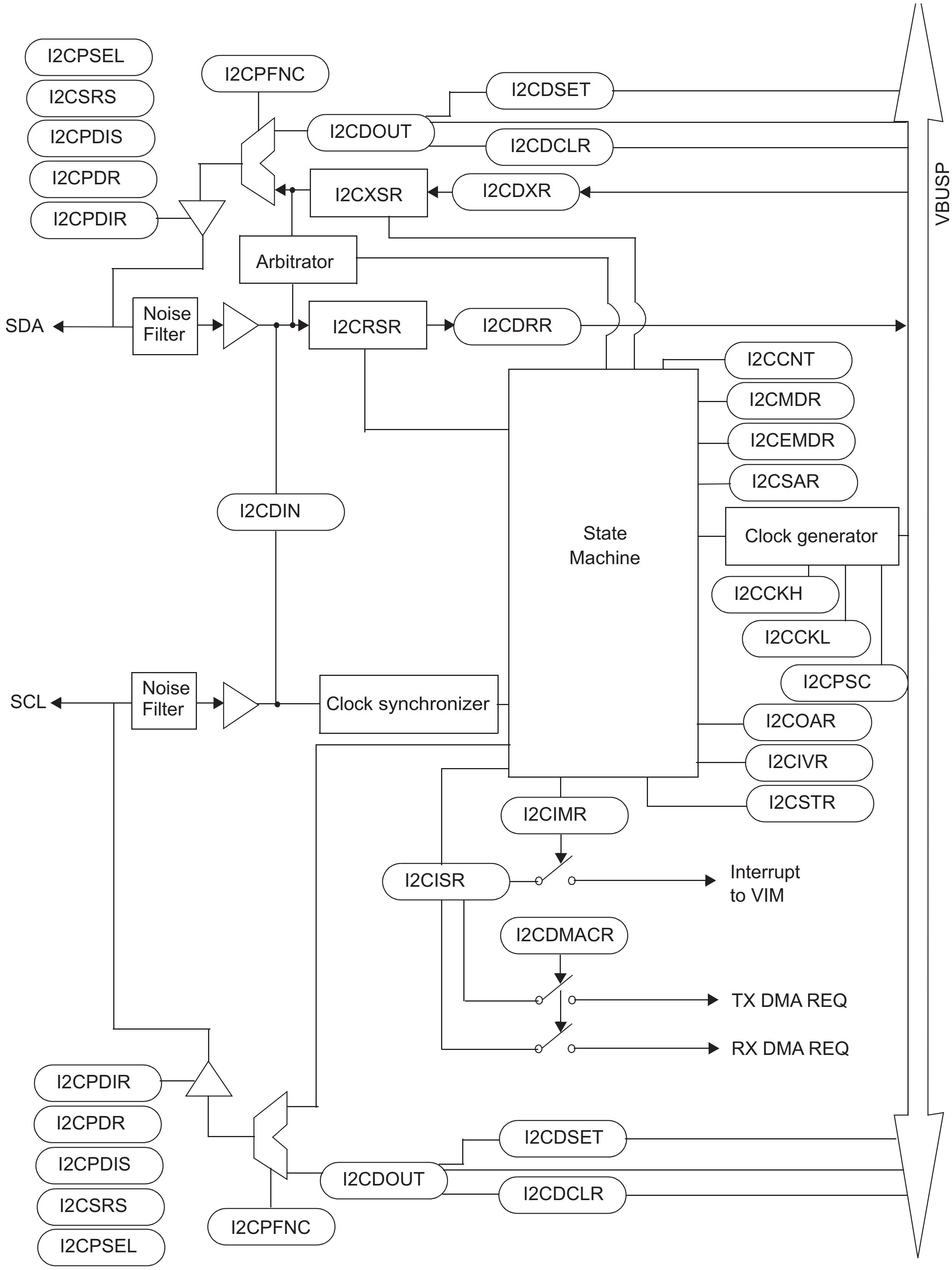
- A serial Interface: one data pin (SDA) and one clock pin (SCL)
- The device register interface
  - Data registers to temporarily hold received data and transmitted data traveling between the SDA pin and the CPU or the DMA
  - Control and status registers
- A prescaler to divide down the input clock that is driven to the I2C module
- A peripheral bus interface to enable the CPU and DMA to access the I2C module registers
- An arbitrator to handle arbitration between the I2C module (when configured as a master) and another master
- Interrupt generation logic (interrupts can be sent to the CPU)
- A clock synchronizer that synchronizes the I2C input clock (from the system module) and the clock on the SCL pin, and synchronizes data transfers with masters of different clock speeds.
- A noise filter on each of the two serial pins
- DMA event generation logic that synchronizes data reception and data transmission in the I2C module for DMA transmission

  In Figure 31-2, the CPU or the DMA writes data for transmission to I2CDXR and reads received data from I2CDRR. When the I2C module is configured as a transmitter, data written to I2CDXR is copied to I2CXSR and shifted out one bit at a time. When the I2C module is configured as a receiver, received data is shifted into I2CRSR and then copied to I2CDRR.

  When the I2C function is not needed, the pins may be controlled as general-purpose input/output (GPIO) pins. The I/O structure of each pin includes:

- programmable slew rate control of the outputs
- open drain mode
- programmable pull enable/disable on the input
- programmable pull up/pull down function on the input

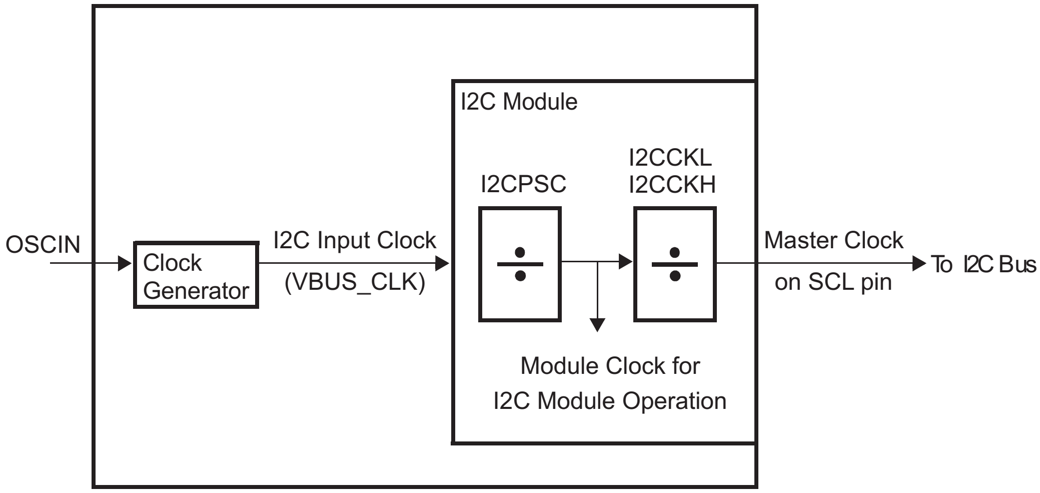
Figure 31-2. Simple I2C Block Diagram



31.1.3 Clock Generation

As shown in Figure 31-3, the I2C module uses the input clock generated from the device clock generator to generate the module clock and master clock. The I2C input clock is the device peripheral clock (VBUS_CLK). The clock is then divided twice more inside the I2C module to produce the module clock and the master clock.

Figure 31-3. Clocking Diagram for the I2C Module



The module clock determines the frequency at which the I2C module operates. A programmable prescaler in the I2C module divides down the input clock to produce the module clock. To specify the divide-down value, initialize the I2CPSC field of the prescaler register, I2CPSC. The resulting frequency is:

$$ModuleClockFrequency = \frac{I2CInputClockFrequency}{(I2CPSC+1)} \tag{63}$$

The module clock frequency must be between 6.7MHz and 13.3MHz. The prescaler can only be initialized while the I2C module is in the reset state (IRS = 0 in I2CMDR). The prescaled frequency takes effect only when IRS is changed to 1. Changing the I2CPSC value while IRS = 1 has no effect.

The master clock appears on the SCL pin when the I2C module is configured to be a master on the I2C bus. This clock controls the timing of the communication between the I2C module and a slave. As shown in Figure 31-3, a second clock divider in the I2C module divides down the module clock to produce the master clock. The clock divider uses the I2CCKL to divide down the low portion of the module clock signal and uses the I2CCKH to divide down the high portion of the module clock signal.

The resulting frequency is:

$$MasterClockFrequency = \frac{ModuleClockFrequency}{(I2CCKL+d)+(I2CCKH+d)} \tag{64}$$

$$MasterClockFrequency = \frac{I2CInputClockFrequency}{(I2CPSC+1)((I2CCKL+d)+(I2CCKH+d))} \tag{65}$$

where *d* depends on the value of I2CPSC:

I2CPSC	d
0	7
1	6
Greater than 1	5

NOTE: The master clock frequency defined above does not include rise/fall time and latency of the synchronizer inside the module. The actual transfer rate will be slower than the value calculated from the formula above. Also, due to the nature of SCL synchronization, the SCL clock period could change if SCL synchronization is taking place.

31.2 I2C Module Operation

The following section discusses how the I2C module operates.

31.2.1 Input and Output Voltage Levels

One clock pulse is generated by the master device for each data bit transferred. Because of a variety of different technology devices that can be connected to the I2C-bus, the levels of logic 0 (low) and logic 1 (high) are not fixed and depend on the associated level of V_{CCIO}. For details, see the device specific data sheet.

31.2.2 I2C Module Reset Conditions

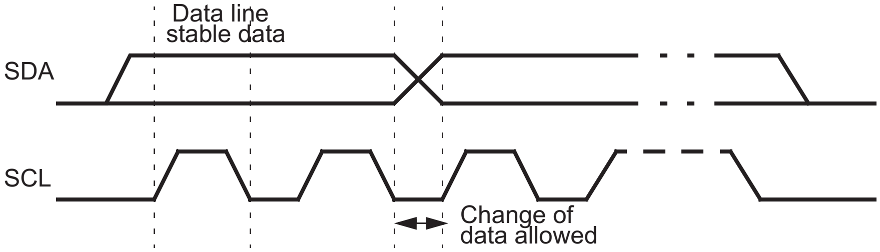
The I2C module can be reset in the following two ways:

- Through the global peripheral reset. A device reset causes a global peripheral reset.
- By clearing the $\overline{\text{IRS}}$ bit in the I2C mode register (I2CMDR). When the global peripheral reset is removed, the $\overline{\text{IRS}}$ bit is cleared to 0, keeping the I2C module in the reset state.

31.2.3 I2C Module Data Validity

The data on the SDA must be stable during the high period of the clock. See Figure 31-4. The high and low state of the data line, the SDA, can only change when the clock signal is low.

Figure 31-4. Bit Transfer on the I2C Bus

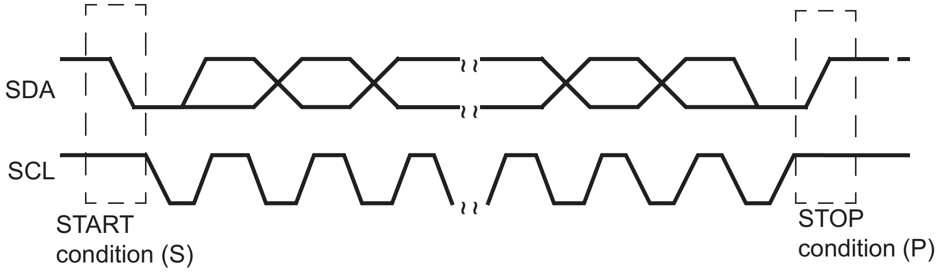


31.2.4 I2C Module Start and Stop Conditions

START and STOP conditions are generated by a master I2C module.

- The START condition is defined as a high-to-low transition on the SDA line while SCL is high. A master drives this condition to indicate the start of data transfer. The bus is considered to be busy after the START condition, and the bus busy bit (BB) in I2CSR is set to 1.
- The STOP condition is defined as a low-to-high transition on the SDA line while SCL is high. A master drives this condition to indicate the end of data transfer. The bus is considered to be free after the STOP condition, therefore the BB bit in I2CSR is cleared to 0.

Figure 31-5. I2C Module START and STOP Conditions

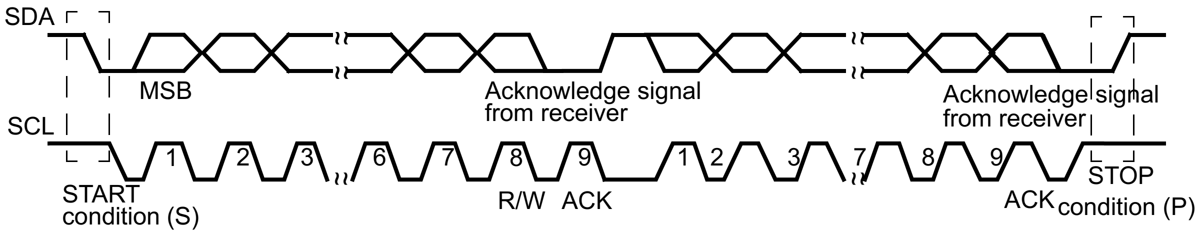


For the I2C module to start a data transfer with a START condition, the master mode bit (MST) and the START condition bit (STT) in the I2CMDR must both be set to 1. For the I2C module to end a data transfer with a STOP condition, the STOP condition bit (STP) must be set to 1. When the BB bit is set to 1 and the STT bit is set to 1, a repeated START condition is generated.

31.2.5 Serial Data Formats

The I2C module operates in byte data format. Each message put on the SDA line is 2 to 8-bits long. The number of messages that can be transmitted or received is unrestricted. The data is transferred with the most significant bit (MSB) first (Figure 31-6). Each message is followed by an acknowledge bit from the I2C if it is in receiver mode. The I2C module does not support little endian systems.

Figure 31-6. I2C Module Data Transfer



The first byte after a START condition (S) always consists of 8 bits that comprise either a 7-bit address plus the R/$\overline{\text{W}}$ bit, or 8 data bits. The eighth bit, R/W, in the first byte determines the direction of the data. When the R/$\overline{\text{W}}$ bit is 0, the master writes (transmits) data to a selected slave device; when the R/$\overline{\text{W}}$ bit is 1, the master reads (receives) data from the slave device. In acknowledge mode, an extra bit dedicated for the acknowledgement (ACK) bit is inserted after each message.

The I2C module supports the following formats:

- 7-bit addressing format (Figure 31-7)
- 10-bit addressing format (Figure 31-8)
- 7-bit/10-bit addressing format with repeated START condition (Figure 31-9)
- Free-data format (Figure 31-10)

31.2.5.1 7-Bit Addressing Format

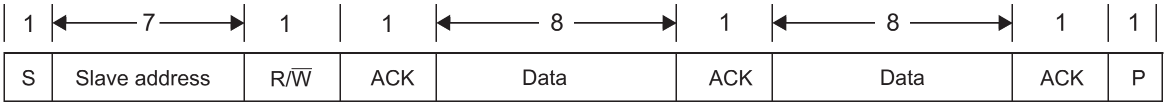
In the 7-bit addressing format (Figure 31-7), the first byte after the START condition consists of a 7-bit slave address followed by the R/$\overline{W}$ bit (in the LSB). The R/$\overline{W}$ bit determines the direction of the data transfer:

- R/$\overline{W}$ = 0: The master writes (transmits) data to the addressed slave.
- R/$\overline{W}$ = 1: The master reads (receives) data from the slave.

An extra clock cycle dedicated for acknowledgement (ACK) is inserted after each byte. If the ACK is inserted by the slave after the first byte from the master, it is followed by n bits of data from the transmitter (master or slave, depending on the R/$\overline{W}$ bit). The device I2C allows n to be a number between 2 to 8, programmable by the bit count (BC) field of I2CMDR. After the data bits have been transferred, the receiver inserts an ACK bit.

To select the 7-bit addressing format, write 0 to the expanded address enable (XA) bit of I2CMDR and make sure the free data format mode is off (FDF = 0 in I2CMDR).

Figure 31-7. I2C Module 7-Bit Addressing Format



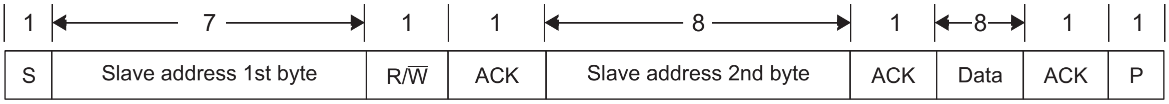
1	7	1	1	8	1	8	1	1
S	Slave address	R/$\overline{W}$	ACK	Data	ACK	Data	ACK	P

31.2.5.2 10-Bit Addressing Format

The 10-bit addressing format is similar to the 7-bit addressing format, but the master sends the slave address in two separate byte transfers. In the 10-bit addressing format (Figure 31-8), the first byte is 11110b, the two MSBs of the 10-bit slave address, and the R/$\overline{W}$ bit. The ACK bit is inserted after each byte. The second byte is the remaining 8 bits of the 10-bit slave address. The slave must send an acknowledgement after each of the two byte transfers. Once the master has written the second byte to the slave, the master can either write data or use repeated a START condition to change the data direction.

To select the 10-bit addressing format, write 1 to the expanded address enable (XA) bit of I2CMDR and make sure the free data format mode is off (FDF = 0 in I2CMDR).

Figure 31-8. I2C Module 10-bit Addressing Format

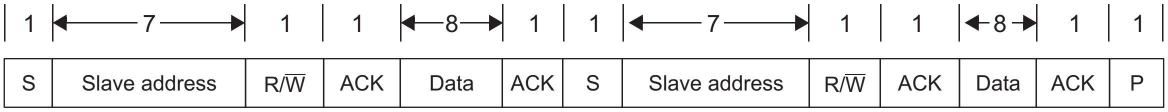


1	7	1	1	8	1	8	1	1
S	Slave address 1st byte	R/$\overline{W}$	ACK	Slave address 2nd byte	ACK	Data	ACK	P

31.2.5.3 Using the Repeated START Condition

At the end of each byte, the master can drive another START condition (Figure 31-9). Using this capability, a master can transmit/receive any number of data bytes before generating a STOP condition. The length of a data byte can be from 2 to 8 bits. The repeated START condition can be used with the 7-bit addressing, 10-bit addressing, or the free data formats.

Figure 31-9. I2C Module 7-Bit Addressing Format with Repeated START



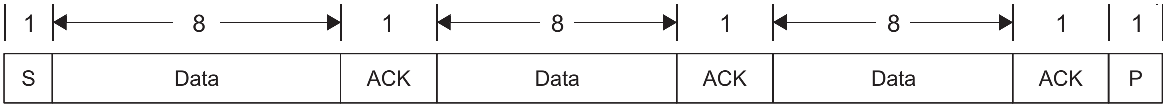
1	7	1	1	8	1	1	7	1	1	8	1	1
S	Slave address	R/$\overline{W}$	ACK	Data	ACK	S	Slave address	R/$\overline{W}$	ACK	Data	ACK	P

31.2.5.4 Free Data Format

In this format (Figure 31-10), the first byte after a START condition is a data byte. The ACK bit is inserted after each byte, followed by another 8 bits of data. No address or data direction bit is sent. Therefore, the transmitter and receiver must both support the free data format. The direction of data transmission (transmit or receive) remains constant throughout the transfer.

To select the free data format, write a 1 to the free data format (FDF) bit of the I2CMDR. The free data format is not supported in the digital loop back mode.

Figure 31-10. I2C Module in Free Data Format



1	8	1	8	1	8	1	1
S	Data	ACK	Data	ACK	Data	ACK	P

31.2.6 NACK Bit Generation

When the I2C module is a receiver (master or slave), it can acknowledge or ignore bits sent by the transmitter. To ignore any new bits, the I2C module must send a no-acknowledge (NACK) bit during the acknowledge cycle on the bus. Table 31-1 summarizes the various ways a NACK can be generated.

Table 31-1. Ways to Generate a NACK Bit

I2C Module Condition	Basic NACK Bit Generation Options	Additional Option
Slave receiver mode	Disable data transfers (STT = 0) Allow an overrun condition (RSFULL = 1) Reset the module (IRS = 0)	Set the NACKMOD bit before the rising edge of the last data bit you intend to receive.
Master receiver mode and repeat mode (RM = 1)	Generate a STOP condition (STP = 1) Reset the module (IRS = 0)	Set the NACKMOD bit before the rising edge of the last data bit you intend to receive.
Master receiver mode with non-repeat mode (RM = 0)	If STP = 1, allow the internal data counter to count down to 0 and thus force a STOP condition. If STP = 0, make STP = 1 to generate a STOP condition. Reset the module (IRS = 0)	Set the NACKMOD bit before the rising edge of the last data bit you intend to receive.

In some applications, the slave cannot generate the ACK signal. If the IGNACK bit is set in the I2CEMDR register, the resulting NACK will be ignored and the I2C block will continue the data transfer.

31.3 I2C Operation Modes

31.3.1 Master Transmitter Mode

All masters begin in this mode. The I2C module is a master and transmits control information and data to a slave. In this mode, data assembled in any of the addressing formats shown in Figure 31-7, Figure 31-8, or Figure 31-9 is shifted out onto the SDA pin and synchronized with the self-generated clock pulses on the SCL pin. The clock pulses are inhibited and the SCL pin is held low when the intervention of the device is required ($\overline{\text{XSMT}}$ = 0) after a byte has been transmitted.

NOTE: If the I2C is configured for two simultaneous master transmissions, wait until the MST and BB have been reset before performing the second master transmission.

Failure to wait for the MST and BB to reset will prevent the start condition on the second transfer from being issued and the bus BB will not be set. Typically the end of the first transfer is handled by polling BB. However, the MST bit is not reset at the same instant as the BB bit. As a result, when the second master transmission is initiated before the resetting of the MST, the MST bit for the second transfer is reset. This prevents the I2C from recognizing itself as the master, thus failing to occupy the bus.

31.3.2 Master Receiver Mode

In this mode, the I2C module is a master and receives data from a slave. This mode can only be entered from the master transmitter mode (the I2C module must first transmit a command to the slave). In any of the addressing formats shown in Figure 31-7, Figure 31-8, or Figure 31-9, the master receiver mode is entered after the slave address byte and the R/$\overline{\text{W}}$ bit have been transmitted (if the R/$\overline{\text{W}}$ bit is 1). Serial data bits received on the SDA pin are shifted in with the self-generated clock pulses on the SCL pin. The clock pulses are inhibited and the SCL is held low when the intervention of the device is required (RSFULL = 1) after a byte has been received. At the end of the transfer, the master-receiver signals the end of data to the slave-transmitter by not generating an acknowledge on the last byte that was clocked out of the slave. The slave-transmitter then releases the data line allowing the master-receiver to generate a STOP condition or a repeated START condition.

In many applications, the size of the message is in the initial bytes of the message itself. Since the size of the message is not known to the master before the transmission/reception starts, the master must use the repeat mode in order to force the stop condition when the reception is completed. The repeat mode is enabled by setting the RM bit to 1. Due to the double buffer implementation on the receive side, the master must generate the stop condition (STP =1) after reading the (message size - 1)th data.

31.3.3 Slave Transmitter Mode

In this mode, the I2C module is a slave and transmits data to a master. This mode can only be entered from the slave receiver mode (The I2C module must first receive a command from the master). In any of the addressing formats shown in Figure 31-7, Figure 31-8, or Figure 31-9, the slave transmitter mode is entered if the slave address byte is the same as its own address and the R/$\overline{\text{W}}$ bit has been transmitted (if the R/$\overline{\text{W}}$ bit is set to 1). The slave transmitter shifts the serial data out on the SDA pin with the clock pulses that are generated by the master device. The slave device does not generate the clock, but it can hold the SCL pin low when intervention of the device is required ($\overline{\text{XSMT}}$ = 0) after a byte has been transmitted.

31.3.4 Slave Receiver Mode

In this mode, the I2C module is a slave and receives data from a master. All slaves begin in this mode. Serial data bits received on the SDA pin are shifted in with the clock pulses that are generated by the master device. The slave device does not generate the clock, but it can hold the SCL pin low while intervention of the device is required (RSFULL = 1) after a byte has been received.

31.3.5 Low Power Mode

The I2C module can be placed in low-power mode by a global low-power mode initiated by the system (by writing to the Peripheral Power-Down Set Register in the Peripheral Central Resource (PCR) module.

In effect, low-power mode shuts down all the clocks to the module. In global low-power mode, no registers are visible to the software; nothing can be written to or read from any register.

31.3.6 Free Run Mode

The I2C module can be placed in free run mode when the FREE bit (I2CMDR.14) is set to 1. This bit is primarily used on an emulator when encountering a breakpoint while debugging software. When the FREE bit is set to 0, the I2C responds differently depending on whether the SCL is high or low. If the SCL is low, the I2C stops immediately and keeps driving the SCL low whether the I2C is the master transmitter or receiver. If the SCL is high, the I2C waits until the SCL becomes a low and then stops. If the I2C is a slave, it stops when the transmission/reception completes.

31.3.7 Ignore NACK Mode

The I2C module can be placed in the ignore NACK mode by setting the IGNACK bit in the I2CEMDR register. This mode allows an I2C module that is configured as a master transmitter to ignore a NACK from a slave device that is not capable of generating a proper ACK signal.

31.4 I2C Module Integrity

The following section discusses how the I2C module maintains priorities and order among signals and commands.

31.4.1 Arbitration

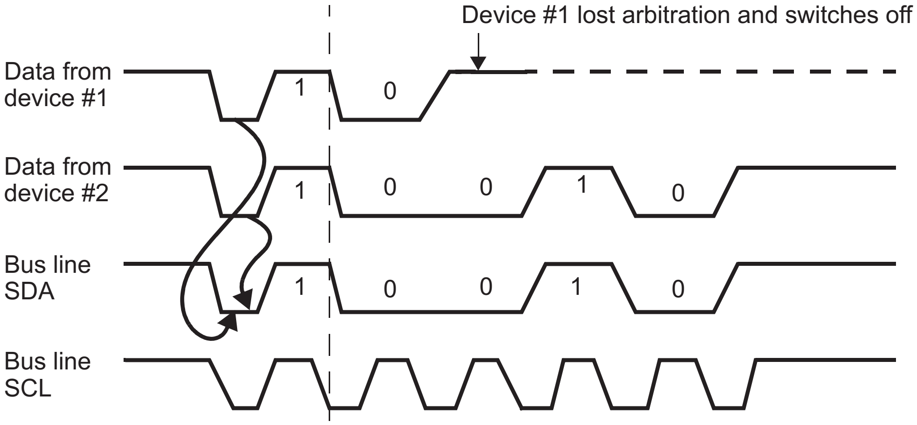
If two or more master transmitters simultaneously start a transmission on the same bus, an arbitration procedure is invoked. Figure 31-11 illustrates the arbitration procedure between two devices. The arbitration procedure uses the data presented on the SDA bus by the competing transmitters. The first master transmitter that generates a high is overruled by the other master that generates a low. The arbitration procedure gives priority to the device that transmits the serial data stream with the lowest binary value. The master transmitter that loses the arbitration switches to the slave receiver mode, sets the arbitration lost (AL) flag, and generates the arbitration-lost interrupt. The data transmitted by the other master module is salvaged, and the I2C continues to receive data from the master module. Should two or more devices send identical first bytes, arbitration continues on the subsequent bytes.

If, during a serial transfer, the arbitration procedure is still in progress when a repeated START condition or STOP condition is transmitted to I2C bus, the master transmitters involved must send the repeated START condition or STOP condition at the same position in the format frame. In other words, arbitration is not allowed between:

- A repeated START condition and a data bit
- A STOP condition and a data bit
- A repeated START condition and a STOP condition

Slaves are not involved in the arbitration procedure.

Figure 31-11. Arbitration Procedure Between Two Master Transmitters



31.4.2 I2C Clock Generation and Synchronization

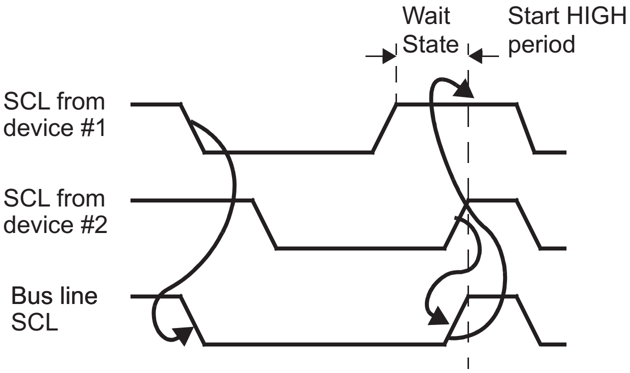
Under normal conditions only one master device generates the clock signal; the SCL. During the arbitration procedure, however, there are two or more master devices and the clock must be synchronized so that the data output can be compared. Figure 31-12 illustrates clock synchronization. The wired-AND property of the SCL line means that a device that first generates a low period on the SCL overrules the other devices. At this high-to-low transition, the clock generators of the other devices are forced to start their own low period. The SCL line is held low by the device with the longest low period. The other devices that finish their low periods must wait for the SCL line to be released before starting their high periods. A synchronized signal on the SCL is obtained where the slowest device determines the length of the low period and the fastest device determines the length of the high period.

If a device pulls down the clock line for a longer time, the result is that all clock generators must enter the wait state. In this way, a slave slows down a fast master and the slow device creates enough time to store a received byte or to prepare a byte to be transmitted.

NOTE: I2C Protocol Fault

The following conditions violate the clock spec as defined in the Philips I²C bus specification, v2.1 (*The I2C Specification,* Philips document number 9398 393 40011), and will result in an I2C protocol fault: I2CCLKH = 2 I2CCLKL = 2I2CPSC = 2. This will cause the SDA data transition to occur while the SCL is high.

Figure 31-12. Synchronization of Two I2C Clock Generators During Arbitration



31.4.3 Prescaler

The I2C module is operated by the module clock. This clock is generated by way of the I2C prescaler block. The prescaler block consists of a 8-bit register, I2CPSC, used for dividing down the device peripheral clock (VBUS_CLK) to obtain a module clock between 6.7 MHz and 13.3 MHz.

31.4.4 Noise Filter

The noise filter is used to suppress any noises that are 50ns or less. It is designed to suppress noise with one module clock, assuming the lower and upper limits of the module clock are 6.7MHz and 13.3MHz, respectively.

31.5 Operational Information

The following section provides specific information about how the I2C module operates.

31.5.1 I2C Module Interrupts

The I2C module generates seven types of interrupts. These seven interrupts are accompanied with seven interrupt mask bits in the interrupt mask register (I2CIMR) and with seven interrupt flag bits in the status register (I2CSR).

31.5.1.1 I2C Interrupt Requests

The I2C module generates the interrupt requests described below. All requests are multiplexed through an arbiter into a single I2C interrupt request to the CPU. Each interrupt request has a flag bit and an enable bit. Interrupts must be enabled prior to the occurrence of the expected interrupt condition. When one of the specified events occurs, the flag bit is set. If the corresponding enable bit is 0, the interrupt request is blocked. If the enable bit is 1, the interrupt request is forwarded to the CPU as an I2C interrupt request. As an alternative, the CPU can poll all of the bits shown in Table 31-2.

Table 31-2. Interrupt Requests Generated by I2C Module

Flag	Name	Generated
AL	Arbitration-lost interrupt	Generated when the I2C module has lost an arbitration contest with another master-transmitter
NACK	No-acknowledge interrupt	Generated when the master I2C does not receive an acknowledge from the receiver
ARDY	Register-access-ready interrupt	Generated when the previously programmed address, data and command have been performed and the status bits have been updated. The interrupt is used to notify the device that the I2C registers are ready to be accessed.
RXRDY	Receive-data-ready interrupt	Generated when the received data in the receive-shift register (I2CSR) has been copied into the data receive register (I2CDRR). The RXRDY bit can also be polled by the device to determine when to read the received data in the I2CDRR.
TXRDY	Transmit-data-ready interrupt	Generated when the transmitted data has been copied from the data transmit register (I2CDXR) into the transmit-shift register (I2CXSR). The TXRDY bit can also be polled by the device to determine when to write the next data into I2CDXR.
SCD	Stop-condition-detect interrupt	Generated when a STOP condition has been detected.
AAS	Address-as-slave interrupt	Generated when the I2C has recognized its own slave address or an address of all zeroes.

The interrupt vector register (I2CIVR) contains the binary-coded-interrupt vector that indicates the highest priority interrupt that is pending and enabled. When I2CIVR is read, the corresponding interrupt flags for AL, NACK and SCD are automatically cleared, if their interrupts are enabled. Reading the I2CIVR will not clear the AAS, ARDY, RXRDY, or TXRDY interrupt pending flags. Please see Section 31.6.3 for the method to clear these four flags.

If more than one interrupt is pending, a new interrupt will be generated for the next highest priority pending interrupt when you re-enable the I2C interrupt.

A transmit interrupt is generated just after the START condition in master transmitter mode. This ensures that the CPU will get an interrupt even if no slave returns an ACK to the slave address following the START condition.

It is important to note that when the I2C is configured to generate interrupts as a slave transmitter and the backward compatibility mode (BCM) bit is set to 1, an extra transmit interrupt occurs. The application should monitor the ACK from the master to determine whether to load another byte into I2CDXR.

31.5.2 DMA Controller Events

The I2C module has two events that use the DMA controller to synchronously read received data (I2CREVNT) from I2CDRR, and synchronously write data (I2CWEVNT) to the transmit buffer, I2CDXR. The read and write events have the same timing as I2CRRDY (I2CRINT) and I2CXRDY (I2CXINT), respectively.

The CPU or the DMA controller reads the received data from I2CDRR and writes the data to be transmitted to I2CDXR. The RXRDY bit is automatically cleared when the DMA controller reads the I2CDRR register, and the TXRDY bit is automatically cleared when the DMA controller writes to the I2CDXR register.

Data written to I2CDXR is copied to I2CXSR and shifted out from the SDA pin when the I2C module is configured as a transmitter. When the I2C module is configured as a receiver, received data is shifted into ICRSR and copied to I2CDRR, which can be read by the CPU or the DMA controller.

A transmit event (I2CWEVNT) is generated after a START condition in master transmitter mode. This ensures that the DMA gets an event even if no slave returns an ACK to the slave address following the START condition.

NOTE: Unexpected DMA transmit and receive event

An unexpected DMA transmit event (ICXEVT) and a DMA receive event (ICXRDY) are generated in 10-bit, master transmit, repeat mode. This event occurs soon after the start condition but before the first bit of the address is transmitted. In this event, no DMA activity should be initiated without the slave ACK being received.

31.5.3 I2C Enable/Disable

The I2C module can be enabled or disabled with the I2C reset enable bit (IRS) in the I2C module register (I2CMDR). This occurs in one of two ways:

- Write 0 to the I2C reset bit (IRS) in I2CMDR. All status bits are forced to the default values and the I2C mode remains disabled until IRS is changed to 1. The SDA and SCL pins are in the high impedance state.
- Initiate a device reset by driving the $\overline{\text{PORRST}}$ pin low. The entire device is reset and is held in the reset state until the pin is released and is driven high. When $\overline{\text{PORRST}}$ is released, all I2C module registers are reset to their default values. The IRS bit is forced to 0, which resets the I2C module. The I2C module stays in the reset state until a 1 is written to the IRS bit.

IRS must be 0 while the I2C module is being configured. Forcing IRS to 0 can be used to save power and also clear error conditions.

31.5.4 General Purpose I/O

Both of the I2C pins can be programmed to be general-purpose I/O pins via the I2C pin control registers (I2CPFNC, I2CDIR, I2CDOUT, and I2CDIN).

When the I2C module is not used, the I2C pins may be programmed to be either general purpose input or general-purpose output pins. This function is controlled in the I2CDIR and I2CPFNC registers. Note that each pin can be programmed to be either an I2C pin or a GIO pin.

If the I2C function is to be used, the application software must ensure that each pin is configured as an I2C pin and not a GIO pin, or else unexpected behavior may result.

31.5.5 Pull Up/Pull Down Function

I2C module pins can have either an active pull up or active pull down that makes it possible to leave the pins unconnected externally. The pins can be programmed to have the active pull function enabled or disabled by writing to the corresponding bit in the I2CPDIS register. Please see the device-specific data sheet for the default internal pull (pull-up, pull-down or no pull) on the pins.

The pull on the pins is programmable to a setting other than the default internal pull as specified in the data sheet. The pins can be programmed to have either an active pull up or an active pull down function by writing to the corresponding bit in I2CPSEL register. The pull up/pull down function is active on the pin only when the pull enabled is programmed in the I2CPDIS register.

The pull up/pull down functions are deactivated when a bidirectional pin is configured as an output. At system reset, the pull up function of all the pins is enabled. Please see the device-specific data sheet for the current supplied by the pull up/pull down.

31.5.6 Open Drain Function

The I2C pins can be programmed to include an open drain function when they are configured as output pins. This is done by writing to the corresponding bit of the I2CPDR register. When the open drain function is enabled, a low value (0) written to the data output register forces the pin to a low output voltage (V_{OL} or lower), whereas a high value (1) written to the data output register forces the pin to a high-impedance state. The open drain function is disabled when the pin is configured as an input pin.

31.6 I2C Control Registers

Table 31-3 provides a summary of the control registers. The upper word (upper 16 bits) of the registers all read as 0s. Writes have no effect on these bits. The base address for the control registers is FFF7 D400h for I2C1 and FFF7 D500h for I2C2.

Table 31-3. I2C Control Registers

Offset	Acronym	Register Description	Section
00h	I2COAR	I2C Own Address Manager	Section 31.6.1
04h	I2CIMR	I2C Interrupt Mask Register	Section 31.6.2
08h	I2CSTR	I2C Status Register	Section 31.6.3
0Ch	I2CCKL	I2C Clock Divider Low Register	Section 31.6.4
10h	I2CCKH	I2C Clock Control High Register	Section 31.6.5
14h	I2CCNT	I2C Data Count Register	Section 31.6.6
18h	I2CDRR	I2C Data Receive Register	Section 31.6.7
1Ch	I2CSAR	I2C Slave Address Register	Section 31.6.8
20h	I2CDXR	I2C Data Transmit Register	Section 31.6.9
24h	I2CMDR	I2C Mode Register	Section 31.6.10
28h	I2CIVR	I2C Interrupt Vector Register	Section 31.6.11
2Ch	I2CEMDR	I2C Extended Mode Register	Section 31.6.12
30h	I2CPSC	I2C Prescale Register	Section 31.6.13
34h	I2CPID1	I2C Peripheral ID Register 1	Section 31.6.14
38h	I2CPID2	I2C Peripheral ID Register 2	Section 31.6.15
3Ch	I2CDMACR	I2C DMA Control Register	Section 31.6.16
48h	I2CPFNC	I2C Pin Function Register	Section 31.6.17
4Ch	I2CPDIR	I2C Pin Direction Register	Section 31.6.18
50h	I2CDIN	I2C Data Input Register	Section 31.6.19
54h	I2CDOUT	I2C Data Output Register	Section 31.6.20
58h	I2CDSET	I2C Data Set Register	Section 31.6.21
5Ch	I2CDCLR	I2C Data Clear Register	Section 31.6.22
60h	I2CPDR	I2C Pin Open Drain Register	Section 31.6.23
64h	I2CPDIS	I2C Pull Disable Register	Section 31.6.24
68h	I2CPSEL	I2C Pull Select Register	Section 31.6.25
6Ch	I2CSRS	I2C Pins Slew Rate Select Register	Section 31.6.26

31.6.1 I2C Own Address Manager (I2COAR)

The 16-bit memory-mapped I2C own address register is used to specify its own address. Figure 31-13 and Table 31-4 describe this register.

Figure 31-13. I2C Own Address Manager Register (I2COAR) [offset = 00]

15 — 10	9 — 0
Reserved	OA
R-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 31-4. I2C Own Address Manager Register (I2COAR) Field Descriptions

Bit	Field	Value	Description
15-10	Reserved	0	Reads return 0. Writes have no effect.
9-0	OA	0-3FFh	Own address These bits reflect the bus address of the I2C module. When the expand address (XA) bit I2CMDR.8 is set to 1, the I2C is in expand address mode (10-bit addressing mode). In either 7 or 10-bit address mode, all 10-bits are both readable and writable. Bits 7, 8, and 9 should only be used in 10-bit address mode. Table 31-5 provides the correct modes for these bits. Note that the user can program the I2C own address to any value as long as it does not conflict with other components in the system.

Table 31-5. Correct Mode for OA Bits

Bits Used	Mode	Value of XA
OA.6:0	7 Bit Addressing	0
OA.9:0	10 Bit Addressing	1

31.6.2 I2C Interrupt Mask Register (I2CIMR)

The 7-bit memory-mapped I2C interrupt mask register is used by the device to enable/disable the interrupts. Figure 31-14 and Table 31-6 describe this register.

Figure 31-14. I2C Interrupt Mask Register (I2CIMR) [offset = 04h]

15	8
Reserved	
R-0	

7	6	5	4	3	2	1	0
Reserved	AASEN	SCDEN	TXRDYEN	RXRDYRN	ARDYEN	NACKEN	ALEN
R-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 31-6. I2C Interrupt Mask Register (I2CIMR) Field Descriptions

Bit	Field	Value	Description
15-7	Reserved	0	Reads return 0. Writes have no effect.
6	AASEN		Address As Slave Interrupt Enable.
		0	AASEN interrupt is disabled.
		1	AASEN interrupt is enabled.
5	SCDEN		Stop Condition Interrupt Enable.
		0	SCDEN interrupt is disabled.
		1	SCDEN interrupt is enabled.
4	TXRDYEN		Transmit Data Ready Interrupt Enable.
		0	TXRDYEN interrupt is disabled.
		1	TXRDYEN interrupt is enabled.
3	RXRDYEN		Receive Data Ready Interrupt Enable.
		0	RXRDYEN interrupt is disabled.
		1	RXRDYEN interrupt is enabled.
2	ARDYEN		Register Access Ready Interrupt Enable.
		0	ARDYEN interrupt is disabled.
		1	ARDYEN interrupt is enabled.
1	NACKEN		No Acknowledgement Interrupt Enable.
		0	NACKEN interrupt is disabled.
		1	NACKEN interrupt is enabled.
0	ALEN		Arbitration Lost Interrupt Enable.
		0	ALEN interrupt is disabled.
		1	ALEN interrupt is enabled.

31.6.3 I2C Status Register (I2CSTR)

Figure 31-15 and Table 31-7 describe this register.

Figure 31-15. I2C Status Register (I2CSR) [offset = 08h]

15	14	13	12	11	10	9	8
Reserved	SDIR	NACKSNT	BB	RSFULL	XSMT	AAS	AD0
R-0	R/W1C-0	R/W1C-0	R-0	R-0	R/W-1	R-0	R-0

7	6	5	4	3	2	1	0
Reserved		SCD	TXRDY	RXRDY	ARDY	NACK	AL
R-0		R/W1C-0	R/W-1	R/W1C-0	R/W1C-0	R/W1C-0	R/W1C-0

LEGEND: R/W = Read/Write; R = Read only; W1C = Write 1 to clear; -*n* = value after reset

Table 31-7. I2C Status Register (I2CSTR) Field Descriptions

Bit	Field	Value	Description
15	Reserved	0	Reads return 0. Writes have no effect.
14	SDIR		Slave direction. Setting this bit to 1 indicates that the I2C slave is a transmitter. Clearing this bit to 0 indicates that the I2C is a master transmitter/receiver or a slave receiver. This bit is also cleared by the STOP or START conditions. In DLB mode (in which the configuration should be master-transmitter slave-receiver), this bit is cleared to 0. **Writing a 1 to this bit will clear it.**
		0	The I2C is a master transmitter/receiver or a slave receiver.
		1	The I2C is a slave transmitter.
13	NACKSNT		No acknowledge sent. This bit is set to 1 to indicate that a no acknowledgement (NACK) has been sent because the NACKMOD bit was set to 1. **Writing a 1 to this bit will clear it.**
		0	A NACK has not been sent.
		1	A NACK was sent because the NACKMOD was set to 1.
12	BB		Bus busy. This bit indicates the state of the serial bus. On reception of a START condition or if the I2C detects a low state on I2CSCL, the device sets BB = 1. If the nIRS is set to 1 during transaction between other I2C devices, the BB bit is set at the first falling edge of SCL or START condition. BB is cleared to 0 after the reception of a STOP condition. BB is kept to 0 regardless of the SCL state when the I2C is in reset (nIRS = 0).
		0	The bus is free.
		1	The bus is busy.
11	RSFULL		Receiver shift full. This bit is set to 1 to indicate that the receiver has experienced overrun. Overrun occurs when the receive shift register is full and I2CDRR has not been read since the receive shift register to I2CDRR transfer. The contents of I2CDRR are not lost. The I2C core logic is holding for I2CDRR read access. This bit is also set when, in master-repeat-mode, the I2C receives a byte of data. There is no difference between RXRDY and RSFULL in this case. The I2C master will not continue the transfer as long as the received data is in the I2CDRR or receive shift register. **RSFULL is cleared when reading the I2CDRR, resetting the I2C (nIRS = 0), or resetting the device.**
		0	No overrun has occurred.
		1	An overrun has occurred.

Table 31-7. I2C Status Register (I2CSTR) Field Descriptions (continued)

Bit	Field	Value	Description
10	XSMT		Transmit shift empty. This bit is cleared to 0 to indicate that the transmitter has experienced underflow. Underflow occurs when the transmit shift register is empty and I2CDXR has not been loaded since the last I2CDXR to transmit shift register transfer. The I2C core logic is waiting for I2CDXR write access. **XSMT is set to 1 as a result of writing to I2CDXR, by resetting the I2C block (nIRS = 0), or by resetting the device.** In repeat mode, if the I2C in master transmitter mode is holding transfer with $\overline{\text{XSMT}}$ = 0 (that is, waiting for further action) and the STT or STP bit is set, XSMT is set to 1 by hardware.
		0	An underflow has occurred.
		1	No underflow has occurred.
9	AAS		Address as slave. **This bit cannot be cleared by writing a 1 to the bit or by reading the I2CIVR register.**
		0	This bit is cleared by a STOP condition or detection of any address byte that does not match I2COAR.
		1	This bit is set to 1 by the device when it has recognized its own slave address or an address of all zeros (general call).
8	AD0		Address zero status.
		0	A START or STOP condition was detected. No general call was detected.
		1	An address of all zeros (general call) was detected.
7-6	Reserved	0	Reads return 0. Writes have no effect.
5	SCD		Stop condition detect interrupt flag. This bit is set to 1 when the I2C receives or sends a STOP condition. **This bit is cleared to 0 by writing a 1 to this bit or reading the value 0x0006 from I2CIVR. Writing a 1 to this bit will clear the value 0x0006 from I2CIVR.**
		0	No STOP condition has been sent or received.
		1	A STOP condition has been sent or received.
4	TXRDY		Transmit data ready interrupt flag. This bit is set to 1 to indicate when data in the transmit data register, I2CDXR, has been copied into the transmit shift register. This bit can also be polled by the device to indicate when to write the next transmitted data into the I2CDXR. **Writing a 1 to this bit will set it.** **This bit is cleared to 0 and code 0x0005 in I2CIVR is cleared when the I2CDXR is written. This bit cannot be cleared by reading the I2CIVR register.**
		0	I2CDXR contains data to transmit.
		1	I2CDXR is empty.
3	RXRDY		Receive data ready interrupt flag. This bit is set to 1 to indicate when the data in the receive shift register has been copied into the data receive register (I2CDRR). This bit can also be polled by the device to indicate when to read the received data in the I2CDRR. **Writing a 1 to this bit or reading from I2CDRR will clear this bit, and will also clear code 0x0004 from I2CIVR. This bit cannot be cleared by reading the I2CIVR register.**
		0	The I2CDRR has been read.
		1	The received data has been written into the I2CDRR.

Table 31-7. I2C Status Register (I2CSTR) Field Descriptions (continued)

Bit	Field	Value	Description
2	ARDY		Register access ready interrupt flag. This bit is set to 1 when the previously programmed address, data and command has been performed and the status bit has been updated. The flag is used by the device to indicate that the I2C registers are ready to be accessed again. **This bit is automatically cleared by hardware when writing data to I2CDXR in transmit mode, reading data from I2CDRR in receive mode, or setting STT or STP bit. Writing a 1 to this bit will clear this bit. This bit cannot be cleared by reading the I2CIVR register.** When RM = 0, ARDY is set when I2CCNT is passed 0 if STP register bit has not been set. When RM = 1, ARDY is set at each byte end. When FDF = 0, ARDY is asserted after the ACK for the slave address. When FDF = 1, there is no slave address. Therefore, ARDY is asserted after sending the start condition.
		0	*Nonrepeat mode, (RM = 0):* I2C registers are not ready to be accessed. *Repeat mode (RM = 1):* I2C registers are not ready to be accessed.
		1	*Nonrepeat mode, (RM = 0):* ICCNT passes 0 (if STP bit has not been set). *Repeat mode (RM = 1)*: The end of each byte was transmitted from I2CDXR.
1	NACK		No acknowledgement interrupt. This bit is set to 1 when the master I2C does not receive an acknowledgement from the receiver. This bit is set only when the I2C has received a no-acknowledge in master mode. This bit is NOT set by no-acknowledgement after Start byte. In master start byte mode, the first byte (address of all zeroes) receives a NACK but does not clear the stop bit. **Writing a 1 to this bit or reading the value 0x0002 from I2CIVR will clear this bit.**
		0	An acknowledge was detected.
		1	No acknowledge was detected or the I2C is operating in the general call, even though an acknowledgement was received. This value clears the STP bit.
0	AL		Arbitration lost interrupt flag. This bit is set to 1 when arbitration has been lost. **Writing a 1 to this bit or reading the value 0x0001 from I2CIVR will clear this bit.**
		0	No loss of arbitration has been detected.
		1	The device in the master transmitter mode senses it has lost an arbitration. This occurs when two or more transmitters start a transmission almost simultaneously or when the I2C attempts to start a transfer while BB=1. When this is set to 1 due to arbitration lost, the device becomes a slave receiver and the MST, STT and STP bits in I2CMDR are cleared to 0.

31.6.4 I2C Clock Divider Low Register (I2CCKL)

The I2C clock divider low register is a 16-bit memory-mapped register used to divide the master clock down to obtain the I2C serial clock low time. Figure 31-16 and Table 31-8 describe this register.

Figure 31-16. I2C Clock Divider Low Register (I2CCKL) [offset = 0Ch]

15 ... 0
CLKL
R/W-0

LEGEND: R/W = Read/Write; -*n* = value after reset

Table 31-8. I2C Clock Divider Low Register (I2CCKL) Field Descriptions

Bit	Field	Description
15-0	CLKL	Low time clock division factor. Used to divide down the module clock to create the low-time portion of the master clock signal that will appear on the SCL pin: $$LowTime = \left(\frac{I2CCLKL + d}{ModuleClockFrequency}\right) \quad (66)$$ where *d* is the value that depends on the I2CPSC (see Section 31.1.3). This register must be configured while the I2C is still in reset (nIRS = 0).

31.6.5 I2C Clock Control High Register (I2CCKH)

The I2C clock divider high register is a 16-bit memory-mapped register used to divide the master clock down to obtain the I2C serial clock high time. Figure 31-17 and Table 31-9 describe this register.

Figure 31-17. I2C Clock Control High Register (I2CCKH) [offset = 10h]

15 ... 0
CLKH
R/W-0

LEGEND: R/W = Read/Write; -*n* = value after reset

Table 31-9. I2C Clock Control High Register (I2CCKH) Field Descriptions

Bit	Field	Description
15-0	CLKH	High time clock division factor. Used to divide down the module clock to create the high-time portion of the master clock signal that will appear on the SCL pin: $$HighTime = \left(\frac{I2CCLKH + d}{ModuleClockFrequency}\right) \quad (67)$$ where *d* is the value that depends on the I2CPSC (see Section 31.1.3). This register must be configured while the I2C is still in reset (nIRS = 0).

31.6.6 I2C Data Count Register (I2CCNT)

The I2C data count register is a 16-bit memory-mapped register used to count received or transmitted data bytes. This register is also used to generate the STOP condition that terminates the transfer after the counter reaches zero. Figure 31-18 and Table 31-10 describe this register.

Figure 31-18. I2C Data Count Register (I2CCNT) [offset = 14h]

15 … 0
CNT
R/W-0

LEGEND: R/W = Read/Write; -*n* = value after reset

Table 31-10. I2C Data Count Register (I2CCNT) Field Descriptions

Bit	Field	Value	Description
15-0	CNT		Data counter. This down counter is used to generate a stop condition if a stop condition is specified (STP = 1). **Note: ICCNT is a don't care when RM is set to 1.**
		0	The data counter is 65536.
		1	The data counter is 1.

31.6.7 I2C Data Receive Register (I2CDRR)

The I2C data receive register is a 16-bit memory-mapped register used by the device to read the received data. Figure 31-19 and Table 31-11 describe this register.

Figure 31-19. I2C Data Receive Register (I2CDRR) [offset = 18h]

15 … 8	7 … 0
Reserved	DATARX
R-0	R-0

LEGEND: R = Read only; -*n* = value after reset

Table 31-11. I2C Data Receive Register (I2CDRR) Field Descriptions

Bit	Field	Value	Description
15-8	Reserved	0	Reads return 0. Writes have no effect.
7-0	DATARX	0-FFh	Receive data. A read from this register clears the RXRDY bit and clears code 4h from the I2CIVR register.

31.6.8 I2C Slave Address Register (I2CSAR)

The I2C slave address register is a 16-bit memory-mapped register used to specify the address of the slave device to communicate to on the I2C bus. Figure 31-20 and Table 31-12 describe this register.

Figure 31-20. I2C Slave Address Register (I2CSAR) [offset = 1Ch]

15 10	9 0
Reserved	SA
R-0	R/W-3FFh

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 31-12. I2C Slave Address Register (I2CSAR) Field Descriptions

Bit	Field	Value	Description
15-10	Reserved	0	Reads return 0. Writes have no effect.
9-0	SA		7- or 10-bit programmable slave address. In either mode, all 10-bits are readable and writable. Bits 7, 8, and 9 should only be used in 10-bit address mode. Table 31-13 illustrates the correct mode for each bit.

Table 31-13. Correct Mode for SA Bits

Bits Used	Mode	Value of XA
SA(6–0)	7-bit addressing	0
SA(9–0)	10-bit addressing	1

31.6.9 I2C Data Transmit Register (I2CDXR)

Figure 31-21 and Table 31-14 describe this register.

Figure 31-21. I2C Data Transmit Register (I2CDXR) [offset = 20h]

15 8	7 0
Reserved	DATATX
R-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 31-14. I2C Data Transmit Register (I2CDXR) Field Descriptions

Bit	Field	Value	Description
15-8	Reserved	0	Reads return 0. Writes have no effect.
7-0	DATATX	0-FFh	Transmit data. Data written to this register will be transmitted on the I2C bus. A write to this register clears the TXRDY bit and clears code 0x05 from the I2CIVR register.

31.6.10 I2C Mode Register (I2CMDR)

Figure 31-22 and Table 31-15 describe this register.

Figure 31-22. I2C Mode Register (I2CMDR) [offset = 24h]

15	14	13	12	11	10	9	8
NACKMOD	FREE	STT	Reserved	STP	MST	TRX	XA
R/W-0	R/W-0	R/W-0	R-0	R/W-0	R/W-0	R/W-0	R/W-0

7	6	5	4	3	2		0
RM	DLB	nIRS	STB	FDF	BC		
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0		

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 31-15. I2C Mode Register (I2CMDR) Field Descriptions

Bit	Field	Value	Description
15	NACKMOD		No-acknowledge (NACK) mode. This bit is used to send an acknowledge (ACK) or a no-acknowledge (NACK) to the transmitter. This bit is only applicable when the I2C is in receiver mode. In master receiver mode, when the internal data counter decrements to 0, the I2C sends a NACK. The master receiver I2C finishes a transfer when it sends a NACK. The I2C ignores ICCNT when NACKMOD is 1. The NACKMOD bit should be set before the rising edge of the last data bit if a NACK must be sent, and this bit is cleared once a NACK has been sent.
		0	The I2C sends an ACK to the transmitter during the acknowledge cycle.
		1	The I2C sends a NACK to the transmitter during the acknowledge cycle.
14	FREE		Free running bit. This bit is used to determine the state of the I2C when a breakpoint is encountered in the high level language (HLL) debugger.
		0	The I2C stops immediately if SCL is low and keeps driving SCL low if the I2C master is a transmitter/receiver. If SCL is high, I2C waits until SCL becomes low and then stops. If the I2C is a slave, it will stop when the transmission/reception completes.
		1	The I2C runs free.
13	STT		Start condition. The start condition bit works with the STP bit (master only mode). The STT and STP bits are configured to generate different transfer formats (see Table 31-16). The STT and STP bits can be used to terminate the repeat mode. This bit takes one I2C module clock cycle to set.
		0	STT is reset to 0 by the hardware after the START condition has been generated.
		1	STT is set to 1 by the device to generate a START condition. In master mode, setting STT to 1 generates a START condition.
12	Reserved	0	Reads return 0. Writes have no effect.
11	STP		Stop condition (Master mode only). This bit can be set to a 1 by the CPU to generate a stop condition. It is reset to 0 by the hardware after the stop condition has been generated. The stop condition is generated when ICCNT passes 0 when the I2C is in non-repeat mode (RM=0). In repeat mode (RM=1), the stop condition is generated if STP bit is 1. In transmitter mode, I2CTXRDY needs to be 1 (that is, you have to set STP bit unless you write data into I2CDXR).
		0	STP is reset to 0 by the hardware after the STOP condition has been generated.
		1	STP is set to 1 by the device to generate a STOP condition.
10	MST		Master/slave mode bit. This bit determines whether the module will operate in master or slave mode; see Table 31-17. This bit is cleared after generating a STOP condition. The BB bit is cleared first, and MST bit is cleared second. Before starting the next transaction in master mode, this bit must be confirmed to be cleared.
		0	The module is in the slave mode and the clock is received from the master device.
		1	The module is in the master mode and it generates the clock. This bit is cleared when the transfer has completed.

Table 31-15. I2C Mode Register (I2CMDR) Field Descriptions (continued)

Bit	Field	Value	Description
9	TRX		Transmit/receive bit.
			This bit determines the direction of data transmission of the I2C module. See Table 31-17.
		0	The module is in the receive mode and data on the SDA line is shifted into the data register I2CDRR.
		1	The module is in the transmit mode and the data in the I2CDXR is shifted out on the SDA line.
8	XA		Expand address enable bit.
			This bit controls the addressing mode. When XA is set to 1, the I2C does not support the combined format in master mode operations. However, the I2C will acknowledge and support the formats when configured as a slave. This bit needs to be configured even if the I2C is in slave mode.
		0	The mode is set to 7-bit addressing mode (normal address mode).
		1	The mode is set to 10-bit addressing mode (expanded address mode).
7	RM		Repeat mode enable bit (Master mode only).
			This bit is a ‘don’t care’ if the module is configured in slave mode (MST = 0); see Table 31-16. Each time a byte of data is received, the user should decide whether or not to continue receiving more data. See Figure 31-23 for a diagram of this function.
		0	The mode is not in repeat mode.
		1	In repeat mode, data is continuously transmitted out of the ICDXR or received into the ICDRR until the STP bit is set to 1 regardless of ICCNT value. See Table 31-16 for module conditions.
6	DLB		Digital loop back enable bit.
			This bit enables the digital loopback mode of the I2C. This bit only applies in Master transmitter mode.
		0	Digital loop back mode is disabled.
		1	Digital loop back mode is enabled. In digital loop back mode, data transmitted out of the I2CDXR will be received in the I2CDRR. The address of the I2COAR is output on SDA.
5	nIRS		I2C reset enable bit.
			When cleared to 0, this bit will place all status registers in this module to their default state. Resetting nIRS during a data transfer can hang the I2C bus.
		0	I2C is in reset.
		1	I2C is out of reset.
4	STB		Start byte mode enable bit (Master mode only).
			The Start byte mode bit is set to 1 by the CPU to configure the I2C in Start byte mode. The I2C sends 00000001 regardless of the I2CSAR value. Refer to the Philips I2C specification for more details.
		0	The module is not in START byte mode.
		1	The module is in START byte mode.
3	FDF		Free data format enable bit.
			This bit configures the module to operate in free data format mode (see Table 31-17) in both master and slave modes. When FDF is 0, ARDY is asserted after ACK for the slave address. When FDF is 1, there is no slave address. Therefore, ARDY is asserted after sending the start condition. FDF mode is not supported in digital loop back mode.
		0	The module is not in free data format mode.
		1	The module is in free data format mode.
2-0	BC		Bit count.
			This bit defines the number of bits starting from the LSB (excluding the acknowledge bit) that are sent on the bus when data is written to the data transmit register.
			If the bits BC0, BC1, and BC2 are all 0, then the number of bits sent on the bus is 8. If the bit count bits are a non-zero value, then the number of bits sent on the bus is that value. The value 001 is reserved. When performing a transfer using the bit count of, for example, n (where n is nonzero), only the n least significant bits in the data receive register are valid and correct. The rest of the bits should be disregarded. See Table 31-18 for more information.

Figure 31-23. Typical Timing Diagram of Repeat Mode

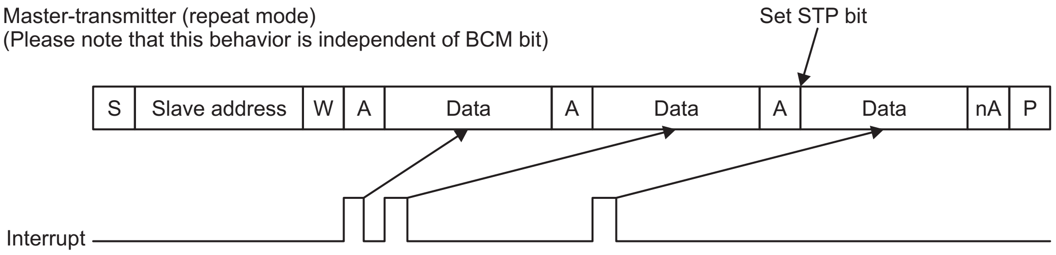

Master-transmitter (repeat mode)
(Please note that this behavior is independent of BCM bit)
Set STP bit
S
Slave address
W
A
Data
A
Data
A
Data
nA
P
Interrupt

Table 31-16. I2C Module Condition, Bus Activity, and Mode

RM	STT	STP	Condition	Bus Activities [1]	Mode
0	0	0	Idle	None	N/A
0	0	1	Stop	P	N/A
0	1	0	(Repeat) Start	S-A-D..(n)..D	Repeat n
0	1	1	(Repeat) Start-Stop	S-A-D..(n)..D-P	Repeat n
1	0	0	Idle	None	N/A
1	0	1	Stop	P	N/A
1	1	0	(Repeat) Start	S-A-D-D-D-....	Continuous
1	1	1	Reserved	None	N/A

[1] P = Stop condition; S = Start condition; A = Acknowledge bit; D = data

Table 31-17. I2C Module Operating Modes

FDF	MST	TRX	Operating Mode
0	0	x	Slave in non-FDF mode
0	1	0	Master receive in non-FDF mode
0	1	1	Master transmit in non-FDF mode
1	0	0	Slave receive in FDF mode
1	0	1	Slave transmit in FDF mode
1	1	0	Master receive in FDF mode
1	1	1	Master transmit in FDF mode

Table 31-18. Number of Bits Sent on Bus

BC2	BC1	BC0	Bits in FDF	Bits with ACK
0	0	0	8	9
0	0	1	NA (reserved)	NA (reserved)
0	1	0	2	3
0	1	1	3	4
1	0	0	4	5
1	0	1	5	6
1	1	0	6	7
1	1	1	7	8

31.6.11 I2C Interrupt Vector Register (I2CIVR)

The I2C interrupt vector register is a 16-bit memory-mapped register used to indicate the occurrence of an interrupt. Figure 31-24 and Table 31-19 describe this register.

Figure 31-24. I2C Interrupt Vector Register (I2CIVR) [offset = 28h]

15 12	11 8	7 3	2 0
Reserved	TESTMD	Reserved	INTCODE
R-0	R/W-0	R-0	R/WC-0

LEGEND: R/W = Read/Write; R = Read only; C = Clear; -*n* = value after reset

Table 31-19. I2C Interrupt Vector Register (I2CIVR) Field Descriptions

Bit	Field	Value	Description
15-12	Reserved	0	Reads return 0. Writes have no effect.
11-8	TESTMD	0-3h	Reserved for internal testing.
7-3	Reserved	0	Reads return 0. Writes have no effect.
2-0	INTCODE	0-3h	Interrupt Code Bits. This binary coded interrupt vector indicates which interrupt has occurred. If there is more than one interrupt pending, reading I2CIVR provides the vector for the highest priority interrupt that is pending. Reading the I2CIVR will clear the corresponding flags in I2CSTR for AL, NACK and SCD as long as those interrupts are enabled. A new interrupt will be generated for each pending source. Reading I2CIVR will clear the INTCODE for AL, NACK, SCD, AAS, RXRDY and TXRDY. Reading I2CIVR **will not** clear the INTCODE for ARDY. The INTCODE for certains codes can also be cleared by either writing a 1 to the corresponding interrupt flag bits in I2CSTR, or by reading and writing to the receive or transmit registers. See Section 31.6.3 for more details. Users must read (clear) the I2CIVR before doing another start otherwise the I2CIVR could contain incorrect (old interrupt flag) value.

Table 31-20. Interrupt Codes for INTCODE Bits

Code	INTCODE(2-0)	Interrupt Occurred
00h	000	None
01h	001 (highest priority)	Arbitration lost (AL)
02h	010	No acknowledgement (NACK)
03h	011	Receive access ready (ARDY)
04h	100	Receive data ready (RXRDY)
05h	101	Transmit data ready (TXRDY)
06h	110	Stop condition detection (SCD)
07h	111 (lowest priority)	Address as slave (AAS)

31.6.12 I2C Extended Mode Register (I2CEMDR)

The I2C extended mode register is a 16-bit memory-mapped register that contains additional mode select bits. Figure 31-25 and Table 31-21 describe this register.

Figure 31-25. I2C Extended Mode Register (I2CEMDR) [offset = 2Ch]

15 2	1	0
Reserved	IGNACK	BCM
R-0	R/W-0	R/W-1

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 31-21. I2C Extended Mode Register (I2CEMDR) Field Descriptions

Bit	Field	Value	Description
15-2	Reserved	0	Reads return 0. Writes have no effect.
1	IGNACK		Ignore NACK mode.
		0	The master transmitter will operate normally, discontinue the data transfer, and set the ARDY and NACK status bits when a NACK signal is received from the slave.
		1	The master transmitter will ignore a NACK received from the slave.
0	BCM		Backwards compatibility mode. When set to 1, the I2C is compatible with previous versions of the I2C. This means the TXRDY interrupt is generated in slave-transmit mode when TXRDY is set and the I2C needs more data to transmit. This behavior causes an extra TXRDY interrupt to be generated because the I2C recognizes the end of transfer after generating an interrupt for the next byte of data. When BCM is 0, the TXRDY interrupt in slave-transmit mode is generated when XSMT = 1. In this case, the I2C generates an interrupt for the next byte after receiving the ACK from previous data. The setting of this bit only applies to slave transmit mode.
		0	The I2C is not in compatibility mode.
		1	The I2C is in compatibility mode.

31.6.13 I2C Prescale Register (I2CPSC)

The I2C prescaler register is a 16-bit memory-mapped register used for dividing down the VBUS_CLK to obtain a module clock frequency between 6.7 MHz and 13.3 MHz. Figure 31-26 and Table 31-22 describe this register.

Figure 31-26. I2C Prescale Register (I2CPSC) [offset = 30h]

15 8	7 0
Reserved	PSC
R-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 31-22. I2C Prescale Register (I2CPSC) Field Descriptions

Bit	Field	Value	Description
15-8	Reserved	0	Reads return 0. Writes have no effect.
7-0	PSC	0-FFh	Prescale 8-bit prescaler to divide down the VBUS clock to obtain the I2C module clock. This register must be initialized while the I2C is still in reset (nIRS = 0). The value takes effect on the rising edge of nIRS. See Section 31.1.3 for more information.

31.6.14 I2C Peripheral ID Register 1 (I2CPID1)

Figure 31-27 and Table 31-23 describe this register.

Figure 31-27. I2C Peripheral ID Register 1 (I2CPID1) [offset = 34h]

15 8	7 0
CLASS	REVISION
R-1	R-46h

LEGEND: R = Read only; -*n* = value after reset

Table 31-23. I2C Peripheral ID Register 1 (I2CPID1) Field Descriptions

Bit	Field	Value	Description
15-8	CLASS	0-FFh	Peripheral class. These bits identify the class of peripheral.
7-0	REVISION	0-FFh	Revision level of the I2C. These bits identify the revision level of the I2C.

31.6.15 I2C Peripheral ID Register 2 (I2CPID2)

Figure 31-28 and Table 31-24 describe this register.

Figure 31-28. I2C Peripheral ID Register 2 (I2CPID2) [offset = 38h]

15 8	7 0
Reserved	TYPE
R-0	R-5h

LEGEND: R = Read only; -*n* = value after reset

Table 31-24. I2C Peripheral ID Register 2 (I2CPID2) Field Descriptions

Bit	Field	Value	Description
15-8	Reserved	0	Reads return 0. Writes have no effect.
7-0	TYPE	0-FFh	Peripheral type. These bits identify the type of peripheral.

31.6.16 I2C DMA Control Register (I2CDMACR)

This register contains the transmit and receive DMA enable bits. Figure 31-29 and Table 31-25 describe this register.

Figure 31-29. I2C DMA Control Register (I2CDMACR) [offset = 3Ch]

15 2	1	0
Reserved	TXDMAEN	RXDMAEN
R-0	R/W-1	R/W-1

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 31-25. I2C DMA Control Register (I2CDMACR) Field Descriptions

Bit	Field	Value	Description
15-2	Reserved	0	Reads return 0. Writes have no effect.
1	TXDMAEN		Transmitter DMA enable.
			This bit controls the transmit DMA event pin to the system. When this bit is a 1, the DMA transmit event is enabled and the DMA can occur. When this bit is a 0, the DMA transmit event is disabled.
			Writing a 1 to this bit will send a TXDMA request to the DMA module, if PINFUNC is also cleared to 0.
		0	The transmit DMA is disabled.
		1	The transmit DMA is enabled.
0	RXDMAEN		Receive DMA enable.
			This bit controls the receive DMA event pin to the system. When this bit is 1, the DMA receive event is enabled and the DMA can occur. When this bit is a 0, the DMA receive event is disabled.
		0	The receive DMA is disabled.
		1	The receive DMA is enabled.

31.6.17 I2C Pin Function Register (I2CPFNC)

Figure 31-30 and Table 31-26 describe this register.

Figure 31-30. I2C Pin Function Register (I2CPFNC) [offset = 48h]

15 1	0
Reserved	PINFUNC
R-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 31-26. I2C Pin Function Register (I2CPFNC) Field Descriptions

Bit	Field	Value	Description
15-1	Reserved	0	Reads return 0. Writes have no effect.
0	PINFUNC		SDA and SCL pin function.
			This bit controls whether the SDA and SCL pins function as I2C pins or as I/O pins.
		0	SDA and SCL pins function as I2C pins.
		1	SDA and SCL pins function as I/O pins.

31.6.18 I2C Pin Direction Register (I2CPDIR)

This register is used to independently configure each I2C pin, when configured as a general-purpose I/O, as either an input or output. Figure 31-31 and Table 31-27 describe this register.

Figure 31-31. I2C Pin Direction Register (I2CPDIR) [offset = 4Ch]

15 … 2	1	0
Reserved	SDADIR	SCLDIR
R-0	R/W-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 31-27. I2C Pin Direction Register (I2CPDIR) Field Descriptions

Bit	Field	Value	Description
15-2	Reserved	0	Reads return 0. Writes have no effect.
1	SDADIR		SDA pin direction. This bit controls the direction of the I2C SDA pin when configured as a GPIO.
		0	SDA pin functions as an input.
		1	SDA pin functions as an output.
0	SCLDIR		SCL pin direction. This bit controls the direction of the I2C SCL pin when configured as a GPIO.
		0	SCL pin functions as an input.
		1	SCL pin functions as an output.

31.6.19 I2C Data Input Register (I2CDIN)

Figure 31-32 and Table 31-28 describe this register.

Figure 31-32. I2C Data Input Register (I2CDIN) [offset = 50h]

15 … 2	1	0
Reserved	SDAIN	SCLIN
R-0	R-X	R-X

LEGEND: R = Read only; -X = value is indeterminate; -*n* = value after reset

Table 31-28. I2C Data Input Register (I2CDIN) Field Descriptions

Bit	Field	Value	Description
15-2	Reserved	0	Reads return 0. Writes have no effect.
1	SDAIN		Serial data in. The value of this bit reflects the value on the SDA pin.
0	SCLIN		Serial clock data in. The value of this bit reflects the value on the SCL pin.

31.6.20 I2C Data Output Register (I2CDOUT)

This register contains the values sent to the I2C pins. Figure 31-33 and Table 31-29 describe this register.

Figure 31-33. I2C Data Output Register (I2CDOUT) [offset 0x54]

15 - 2	1	0
Reserved	SDAOUT	SCLOUT
R-0	R/W-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 31-29. I2C Data Output Register (I2CDOUT) Field Descriptions

Bit	Field	Value	Description
15-2	Reserved	0	Reads return 0. Writes have no effect.
1	SDAOUT		SDA data output. This function is only active if the SDA pin is configured as an I/O pin with PINFUNC = 1. This bit contains the value sent to the SDA pin.
		0	The pin is driven low.
		1	The pin is driven high.
0	SCLOUT		SCL data output. This function is only active if the SCL pin is configured as an I/O pin with PINFUNC = 1. This bit contains the value sent to the SCL pin.
		0	The pin is driven low.
		1	The pin is driven high.

31.6.21 I2C Data Set Register (I2CDSET)

The I2CDSET register is an alias of the I2CDOUT register. Figure 31-34 and Table 31-30 describe this register.

Figure 31-34. I2C Data Set Register (I2CDSET) [offset = 58h]

15 - 2	1	0
Reserved	SDASET	SCLSET
R-0	R/W-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 31-30. I2C Data Set Register (I2CDSET) Field Description

Bit	Field	Value	Description
15-2	Reserved	0	Reads return 0. Writes have no effect.
1	SDASET		Serial data set. This bit is used to set the SDA GPIO pin.
		0	Read: Reads return value of SDAOUT. Write: No effect.
		1	Read: Reads return value of SDAOUT. Write: SDAOUT is set to logic high (1).
0	SCLSET		Serial clock set. This bit is used to set the SCL GPIO pin.
		0	Read: Reads return value of SCLOUT. Write: No effect.
		1	Read: Reads return value of SCLOUT. Write: SCLOUT is set to logic high (1).

31.6.22 I2C Data Clear Register (I2CDCLR)

The I2CDCLR register is an alias of the I2CDOUT register. Figure 31-35 and Table 31-31 describe this register.

Figure 31-35. I2C Data Clear Register (I2CDCLR) [offset = 5Ch]

15	2	1	0
Reserved		SDACLR	SCLCLR
R-0		R/W-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 31-31. I2C Data Clear Register (I2CDSET) Field Descriptions

Bit	Field	Value	Description
15-2	Reserved	0	Reads return 0. Writes have no effect.
1	SDACLR		Serial data clear. This bit is used to clear the SDA GPIO pin.
		0	Read: Reads return value of SDAOUT. Write: No effect.
		1	Read: Reads return value of SDAOUT. Write: SDAOUT is cleared to logic low (0).
0	SCLCLR		Serial clock clear. This bit is used to clear the SCL GPIO pin.
		0	Read: Reads return value of SCLOUT. Write: No effect.
		1	Read: Reads return value of SCLOUT. Write: SCLOUT is cleared to logic low (0).

31.6.23 I2C Pin Open Drain Register (I2CPDR)

Figure 31-36 and Table 31-32 describe this register.

Figure 31-36. I2C Pin Open Drain Register (I2CPDR) [offset = 60h]

15	2	1	0
Reserved		SDAPDR	SCLPDR
R-0		R/W-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 31-32. I2C Pin Open Drain Register (I2CPDR) Field Descriptions

Bit	Field	Value	Description
15-2	Reserved	0	Reads return 0. Writes have no effect.
1	SDAPDR		SDA pin open drain enable.
		0	The open drain function is enabled (the output voltage is V_{OL} or lower if SDAOUT = 0 and high-impedance if SDAOUT = 1).
		1	The open drain function is disabled (output voltage is V_{OL} or lower if SDAOUT = 0; V_{OH} or higher if SDAOUT = 1).
0	SCLPDR		SCL pin open drain enable.
		0	The open drain function is enabled (the output voltage is V_{OL} or lower if SCLOUT = 0 and high-impedance if SCLOUT = 1).
		1	The open drain function is disabled (output voltage is V_{OL} or lower if SCLOUT = 0; V_{OH} or higher if SCLOUT = 1).

31.6.24 I2C Pull Disable Register (I2CPDIS)

Values in the I2CPDIS register enable or disable the pull control capability of the pins. Figure 31-37 and Table 31-33 describe this register.

Figure 31-37. I2C Pull Disable Register (I2CPDIS) [offset = 64h]

15 2	1	0
Reserved	SDAPDIS	SCLPDIS
R-0	R/W-1	R/W-1

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 31-33. I2C Pull Disable Register (I2CPDIS) Field Descriptions

Bit	Field	Value	Description
15-2	Reserved	0	Reads return 0. Writes have no effect.
1	SDAPDIS		SDA pull disable.
		0	The pull function is enabled.
		1	The pull function is disabled.
0	SCLPDIS		SCL pull disable.
		0	The pull function is enabled.
		1	The pull function is disabled.

31.6.25 I2C Pull Select Register (I2CPSEL)

Values in the I2CPSEL register select the pull up or pull down functions of the corresponding pins. Figure 31-38 and Table 31-34 describe this register.

Figure 31-38. I2C Pull Select Register (I2CPSEL) [offset = 68h]

15 2	1	0
Reserved	SDAPSEL	SCLPSEL
R-0	R/W-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 31-34. I2C Pull Select Register (I2CPSEL) Field Descriptions

Bit	Field	Value	Description
15-2	Reserved	0	Reads return 0. Writes have no effect.
1	SDAPSEL		SDA pull select enable.
		0	The pull down function is enabled.
		1	The pull up function is enabled.
0	SCLPSEL		SCL pull select enable.
		0	The pull down function is enabled.
		1	The pull up function is enabled.

31.6.25.1 Summary

The behavior of the input buffer, output buffer, and the pull control is summarized in Table 31-35.

Table 31-35. Input Buffer, Output Buffer, and Pull Control Behavior as GPIO Pins

Device under Reset?	Pin Direction (DIR) [1][2]	Pull Disable (PULDIS) [1][3]	Pull Select (PULSEL) [1][4]	Pull Control	Output Buffer	Input Buffer
Yes	X	X	X	Enabled	Disabled	Enabled
No	0	0	0	Pull down	Disabled	Enabled
No	0	0	1	Pull up	Disabled	Enabled
No	0	1	0	Disabled	Disabled	Enabled
No	0	1	1	Disabled	Disabled	Enabled
No	1	X	X	Disabled	Enabled	Enabled

[1] X = Don't care

[2] DIR = 0 for input, = 1 for output

[3] PULDIS = 0 for enabling pull control
= 1 for disabling pull control

[4] PULSEL= 0 for pull-down functionality
= 1 for pull-up functionality

31.6.26 *I2C Pins Slew Rate Select Register (I2CSRS)*

This register controls the slew rate of the signal on the I2C pins. Figure 31-39 and Table 31-36 describe this register.

Figure 31-39. I2C Pins Slew Rate Select Register (I2CSRS) [offset = 6Ch]

15 2	1	0
Reserved	SDASRS	SCLSRS
R-0	R/W-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

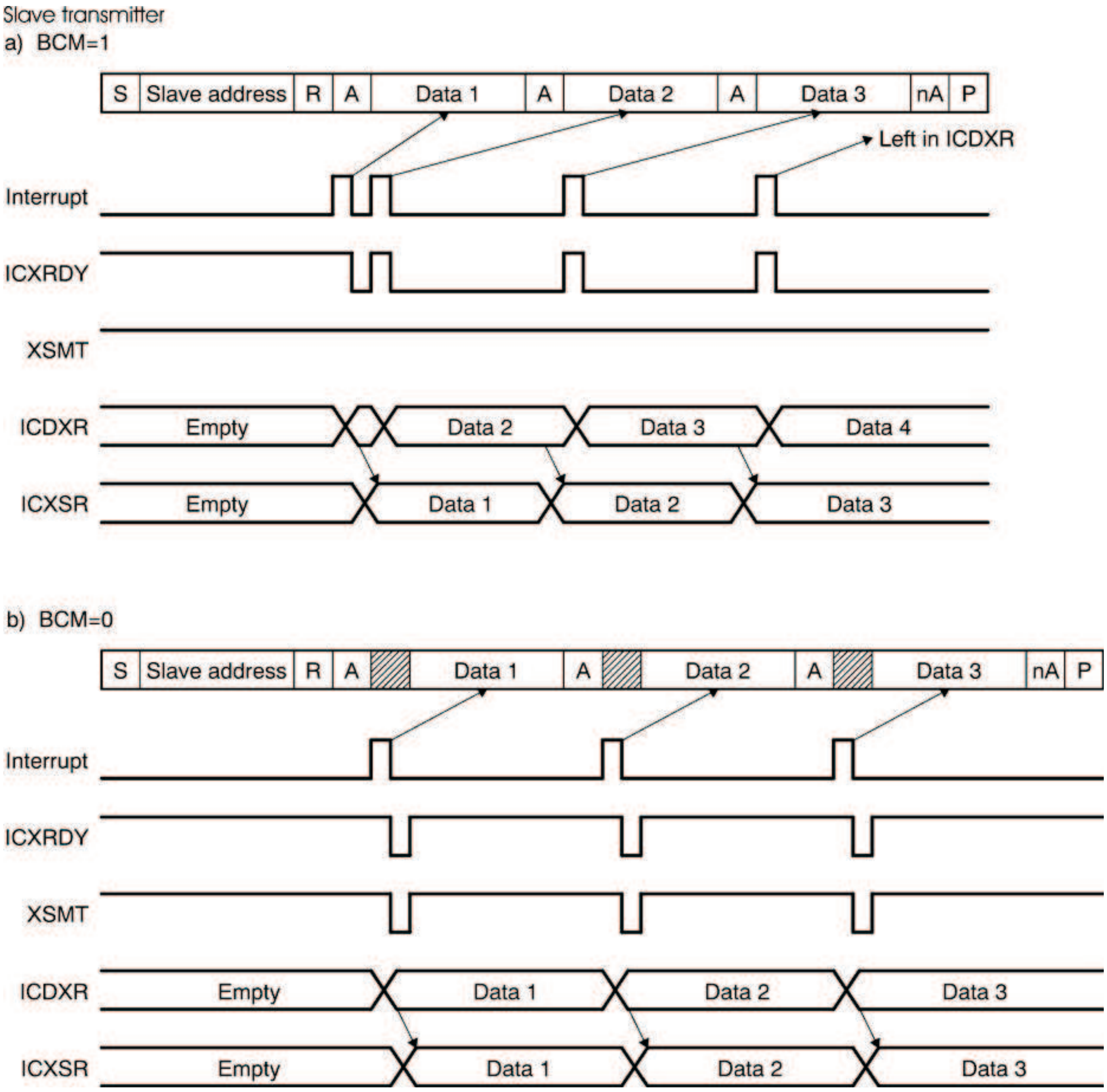
Table 31-36. I2C Pins Slew Rate Select Register (I2CSRS) Field Descriptions

Bit	Field	Value	Description
15-2	Reserved	0	Reads return 0. Writes have no effect.
1	SDASRS		SDA slew rate select.
		0	The slow buffer is selected.
		1	The normal buffer is selected.
0	SCLSRS		SCL slew rate select.
		0	The slow buffer is selected.
		1	The normal buffer is selected.

31.7 Sample Waveforms

Figure 31-40 provides waveforms to illustrate the difference between normal operation and backward compatibility mode.

Figure 31-40. Difference between Normal Operation and Backward Compatibility Mode



Chapter 32
SPNU563A – March 2018

EMAC/MDIO Module

This chapter describes the Ethernet Media Access Controller (EMAC) and physical layer (PHY) device Management Data Input/Output (MDIO) module.

32.1 Introduction

This document provides a functional description of the Ethernet Media Access Controller (EMAC) and physical layer (PHY) device Management Data Input/Output (MDIO) module integrated in the microcontroller. Included are the features of the EMAC and MDIO modules, a discussion of their architecture and operation, how these modules connect to the outside world, and a description of the registers for each module.

The EMAC controls the flow of packet data from the system to the PHY. The MDIO module controls PHY configuration and status monitoring.

Both the EMAC and the MDIO modules interface to the system core through a custom interface that allows efficient data transmission and reception. This custom interface is referred to as the EMAC control module and is considered integral to the EMAC/MDIO peripheral.

32.1.1 Purpose of the Peripheral

The EMAC module is used to move data between the device and another host connected to the same network, in compliance with the Ethernet protocol.

32.1.2 Features

The EMAC/MDIO has the following features:

- Synchronous 10/100 Mbps operation.
- Standard Media Independent Interface (MII) and/or Reduced Media Independent Interface (RMII) to physical layer device (PHY).
- EMAC acts as DMA master to either internal or external device memory space.
- Eight receive channels with VLAN tag discrimination for receive quality-of-service (QOS) support.
- Eight transmit channels with round-robin or fixed priority for transmit quality-of-service (QOS) support.
- Ether-Stats and 802.3-Stats statistics gathering.
- Transmit CRC generation selectable on a per channel basis.
- Broadcast frames selection for reception on a single channel.
- Multicast frames selection for reception on a single channel.
- Promiscuous receive mode frames selection for reception on a single channel (all frames, all good frames, short frames, error frames).
- Hardware flow control.
- 8k-byte local EMAC descriptor memory that allows the peripheral to operate on descriptors without affecting the CPU. The descriptor memory holds enough information to transfer up to 512 Ethernet packets without CPU intervention. (This memory is also known as CPPI RAM.)
- Programmable interrupt logic permits the software driver to restrict the generation of back-to-back interrupts, which allows more work to be performed in a single call to the interrupt service routine.

32.1.3 Functional Block Diagram

Figure 32-1 shows the three main functional modules of the EMAC/MDIO peripheral:

- EMAC control module
- EMAC module
- MDIO module

The EMAC control module is the main interface between the device core processor to the EMAC and MDIO modules. The EMAC control module controls device interrupts and incorporates an 8k-byte internal RAM to hold EMAC buffer descriptors (also known as CPPI RAM).

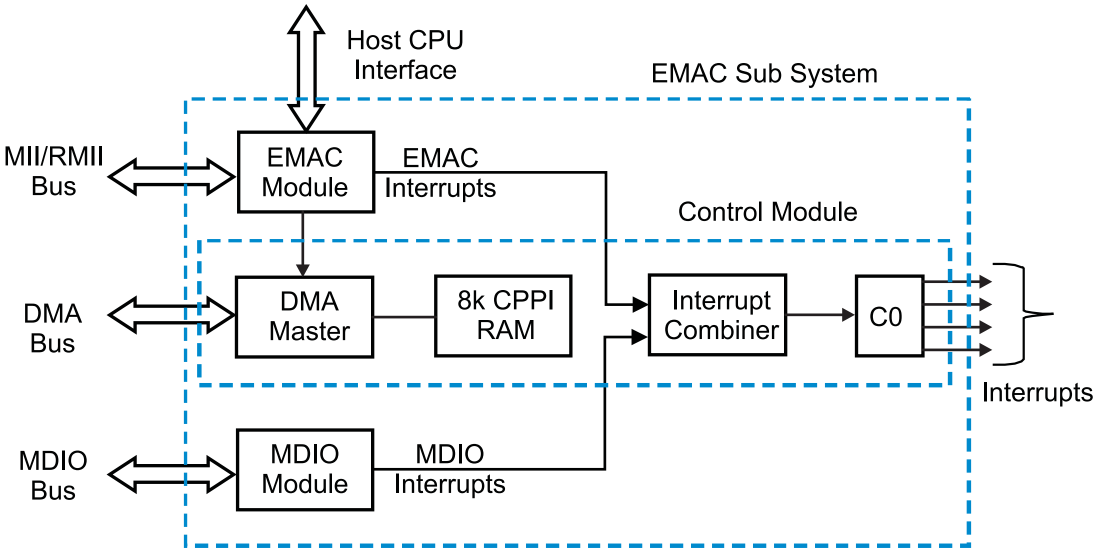
The MDIO module implements the 802.3 serial management interface to interrogate and control up to 32 Ethernet PHYs connected to the device by using a shared two-wire bus. Host software uses the MDIO module to configure the autonegotiation parameters of each PHY attached to the EMAC, retrieve the negotiation results, and configure required parameters in the EMAC module for correct operation. The module is designed to allow almost transparent operation of the MDIO interface, with very little maintenance from the core processor.

The EMAC module provides an efficient interface between the processor and the network. The EMAC on this device supports 10Base-T (10 Mbits/sec) and 100BaseTX (100 Mbits/sec), half-duplex and full-duplex mode, and hardware flow control and quality-of-service (QOS) support.

Figure 32-1 shows the main interface between the EMAC control module and the CPU. The following connections are made to the device core:

- The DMA bus connection from the EMAC control module allows the EMAC module to read and write both internal and external memory through the DMA memory transfer controller.
- The EMAC control, EMAC, and MDIO modules all have control registers. These registers are memory-mapped into device memory space. Along with these registers, the control module's internal CPPI RAM is mapped into this same range.
- The EMAC and MDIO interrupts are combined into four interrupt signals within the control module. The Vectored Interrupt Manager (VIM) receives all four interrupt signals from the combiner and submits these interrupt requests to the CPU.

Figure 32-1. EMAC and MDIO Block Diagram



32.1.4 Industry Standard(s) Compliance Statement

The EMAC peripheral conforms to the IEEE 802.3 standard, describing the Carrier Sense Multiple Access with Collision Detection (CSMA/CD) Access Method and Physical Layer specifications. The IEEE 802.3 standard has also been adopted by ISO/IEC and re-designated as ISO/IEC 8802-3:2000(E).

However, the EMAC deviates from the standard in the way it handles transmit underflow errors. The EMAC MII interface does not use the Transmit Coding Error signal MTXER. Instead of driving the error pin when an underflow condition occurs on a transmitted frame, the EMAC intentionally generates an incorrect checksum by inverting the frame CRC, so that the transmitted frame is detected as an error by the network.

32.2 Architecture

This section discusses the architecture and basic function of the EMAC peripheral.

32.2.1 Clock Control

All internal EMAC logic is clocked synchronously on the VCLKA4_DIVR domain. Please refer to the *Architecture* chapter for more details.

The MDIO clock is based on a divide-down of the VCLK3 peripheral bus clock and is specified to run up to 2.5 MHz (although typical operation would be 1.0 MHz). Because the VCLK3 peripheral clock frequency is configurable, the application software or driver must control the divide-down value.

The transmit and receive clock sources are provided by the external PHY to the MII_TXCLK and MII_RXCLK pins or to the RMII reference clock pin. Data is transmitted and received with respect to the reference clocks of the interface pins.

The MII interface frequencies for the transmit and receive clocks are fixed by the IEEE 802.3 specification as:

- 2.5 MHz at 10 Mbps
- 25 MHz at 100 Mbps

The RMII interface frequency for the transmit and receive clocks are fixed at 50 MHz for both 10 Mbps and 100 Mbps.

32.2.2 Memory Map

The EMAC peripheral includes internal memory that is used to hold buffer descriptions of the Ethernet packets to be received and transmitted. This internal RAM is 2K × 32 bits in size. Data can be written to and read from the EMAC internal memory by either the EMAC or the CPU. It is used to store buffer descriptors that are 4-words (16-bytes) deep. This 8K local memory holds enough information to transfer up to 512 Ethernet packets without CPU intervention. This EMAC RAM is also referred to as the CPPI buffer descriptor memory because it complies with the Communications Port Programming Interface (CPPI) v3.0 standard.

The packet buffer descriptors can also be placed in other on- and off-chip memories such as the CPU RAM. There are some tradeoffs in terms of interconnect bandwidth when descriptors are placed in the CPU RAM, versus when they are placed in the EMAC's internal memory. In general, the EMAC throughput is better when the descriptors are placed in the local EMAC CPPI RAM.

32.2.3 Signal Descriptions

The microcontrollers support both the MII and the RMII interfaces. Only one of these two interfaces can be used at a time. A separate control register in the I/O Multiplexing Module (IOMM) allows the application to indicate the actual interface being used. This is the bit 24 of the PINMMR160 control register. This bit is set by default and selects the RMII interface. The application can select the MII interface by clearing this bit. Please refer to the *I/O Multiplexing and Control Module (IOMM)* chapter for more details on the procedure to configure the PINMMR registers.

Each of the EMAC and MDIO signals for the MII and RMII interfaces are multiplexed with other I/O functions on this microcontroller. Please refer to Section 32.2.4 for information on configuration of the multiplexing control registers to enable the MII / RMII connections to these I/Os.

32.2.3.1 Media Independent Interface (MII) Connections

Figure 32-2 shows a device with integrated EMAC and MDIO interfaced via a MII connection in a typical system. The EMAC module does not include a transmit error (MTXER) pin. In the case of transmit error, CRC inversion is used to negate the validity of the transmitted frame.

The individual EMAC and MDIO signals for the MII interface are summarized in Table 32-1. For more information, refer to either the IEEE 802.3 standard or ISO/IEC 8802-3:2000(E).

Figure 32-2. Ethernet Configuration—MII Connections

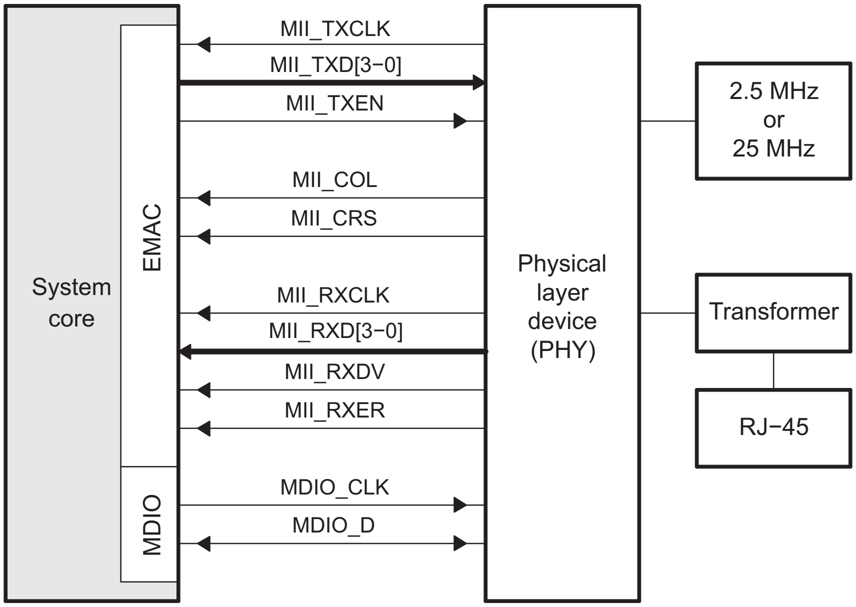


Table 32-1. EMAC and MDIO Signals for MII Interface

Signal	Type	Description
MII_TXCLK	I	Transmit clock (MII_TXCLK). The transmit clock is a continuous clock that provides the timing reference for transmit operations. The MII_TXD and MII_TXEN signals are tied to this clock. The clock is generated by the PHY and is 2.5 MHz at 10 Mbps operation and 25 MHz at 100 Mbps operation.
MII_TXD[3-0]	O	Transmit data (MII_TXD). The transmit data pins are a collection of 4 data signals comprising 4 bits of data. MTDX0 is the least-significant bit (LSB). The signals are synchronized by MII_TXCLK and valid only when MII_TXEN is asserted.
MII_TXEN	O	Transmit enable (MII_TXEN). The transmit enable signal indicates that the MII_TXD pins are generating nibble data for use by the PHY. It is driven synchronously to MII_TXCLK.
MII_COL	I	Collision detected (MII_COL). In half-duplex operation, the MII_COL pin is asserted by the PHY when it detects a collision on the network. It remains asserted while the collision condition persists. This signal is not necessarily synchronous to MII_TXCLK nor MII_RXCLK. In full-duplex operation, the MII_COL pin is used for hardware transmit flow control. Asserting the MII_COL pin will stop packet transmissions; packets in the process of being transmitted when MII_COL is asserted will complete transmission. The MII_COL pin should be held low if hardware transmit flow control is not used.
MII_CRS	I	Carrier sense (MII_CRS). In half-duplex operation, the MII_CRS pin is asserted by the PHY when the network is not idle in either transmit or receive. The pin is deasserted when both transmit and receive are idle. This signal is not necessarily synchronous to MII_TXCLK nor MII_RXCLK. In full-duplex operation, the MII_CRS pin should be held low.
MII_RXCLK	I	Receive clock (MII_RXCLK). The receive clock is a continuous clock that provides the timing reference for receive operations. The MII_RXD, MII_RXDV, and MII_RXER signals are tied to this clock. The clock is generated by the PHY and is 2.5 MHz at 10 Mbps operation and 25 MHz at 100 Mbps operation.
MII_RXD[3-0]	I	Receive data (MII_RXD). The receive data pins are a collection of 4 data signals comprising 4 bits of data. MRDX0 is the least-significant bit (LSB). The signals are synchronized by MII_RXCLK and valid only when MII_RXDV is asserted.
MII_RXDV	I	Receive data valid (MII_RXDV). The receive data valid signal indicates that the MII_RXD pins are generating nibble data for use by the EMAC. It is driven synchronously to MII_RXCLK.
MII_RXER	I	Receive error (MII_RXER). The receive error signal is asserted for one or more MII_RXCLK periods to indicate that an error was detected in the received frame. This is meaningful only during data reception when MII_RXDV is active.
MDIO_CLK	O	Management data clock (MDIO_CLK). The MDIO data clock is sourced by the MDIO module on the system. It is used to synchronize MDIO data access operations done on the MDIO pin. The frequency of this clock is controlled by the CLKDIV bits in the MDIO control register (CONTROL).
MDIO_D	I/O	Management data input output (MDIO_D). The MDIO data pin drives PHY management data into and out of the PHY by way of an access frame consisting of start of frame, read/write indication, PHY address, register address, and data bit cycles. The MDIO_D pin acts as an output for all but the data bit cycles at which time it is an input for read operations.

NOTE: The MII interface of this device is bonded out to two different sets of package pins. In one set of package pins, the interface is multiplexed with other functions on this device. The application must configure the control registers in the I/O multiplexing module in order to enable the MII functionality on the corresponding I/Os. The IO pins to which the MII interface is brought are pin compatible with other devices in the family. Please refer to Section 32.2.4 for information. This device also has a second set of package pins that brings out the MII interface. The second set of package pins for the MII interface allows the MII interface to operate in parallel with other non-MII functions in the first set of package pins. Please see the device datasheet terminal function table for detail.

32.2.3.2 Reduced Media Independent Interface (RMII) Connections

Figure 32-3 shows a device with integrated EMAC and MDIO interfaced via a RMII connection in a typical system.

The individual EMAC and MDIO signals for the RMII interface are summarized in Table 32-2. For more information, refer to either the IEEE 802.3 standard or ISO/IEC 8802-3:2000(E).

Figure 32-3. Ethernet Configuration—RMII Connections

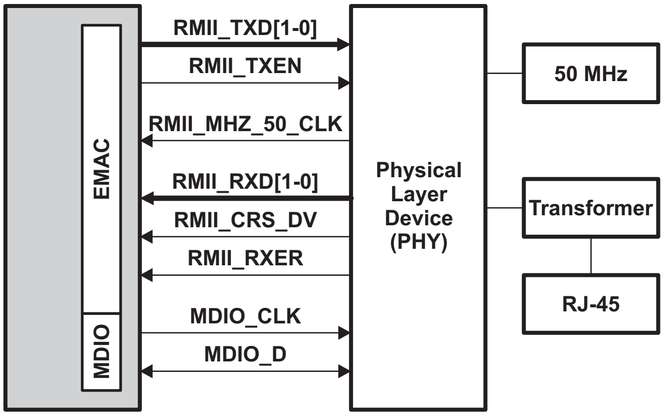


Table 32-2. EMAC and MDIO Signals for RMII Interface

Signal	Type	Description
RMII_TXD[1-0]	O	Transmit data (RMII_TXD). The transmit data pins are a collection of 2 bits of data. RMTDX0 is the least-significant bit (LSB). The signals are synchronized by RMII_MHZ_50_CLK and valid only when RMII_TXEN is asserted.
RMII_TXEN	O	Transmit enable (RMII_TXEN). The transmit enable signal indicates that the RMII_TXD pins are generating data for use by the PHY. RMII_TXEN is synchronous to RMII_MHZ_50_CLK.
RMII_MHZ_50_CLK	I	RMII reference clock (RMII_MHZ_50_CLK). The reference clock is used to synchronize all RMII signals. RMII_MHZ_50_CLK must be continuous and fixed at 50 MHz.
RMII_RXD[1-0]	I	Receive data (RMII_RXD). The receive data pins are a collection of 2 bits of data. RMRDX0 is the least-significant bit (LSB). The signals are synchronized by RMII_MHZ_50_CLK and valid only when RMII_CRS_DV is asserted and RMII_RXER is deasserted.
RMII_CRS_DV	I	Carrier sense/receive data valid (RMII_CRS_DV). Multiplexed signal between carrier sense and receive data valid.
RMII_RXER	I	Receive error (RMII_RXER). The receive error signal is asserted to indicate that an error was detected in the received frame.
MDIO_CLK	O	Management data clock (MDIO_CLK). The MDIO data clock is sourced by the MDIO module on the system. It is used to synchronize MDIO data access operations done on the MDIO pin. The frequency of this clock is controlled by the CLKDIV bits in the MDIO control register (CONTROL).
MDIO_D	I/O	Management data input output (MDIO_D). The MDIO data pin drives PHY management data into and out of the PHY by way of an access frame consisting of start of frame, read/write indication, PHY address, register address, and data bit cycles. The MDIO_D pin acts as an output for all but the data bit cycles at which time it is an input for read operations.

32.2.4 MII / RMII Signal Multiplexing Control

In Each of the MII and RMII interface signals are multiplexed with other functions on this microcontroller. The application must configure the control registers in the I/O multiplexing module in order to enable the MII/RMII functionality on the corresponding I/Os. Table 32-3 shows the byte to be configured to enable the MDIO functions. Table 32-4 shows the byte to be configured to enable the MII or RMII functions. Please refer to the *I/O Multiplexing and Control Module (IOMM)* chapter for more details on the procedure to configure the PINMMR registers.

Table 32-3. MDIO Multiplexing Control

MDIO Signal Name	Control for Selecting EMAC / MDIO Signal
MDIO_CLK	PINMMR21[31:24] = 0b00000100
MDIO_D	PINMMR23 [7:0] = 0b00000100

Table 32-4. MII/RMII Multiplexing Control

MII / RMII Signal Name	Control for Selecting MII Signal	Control for Selecting RMII Signal
MII_TXCLK	PINMMR30 [23:16] = 0b00000010	-
MII_TXD[3]	PINMMR30 [7:0] = 0b00000100	-
MII_TXD[2]	PINMMR21 [15:8] = 0b00000100	-
MII_TXD[1] / RMII_TXD[1]	PINMMR26 [7:0] = 0b00000100	PINMMR26 [7:0] = 0b00001000
MII_TXD[0] / RMII_TXD[0]	PINMMR27 [7:0] = 0b00000100	PINMMR27 [7:0] = 0b00001000
MII_TXEN / RMII_TXEN	PINMMR24 [23:16] = 0b00000100	PINMMR24 [23:16] = 0b00001000
MII_COL	PINMMR21 [23:16] = 0b00000100	-
MII_CRS / RMII_CRSDV	PINMMR31 [7:0] = 0b00000100	PINMMR31 [7:0] = 0b00001000
MII_RXCLK / RMII_50MHZ_CLK	PINMMR34 [15:8] = 0b00000100	PINMMR34 [15:8] = 0b00001000
MII_RXD[3]	PINMMR25 [31:24] = 0b00000100	-
MII_RXD[2]	PINMMR22 [15:8] = 0b00000100	-
MII_RXD[1] / RMII_RXD[1]	PINMMR34 [7:0] = 0b00000010	PINMMR34 [7:0] = 0b00000100
MII_RXD[0] / RMII_RXD[0]	PINMMR33 [31:24] = 0b00000100	PINMMR33 [31:24] = 0b00001000
MII_RXDV	PINMMR34 23:16] = 0b00000100	-
MII_RXER / RMII_RXER	PINMMR0 [7:0] = 0b00000100	PINMMR0 [7:0] = 0b00001000

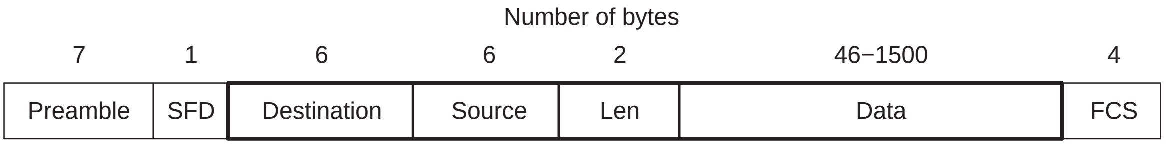
32.2.5 Ethernet Protocol Overview

A brief overview of the Ethernet protocol is given in the following subsections. See the IEEE 802.3 standard document for in-depth information on the Carrier Sense Multiple Access with Collision Detection (CSMA/CD) Access Method.

32.2.5.1 Ethernet Frame Format

All the Ethernet technologies use the same frame structure. The format of an Ethernet frame is shown in Figure 32-4 and described in Table 32-5. The Ethernet packet, which is the collection of bytes representing the data portion of a single Ethernet frame on the wire, is shown outlined in bold. The Ethernet frames are of variable lengths, with no frame smaller than 64 bytes or larger than RXMAXLEN bytes (header, data, and CRC).

Figure 32-4. Ethernet Frame Format



Number of bytes

7	1	6	6	2	46−1500	4
Preamble	SFD	Destination	Source	Len	Data	FCS

Legend: SFD=Start Frame Delimeter; FCS=Frame Check Sequence (CRC)

Table 32-5. Ethernet Frame Description

Field	Bytes	Description
Preamble	7	Preamble. These 7 bytes have a fixed value of 55h and serve to wake up the receiving EMAC ports and to synchronize their clocks to that of the sender's clock.
SFD	1	Start of Frame Delimiter. This field with a value of 5Dh immediately follows the preamble pattern and indicates the start of important data.
Destination	6	Destination address. This field contains the Ethernet MAC address of the EMAC port for which the frame is intended. It may be an individual or multicast (including broadcast) address. When the destination EMAC port receives an Ethernet frame with a destination address that does not match any of its MAC physical addresses, and no promiscuous, multicast or broadcast channel is enabled, it discards the frame.
Source	6	Source address. This field contains the MAC address of the Ethernet port that transmits the frame to the Local Area Network.
Len	2	Length/Type field. The length field indicates the number of EMAC client data bytes contained in the subsequent data field of the frame. This field can also be used to identify the type of data the frame is carrying.
Data	46 to (RXMAXLEN - 18)	Data field. This field carries the datagram containing the upper layer protocol frame, that is, IP layer datagram. The maximum transfer unit (MTU) of Ethernet is (RXMAXLEN - 18) bytes. This means that if the upper layer protocol datagram exceeds (RXMAXLEN - 18) bytes, then the host has to fragment the datagram and send it in multiple Ethernet packets. The minimum size of the data field is 46 bytes. This means that if the upper layer datagram is less then 46 bytes, the data field has to be extended to 46 bytes by appending extra bits after the data field, but prior to calculating and appending the FCS.
FCS	4	Frame Check Sequence. A cyclic redundancy check (CRC) is used by the transmit and receive algorithms to generate a CRC value for the FCS field. The frame check sequence covers the 60 to 1514 bytes of the packet data. Note that this 4-byte field may or may not be included as part of the packet data, depending on how the EMAC is configured.

32.2.5.2 Ethernet's Multiple Access Protocol

Nodes in an Ethernet Local Area Network are interconnected by a broadcast channel -- when an EMAC port transmits a frame, all the adapters on the local network receive the frame. Carrier Sense Multiple Access with Collision Detection (CSMA/CD) algorithms are used when the EMAC operates in half-duplex mode. When operating in full-duplex mode, there is no contention for use of a shared medium because there are exactly two ports on the local network.

Each port runs the CSMA/CD protocol without explicit coordination with the other ports on the Ethernet network. Within a specific port, the CSMA/CD protocol works as follows:

1. The port obtains data from upper layer protocols at its node, prepares an Ethernet frame, and puts the frame in a buffer.
2. If the port senses that the medium is idle, it starts to transmit the frame. If the port senses that the transmission medium is busy, it waits until it no longer senses energy (plus an Inter-Packet Gap time) and then starts to transmit the frame.
3. While transmitting, the port monitors for the presence of signal energy coming from other ports. If the port transmits the entire frame without detecting signal energy from other Ethernet devices, the port is done with the frame.
4. If the port detects signal energy from other ports while transmitting, it stops transmitting its frame and instead transmits a 48-bit jam signal.
5. After transmitting the jam signal, the port enters an exponential backoff phase. If a data frame encounters back-to-back collisions, the port chooses a random value that is dependent on the number of collisions. The port then waits an amount of time that is a multiple of this random value and returns to step 2.

32.2.6 Programming Interface

32.2.6.1 Packet Buffer Descriptors

The buffer descriptor is a central part of the EMAC module and is how the application software describes Ethernet packets to be sent and empty buffers to be filled with incoming packet data. The basic descriptor format is shown in Figure 32-5 and described in Table 32-6.

For example, consider three packets to be transmitted: Packet A is a single fragment (60 bytes), Packet B is fragmented over three buffers (1514 bytes total), and Packet C is a single fragment (1514 bytes). The linked list of descriptors to describe these three packets is shown in Figure 32-6.

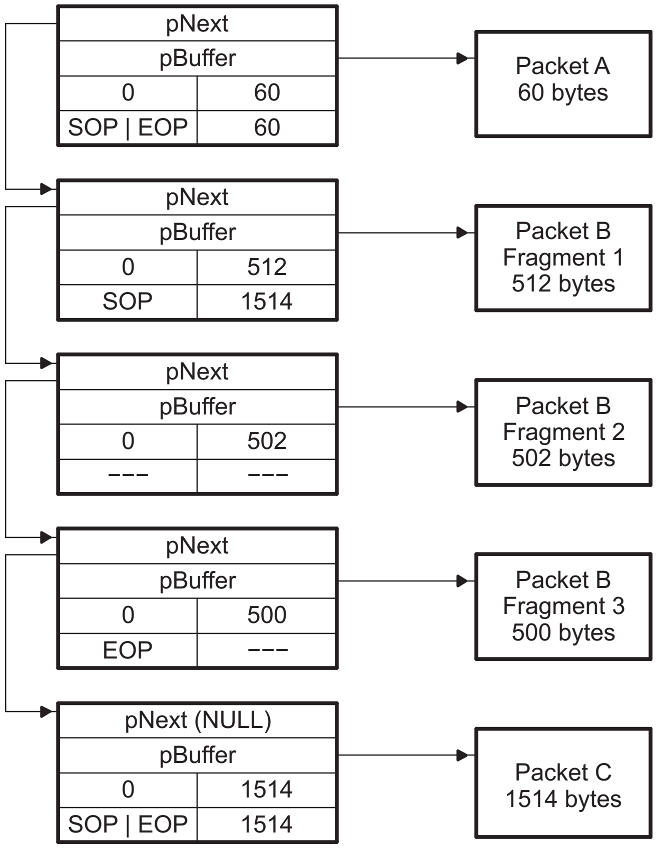
Figure 32-5. Basic Descriptor Format

Word Offset	Bit Fields	
	31 16	15 0
0	Next Descriptor Pointer	
1	Buffer Pointer	
2	Buffer Offset	Buffer Length
3	Flags	Packet Length

Table 32-6. Basic Descriptor Description

Word Offset	Field	Field Description
0	Next Descriptor Pointer	The next descriptor pointer is used to create a single linked list of descriptors. Each descriptor describes a packet or a packet fragment. When a descriptor points to a single buffer packet or the first fragment of a packet, the start of packet (SOP) flag is set in the flags field. When a descriptor points to a single buffer packet or the last fragment of a packet, the end of packet (EOP) flag is set. When a packet is fragmented, each fragment must have its own descriptor and appear sequentially in the descriptor linked list.
1	Buffer Pointer	The buffer pointer refers to the actual memory buffer that contains packet data during transmit operations, or is an empty buffer ready to receive packet data during receive operations.
2	Buffer Offset	The buffer offset is the offset from the start of the packet buffer to the first byte of valid data. This field only has meaning when the buffer descriptor points to a buffer that actually contains data.
	Buffer Length	The buffer length is the actual number of valid packet data bytes stored in the buffer. If the buffer is empty and waiting to receive data, this field represents the size of the empty buffer.
3	Flags	The flags field contains more information about the buffer, such as, is it the first fragment in a packet (SOP), the last fragment in a packet (EOP), or contains an entire contiguous Ethernet packet (both SOP and EOP). The flags are described in Section 32.2.6.4 and Section 32.2.6.5.
	Packet Length	The packet length only has meaning for buffers that both contain data and are the start of a new packet (SOP). In the case of SOP descriptors, the packet length field contains the length of the entire Ethernet packet, regardless if it is contained in a single buffer or fragmented over several buffers.

Figure 32-6. Typical Descriptor Linked List



32.2.6.2 Transmit and Receive Descriptor Queues

The EMAC module processes descriptors in linked lists as discussed in Section 32.2.6.1. The lists used by the EMAC are maintained by the application software through the use of the head descriptor pointer registers (HDP). The EMAC supports eight channels for transmit and eight channels for receive. The corresponding head descriptor pointers are:

- TX*n*HDP - Transmit Channel *n* DMA Head Descriptor Pointer Register
- RX*n*HDP - Receive Channel *n* DMA Head Descriptor Pointer Register

After an EMAC reset and before enabling the EMAC for send and receive, all 16 head descriptor pointer registers must be initialized to 0.

The EMAC uses a simple system to determine if a descriptor is currently owned by the EMAC or by the application software. There is a flag in the buffer descriptor flags called OWNER. When this flag is set, the packet that is referenced by the descriptor is considered to be owned by the EMAC. Note that ownership is done on a packet based granularity, not on descriptor granularity, so only SOP descriptors make use of the OWNER flag. As packets are processed, the EMAC patches the SOP descriptor of the corresponding packet and clears the OWNER flag. This is an indication that the EMAC has finished processing all descriptors up to and including the first with the EOP flag set, indicating the end of the packet (note this may only be one descriptor with both the SOP and EOP flags set).

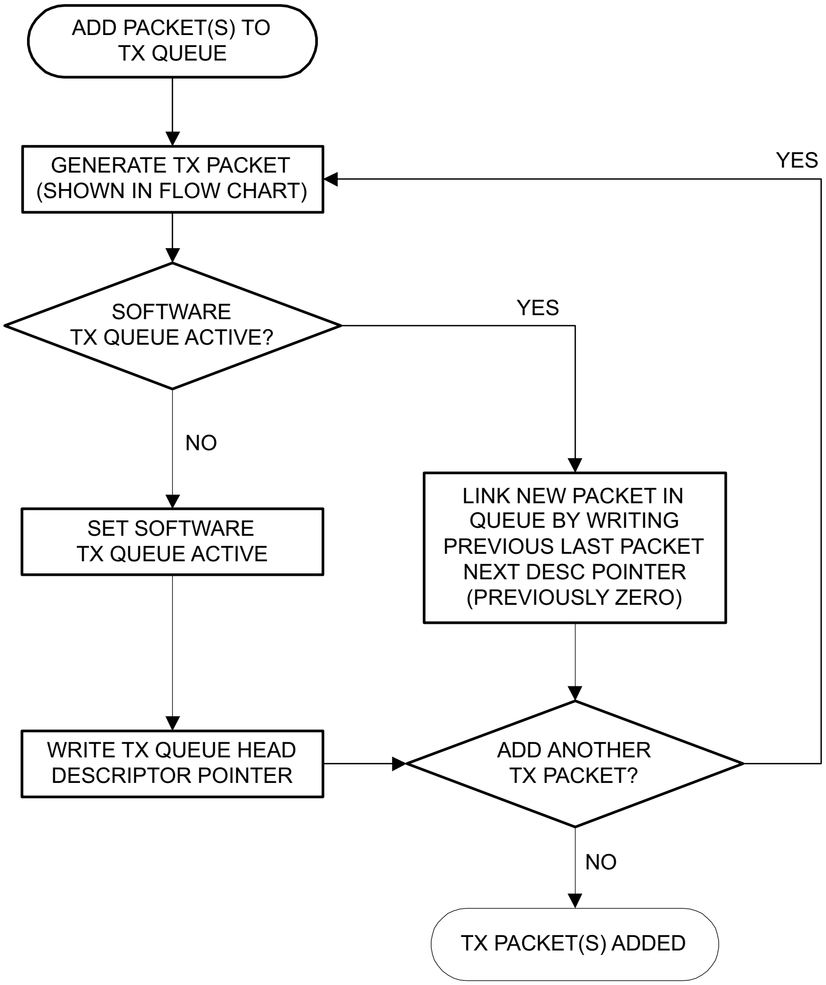
To add a descriptor or a linked list of descriptors to an EMAC descriptor queue for the first time, the software application simply writes the pointer to the descriptor or first descriptor of a list to the corresponding HDP register. Note that the last descriptor in the list must have its “next” pointer cleared to 0. This is the only way the EMAC has of detecting the end of the list. Therefore, in the case where only a single descriptor is added, its “next descriptor” pointer must be initialized to 0.

The HDP must never be written to while a list is active. To add additional descriptors to a descriptor list already owned by the EMAC, the NULL “next” pointer of the last descriptor of the previous list is patched with a pointer to the first descriptor of the new list. The list of new descriptors to be appended to the existing list must itself be NULL terminated before the pointer patch is performed.

There is a potential race condition where the EMAC may read the “next” pointer of a descriptor as NULL in the instant before an application appends additional descriptors to the list by patching the pointer. This case is handled by the software application always examining the buffer descriptor flags of all EOP packets, looking for a special flag called end of queue (EOQ). The EOQ flag is set by the EMAC on the last descriptor of a packet when the descriptor’s “next” pointer is NULL. This is the way the EMAC indicates to the software application that it believes it has reached the end of the list. When the software application sees the EOQ flag set, the application may at that time submit the new list, or the portion of the appended list that was missed by writing the new list pointer to the same HDP that started the process.

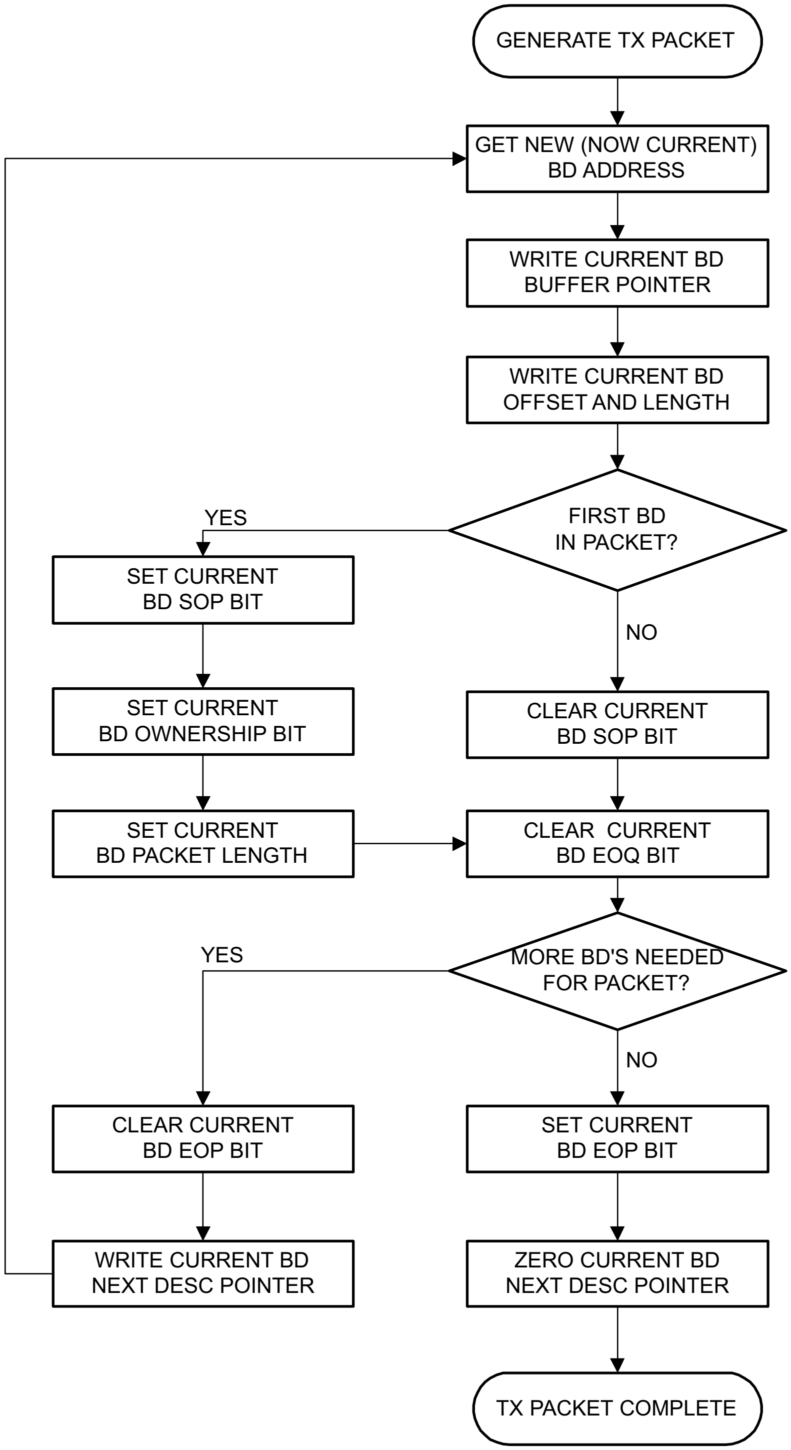
This process applies when adding packets to a transmit list, and empty buffers to a receive list. Figure 32-7, Figure 32-8, and Figure 32-9 illustrate transmit operations.

Figure 32-7. Transmit Packet Add Flow Chart


ADD PACKET(S) TO TX QUEUE
GENERATE TX PACKET (SHOWN IN FLOW CHART)
YES
SOFTWARE TX QUEUE ACTIVE?
YES
NO
SET SOFTWARE TX QUEUE ACTIVE
LINK NEW PACKET IN QUEUE BY WRITING PREVIOUS LAST PACKET NEXT DESC POINTER (PREVIOUSLY ZERO)
WRITE TX QUEUE HEAD DESCRIPTOR POINTER
ADD ANOTHER TX PACKET?
NO
TX PACKET(S) ADDED

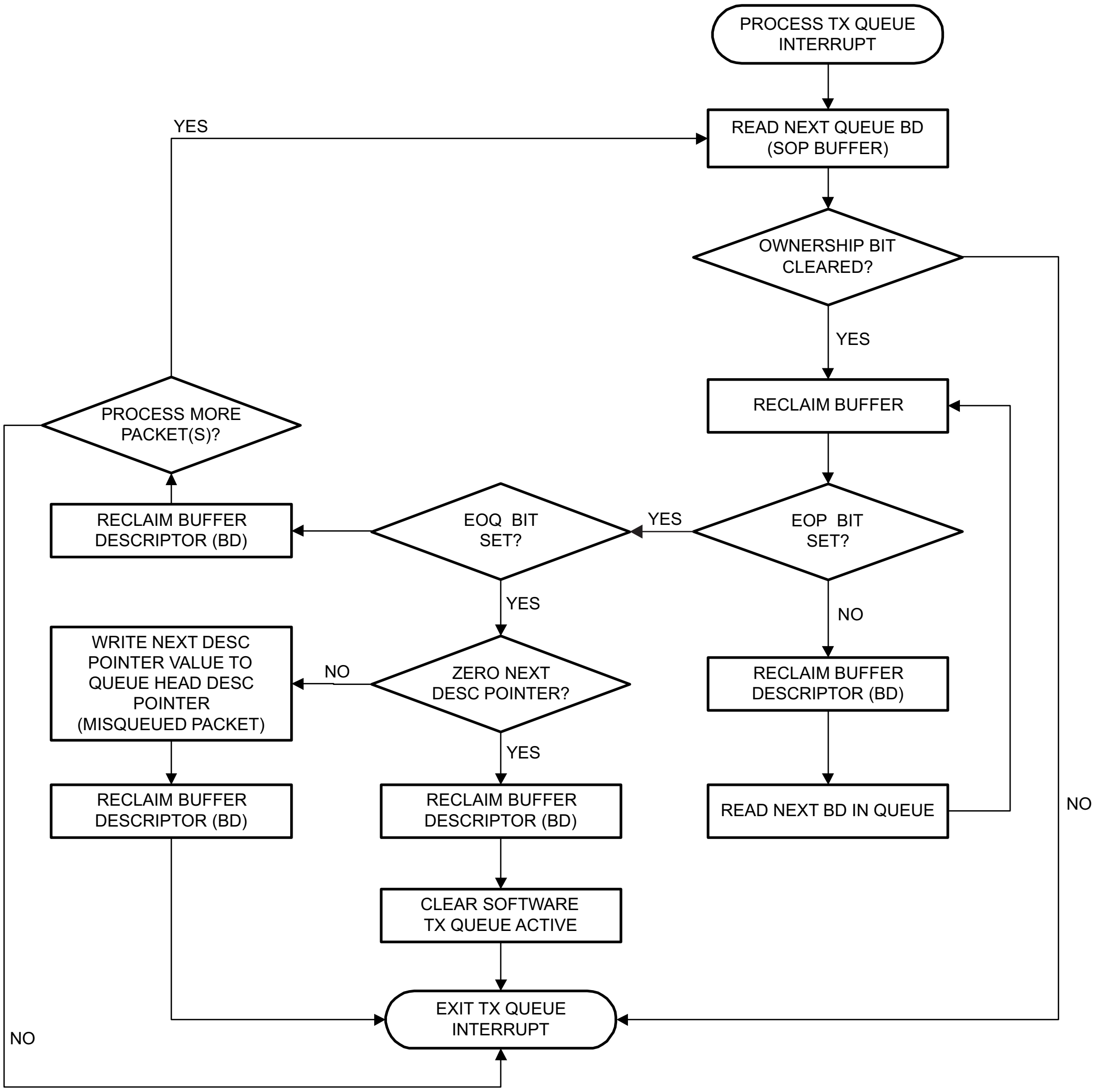
Note: Software TX QUEUE ACTIVE is an indication that at least one packet is in the TX queue (from the software viewpoint).

Figure 32-8. Generate Transmit Packet Flow Chart


GENERATE TX PACKET
GET NEW (NOW CURRENT) BD ADDRESS
WRITE CURRENT BD BUFFER POINTER
WRITE CURRENT BD OFFSET AND LENGTH
FIRST BD IN PACKET?
YES
SET CURRENT BD SOP BIT
SET CURRENT BD OWNERSHIP BIT
SET CURRENT BD PACKET LENGTH
NO
CLEAR CURRENT BD SOP BIT
CLEAR CURRENT BD EOQ BIT
MORE BD'S NEEDED FOR PACKET?
YES
CLEAR CURRENT BD EOP BIT
WRITE CURRENT BD NEXT DESC POINTER
NO
SET CURRENT BD EOP BIT
ZERO CURRENT BD NEXT DESC POINTER
TX PACKET COMPLETE

BD = Buffer Descriptor

Figure 32-9. Transmit Queue Interrupt Processing Flow Chart


PROCESS TX QUEUE INTERRUPT
READ NEXT QUEUE BD (SOP BUFFER)
OWNERSHIP BIT CLEARED?
YES
NO
RECLAIM BUFFER
EOP BIT SET?
YES
NO
EOQ BIT SET?
YES
RECLAIM BUFFER DESCRIPTOR (BD)
PROCESS MORE PACKET(S)?
YES
NO
ZERO NEXT DESC POINTER?
NO
YES
WRITE NEXT DESC POINTER VALUE TO QUEUE HEAD DESC POINTER (MISQUEUED PACKET)
RECLAIM BUFFER DESCRIPTOR (BD)
RECLAIM BUFFER DESCRIPTOR (BD)
CLEAR SOFTWARE TX QUEUE ACTIVE
RECLAIM BUFFER DESCRIPTOR (BD)
READ NEXT BD IN QUEUE
EXIT TX QUEUE INTERRUPT

BD = Buffer Descriptor

Note: Whether or not to process more than one packet is a software decision.

32.2.6.3 Transmit and Receive EMAC Interrupts

The EMAC processes descriptors in linked list chains as discussed in Section 32.2.6.1, using the linked list queue mechanism discussed in Section 32.2.6.2.

The EMAC synchronizes descriptor list processing through the use of interrupts to the software application. The interrupts are controlled by the application using the interrupt masks, global interrupt enable, and the completion pointer register (CP). The CP is also called the interrupt acknowledge register.

The EMAC supports eight channels for transmit and eight channels for receive. The corresponding completion pointer registers are:

- TX*n*CP - Transmit Channel *n* Completion Pointer (Interrupt Acknowledge) Register
- RX*n*CP - Receive Channel *n* Completion Pointer (Interrupt Acknowledge) Register

These registers serve two purposes. When read, they return the pointer to the last descriptor that the EMAC has processed. When written by the software application, the value represents the last descriptor processed by the software application. When these two values do not match, the interrupt is active.

Interrupts in the EMAC control module are routed to the Vectored Interrupt Manager (VIM) as four separate interrupt requests. The interrupt configuration determines whether or not an active interrupt request actually interrupts the CPU. In general the following settings are required for basic EMAC transmit and receive interrupts:

1. EMAC transmit and receive interrupts are enabled by setting the mask registers RXINTMASKSET and TXINTMASKSET
2. Global interrupts are set in the EMAC control module: C0RXEN and C0TXEN
3. The VIM is configured to accept C0_RX_PULSE and C0_TX_PULSE interrupts from the EMAC control module
4. The normal mode (IRQ) interrupts are enabled in the Cortex-R4F CPU

Whether or not the interrupt is enabled, the current state of the receive or transmit channel interrupt can be examined directly by the software application reading the EMAC receive interrupt status (unmasked) register (RXINTSTATRAW) and transmit interrupt status (unmasked) register (TXINTSTATRAW).

After servicing transmit or receive interrupts, the application software must acknowledge both the EMAC and EMAC control module interrupts.

EMAC interrupts are acknowledged when the application software updates the value of TX*n*CP or RX*n*CP with a value that matches the internal value kept by the EMAC. This mechanism ensures that the application software never misses an EMAC interrupt because the interrupt acknowledgment is tied directly to the buffer descriptor processing.

EMAC control module interrupts are acknowledged when the application software writes the appropriate C0TX or C0RX key to the EMAC End-Of-Interrupt Vector register (MACEOIVECTOR). The MACEOIVECTOR behaves as an interrupt pulse interlock -- once the EMAC control module has issued an interrupt pulse to the CPU, it will not generate further pulses of the same type until the original pulse has been acknowledged.

32.2.6.4 Transmit Buffer Descriptor Format

A transmit (TX) buffer descriptor (Figure 32-10) is a contiguous block of four 32-bit data words aligned on a 32-bit boundary that describes a packet or a packet fragment. Example 32-1 shows the transmit buffer descriptor described by a C structure.

Figure 32-10. Transmit Buffer Descriptor Format

Word 0

31 … 0
Next Descriptor Pointer

Word 1

31 … 0
Buffer Pointer

Word 2

31 … 16	15 … 0
Buffer Offset	Buffer Length

Word 3

31	30	29	28	27	26	25 … 16
SOP	EOP	OWNER	EOQ	TDOWNCMPLT	PASSCRC	Reserved

15 … 0
Packet Length

Example 32-1. Transmit Buffer Descriptor in C Structure Format

```
/*
// EMAC Descriptor
//
// The following is the format of a single buffer descriptor
// on the EMAC.
*/
typedef struct _EMAC_Desc {
 struct _EMAC_Desc *pNext; /* Pointer to next descriptor in chain */
 Uint8 *pBuffer; /* Pointer to data buffer */
 Uint32 BufOffLen; /* Buffer Offset(MSW) and Length(LSW) */
 Uint32 PktFlgLen; /* Packet Flags(MSW) and Length(LSW) */
} EMAC_Desc;

/* Packet Flags */
#define EMAC_DSC_FLAG_SOP 0x80000000u
#define EMAC_DSC_FLAG_EOP 0x40000000u
#define EMAC_DSC_FLAG_OWNER 0x20000000u
#define EMAC_DSC_FLAG_EOQ 0x10000000u
#define EMAC_DSC_FLAG_TDOWNCMPLT 0x08000000u
#define EMAC_DSC_FLAG_PASSCRC 0x04000000u
```

32.2.6.4.1 Next Descriptor Pointer

The next descriptor pointer points to the 32-bit word aligned memory address of the next buffer descriptor in the transmit queue. This pointer is used to create a linked list of buffer descriptors. If the value of this pointer is 0, then the current buffer is the last buffer in the queue. The software application must set this value prior to adding the descriptor to the active transmit list. This pointer is not altered by the EMAC.

The value of pNext should never be altered once the descriptor is in an active transmit queue, unless its current value is NULL. If the pNext pointer is initially NULL, and more packets need to be queued for transmit, the software application may alter this pointer to point to a newly appended descriptor. The EMAC will use the new pointer value and proceed to the next descriptor unless the pNext value has already been read. In this latter case, the transmitter will halt on the transmit channel in question, and the software application may restart it at that time. The software can detect this case by checking for an end of queue (EOQ) condition flag on the updated packet descriptor when it is returned by the EMAC.

32.2.6.4.2 Buffer Pointer

The buffer pointer is the byte-aligned memory address of the memory buffer associated with the buffer descriptor. The software application must set this value prior to adding the descriptor to the active transmit list. This pointer is not altered by the EMAC.

32.2.6.4.3 Buffer Offset

This 16-bit field indicates how many unused bytes are at the start of the buffer. For example, a value of 0000h indicates that no unused bytes are at the start of the buffer and that valid data begins on the first byte of the buffer, while a value of 000Fh indicates that the first 15 bytes of the buffer are to be ignored by the EMAC and that valid buffer data starts on byte 16 of the buffer. The software application must set this value prior to adding the descriptor to the active transmit list. This field is not altered by the EMAC.

Note that this value is only checked on the first descriptor of a given packet (where the start of packet (SOP) flag is set). It can not be used to specify the offset of subsequent packet fragments. Also, since the buffer pointer may point to any byte–aligned address, this field may be entirely superfluous, depending on the device driver architecture.

The range of legal values for this field is 0 to (Buffer Length – 1).

32.2.6.4.4 Buffer Length

This 16-bit field indicates how many valid data bytes are in the buffer. On single fragment packets, this value is also the total length of the packet data to be transmitted. If the buffer offset field is used, the offset bytes are not counted as part of this length. This length counts only valid data bytes. The software application must set this value prior to adding the descriptor to the active transmit list. This field is not altered by the EMAC.

32.2.6.4.5 Packet Length

This 16-bit field specifies the number of data bytes in the entire packet. Any leading buffer offset bytes are not included. The sum of the buffer length fields of each of the packet's fragments (if more than one) must be equal to the packet length. The software application must set this value prior to adding the descriptor to the active transmit list. This field is not altered by the EMAC. This value is only checked on the first descriptor of a given packet (where the start of packet (SOP) flag is set).

32.2.6.4.6 Start of Packet (SOP) Flag

When set, this flag indicates that the descriptor points to a packet buffer that is the start of a new packet. In the case of a single fragment packet, both the SOP and end of packet (EOP) flags are set. Otherwise, the descriptor pointing to the last packet buffer for the packet sets the EOP flag. This bit is set by the software application and is not altered by the EMAC.

32.2.6.4.7 End of Packet (EOP) Flag

When set, this flag indicates that the descriptor points to a packet buffer that is last for a given packet. In the case of a single fragment packet, both the start of packet (SOP) and EOP flags are set. Otherwise, the descriptor pointing to the last packet buffer for the packet sets the EOP flag. This bit is set by the software application and is not altered by the EMAC.

32.2.6.4.8 Ownership (OWNER) Flag

When set, this flag indicates that all the descriptors for the given packet (from SOP to EOP) are currently owned by the EMAC. This flag is set by the software application on the SOP packet descriptor before adding the descriptor to the transmit descriptor queue. For a single fragment packet, the SOP, EOP, and OWNER flags are all set. The OWNER flag is cleared by the EMAC once it is finished with all the descriptors for the given packet. Note that this flag is valid on SOP descriptors only.

32.2.6.4.9 End of Queue (EOQ) Flag

When set, this flag indicates that the descriptor in question was the last descriptor in the transmit queue for a given transmit channel, and that the transmitter has halted. This flag is initially cleared by the software application prior to adding the descriptor to the transmit queue. This bit is set by the EMAC when the EMAC identifies that a descriptor is the last for a given packet (the EOP flag is set), and there are no more descriptors in the transmit list (next descriptor pointer is NULL).

The software application can use this bit to detect when the EMAC transmitter for the corresponding channel has halted. This is useful when the application appends additional packet descriptors to a transmit queue list that is already owned by the EMAC. Note that this flag is valid on EOP descriptors only.

32.2.6.4.10 Teardown Complete (TDOWNCMPLT) Flag

This flag is used when a transmit queue is being torn down, or aborted, instead of allowing it to be transmitted. This would happen under device driver reset or shutdown conditions. The EMAC sets this bit in the SOP descriptor of each packet as it is aborted from transmission.

Note that this flag is valid on SOP descriptors only. Also note that only the first packet in an unsent list has the TDOWNCMPLT flag set. Subsequent descriptors are not processed by the EMAC.

32.2.6.4.11 Pass CRC (PASSCRC) Flag

This flag is set by the software application in the SOP packet descriptor before it adds the descriptor to the transmit queue. Setting this bit indicates to the EMAC that the 4 byte Ethernet CRC is already present in the packet data, and that the EMAC should not generate its own version of the CRC.

When the CRC flag is cleared, the EMAC generates and appends the 4-byte CRC. The buffer length and packet length fields do not include the CRC bytes. When the CRC flag is set, the 4-byte CRC is supplied by the software application and is already appended to the end of the packet data. The buffer length and packet length fields include the CRC bytes, as they are part of the valid packet data. Note that this flag is valid on SOP descriptors only.

32.2.6.5 Receive Buffer Descriptor Format

A receive (RX) buffer descriptor (Figure 32-11) is a contiguous block of four 32-bit data words aligned on a 32-bit boundary that describes a packet or a packet fragment. Example 32-2 shows the receive buffer descriptor described by a C structure.

32.2.6.5.1 Next Descriptor Pointer

This pointer points to the 32–bit word aligned memory address of the next buffer descriptor in the receive queue. This pointer is used to create a linked list of buffer descriptors. If the value of this pointer is 0, then the current buffer is the last buffer in the queue. The software application must set this value prior to adding the descriptor to the active receive list. This pointer is not altered by the EMAC.

The value of pNext should never be altered once the descriptor is in an active receive queue, unless its current value is NULL. If the pNext pointer is initially NULL, and more empty buffers can be added to the pool, the software application may alter this pointer to point to a newly appended descriptor. The EMAC will use the new pointer value and proceed to the next descriptor unless the pNext value has already been read. In this latter case, the receiver will halt the receive channel in question, and the software application may restart it at that time. The software can detect this case by checking for an end of queue (EOQ) condition flag on the updated packet descriptor when it is returned by the EMAC.

32.2.6.5.2 Buffer Pointer

The buffer pointer is the byte-aligned memory address of the memory buffer associated with the buffer descriptor. The software application must set this value prior to adding the descriptor to the active receive list. This pointer is not altered by the EMAC.

Figure 32-11. Receive Buffer Descriptor Format

Word 0

31 … 0
Next Descriptor Pointer

Word 1

31 … 0
Buffer Pointer

Word 2

31 … 16	15 … 0
Buffer Offset	Buffer Length

Word 3

31	30	29	28	27	26	25	24
SOP	EOP	OWNER	EOQ	TDOWNCMPLT	PASSCRC	JABBER	OVERSIZE

23	22	21	20	19	18	17	16
FRAGMENT	UNDERSIZED	CONTROL	OVERRUN	CODEERROR	ALIGNERROR	CRCERROR	NOMATCH

15 … 0
Packet Length

Example 32-2. Receive Buffer Descriptor in C Structure Format

```
/*
// EMAC Descriptor
//
// The following is the format of a single buffer descriptor
// on the EMAC.
*/
typedef struct _EMAC_Desc {
 struct _EMAC_Desc *pNext; /* Pointer to next descriptor in chain */
 Uint8 *pBuffer; /* Pointer to data buffer */
 Uint32 BufOffLen; /* Buffer Offset(MSW) and Length(LSW) */
 Uint32 PktFlgLen; /* Packet Flags(MSW) and Length(LSW) */
} EMAC_Desc;

/* Packet Flags */
#define EMAC_DSC_FLAG_SOP 0x80000000u
#define EMAC_DSC_FLAG_EOP 0x40000000u
#define EMAC_DSC_FLAG_OWNER 0x20000000u
#define EMAC_DSC_FLAG_EOQ 0x10000000u
#define EMAC_DSC_FLAG_TDOWNCMPLT 0x08000000u
#define EMAC_DSC_FLAG_PASSCRC 0x04000000u
#define EMAC_DSC_FLAG_JABBER 0x02000000u
#define EMAC_DSC_FLAG_OVERSIZE 0x01000000u
#define EMAC_DSC_FLAG_FRAGMENT 0x00800000u
```

Example 32-2. Receive Buffer Descriptor in C Structure Format (continued)

```
#define EMAC_DSC_FLAG_UNDERSIZED 0x00400000u
#define EMAC_DSC_FLAG_CONTROL 0x00200000u
#define EMAC_DSC_FLAG_OVERRUN 0x00100000u
#define EMAC_DSC_FLAG_CODEERROR 0x00080000u
#define EMAC_DSC_FLAG_ALIGNERROR 0x00040000u
#define EMAC_DSC_FLAG_CRCERROR 0x00020000u
#define EMAC_DSC_FLAG_NOMATCH 0x00010000u
```

32.2.6.5.3 Buffer Offset

This 16-bit field must be initialized to 0 by the software application before adding the descriptor to a receive queue.

Whether or not this field is updated depends on the setting of the RXBUFFEROFFSET register. When the offset register is set to a nonzero value, the received packet is written to the packet buffer at an offset given by the value of the register, and this value is also written to the buffer offset field of the descriptor.

When a packet is fragmented over multiple buffers because it does not fit in the first buffer supplied, the buffer offset only applies to the first buffer in the list, which is where the start of packet (SOP) flag is set in the corresponding buffer descriptor. In other words, the buffer offset field is only updated by the EMAC on SOP descriptors.

The range of legal values for the BUFFEROFFSET register is 0 to (Buffer Length – 1) for the smallest value of buffer length for all descriptors in the list.

32.2.6.5.4 Buffer Length

This 16-bit field is used for two purposes:

- Before the descriptor is first placed on the receive queue by the application software, the buffer length field is first initialized by the software to have the physical size of the empty data buffer pointed to by the buffer pointer field.
- After the empty buffer has been processed by the EMAC and filled with received data bytes, the buffer length field is updated by the EMAC to reflect the actual number of valid data bytes written to the buffer.

32.2.6.5.5 Packet Length

This 16-bit field specifies the number of data bytes in the entire packet. This value is initialized to 0 by the software application for empty packet buffers. The value is filled in by the EMAC on the first buffer used for a given packet. This is signified by the EMAC setting a start of packet (SOP) flag. The packet length is set by the EMAC on all SOP buffer descriptors.

32.2.6.5.6 Start of Packet (SOP) Flag

When set, this flag indicates that the descriptor points to a packet buffer that is the start of a new packet. In the case of a single fragment packet, both the SOP and end of packet (EOP) flags are set. Otherwise, the descriptor pointing to the last packet buffer for the packet has the EOP flag set. This flag is initially cleared by the software application before adding the descriptor to the receive queue. This bit is set by the EMAC on SOP descriptors.

32.2.6.5.7 End of Packet (EOP) Flag

When set, this flag indicates that the descriptor points to a packet buffer that is last for a given packet. In the case of a single fragment packet, both the start of packet (SOP) and EOP flags are set. Otherwise, the descriptor pointing to the last packet buffer for the packet has the EOP flag set. This flag is initially cleared by the software application before adding the descriptor to the receive queue. This bit is set by the EMAC on EOP descriptors.

32.2.6.5.8 Ownership (OWNER) Flag

When set, this flag indicates that the descriptor is currently owned by the EMAC. This flag is set by the software application before adding the descriptor to the receive descriptor queue. This flag is cleared by the EMAC once it is finished with a given set of descriptors, associated with a received packet. The flag is updated by the EMAC on SOP descriptor only. So when the application identifies that the OWNER flag is cleared on an SOP descriptor, it may assume that all descriptors up to and including the first with the EOP flag set have been released by the EMAC. (Note that in the case of single buffer packets, the same descriptor will have both the SOP and EOP flags set.)

32.2.6.5.9 End of Queue (EOQ) Flag

When set, this flag indicates that the descriptor in question was the last descriptor in the receive queue for a given receive channel, and that the corresponding receiver channel has halted. This flag is initially cleared by the software application prior to adding the descriptor to the receive queue. This bit is set by the EMAC when the EMAC identifies that a descriptor is the last for a given packet received (also sets the EOP flag), and there are no more descriptors in the receive list (next descriptor pointer is NULL).

The software application can use this bit to detect when the EMAC receiver for the corresponding channel has halted. This is useful when the application appends additional free buffer descriptors to an active receive queue. Note that this flag is valid on EOP descriptors only.

32.2.6.5.10 Teardown Complete (TDOWNCMPLT) Flag

This flag is used when a receive queue is being torn down, or aborted, instead of being filled with received data. This would happen under device driver reset or shutdown conditions. The EMAC sets this bit in the descriptor of the first free buffer when the tear down occurs. No additional queue processing is performed.

32.2.6.5.11 Pass CRC (PASSCRC) Flag

This flag is set by the EMAC in the SOP buffer descriptor if the received packet includes the 4-byte CRC. This flag should be cleared by the software application before submitting the descriptor to the receive queue.

32.2.6.5.12 Jabber Flag

This flag is set by the EMAC in the SOP buffer descriptor, if the received packet is a jabber frame and was not discarded because the RXCEFEN bit was set in the RXMBPENABLE. Jabber frames are frames that exceed the RXMAXLEN in length, and have CRC, code, or alignment errors.

32.2.6.5.13 Oversize Flag

This flag is set by the EMAC in the SOP buffer descriptor, if the received packet is an oversized frame and was not discarded because the RXCEFEN bit was set in the RXMBPENABLE.

32.2.6.5.14 Fragment Flag

This flag is set by the EMAC in the SOP buffer descriptor, if the received packet is only a packet fragment and was not discarded because the RXCEFEN bit was set in the RXMBPENABLE.

32.2.6.5.15 Undersized Flag

This flag is set by the EMAC in the SOP buffer descriptor, if the received packet is undersized and was not discarded because the RXCSFEN bit was set in the RXMBPENABLE.

32.2.6.5.16 Control Flag

This flag is set by the EMAC in the SOP buffer descriptor, if the received packet is an EMAC control frame and was not discarded because the RXCMFEN bit was set in the RXMBPENABLE.

32.2.6.5.17 Overrun Flag

This flag is set by the EMAC in the SOP buffer descriptor, if the received packet was aborted due to a receive overrun.

32.2.6.5.18 Code Error (CODEERROR) Flag

This flag is set by the EMAC in the SOP buffer descriptor, if the received packet contained a code error and was not discarded because the RXCEFEN bit was set in the RXMBPENABLE.

32.2.6.5.19 Alignment Error (ALIGNERROR) Flag

This flag is set by the EMAC in the SOP buffer descriptor, if the received packet contained an alignment error and was not discarded because the RXCEFEN bit was set in the RXMBPENABLE.

32.2.6.5.20 CRC Error (CRCERROR) Flag

This flag is set by the EMAC in the SOP buffer descriptor, if the received packet contained a CRC error and was not discarded because the RXCEFEN bit was set in the RXMBPENABLE.

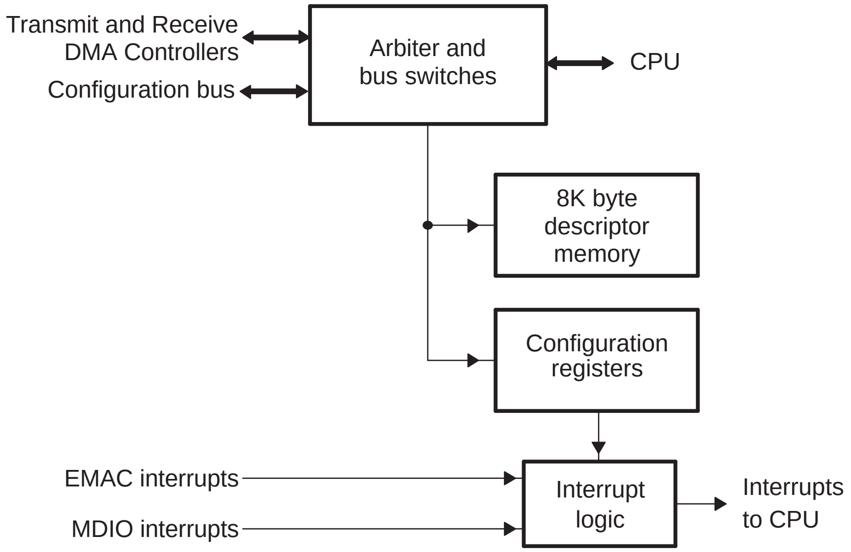
32.2.6.5.21 No Match (NOMATCH) Flag

This flag is set by the EMAC in the SOP buffer descriptor, if the received packet did not pass any of the EMAC’s address match criteria and was not discarded because the RXCAFEN bit was set in the RXMBPENABLE. Although the packet is a valid Ethernet data packet, it was only received because the EMAC is in promiscuous mode.

32.2.7 EMAC Control Module

The EMAC control module (Figure 32-12) interfaces the EMAC and MDIO modules to the rest of the system, and also provides a local memory space to hold EMAC packet buffer descriptors. Local memory is used to help avoid contention with device memory spaces. Other functions include the bus arbiter and the interrupt logic control.

Figure 32-12. EMAC Control Module Block Diagram



32.2.7.1 Internal Memory

The EMAC control module includes 8K bytes of internal memory (CPPI buffer descriptor memory). The internal memory block is essential for allowing the EMAC to operate more independently of the CPU. It also prevents memory underflow conditions when the EMAC issues read or write requests to descriptor memory. (Memory accesses to read or write the actual Ethernet packet data are protected by the EMAC's internal FIFOs).

A descriptor is a 16-byte memory structure that holds information about a single Ethernet packet buffer, which may contain a full or partial Ethernet packet. Thus with the 8K memory block provided for descriptor storage, the EMAC module can send and received up to a combined 512 packets before it needs to be serviced by application or driver software.

32.2.7.2 Interrupt Control

Interrupt conditions generated by the EMAC and MDIO modules are combined into four interrupt signals that are routed to the Vectored Interrupt Manager (VIM); the VIM then relays the interrupt signals to the CPU. The EMAC control module uses two sets of registers to control the interrupt signals to the CPU:

- C0RXTHRESHEN, C0RXEN, C0TXEN, and C0MISCEN registers enable the pulse signals that are mapped to the VIM
- INTCONTROL, C0RXIMAX, and C0TXIMAX registers enable interrupt pacing to limit the number of interrupt pulses generated per millisecond

Interrupts must be acknowledged by writing the appropriate value to the EMAC End-Of-Interrupt Vector (MACEOIVECTOR). The MACEOIVECTOR behaves as an interrupt pulse interlock -- once the EMAC control module has issued an interrupt pulse to the CPU, it will not generate further pulses of the same type until the original pulse has been acknowledged.

32.2.7.3 Bus Arbiter

The EMAC control module bus arbiter operates transparently to the rest of the system. It is used:

- To arbitrate between the CPU and EMAC buses for access to internal descriptor memory.
- To arbitrate between internal EMAC buses for access to system memory.

32.2.8 MDIO Module

The MDIO module is used to manage up to 32 physical layer (PHY) devices connected to the Ethernet Media Access Controller (EMAC). The device supports a single PHY being connected to the EMAC at any given time. The MDIO module is designed to allow almost transparent operation of the MDIO interface with little maintenance from the CPU.

The MDIO module continuously polls 32 MDIO addresses in order to enumerate all PHY devices in the system. Once a PHY device has been detected, the MDIO module reads the MDIO PHY link status register (LINK) to monitor the PHY link state. Link change events are stored in the MDIO module, which can interrupt the CPU. This storing of the events allows the CPU to poll the link status of the PHY device without continuously performing MDIO module accesses. However, when the CPU must access the MDIO module for configuration and negotiation, the MDIO module performs the MDIO read or write operation independent of the CPU. This independent operation allows the processor to poll for completion or interrupt the CPU once the operation has completed.

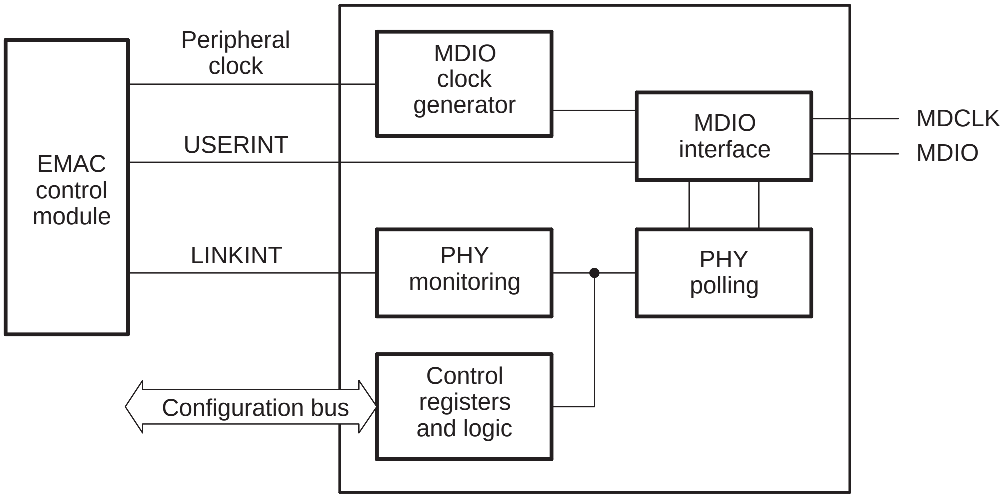
The MDIO module does not support the "Clause 45" interface.

32.2.8.1 MDIO Module Components

The MDIO module (Figure 32-13) interfaces to the PHY components through two MDIO pins (MDIO_CLK and MDIO), and to the CPU through the EMAC control module and the configuration bus. The MDIO module consists of the following logical components:

- MDIO clock generator
- Global PHY detection and link state monitoring
- Active PHY monitoring
- PHY register user access

Figure 32-13. MDIO Module Block Diagram



32.2.8.1.1 MDIO Clock Generator

The MDIO clock generator controls the MDIO clock based on a divide-down of the VCLK3 peripheral clock in the EMAC control module. The MDIO clock is specified to run up to 2.5 MHz, although typical operation would be 1.0 MHz. Since the VCLK3 peripheral clock frequency is configurable, the application software or driver controls the divide-down amount. See the device datasheet for peripheral clock speed specifications.

32.2.8.1.2 Global PHY Detection and Link State Monitoring

The MDIO module continuously polls all 32 MDIO addresses in order to enumerate the PHY devices in the system. The module tracks whether or not a PHY on a particular address has responded, and whether or not the PHY currently has a link. Using this information allows the software application to quickly determine which MDIO address the PHY is using.

32.2.8.1.3 Active PHY Monitoring

Once a PHY candidate has been selected for use, the MDIO module transparently monitors its link state by reading the MDIO PHY link status register (LINK). Link change events are stored on the MDIO device and can optionally interrupt the CPU. This allows the system to poll the link status of the PHY device without continuously performing costly MDIO accesses.

32.2.8.1.4 PHY Register User Access

When the CPU must access MDIO for configuration and negotiation, the PHY access module performs the actual MDIO read or write operation independent of the CPU. This allows the CPU to poll for completion or receive an interrupt when the read or write operation has been performed. The user access registers USERACCESS*n* allows the software to submit the access requests for the PHY connected to the device.

32.2.8.2 MDIO Module Operational Overview

The MDIO module implements the 802.3 serial management interface to interrogate and control an Ethernet PHY, using a shared two-wired bus. It separately performs autodetection and records the current link status of up to 32 PHYs, polling all 32 MDIO addresses.

Application software uses the MDIO module to configure the autonegotiation parameters of the PHY attached to the EMAC, retrieve the negotiation results, and configure required parameters in the EMAC.

In this device, the Ethernet PHY attached to the system can be directly controlled and queried. The Media Independent Interface (MII) address of this PHY device is specified in one of the PHYADRMON bits in the MDIO user PHY select register (USERPHYSEL*n*). The MDIO module can be programmed to trigger a CPU interrupt on a PHY link change event, by setting the LINKINTENB bit in USERPHYSEL*n*. Reads and writes to registers in this PHY device are performed using the MDIO user access register (USERACCESS*n*).

The MDIO module powers-up in an idle state until specifically enabled by setting the ENABLE bit in the MDIO control register (CONTROL). At this time, the MDIO clock divider and preamble mode selection are also configured. The MDIO preamble is enabled by default, but can be disabled when the connected PHY does not require it. Once the MDIO module is enabled, the MDIO interface state machine continuously polls the PHY link status (by reading the generic status register) of all possible 32 PHY addresses and records the results in the MDIO PHY alive status register (ALIVE) and MDIO PHY link status register (LINK). The corresponding bit for the connected PHY (0-31) is set in ALIVE, if the PHY responded to the read request. The corresponding bit is set in LINK, if the PHY responded and also is currently linked. In addition, any PHY register read transactions initiated by the application software using USERACCESS*n* causes ALIVE to be updated.

The USERPHYSEL*n* is used to track the link status of the connected PHY address. A change in the link status of the PHY being monitored sets the appropriate bit in the MDIO link status change interrupt registers (LINKINTRAW and LINKINTMASKED), if enabled by the LINKINTENB bit in USERPHYSEL*n*.

While the MDIO module is enabled, the host issues a read or write transaction over the MII management interface using the DATA, PHYADR, REGADR, and WRITE bits in USERACCESS*n*. When the application sets the GO bit in USERACCESS*n*, the MDIO module begins the transaction without any further intervention from the CPU. Upon completion, the MDIO module clears the GO bit and sets the corresponding USERINTRAW bit (0 or 1) in the MDIO user command complete interrupt register (USERINTRAW) corresponding to USERACCESS*n* used. The corresponding USERINTMASKED bit (0 or 1) in the MDIO user command complete interrupt register (USERINTMASKED) may also be set, depending on the mask setting configured in the MDIO user command complete interrupt mask set register (USERINTMASKSET) and the MDIO user interrupt mask clear register (USERINTMASKCLEAR).

A round-robin arbitration scheme is used to schedule transactions that may be queued using both USERACCESS0 and USERACCESS1. The application software must check the status of the GO bit in USERACCESS*n* before initiating a new transaction, to ensure that the previous transaction has completed. The application software can use the ACK bit in USERACCESS*n* to determine the status of a read transaction.

32.2.8.2.1 Initializing the MDIO Module

The following steps are performed by the application software or device driver to initialize the MDIO device:

1. Configure the PREAMBLE and CLKDIV bits in the MDIO control register (CONTROL).
2. Enable the MDIO module by setting the ENABLE bit in CONTROL.
3. The MDIO PHY alive status register (ALIVE) can be read in polling fashion until a PHY connected to the system responded, and the MDIO PHY link status register (LINK) can determine whether this PHY already has a link.
4. Setup the appropriate PHY addresses in the MDIO user PHY select register (USERPHYSEL*n*), and set the LINKINTENB bit to enable a link change event interrupt if desirable.
5. If an interrupt on general MDIO register access is desired, set the corresponding bit in the MDIO user command complete interrupt mask set register (USERINTMASKSET) to use the MDIO user access register (USERACCESS*n*). Since only one PHY is used in this device, the application software can use one USERACCESS*n* to trigger a completion interrupt; the other USERACCESS*n* is not setup.

32.2.8.2.2 Writing Data To a PHY Register

The MDIO module includes a user access register (USERACCESS*n*) to directly access a specified PHY device. To write a PHY register, perform the following:

1. Check to ensure that the GO bit in the MDIO user access register (USERACCESS*n*) is cleared.
2. Write to the GO, WRITE, REGADR, PHYADR, and DATA bits in USERACCESS*n* corresponding to the PHY and PHY register you want to write.
3. The write operation to the PHY is scheduled and completed by the MDIO module. Completion of the write operation can be determined by polling the GO bit in USERACCESS*n* for a 0.
4. Completion of the operation sets the corresponding USERINTRAW bit (0 or 1) in the MDIO user command complete interrupt register (USERINTRAW) corresponding to USERACCESS*n* used. If interrupts have been enabled on this bit using the MDIO user command complete interrupt mask set register (USERINTMASKSET), then the bit is also set in the MDIO user command complete interrupt register (USERINTMASKED) and an interrupt is triggered on the CPU.

32.2.8.2.3 Reading Data From a PHY Register

The MDIO module includes a user access register (USERACCESS*n*) to directly access a specified PHY device. To read a PHY register, perform the following:

1. Check to ensure that the GO bit in the MDIO user access register (USERACCESS*n*) is cleared.
2. Write to the GO, REGADR, and PHYADR bits in USERACCESS*n* corresponding to the PHY and PHY register you want to read.
3. The read data value is available in the DATA bits in USERACCESS*n* after the module completes the read operation on the serial bus. Completion of the read operation can be determined by polling the GO and ACK bits in USERACCESS*n*. Once the GO bit has cleared, the ACK bit is set on a successful read.
4. Completion of the operation sets the corresponding USERINTRAW bit (0 or 1) in the MDIO user command complete interrupt register (USERINTRAW) corresponding to USERACCESS*n* used. If interrupts have been enabled on this bit using the MDIO user command complete interrupt mask set register (USERINTMASKSET), then the bit is also set in the MDIO user command complete interrupt register (USERINTMASKED) and an interrupt is triggered on the CPU.

32.2.8.2.4 Example of MDIO Register Access Code

The MDIO module uses the MDIO user access register (USERACCESS*n*) to access the PHY control registers. Software functions that implement the access process may simply be the following four macros:

• PHYREG_read(regadr, phyadr)	Start the process of reading a PHY register
• PHYREG_write(regadr, phyadr, data)	Start the process of writing a PHY register
• PHYREG_wait( )	Synchronize operation (make sure read/write is idle)
• PHYREG_waitResults(results)	Wait for read to complete and return data read

Note that it is not necessary to wait after a write operation, as long as the status is checked before every operation to make sure the MDIO hardware is idle. An alternative approach is to call PHYREG_wait() after every write, and PHYREG_waitResults( ) after every read, then the hardware can be assumed to be idle when starting a new operation.

The implementation of these macros using the chip support library (CSL) is shown in Example 32-3 (USERACCESS0 is assumed).

Note that this implementation does not check the ACK bit in USERACCESS*n* on PHY register reads (does not follow the procedure outlined in Section 32.2.8.2.3). Since the MDIO PHY alive status register (ALIVE) is used to initially select a PHY, it is assumed that the PHY is acknowledging read operations. It is possible that a PHY could become inactive at a future point in time. An example of this would be a PHY that can have its MDIO addresses changed while the system is running. It is not very likely, but this condition can be tested by periodically checking the PHY state in ALIVE.

Example 32-3. MDIO Register Access Macros

```
#define PHYREG_read(regadr, phyadr)
    MDIO_REGS->USERACCESS0 =
        CSL_FMK(MDIO_USERACCESS0_GO,1u)                        | /
        CSL_FMK(MDIO_USERACCESS0_REGADR,regadr)                | /
        CSL_FMK(MDIO_USERACCESS0_PHYADR,phyadr)
#define PHYREG_write(regadr, phyadr, data)
    MDIO_REGS->USERACCESS0 =
        CSL_FMK(MDIO_USERACCESS0_GO,1u)                        | /
        CSL_FMK(MDIO_USERACCESS0_WRITE,1)                      | /
        CSL_FMK(MDIO_USERACCESS0_REGADR,regadr)                | /
        CSL_FMK(MDIO_USERACCESS0_PHYADR,phyadr)                | /
        CSL_FMK(MDIO_USERACCESS0_DATA, data)
#define PHYREG_wait()
    while( CSL_FEXT(MDIO_REGS->USERACCESS0,MDIO_USERACCESS0_GO) )
#define PHYREG_waitResults( results ) {
    while( CSL_FEXT(MDIO_REGS->USERACCESS0,MDIO_USERACCESS0_GO) );
results = CSL_FEXT(MDIO_REGS->USERACCESS0, MDIO_USERACCESS0_DATA); }
```

32.2.9 *EMAC Module*

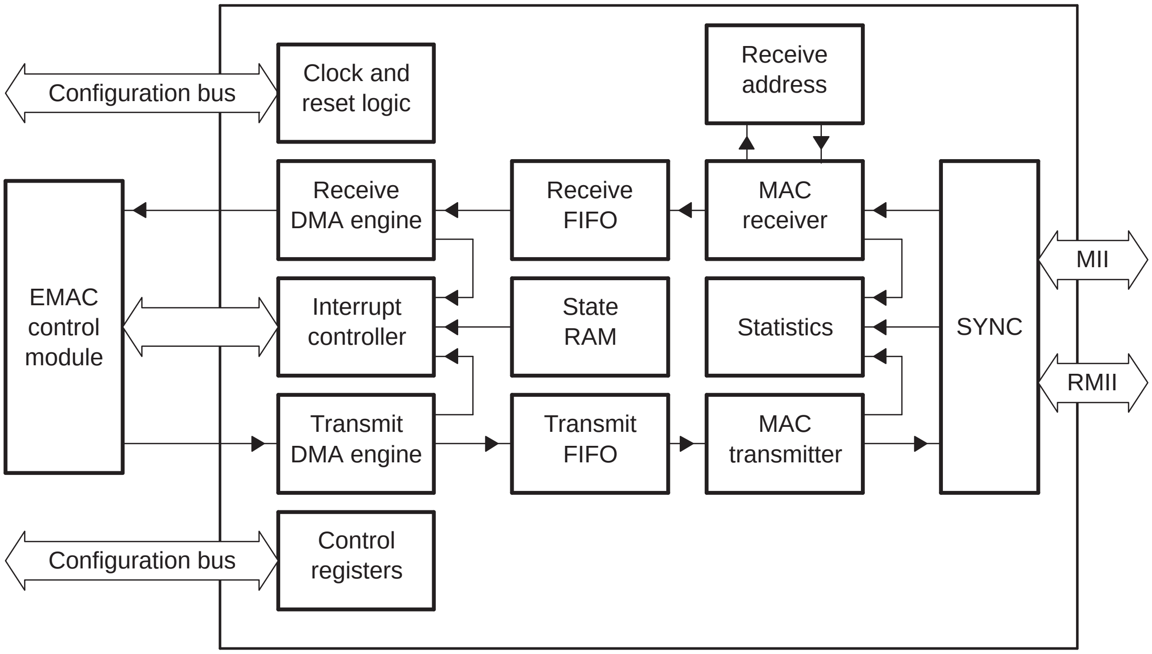
This section discusses the architecture and basic function of the EMAC module.

32.2.9.1 EMAC Module Components

The EMAC module (Figure 32-14) interfaces to the outside world through the Media Independent Interface (MII) or Reduced Media Independent Interface (RMII). The interface between the EMAC module and the system core is provided through the EMAC control module. The EMAC consists of the following logical components:

- The receive path includes: receive DMA engine, receive FIFO, and MAC receiver
- The transmit path includes: transmit DMA engine, transmit FIFO, and MAC transmitter
- Statistics logic
- State RAM
- Interrupt controller
- Control registers and logic
- Clock and reset logic

Figure 32-14. EMAC Module Block Diagram



32.2.9.1.1 Receive DMA Engine

The receive DMA engine is the interface between the receive FIFO and the system core. It interfaces to the CPU through the bus arbiter in the EMAC control module. This DMA engine is totally independent of the device DMA.

32.2.9.1.2 Receive FIFO

The receive FIFO consists of three cells of 64-bytes each and associated control logic. The FIFO buffers receive data in preparation for writing into packet buffers in device memory.

32.2.9.1.3 MAC Receiver

The MAC receiver detects and processes incoming network frames, de-frames them, and puts them into the receive FIFO. The MAC receiver also detects errors and passes statistics to the statistics RAM.

32.2.9.1.4 Transmit DMA Engine

The transmit DMA engine is the interface between the transmit FIFO and the CPU. It interfaces to the CPU through the bus arbiter in the EMAC control module.

32.2.9.1.5 Transmit FIFO

The transmit FIFO consists of three cells of 64-bytes each and associated control logic. The FIFO buffers data in preparation for transmission.

32.2.9.1.6 MAC Transmitter

The MAC transmitter formats frame data from the transmit FIFO and transmits the data using the CSMA/CD access protocol. The frame CRC can be automatically appended, if required. The MAC transmitter also detects transmission errors and passes statistics to the statistics registers.

32.2.9.1.7 Statistics Logic

The Ethernet statistics are counted and stored in the statistics logic RAM. This statistics RAM keeps track of 36 different Ethernet packet statistics.

32.2.9.1.8 State RAM

State RAM contains the head descriptor pointers and completion pointers registers for both transmit and receive channels.

32.2.9.1.9 EMAC Interrupt Controller

The interrupt controller contains the interrupt related registers and logic. The 26 raw EMAC interrupts are input to this submodule and masked module interrupts are output.

32.2.9.1.10 Control Registers and Logic

The EMAC is controlled by a set of memory-mapped registers. The control logic also signals transmit, receive, and status related interrupts to the CPU through the EMAC control module.

32.2.9.1.11 Clock and Reset Logic

The clock and reset submodule generates all the EMAC clocks and resets. For more details on reset capabilities, see Section 32.2.15.1.

32.2.9.2 EMAC Module Operational Overview

After reset, initialization, and configuration, the host may initiate transmit operations. Transmit operations are initiated by host writes to the appropriate transmit channel head descriptor pointer contained in the state RAM block. The transmit DMA controller then fetches the first packet in the packet chain from memory. The DMA controller writes the packet into the transmit FIFO in bursts of 64-byte cells. When the threshold number of cells, configurable using the TXCELLTHRESH bit in the FIFO control register (FIFOCONTROL), have been written to the transmit FIFO, or a complete packet, whichever is smaller, the MAC transmitter then initiates the packet transmission. The SYNC block transmits the packet over the MII or RMII interfaces in accordance with the 802.3 protocol. Transmit statistics are counted by the statistics block.

Receive operations are initiated by host writes to the appropriate receive channel head descriptor pointer after host initialization and configuration. The SYNC submodule receives packets and strips off the Ethernet related protocol. The packet data is input to the MAC receiver, which checks for address match and processes errors. Accepted packets are then written to the receive FIFO in bursts of 64-byte cells. The receive DMA controller then writes the packet data to memory. Receive statistics are counted by the statistics block.

The EMAC module operates independently of the CPU. It is configured and controlled by its register set mapped into device memory. Information about data packets is communicated by use of 16-byte descriptors that are placed in an 8K-byte block of RAM in the EMAC control module (CPPI buffer descriptor memory).

For transmit operations, each 16-byte descriptor describes a packet or packet fragment in the system's internal or external memory. For receive operations, each 16-byte descriptor represents a free packet buffer or buffer fragment. On both transmit and receive, an Ethernet packet is allowed to span one or more memory fragments, represented by one 16-byte descriptor per fragment. In typical operation, there is only one descriptor per receive buffer, but transmit packets may be fragmented, depending on the software architecture.

An interrupt is issued to the CPU whenever a transmit or receive operation has completed. However, it is not necessary for the CPU to service the interrupt while there are additional resources available. In other words, the EMAC continues to receive Ethernet packets until its receive descriptor list has been exhausted. On transmit operations, the transmit descriptors need only be serviced to recover their associated memory buffer. Thus, it is possible to delay servicing of the EMAC interrupt if there are real-time tasks to perform.

Eight channels are supplied for both transmit and receive operations. On transmit, the eight channels represent eight independent transmit queues. The EMAC can be configured to treat these channels as an equal priority "round-robin" queue or as a set of eight fixed-priority queues. On receive, the eight channels represent eight independent receive queues with packet classification. Packets are classified based on the destination MAC address. Each of the eight channels is assigned its own MAC address, enabling the EMAC module to act like eight virtual MAC adapters. Also, specific types of frames can be sent to specific channels. For example, multicast, broadcast, or other (promiscuous, error, etc.), can each be received on a specific receive channel queue.

The EMAC keeps track of 36 different statistics, plus keeps the status of each individual packet in its corresponding packet descriptor.

32.2.10 MAC Interface

The following sections discuss the operation of the Media Independent Interface (MII) and Reduced Media Independent Interface (RMII) in 10 Mbps and 100 Mbps mode. An IEEE 802.3 compliant Ethernet MAC controls the interface.

32.2.10.1 Data Reception

32.2.10.1.1 Receive Control

Data received from the PHY is interpreted and output to the EMAC receive FIFO. Interpretation involves detection and removal of the preamble and start-of-frame delimiter, extraction of the address and frame length, data handling, error checking and reporting, cyclic redundancy checking (CRC), and statistics control signal generation. Address detection and frame filtering is performed outside the MAC interface.

32.2.10.1.2 Receive Inter-Frame Interval

The 802.3 standard requires an interpacket gap (IPG), which is 96 bit times. However, the EMAC can tolerate a reduced IPG of 8 bit times with a correct preamble and start frame delimiter. This interval between frames must comprise (in the following order):

1. An Interpacket Gap (IPG).
2. A 7-byte preamble (all bytes 55h).
3. A 1-byte start of frame delimiter (5Dh).

32.2.10.1.3 Receive Flow Control

When enabled and triggered, receive flow control is initiated to limit the EMAC from further frame reception. Two forms of receive buffer flow control are available:

- Collision-based flow control for half-duplex mode
- IEEE 802.3x pause frames flow control for full-duplex mode

In either case, receive flow control prevents frame reception by issuing the flow control appropriate for the current mode of operation. Receive flow control prevents reception of frames on the EMAC until all of the triggering conditions clear, at which time frames may again be received by the EMAC.

Receive flow control is enabled by the RXBUFFERFLOWEN bit in the MAC control register (MACCONTROL). The EMAC is configured for collision or IEEE 802.3X flow control using the FULLDUPLEX bit in MACCONTROL. Receive flow control is triggered when the number of free buffers in any enabled receive channel free buffer count register (RX*n*FREEBUFFER) is less than or equal to the receive channel flow control threshold register (RX*n*FLOWTHRESH) value. Receive flow control is independent of receive QOS, except that both use the free buffer values.

32.2.10.1.3.1 Collision-Based Receive Buffer Flow Control

Collision-based receive buffer flow control provides a means of preventing frame reception when the EMAC is operating in half-duplex mode (the FULLDUPLEX bit is cleared in MACCONTROL). When receive flow control is enabled and triggered, the EMAC generates collisions for received frames. The jam sequence transmitted is the 12-byte sequence C3.C3.C3.C3.C3.C3.C3.C3.C3.C3.C3.C3h. The jam sequence begins no later than approximately as the source address starts to be received. Note that these forced collisions are not limited to a maximum of 16 consecutive collisions, and are independent of the normal back-off algorithm.

Receive flow control does not depend on the value of the incoming frame destination address. A collision is generated for any incoming packet, regardless of the destination address, if any EMAC enabled channel's free buffer register value is less than or equal to the channel's flow threshold value.

32.2.10.1.3.2 IEEE 802.3x-Based Receive Buffer Flow Control

IEEE 802.3x-based receive buffer flow control provides a means of preventing frame reception when the EMAC is operating in full-duplex mode (the FULLDUPLEX bit is set in MACCONTROL). When receive flow control is enabled and triggered, the EMAC transmits a pause frame to request that the sending station stop transmitting for the period indicated within the transmitted pause frame.

The EMAC transmits a pause frame to the reserved multicast address at the first available opportunity (immediately if currently idle or following the completion of the frame currently being transmitted). The pause frame contains the maximum possible value for the pause time (FFFFh). The EMAC counts the receive pause frame time (decrements FF00h to 0) and retransmits an outgoing pause frame, if the count reaches 0. When the flow control request is removed, the EMAC transmits a pause frame with a zero pause time to cancel the pause request.

Note that transmitted pause frames are only a request to the other end station to stop transmitting. Frames that are received during the pause interval are received normally (provided the receive FIFO is not full).

Pause frames are transmitted if enabled and triggered, regardless of whether or not the EMAC is observing the pause time period from an incoming pause frame.

The EMAC transmits pause frames as described below:

- The 48-bit reserved multicast destination address 01.80.C2.00.00.01h.
- The 48-bit source address (set using the MACSRCADDRLO and MACSRCADDRHI registers).
- The 16-bit length/type field containing the value 88.08h.
- The 16-bit pause opcode equal to 00.01h.
- The 16-bit pause time value of FF.FFh. A pause-quantum is 512 bit-times. Pause frames sent to cancel a pause request have a pause time value of 00.00h.
- Zero padding to 64-byte data length (EMAC transmits only 64-byte pause frames).

- The 32-bit frame-check sequence (CRC word).

All quantities are hexadecimal and are transmitted most-significant byte first. The least-significant bit (LSB) is transferred first in each byte.

If the RXBUFFERFLOWEN bit in MACCONTROL is cleared to 0 while the pause time is nonzero, then the pause time is cleared to 0 and a zero count pause frame is sent.

32.2.10.2 Data Transmission

The EMAC passes data to the PHY from the transmit FIFO (when enabled). Data is synchronized to the transmit clock rate. Transmission begins when there are TXCELLTHRESH cells of 64 bytes each, or a complete packet, in the FIFO.

32.2.10.2.1 Transmit Control

A jam sequence is output if a collision is detected on a transmit packet. If the collision was late (after the first 64 bytes have been transmitted), the collision is ignored. If the collision is not late, the controller will back off before retrying the frame transmission. When operating in full-duplex mode, the carrier sense (MII_CRS) and collision-sensing (MII_COL) modes are disabled.

32.2.10.2.2 CRC Insertion

If the SOP buffer descriptor PASSCRC flag is cleared, the EMAC generates and appends a 32-bit Ethernet CRC onto the transmitted data. For the EMAC-generated CRC case, a CRC (or placeholder) at the end of the data is allowed but not required. The buffer byte count value should not include the CRC bytes, if they are present.

If the SOP buffer descriptor PASSCRC flag is set, then the last four bytes of the transmit data are transmitted as the frame CRC. The four CRC data bytes should be the last four bytes of the frame and should be included in the buffer byte count value. The MAC performs no error checking on the outgoing CRC.

32.2.10.2.3 Adaptive Performance Optimization (APO)

The EMAC incorporates adaptive performance optimization (APO) logic that may be enabled by setting the TXPACE bit in the MAC control register (MACCONTROL). Transmission pacing to enhance performance is enabled when the TXPACE bit is set. Adaptive performance pacing introduces delays into the normal transmission of frames, delaying transmission attempts between stations, reducing the probability of collisions occurring during heavy traffic (as indicated by frame deferrals and collisions), thereby, increasing the chance of successful transmission.

When a frame is deferred, suffers a single collision, multiple collisions, or excessive collisions, the pacing counter is loaded with an initial value of 31. When a frame is transmitted successfully (without experiencing a deferral, single collision, multiple collision, or excessive collision), the pacing counter is decremented by 1, down to 0.

With pacing enabled, a new frame is permitted to immediately (after one interpacket gap) attempt transmission only if the pacing counter is 0. If the pacing counter is nonzero, the frame is delayed by the pacing delay of approximately four interpacket gap (IPG)delays. APO only affects the IPG preceding the first attempt at transmitting a frame; APO does not affect the back-off algorithm for retransmitted frames.

32.2.10.2.4 Interpacket-Gap (IPG) Enforcement

The measurement reference for the IPG of 96 bit times is changed depending on frame traffic conditions. If a frame is successfully transmitted without collision and MII_CRS is deasserted within approximately 48 bit times of MII_TXEN being deasserted, then 96 bit times is measured from MII_TXEN. If the frame suffered a collision or MII_CRS is not deasserted until more than approximately 48 bit times after MII_TXEN is deasserted, then 96 bit times (approximately, but not less) is measured from MII_CRS.

32.2.10.2.5 Back Off

The EMAC implements the 802.3 binary exponential back-off algorithm.

32.2.10.2.6 Transmit Flow Control

Incoming pause frames are acted upon, when enabled, to prevent the EMAC from transmitting any further frames. Incoming pause frames are only acted upon when the FULLDUPLEX and TXFLOWEN bits in the MAC control register (MACCONTROL) are set. Pause frames are not acted upon in half-duplex mode. Pause frame action is taken if enabled, but normally the frame is filtered and not transferred to memory. MAC control frames are transferred to memory, if the RXCMFEN bit in the receive multicast/broadcast/promiscuous channel enable register (RXMBPENABLE) is set. The TXFLOWEN and FULLDUPLEX bits affect whether or not MAC control frames are acted upon, but they have no affect upon whether or not MAC control frames are transferred to memory or filtered.

Pause frames are a subset of MAC control frames with an opcode field of 0001h. Incoming pause frames are only acted upon by the EMAC if:

- TXFLOWEN bit is set in MACCONTROL
- The frame’s length is 64 to RXMAXLEN bytes inclusive
- The frame contains no CRC error or align/code errors

The pause time value from valid frames is extracted from the two bytes following the opcode. The pause time is loaded into the EMAC transmit pause timer and the transmit pause time period begins. If a valid pause frame is received during the transmit pause time period of a previous transmit pause frame then:

- If the destination address is not equal to the reserved multicast address or any enabled or disabled unicast address, then the transmit pause timer immediately expires, or
- If the new pause time value is 0, then the transmit pause timer immediately expires, else
- The EMAC transmit pause timer immediately is set to the new pause frame pause time value. (Any remaining pause time from the previous pause frame is discarded).

If the TXFLOWEN bit in MACCONTROL is cleared, then the pause timer immediately expires.

The EMAC does not start the transmission of a new data frame any sooner than 512 bit-times after a pause frame with a nonzero pause time has finished being received (MII_RXDV going inactive). No transmission begins until the pause timer has expired (the EMAC may transmit pause frames in order to initiate outgoing flow control). Any frame already in transmission when a pause frame is received is completed and unaffected.

Incoming pause frames consist of:

- A 48-bit destination address equal to one of the following:
  - The reserved multicast destination address 01.80.C2.00.00.01h
  - Any EMAC 48-bit unicast address. Pause frames are accepted, regardless of whether the channel is enabled or not.
- The 16-bit length/type field containing the value 88.08h.
- The 48-bit source address of the transmitting device.
- The 16-bit pause opcode equal to 00.01h.
- The 16-bit pause time. A pause-quantum is 512 bit-times.
- Padding to 64-byte data length.
- The 32-bit frame-check sequence (CRC word).

All quantities are hexadecimal and are transmitted most-significant byte first. The least-significant bit (LSB) is transferred first in each byte.

The padding is required to make up the frame to a minimum of 64 bytes. The standard allows pause frames longer than 64 bytes to be discarded or interpreted as valid pause frames. The EMAC recognizes any pause frame between 64 bytes and RXMAXLEN bytes in length.

32.2.10.2.7 Speed, Duplex, and Pause Frame Support

The MAC operates at 10 Mbps or 100 Mbps, in half-duplex or full-duplex mode, and with or without pause frame support as configured by the host.

32.2.11 Packet Receive Operation

32.2.11.1 Receive DMA Host Configuration

To configure the receive DMA for operation the host must:

- Initialize the receive addresses.
- Initialize the receive channel *n* DMA head descriptor pointer registers (RX*n*HDP) to 0.
- Write the MAC address hash *n* registers (MACHASH1 and MACHASH2), if multicast addressing is desired.
- If flow control is to be enabled, initialize:
  - the receive channel *n* free buffer count registers (RX*n*FREEBUFFER)
  - the receive channel *n* flow control threshold register (RX*n*FLOWTHRESH)
  - the receive filter low priority frame threshold register (RXFILTERLOWTHRESH)
- Enable the desired receive interrupts using the receive interrupt mask set register (RXINTMASKSET) and the receive interrupt mask clear register (RXINTMASKCLEAR).
- Set the appropriate configuration bits in the MAC control register (MACCONTROL).
- Write the receive buffer offset register (RXBUFFEROFFSET) value (typically 0).
- Setup the receive channel(s) buffer descriptors and initialize RX*n*HDP.
- Enable the receive DMA controller by setting the RXEN bit in the receive control register (RXCONTROL).
- Configure and enable the receive operation, as desired, in the receive multicast/broadcast/promiscuous channel enable register (RXMBPENABLE) and by using the receive unicast set register (RXUNICASTSET) and the receive unicast clear register (RXUNICASTCLEAR).

32.2.11.2 Receive Channel Enabling

Each of the eight receive channels has an enable bit (RXCH*n*EN) in the receive unicast set register (RXUNICASTSET) that is controlled using RXUNICASTSET and the receive unicast clear register (RXUNICASTCLEAR). The RXCH*n*EN bits determine whether the given channel is enabled (when set to 1) to receive frames with a matching unicast or multicast destination address.

The RXBROADEN bit in the receive multicast/broadcast/promiscuous channel enable register (RXMBPENABLE) determines if broadcast frames are enabled or filtered. If broadcast frames are enabled (when set to 1), then they are copied to only a single channel selected by the RXBROADCH bit in RXMBPENABLE.

The RXMULTEN bit in RXMBPENABLE determines if hash matching multicast frames are enabled or filtered. Incoming multicast addresses (group addresses) are hashed into an index in the hash table. If the indexed bit is set then the frame hash matches and will be transferred to the channel selected by the RXMULTCH bit in RXMBPENABLE when multicast frames are enabled. The multicast hash bits are set in the MAC address hash *n* registers (MACHASH1 and MACHASH2).

The RXPROMCH bit in RXMBPENABLE selects the promiscuous channel to receive frames selected by the RXCMFEN, RXCSFEN, RXCEFEN, and RXCAFEN bits. These four bits allow reception of MAC control frames, short frames, error frames, and all frames (promiscuous), respectively.

32.2.11.3 Receive Address Matching

All eight MAC addresses corresponding to the eight receive channels share the upper 40 bits. Only the lower byte is unique for each address. All eight receive addresses should be initialized, because pause frames are acted upon regardless of whether a channel is enabled or not.

A MAC address is written by first writing the address number (channel) to be written into the MAC index register (MACINDEX). The upper 32 bits of address are then written to the MAC address high bytes register (MACADDRHI), which is followed by writing the lower 16 bits of address to the MAC address low bytes register (MACADDRLO). Since all eight MAC addresses share the upper 40 bits of address, MACADDRHI needs to be written only the first time (for the first channel configured).

32.2.11.4 Hardware Receive QOS Support

Hardware receive quality of service (QOS) is supported, when enabled, by the Tag Protocol Identifier format and the associated Tag Control Information (TCI) format priority field. When the incoming frame length/type value is equal to 81.00h, the EMAC recognizes the frame as an Ethernet Encoded Tag Protocol Type. The two octets immediately following the protocol type contain the 16-bit TCI field. Bits 15-13 of the TCI field contain the received frames priority (0 to 7). The received frame is a low-priority frame, if the priority value is 0 to 3; the received frame is a high-priority frame, if the priority value is 4 to 7. All frames that have a length/type field value not equal to 81.00h are low-priority frames. Received frames that contain priority information are determined by the EMAC as:

- A 48-bit (6 bytes) destination address equal to:
  - The destination station's individual unicast address.
  - The destination station's multicast address (MACHASH1 and MACHASH2).
  - The broadcast address of all ones.
- A 48-byte (6 bytes) source address.
- The 16-bit (2 bytes) length/type field containing the value 81.00h.
- The 16-bit (2 bytes) TCI field with the priority field in the upper 3 bits.
- Data bytes
- The 4 bytes CRC.

The receive filter low priority frame threshold register (RXFILTERLOWTHRESH) and the receive channel *n* free buffer count registers (RX*n*FREEBUFFER) are used in conjunction with the priority information to implement receive hardware QOS. Low-priority frames are filtered if the number of free buffers (RX*n*FREEBUFFER) for the frame channel is less than or equal to the filter low threshold (RXFILTERLOWTHRESH) value. Hardware QOS is enabled by the RXQOSEN bit in the receive multicast/broadcast/promiscuous channel enable register (RXMBPENABLE).

32.2.11.5 Host Free Buffer Tracking

The host must track free buffers for each enabled channel (including unicast, multicast, broadcast, and promiscuous), if receive QOS or receive flow control is used. Disabled channel free buffer values are do not cares. During initialization, the host should write the number of free buffers for each enabled channel to the appropriate receive channel *n* free buffer count registers (RX*n*FREEBUFFER). The EMAC decrements the appropriate channel’s free buffer value for each buffer used. When the host reclaims the frame buffers, the host should write the channel free buffer register with the number of reclaimed buffers (write to increment). There are a maximum of 65,535 free buffers available. RX*n*FREEBUFFER only needs to be updated by the host if receive QOS or flow control is used.

32.2.11.6 Receive Channel Teardown

The host commands a receive channel teardown by writing the channel number to the receive teardown register (RXTEARDOWN). When a teardown command is issued to an enabled receive channel, the following occurs:

- Any current frame in reception completes normally.
- The TDOWNCMPLT flag is set in the next buffer descriptor in the chain, if there is one.
- The channel head descriptor pointer is cleared to 0.
- A receive interrupt for the channel is issued to the host.
- The corresponding receive channel *n* completion pointer register (RX*n*CP) contains the value FFFF FFFCh.

Channel teardown may be commanded on any channel at any time. The host is informed of the teardown completion by the set teardown complete (TDOWNCMPLT) buffer descriptor bit. The EMAC does not clear any channel enables due to a teardown command. A teardown command to an inactive channel issues an interrupt that software should acknowledge with an FFFF FFFCh acknowledge value to RX*n*CP (note that there is no buffer descriptor in this case). Software may read RX*n*CP to determine if the interrupt was due to a commanded teardown. The read value is FFFF FFFCh, if the interrupt was due to a teardown command.

32.2.11.7 Receive Frame Classification

Received frames are proper (good) frames, if they are between 64 bytes and the value in the receive maximum length register (RXMAXLEN) bytes in length (inclusive) and contain no code, align, or CRC errors.

Received frames are long frames, if their frame count exceeds the value in RXMAXLEN. The RXMAXLEN reset (default) value is 5EEh (1518 in decimal). Long received frames are either oversized or jabber frames. Long frames with no errors are oversized frames; long frames with CRC, code, or alignment errors are jabber frames.

Received frames are short frames, if their frame count is less than 64 bytes. Short frames that address match and contain no errors are undersized frames; short frames with CRC, code, or alignment errors are fragment frames. If the frame length is less than or equal to 20, then the frame CRC is passed, regardless of whether the RXPASSCRC bit is set or cleared in the receive multicast/broadcast/promiscuous channel enable register (RXMBPENABLE).

A received long packet always contains RXMAXLEN number of bytes transferred to memory (if the RXCEFEN bit is set in RXMBPENABLE), regardless of the value of the RXPASSCRC bit. Following is an example with RXMAXLEN set to 1518:

- If the frame length is 1518, then the packet is not a long packet and there are 1514 or 1518 bytes transferred to memory depending on the value of the RXPASSCRC bit.
- If the frame length is 1519, there are 1518 bytes transferred to memory regardless of the RXPASSCRC bit value. The last three bytes are the first three CRC bytes.
- If the frame length is 1520, there are 1518 bytes transferred to memory regardless of the RXPASSCRC bit value. The last two bytes are the first two CRC bytes.
- If the frame length is 1521, there are 1518 bytes transferred to memory regardless of the RXPASSCRC bit value. The last byte is the first CRC byte.
- If the frame length is 1522, there are 1518 bytes transferred to memory. The last byte is the last data byte.

32.2.11.8 Promiscuous Receive Mode

When the promiscuous receive mode is enabled by setting the RXCAFEN bit in the receive multicast/broadcast/promiscuous channel enable register (RXMBPENABLE), nonaddress matching frames that would normally be filtered are transferred to the promiscuous channel. Address matching frames that would normally be filtered due to errors are transferred to the address match channel when the RXCAFEN and RXCEFEN bits in RXMBPENABLE are set. A frame is considered to be an address matching frame only if it is enabled to be received on a unicast, multicast, or broadcast channel. Frames received to disabled unicast, multicast, or broadcast channels are considered nonaddress matching.

MAC control frames address match only if the RXCMFEN bit in RXMBPENABLE is set. The RXCEFEN and RXCSFEN bits in RXMBPENABLE determine whether error frames are transferred to memory or not, but they do not determine whether error frames are address matching or not. Short frames are a special type of error frames.

A single channel is selected as the promiscuous channel by the RXPROMCH bit in RXMBPENABLE. The promiscuous receive mode is enabled by the RXCMFEN, RXCEFEN, RXCSFEN, and RXCAFEN bits in RXMBPENABLE. Table 32-7 shows the effects of the promiscuous enable bits. Proper frames are frames that are between 64 bytes and the value in the receive maximum length register (RXMAXLEN) bytes in length inclusive and contain no code, align, or CRC errors.

Table 32-7. Receive Frame Treatment Summary

Address Match	RXCAFEN	RXCEFEN	RXCMFEN	RXCSFEN	Receive Frame Treatment
0	0	X	X	X	No frames transferred.
0	1	0	0	0	Proper frames transferred to promiscuous channel.
0	1	0	0	1	Proper/undersized data frames transferred to promiscuous channel.
0	1	0	1	0	Proper data and control frames transferred to promiscuous channel.
0	1	0	1	1	Proper/undersized data and control frames transferred to promiscuous channel.
0	1	1	0	0	Proper/oversize/jabber/code/align/CRC data frames transferred to promiscuous channel. No control or undersized/fragment frames are transferred.
0	1	1	0	1	Proper/undersized/fragment/oversize/jabber/code/align/CRC data frames transferred to promiscuous channel. No control frames are transferred.
0	1	1	1	0	Proper/oversize/jabber/code/align/CRC data and control frames transferred to promiscuous channel. No undersized frames are transferred.
0	1	1	1	1	All nonaddress matching frames with and without errors transferred to promiscuous channel.
1	X	0	0	0	Proper data frames transferred to address match channel.
1	X	0	0	1	Proper/undersized data frames transferred to address match channel.
1	X	0	1	0	Proper data and control frames transferred to address match channel.
1	X	0	1	1	Proper/undersized data and control frames transferred to address match channel.
1	X	1	0	0	Proper/oversize/jabber/code/align/CRC data frames transferred to address match channel. No control or undersized frames are transferred.
1	X	1	0	1	Proper/oversize/jabber/fragment/undersized/code/align/CRC data frames transferred to address match channel. No control frames are transferred.
1	X	1	1	0	Proper/oversize/jabber/code/align/CRC data and control frames transferred to address match channel. No undersized/fragment frames are transferred.
1	X	1	1	1	All address matching frames with and without errors transferred to the address match channel

32.2.11.9 Receive Overrun

The types of receive overrun are:

- FIFO start of frame overrun (FIFO_SOF)
- FIFO middle of frame overrun (FIFO_MOF)
- DMA start of frame overrun (DMA_SOF)
- DMA middle of frame overrun (DMA_MOF)

The statistics counters used to track these types of receive overrun are:

- Receive start of frame overruns register (RXSOFOVERRUNS)
- Receive middle of frame overruns register (RXMOFOVERRUNS)
- Receive DMA overruns register (RXDMAOVERRUNS)

Start of frame overruns happen when there are no resources available when frame reception begins. Start of frame overruns increment the appropriate overrun statistic(s) and the frame is filtered.

Middle of frame overruns happen when there are some resources to start the frame reception, but the resources run out during frame reception. In normal operation, a frame that overruns after starting the frame reception is filtered and the appropriate statistic(s) are incremented; however, the RXCEFEN bit in the receive multicast/broadcast/promiscuous channel enable register (RXMBPENABLE) affects overrun frame treatment. Table 32-8 shows how the overrun condition is handled for the middle of frame overrun.

Table 32-8. Middle of Frame Overrun Treatment

Address Match	RXCAFEN	RXCEFEN	Middle of Frame Overrun Treatment
0	0	X	Overrun frame filtered.
0	1	0	Overrun frame filtered.
0	1	1	As much frame data as possible is transferred to the promiscuous channel until overrun. The appropriate overrun statistic(s) is incremented and the OVERRUN and NOMATCH flags are set in the SOP buffer descriptor. Note that the RXMAXLEN number of bytes cannot be reached for an overrun to occur (it would be truncated and be a jabber or oversize).
1	X	0	Overrun frame filtered with the appropriate overrun statistic(s) incremented.
1	X	1	As much frame data as possible is transferred to the address match channel until overrun. The appropriate overrun statistic(s) is incremented and the OVERRUN flag is set in the SOP buffer descriptor. Note that the RXMAXLEN number of bytes cannot be reached for an overrun to occur (it would be truncated).

32.2.12 Packet Transmit Operation

The transmit DMA is an eight channel interface. Priority between the eight queues may be either fixed or round-robin as selected by the TXPTYPE bit in the MAC control register (MACCONTROL). If the priority type is fixed, then channel 7 has the highest priority and channel 0 has the lowest priority. Round-robin priority proceeds from channel 0 to channel 7.

32.2.12.1 Transmit DMA Host Configuration

To configure the transmit DMA for operation the host must perform:

- Write the MAC source address low bytes register (MACSRCADDRLO) and the MAC source address high bytes register (MACSRCADDRHI) (used for pause frames on transmit).
- Initialize the transmit channel *n* DMA head descriptor pointer registers (TX*n*HDP) to 0.
- Enable the desired transmit interrupts using the transmit interrupt mask set register (TXINTMASKSET) and the transmit interrupt mask clear register (TXINTMASKCLEAR).
- Set the appropriate configuration bits in the MAC control register (MACCONTROL).
- Setup the transmit channel(s) buffer descriptors in host memory.
- Enable the transmit DMA controller by setting the TXEN bit in the transmit control register (TXCONTROL).
- Write the appropriate TX*n*HDP with the pointer to the first descriptor to start transmit operations.

32.2.12.2 Transmit Channel Teardown

The host commands a transmit channel teardown by writing the channel number to the transmit teardown register (TXTEARDOWN). When a teardown command is issued to an enabled transmit channel, the following occurs:

- Any frame currently in transmission completes normally.
- The TDOWNCMPLT flag is set in the next SOP buffer descriptor in the chain, if there is one.
- The channel head descriptor pointer is cleared to 0.
- A transmit interrupt is issued to inform the host of the channel teardown.
- The corresponding transmit channel *n* completion pointer register (TX*n*CP) contains the value FFFF FFFCh.
- The host should acknowledge a teardown interrupt with an FFFF FFFCh acknowledge value.

Channel teardown may be commanded on any channel at any time. The host is informed of the teardown completion by the set teardown complete (TDOWNCMPLT) buffer descriptor bit. The EMAC does not clear any channel enables due to a teardown command. A teardown command to an inactive channel issues an interrupt that software should acknowledge with an FFFF FFFCh acknowledge value to TX*n*CP (note that there is no buffer descriptor in this case). Software may read the interrupt acknowledge location (TX*n*CP) to determine if the interrupt was due to a commanded teardown. The read value is FFFF FFFCh, if the interrupt was due to a teardown command.

32.2.13 Receive and Transmit Latency

The transmit and receive FIFOs each contain three 64-byte cells. The EMAC begins transmission of a packet on the wire after TXCELLTHRESH (configurable through the FIFO control register) cells, or a complete packet, are available in the FIFO.

Transmit underrun cannot occur for packet sizes of TXCELLTHRESH times 64 bytes (or less). For larger packet sizes, transmit underrun occurs if the memory latency is greater than the time required to transmit a 64-byte cell on the wire; this is 5.12 μs in 100 Mbps mode and 51.2 μs in 10 Mbps mode. The memory latency time includes all buffer descriptor reads for the entire cell data.

Receive overrun is prevented if the receive memory cell latency is less than the time required to transmit a 64-byte cell on the wire: 5.12 μs in 100 Mbps mode, or 51.2 μs in 10 Mbps mode. The latency time includes any required buffer descriptor reads for the cell data.

Latency to system’s internal and external RAM can be controlled through the use of the transfer node priority allocation register available at the device level. Latency to descriptor RAM is low because RAM is local to the EMAC, as it is part of the EMAC control module.

32.2.14 Transfer Node Priority

The device contains a chip-level master priority register that is used to set the priority of the transfer node used in issuing memory transfer requests to system memory.

Although the EMAC has internal FIFOs to help alleviate memory transfer arbitration problems, the average transfer rate of data read and written by the EMAC to internal or external processor memory must be at least that of the Ethernet wire rate. In addition, the internal FIFO system can not withstand a single memory latency event greater than the time it takes to fill or empty a TXCELLTHRESH number of internal 64 byte FIFO cells.

For 100 Mbps operation, these restrictions translate into the following rules:

- The short-term average, each 64-byte memory read/write request from the EMAC must be serviced in no more than 5.12 μs.
- Any single latency event in request servicing can be no longer than (5.12 × TXCELLTHRESH) μs.

32.2.15 Reset Considerations

32.2.15.1 Software Reset Considerations

The peripheral clock is controlled by the Global Clock Module (GCM), while the reset to the EMAC, MDIO and EMAC control module is controlled by the system module. See the "Architecture" chapter of the Technical Reference Manual for more on how to enable or disable the peripheral clock to the EMAC, MDIO and EMAC control module. For more on how the EMAC, MDIO, and EMAC control module are disabled or placed in reset at runtime, see Section 32.2.18.

Within the peripheral itself, the EMAC component of the Ethernet MAC peripheral can be placed in a reset state by writing to the soft reset register (SOFTRESET). Writing a 1 to the SOFTRESET bit causes the EMAC logic to be reset and the register values to be set to their default values. Software reset occurs when the receive and transmit DMA controllers are in an idle state to avoid locking up the configuration bus; it is the responsibility of the software to verify that there are no pending frames to be transferred. After writing a 1 to the SOFTRESET bit, it may be polled to determine if the reset has occurred. If a 1 is read, the reset has not yet occurred; if a 0 is read, then a reset has occurred.

After a software reset operation, all the EMAC registers need to be reinitialized for proper data transmission, including the FULLDUPLEX bit setting in the MAC control register (MACCONTROL).

Unlike the EMAC module, the MDIO and EMAC control modules cannot be placed in reset from a register inside their memory map.

32.2.15.2 Hardware Reset Considerations

When a hardware reset occurs, the EMAC peripheral has its register values reset and all the components return to their default state. After the hardware reset, the EMAC needs to be initialized before being able to resume its data transmission, as described in Section 32.2.16.

A hardware reset is the only means of recovering from the error interrupts (HOSTPEND), which are triggered by errors in packet buffer descriptors. Before doing a hardware reset, you should inspect the error codes in the MAC status register (MACSTATUS) that gives information about the type of software error that needs to be corrected. For detailed information on error interrupts, see Section 32.2.17.1.4.

32.2.16 Initialization

32.2.16.1 Enabling the EMAC/MDIO Peripheral

When the device is powered on, the EMAC peripheral becomes enabled as soon as the system reset is released, and the EMAC peripheral registers are set to their default values. The application software can configure the EMAC peripheral registers as required.

32.2.16.2 EMAC Control Module Initialization

The EMAC control module is used for global interrupt enables and to pace interrupts using 1ms time windows. There is also an 8K block of CPPI RAM local to the EMAC that is used to hold packet buffer descriptors.

Note that although the EMAC control module and the EMAC module have slightly different functions, in practice, the type of maintenance performed on the EMAC control module is more commonly conducted from the EMAC module software (as opposed to the MDIO module).

The initialization of the EMAC control module consists of two parts:

1. Configuration of the interrupt to the CPU.
2. Initialization of the EMAC control module:
   - Setting the interrupt pace counts using the EMAC control module registers INTCONTROL, C0RXIMAX, and C0TXIMAX
   - Initializing the EMAC and MDIO modules
   - Enabling interrupts in the EMAC control module using the EMAC control module interrupt control registers C0RXTHRESHEN, C0RXEN, C0TXEN, and C0MISCEN.

The process of mapping the EMAC interrupts to the CPU is done through the Vectored Interrupt Manager (VIM). Once the interrupt is mapped to a CPU interrupt, general masking and unmasking of interrupts (to control reentrancy) should be done at the chip level by manipulating the interrupt core enable mask registers.

32.2.16.3 MDIO Module Initialization

The MDIO module is used to initially configure and monitor one or more external PHY devices. Other than initializing the software state machine (details on this state machine can be found in the IEEE 802.3 standard), all that needs to be done for the MDIO module is to enable the MDIO engine and to configure the clock divider. To set the clock divider, supply an MDIO clock of 1 MHz. For example, if the peripheral clock is 50 MHz, the divider can be set to 50.

Both the state machine enable and the MDIO clock divider are controlled through the MDIO control register (CONTROL). If none of the potentially connected PHYs require the access preamble, the PREAMBLE bit in CONTROL can also be set to speed up PHY register access.

If the MDIO module is to operate on an interrupt basis, the interrupts can be enabled at this time using the MDIO user command complete interrupt mask set register (USERINTMASKSET) for register access and the MDIO user PHY select register (USERPHYSEL*n*) if a target PHY is already known.

Once the MDIO state machine has been initialized and enabled, it starts polling all 32 PHY addresses on the MDIO bus, looking for an active PHY. Since it can take up to 50 μs to read one register, it can be some time before the MDIO module provides an accurate representation of whether a PHY is available. Also, a PHY can take up to 3 seconds to negotiate a link. Thus, it is advisable to run the MDIO software off a time-based event rather than polling.

For more information on PHY control registers, see your PHY device documentation.

32.2.16.4 EMAC Module Initialization

The EMAC module is used to send and receive data packets over the network. This is done by maintaining up to eight transmit and receive descriptor queues. The EMAC module configuration must also be kept up-to-date based on PHY negotiation results returned from the MDIO module. Most of the work in developing an application or device driver for Ethernet is programming this module.

The following is the initialization procedure a device driver would follow to get the EMAC to the state where it is ready to receive and send Ethernet packets. Some of these steps are not necessary when performed immediately after device reset.

1. If enabled, clear the device interrupt enable bits in the EMAC control module interrupt control registers C0RXTHRESHEN, C0RXEN, C0TXEN, and C0MISCEN.
2. Clear the MAC control register (MACCONTROL), receive control register (RXCONTROL), and transmit control register (TXCONTROL) (not necessary immediately after reset).
3. Initialize all 16 header descriptor pointer registers (RX*n*HDP and TX*n*HDP) to 0.
4. Clear all 36 statistics registers by writing 0 (not necessary immediately after reset).
5. Setup the local Ethernet MAC address by programming the MAC index register (MACINDEX), MAC address high bytes register (MACADDRHI), and MAC address low bytes register (MACADDRLO). Be sure to program all eight MAC address registers - whether the receive channel is to be enabled or not. Duplicate the same MAC address across all unused channels. When using more than one receive channel, start with channel 0 and progress upwards.
6. If buffer flow control is to be enabled, initialize the receive channel *n* free buffer count registers (RX*n*FREEBUFFER), receive channel *n* flow control threshold register (RX*n*FLOWTHRESH), and receive filter low priority frame threshold register (RXFILTERLOWTHRESH).
7. Most device drivers open with no multicast addresses, so clear the MAC address hash registers (MACHASH1 and MACHASH2) to 0.
8. Write the receive buffer offset register (RXBUFFEROFFSET) value (typically 0).
9. Initially clear all unicast channels by writing FFh to the receive unicast clear register (RXUNICASTCLEAR). If unicast is desired, it can be enabled now by writing the receive unicast set register (RXUNICASTSET). Some drivers will default to unicast on device open while others will not.
10. Setup the receive multicast/broadcast/promiscuous channel enable register (RXMBPENABLE) with an initial configuration. The configuration is based on the current receive filter settings of the device driver. Some drivers may enable things like broadcast and multicast packets immediately, while others may not.
11. Set the appropriate configuration bits in MACCONTROL (do not set the GMIIEN bit yet).
12. Clear all unused channel interrupt bits by writing the receive interrupt mask clear register (RXINTMASKCLEAR) and the transmit interrupt mask clear register (TXINTMASKCLEAR).
13. Enable the receive and transmit channel interrupt bits in the receive interrupt mask set register (RXINTMASKSET) and the transmit interrupt mask set register (TXINTMASKSET) for the channels to be used, and enable the HOSTMASK and STATMASK bits using the MAC interrupt mask set register (MACINTMASKSET).
14. Initialize the receive and transmit descriptor list queues.
15. Prepare receive by writing a pointer to the head of the receive buffer descriptor list to RX*n*HDP.
16. Enable the receive and transmit DMA controllers by setting the RXEN bit in RXCONTROL and the TXEN bit in TXCONTROL. Then set the GMIIEN bit in MACCONTROL.
17. Enable the device interrupt in EMAC control module registers C0RXTHRESHEN, C0RXEN, C0TXEN, and C0MISCEN.

32.2.17 *Interrupt Support*

32.2.17.1 EMAC Module Interrupt Events and Requests

The EMAC module generates 26 interrupt events:

- TXPEND*n*: Transmit packet completion interrupt for transmit channels 0 through 7
- RXPEND*n*: Receive packet completion interrupt for receive channels 0 through 7
- RXTHRESHPEND*n*: Receive packet completion interrupt for receive channels 0 through 7 when flow control is enabled and the number of free buffers is below the threshold
- STATPEND: Statistics interrupt
- HOSTPEND: Host error interrupt

32.2.17.1.1 Transmit Packet Completion Interrupts

The transmit DMA engine has eight channels, with each channel having a corresponding interrupt (TXPEND*n*). The transmit interrupts are level interrupts that remain asserted until cleared by the CPU.

Each of the eight transmit channel interrupts may be individually enabled by setting the appropriate bit in the transmit interrupt mask set register (TXINTMASKSET) to 1. Each of the eight transmit channel interrupts may be individually disabled by clearing the appropriate bit by writing a 1 to the transmit interrupt mask clear register (TXINTMASKCLEAR). The raw and masked transmit interrupt status may be read by reading the transmit interrupt status (unmasked) register (TXINTSTATRAW) and the transmit interrupt status (masked) register (TXINTSTATMASKED), respectively.

When the EMAC completes the transmission of a packet, the EMAC issues an interrupt to the CPU (via the EMAC control module) when it writes the packet’s last buffer descriptor address to the appropriate channel queue’s transmit completion pointer located in the state RAM block. The interrupt is generated by the write when enabled by the interrupt mask, regardless of the value written.

Upon interrupt reception, the CPU processes one or more packets from the buffer chain and then acknowledges an interrupt by writing the address of the last buffer descriptor processed to the queue’s associated transmit completion pointer in the transmit DMA state RAM.

The data written by the host (buffer descriptor address of the last processed buffer) is compared to the data in the register written by the EMAC port (address of last buffer descriptor used by the EMAC). If the two values are not equal (which means that the EMAC has transmitted more packets than the CPU has processed interrupts for), the transmit packet completion interrupt signal remains asserted. If the two values are equal (which means that the host has processed all packets that the EMAC has transferred), the pending interrupt is cleared. The value that the EMAC is expecting is found by reading the transmit channel *n* completion pointer register (TX*n*CP).

The EMAC write to the completion pointer actually stores the value in the state RAM. The CPU written value does not actually change the register value. The host written value is compared to the register content (which was written by the EMAC) and if the two values are equal then the interrupt is removed; otherwise, the interrupt remains asserted. The host may process multiple packets prior to acknowledging an interrupt, or the host may acknowledge interrupts for every packet.

The application software must acknowledge the EMAC control module after processing packets by writing the appropriate C0RX key to the EMAC End-Of-Interrupt Vector register (MACEOIVECTOR). See Section 32.5.12 for the acknowledge key values.

32.2.17.1.2 Receive Packet Completion Interrupts

The receive DMA engine has eight channels, which each channel having a corresponding interrupt (RXPEND*n*). The receive interrupts are level interrupts that remain asserted until cleared by the CPU.

Each of the eight receive channel interrupts may be individually enabled by setting the appropriate bit in the receive interrupt mask set register (RXINTMASKSET) to 1. Each of the eight receive channel interrupts may be individually disabled by clearing the appropriate bit by writing a 1 in the receive interrupt mask clear register (RXINTMASKCLEAR). The raw and masked receive interrupt status may be read by reading the receive interrupt status (unmasked) register (RXINTSTATRAW) and the receive interrupt status (masked) register (RXINTSTATMASKED), respectively.

When the EMAC completes a packet reception, the EMAC issues an interrupt to the CPU by writing the packet's last buffer descriptor address to the appropriate channel queue's receive completion pointer located in the state RAM block. The interrupt is generated by the write when enabled by the interrupt mask, regardless of the value written.

Upon interrupt reception, the CPU processes one or more packets from the buffer chain and then acknowledges one or more interrupt(s) by writing the address of the last buffer descriptor processed to the queue's associated receive completion pointer in the receive DMA state RAM.

The data written by the host (buffer descriptor address of the last processed buffer) is compared to the data in the register written by the EMAC (address of last buffer descriptor used by the EMAC). If the two values are not equal (which means that the EMAC has received more packets than the CPU has processed interrupts for), the receive packet completion interrupt signal remains asserted. If the two values are equal (which means that the host has processed all packets that the EMAC has received), the pending interrupt is de-asserted. The value that the EMAC is expecting is found by reading the receive channel *n* completion pointer register (RX*n*CP).

The EMAC write to the completion pointer actually stores the value in the state RAM. The CPU written value does not actually change the register value. The host written value is compared to the register content (which was written by the EMAC) and if the two values are equal then the interrupt is removed; otherwise, the interrupt remains asserted. The host may process multiple packets prior to acknowledging an interrupt, or the host may acknowledge interrupts for every packet.

The application software must acknowledge the EMAC control module after processing packets by writing the appropriate C0TX key to the EMAC End-Of-Interrupt Vector register (MACEOIVECTOR). See Section 32.5.12 for the acknowledge key values.

32.2.17.1.3 Statistics Interrupt

The statistics level interrupt (STATPEND) is issued when any statistics value is greater than or equal to 8000 0000h, if enabled by setting the STATMASK bit in the MAC interrupt mask set register (MACINTMASKSET) to 1. The statistics interrupt is removed by writing to decrement any statistics value greater than 8000 0000h. As long as the most-significant bit of any statistics value is set, the interrupt remains asserted.

The application software must akcnowledge the EMAC control module after receiving statistics interrupts by writing the appropriate C0MISC key to the EMAC End-Of-Interrupt Vector register (MACEOIVECTOR). See Section 32.5.12 for the acknowledge key values.

32.2.17.1.4 Host Error Interrupt

The host error interrupt (HOSTPEND) is issued, if enabled, under error conditions dealing with the handling of buffer descriptors, detected during transmit or receive DMA transactions. The failure of the software application to supply properly formatted buffer descriptors results in this error. The error bit can only be cleared by resetting the EMAC module in hardware.

The host error interrupt is enabled by setting the HOSTMASK bit in the MAC interrupt mask set register (MACINTMASKSET) to 1. The host error interrupt is disabled by clearing the appropriate bit by writing a 1 in the MAC interrupt mask clear register (MACINTMASKCLEAR). The raw and masked host error interrupt status may be read by reading the MAC interrupt status (unmasked) register (MACINTSTATRAW) and the MAC interrupt status (masked) register (MACINTSTATMASKED), respectively.

The transmit host error conditions are:

- SOP error
- Ownership bit not set in SOP buffer
- Zero next buffer descriptor pointer with EOP
- Zero buffer pointer
- Zero buffer length
- Packet length error

The receive host error conditions are:

- Ownership bit not set in input buffer
- Zero buffer pointer

The application software must acknowledge the EMAC control module after receiving host error interrupts by writing the appropriate C0MISC key to the EMAC End-Of-Interrupt Vector (MACEOIVECTOR). See Section 32.5.12 for the acknowledge key values.

32.2.17.1.5 Receive Threshold Interrupts

Each of the eight receive channels have a corresponding receive threshold interrupt (RX*n*THRESHPEND). The receive threshold interrupts are level interrupts that remain asserted until the triggering condition is cleared by the host. Each of the eight threshold interrupts may be individually enabled by setting to 1 the appropriate bit in the RXINTMASKSET register. Each of the eight channel interrupts may be individually disabled by clearing to 0 the appropriate bit by writing a 1 in the receive interrupt mask clear register (RXINTMASKCLEAR). The raw and masked interrupt receive interrupt status may be read by reading the receive interrupt status (unmasked) register (RXINTSTATRAW) and the receive interrupt status (masked) register (RXINTSTATMASKED),respectively.

An RX*n*THRESHPEND interrupt bit is asserted when enabled and when the channel's associated free buffer count (RX*n*FREEBUFFER) is less than or equal to the channel's associated flow control threshold register (RX*n*FLOWTHRESH). The receive threshold interrupts use the same free buffer count and threshold logic as does flow control, but the interrupts are independently enabled from flow control. The threshold interrupts are intended to give the host an indication that resources are running low for a particular channel(s).

The applications software must acknowledge the EMAC control module after receiving threshold interrupts by writing the appropriate C0RXTHRESH key to the EMAC End-Of-Interrupt Vector (MACEOIVECTOR). See Section 32.5.12 for the acknowledge key values.

32.2.17.2 MDIO Module Interrupt Events and Requests

The MDIO module generates two interrupt events:

- LINKINT0: Serial interface link change interrupt. Indicates a change in the state of the PHY link selected by the USERPHYSEL0 register
- USERINT0: Serial interface user command event complete interrupt selected by the USERACCESS0 register

32.2.17.2.1 Link Change Interrupt

The MDIO module asserts a link change interrupt (LINKINT0) if there is a change in the link state of the PHY corresponding to the address in the PHYADRMON bit in the MDIO register USERPHYSEL0, and if the LINKINTENB bit is also set in USERPHYSEL0. This interrupt event is also captured in the LINKINTRAW bit in the MDIO link status change interrupt register (LINKINTRAW). LINKINTRAW bits 0 and 1 correspond to USERPHYSEL0 and USERPHYSEL1, respectively.

When the interrupt is enabled and generated, the corresponding LINKINTMASKED bit is also set in the MDIO link status change interrupt register (LINKINTMASKED). The interrupt is cleared by writing back the same bit to LINKINTMASKED (write to clear).

The application software must acknowledge the EMAC control module after receiving MDIO interrupts by writing the appropriate C0MISC key to the EMAC End-Of-Interrupt Vector (MACEOIVECTOR). See Section 32.5.12 for the acknowledge key values.

32.2.17.2.2 User Access Completion Interrupt

When the GO bit in one of the MDIO register USERACCESS0 transitions from 1 to 0 (indicating completion of a user access) and the corresponding USERINTMASKSET bit in the MDIO user command complete interrupt mask set register (USERINTMASKSET) corresponding to USERACCESS0 is set, a user access completion interrupt (USERINT) is asserted. This interrupt event is also captured in the USERINTRAW bit in the MDIO user command complete interrupt register (USERINTRAW). USERINTRAW bits 0 and bit 1 correspond to USERACCESS0 and USERACCESS1, respectively.

When the interrupt is enabled and generated, the corresponding USERINTMASKED bit is also set in the MDIO user command complete interrupt register (USERINTMASKED). The interrupt is cleared by writing back the same bit to USERINTMASKED (write to clear).

The application software must acknowledge the EMAC control module after receiving MDIO interrupts by writing the appropriate C0MISC key to the EMAC End-Of-Interrupt Vector (MACEOIVECTOR). See Section 32.5.12 for the acknowledge key values.

32.2.17.3 Proper Interrupt Processing

All the interrupts signaled from the EMAC and MDIO modules are level driven, so if they remain active, their level remains constant; the CPU core may require edge- or pulse-triggered interrupts. In order to properly convert the level-driven interrupt signal to an edge- or pulse-triggered signal, the application software must make use of the interrupt control logic contained in the EMAC control module.

Section 32.2.7.2 discusses the interrupt control contained in the EMAC control module. For safe interrupt processing, upon entry to the ISR, the software application should disable interrupts using the EMAC control module registers C0RXTHRESHEN, C0RXEN, C0TXEN, C0MISCEN, and then reenable them upon leaving the ISR. If any interrupt signals are active at that time, this creates another rising edge on the interrupt signal going to the CPU interrupt controller, thus triggering another interrupt. The EMAC control module also uses the EMAC control module registers INTCONTROL, C0TXIMAX, and C0RXIMAX to implement interrupt pacing. The application software must acknowledge the EMAC control module by writing the appropriate key to the EMAC End-Of-Interrupt Vector (MACEOIVECTOR). See Section 32.5.12 for the acknowledge key values.

32.2.17.4 Interrupt Multiplexing

The EMAC control module combines different interrupt signals from both the EMAC and MDIO modules into four interrupt signals (C0RXTHRESHPULSE, C0RXPULSE, C0TXPULSE, C0MISCPULSE) that are routed to the Vectored Interrupt Manager (VIM). The VIM is capable of relaying all four interrupt signals to the CPU.

When an interrupt is generated, the reason for the interrupt can be read from the MAC input vector register (MACINVECTOR) located in the EMAC memory map. MACINVECTOR combines the status of the following 28 interrupt signals: TXPEND*n*, RXPEND*n*, RXTHRESHPEND*n*, STATPEND, HOSTPEND, LINKINT0, and USERINT0.

For more details on the interrupt mapping, see your device-specific datasheet and VIM chapter of the Technical Reference Manual.

32.2.18 Power Management

Each of the three main components of the EMAC peripheral can be placed in a reduced-power mode to conserve power during periods of low activity. The peripheral clock to the EMAC peripheral is controlled by the processor Global Clock Module (GCM). The GCM allows the application to enable or disable the peripheral clock to the EMAC peripheral.

The power conservation modes available for each of the three components of the EMAC/MDIO peripheral are:

- *Idle/Disabled state*. This mode stops the clocks going to the peripheral, and prevents all the register accesses. After reenabling the peripheral from this idle state, all the registers values prior to setting into the disabled state are restored, and data transmission can proceed. No reinitialization is required.
- *System reset.* The EMAC peripheral is reset by the system reset signal output from the System module. Refer to the "Architecture" chapter of the Technical Reference Manual to identify the causes of a system reset. Upon a system reset, the registers are reset to their default value. When powering-up after a system reset, all the EMAC submodules need to be reinitialized before any data transmission can happen.

For more information on the use of the GCM, see your device-specific Technical Reference Manual.

32.2.19 Emulation Considerations

EMAC emulation control is implemented for compatibility with other peripherals. The SOFT and FREE bits in the emulation control register (EMCONTROL) allow EMAC operation to be suspended.

When the emulation suspend state is entered, the EMAC stops processing receive and transmit frames at the next frame boundary. Any frame currently in reception or transmission is completed normally without suspension. For transmission, any complete or partial frame in the transmit cell FIFO is transmitted. For receive, frames that are detected by the EMAC after the suspend state is entered are ignored. No statistics are kept for ignored frames.

Table 32-9 shows how the SOFT and FREE bits affect the operation of the emulation suspend.

NOTE: Emulation suspend has not been tested.

Table 32-9. Emulation Control

SOFT	FREE	Description
0	0	Normal operation
1	0	Emulation suspend
X	1	Normal operation

32.3 EMAC Control Module Registers

Table 32-10 lists the memory-mapped registers for the EMAC control module. The base address for these registers is FCF7 8800h.

Table 32-10. EMAC Control Module Registers

Offset	Acronym	Register Description	Section
0h	REVID	EMAC Control Module Revision ID Register	Section 32.3.1
4h	SOFTRESET	EMAC Control Module Software Reset Register	Section 32.3.2
Ch	INTCONTROL	EMAC Control Module Interrupt Control Register	Section 32.3.3
10h	C0RXTHRESHEN	EMAC Control Module Receive Threshold Interrupt Enable Register	Section 32.3.4
14h	C0RXEN	EMAC Control Module Receive Interrupt Enable Register	Section 32.3.5
18h	C0TXEN	EMAC Control Module Transmit Interrupt Enable Register	Section 32.3.6
1Ch	C0MISCEN	EMAC Control Module Miscellaneous Interrupt Enable Register	Section 32.3.7
40h	C0RXTHRESHSTAT	EMAC Control Module Receive Threshold Interrupt Status Register	Section 32.3.8
44h	C0RXSTAT	EMAC Control Module Receive Interrupt Status Register	Section 32.3.9
48h	C0TXSTAT	EMAC Control Module Transmit Interrupt Status Register	Section 32.3.10
4Ch	C0MISCSTAT	EMAC Control Module Miscellaneous Interrupt Status Register	Section 32.3.11
70h	C0RXIMAX	EMAC Control Module Receive Interrupts Per Millisecond Register	Section 32.3.12
74h	C0TXIMAX	EMAC Control Module Transmit Interrupts Per Millisecond Register	Section 32.3.13

32.3.1 EMAC Control Module Revision ID Register (REVID)

The EMAC control module revision ID register (REVID) is shown in Figure 32-15 and described in Table 32-11.

Figure 32-15. EMAC Control Module Revision ID Register (REVID) (offset = 00h)

31 … 0
REV
R-4EC8 0100h

LEGEND: R = Read only; -*n* = value after reset

Table 32-11. EMAC Control Module Revision ID Register (REVID) Field Descriptions

Bit	Field	Value	Description
31-0	REV		Identifies the EMAC Control Module revision.
		4EC8 0100h	Current revision of the EMAC Control Module.

32.3.2 EMAC Control Module Software Reset Register (SOFTRESET)

The EMAC Control Module Software Reset Register (SOFTRESET) is shown in Figure 32-16 and described in Table 32-12.

Figure 32-16. EMAC Control Module Software Reset Register (SOFTRESET) (offset = 04h)

31 … 16
Reserved
R-0

15 … 1	0
Reserved	RESET
R-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 32-12. EMAC Control Module Software Reset Register (SOFTRESET)

Bit	Field	Value	Description
31-1	Reserved	0	Reserved
0	RESET		Software reset bit for the EMAC Control Module. Clears the interrupt status, control registers, and CPPI Ram on the clock cycle following a write of 1.
		0	No software reset.
		1	Perform a software reset.

32.3.3 EMAC Control Module Interrupt Control Register (INTCONTROL)

The EMAC control module interrupt control register (INTCONTROL) is shown in Figure 32-17 and described in Table 32-13. The settings in the INTCONTROL register are used in conjunction with the CnRXIMAX and CnTXIMAX registers.

Figure 32-17. EMAC Control Module Interrupt Control Register (INTCONTROL) (offset = 0Ch)

31-24
Reserved
R-0

23-18	17	16
Reserved	C0TXPACEEN	C0RXPACEEN
R-0	R/W-0	R/W-0

15-12	11-0
Reserved	INTPRESCALE
R-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 32-13. EMAC Control Module Interrupt Control Register (INTCONTROL)

Bit	Field	Value	Description
31-18	Reserved	0	Reserved
17	C0TXPACEEN		Enable pacing for TX interrupt pulse generation.
		0	Pacing for TX interrupts is disabled.
		1	Pacing for TX interrupts is enabled.
16	C0RXPACEEN		Enable pacing for RX interrupt pulse generation.
		0	Pacing for RX interrupts is disabled.
		1	Pacing for RX interrupts is enabled.
15-12	Reserved	0	Reserved
11-0	INTPRESCALE	0-7FFh	Number of internal EMAC module reference clock periods within a 4 μs time window (see your device-specific data manual for information).

32.3.4 EMAC Control Module Receive Threshold Interrupt Enable Registers (C0RXTHRESHEN)

The EMAC control module receive threshold interrupt enable register (C0RXTHRESHEN) is shown in Figure 32-18 and described in Table 32-14.

Figure 32-18. EMAC Control Module Receive Threshold Interrupt Enable Register (C0RXTHRESHEN) (offset = 10h)

31	16
Reserved	
R-0	

15	8
Reserved	
R-0	

7	6	5	4	3	2	1	0
RXCH7 THRESHEN	RXCH6 THRESHEN	RXCH5 THRESHEN	RXCH4 THRESHEN	RXCH3 THRESHEN	RXCH2 THRESHEN	RXCH1 THRESHEN	RXCH0 THRESHEN
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 32-14. EMAC Control Module Receive Threshold Interrupt Enable Register (C0RXTHRESHEN)

Bit	Field	Value	Description
31-8	Reserved	0	Reserved
7	RXCH7THRESHEN		Enable C0RXTHRESHPULSE interrupt generation for RX Channel 7.
		0	C0RXTHRESHPULSE generation is disabled for RX Channel 7.
		1	C0RXTHRESHPULSE generation is enabled for RX Channel 7.
6	RXCH6THRESHEN		Enable C0RXTHRESHPULSE interrupt generation for RX Channel 6.
		0	C0RXTHRESHPULSE generation is disabled for RX Channel 6.
		1	C0RXTHRESHPULSE generation is enabled for RX Channel 6.
5	RXCH5THRESHEN		Enable C0RXTHRESHPULSE interrupt generation for RX Channel 5.
		0	C0RXTHRESHPULSE generation is disabled for RX Channel 5.
		1	C0RXTHRESHPULSE generation is enabled for RX Channel 5.
4	RXCH4THRESHEN		Enable C0RXTHRESHPULSE interrupt generation for RX Channel 4.
		0	C0RXTHRESHPULSE generation is disabled for RX Channel 4.
		1	C0RXTHRESHPULSE generation is enabled for RX Channel 4.
3	RXCH3THRESHEN		Enable C0RXTHRESHPULSE interrupt generation for RX Channel 3.
		0	C0RXTHRESHPULSE generation is disabled for RX Channel 3.
		1	C0RXTHRESHPULSE generation is enabled for RX Channel 3.
2	RXCH2THRESHEN		Enable C0RXTHRESHPULSE interrupt generation for RX Channel 2.
		0	C0RXTHRESHPULSE generation is disabled for RX Channel 2.
		1	C0RXTHRESHPULSE generation is enabled for RX Channel 2.
1	RXCH1THRESHEN		Enable C0RXTHRESHPULSE interrupt generation for RX Channel 1.
		0	C0RXTHRESHPULSE generation is disabled for RX Channel 1.
		1	C0RXTHRESHPULSE generation is enabled for RX Channel 1.
0	RXCH0THRESHEN		Enable C0RXTHRESHPULSE interrupt generation for RX Channel 0.
		0	C0RXTHRESHPULSE generation is disabled for RX Channel 0.
		1	C0RXTHRESHPULSE generation is enabled for RX Channel 0.

32.3.5 EMAC Control Module Receive Interrupt Enable Registers (C0RXEN)

The EMAC control module receive interrupt enable register (C0RXEN) is shown in Figure 32-19 and described in Table 32-15

Figure 32-19. EMAC Control Module Receive Interrupt Enable Register (C0RXEN) (offset = 14h)

31	16
Reserved	
R-0	

15	8
Reserved	
R-0	

7	6	5	4	3	2	1	0
RXCH7EN	RXCH6EN	RXCH5EN	RXCH4EN	RXCH3EN	RXCH2EN	RXCH1EN	RXCH0EN
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 32-15. EMAC Control Module Receive Interrupt Enable Register (C0RXEN)

Bit	Field	Value	Description
31-8	Reserved	0	Reserved
7	RXCH7EN		Enable C0RXPULSE interrupt generation for RX Channel 7.
		0	C0RXPULSE generation is disabled for RX Channel 7.
		1	C0RXPULSE generation is enabled for RX Channel 7.
6	RXCH6EN		Enable C0RXPULSE interrupt generation for RX Channel 6.
		0	C0RXPULSE generation is disabled for RX Channel 6.
		1	C0RXPULSE generation is enabled for RX Channel 6.
5	RXCH5EN		Enable C0RXPULSE interrupt generation for RX Channel 5.
		0	C0RXPULSE generation is disabled for RX Channel 5.
		1	C0RXPULSE generation is enabled for RX Channel 5.
4	RXCH4EN		Enable C0RXPULSE interrupt generation for RX Channel 4.
		0	C0RXPULSE generation is disabled for RX Channel 4.
		1	C0RXPULSE generation is enabled for RX Channel 4.
3	RXCH3EN		Enable C0RXPULSE interrupt generation for RX Channel 3.
		0	C0RXPULSE generation is disabled for RX Channel 3.
		1	C0RXPULSE generation is enabled for RX Channel 3.
2	RXCH2EN		Enable C0RXPULSE interrupt generation for RX Channel 2.
		0	C0RXPULSE generation is disabled for RX Channel 2.
		1	C0RXPULSE generation is enabled for RX Channel 2.
1	RXCH1EN		Enable C0RXPULSE interrupt generation for RX Channel 1.
		0	C0RXPULSE generation is disabled for RX Channel 1.
		1	C0RXPULSE generation is enabled for RX Channel 1.
0	RXCH0EN		Enable C0RXPULSE interrupt generation for RX Channel 0.
		0	C0RXPULSE generation is disabled for RX Channel 0.
		1	C0RXPULSE generation is enabled for RX Channel 0.

32.3.6 EMAC Control Module Transmit Interrupt Enable Registers (C0TXEN)

The EMAC control module transmit interrupt enable register (C0TXEN) is shown in Figure 32-20 and described in Table 32-16

Figure 32-20. EMAC Control Module Transmit Interrupt Enable Register (C0TXEN) (offset = 18h)

31	16
Reserved	
R-0	

15	8
Reserved	
R-0	

7	6	5	4	3	2	1	0
TXCH7EN	TXCH6EN	TXCH5EN	TXCH4EN	TXCH3EN	TXCH2EN	TXCH1EN	TXCH0EN
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 32-16. EMAC Control Module Transmit Interrupt Enable Register (C0TXEN)

Bit	Field	Value	Description
31-8	Reserved	0	Reserved
7	TXCH7EN		Enable C0TXPULSE interrupt generation for TX Channel 7.
		0	C0TXPULSE generation is disabled for TX Channel 7.
		1	C0TXPULSE generation is enabled for TX Channel 7.
6	TXCH6EN		Enable C0TXPULSE interrupt generation for TX Channel 6.
		0	C0TXPULSE generation is disabled for TX Channel 6.
		1	C0TXPULSE generation is enabled for TX Channel 6.
5	TXCH5EN		Enable C0TXPULSE interrupt generation for TX Channel 5.
		0	C0TXPULSE generation is disabled for TX Channel 5.
		1	C0TXPULSE generation is enabled for TX Channel 5.
4	TXCH4EN		Enable C0TXPULSE interrupt generation for TX Channel 4.
		0	C0TXPULSE generation is disabled for TX Channel 4.
		1	C0TXPULSE generation is enabled for TX Channel 4.
3	TXCH3EN		Enable C0TXPULSE interrupt generation for TX Channel 3.
		0	C0TXPULSE generation is disabled for TX Channel 3.
		1	C0TXPULSE generation is enabled for TX Channel 3.
2	TXCH2EN		Enable C0TXPULSE interrupt generation for TX Channel 2.
		0	C0TXPULSE generation is disabled for TX Channel 2.
		1	C0TXPULSE generation is enabled for TX Channel 2.
1	TXCH1EN		Enable C0TXPULSE interrupt generation for TX Channel 1.
		0	C0TXPULSE generation is disabled for TX Channel 1.
		1	C0TXPULSE generation is enabled for TX Channel 1.
0	TXCH0EN		Enable C0TXPULSE interrupt generation for TX Channel 0.
		0	C0TXPULSE generation is disabled for TX Channel 0.
		1	C0TXPULSE generation is enabled for TX Channel 0.

32.3.7 EMAC Control Module Miscellaneous Interrupt Enable Registers (C0MISCEN)

The EMAC control module miscellaneous interrupt enable register (C0MISCEN) is shown in Figure 32-21 and described in Table 32-17.

Figure 32-21. EMAC Control Module Miscellaneous Interrupt Enable Register (C0MISCEN) (offset = 1Ch)

31–16
Reserved
R-0

15–4	3	2	1	0
Reserved	STATPENDEN	HOSTPENDEN	LINKINT0EN	USERINT0EN
R-0	R/W-0	R/W-0	R/W-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 32-17. EMAC Control Module Miscellaneous Interrupt Enable Register (C0MISCEN)

Bit	Field	Value	Description
31-4	Reserved	0	Reserved
3	STATPENDEN		Enable C0MISCPULSE interrupt generation when EMAC statistics interrupts are generated.
		0	C0MISCPULSE generation is disabled for EMAC STATPEND interrupts.
		1	C0MISCPULSE generation is enabled for EMAC STATPEND interrupts.
2	HOSTPENDEN		Enable C0MISCPULSE interrupt generation when EMAC host interrupts are generated.
		0	C0MISCPULSE generation is disabled for EMAC HOSTPEND interrupts.
		1	C0MISCPULSE generation is enabled for EMAC HOSTPEND interrupts.
1	LINKINT0EN		Enable C0MISCPULSE interrupt generation when MDIO LINKINT0 interrupts (corresponding to USERPHYSEL0) are generated.
		0	C0MISCPULSE generation is disabled for MDIO LINKINT0 interrupts.
		1	C0MISCPULSE generation is enabled for MDIO LINKINT0 interrupts.
0	USERINT0EN		Enable C0MISCPULSE interrupt generation when MDIO USERINT0 interrupts (corresponding to USERACCESS0) are generated.
		0	C0MISCPULSE generation is disabled for MDIO USERINT0.
		1	C0MISCPULSE generation is enabled for MDIO USERINT0.

32.3.8 EMAC Control Module Receive Threshold Interrupt Status Registers (C0RXTHRESHSTAT)

The EMAC control module receive threshold interrupt status register (C0RXTHRESHSTAT) is shown in Figure 32-22 and described in Table 32-18

Figure 32-22. EMAC Control Module Receive Threshold Interrupt Status Register (C0RXTHRESHSTAT) (offset = 40h)

31 ... 16
Reserved
R-0

15 ... 8
Reserved
R-0

7	6	5	4	3	2	1	0
RXCH7THRESH STAT	RXCH6THRESH STAT	RXCH5THRESH STAT	RXCH4THRESH STAT	RXCH3THRESH STAT	RXCH2THRESH STAT	RXCH1THRESH STAT	RXCH0THRESH STAT
R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0

LEGEND: R = Read only; -*n* = value after reset

Table 32-18. EMAC Control Module Receive Threshold Interrupt Status Register (C0RXTHRESHSTAT)

Bit	Field	Value	Description
31-8	Reserved	0	Reserved
7	RXCH7THRESHSTAT		Interrupt status for RX Channel 7 masked by the C0RXTHRESHEN register.
		0	RX Channel 7 does not satisfy conditions to generate a C0RXTHRESHPULSE interrupt.
		1	RX Channel 7 satisfies conditions to generate a C0RXTHRESHPULSE interrupt.
6	RXCH6THRESHSTAT		Interrupt status for RX Channel 6 masked by the C0RXTHRESHEN register.
		0	RX Channel 6 does not satisfy conditions to generate a C0RXTHRESHPULSE interrupt.
		1	RX Channel 6 satisfies conditions to generate a C0RXTHRESHPULSE interrupt.
5	RXCH5THRESHSTAT		Interrupt status for RX Channel 5 masked by the C0RXTHRESHEN register.
		0	RX Channel 5 does not satisfy conditions to generate a C0RXTHRESHPULSE interrupt.
		1	RX Channel 5 satisfies conditions to generate a C0RXTHRESHPULSE interrupt.
4	RXCH4THRESHSTAT		Interrupt status for RX Channel 4 masked by the C0RXTHRESHEN register.
		0	RX Channel 4 does not satisfy conditions to generate a C0RXTHRESHPULSE interrupt.
		1	RX Channel 4 satisfies conditions to generate a C0RXTHRESHPULSE interrupt.
3	RXCH3THRESHSTAT		Interrupt status for RX Channel 3 masked by the C0RXTHRESHEN register.
		0	RX Channel 3 does not satisfy conditions to generate a C0RXTHRESHPULSE interrupt.
		1	RX Channel 3 satisfies conditions to generate a C0RXTHRESHPULSE interrupt.
2	RXCH2THRESHSTAT		Interrupt status for RX Channel 2 masked by the C0RXTHRESHEN register.
		0	RX Channel 2 does not satisfy conditions to generate a C0RXTHRESHPULSE interrupt.
		1	RX Channel 2 satisfies conditions to generate a C0RXTHRESHPULSE interrupt.
1	RXCH1THRESHSTAT		Interrupt status for RX Channel 1 masked by the C0RXTHRESHEN register.
		0	RX Channel 1 does not satisfy conditions to generate a C0RXTHRESHPULSE interrupt.
		1	RX Channel 1 satisfies conditions to generate a C0RXTHRESHPULSE interrupt.
0	RXCH0THRESHSTAT		Interrupt status for RX Channel 0 masked by the C0RXTHRESHEN register.
		0	RX Channel 0 does not satisfy conditions to generate a C0RXTHRESHPULSE interrupt.
		1	RX Channel 0 satisfies conditions to generate a C0RXTHRESHPULSE interrupt.

32.3.9 EMAC Control Module Receive Interrupt Status Registers (C0RXSTAT)

The EMAC control module receive interrupt status register (C0RXSTAT) is shown in Figure 32-23 and described in Table 32-19

Figure 32-23. EMAC Control Module Receive Interrupt Status Register (C0RXSTAT) (offset = 44h)

31	16
Reserved	
R-0	

15	8
Reserved	
R-0	

7	6	5	4	3	2	1	0
RXCH7STAT	RXCH6STAT	RXCH5STAT	RXCH4STAT	RXCH3STAT	RXCH2STAT	RXCH1STAT	RXCH0STAT
R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0

LEGEND: R = Read only; -*n* = value after reset

Table 32-19. EMAC Control Module Receive Interrupt Status Register (C0RXSTAT)

Bit	Field	Value	Description
31-8	Reserved	0	Reserved
7	RXCH7STAT		Interrupt status for RX Channel 7 masked by the C0RXEN register.
		0	RX Channel 7 does not satisfy conditions to generate a C0RXPULSE interrupt.
		1	RX Channel 7 satisfies conditions to generate a C0RXPULSE interrupt.
6	RXCH6STAT		Interrupt status for RX Channel 6 masked by the C0RXEN register.
		0	RX Channel 6 does not satisfy conditions to generate a C0RXPULSE interrupt.
		1	RX Channel 6 satisfies conditions to generate a C0RXPULSE interrupt.
5	RXCH5STAT		Interrupt status for RX Channel 5 masked by the C0RXEN register.
		0	RX Channel 5 does not satisfy conditions to generate a C0RXPULSE interrupt.
		1	RX Channel 5 satisfies conditions to generate a C0RXPULSE interrupt.
4	RXCH4STAT		Interrupt status for RX Channel 4 masked by the C0RXEN register.
		0	RX Channel 4 does not satisfy conditions to generate a C0RXPULSE interrupt.
		1	RX Channel 4 satisfies conditions to generate a C0RXPULSE interrupt.
3	RXCH3STAT		Interrupt status for RX Channel 3 masked by the C0RXEN register.
		0	RX Channel 3 does not satisfy conditions to generate a C0RXPULSE interrupt.
		1	RX Channel 3 satisfies conditions to generate a C0RXPULSE interrupt.
2	RXCH2STAT		Interrupt status for RX Channel 2 masked by the C0RXEN register.
		0	RX Channel 2 does not satisfy conditions to generate a C0RXPULSE interrupt.
		1	RX Channel 2 satisfies conditions to generate a C0RXPULSE interrupt.
1	RXCH1STAT		Interrupt status for RX Channel 1 masked by the C0RXEN register.
		0	RX Channel 1 does not satisfy conditions to generate a C0RXPULSE interrupt.
		1	RX Channel 1 satisfies conditions to generate a C0RXPULSE interrupt.
0	RXCH0STAT		Interrupt status for RX Channel 0 masked by the C0RXEN register.
		0	RX Channel 0 does not satisfy conditions to generate a C0RXPULSE interrupt.
		1	RX Channel 0 satisfies conditions to generate a C0RXPULSE interrupt.

32.3.10 EMAC Control Module Transmit Interrupt Status Registers (C0TXSTAT)

The EMAC control module transmit interrupt status register (C0TXSTAT) is shown in Figure 32-24 and described in Table 32-20

Figure 32-24. EMAC Control Module Transmit Interrupt Status Register (C0TXSTAT) (offset = 48h)

31-16
Reserved
R-0

15-8
Reserved
R-0

7	6	5	4	3	2	1	0
TXCH7STAT	TXCH6STAT	TXCH5STAT	TXCH4STAT	TXCH3STAT	TXCH2STAT	TXCH1STAT	TXCH0STAT
R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0

LEGEND: R = Read only; -*n* = value after reset

Table 32-20. EMAC Control Module Transmit Interrupt Status Register (C0TXSTAT)

Bit	Field	Value	Description
31-8	Reserved	0	Reserved
7	TXCH7STAT		Interrupt status for TX Channel 7 masked by the C0TXEN register.
		0	TX Channel 7 does not satisfy conditions to generate a C0TXPULSE interrupt.
		1	TX Channel 7 satisfies conditions to generate a C0TXPULSE interrupt.
6	TXCH6STAT		Interrupt status for TX Channel 6 masked by the C0TXEN register.
		0	TX Channel 6 does not satisfy conditions to generate a C0TXPULSE interrupt.
		1	TX Channel 6 satisfies conditions to generate a C0TXPULSE interrupt.
5	TXCH5STAT		Interrupt status for TX Channel 5 masked by the C0TXEN register.
		0	TX Channel 5 does not satisfy conditions to generate a C0TXPULSE interrupt.
		1	TX Channel 5 satisfies conditions to generate a C0TXPULSE interrupt.
4	TXCH4STAT		Interrupt status for TX Channel 4 masked by the C0TXEN register.
		0	TX Channel 4 does not satisfy conditions to generate a C0TXPULSE interrupt.
		1	TX Channel 4 satisfies conditions to generate a C0TXPULSE interrupt.
3	TXCH3STAT		Interrupt status for TX Channel 3 masked by the C0TXEN register.
		0	TX Channel 3 does not satisfy conditions to generate a C0TXPULSE interrupt.
		1	TX Channel 3 satisfies conditions to generate a C0TXPULSE interrupt.
2	TXCH2STAT		Interrupt status for TX Channel 2 masked by the C0TXEN register.
		0	TX Channel 2 does not satisfy conditions to generate a C0TXPULSE interrupt.
		1	TX Channel 2 satisfies conditions to generate a C0TXPULSE interrupt.
1	TXCH1STAT		Interrupt status for TX Channel 1 masked by the C0TXEN register.
		0	TX Channel 1 does not satisfy conditions to generate a C0TXPULSE interrupt.
		1	TX Channel 1 satisfies conditions to generate a C0TXPULSE interrupt.
0	TXCH0STAT		Interrupt status for TX Channel 0 masked by the C0TXEN register.
		0	TX Channel 0 does not satisfy conditions to generate a C0TXPULSE interrupt.
		1	TX Channel 0 satisfies conditions to generate a C0TXPULSE interrupt.

32.3.11 EMAC Control Module Miscellaneous Interrupt Status Registers (C0MISCSTAT)

The EMAC control module miscellaneous interrupt status register (C0MISCSTAT) is shown in Figure 32-25 and described in Table 32-21

Figure 32-25. EMAC Control Module Miscellaneous Interrupt Status Register (C0MISCSTAT) (offset = 4Ch)

31 – 16
Reserved
R-0

15 – 4	3	2	1	0
Reserved	STATPEND STAT	HOSTPEND STAT	LINKINT0 STAT	USERINT0 STAT
R-0	R-0	R-0	R-0	R-0

LEGEND: R = Read only; -*n* = value after reset

Table 32-21. EMAC Control Module Miscellaneous Interrupt Status Register (C0MISCSTAT)

Bit	Field	Value	Description
31-4	Reserved	0	Reserved
3	STATPENDSTAT		Interrupt status for EMAC STATPEND masked by the C0MISCEN register.
		0	EMAC STATPEND does not satisfy conditions to generate a C0MISCPULSE interrupt.
		1	EMAC STATPEND satisfies conditions to generate a C0MISCPULSE interrupt.
2	HOSTPENDSTAT		Interrupt status for EMAC HOSTPEND masked by the C0MISCEN register.
		0	EMAC HOSTPEND does not satisfy conditions to generate a C0MISCPULSE interrupt.
		1	EMAC HOSTPEND satisfies conditions to generate a C0MISCPULSE interrupt.
1	LINKINT0STAT		Interrupt status for MDIO LINKINT0 masked by the C0MISCEN register.
		0	MDIO LINKINT0 does not satisfy conditions to generate a C0MISCPULSE interrupt.
		1	MDIO LINKINT0 satisfies conditions to generate a C0MISCPULSE interrupt.
0	USERINT0STAT		Interrupt status for MDIO USERINT0 masked by the C0MISCEN register.
		0	MDIO USERINT0 does not satisfy conditions to generate a C0MISCPULSE interrupt.
		1	MDIO USERINT0 satisfies conditions to generate a C0MISCPULSE interrupt.

32.3.12 EMAC Control Module Receive Interrupts Per Millisecond Registers (C0RXIMAX)

The EMAC control module receive interrupts per millisecond register (C0RXIMAX) is shown in Figure 32-26 and described in Table 32-22.

Figure 32-26. EMAC Control Module Receive Interrupts Per Millisecond Register (C0RXIMAX) (offset = 70h)

31 … 16
Reserved
R-0

15 … 6	5 … 0
Reserved	RXIMAX
R-0	R/W-0

LEGEND: R = Read only; R/W = Read/Write; -*n* = value after reset

Table 32-22. EMAC Control Module Receive Interrupts Per Millisecond Register (C0RXIMAX)

Bit	Field	Value	Description
31-6	Reserved	0	Reserved
5-0	RXIMAX	2-3Fh	RXIMAX is the desired number of C0RXPULSE interrupts generated per millisecond when C0RXPACEEN is enabled in INTCONTROL.

The pacing mechanism can be described by the following pseudo-code:

```
while(1) {
    interrupt_count = 0;

    /* Count interrupts over a 1ms window */
    for(i = 0; i < INTCONTROL[INTPRESCALE]*250; i++) {
        interrupt_count += NEW_INTERRUPT_EVENTS();

        if(i < INTCONTROL[INTPRESCALE]*pace_counter)
            BLOCK_EMAC_INTERRUPTS();
        else
            ALLOW_EMAC_INTERRUPTS();
    }

    ALLOW_EMAC_INTERRUPTS();

    if(interrupt_count > 2*RXIMAX)
        pace_counter = 255;
    else if(interrupt_count > 1.5*RXIMAX)
        pace_counter = previous_pace_counter*2 + 1;
    else if(interrupt_count > 1.0*RXIMAX)
        pace_counter = previous_pace_counter + 1;
    else if(interrupt_count > 0.5*RXIMAX)
        pace_counter = previous_pace_counter - 1;
    else if(interrupt_count != 0)
        pace_counter = previous_pace_counter/2;
    else
        pace_counter = 0;

    previous_pace_counter = pace_counter;
}
```

32.3.13 EMAC Control Module Transmit Interrupts Per Millisecond Registers (C0TXIMAX)

The EMAC control module transmit interrupts per millisecond register (C0TXIMAX) is shown in Figure 32-27 and described in Table 32-23

Figure 32-27. EMAC Control Module Transmit Interrupts Per Millisecond Register (C0TXIMAX) (offset = 74h)

31–16
Reserved
R-0

15–6	5–0
Reserved	TXIMAX
R-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 32-23. EMAC Control Module Transmit Interrupts Per Millisecond Register (C0TXIMAX)

Bit	Field	Value	Description
31-6	Reserved	0	Reserved
5-0	TXIMAX	2-3Fh	TXIMAX is the desired number of C0TXPULSE interrupts generated per millisecond when C0TXPACEEN is enabled in INTCONTROL.

The pacing mechanism can be described by the following pseudo-code:

```
while(1) {
    interrupt_count = 0;

    /* Count interrupts over a 1ms window */
    for(i = 0; i < INTCONTROL[INTPRESCALE]*250; i++) {
        interrupt_count += NEW_INTERRUPT_EVENTS();

        if(i < INTCONTROL[INTPRESCALE]*pace_counter)
            BLOCK_EMAC_INTERRUPTS();
        else
            ALLOW_EMAC_INTERRUPTS();
    }
    ALLOW_EMAC_INTERRUPTS();

    if(interrupt_count > 2*TXIMAX)
        pace_counter = 255;
    else if(interrupt_count > 1.5*TXIMAX)
        pace_counter = previous_pace_counter*2 + 1;
    else if(interrupt_count > 1.0*TXIMAX)
        pace_counter = previous_pace_counter + 1;
    else if(interrupt_count > 0.5*TXIMAX)
        pace_counter = previous_pace_counter - 1;
    else if(interrupt_count != 0)
        pace_counter = previous_pace_counter/2;
    else
        pace_counter = 0;

    previous_pace_counter = pace_counter;
}
```

32.4 MDIO Registers

Table 32-24 lists the memory-mapped registers for the MDIO module. The base address for these registers is FCF7 8900h.

Table 32-24. Management Data Input/Output (MDIO) Registers

Offset	Acronym	Register Description	Section
0h	REVID	MDIO Revision ID Register	Section 32.4.1
4h	CONTROL	MDIO Control Register	Section 32.4.2
8h	ALIVE	PHY Alive Status register	Section 32.4.3
Ch	LINK	PHY Link Status Register	Section 32.4.4
10h	LINKINTRAW	MDIO Link Status Change Interrupt (Unmasked) Register	Section 32.4.5
14h	LINKINTMASKED	MDIO Link Status Change Interrupt (Masked) Register	Section 32.4.6
20h	USERINTRAW	MDIO User Command Complete Interrupt (Unmasked) Register	Section 32.4.7
24h	USERINTMASKED	MDIO User Command Complete Interrupt (Masked) Register	Section 32.4.8
28h	USERINTMASKSET	MDIO User Command Complete Interrupt Mask Set Register	Section 32.4.9
2Ch	USERINTMASKCLEAR	MDIO User Command Complete Interrupt Mask Clear Register	Section 32.4.10
80h	USERACCESS0	MDIO User Access Register 0	Section 32.4.11
84h	USERPHYSEL0	MDIO User PHY Select Register 0	Section 32.4.12
88h	USERACCESS1	MDIO User Access Register 1	Section 32.4.13
8Ch	USERPHYSEL1	MDIO User PHY Select Register 1	Section 32.4.14

32.4.1 MDIO Revision ID Register (REVID)

The MDIO revision ID register (REVID) is shown in Figure 32-28 and described in Table 32-25.

Figure 32-28. MDIO Revision ID Register (REVID) (offset = 00h)

31 … 0
REV
R-0007 0105h

LEGEND: R = Read only; -*n* = value after reset

Table 32-25. MDIO Revision ID Register (REVID) Field Descriptions

Bit	Field	Value	Description
31-0	REV		Identifies the MDIO Module revision.
		0007 0105h	Current revision of the MDIO Module.

32.4.2 MDIO Control Register (CONTROL)

The MDIO control register (CONTROL) is shown in Figure 32-29 and described in Table 32-26.

Figure 32-29. MDIO Control Register (CONTROL) (offset = 04h)

31	30	29	28 24	23 21	20	19	18	17 16
IDLE	ENABLE	Rsvd	HIGHEST_USER_CHANNEL	Reserved	PREAMBLE	FAULT	FAULTENB	Reserved
R-1	R/W-0	R-0	R-1	R-0	R/W-0	R/W1C-0	R/W-0	R-0

15 0
CLKDIV
R/W-FFh

LEGEND: R/W = Read/Write; R = Read only; W1C = Write 1 to clear (writing a 0 has no effect); -*n* = value after reset

Table 32-26. MDIO Control Register (CONTROL) Field Descriptions

Bit	Field	Value	Description
31	IDLE		State machine IDLE status bit.
		0	State machine is not in idle state.
		1	State machine is in idle state.
30	ENABLE		State machine enable control bit. If the MDIO state machine is active at the time it is disabled, it will complete the current operation before halting and setting the idle bit.
		0	Disables the MDIO state machine.
		1	Enable the MDIO state machine.
29	Reserved	0	Reserved
28-24	HIGHEST_USER_CHANNEL	0-1Fh	Highest user channel that is available in the module. It is currently set to 1. This implies that MDIOUserAccess1 is the highest available user access channel.
23-21	Reserved	0	Reserved
20	PREAMBLE		Preamble disable.
		0	Standard MDIO preamble is used.
		1	Disables this device from sending MDIO frame preambles.
19	FAULT		Fault indicator. This bit is set to 1 if the MDIO pins fail to read back what the device is driving onto them. This indicates a physical layer fault and the module state machine is reset. Writing a 1 to this bit clears this bit, writing a 0 has no effect.
		0	No failure.
		1	Physical layer fault; the MDIO state machine is reset.
18	FAULTENB		Fault detect enable. This bit has to be set to 1 to enable the physical layer fault detection.
		0	Disables the physical layer fault detection.
		1	Enables the physical layer fault detection.
17-16	Reserved	0	Reserved
15-0	CLKDIV	0-FFFFh	Clock Divider bits. This field specifies the division ratio between the peripheral clock and the frequency of MDIO_CLK. MDIO_CLK is disabled when CLKDIV is cleared to 0. MDIO_CLK frequency = peripheral clock frequency/(CLKDIV + 1).

32.4.3 PHY Acknowledge Status Register (ALIVE)

The PHY acknowledge status register (ALIVE) is shown in Figure 32-30 and described in Table 32-27.

Figure 32-30. PHY Acknowledge Status Register (ALIVE) (offset = 08h)

31 ... 0
ALIVE
R/W1C-0

LEGEND: R/W = Read/Write; W1C = Write 1 to clear (writing a 0 has no effect); -*n* = value after reset

Table 32-27. PHY Acknowledge Status Register (ALIVE) Field Descriptions

Bit	Field	Value	Description
31-0	ALIVE		MDIO Alive bits. Each of the 32 bits of this register is set if the most recent access to the PHY with address corresponding to the register bit number was acknowledged by the PHY; the bit is reset if the PHY fails to acknowledge the access. Both the user and polling accesses to a PHY will cause the corresponding alive bit to be updated. The alive bits are only meant to be used to give an indication of the presence or not of a PHY with the corresponding address. Writing a 1 to any bit will clear it, writing a 0 has no effect.
		0	The PHY fails to acknowledge the access.
		1	The most recent access to the PHY with an address corresponding to the register bit number was acknowledged by the PHY.

32.4.4 PHY Link Status Register (LINK)

The PHY link status register (LINK) is shown in Figure 32-31 and described in Table 32-28.

Figure 32-31. PHY Link Status Register (LINK) (offset = 0Ch)

31 ... 0
LINK
R-0

LEGEND: R = Read only; -*n* = value after reset

Table 32-28. PHY Link Status Register (LINK) Field Descriptions

Bit	Field	Value	Description
31-0	LINK		MDIO Link state bits. This register is updated after a read of the generic status register of a PHY. The bit is set if the PHY with the corresponding address has link and the PHY acknowledges the read transaction. The bit is reset if the PHY indicates it does not have link or fails to acknowledge the read transaction. Writes to the register have no effect.
		0	The PHY indicates it does not have a link or fails to acknowledge the read transaction.
		1	The PHY with the corresponding address has a link and the PHY acknowledges the read transaction.

32.4.5 MDIO Link Status Change Interrupt (Unmasked) Register (LINKINTRAW)

The MDIO link status change interrupt (unmasked) register (LINKINTRAW) is shown in Figure 32-32 and described in Table 32-29.

Figure 32-32. MDIO Link Status Change Interrupt (Unmasked) Register (LINKINTRAW) (offset = 10h)

31	16
Reserved	
R-0	

15	2	1	0
Reserved		USERPHY1	USERPHY0
R-0		R/W1C-0	R/W1C-0

LEGEND: R/W = Read/Write; R = Read only; W1C = Write 1 to clear (writing a 0 has no effect); -*n* = value after reset

Table 32-29. MDIO Link Status Change Interrupt (Unmasked) Register (LINKINTRAW) Field Descriptions

Bit	Field	Value	Description
31-2	Reserved	0	Reserved
1	USERPHY1		MDIO Link change event, raw value. When asserted, the bit indicates that there was an MDIO link change event (that is, change in the LINK register) corresponding to the PHY address in USERPHYSEL1. Writing a 1 will clear the event, writing a 0 has no effect.
		0	No MDIO link change event.
		1	An MDIO link change event (change in the LINK register) corresponding to the PHY address in MDIO user PHY select register USERPHYSEL1.
0	USERPHY0		MDIO Link change event, raw value. When asserted, the bit indicates that there was an MDIO link change event (that is, change in the LINK register) corresponding to the PHY address in USERPHYSEL0. Writing a 1 will clear the event, writing a 0 has no effect.
		0	No MDIO link change event.
		1	An MDIO link change event (change in the LINK register) corresponding to the PHY address in MDIO user PHY select register USERPHYSEL0.

32.4.6 MDIO Link Status Change Interrupt (Masked) Register (LINKINTMASKED)

The MDIO link status change interrupt (masked) register (LINKINTMASKED) is shown in Figure 32-33 and described in Table 32-30.

Figure 32-33. MDIO Link Status Change Interrupt (Masked) Register (LINKINTMASKED) (offset = 14h)

31	16
Reserved	
R-0	

15	2	1	0
Reserved		USERPHY1	USERPHY0
R-0		R/W1C-0	R/W1C-0

LEGEND: R/W = Read/Write; R = Read only; W1C = Write 1 to clear (writing a 0 has no effect); -*n* = value after reset

Table 32-30. MDIO Link Status Change Interrupt (Masked) Register (LINKINTMASKED) Field Descriptions

Bit	Field	Value	Description
31-2	Reserved	0	Reserved
1	USERPHY1		MDIO Link change interrupt, masked value. When asserted, the bit indicates that there was an MDIO link change event (that is, change in the LINK register) corresponding to the PHY address in USERPHYSEL1 and the corresponding LINKINTENB bit was set. Writing a 1 will clear the event, writing a 0 has no effect.
		0	No MDIO link change event.
		1	An MDIO link change event (change in the LINK register) corresponding to the PHY address in MDIO user PHY select register USERPHYSEL1 and the LINKINTENB bit in USERPHYSEL1 is set to 1.
0	USERPHY0		MDIO Link change interrupt, masked value. When asserted, the bit indicates that there was an MDIO link change event (that is, change in the LINK register) corresponding to the PHY address in USERPHYSEL0 and the corresponding LINKINTENB bit was set. Writing a 1 will clear the event, writing a 0 has no effect.
		0	No MDIO link change event.
		1	An MDIO link change event (change in the LINK register) corresponding to the PHY address in MDIO user PHY select register USERPHYSEL0 and the LINKINTENB bit in USERPHYSEL0 is set to 1.

32.4.7 MDIO User Command Complete Interrupt (Unmasked) Register (USERINTRAW)

The MDIO user command complete interrupt (unmasked) register (USERINTRAW) is shown in Figure 32-34 and described in Table 32-31.

Figure 32-34. MDIO User Command Complete Interrupt (Unmasked) Register (USERINTRAW) (offset = 20h)

31	16
Reserved	
R-0	

15	2	1	0
Reserved		USERACCESS1	USERACCESS0
R-0		R/W1C-0	R/W1C-0

LEGEND: R/W = Read/Write; R = Read only; W1C = Write 1 to clear (writing a 0 has no effect); -*n* = value after reset

Table 32-31. MDIO User Command Complete Interrupt (Unmasked) Register (USERINTRAW) Field Descriptions

Bit	Field	Value	Description
31-2	Reserved	0	Reserved
1	USERACCESS1		MDIO User command complete event bit. When asserted, the bit indicates that the previously scheduled PHY read or write command using the USERACCESS1 register has completed. Writing a 1 will clear the event, writing a 0 has no effect.
		0	No MDIO user command complete event.
		1	The previously scheduled PHY read or write command using MDIO user access register USERACCESS1 has completed.
0	USERACCESS0		MDIO User command complete event bit. When asserted, the bit indicates that the previously scheduled PHY read or write command using the USERACCESS0 register has completed. Writing a 1 will clear the event, writing a 0 has no effect.
		0	No MDIO user command complete event.
		1	The previously scheduled PHY read or write command using MDIO user access register USERACCESS0 has completed.

32.4.8 MDIO User Command Complete Interrupt (Masked) Register (USERINTMASKED)

The MDIO user command complete interrupt (masked) register (USERINTMASKED) is shown in Figure 32-35 and described in Table 32-32.

Figure 32-35. MDIO User Command Complete Interrupt (Masked) Register (USERINTMASKED) (offset = 24h)

31	16
Reserved	
R-0	

15	2	1	0
Reserved		USERACCESS1	USERACCESS0
R-0		R/W1C-0	R/W1C-0

LEGEND: R/W = Read/Write; R = Read only; W1C = Write 1 to clear (writing a 0 has no effect); -*n* = value after reset

Table 32-32. MDIO User Command Complete Interrupt (Masked) Register (USERINTMASKED) Field Descriptions

Bit	Field	Value	Description
31-2	Reserved	0	Reserved
1	USERACCESS1		Masked value of MDIO User command complete interrupt. When asserted, The bit indicates that the previously scheduled PHY read or write command using that particular USERACCESS1 register has completed. Writing a 1 will clear the interrupt, writing a 0 has no effect.
		0	No MDIO user command complete event.
		1	The previously scheduled PHY read or write command using MDIO user access register USERACCESS1 has completed and the corresponding bit in USERINTMASKSET is set to 1.
0	USERACCESS0		Masked value of MDIO User command complete interrupt. When asserted, The bit indicates that the previously scheduled PHY read or write command using that particular USERACCESS0 register has completed. Writing a 1 will clear the interrupt, writing a 0 has no effect.
		0	No MDIO user command complete event.
		1	The previously scheduled PHY read or write command using MDIO user access register USERACCESS0 has completed and the corresponding bit in USERINTMASKSET is set to 1.

32.4.9 MDIO User Command Complete Interrupt Mask Set Register (USERINTMASKSET)

The MDIO user command complete interrupt mask set register (USERINTMASKSET) is shown in Figure 32-36 and described in Table 32-33.

Figure 32-36. MDIO User Command Complete Interrupt Mask Set Register (USERINTMASKSET) (offset = 28h)

31–16
Reserved
R-0

15–2	1	0
Reserved	USERACCESS1	USERACCESS0
R-0	R/W1S-0	R/W1S-0

LEGEND: R/W = Read/Write; R = Read only; W1S = Write 1 to set (writing a 0 has no effect); -*n* = value after reset

Table 32-33. MDIO User Command Complete Interrupt Mask Set Register (USERINTMASKSET) Field Descriptions

Bit	Field	Value	Description
31-2	Reserved	0	Reserved
1	USERACCESS1		MDIO user interrupt mask set for USERINTMASKED[1]. Setting a bit to 1 will enable MDIO user command complete interrupts for the USERACCESS1 register. MDIO user interrupt for USERACCESS1 is disabled if the corresponding bit is 0. Writing a 0 to this bit has no effect.
		0	MDIO user command complete interrupts for the MDIO user access register USERACCESS0 is disabled.
		1	MDIO user command complete interrupts for the MDIO user access register USERACCESS0 is enabled.
0	USERACCESS0		MDIO user interrupt mask set for USERINTMASKED[0]. Setting a bit to 1 will enable MDIO user command complete interrupts for the USERACCESS0 register. MDIO user interrupt for USERACCESS0 is disabled if the corresponding bit is 0. Writing a 0 to this bit has no effect.
		0	MDIO user command complete interrupts for the MDIO user access register USERACCESS0 is disabled.
		1	MDIO user command complete interrupts for the MDIO user access register USERACCESS0 is enabled.

32.4.10 MDIO User Command Complete Interrupt Mask Clear Register (USERINTMASKCLEAR)

The MDIO user command complete interrupt mask clear register (USERINTMASKCLEAR) is shown in Figure 32-37 and described in Table 32-34.

Figure 32-37. MDIO User Command Complete Interrupt Mask Clear Register (USERINTMASKCLEAR) (offset = 2Ch)

31 16
Reserved
R-0

15 2	1	0
Reserved	USERACCESS1	USERACCESS0
R-0	R/W1C-0	R/W1C-0

LEGEND: R/W = Read/Write; R = Read only; W1C = Write 1 to clear (writing a 0 has no effect); -*n* = value after reset

Table 32-34. MDIO User Command Complete Interrupt Mask Clear Register (USERINTMASKCLEAR) Field Descriptions

Bit	Field	Value	Description
31-2	Reserved	0	Reserved
1	USERACCESS1		MDIO user command complete interrupt mask clear for USERINTMASKED[1]. Setting the bit to 1 will disable further user command complete interrupts for USERACCESS1. Writing a 0 to this bit has no effect.
		0	MDIO user command complete interrupts for the MDIO user access register USERACCESS1 is enabled.
		1	MDIO user command complete interrupts for the MDIO user access register USERACCESS1 is disabled.
0	USERACCESS0		MDIO user command complete interrupt mask clear for USERINTMASKED[0]. Setting the bit to 1 will disable further user command complete interrupts for USERACCESS0. Writing a 0 to this bit has no effect.
		0	MDIO user command complete interrupts for the MDIO user access register USERACCESS0 is enabled.
		1	MDIO user command complete interrupts for the MDIO user access register USERACCESS0 is disabled.

32.4.11 MDIO User Access Register 0 (USERACCESS0)

The MDIO user access register 0 (USERACCESS0) is shown in Figure 32-38 and described in Table 32-35.

Figure 32-38. MDIO User Access Register 0 (USERACCESS0) (offset = 80h)

31	30	29	28 26	25 21	20 16
GO	WRITE	ACK	Reserved	REGADR	PHYADR
R/W1S-0	R/W-0	R/W-0	R-0	R/W-0	R/W-0

15 0
DATA
R/W-0

LEGEND: R/W = Read/Write; R = Read only; W1S = Write 1 to set (writing a 0 has no effect); -*n* = value after reset

Table 32-35. MDIO User Access Register 0 (USERACCESS0) Field Descriptions

Bit	Field	Value	Description
31	GO	0-1	Go bit. Writing a 1 to this bit causes the MDIO state machine to perform an MDIO access when it is convenient for it to do so; this is not an instantaneous process. Writing a 0 to this bit has no effect. This bit is writeable only if the MDIO state machine is enabled. This bit will self clear when the requested access has been completed. Any writes to USERACCESS0 are blocked when the GO bit is 1.
30	WRITE		Write enable bit. Setting this bit to 1 causes the MDIO transaction to be a register write; otherwise, it is a register read.
		0	The user command is a read operation.
		1	The user command is a write operation.
29	ACK	0-1	Acknowledge bit. This bit is set if the PHY acknowledged the read transaction.
28-26	Reserved	0	Reserved
25-21	REGADR	0-1Fh	Register address bits. This field specifies the PHY register to be accessed for this transaction.
20-16	PHYADR	0-1Fh	PHY address bits. This field specifies the PHY to be accessed for this transaction.
15-0	DATA	0-FFFFh	User data bits. These bits specify the data value read from or to be written to the specified PHY register.

32.4.12 MDIO User PHY Select Register 0 (USERPHYSEL0)

The MDIO user PHY select register 0 (USERPHYSEL0) is shown in Figure 32-39 and described in Table 32-36.

Figure 32-39. MDIO User PHY Select Register 0 (USERPHYSEL0) (offset = 84h)

31	16
Reserved	
R-0	

15	8	7	6	5	4	0
Reserved		LINKSEL	LINKINTENB	Rsvd	PHYADRMON	
R-0		R/W-0	R/W-0	R-0	R/W-0	

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 32-36. MDIO User PHY Select Register 0 (USERPHYSEL0) Field Descriptions

Bit	Field	Value	Description
31-8	Reserved	0	Reserved
7	LINKSEL		Link status determination select bit. Default value is 0, which implies that the link status is determined by the MDIO state machine. This is the only option supported on this device.
		0	The link status is determined by the MDIO state machine.
		1	Not supported.
6	LINKINTENB		Link change interrupt enable. Set to 1 to enable link change status interrupts for PHY address specified in PHYADRMON. Link change interrupts are disabled if this bit is cleared to 0.
		0	Link change interrupts are disabled.
		1	Link change status interrupts for PHY address specified in PHYADDRMON bits are enabled.
5	Reserved	0	Reserved
4-0	PHYADRMON	0-1Fh	PHY address whose link status is to be monitored.

32.4.13 MDIO User Access Register 1 (USERACCESS1)

The MDIO user access register 1 (USERACCESS1) is shown in Figure 32-40 and described in Table 32-37.

Figure 32-40. MDIO User Access Register 1 (USERACCESS1) (offset = 88h)

31	30	29	28 26	25 21	20 16
GO	WRITE	ACK	Reserved	REGADR	PHYADR
R/W1S-0	R/W-0	R/W-0	R-0	R/W-0	R/W-0

15 0
DATA
R/W-0

LEGEND: R/W = Read/Write; R = Read only; W1S = Write 1 to set (writing a 0 has no effect); -*n* = value after reset

Table 32-37. MDIO User Access Register 1 (USERACCESS1) Field Descriptions

Bit	Field	Value	Description
31	GO	0-1	Go bit. Writing 1 to this bit causes the MDIO state machine to perform an MDIO access when it is convenient for it to do so; this is not an instantaneous process. Writing 0 to this bit has no effect. This bit is writeable only if the MDIO state machine is enabled. This bit will self clear when the requested access has been completed. Any writes to USERACCESS0 are blocked when the GO bit is 1.
30	WRITE		Write enable bit. Setting this bit to 1 causes the MDIO transaction to be a register write; otherwise, it is a register read.
		0	The user command is a read operation.
		1	The user command is a write operation.
29	ACK	0-1	Acknowledge bit. This bit is set if the PHY acknowledged the read transaction.
28-26	Reserved	0	Reserved
25-21	REGADR	0-1Fh	Register address bits. This field specifies the PHY register to be accessed for this transaction.
20-16	PHYADR	0-1Fh	PHY address bits. This field specifies the PHY to be accessed for this transaction.
15-0	DATA	0-FFFFh	User data bits. These bits specify the data value read from or to be written to the specified PHY register.

32.4.14 MDIO User PHY Select Register 1 (USERPHYSEL1)

The MDIO user PHY select register 1 (USERPHYSEL1) is shown in Figure 32-41 and described in Table 32-38.

Figure 32-41. MDIO User PHY Select Register 1 (USERPHYSEL1) (offset = 8Ch)

31–16
Reserved
R-0

15–8	7	6	5	4–0
Reserved	LINKSEL	LINKINTENB	Rsvd	PHYADRMON
R-0	R/W-0	R/W-0	R-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 32-38. MDIO User PHY Select Register 1 (USERPHYSEL1) Field Descriptions

Bit	Field	Value	Description
31-8	Reserved	0	Reserved
7	LINKSEL		Link status determination select bit. Default value is 0, which implies that the link status is determined by the MDIO state machine. This is the only option supported on this device.
		0	The link status is determined by the MDIO state machine.
		1	Not supported.
6	LINKINTENB		Link change interrupt enable. Set to 1 to enable link change status interrupts for the PHY address specified in PHYADRMON. Link change interrupts are disabled if this bit is cleared to 0.
		0	Link change interrupts are disabled.
		1	Link change status interrupts for PHY address specified in PHYADDRMON bits are enabled.
5	Reserved	0	PHY address whose link status is to be monitored.
4-0	PHYADRMON	0-1Fh	PHY address whose link status is to be monitored.

32.5 EMAC Module Registers

Table 32-39 lists the memory-mapped registers for the EMAC. The base address for these registers is FCF7 8000h.

Table 32-39. Ethernet Media Access Controller (EMAC) Registers

Offset	Acronym	Register Description	Section
0h	TXREVID	Transmit Revision ID Register	Section 32.5.1
4h	TXCONTROL	Transmit Control Register	Section 32.5.2
8h	TXTEARDOWN	Transmit Teardown Register	Section 32.5.3
10h	RXREVID	Receive Revision ID Register	Section 32.5.4
14h	RXCONTROL	Receive Control Register	Section 32.5.5
18h	RXTEARDOWN	Receive Teardown Register	Section 32.5.6
80h	TXINTSTATRAW	Transmit Interrupt Status (Unmasked) Register	Section 32.5.7
84h	TXINTSTATMASKED	Transmit Interrupt Status (Masked) Register	Section 32.5.8
88h	TXINTMASKSET	Transmit Interrupt Mask Set Register	Section 32.5.9
8Ch	TXINTMASKCLEAR	Transmit Interrupt Clear Register	Section 32.5.10
90h	MACINVECTOR	MAC Input Vector Register	Section 32.5.11
94h	MACEOIVECTOR	MAC End Of Interrupt Vector Register	Section 32.5.12
A0h	RXINTSTATRAW	Receive Interrupt Status (Unmasked) Register	Section 32.5.13
A4h	RXINTSTATMASKED	Receive Interrupt Status (Masked) Register	Section 32.5.14
A8h	RXINTMASKSET	Receive Interrupt Mask Set Register	Section 32.5.15
ACh	RXINTMASKCLEAR	Receive Interrupt Mask Clear Register	Section 32.5.16
B0h	MACINTSTATRAW	MAC Interrupt Status (Unmasked) Register	Section 32.5.17
B4h	MACINTSTATMASKED	MAC Interrupt Status (Masked) Register	Section 32.5.18
B8h	MACINTMASKSET	MAC Interrupt Mask Set Register	Section 32.5.19
BCh	MACINTMASKCLEAR	MAC Interrupt Mask Clear Register	Section 32.5.20
100h	RXMBPENABLE	Receive Multicast/Broadcast/Promiscuous Channel Enable Register	Section 32.5.21
104h	RXUNICASTSET	Receive Unicast Enable Set Register	Section 32.5.22
108h	RXUNICASTCLEAR	Receive Unicast Clear Register	Section 32.5.23
10Ch	RXMAXLEN	Receive Maximum Length Register	Section 32.5.24
110h	RXBUFFEROFFSET	Receive Buffer Offset Register	Section 32.5.25
114h	RXFILTERLOWTHRESH	Receive Filter Low Priority Frame Threshold Register	Section 32.5.26
120h	RX0FLOWTHRESH	Receive Channel 0 Flow Control Threshold Register	Section 32.5.27
124h	RX1FLOWTHRESH	Receive Channel 1 Flow Control Threshold Register	Section 32.5.27
128h	RX2FLOWTHRESH	Receive Channel 2 Flow Control Threshold Register	Section 32.5.27
12Ch	RX3FLOWTHRESH	Receive Channel 3 Flow Control Threshold Register	Section 32.5.27
130h	RX4FLOWTHRESH	Receive Channel 4 Flow Control Threshold Register	Section 32.5.27
134h	RX5FLOWTHRESH	Receive Channel 5 Flow Control Threshold Register	Section 32.5.27
138h	RX6FLOWTHRESH	Receive Channel 6 Flow Control Threshold Register	Section 32.5.27
13Ch	RX7FLOWTHRESH	Receive Channel 7 Flow Control Threshold Register	Section 32.5.27
140h	RX0FREEBUFFER	Receive Channel 0 Free Buffer Count Register	Section 32.5.28
144h	RX1FREEBUFFER	Receive Channel 1 Free Buffer Count Register	Section 32.5.28
148h	RX2FREEBUFFER	Receive Channel 2 Free Buffer Count Register	Section 32.5.28
14Ch	RX3FREEBUFFER	Receive Channel 3 Free Buffer Count Register	Section 32.5.28
150h	RX4FREEBUFFER	Receive Channel 4 Free Buffer Count Register	Section 32.5.28
154h	RX5FREEBUFFER	Receive Channel 5 Free Buffer Count Register	Section 32.5.28
158h	RX6FREEBUFFER	Receive Channel 6 Free Buffer Count Register	Section 32.5.28
15Ch	RX7FREEBUFFER	Receive Channel 7 Free Buffer Count Register	Section 32.5.28
160h	MACCONTROL	MAC Control Register	Section 32.5.29

Table 32-39. Ethernet Media Access Controller (EMAC) Registers (continued)

Offset	Acronym	Register Description	Section
164h	MACSTATUS	MAC Status Register	Section 32.5.30
168h	EMCONTROL	Emulation Control Register	Section 32.5.31
16Ch	FIFOCONTROL	FIFO Control Register	Section 32.5.32
170h	MACCONFIG	MAC Configuration Register	Section 32.5.33
174h	SOFTRESET	Soft Reset Register	Section 32.5.34
1D0h	MACSRCADDRLO	MAC Source Address Low Bytes Register	Section 32.5.35
1D4h	MACSRCADDRHI	MAC Source Address High Bytes Register	Section 32.5.36
1D8h	MACHASH1	MAC Hash Address Register 1	Section 32.5.37
1DCh	MACHASH2	MAC Hash Address Register 2	Section 32.5.38
1E0h	BOFFTEST	Back Off Test Register	Section 32.5.39
1E4h	TPACETEST	Transmit Pacing Algorithm Test Register	Section 32.5.40
1E8h	RXPAUSE	Receive Pause Timer Register	Section 32.5.41
1ECh	TXPAUSE	Transmit Pause Timer Register	Section 32.5.42
500h	MACADDRLO	MAC Address Low Bytes Register	Section 32.5.43
504h	MACADDRHI	MAC Address High Bytes Register	Section 32.5.44
508h	MACINDEX	MAC Index Register	Section 32.5.45
600h	TX0HDP	Transmit Channel 0 DMA Head Descriptor Pointer Register	Section 32.5.46
604h	TX1HDP	Transmit Channel 1 DMA Head Descriptor Pointer Register	Section 32.5.46
608h	TX2HDP	Transmit Channel 2 DMA Head Descriptor Pointer Register	Section 32.5.46
60Ch	TX3HDP	Transmit Channel 3 DMA Head Descriptor Pointer Register	Section 32.5.46
610h	TX4HDP	Transmit Channel 4 DMA Head Descriptor Pointer Register	Section 32.5.46
614h	TX5HDP	Transmit Channel 5 DMA Head Descriptor Pointer Register	Section 32.5.46
618h	TX6HDP	Transmit Channel 6 DMA Head Descriptor Pointer Register	Section 32.5.46
61Ch	TX7HDP	Transmit Channel 7 DMA Head Descriptor Pointer Register	Section 32.5.46
620h	RX0HDP	Receive Channel 0 DMA Head Descriptor Pointer Register	Section 32.5.47
624h	RX1HDP	Receive Channel 1 DMA Head Descriptor Pointer Register	Section 32.5.47
628h	RX2HDP	Receive Channel 2 DMA Head Descriptor Pointer Register	Section 32.5.47
62Ch	RX3HDP	Receive Channel 3 DMA Head Descriptor Pointer Register	Section 32.5.47
630h	RX4HDP	Receive Channel 4 DMA Head Descriptor Pointer Register	Section 32.5.47
634h	RX5HDP	Receive Channel 5 DMA Head Descriptor Pointer Register	Section 32.5.47
638h	RX6HDP	Receive Channel 6 DMA Head Descriptor Pointer Register	Section 32.5.47
63Ch	RX7HDP	Receive Channel 7 DMA Head Descriptor Pointer Register	Section 32.5.47
640h	TX0CP	Transmit Channel 0 Completion Pointer Register	Section 32.5.48
644h	TX1CP	Transmit Channel 1 Completion Pointer Register	Section 32.5.48
648h	TX2CP	Transmit Channel 2 Completion Pointer Register	Section 32.5.48
64Ch	TX3CP	Transmit Channel 3 Completion Pointer Register	Section 32.5.48
650h	TX4CP	Transmit Channel 4 Completion Pointer Register	Section 32.5.48
654h	TX5CP	Transmit Channel 5 Completion Pointer Register	Section 32.5.48
658h	TX6CP	Transmit Channel 6 Completion Pointer Register	Section 32.5.48
65Ch	TX7CP	Transmit Channel 7 Completion Pointer Register	Section 32.5.48
660h	RX0CP	Receive Channel 0 Completion Pointer Register	Section 32.5.49
664h	RX1CP	Receive Channel 1 Completion Pointer Register	Section 32.5.49
668h	RX2CP	Receive Channel 2 Completion Pointer Register	Section 32.5.49
66Ch	RX3CP	Receive Channel 3 Completion Pointer Register	Section 32.5.49
670h	RX4CP	Receive Channel 4 Completion Pointer Register	Section 32.5.49
674h	RX5CP	Receive Channel 5 Completion Pointer Register	Section 32.5.49
678h	RX6CP	Receive Channel 6 Completion Pointer Register	Section 32.5.49

Table 32-39. Ethernet Media Access Controller (EMAC) Registers (continued)

Offset	Acronym	Register Description	Section
67Ch	RX7CP	Receive Channel 7 Completion Pointer Register	Section 32.5.49
		Network Statistics Registers	
200h	RXGOODFRAMES	Good Receive Frames Register	Section 32.5.50.1
204h	RXBCASTFRAMES	Broadcast Receive Frames Register	Section 32.5.50.2
208h	RXMCASTFRAMES	Multicast Receive Frames Register	Section 32.5.50.3
20Ch	RXPAUSEFRAMES	Pause Receive Frames Register	Section 32.5.50.4
210h	RXCRCERRORS	Receive CRC Errors Register	Section 32.5.50.5
214h	RXALIGNCODEERRORS	Receive Alignment/Code Errors Register	Section 32.5.50.6
218h	RXOVERSIZED	Receive Oversized Frames Register	Section 32.5.50.7
21Ch	RXJABBER	Receive Jabber Frames Register	Section 32.5.50.8
220h	RXUNDERSIZED	Receive Undersized Frames Register	Section 32.5.50.9
224h	RXFRAGMENTS	Receive Frame Fragments Register	Section 32.5.50.10
228h	RXFILTERED	Filtered Receive Frames Register	Section 32.5.50.11
22Ch	RXQOSFILTERED	Receive QOS Filtered Frames Register	Section 32.5.50.12
230h	RXOCTETS	Receive Octet Frames Register	Section 32.5.50.13
234h	TXGOODFRAMES	Good Transmit Frames Register	Section 32.5.50.14
238h	TXBCASTFRAMES	Broadcast Transmit Frames Register	Section 32.5.50.15
23Ch	TXMCASTFRAMES	Multicast Transmit Frames Register	Section 32.5.50.16
240h	TXPAUSEFRAMES	Pause Transmit Frames Register	Section 32.5.50.17
244h	TXDEFERRED	Deferred Transmit Frames Register	Section 32.5.50.18
248h	TXCOLLISION	Transmit Collision Frames Register	Section 32.5.50.19
24Ch	TXSINGLECOLL	Transmit Single Collision Frames Register	Section 32.5.50.20
250h	TXMULTICOLL	Transmit Multiple Collision Frames Register	Section 32.5.50.21
254h	TXEXCESSIVECOLL	Transmit Excessive Collision Frames Register	Section 32.5.50.22
258h	TXLATECOLL	Transmit Late Collision Frames Register	Section 32.5.50.23
25Ch	TXUNDERRUN	Transmit Underrun Error Register	Section 32.5.50.24
260h	TXCARRIERSENSE	Transmit Carrier Sense Errors Register	Section 32.5.50.25
264h	TXOCTETS	Transmit Octet Frames Register	Section 32.5.50.26
268h	FRAME64	Transmit and Receive 64 Octet Frames Register	Section 32.5.50.27
26Ch	FRAME65T127	Transmit and Receive 65 to 127 Octet Frames Register	Section 32.5.50.28
270h	FRAME128T255	Transmit and Receive 128 to 255 Octet Frames Register	Section 32.5.50.29
274h	FRAME256T511	Transmit and Receive 256 to 511 Octet Frames Register	Section 32.5.50.30
278h	FRAME512T1023	Transmit and Receive 512 to 1023 Octet Frames Register	Section 32.5.50.31
27Ch	FRAME1024TUP	Transmit and Receive 1024 to RXMAXLEN Octet Frames Register	Section 32.5.50.32
280h	NETOCTETS	Network Octet Frames Register	Section 32.5.50.33
284h	RXSOFOVERRUNS	Receive FIFO or DMA Start of Frame Overruns Register	Section 32.5.50.34
288h	RXMOFOVERRUNS	Receive FIFO or DMA Middle of Frame Overruns Register	Section 32.5.50.35
28Ch	RXDMAOVERRUNS	Receive DMA Overruns Register	Section 32.5.50.36

32.5.1 Transmit Revision ID Register (TXREVID)

The transmit revision ID register (TXREVID) is shown in Figure 32-42 and described in Table 32-40.

Figure 32-42. Transmit Revision ID Register (TXREVID) (offset = 00h)

31 ... 0
TXREV
R-4EC0 020Dh

LEGEND: R = Read only; -*n* = value after reset

Table 32-40. Transmit Revision ID Register (TXREVID) Field Descriptions

Bit	Field	Value	Description
31-0	TXREV		Transmit module revision.
		4EC0 020Dh	Current transmit revision value.

32.5.2 Transmit Control Register (TXCONTROL)

The transmit control register (TXCONTROL) is shown in Figure 32-43 and described in Table 32-41.

Figure 32-43. Transmit Control Register (TXCONTROL) (offset = 04h)

31 ... 16
Reserved
R-0

15 ... 1	0
Reserved	TXEN
R-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 32-41. Transmit Control Register (TXCONTROL) Field Descriptions

Bit	Field	Value	Description
31-1	Reserved	0	Reserved
0	TXEN		Transmit enable.
		0	Transmit is disabled.
		1	Transmit is enabled.

32.5.3 Transmit Teardown Register (TXTEARDOWN)

The transmit teardown register (TXTEARDOWN) is shown in Figure 32-44 and described in Table 32-42.

Figure 32-44. Transmit Teardown Register (TXTEARDOWN) (offset = 08h)

31	16
Reserved	
R-0	

15	3	2	0
Reserved		TXTDNCH	
R-0		R/W-0	

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 32-42. Transmit Teardown Register (TXTEARDOWN) Field Descriptions

Bit	Field	Value	Description
31-3	Reserved	0	Reserved
2-0	TXTDNCH	0-7h	Transmit teardown channel. The transmit channel teardown is commanded by writing the encoded value of the transmit channel to be torn down. The teardown register is read as 0.
		0	Teardown transmit channel 0.
		1h	Teardown transmit channel 1.
		2h	Teardown transmit channel 2.
		3h	Teardown transmit channel 3.
		4h	Teardown transmit channel 4.
		5h	Teardown transmit channel 5.
		6h	Teardown transmit channel 6.
		7h	Teardown transmit channel 7.

32.5.4 Receive Revision ID Register (RXREVID)

The receive revision ID register (RXREVID) is shown in Figure 32-45 and described in Table 32-43.

Figure 32-45. Receive Revision ID Register (RXREVID) (offset = 10h)

31	0
RXREV	
R-4EC0 020Dh	

LEGEND: R = Read only; -*n* = value after reset

Table 32-43. Receive Revision ID Register (RXREVID) Field Descriptions

Bit	Field	Value	Description
31-0	RXREV		Receive module revision.
		4EC0 020Dh	Current receive revision value.

32.5.5 Receive Control Register (RXCONTROL)

The receive control register (RXCONTROL) is shown in Figure 32-46 and described in Table 32-44.

Figure 32-46. Receive Control Register (RXCONTROL) (offset = 14h)

31–16
Reserved
R-0

15–1	0
Reserved	RXEN
R-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 32-44. Receive Control Register (RXCONTROL) Field Descriptions

Bit	Field	Value	Description
31-1	Reserved	0	Reserved
0	RXEN		Receive enable.
		0	Receive is disabled.
		1	Receive is enabled.

32.5.6 Receive Teardown Register (RXTEARDOWN)

The receive teardown register (RXTEARDOWN) is shown in Figure 32-47 and described in Table 32-45.

Figure 32-47. Receive Teardown Register (RXTEARDOWN) (offset = 18h)

31–16
Reserved
R-0

15–3	2–0
Reserved	RXTDNCH
R-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 32-45. Receive Teardown Register (RXTEARDOWN) Field Descriptions

Bit	Field	Value	Description
31-3	Reserved	0	Reserved
2-0	RXTDNCH		Receive teardown channel. The receive channel teardown is commanded by writing the encoded value of the receive channel to be torn down. The teardown register is read as 0.
		0	Teardown receive channel 0.
		1h	Teardown receive channel 1.
		2h	Teardown receive channel 2.
		3h	Teardown receive channel 3.
		4h	Teardown receive channel 4.
		5h	Teardown receive channel 5.
		6h	Teardown receive channel 6.
		7h	Teardown receive channel 7.

32.5.7 Transmit Interrupt Status (Unmasked) Register (TXINTSTATRAW)

The transmit interrupt status (unmasked) register (TXINTSTATRAW) is shown in Figure 32-48 and described in Table 32-46.

Figure 32-48. Transmit Interrupt Status (Unmasked) Register (TXINTSTATRAW) (offset = 80h)

31-16
Reserved
R-0

15-8
Reserved
R-0

7	6	5	4	3	2	1	0
TX7PEND	TX6PEND	TX5PEND	TX4PEND	TX3PEND	TX2PEND	TX1PEND	TX0PEND
R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0

LEGEND: R = Read only; -*n* = value after reset

Table 32-46. Transmit Interrupt Status (Unmasked) Register (TXINTSTATRAW) Field Descriptions

Bit	Field	Value	Description
31-8	Reserved	0	Reserved
7	TX7PEND	0-1	TX7PEND raw interrupt read (before mask).
6	TX6PEND	0-1	TX6PEND raw interrupt read (before mask).
5	TX5PEND	0-1	TX5PEND raw interrupt read (before mask).
4	TX4PEND	0-1	TX4PEND raw interrupt read (before mask).
3	TX3PEND	0-1	TX3PEND raw interrupt read (before mask).
2	TX2PEND	0-1	TX2PEND raw interrupt read (before mask).
1	TX1PEND	0-1	TX1PEND raw interrupt read (before mask).
0	TX0PEND	0-1	TX0PEND raw interrupt read (before mask).

32.5.8 Transmit Interrupt Status (Masked) Register (TXINTSTATMASKED)

The transmit interrupt status (masked) register (TXINTSTATMASKED) is shown in Figure 32-49 and described in Table 32-47.

Figure 32-49. Transmit Interrupt Status (Masked) Register (TXINTSTATMASKED) (offset = 84h)

31	16
Reserved	
R-0	

15	8
Reserved	
R-0	

7	6	5	4	3	2	1	0
TX7PEND	TX6PEND	TX5PEND	TX4PEND	TX3PEND	TX2PEND	TX1PEND	TX0PEND
R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0

LEGEND: R = Read only; -*n* = value after reset

Table 32-47. Transmit Interrupt Status (Masked) Register (TXINTSTATMASKED) Field Descriptions

Bit	Field	Value	Description
31-8	Reserved	0	Reserved
7	TX7PEND	0-1	TX7PEND masked interrupt read.
6	TX6PEND	0-1	TX6PEND masked interrupt read.
5	TX5PEND	0-1	TX5PEND masked interrupt read.
4	TX4PEND	0-1	TX4PEND masked interrupt read.
3	TX3PEND	0-1	TX3PEND masked interrupt read.
2	TX2PEND	0-1	TX2PEND masked interrupt read.
1	TX1PEND	0-1	TX1PEND masked interrupt read.
0	TX0PEND	0-1	TX0PEND masked interrupt read.

32.5.9 Transmit Interrupt Mask Set Register (TXINTMASKSET)

The transmit interrupt mask set register (TXINTMASKSET) is shown in Figure 32-50 and described in Table 32-48.

Figure 32-50. Transmit Interrupt Mask Set Register (TXINTMASKSET) (offset = 88h)

31	16
Reserved	
R-0	

15	8
Reserved	
R-0	

7	6	5	4	3	2	1	0
TX7MASK	TX6MASK	TX5MASK	TX4MASK	TX3MASK	TX2MASK	TX1MASK	TX0MASK
R/W1S-0	R/W1S-0	R/W1S-0	R/W1S-0	R/W1S-0	R/W1S-0	R/W1S-0	R/W1S-0

LEGEND: R/W = Read/Write; R = Read only; W1S = Write 1 to set (writing a 0 has no effect); -*n* = value after reset

Table 32-48. Transmit Interrupt Mask Set Register (TXINTMASKSET) Field Descriptions

Bit	Field	Value	Description
31-8	Reserved	0	Reserved
7	TX7MASK	0-1	Transmit channel 7 interrupt mask set bit. Write 1 to enable interrupt, a write of 0 has no effect.
6	TX6MASK	0-1	Transmit channel 6 interrupt mask set bit. Write 1 to enable interrupt, a write of 0 has no effect.
5	TX5MASK	0-1	Transmit channel 5 interrupt mask set bit. Write 1 to enable interrupt, a write of 0 has no effect.
4	TX4MASK	0-1	Transmit channel 4 interrupt mask set bit. Write 1 to enable interrupt, a write of 0 has no effect.
3	TX3MASK	0-1	Transmit channel 3 interrupt mask set bit. Write 1 to enable interrupt, a write of 0 has no effect.
2	TX2MASK	0-1	Transmit channel 2 interrupt mask set bit. Write 1 to enable interrupt, a write of 0 has no effect.
1	TX1MASK	0-1	Transmit channel 1 interrupt mask set bit. Write 1 to enable interrupt, a write of 0 has no effect.
0	TX0MASK	0-1	Transmit channel 0 interrupt mask set bit. Write 1 to enable interrupt, a write of 0 has no effect.

32.5.10 Transmit Interrupt Mask Clear Register (TXINTMASKCLEAR)

The transmit interrupt mask clear register (TXINTMASKCLEAR) is shown in Figure 32-51 and described in Table 32-49.

Figure 32-51. Transmit Interrupt Mask Clear Register (TXINTMASKCLEAR) (offset = 8Ch)

31	16
Reserved	
R-0	

15	8
Reserved	
R-0	

7	6	5	4	3	2	1	0
TX7MASK	TX6MASK	TX5MASK	TX4MASK	TX3MASK	TX2MASK	TX1MASK	TX0MASK
R/W1C-0	R/W1C-0	R/W1C-0	R/W1C-0	R/W1C-0	R/W1C-0	R/W1C-0	R/W1C-0

LEGEND: R/W = Read/Write; R = Read only; W1C = Write 1 to clear (writing a 0 has no effect); -*n* = value after reset

Table 32-49. Transmit Interrupt Mask Clear Register (TXINTMASKCLEAR) Field Descriptions

Bit	Field	Value	Description
31-8	Reserved	0	Reserved
7	TX7MASK	0-1	Transmit channel 7 interrupt mask clear bit. Write 1 to disable interrupt, a write of 0 has no effect.
6	TX6MASK	0-1	Transmit channel 6 interrupt mask clear bit. Write 1 to disable interrupt, a write of 0 has no effect.
5	TX5MASK	0-1	Transmit channel 5 interrupt mask clear bit. Write 1 to disable interrupt, a write of 0 has no effect.
4	TX4MASK	0-1	Transmit channel 4 interrupt mask clear bit. Write 1 to disable interrupt, a write of 0 has no effect.
3	TX3MASK	0-1	Transmit channel 3 interrupt mask clear bit. Write 1 to disable interrupt, a write of 0 has no effect.
2	TX2MASK	0-1	Transmit channel 2 interrupt mask clear bit. Write 1 to disable interrupt, a write of 0 has no effect.
1	TX1MASK	0-1	Transmit channel 1 interrupt mask clear bit. Write 1 to disable interrupt, a write of 0 has no effect.
0	TX0MASK	0-1	Transmit channel 0 interrupt mask clear bit. Write 1 to disable interrupt, a write of 0 has no effect.

32.5.11 MAC Input Vector Register (MACINVECTOR)

The MAC input vector register (MACINVECTOR) is shown in Figure 32-52 and described in Table 32-50.

Figure 32-52. MAC Input Vector Register (MACINVECTOR) (offset = 90h)

31-28	27	26	25	24	23-16
Reserved	STATPEND	HOSTPEND	LINKINT0	USERINT0	TXPEND
R-0	R-0	R-0	R-0	R-0	R-0

15-8	7-0
RXTHRESHPEND	RXPEND
R-0	R-0

LEGEND: R = Read only; -*n* = value after reset

Table 32-50. MAC Input Vector Register (MACINVECTOR) Field Descriptions

Bit	Field	Value	Description
31-28	Reserved	0	Reserved
27	STATPEND	0-1	EMAC module statistics interrupt (STATPEND) pending status bit.
26	HOSTPEND	0-1	EMAC module host error interrupt (HOSTPEND) pending status bit.
25	LINKINT0	0-1	MDIO module USERPHYSEL0 (LINKINT0) status bit.
24	USERINT0	0-1	MDIO module USERACCESS0 (USERINT0) status bit.
23-16	TXPEND	0-FFh	Transmit channels 0-7 interrupt (TX*n*PEND) pending status. Bit 16 is TX0PEND.
15-8	RXTHRESHPEND	0-FFh	Receive channels 0-7 interrupt (RX*n*THRESHPEND) pending status. Bit 8 is RX0THRESHPEND.
7-0	RXPEND	0-FFh	Receive channels 0-7 interrupt (RX*n*PEND) pending status bit. Bit 0 is RX0PEND.

32.5.12 MAC End Of Interrupt Vector Register (MACEOIVECTOR)

The MAC end of interrupt vector register (MACEOIVECTOR) is shown in Figure 32-53 and described in Table 32-51.

Figure 32-53. MAC End Of Interrupt Vector Register (MACEOIVECTOR) (offset = 94h)

31	16
Reserved	
R-0	

15	5	4	0
Reserved		INTVECT	
R-0		R/W-0	

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 32-51. MAC End Of Interrupt Vector Register (MACEOIVECTOR) Field Descriptions

Bit	Field	Value	Description
31-5	Reserved	0	Reserved
4-0	INTVECT		Acknowledge EMAC control module interrupts.
		0h	Acknowledge C0RXTHRESH Interrupt
		1h	Acknowledge C0RX Interrupt
		2h	Acknowledge C0TX Interrupt
		3h	Acknowledge C0MISC Interrupt (STATPEND, HOSTPEND, MDIO LINKINT0, MDIO USERINT0)
		4h	Acknowledge C1RXTHRESH Interrupt
		5h	Acknowledge C1RX Interrupt
		6h	Acknowledge C1TX Interrupt
		7h	Acknowledge C1MISC Interrupt (STATPEND, HOSTPEND, MDIO LINKINT0, MDIO USERINT0)
		8h	Acknowledge C2RXTHRESH Interrupt
		9h	Acknowledge C2RX Interrupt
		Ah	Acknowledge C2TX Interrupt
		Bh	Acknowledge C2MISC Interrupt (STATPEND, HOSTPEND, MDIO LINKINT0, MDIO USERINT0)
		Ch-1Fh	Reserved

32.5.13 Receive Interrupt Status (Unmasked) Register (RXINTSTATRAW)

The receive interrupt status (unmasked) register (RXINTSTATRAW) is shown in Figure 32-54 and described in Table 32-52.

Figure 32-54. Receive Interrupt Status (Unmasked) Register (RXINTSTATRAW) (offset = A0h)

31–16
Reserved
R-0

15	14	13	12	11	10	9	8
RX7THRESH PEND	RX6THRESH PEND	RX5THRESH PEND	RX4THRESH PEND	RX3THRESH PEND	RX2THRESH PEND	RX1THRESH PEND	RX0THRESH PEND
R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0

7	6	5	4	3	2	1	0
RX7PEND	RX6PEND	RX5PEND	RX4PEND	RX3PEND	RX2PEND	RX1PEND	RX0PEND
R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0

LEGEND: R = Read only; -*n* = value after reset

Table 32-52. Receive Interrupt Status (Unmasked) Register (RXINTSTATRAW) Field Descriptions

Bit	Field	Value	Description
31-16	Reserved	0	Reserved
15	RX7THRESHPEND	0-1	RX7THRESHPEND raw interrupt read (before mask).
14	RX6THRESHPEND	0-1	RX6THRESHPEND raw interrupt read (before mask).
13	RX5THRESHPEND	0-1	RX5THRESHPEND raw interrupt read (before mask).
12	RX4THRESHPEND	0-1	RX4THRESHPEND raw interrupt read (before mask).
11	RX3THRESHPEND	0-1	RX3THRESHPEND raw interrupt read (before mask).
10	RX2THRESHPEND	0-1	RX2THRESHPEND raw interrupt read (before mask).
9	RX1THRESHPEND	0-1	RX1THRESHPEND raw interrupt read (before mask).
8	RX0THRESHPEND	0-1	RX0THRESHPEND raw interrupt read (before mask).
7	RX7PEND	0-1	RX7PEND raw interrupt read (before mask).
6	RX6PEND	0-1	RX6PEND raw interrupt read (before mask).
5	RX5PEND	0-1	RX5PEND raw interrupt read (before mask).
4	RX4PEND	0-1	RX4PEND raw interrupt read (before mask).
3	RX3PEND	0-1	RX3PEND raw interrupt read (before mask).
2	RX2PEND	0-1	RX2PEND raw interrupt read (before mask).
1	RX1PEND	0-1	RX1PEND raw interrupt read (before mask).
0	RX0PEND	0-1	RX0PEND raw interrupt read (before mask).

32.5.14 Receive Interrupt Status (Masked) Register (RXINTSTATMASKED)

The receive interrupt status (masked) register (RXINTSTATMASKED) is shown in Figure 32-55 and described in Table 32-53.

Figure 32-55. Receive Interrupt Status (Masked) Register (RXINTSTATMASKED) (offset = A4h)

31-16
Reserved
R-0

15	14	13	12	11	10	9	8
RX7THRESH PEND	RX6THRESH PEND	RX5THRESH PEND	RX4THRESH PEND	RX3THRESH PEND	RX2THRESH PEND	RX1THRESH PEND	RX0THRESH PEND
R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0

7	6	5	4	3	2	1	0
RX7PEND	RX6PEND	RX5PEND	RX4PEND	RX3PEND	RX2PEND	RX1PEND	RX0PEND
R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0

LEGEND: R = Read only; -*n* = value after reset

Table 32-53. Receive Interrupt Status (Masked) Register (RXINTSTATMASKED) Field Descriptions

Bit	Field	Value	Description
31-16	Reserved	0	Reserved
15	RX7THRESHPEND	0-1	RX7THRESHPEND masked interrupt read.
14	RX6THRESHPEND	0-1	RX6THRESHPEND masked interrupt read.
13	RX5THRESHPEND	0-1	RX5THRESHPEND masked interrupt read.
12	RX4THRESHPEND	0-1	RX4THRESHPEND masked interrupt read.
11	RX3THRESHPEND	0-1	RX3THRESHPEND masked interrupt read.
10	RX2THRESHPEND	0-1	RX2THRESHPEND masked interrupt read.
9	RX1THRESHPEND	0-1	RX1THRESHPEND masked interrupt read.
8	RX0THRESHPEND	0-1	RX0THRESHPEND masked interrupt read.
7	RX7PEND	0-1	RX7PEND masked interrupt read.
6	RX6PEND	0-1	RX6PEND masked interrupt read.
5	RX5PEND	0-1	RX5PEND masked interrupt read.
4	RX4PEND	0-1	RX4PEND masked interrupt read.
3	RX3PEND	0-1	RX3PEND masked interrupt read.
2	RX2PEND	0-1	RX2PEND masked interrupt read.
1	RX1PEND	0-1	RX1PEND masked interrupt read.
0	RX0PEND	0-1	RX0PEND masked interrupt read.

32.5.15 Receive Interrupt Mask Set Register (RXINTMASKSET)

The receive interrupt mask set register (RXINTMASKSET) is shown in Figure 32-56 and described in Table 32-54.

Figure 32-56. Receive Interrupt Mask Set Register (RXINTMASKSET) (offset = A8h)

31	16
Reserved	
R-0	

15	14	13	12	11	10	9	8
RX7THRESH MASK	RX6THRESH MASK	RX5THRESH MASK	RX4THRESH MASK	RX3THRESH MASK	RX2THRESH MASK	RX1THRESH MASK	RX0THRESH MASK
R/W1S-0	R/W1S-0	R/W1S-0	R/W1S-0	R/W1S-0	R/W1S-0	R/W1S-0	R/W1S-0

7	6	5	4	3	2	1	0
RX7MASK	RX6MASK	RX5MASK	RX4MASK	RX3MASK	RX2MASK	RX1MASK	RX0MASK
R/W1S-0	R/W1S-0	R/W1S-0	R/W1S-0	R/W1S-0	R/W1S-0	R/W1S-0	R/W1S-0

LEGEND: R/W = Read/Write; R = Read only; W1S = Write 1 to set (writing a 0 has no effect); -*n* = value after reset

Table 32-54. Receive Interrupt Mask Set Register (RXINTMASKSET) Field Descriptions

Bit	Field	Value	Description
31-16	Reserved	0	Reserved
15	RX7THRESHMASK	0-1	Receive channel 7 threshold mask set bit. Write 1 to enable interrupt; a write of 0 has no effect.
14	RX6THRESHMASK	0-1	Receive channel 6 threshold mask set bit. Write 1 to enable interrupt; a write of 0 has no effect.
13	RX5THRESHMASK	0-1	Receive channel 5 threshold mask set bit. Write 1 to enable interrupt; a write of 0 has no effect.
12	RX4THRESHMASK	0-1	Receive channel 4 threshold mask set bit. Write 1 to enable interrupt; a write of 0 has no effect.
11	RX3THRESHMASK	0-1	Receive channel 3 threshold mask set bit. Write 1 to enable interrupt; a write of 0 has no effect.
10	RX2THRESHMASK	0-1	Receive channel 2 threshold mask set bit. Write 1 to enable interrupt; a write of 0 has no effect.
9	RX1THRESHMASK	0-1	Receive channel 1 threshold mask set bit. Write 1 to enable interrupt; a write of 0 has no effect.
8	RX0THRESHMASK	0-1	Receive channel 0 threshold mask set bit. Write 1 to enable interrupt; a write of 0 has no effect.
7	RX7MASK	0-1	Receive channel 7 mask set bit. Write 1 to enable interrupt; a write of 0 has no effect.
6	RX6MASK	0-1	Receive channel 6 mask set bit. Write 1 to enable interrupt; a write of 0 has no effect.
5	RX5MASK	0-1	Receive channel 5 mask set bit. Write 1 to enable interrupt; a write of 0 has no effect.
4	RX4MASK	0-1	Receive channel 4 mask set bit. Write 1 to enable interrupt; a write of 0 has no effect.
3	RX3MASK	0-1	Receive channel 3 mask set bit. Write 1 to enable interrupt; a write of 0 has no effect.
2	RX2MASK	0-1	Receive channel 2 mask set bit. Write 1 to enable interrupt; a write of 0 has no effect.
1	RX1MASK	0-1	Receive channel 1 mask set bit. Write 1 to enable interrupt; a write of 0 has no effect.
0	RX0MASK	0-1	Receive channel 0 mask set bit. Write 1 to enable interrupt; a write of 0 has no effect.

32.5.16 Receive Interrupt Mask Clear Register (RXINTMASKCLEAR)

The receive interrupt mask clear register (RXINTMASKCLEAR) is shown in Figure 32-57 and described in Table 32-55.

Figure 32-57. Receive Interrupt Mask Clear Register (RXINTMASKCLEAR) (offset = ACh)

31-16
Reserved
R-0

15	14	13	12	11	10	9	8
RX7THRESH MASK	RX6THRESH MASK	RX5THRESH MASK	RX4THRESH MASK	RX3THRESH MASK	RX2THRESH MASK	RX1THRESH MASK	RX0THRESH MASK
R/W1C-0	R/W1C-0	R/W1C-0	R/W1C-0	R/W1C-0	R/W1C-0	R/W1C-0	R/W1C-0

7	6	5	4	3	2	1	0
RX7MASK	RX6MASK	RX5MASK	RX4MASK	RX3MASK	RX2MASK	RX1MASK	RX0MASK
R/W1C-0	R/W1C-0	R/W1C-0	R/W1C-0	R/W1C-0	R/W1C-0	R/W1C-0	R/W1C-0

LEGEND: R/W = Read/Write; R = Read only; W1C = Write 1 to clear (writing a 0 has no effect); -*n* = value after reset

Table 32-55. Receive Interrupt Mask Clear Register (RXINTMASKCLEAR) Field Descriptions

Bit	Field	Value	Description
31-16	Reserved	0	Reserved
15	RX7THRESHMASK	0-1	Receive channel 7 threshold mask clear bit. Write 1 to disable interrupt; a write of 0 has no effect.
14	RX6THRESHMASK	0-1	Receive channel 6 threshold mask clear bit. Write 1 to disable interrupt; a write of 0 has no effect.
13	RX5THRESHMASK	0-1	Receive channel 5 threshold mask clear bit. Write 1 to disable interrupt; a write of 0 has no effect.
12	RX4THRESHMASK	0-1	Receive channel 4 threshold mask clear bit. Write 1 to disable interrupt; a write of 0 has no effect.
11	RX3THRESHMASK	0-1	Receive channel 3 threshold mask clear bit. Write 1 to disable interrupt; a write of 0 has no effect.
10	RX2THRESHMASK	0-1	Receive channel 2 threshold mask clear bit. Write 1 to disable interrupt; a write of 0 has no effect.
9	RX1THRESHMASK	0-1	Receive channel 1 threshold mask clear bit. Write 1 to disable interrupt; a write of 0 has no effect.
8	RX0THRESHMASK	0-1	Receive channel 0 threshold mask clear bit. Write 1 to disable interrupt; a write of 0 has no effect.
7	RX7MASK	0-1	Receive channel 7 mask clear bit. Write 1 to disable interrupt; a write of 0 has no effect.
6	RX6MASK	0-1	Receive channel 6 mask clear bit. Write 1 to disable interrupt; a write of 0 has no effect.
5	RX5MASK	0-1	Receive channel 5 mask clear bit. Write 1 to disable interrupt; a write of 0 has no effect.
4	RX4MASK	0-1	Receive channel 4 mask clear bit. Write 1 to disable interrupt; a write of 0 has no effect.
3	RX3MASK	0-1	Receive channel 3 mask clear bit. Write 1 to disable interrupt; a write of 0 has no effect.
2	RX2MASK	0-1	Receive channel 2 mask clear bit. Write 1 to disable interrupt; a write of 0 has no effect.
1	RX1MASK	0-1	Receive channel 1 mask clear bit. Write 1 to disable interrupt; a write of 0 has no effect.
0	RX0MASK	0-1	Receive channel 0 mask clear bit. Write 1 to disable interrupt; a write of 0 has no effect.

32.5.17 MAC Interrupt Status (Unmasked) Register (MACINTSTATRAW)

The MAC interrupt status (unmasked) register (MACINTSTATRAW) is shown in Figure 32-58 and described in Table 32-56.

Figure 32-58. MAC Interrupt Status (Unmasked) Register (MACINTSTATRAW) (offset = B0h)

31	16
Reserved	
R-0	

15	2	1	0
Reserved		HOSTPEND	STATPEND
R-0		R-0	R-0

LEGEND: R = Read only; -*n* = value after reset

Table 32-56. MAC Interrupt Status (Unmasked) Register (MACINTSTATRAW) Field Descriptions

Bit	Field	Value	Description
31-2	Reserved	0	Reserved
1	HOSTPEND	0-1	Host pending interrupt (HOSTPEND); raw interrupt read (before mask).
0	STATPEND	0-1	Statistics pending interrupt (STATPEND); raw interrupt read (before mask).

32.5.18 MAC Interrupt Status (Masked) Register (MACINTSTATMASKED)

The MAC interrupt status (masked) register (MACINTSTATMASKED) is shown in Figure 32-59 and described in Table 32-57.

Figure 32-59. MAC Interrupt Status (Masked) Register (MACINTSTATMASKED) (offset = B4h)

31	16
Reserved	
R-0	

15	2	1	0
Reserved		HOSTPEND	STATPEND
R-0		R-0	R-0

LEGEND: R = Read only; -*n* = value after reset

Table 32-57. MAC Interrupt Status (Masked) Register (MACINTSTATMASKED) Field Descriptions

Bit	Field	Value	Description
31-2	Reserved	0	Reserved
1	HOSTPEND	0-1	Host pending interrupt (HOSTPEND); masked interrupt read.
0	STATPEND	0-1	Statistics pending interrupt (STATPEND); masked interrupt read.

32.5.19 MAC Interrupt Mask Set Register (MACINTMASKSET)

The MAC interrupt mask set register (MACINTMASKSET) is shown in Figure 32-60 and described in Table 32-58.

Figure 32-60. MAC Interrupt Mask Set Register (MACINTMASKSET) (offset = B8h)

31–16
Reserved
R-0

15–2	1	0
Reserved	HOSTMASK	STATMASK
R-0	R/W1S-0	R/W1S-0

LEGEND: R/W = Read/Write; R = Read only; W1S = Write 1 to set (writing a 0 has no effect); -*n* = value after reset

Table 32-58. MAC Interrupt Mask Set Register (MACINTMASKSET) Field Descriptions

Bit	Field	Value	Description
31-2	Reserved	0	Reserved
1	HOSTMASK	0-1	Host error interrupt mask set bit. Write 1 to enable interrupt, a write of 0 has no effect.
0	STATMASK	0-1	Statistics interrupt mask set bit. Write 1 to enable interrupt, a write of 0 has no effect.

32.5.20 MAC Interrupt Mask Clear Register (MACINTMASKCLEAR)

The MAC interrupt mask clear register (MACINTMASKCLEAR) is shown in Figure 32-61 and described in Table 32-59.

Figure 32-61. MAC Interrupt Mask Clear Register (MACINTMASKCLEAR) (offset = BCh)

31–16
Reserved
R-0

15–2	1	0
Reserved	HOSTMASK	STATMASK
R-0	R/W1C-0	R/W1C-0

LEGEND: R/W = Read/Write; R = Read only; W1C = Write 1 to clear (writing a 0 has no effect); -*n* = value after reset

Table 32-59. MAC Interrupt Mask Clear Register (MACINTMASKCLEAR) Field Descriptions

Bit	Field	Value	Description
31-2	Reserved	0	Reserved
1	HOSTMASK	0-1	Host error interrupt mask clear bit. Write 1 to disable interrupt, a write of 0 has no effect.
0	STATMASK	0-1	Statistics interrupt mask clear bit. Write 1 to disable interrupt, a write of 0 has no effect.

32.5.21 Receive Multicast/Broadcast/Promiscuous Channel Enable Register (RXMBPENABLE)

The receive multicast/broadcast/promiscuous channel enable register (RXMBPENABLE) is shown in Figure 32-62 and described in Table 32-60.

Figure 32-62. Receive Multicast/Broadcast/Promiscuous Channel Enable Register (RXMBPENABLE) (offset = 100h)

<table>
<tr><td>31</td><td>30</td><td>29</td><td>28</td><td>27</td><td></td><td>25</td><td>24</td></tr>
<tr><td>Reserved</td><td>RXPASSCRC</td><td>RXQOSEN</td><td>RXNOCHAIN</td><td colspan="3">Reserved</td><td>RXCMFEN</td></tr>
<tr><td>R-0</td><td>R/W-0</td><td>R/W-0</td><td>R/W-0</td><td colspan="3">R-0</td><td>R/W-0</td></tr>
<tr><td>23</td><td>22</td><td>21</td><td>20</td><td>19</td><td>18</td><td></td><td>16</td></tr>
<tr><td>RXCSFEN</td><td>RXCEFEN</td><td>RXCAFEN</td><td colspan="2">Reserved</td><td colspan="3">RXPROMCH</td></tr>
<tr><td>R/W-0</td><td>R/W-0</td><td>R/W-0</td><td colspan="2">R-0</td><td colspan="3">R/W-0</td></tr>
<tr><td>15</td><td>14</td><td>13</td><td>12</td><td>11</td><td>10</td><td></td><td>8</td></tr>
<tr><td colspan="2">Reserved</td><td>RXBROADEN</td><td colspan="2">Reserved</td><td colspan="3">RXBROADCH</td></tr>
<tr><td colspan="2">R-0</td><td>R/W-0</td><td colspan="2">R-0</td><td colspan="3">R/W-0</td></tr>
<tr><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td></td><td>0</td></tr>
<tr><td colspan="2">Reserved</td><td>RXMULTEN</td><td colspan="2">Reserved</td><td colspan="3">RXMULTCH</td></tr>
<tr><td colspan="2">R-0</td><td>R/W-0</td><td colspan="2">R-0</td><td colspan="3">R/W-0</td></tr>
</table>

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 32-60. Receive Multicast/Broadcast/Promiscuous Channel Enable Register (RXMBPENABLE) Field Descriptions

Bit	Field	Value	Description
31	Reserved	0	Reserved
30	RXPASSCRC		Pass receive CRC enable bit.
		0	Received CRC is discarded for all channels and is not included in the buffer descriptor packet length field.
		1	Received CRC is transferred to memory for all channels and is included in the buffer descriptor packet length.
29	RXQOSEN		Receive quality of service enable bit.
		0	Receive QOS is disabled.
		1	Receive QOS is enabled.
28	RXNOCHAIN		Receive no buffer chaining bit.
		0	Received frames can span multiple buffers.
		1	The Receive DMA controller transfers each frame into a single buffer, regardless of the frame or buffer size. All remaining frame data after the first buffer is discarded. The buffer descriptor buffer length field will contain the entire frame byte count (up to 65535 bytes).
27-25	Reserved	0	Reserved
24	RXCMFEN		Receive copy MAC control frames enable bit. Enables MAC control frames to be transferred to memory. MAC control frames are normally acted upon (if enabled), but not copied to memory. MAC control frames that are pause frames will be acted upon if enabled in MACCONTROL, regardless of the value of RXCMFEN. Frames transferred to memory due to RXCMFEN will have the CONTROL bit set in their EOP buffer descriptor.
		0	MAC control frames are filtered (but acted upon if enabled).
		1	MAC control frames are transferred to memory.
23	RXCSFEN		Receive copy short frames enable bit. Enables frames or fragments shorter than 64 bytes to be copied to memory. Frames transferred to memory due to RXCSFEN will have the FRAGMENT or UNDERSIZE bit set in their EOP buffer descriptor. Fragments are short frames that contain CRC / align / code errors and undersized are short frames without errors.
		0	Short frames are filtered.
		1	Short frames are transferred to memory.

Table 32-60. Receive Multicast/Broadcast/Promiscuous Channel Enable Register (RXMBPENABLE) Field Descriptions (continued)

Bit	Field	Value	Description
22	RXCEFEN		Receive copy error frames enable bit. Enables frames containing errors to be transferred to memory. The appropriate error bit will be set in the frame EOP buffer descriptor.
		0	Frames containing errors are filtered.
		1	Frames containing errors are transferred to memory.
21	RXCAFEN		Receive copy all frames enable bit. Enables frames that do not address match (includes multicast frames that do not hash match) to be transferred to the promiscuous channel selected by RXPROMCH bits. Such frames will be marked with the NOMATCH bit in their EOP buffer descriptor.
		0	Frames that do not address match are filtered.
		1	Frames that do not address match are transferred to the promiscuous channel selected by RXPROMCH bits.
20-19	Reserved	0	Reserved
18-16	RXPROMCH		Receive promiscuous channel select.
		0	Select channel 0 to receive promiscuous frames.
		1h	Select channel 1 to receive promiscuous frames.
		2h	Select channel 2 to receive promiscuous frames.
		3h	Select channel 3 to receive promiscuous frames.
		4h	Select channel 4 to receive promiscuous frames.
		5h	Select channel 5 to receive promiscuous frames.
		6h	Select channel 6 to receive promiscuous frames.
		7h	Select channel 7 to receive promiscuous frames.
15-14	Reserved	0	Reserved
13	RXBROADEN		Receive broadcast enable. Enable received broadcast frames to be copied to the channel selected by RXBROADCH bits.
		0	Broadcast frames are filtered.
		1	Broadcast frames are copied to the channel selected by RXBROADCH bits.
12-11	Reserved	0	Reserved
10-8	RXBROADCH		Receive broadcast channel select.
		0	Select channel 0 to receive broadcast frames.
		1h	Select channel 1 to receive broadcast frames.
		2h	Select channel 2 to receive broadcast frames.
		3h	Select channel 3 to receive broadcast frames.
		4h	Select channel 4 to receive broadcast frames.
		5h	Select channel 5 to receive broadcast frames.
		6h	Select channel 6 to receive broadcast frames.
		7h	Select channel 7 to receive broadcast frames.
7-6	Reserved	0	Reserved
5	RXMULTEN		RX multicast enable. Enable received hash matching multicast frames to be copied to the channel selected by RXMULTCH bits.
		0	Multicast frames are filtered.
		1	Multicast frames are copied to the channel selected by RXMULTCH bits.
4-3	Reserved	0	Reserved

Table 32-60. Receive Multicast/Broadcast/Promiscuous Channel Enable Register (RXMBPENABLE) Field Descriptions (continued)

Bit	Field	Value	Description
2-0	RXMULTCH		Receive multicast channel select.
		0	Select channel 0 to receive multicast frames.
		1h	Select channel 1 to receive multicast frames.
		2h	Select channel 2 to receive multicast frames.
		3h	Select channel 3 to receive multicast frames.
		4h	Select channel 4 to receive multicast frames.
		5h	Select channel 5 to receive multicast frames.
		6h	Select channel 6 to receive multicast frames.
		7h	Select channel 7 to receive multicast frames.

32.5.22 Receive Unicast Enable Set Register (RXUNICASTSET)

The receive unicast enable set register (RXUNICASTSET) is shown in Figure 32-63 and described in Table 32-61.

Figure 32-63. Receive Unicast Enable Set Register (RXUNICASTSET) (offset = 104h)

31	16
Reserved	
R-0	

15	8
Reserved	
R-0	

7	6	5	4	3	2	1	0
RXCH7EN	RXCH6EN	RXCH5EN	RXCH4EN	RXCH3EN	RXCH2EN	RXCH1EN	RXCH0EN
R/W1S-0	R/W1S-0	R/W1S-0	R/W1S-0	R/W1S-0	R/W1S-0	R/W1S-0	R/W1S-0

LEGEND: R/W = Read/Write; R = Read only; W1S = Write 1 to set (writing a 0 has no effect); -*n* = value after reset

Table 32-61. Receive Unicast Enable Set Register (RXUNICASTSET) Field Descriptions

Bit	Field	Value	Description
31-8	Reserved	0	Reserved
7	RXCH7EN	0-1	Receive channel 7 unicast enable set bit. Write 1 to set the enable, a write of 0 has no effect. May be read.
6	RXCH6EN	0-1	Receive channel 6 unicast enable set bit. Write 1 to set the enable, a write of 0 has no effect. May be read.
5	RXCH5EN	0-1	Receive channel 5 unicast enable set bit. Write 1 to set the enable, a write of 0 has no effect. May be read.
4	RXCH4EN	0-1	Receive channel 4 unicast enable set bit. Write 1 to set the enable, a write of 0 has no effect. May be read.
3	RXCH3EN	0-1	Receive channel 3 unicast enable set bit. Write 1 to set the enable, a write of 0 has no effect. May be read.
2	RXCH2EN	0-1	Receive channel 2 unicast enable set bit. Write 1 to set the enable, a write of 0 has no effect. May be read.
1	RXCH1EN	0-1	Receive channel 1 unicast enable set bit. Write 1 to set the enable, a write of 0 has no effect. May be read.
0	RXCH0EN	0-1	Receive channel 0 unicast enable set bit. Write 1 to set the enable, a write of 0 has no effect. May be read.

32.5.23 Receive Unicast Clear Register (RXUNICASTCLEAR)

The receive unicast clear register (RXUNICASTCLEAR) is shown in Figure 32-64 and described in Table 32-62.

Figure 32-64. Receive Unicast Clear Register (RXUNICASTCLEAR) (offset = 108h)

31	16
Reserved	
R-0	

15	8
Reserved	
R-0	

7	6	5	4	3	2	1	0
RXCH7EN	RXCH6EN	RXCH5EN	RXCH4EN	RXCH3EN	RXCH2EN	RXCH1EN	RXCH0EN
R/W1C-0	R/W1C-0	R/W1C-0	R/W1C-0	R/W1C-0	R/W1C-0	R/W1C-0	R/W1C-0

LEGEND: R/W = Read/Write; R = Read only; W1C = Write 1 to clear (writing a 0 has no effect); -*n* = value after reset

Table 32-62. Receive Unicast Clear Register (RXUNICASTCLEAR) Field Descriptions

Bit	Field	Value	Description
31-8	Reserved	0	Reserved
7	RXCH7EN	0-1	Receive channel 7 unicast enable clear bit. Write 1 to clear the enable, a write of 0 has no effect.
6	RXCH6EN	0-1	Receive channel 6 unicast enable clear bit. Write 1 to clear the enable, a write of 0 has no effect.
5	RXCH5EN	0-1	Receive channel 5 unicast enable clear bit. Write 1 to clear the enable, a write of 0 has no effect.
4	RXCH4EN	0-1	Receive channel 4 unicast enable clear bit. Write 1 to clear the enable, a write of 0 has no effect.
3	RXCH3EN	0-1	Receive channel 3 unicast enable clear bit. Write 1 to clear the enable, a write of 0 has no effect.
2	RXCH2EN	0-1	Receive channel 2 unicast enable clear bit. Write 1 to clear the enable, a write of 0 has no effect.
1	RXCH1EN	0-1	Receive channel 1 unicast enable clear bit. Write 1 to clear the enable, a write of 0 has no effect.
0	RXCH0EN	0-1	Receive channel 0 unicast enable clear bit. Write 1 to clear the enable, a write of 0 has no effect.

32.5.24 Receive Maximum Length Register (RXMAXLEN)

The receive maximum length register (RXMAXLEN) is shown in Figure 32-65 and described in Table 32-63.

Figure 32-65. Receive Maximum Length Register (RXMAXLEN) (offset = 10Ch)

31	16
Reserved	
R-0	

15	0
RXMAXLEN	
R/W-5EEh	

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 32-63. Receive Maximum Length Register (RXMAXLEN) Field Descriptions

Bit	Field	Value	Description
31-16	Reserved	0	Reserved
15-0	RXMAXLEN	0-FFFFh	Receive maximum frame length. These bits determine the maximum length of a received frame. The reset value is 5EEh (1518). Frames with byte counts greater than RXMAXLEN are long frames. Long frames with no errors are oversized frames. Long frames with CRC, code, or alignment error are jabber frames.

32.5.25 Receive Buffer Offset Register (RXBUFFEROFFSET)

The receive buffer offset register (RXBUFFEROFFSET) is shown in Figure 32-66 and described in Table 32-64.

Figure 32-66. Receive Buffer Offset Register (RXBUFFEROFFSET) (offset = 110h)

31 - 16
Reserved
R-0

15 - 0
RXBUFFEROFFSET
R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 32-64. Receive Buffer Offset Register (RXBUFFEROFFSET) Field Descriptions

Bit	Field	Value	Description
31-16	Reserved	0	Reserved
15-0	RXBUFFEROFFSET	0-FFFFh	Receive buffer offset value. These bits are written by the EMAC into each frame SOP buffer descriptor Buffer Offset field. The frame data begins after the RXBUFFEROFFSET value of bytes. A value of 0 indicates that there are no unused bytes at the beginning of the data, and that valid data begins on the first byte of the buffer. A value of Fh (15) indicates that the first 15 bytes of the buffer are to be ignored by the EMAC and that valid buffer data starts on byte 16 of the buffer. This value is used for all channels.

32.5.26 Receive Filter Low Priority Frame Threshold Register (RXFILTERLOWTHRESH)

The receive filter low priority frame threshold register (RXFILTERLOWTHRESH) is shown in Figure 32-67 and described in Table 32-65.

Figure 32-67. Receive Filter Low Priority Frame Threshold Register (RXFILTERLOWTHRESH) (offset = 114h)

31 - 16
Reserved
R-0

15 - 8	7 - 0
Reserved	RXFILTERTHRESH
R-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 32-65. Receive Filter Low Priority Frame Threshold Register (RXFILTERLOWTHRESH) Field Descriptions

Bit	Field	Value	Description
31-8	Reserved	0	Reserved
7-0	RXFILTERTHRESH	0-FFh	Receive filter low threshold. These bits contain the free buffer count threshold value for filtering low priority incoming frames. This field should remain 0, if no filtering is desired.

32.5.27 Receive Channel Flow Control Threshold Registers (RX0FLOWTHRESH-RX7FLOWTHRESH)

The receive channel 0-7 flow control threshold register (RX*n*FLOWTHRESH) is shown in Figure 32-68 and described in Table 32-66.

Figure 32-68. Receive Channel *n* Flow Control Threshold Register (RX*n*FLOWTHRESH) (offset = 120h-13Ch)

31	16
Reserved	
R-0	

15	8	7	0
Reserved		RX*n*FLOWTHRESH	
R-0		R/W-0	

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 32-66. Receive Channel *n* Flow Control Threshold Register (RX*n*FLOWTHRESH) Field Descriptions

Bit	Field	Value	Description
31-8	Reserved	0	Reserved
7-0	RX*n*FLOWTHRESH	0-FFh	Receive flow threshold. These bits contain the threshold value for issuing flow control on incoming frames for channel *n* (when enabled).

32.5.28 Receive Channel Free Buffer Count Registers (RX0FREEBUFFER-RX7FREEBUFFER)

The receive channel 0-7 free buffer count register (RX*n*FREEBUFFER) is shown in Figure 32-69 and described in Table 32-67.

Figure 32-69. Receive Channel *n* Free Buffer Count Register (RX*n*FREEBUFFER) (offset = 140h-15Ch)

31	16
Reserved	
R-0	

15	0
RX*n*FREEBUF	
WI-0	

LEGEND: R = Read only; WI = Write to increment; -*n* = value after reset

Table 32-67. Receive Channel *n* Free Buffer Count Register (RX*n*FREEBUFFER) Field Descriptions

Bit	Field	Value	Description
31-16	Reserved	0	Reserved
15-0	RX*n*FREEBUF	0-FFh	Receive free buffer count. These bits contain the count of free buffers available. The RXFILTERTHRESH value is compared with this field to determine if low priority frames should be filtered. The RX*n*FLOWTHRESH value is compared with this field to determine if receive flow control should be issued against incoming packets (if enabled). This is a write-to-increment field. This field rolls over to 0 on overflow. If hardware flow control or QOS is used, the host must initialize this field to the number of available buffers (one register per channel). The EMAC decrements the associated channel register for each received frame by the number of buffers in the received frame. The host must write this field with the number of buffers that have been freed due to host processing.

32.5.29 MAC Control Register (MACCONTROL)

The MAC control register (MACCONTROL) is shown in Figure 32-70 and described in Table 32-68.

Figure 32-70. MAC Control Register (MACCONTROL) (offset = 160h)

31-16
Reserved
R-0

15	14	13	12	11	10	9	8
RMIISPEED	RXOFFLENBLOCK	RXOWNERSHIP	Rsvd	CMDIDLE	TXSHORTGAPEN	TXPTYPE	Reserved
R/W-0	R/W-0	R/W-0	R-0	R/W-0	R/W-0	R/W-0	R-0

7	6	5	4	3	2	1	0
Reserved	TXPACE	GMIIEN	TXFLOWEN	RXBUFFERFLOWEN	Reserved	LOOPBACK	FULLDUPLEX
R-0	R/W-0	R/W-0	R/W-0	R/W-0	R-0	R/W-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 32-68. MAC Control Register (MACCONTROL) Field Descriptions

Bit	Field	Value	Description
31-16	Reserved	0	Reserved
15	RMIISPEED		RMII interface transmit and receive speed select.
		0	Operate RMII interface in 10 Mbps speed mode.
		1	Operate RMII interface in 100 Mbps speed mode.
14	RXOFFLENBLOCK		Receive offset / length word write block.
		0	Do not block the DMA writes to the receive buffer descriptor offset / buffer length word.
		1	Block all EMAC DMA controller writes to the receive buffer descriptor offset / buffer length words during packet processing. When this bit is set, the EMAC will never write the third word to any receive buffer descriptor.
13	RXOWNERSHIP		Receive ownership write bit value.
		0	The EMAC writes the Receive ownership bit to 0 at the end of packet processing.
		1	The EMAC writes the Receive ownership bit to 1 at the end of packet processing. If you do not use the ownership mechanism, you can set this mode to preclude the necessity of software having to set this bit each time the buffer descriptor is used.
12	Reserved	0	Reserved
11	CMDIDLE		Command Idle bit.
		0	Idle is not commanded.
		1	Idle is commanded (read IDLE in the MACSTATUS register).
10	TXSHORTGAPEN		Transmit Short Gap enable.
		0	Transmit with a short IPG is disabled. Normal 96-bit time IPG is inserted between packets.
		1	Transmit with a short IPG is enabled. Shorter 88-bit time IPG is inserted between packets.
9	TXPTYPE		Transmit queue priority type.
		0	The queue uses a round-robin scheme to select the next channel for transmission.
		1	The queue uses a fixed-priority (channel 7 highest priority) scheme to select the next channel for transmission.
8-7	Reserved	0	Reserved
6	TXPACE		Transmit pacing enable bit.
		0	Transmit pacing is disabled.
		1	Transmit pacing is enabled.

Table 32-68. MAC Control Register (MACCONTROL) Field Descriptions (continued)

Bit	Field	Value	Description
5	GMIIEN		GMII enable bit. This bit must be set before the MAC transmits or receives data in any of the supported interface modes. (for instance, MII, RMII). This bit does not select the interface mode but rather holds or releases the MAC TX and RX state machines from reset.
		0	The MAC RX and TX state machines are held in reset.
		1	The MAC RX and TX state machines are released from reset and transmit and receive are enabled.
4	TXFLOWEN		Transmit flow control enable bit. This bit determines if incoming pause frames are acted upon in full-duplex mode. Incoming pause frames are not acted upon in half-duplex mode, regardless of this bit setting. The RXMBPENABLE bits determine whether or not received pause frames are transferred to memory.
		0	Transmit flow control is disabled. Full-duplex mode: incoming pause frames are not acted upon.
		1	Transmit flow control is enabled. Full-duplex mode: incoming pause frames are acted upon.
3	RXBUFFERFLOWEN		Receive buffer flow control enable bit.
		0	Receive flow control is disabled. Half-duplex mode: no flow control generated collisions are sent. Full-duplex mode: no outgoing pause frames are sent.
		1	Receive flow control is enabled. Half-duplex mode: collisions are initiated when receive buffer flow control is triggered. Full-duplex mode: outgoing pause frames are sent when receive flow control is triggered.
2	Reserved	0	Reserved
1	LOOPBACK		Loopback mode. The loopback mode forces internal full-duplex mode regardless of the FULLDUPLEX bit. The loopback bit should be changed only when GMIIEN bit is deasserted.
		0	Loopback mode is disabled.
		1	Loopback mode is enabled.
0	FULLDUPLEX		Full-duplex mode.
		0	Half-duplex mode is enabled.
		1	Full-duplex mode is enabled.

32.5.30 MAC Status Register (MACSTATUS)

The MAC status register (MACSTATUS) is shown in Figure 32-71 and described in Table 32-69.

Figure 32-71. MAC Status Register (MACSTATUS) (offset = 164h)

31	30 24	23 20	19	18 16
IDLE	Reserved	TXERRCODE	Rsvd	TXERRCH
R-0	R-0	R-0	R-0	R-0

15 12	11	10 8
RXERRCODE	Reserved	RXERRCH
R-0	R-0	R-0

7 3	2	1	0
Reserved	RXQOSACT	RXFLOWACT	TXFLOWACT
R-0	R-0	R-0	R-0

LEGEND: R = Read only; -*n* = value after reset

Table 32-69. MAC Status Register (MACSTATUS) Field Descriptions

Bit	Field	Value	Description
31	IDLE		EMAC idle bit. This bit is cleared to 0 at reset; one clock after reset, it goes to 1.
		0	The EMAC is not idle.
		1	The EMAC is in the idle state.
30-24	Reserved	0	Reserved
23-20	TXERRCODE		Transmit host error code. These bits indicate that EMAC detected transmit DMA related host errors. The host should read this field after a host error interrupt (HOSTPEND) to determine the error. Host error interrupts require hardware reset in order to recover. A 0 packet length is an error, but it is not detected.
		0	No error.
		1h	SOP error; the buffer is the first buffer in a packet, but the SOP bit is not set in software.
		2h	Ownership bit not set in SOP buffer.
		3h	Zero next buffer descriptor pointer without EOP.
		4h	Zero buffer pointer.
		5h	Zero buffer length.
		6h	Packet length error (sum of buffers is less than packet length).
		7h-Fh	Reserved
19	Reserved	0	Reserved
18-16	TXERRCH		Transmit host error channel. These bits indicate which transmit channel the host error occurred on. This field is cleared to 0 on a host read.
		0	The host error occurred on transmit channel 0.
		1h	The host error occurred on transmit channel 1.
		2h	The host error occurred on transmit channel 2.
		3h	The host error occurred on transmit channel 3.
		4h	The host error occurred on transmit channel 4.
		5h	The host error occurred on transmit channel 5.
		6h	The host error occurred on transmit channel 6.
		7h	The host error occurred on transmit channel 7.

Table 32-69. MAC Status Register (MACSTATUS) Field Descriptions (continued)

Bit	Field	Value	Description
15-12	RXERRCODE		Receive host error code. These bits indicate that EMAC detected receive DMA related host errors. The host should read this field after a host error interrupt (HOSTPEND) to determine the error. Host error interrupts require hardware reset in order to recover.
		0	No error.
		1h	Reserved
		2h	Ownership bit not set in SOP buffer.
		3h	Reserved
		4h	Zero buffer pointer.
		5h-Fh	Reserved
11	Reserved	0	Reserved
10-8	RXERRCH		Receive host error channel. These bits indicate which receive channel the host error occurred on. This field is cleared to 0 on a host read.
		0	The host error occurred on receive channel 0.
		1h	The host error occurred on receive channel 1.
		2h	The host error occurred on receive channel 2.
		3h	The host error occurred on receive channel 3.
		4h	The host error occurred on receive channel 4.
		5h	The host error occurred on receive channel 5.
		6h	The host error occurred on receive channel 6.
		7h	The host error occurred on receive channel 7.
7-3	Reserved	0	Reserved
2	RXQOSACT		Receive Quality of Service (QOS) active bit. When asserted, indicates that receive quality of service is enabled and that at least one channel freebuffer count (RX*n*FREEBUFFER) is less than or equal to the RXFILTERLOWTHRESH value.
		0	Receive quality of service is disabled.
		1	Receive quality of service is enabled.
1	RXFLOWACT		Receive flow control active bit. When asserted, at least one channel freebuffer count (RX*n*FREEBUFFER) is less than or equal to the channel's corresponding RX*n*FILTERTHRESH value.
		0	Receive flow control is inactive.
		1	Receive flow control is active.
0	TXFLOWACT		Transmit flow control active bit. When asserted, this bit indicates that the pause time period is being observed for a received pause frame. No new transmissions will begin while this bit is asserted, except for the transmission of pause frames. Any transmission in progress when this bit is asserted will complete.
		0	Transmit flow control is inactive.
		1	Transmit flow control is active.

32.5.31 Emulation Control Register (EMCONTROL)

The emulation control register (EMCONTROL) is shown in Figure 32-72 and described in Table 32-70.

Figure 32-72. Emulation Control Register (EMCONTROL) (offset = 168h)

31	16
Reserved	
R-0	

15	2	1	0
Reserved		SOFT	FREE
R-0		R/W-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 32-70. Emulation Control Register (EMCONTROL) Field Descriptions

Bit	Field	Value	Description
31-2	Reserved	0	Reserved
1	SOFT		Emulation soft bit. This bit is used in conjunction with FREE bit to determine the emulation suspend mode. This bit has no effect if FREE = 1.
		0	Soft mode is disabled. EMAC stops immediately during emulation halt.
		1	Soft mode is enabled. During emulation halt, EMAC stops after completion of current operation.
0	FREE		Emulation free bit. This bit is used in conjunction with SOFT bit to determine the emulation suspend mode.
		0	Free-running mode is disabled. During emulation halt, SOFT bit determines operation of EMAC.
		1	Free-running mode is enabled. During emulation halt, EMAC continues to operate.

32.5.32 FIFO Control Register (FIFOCONTROL)

The FIFO control register (FIFOCONTROL) is shown in Figure 32-73 and described in Table 32-71.

Figure 32-73. FIFO Control Register (FIFOCONTROL) (offset = 16Ch)

31	16
Reserved	
R-0	

15	2	1	0
Reserved		TXCELLTHRESH	
R-0		R/W-2h	

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 32-71. FIFO Control Register (FIFOCONTROL) Field Descriptions

Bit	Field	Value	Description
31-2	Reserved	0	Reserved
1-0	TXCELLTHRESH		Transmit FIFO cell threshold. Indicates the number of 64-byte packet cells required to be in the transmit FIFO before the packet transfer is initiated. Packets with fewer cells will be initiated when the complete packet is contained in the FIFO. The default value is 2, but 3 is also valid. 0 and 1 are not valid values.
		0-1h	Not a valid value.
		2h	Two 64-byte packet cells required to be in the transmit FIFO.
		3h	Three 64-byte packet cells required to be in the transmit FIFO.

32.5.33 MAC Configuration Register (MACCONFIG)

The MAC configuration register (MACCONFIG) is shown in Figure 32-74 and described in Table 32-72.

Figure 32-74. MAC Configuration Register (MACCONFIG) (offset = 170h)

31 — 24	23 — 16
TXCELLDEPTH	RXCELLDEPTH
R-3h	R-3h

15 — 8	7 — 0
ADDRESSTYPE	MACCFIG
R-2h	R-2h

LEGEND: R = Read only; -*n* = value after reset

Table 32-72. MAC Configuration Register (MACCONFIG) Field Descriptions

Bit	Field	Value	Description
31-24	TXCELLDEPTH	3h	Transmit cell depth. These bits indicate the number of cells in the transmit FIFO.
23-16	RXCELLDEPTH	3h	Receive cell depth. These bits indicate the number of cells in the receive FIFO.
15-8	ADDRESSTYPE	2h	Address type.
7-0	MACCFIG	2h	MAC configuration value.

32.5.34 Soft Reset Register (SOFTRESET)

The soft reset register (SOFTRESET) is shown in Figure 32-75 and described in Table 32-73.

Figure 32-75. Soft Reset Register (SOFTRESET) (offset = 174h)

31 — 16
Reserved
R-0

15 — 1	0
Reserved	SOFTRESET
R-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 32-73. Soft Reset Register (SOFTRESET) Field Descriptions

Bit	Field	Value	Description
31-1	Reserved	0	Reserved
0	SOFTRESET		Software reset. Writing a 1 to this bit causes the EMAC logic to be reset. Software reset occurs when the receive and transmit DMA controllers are in an idle state to avoid locking up the Configuration bus. After writing a 1 to this bit, it may be polled to determine if the reset has occurred. If a 1 is read, the reset has not yet occurred. If a 0 is read, then a reset has occurred.
		0	A software reset has not occurred.
		1	A software reset has occurred.

32.5.35 MAC Source Address Low Bytes Register (MACSRCADDRLO)

The MAC source address low bytes register (MACSRCADDRLO) is shown in Figure 32-76 and described in Table 32-74.

Figure 32-76. MAC Source Address Low Bytes Register (MACSRCADDRLO) (offset = 1D0h)

31	16
Reserved	
R-0	

15 8	7 0
MACSRCADDR0	MACSRCADDR1
R/W-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 32-74. MAC Source Address Low Bytes Register (MACSRCADDRLO) Field Descriptions

Bit	Field	Value	Description
31-16	Reserved	0	Reserved
15-8	MACSRCADDR0	0-FFh	MAC source address lower 8-0 bits (byte 0).
7-0	MACSRCADDR1	0-FFh	MAC source address bits 15-8 (byte 1).

32.5.36 MAC Source Address High Bytes Register (MACSRCADDRHI)

The MAC source address high bytes register (MACSRCADDRHI) is shown in Figure 32-77 and described in Table 32-75.

Figure 32-77. MAC Source Address High Bytes Register (MACSRCADDRHI) (offset = 1D4h)

31 24	23 16
MACSRCADDR2	MACSRCADDR3
R/W-0	R/W-0

15 8	7 0
MACSRCADDR4	MACSRCADDR5
R/W-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 32-75. MAC Source Address High Bytes Register (MACSRCADDRHI) Field Descriptions

Bit	Field	Value	Description
31-24	MACSRCADDR2	0-FFh	MAC source address bits 23-16 (byte 2).
23-16	MACSRCADDR3	0-FFh	MAC source address bits 31-24 (byte 3).
15-8	MACSRCADDR4	0-FFh	MAC source address bits 39-32 (byte 4).
7-0	MACSRCADDR5	0-FFh	MAC source address bits 47-40 (byte 5).

32.5.37 MAC Hash Address Register 1 (MACHASH1)

The MAC hash registers allow group addressed frames to be accepted on the basis of a hash function of the address. The hash function creates a 6-bit data value (Hash_fun) from the 48-bit destination address (DA) as follows:

```
Hash_fun(0)=DA(0) XOR DA(6) XOR DA(12) XOR DA(18) XOR DA(24) XOR DA(30) XOR DA(36) XOR DA(42);

Hash_fun(1)=DA(1) XOR DA(7) XOR DA(13) XOR DA(19) XOR DA(25) XOR DA(31) XOR DA(37) XOR DA(43);

Hash_fun(2)=DA(2) XOR DA(8) XOR DA(14) XOR DA(20) XOR DA(26) XOR DA(32) XOR DA(38) XOR DA(44);

Hash_fun(3)=DA(3) XOR DA(9) XOR DA(15) XOR DA(21) XOR DA(27) XOR DA(33) XOR DA(39) XOR DA(45);

Hash_fun(4)=DA(4) XOR DA(10) XOR DA(16) XOR DA(22) XOR DA(28) XOR DA(34) XOR DA(40) XOR DA(46);

Hash_fun(5)=DA(5) XOR DA(11) XOR DA(17) XOR DA(23) XOR DA(29) XOR DA(35) XOR DA(41) XOR DA(47);
```

This function is used as an offset into a 64-bit hash table stored in MACHASH1 and MACHASH2 that indicates whether a particular address should be accepted or not.

The MAC hash address register 1 (MACHASH1) is shown in Figure 32-78 and described in Table 32-76.

Figure 32-78. MAC Hash Address Register 1 (MACHASH1) (offset = 1D8h)

31 … 0
MACHASH1
R/W-0

LEGEND: R/W = Read/Write; -*n* = value after reset

Table 32-76. MAC Hash Address Register 1 (MACHASH1) Field Descriptions

Bit	Field	Description
31-0	MACHASH1	Least-significant 32 bits of the hash table corresponding to hash values 0 to 31. If a hash table bit is set, then a group address that hashes to that bit index is accepted.

32.5.38 MAC Hash Address Register 2 (MACHASH2)

The MAC hash address register 2 (MACHASH2) is shown in Figure 32-79 and described in Table 32-77.

Figure 32-79. MAC Hash Address Register 2 (MACHASH2) (offset = 1DCh)

31 … 0
MACHASH2
R/W-0

LEGEND: R/W = Read/Write; -*n* = value after reset

Table 32-77. MAC Hash Address Register 2 (MACHASH2) Field Descriptions

Bit	Field	Description
31-0	MACHASH2	Most-significant 32 bits of the hash table corresponding to hash values 32 to 63. If a hash table bit is set, then a group address that hashes to that bit index is accepted.

32.5.39 Back Off Test Register (BOFFTEST)

The back off test register (BOFFTEST) is shown in Figure 32-80 and described in Table 32-78.

Figure 32-80. Back Off Random Number Generator Test Register (BOFFTEST) (offset = 1E0h)

31 26	25 16
Reserved	RNDNUM
R-0	R-0

15 12	11 10	9 0
COLLCOUNT	Reserved	TXBACKOFF
R-0	R-0	R-0

LEGEND: R = Read only; -*n* = value after reset

Table 32-78. Back Off Test Register (BOFFTEST) Field Descriptions

Bit	Field	Value	Description
31-26	Reserved	0	Reserved
25-16	RNDNUM	0-3FFh	Backoff random number generator. This field allows the Backoff Random Number Generator to be read. Reading this field returns the generator's current value. The value is reset to 0 and begins counting on the clock after the deassertion of reset.
15-12	COLLCOUNT	0-Fh	Collision count. These bits indicate the number of collisions the current frame has experienced.
11-10	Reserved	0	Reserved
9-0	TXBACKOFF	0-3FFh	Backoff count. This field allows the current value of the backoff counter to be observed for test purposes. This field is loaded automatically according to the backoff algorithm, and is decremented by 1 for each slot time after the collision.

32.5.40 Transmit Pacing Algorithm Test Register (TPACETEST)

The transmit pacing algorithm test register (TPACETEST) is shown in Figure 32-81 and described in Table 32-79.

Figure 32-81. Transmit Pacing Algorithm Test Register (TPACETEST) (offset = 1E4h)

31 16
Reserved
R-0

15 5	4 0
Reserved	PACEVAL
R-0	R-0

LEGEND: R = Read only; -*n* = value after reset

Table 32-79. Transmit Pacing Algorithm Test Register (TPACETEST) Field Descriptions

Bit	Field	Value	Description
31-5	Reserved	0	Reserved
4-0	PACEVAL	0-1Fh	Pacing register current value. A nonzero value in this field indicates that transmit pacing is active. A transmit frame collision or deferral causes PACEVAL to be loaded with 1Fh (31); good frame transmissions (with no collisions or deferrals) cause PACEVAL to be decremented down to 0. When PACEVAL is nonzero, the transmitter delays four Inter Packet Gaps between new frame transmissions after each successfully transmitted frame that had no deferrals or collisions. If a transmit frame is deferred or suffers a collision, the IPG time is not stretched to four times the normal value. Transmit pacing helps reduce capture effects, which improves overall network bandwidth.

32.5.41 Receive Pause Timer Register (RXPAUSE)

The receive pause timer register (RXPAUSE) is shown in Figure 32-82 and described in Table 32-80.

Figure 32-82. Receive Pause Timer Register (RXPAUSE) (offset = 1E8h)

31	16
Reserved	
R-0	

15	0
PAUSETIMER	
R-0	

LEGEND: R = Read only; -*n* = value after reset

Table 32-80. Receive Pause Timer Register (RXPAUSE) Field Descriptions

Bit	Field	Value	Description
31-16	Reserved	0	Reserved
15-0	PAUSETIMER	0-FFh	Receive pause timer value. These bits allow the contents of the receive pause timer to be observed. The receive pause timer is loaded with FF00h when the EMAC sends an outgoing pause frame (with pause time of FFFFh). The receive pause timer is decremented at slot time intervals. If the receive pause timer decrements to 0, then another outgoing pause frame is sent and the load/decrement process is repeated.

32.5.42 Transmit Pause Timer Register (TXPAUSE)

The transmit pause timer register (TXPAUSE) is shown in Figure 32-83 and described in Table 32-81.

Figure 32-83. Transmit Pause Timer Register (TXPAUSE) (offset = 1ECh)

31	16
Reserved	
R-0	

15	0
PAUSETIMER	
R-0	

LEGEND: R = Read only; -*n* = value after reset

Table 32-81. Transmit Pause Timer Register (TXPAUSE) Field Descriptions

Bit	Field	Value	Description
31-16	Reserved	0	Reserved
15-0	PAUSETIMER	0-FFh	Transmit pause timer value. These bits allow the contents of the transmit pause timer to be observed. The transmit pause timer is loaded by a received (incoming) pause frame, and then decremented at slot time intervals down to 0, at which time EMAC transmit frames are again enabled.

32.5.43 MAC Address Low Bytes Register (MACADDRLO)

The MAC address low bytes register used in receive address matching (MACADDRLO), is shown in Figure 32-84 and described in Table 32-82.

Figure 32-84. MAC Address Low Bytes Register (MACADDRLO) (offset = 500h)

31 ... 21	20	19	18 ... 16
Reserved	VALID	MATCHFILT	CHANNEL
R-0	R/W-x	R/W-x	R/W-x

15 ... 8	7 ... 0
MACADDR0	MACADDR1
R/W-x	R/W-x

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset; -x = value is indeterminate after reset

Table 32-82. MAC Address Low Bytes Register (MACADDRLO) Field Descriptions

Bit	Field	Value	Description
31-21	Reserved	0	Reserved
20	VALID		Address valid bit. This bit should be cleared to 0 for unused address channels.
		0	Address is not valid and will not be used for matching or filtering incoming packets.
		1	Address is valid and will be used for matching or filtering incoming packets.
19	MATCHFILT		Match or filter bit.
		0	The address will be used (if the VALID bit is set) to filter incoming packet addresses.
		1	The address will be used (if the VALID bit is set) to match incoming packet addresses.
18-16	CHANNEL	0-7h	Channel select. Determines which receive channel a valid address match will be transferred to. The channel is a don't care if MATCHFILT is cleared to 0.
15-8	MACADDR0	0-FFh	MAC address lower 8-0 bits (byte 0).
7-0	MACADDR1	0-FFh	MAC address bits 15-8 (byte 1).

32.5.44 MAC Address High Bytes Register (MACADDRHI)

The MAC address high bytes register used in receive address matching (MACADDRHI) is shown in Figure 32-85 and described in Table 32-83.

Figure 32-85. MAC Address High Bytes Register (MACADDRHI) (offset = 504h)

31 ... 24	23 ... 16
MACADDR2	MACADDR3
R/W-x	R/W-x

15 ... 8	7 ... 0
MACADDR4	MACADDR5
R/W-x	R/W-x

LEGEND: R/W = Read/Write; -*n* = value after reset; -x = value is indeterminate after reset

Table 32-83. MAC Address High Bytes Register (MACADDRHI) Field Descriptions

Bit	Field	Value	Description
31-24	MACADDR2	0-FFh	MAC source address bits 23-16 (byte 2).
23-16	MACADDR3	0-FFh	MAC source address bits 31-24 (byte 3).
15-8	MACADDR4	0-FFh	MAC source address bits 39-32 (byte 4).
7-0	MACADDR5	0-FFh	MAC source address bits 47-40 (byte 5). Bit 40 is the group bit. It is forced to 0 and read as 0. Therefore, only unicast addresses are represented in the address table.

32.5.45 MAC Index Register (MACINDEX)

The MAC index register (MACINDEX) is shown in Figure 32-86 and described in Table 32-84.

Figure 32-86. MAC Index Register (MACINDEX) (offset = 508h)

31 ... 16
Reserved
R-0

15 ... 3	2 ... 0
Reserved	MACINDEX
R-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 32-84. MAC Index Register (MACINDEX) Field Descriptions

Bit	Field	Value	Description
31-3	Reserved	0	Reserved
2-0	MACINDEX	0-7h	MAC address index. All eight addresses share the upper 40 bits. Only the lower byte is unique for each address. An address is written by first writing the address number (channel) into the MACINDEX register. The upper 32 bits of the address are then written to the MACADDRHI register, which is followed by writing the lower 16 bits of the address to the MACADDRLO register. Since all eight addresses share the upper 40 bits of the address, the MACADDRHI register only needs to be written the first time.

32.5.46 Transmit Channel DMA Head Descriptor Pointer Registers (TX0HDP-TX7HDP)

The transmit channel 0-7 DMA head descriptor pointer register (TX*n*HDP) is shown in Figure 32-87 and described in Table 32-85.

Figure 32-87. Transmit Channel *n* DMA Head Descriptor Pointer Register (TX*n*HDP) (offset = 600h-61Ch)

31	0
TX*n*HDP	
R/W-x	

LEGEND: R/W = Read/Write; -*n* = value after reset; -x = value is indeterminate after reset

Table 32-85. Transmit Channel *n* DMA Head Descriptor Pointer Register (TX*n*HDP) Field Descriptions

Bit	Field	Description
31-0	TX*n*HDP	Transmit channel *n* DMA Head Descriptor pointer. Writing a transmit DMA buffer descriptor address to a head pointer location initiates transmit DMA operations in the queue for the selected channel. Writing to these locations when they are nonzero is an error (except at reset). Host software must initialize these locations to 0 on reset.

32.5.47 Receive Channel DMA Head Descriptor Pointer Registers (RX0HDP-RX7HDP)

The receive channel 0-7 DMA head descriptor pointer register (RX*n*HDP) is shown in Figure 32-88 and described in Table 32-86.

Figure 32-88. Receive Channel *n* DMA Head Descriptor Pointer Register (RX*n*HDP) (offset = 620h-63Ch)

31	0
RX*n*HDP	
R/W-x	

LEGEND: R/W = Read/Write; -*n* = value after reset; -x = value is indeterminate after reset

Table 32-86. Receive Channel *n* DMA Head Descriptor Pointer Register (RX*n*HDP) Field Descriptions

Bit	Field	Description
31-0	RX*n*HDP	Receive channel *n* DMA Head Descriptor pointer. Writing a receive DMA buffer descriptor address to this location allows receive DMA operations in the selected channel when a channel frame is received. Writing to these locations when they are nonzero is an error (except at reset). Host software must initialize these locations to 0 on reset.

32.5.48 Transmit Channel Completion Pointer Registers (TX0CP-TX7CP)

The transmit channel 0-7 completion pointer register (TX*n*CP) is shown in Figure 32-89 and described in Table 32-87.

Figure 32-89. Transmit Channel *n* Completion Pointer Register (TX*n*CP) (offset = 640h-65Ch)

31 ... 0
TX*n*CP
R/W-x

LEGEND: R/W = Read/Write; -*n* = value after reset; -x = value is indeterminate after reset

Table 32-87. Transmit Channel *n* Completion Pointer Register (TX*n*CP) Field Descriptions

Bit	Field	Description
31-0	TX*n*CP	Transmit channel *n* completion pointer register is written by the host with the buffer descriptor address for the last buffer processed by the host during interrupt processing. The EMAC uses the value written to determine if the interrupt should be deasserted.

32.5.49 Receive Channel Completion Pointer Registers (RX0CP-RX7CP)

The receive channel 0-7 completion pointer register (RX*n*CP) is shown in Figure 32-90 and described in Table 32-88.

Figure 32-90. Receive Channel *n* Completion Pointer Register (RX*n*CP) (offset = 660h-67Ch)

31 ... 0
RX*n*CP
R/W-x

LEGEND: R/W = Read/Write; -*n* = value after reset; -x = value is indeterminate after reset

Table 32-88. Receive Channel *n* Completion Pointer Register (RX*n*CP) Field Descriptions

Bit	Field	Description
31-0	RX*n*CP	Receive channel *n* completion pointer register is written by the host with the buffer descriptor address for the last buffer processed by the host during interrupt processing. The EMAC uses the value written to determine if the interrupt should be deasserted.

32.5.50 Network Statistics Registers

The EMAC has a set of statistics that record events associated with frame traffic. The statistics values are cleared to 0, 38 clocks after the rising edge of reset. When the GMIIEN bit in the MACCONTROL register is set, all statistics registers (see Figure 32-91) are write-to-decrement. The value written is subtracted from the register value with the result stored in the register. If a value greater than the statistics value is written, then zero is written to the register (writing FFFF FFFFh clears a statistics location). When the GMIIEN bit is cleared, all statistics registers are read/write (normal write direct, so writing 0000 0000h clears a statistics location). All write accesses must be 32-bit accesses.

The statistics interrupt (STATPEND) is issued, if enabled, when any statistics value is greater than or equal to 8000 0000h. The statistics interrupt is removed by writing to decrement any statistics value greater than 8000 0000h. The statistics are mapped into internal memory space and are 32-bits wide. All statistics rollover from FFFF FFFFh to 0000 0000h.

Figure 32-91. Statistics Register

31	0
COUNT	
R/WD-0	

LEGEND: R/W = Read/Write; WD = Write to decrement; -*n* = value after reset

32.5.50.1 Good Receive Frames Register (RXGOODFRAMES) (offset = 200h)

The total number of good frames received on the EMAC. A good frame is defined as having all of the following:

- Any data or MAC control frame that matched a unicast, broadcast, or multicast address, or matched due to promiscuous mode
- Was of length 64 to RXMAXLEN bytes inclusive
- Had no CRC error, alignment error, or code error

See Section 32.2.6.5 for definitions of alignment, code, and CRC errors. Overruns have no effect on this statistic.

32.5.50.2 Broadcast Receive Frames Register (RXBCASTFRAMES) (offset = 204h)

The total number of good broadcast frames received on the EMAC. A good broadcast frame is defined as having all of the following:

- Any data or MAC control frame that was destined for address FF-FF-FF-FF-FF-FFh only
- Was of length 64 to RXMAXLEN bytes inclusive
- Had no CRC error, alignment error, or code error

See Section 32.2.6.5 for definitions of alignment, code, and CRC errors. Overruns have no effect on this statistic.

32.5.50.3 Multicast Receive Frames Register (RXMCASTFRAMES) (offset = 208h)

The total number of good multicast frames received on the EMAC. A good multicast frame is defined as having all of the following:

- Any data or MAC control frame that was destined for any multicast address other than FF-FF-FF-FF-FF-FFh
- Was of length 64 to RXMAXLEN bytes inclusive
- Had no CRC error, alignment error, or code error

See Section 32.2.6.5 for definitions of alignment, code, and CRC errors. Overruns have no effect on this statistic.

32.5.50.4 Pause Receive Frames Register (RXPAUSEFRAMES) (offset = 20Ch)

The total number of IEEE 802.3X pause frames received by the EMAC (whether acted upon or not). A pause frame is defined as having all of the following:

- Contained any unicast, broadcast, or multicast address
- Contained the length/type field value 88.08h and the opcode 0001h
- Was of length 64 to RXMAXLEN bytes inclusive
- Had no CRC error, alignment error, or code error
- Pause-frames had been enabled on the EMAC (TXFLOWEN bit is set in MACCONTROL).

The EMAC could have been in either half-duplex or full-duplex mode. See Section 32.2.6.5 for definitions of alignment, code, and CRC errors. Overruns have no effect on this statistic.

32.5.50.5 Receive CRC Errors Register (RXCRCERRORS) (offset = 210h)

The total number of frames received on the EMAC that experienced a CRC error. A frame with CRC errors is defined as having all of the following:

- Was any data or MAC control frame that matched a unicast, broadcast, or multicast address, or matched due to promiscuous mode
- Was of length 64 to RXMAXLEN bytes inclusive
- Had no alignment or code error
- Had a CRC error. A CRC error is defined as having all of the following:
  - A frame containing an even number of nibbles
  - Fails the frame check sequence test

See Section 32.2.6.5 for definitions of alignment, code, and CRC errors. Overruns have no effect on this statistic.

32.5.50.6 Receive Alignment/Code Errors Register (RXALIGNCODEERRORS) (offset = 214h)

The total number of frames received on the EMAC that experienced an alignment error or code error. Such a frame is defined as having all of the following:

- Was any data or MAC control frame that matched a unicast, broadcast, or multicast address, or matched due to promiscuous mode
- Was of length 64 to RXMAXLEN bytes inclusive
- Had either an alignment error or a code error
  - An alignment error is defined as having all of the following:
    - A frame containing an odd number of nibbles
    - Fails the frame check sequence test, if the final nibble is ignored
  - A code error is defined as a frame that has been discarded because the EMACs MII_RXER pin is driven with a 1 for at least one bit-time's duration at any point during the frame's reception.

Overruns have no effect on this statistic.

CRC alignment or code errors can be calculated by summing receive alignment errors, receive code errors, and receive CRC errors.

32.5.50.7 Receive Oversized Frames Register (RXOVERSIZED) (offset = 218h)

The total number of oversized frames received on the EMAC. An oversized frame is defined as having all of the following:

- Was any data or MAC control frame that matched a unicast, broadcast, or multicast address, or matched due to promiscuous mode
- Was greater than RXMAXLEN in bytes
- Had no CRC error, alignment error, or code error

See Section 32.2.6.5 for definitions of alignment, code, and CRC errors. Overruns have no effect on this statistic.

32.5.50.8 Receive Jabber Frames Register (RXJABBER) (offset = 21Ch)

The total number of jabber frames received on the EMAC. A jabber frame is defined as having all of the following:

- Was any data or MAC control frame that matched a unicast, broadcast, or multicast address, or matched due to promiscuous mode
- Was greater than RXMAXLEN bytes long
- Had a CRC error, alignment error, or code error

See Section 32.2.6.5 for definitions of alignment, code, and CRC errors. Overruns have no effect on this statistic.

32.5.50.9 Receive Undersized Frames Register (RXUNDERSIZED) (offset = 220h)

The total number of undersized frames received on the EMAC. An undersized frame is defined as having all of the following:

- Was any data frame that matched a unicast, broadcast, or multicast address, or matched due to promiscuous mode
- Was less than 64 bytes long
- Had no CRC error, alignment error, or code error

See Section 32.2.6.5 for definitions of alignment, code, and CRC errors. Overruns have no effect on this statistic.

32.5.50.10 Receive Frame Fragments Register (RXFRAGMENTS) (offset = 224h)

The total number of frame fragments received on the EMAC. A frame fragment is defined as having all of the following:

- Any data frame (address matching does not matter)
- Was less than 64 bytes long
- Had a CRC error, alignment error, or code error
- Was not the result of a collision caused by half duplex, collision based flow control

See Section 32.2.6.5 for definitions of alignment, code, and CRC errors. Overruns have no effect on this statistic.

32.5.50.11 Filtered Receive Frames Register (RXFILTERED) (offset = 228h)

The total number of frames received on the EMAC that the EMAC address matching process indicated should be discarded. Such a frame is defined as having all of the following:

- Was any data frame (not MAC control frame) destined for any unicast, broadcast, or multicast address
- Did not experience any CRC error, alignment error, code error
- The address matching process decided that the frame should be discarded (filtered) because it did not match the unicast, broadcast, or multicast address, and it did not match due to promiscuous mode.

To determine the number of receive frames discarded by the EMAC for any reason, sum the following statistics (promiscuous mode disabled):

- Receive fragments
- Receive undersized frames
- Receive CRC errors
- Receive alignment/code errors
- Receive jabbers
- Receive overruns
- Receive filtered frames

This may not be an exact count because the receive overruns statistic is independent of the other statistics, so if an overrun occurs at the same time as one of the other discard reasons, then the above sum double-counts that frame.

32.5.50.12 Receive QOS Filtered Frames Register (RXQOSFILTERED) (offset = 22Ch)

The total number of frames received on the EMAC that were filtered due to receive quality of service (QOS) filtering. Such a frame is defined as having all of the following:

- Any data or MAC control frame that matched a unicast, broadcast, or multicast address, or matched due to promiscuous mode
- The frame destination channel flow control threshold register (RX*n*FLOWTHRESH) value was greater than or equal to the channel's corresponding free buffer register (RX*n*FREEBUFFER) value
- Was of length 64 to RXMAXLEN
- RXQOSEN bit is set in RXMBPENABLE
- Had no CRC error, alignment error, or code error

See Section 32.2.6.5 for definitions of alignment, code, and CRC errors. Overruns have no effect on this statistic.

32.5.50.13 Receive Octet Frames Register (RXOCTETS) (offset = 230h)

The total number of bytes in all good frames received on the EMAC. A good frame is defined as having all of the following:

- Any data or MAC control frame that matched a unicast, broadcast, or multicast address, or matched due to promiscuous mode
- Was of length 64 to RXMAXLEN bytes inclusive
- Had no CRC error, alignment error, or code error

See Section 32.2.6.5 for definitions of alignment, code, and CRC errors. Overruns have no effect on this statistic.

32.5.50.14 Good Transmit Frames Register (TXGOODFRAMES) (offset = 234h)

The total number of good frames transmitted on the EMAC. A good frame is defined as having all of the following:

- Any data or MAC control frame that was destined for any unicast, broadcast, or multicast address
- Was any length
- Had no late or excessive collisions, no carrier loss, and no underrun

32.5.50.15 Broadcast Transmit Frames Register (TXBCASTFRAMES) (offset = 238h)

The total number of good broadcast frames transmitted on the EMAC. A good broadcast frame is defined as having all of the following:

- Any data or MAC control frame destined for address FF-FF-FF-FF-FF-FFh only
- Was of any length
- Had no late or excessive collisions, no carrier loss, and no underrun

32.5.50.16 Multicast Transmit Frames Register (TXMCASTFRAMES) (offset = 23Ch)

The total number of good multicast frames transmitted on the EMAC. A good multicast frame is defined as having all of the following:

- Any data or MAC control frame destined for any multicast address other than FF-FF-FF-FF-FF-FFh
- Was of any length
- Had no late or excessive collisions, no carrier loss, and no underrun

32.5.50.17 Pause Transmit Frames Register (TXPAUSEFRAMES) (offset = 240h)

The total number of IEEE 802.3X pause frames transmitted by the EMAC. Pause frames cannot underrun or contain a CRC error because they are created in the transmitting MAC, so these error conditions have no effect on this statistic. Pause frames sent by software are not included in this count. Since pause frames are only transmitted in full-duplex mode, carrier loss and collisions have no effect on this statistic.

Transmitted pause frames are always 64-byte multicast frames so appear in the multicast transmit frames register and 64 octect frames register statistics.

32.5.50.18 Deferred Transmit Frames Register (TXDEFERRED) (offset = 244h)

The total number of frames transmitted on the EMAC that first experienced deferment. Such a frame is defined as having all of the following:

- Was any data or MAC control frame destined for any unicast, broadcast, or multicast address
- Was any size
- Had no carrier loss and no underrun
- Experienced no collisions before being successfully transmitted
- Found the medium busy when transmission was first attempted, so had to wait.

CRC errors have no effect on this statistic.

32.5.50.19 Transmit Collision Frames Register (TXCOLLISION) (offset = 248h)

The total number of times that the EMAC experienced a collision. Collisions occur under two circumstances:

- When a transmit data or MAC control frame has all of the following:
  - Was destined for any unicast, broadcast, or multicast address
  - Was any size
  - Had no carrier loss and no underrun
  - Experienced a collision. A jam sequence is sent for every non-late collision, so this statistic increments on each occasion if a frame experiences multiple collisions (and increments on late collisions).
- When the EMAC is in half-duplex mode, flow control is active, and a frame reception begins.

CRC errors have no effect on this statistic.

32.5.50.20 Transmit Single Collision Frames Register (TXSINGLECOLL) (offset = 24Ch)

The total number of frames transmitted on the EMAC that experienced exactly one collision. Such a frame is defined as having all of the following:

- Was any data or MAC control frame destined for any unicast, broadcast, or multicast address
- Was any size
- Had no carrier loss and no underrun
- Experienced one collision before successful transmission. The collision was not late.

CRC errors have no effect on this statistic.

32.5.50.21 Transmit Multiple Collision Frames Register (TXMULTICOLL) (offset = 250h)

The total number of frames transmitted on the EMAC that experienced multiple collisions. Such a frame is defined as having all of the following:

- Was any data or MAC control frame destined for any unicast, broadcast, or multicast address
- Was any size
- Had no carrier loss and no underrun
- Experienced 2 to 15 collisions before being successfully transmitted. None of the collisions were late.

CRC errors have no effect on this statistic.

32.5.50.22 Transmit Excessive Collision Frames Register (TXEXCESSIVECOLL) (offset = 254h)

The total number of frames when transmission was abandoned due to excessive collisions. Such a frame is defined as having all of the following:

- Was any data or MAC control frame destined for any unicast, broadcast, or multicast address
- Was any size
- Had no carrier loss and no underrun
- Experienced 16 collisions before abandoning all attempts at transmitting the frame. None of the collisions were late.

CRC errors have no effect on this statistic.

32.5.50.23 Transmit Late Collision Frames Register (TXLATECOLL) (offset = 258h)

The total number of frames when transmission was abandoned due to a late collision. Such a frame is defined as having all of the following:

- Was any data or MAC control frame destined for any unicast, broadcast, or multicast address
- Was any size
- Had no carrier loss and no underrun
- Experienced a collision later than 512 bit-times into the transmission. There may have been up to 15 previous (non-late) collisions that had previously required the transmission to be reattempted. The late collisions statistic dominates over the single, multiple, and excessive collisions statistics. If a late collision occurs, the frame is not counted in any of these other three statistics.

CRC errors, carrier loss, and underrun have no effect on this statistic.

32.5.50.24 Transmit Underrun Error Register (TXUNDERRUN) (offset = 25Ch)

The number of frames sent by the EMAC that experienced FIFO underrun. Late collisions, CRC errors, carrier loss, and underrun have no effect on this statistic.

32.5.50.25 Transmit Carrier Sense Errors Register (TXCARRIERSENSE) (offset = 260h)

The total number of frames on the EMAC that experienced carrier loss. Such a frame is defined as having all of the following:

- Was any data or MAC control frame destined for any unicast, broadcast, or multicast address
- Was any size
- The carrier sense condition was lost or never asserted when transmitting the frame (the frame is not retransmitted)

CRC errors and underrun have no effect on this statistic.

32.5.50.26 Transmit Octet Frames Register (TXOCTETS) (offset = 264h)

The total number of bytes in all good frames transmitted on the EMAC. A good frame is defined as having all of the following:

- Any data or MAC control frame that was destined for any unicast, broadcast, or multicast address
- Was any length
- Had no late or excessive collisions, no carrier loss, and no underrun

32.5.50.27 Transmit and Receive 64 Octet Frames Register (FRAME64) (offset = 268h)

The total number of 64-byte frames received and transmitted on the EMAC. Such a frame is defined as having all of the following:

- Any data or MAC control frame that was destined for any unicast, broadcast, or multicast address
- Did not experience late collisions, excessive collisions, underrun, or carrier sense error
- Was exactly 64-bytes long. (If the frame was being transmitted and experienced carrier loss that resulted in a frame of this size being transmitted, then the frame is recorded in this statistic).

CRC errors, alignment/code errors, and overruns do not affect the recording of frames in this statistic.

32.5.50.28 Transmit and Receive 65 to 127 Octet Frames Register (FRAME65T127) (offset = 26Ch)

The total number of 65-byte to 127-byte frames received and transmitted on the EMAC. Such a frame is defined as having all of the following:

- Any data or MAC control frame that was destined for any unicast, broadcast, or multicast address
- Did not experience late collisions, excessive collisions, underrun, or carrier sense error
- Was 65-bytes to 127-bytes long

CRC errors, alignment/code errors, underruns, and overruns do not affect the recording of frames in this statistic.

32.5.50.29 Transmit and Receive 128 to 255 Octet Frames Register (FRAME128T255) (offset = 270h)

The total number of 128-byte to 255-byte frames received and transmitted on the EMAC. Such a frame is defined as having all of the following:

- Any data or MAC control frame that was destined for any unicast, broadcast, or multicast address
- Did not experience late collisions, excessive collisions, underrun, or carrier sense error
- Was 128-bytes to 255-bytes long

CRC errors, alignment/code errors, underruns, and overruns do not affect the recording of frames in this statistic.

32.5.50.30 Transmit and Receive 256 to 511 Octet Frames Register (FRAME256T511) (offset = 274h)

The total number of 256-byte to 511-byte frames received and transmitted on the EMAC. Such a frame is defined as having all of the following:

- Any data or MAC control frame that was destined for any unicast, broadcast, or multicast address
- Did not experience late collisions, excessive collisions, underrun, or carrier sense error
- Was 256-bytes to 511-bytes long

CRC errors, alignment/code errors, underruns, and overruns do not affect the recording of frames in this statistic.

32.5.50.31 Transmit and Receive 512 to 1023 Octet Frames Register (FRAME512T1023) (offset = 278h)

The total number of 512-byte to 1023-byte frames received and transmitted on the EMAC. Such a frame is defined as having all of the following:

- Any data or MAC control frame that was destined for any unicast, broadcast, or multicast address
- Did not experience late collisions, excessive collisions, underrun, or carrier sense error
- Was 512-bytes to 1023-bytes long

CRC errors, alignment/code errors, and overruns do not affect the recording of frames in this statistic.

32.5.50.32 Transmit and Receive 1024 to RXMAXLEN Octet Frames Register (FRAME1024TUP) (offset = 27Ch)

The total number of 1024-byte to RXMAXLEN-byte frames received and transmitted on the EMAC. Such a frame is defined as having all of the following:

- Any data or MAC control frame that was destined for any unicast, broadcast, or multicast address
- Did not experience late collisions, excessive collisions, underrun, or carrier sense error
- Was 1024-bytes to RXMAXLEN-bytes long

CRC/alignment/code errors, underruns, and overruns do not affect frame recording in this statistic.

32.5.50.33 Network Octet Frames Register (NETOCTETS) (offset = 280h)

The total number of bytes of frame data received and transmitted on the EMAC. Each frame counted has all of the following:

- Was any data or MAC control frame destined for any unicast, broadcast, or multicast address (address match does not matter)
- Was of any size (including less than 64-byte and greater than RXMAXLEN-byte frames)

Also counted in this statistic is:

- Every byte transmitted before a carrier-loss was experienced
- Every byte transmitted before each collision was experienced (multiple retries are counted each time)
- Every byte received if the EMAC is in half-duplex mode until a jam sequence was transmitted to initiate flow control. (The jam sequence is not counted to prevent double-counting).

Error conditions such as alignment errors, CRC errors, code errors, overruns, and underruns do not affect the recording of bytes in this statistic. The objective of this statistic is to give a reasonable indication of Ethernet utilization.

32.5.50.34 Receive FIFO or DMA Start of Frame Overruns Register (RXSOFOVERRUNS) (offset = 284h)

The total number of frames received on the EMAC that had either a FIFO or DMA start of frame (SOF) overrun. An SOF overrun frame is defined as having all of the following:

- Was any data or MAC control frame that matched a unicast, broadcast, or multicast address, or matched due to promiscuous mode
- Was of any size (including less than 64-byte and greater than RXMAXLEN-byte frames)
- The EMAC was unable to receive it because it did not have the resources to receive it (cell FIFO full or no DMA buffer available at the start of the frame).

CRC errors, alignment errors, and code errors have no effect on this statistic.

32.5.50.35 Receive FIFO or DMA Middle of Frame Overruns Register (RXMOFOVERRUNS) (offset = 288h)

The total number of frames received on the EMAC that had either a FIFO or DMA middle of frame (MOF) overrun. An MOF overrun frame is defined as having all of the following:

- Was any data or MAC control frame that matched a unicast, broadcast, or multicast address, or matched due to promiscuous mode
- Was of any size (including less than 64-byte and greater than RXMAXLEN-byte frames)
- The EMAC was unable to receive it because it did not have the resources to receive it (cell FIFO full or no DMA buffer available after the frame was successfully started - no SOF overrun).

CRC errors, alignment errors, and code errors have no effect on this statistic.

32.5.50.36 Receive DMA Overruns Register (RXDMAOVERRUNS) (offset = 28Ch)

The total number of frames received on the EMAC that had either a DMA start of frame (SOF) overrun or a DMA middle of frame (MOF) overrun. A receive DMA overrun frame is defined as having all of the following:

- Was any data or MAC control frame that matched a unicast, broadcast, or multicast address, or matched due to promiscuous mode
- Was of any size (including less than 64-byte and greater than RXMAXLEN-byte frames)
- The EMAC was unable to receive it because it did not have the DMA buffer resources to receive it (zero head descriptor pointer at the start or during the middle of the frame reception).

CRC errors, alignment errors, and code errors have no effect on this statistic.

Enhanced Capture (eCAP) Module

The enhanced Capture (eCAP) module is essential in systems where accurate timing of external events is important. This microcontroller implements 6 instances of the eCAP module.

Topic **Page**

33.1 Introduction

Uses for eCAP include:

- Speed measurements of rotating machinery (for example, toothed sprockets sensed via Hall sensors)
- Elapsed time measurements between position sensor pulses
- Period and duty cycle measurements of pulse train signals
- Decoding current or voltage amplitude derived from duty cycle encoded current/voltage sensors

33.1.1 Features

The eCAP module includes the following features:

- 4-event time-stamp registers (each 32 bits)
- Edge polarity selection for up to four sequenced time-stamp capture events
- Interrupt on either of the four events
- Single shot capture of up to four event time-stamps
- Continuous mode capture of time-stamps in a four-deep circular buffer
- Absolute time-stamp capture
- Difference (Delta) mode time-stamp capture
- All above resources dedicated to a single input pin
- When not used in capture mode, the ECAP module can be configured as a single channel PWM output

33.1.2 Description

One eCAP channel has the following independent key resources:

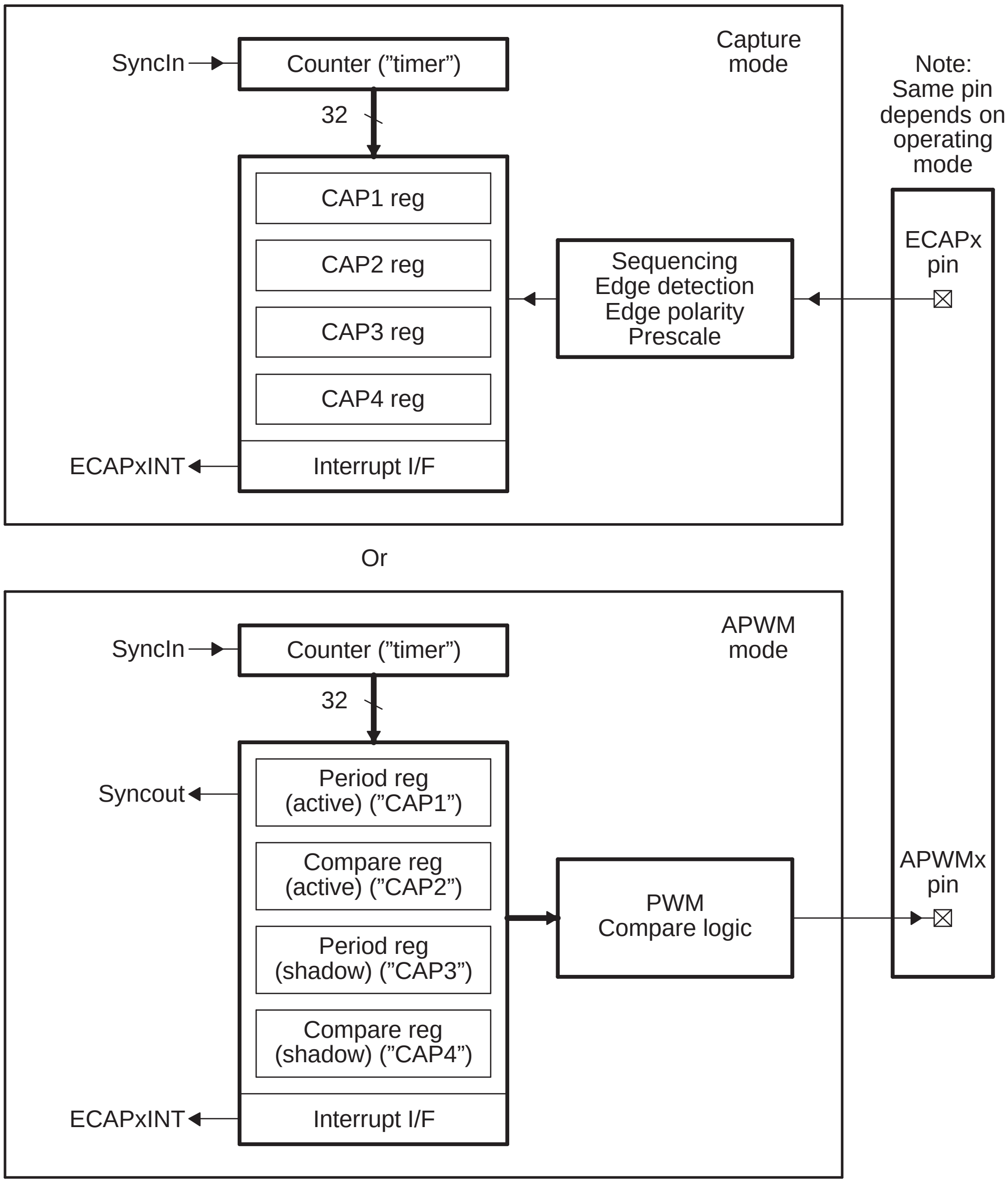
- Dedicated input capture pin
- 32-bit time base (counter)
- 4 x 32-bit time-stamp capture registers (CAP1-CAP4)
- 4-stage sequencer (Modulo4 counter) that is synchronized to external events, ECAP pin rising/falling edges.
- Independent edge polarity (rising/falling edge) selection for all 4 events
- Input capture signal prescaling (from 2 to 62)
- One-shot compare register (2 bits) to freeze captures after 1 to 4 time-stamp events
- Control for continuous time-stamp captures using a 4-deep circular buffer (CAP1-CAP4) scheme
- Interrupt capabilities on any of the 4 capture events

33.2 Basic Operation

33.2.1 Capture and APWM Operating Mode

You can use the eCAP module resources to implement a single-channel PWM generator (with 32 bit capabilities) when it is not being used for input captures. The counter operates in count-up mode, providing a time-base for asymmetrical pulse width modulation (PWM) waveforms. The CAP1 and CAP2 registers become the active period and compare registers, respectively, while CAP3 and CAP4 registers become the period and capture shadow registers, respectively. Figure 33-1 is a high-level view of both the capture and auxiliary pulse-width modulator (APWM) modes of operation.

Figure 33-1. Capture and APWM Modes of Operation



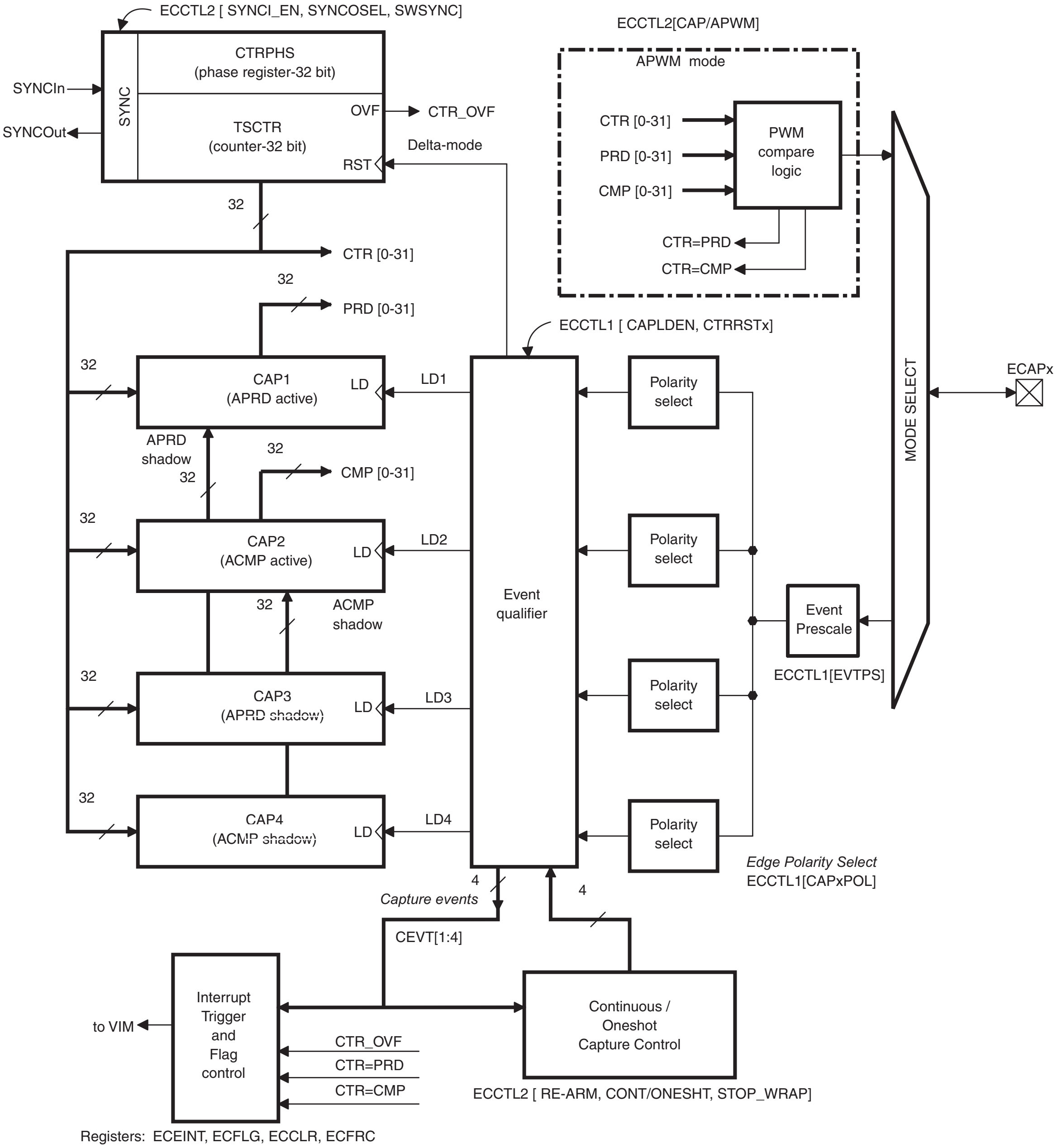
A A single pin is shared between CAP and APWM functions. In capture mode, it is an input; in APWM mode, it is an output.

B In APWM mode, writing any value to CAP1/CAP2 active registers also writes the same value to the corresponding shadow registers CAP3/CAP4. This emulates immediate mode. Writing to the shadow registers CAP3/CAP4 invokes the shadow mode.

33.2.2 Capture Mode Description

Figure 33-2 shows the various components that implement the capture function.

Figure 33-2. Capture Function Diagram

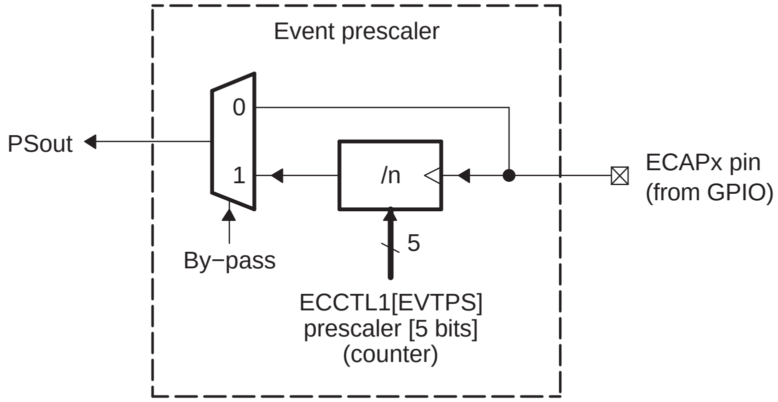


33.2.2.1 Event Prescaler

- An input capture signal (pulse train) can be prescaled by N = 2-62 (in multiples of 2) or can bypass the prescaler.

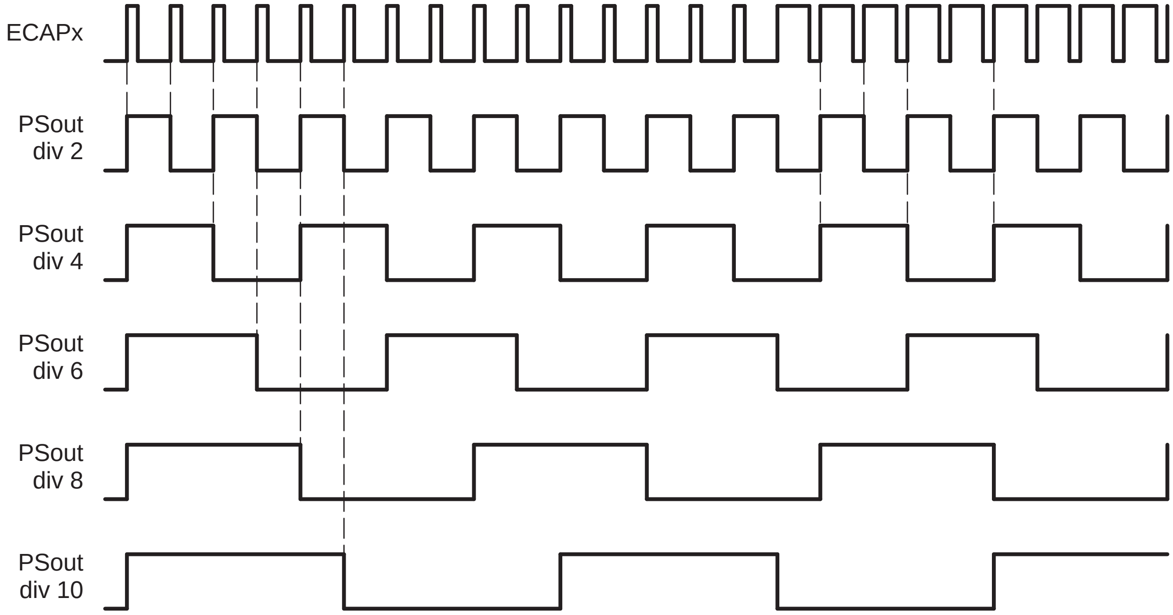
  This is useful when very high frequency signals are used as inputs. Figure 33-3 shows a functional diagram and Figure 33-4 shows the operation of the prescale function.

Figure 33-3. Event Prescale Control



A When a prescale value of 1 is chosen (ECCTL1[13:9] = 0,0,0,0,0), the input capture signal by-passes the prescale logic completely.

Figure 33-4. Prescale Function Waveforms



33.2.2.2 Edge Polarity Select and Qualifier

- Four independent edge polarity (rising edge/falling edge) selection MUXes are used, one for each capture event.
- Each edge (up to 4) is event qualified by the Modulo4 sequencer.
- The edge event is gated to its respective CAPx register by the Mod4 counter. The CAPx register is loaded on the falling edge.

33.2.2.3 Continuous/One-Shot Control

- The Mod4 (2 bit) counter is incremented via edge qualified events (CEVT1-CEVT4).
- The Mod4 counter continues counting (0->1->2->3->0) and wraps around unless stopped.
- A 2-bit stop register is used to compare the Mod4 counter output, and when equal stops the Mod4 counter and inhibits further loads of the CAP1-CAP4 registers. This occurs during one-shot operation.

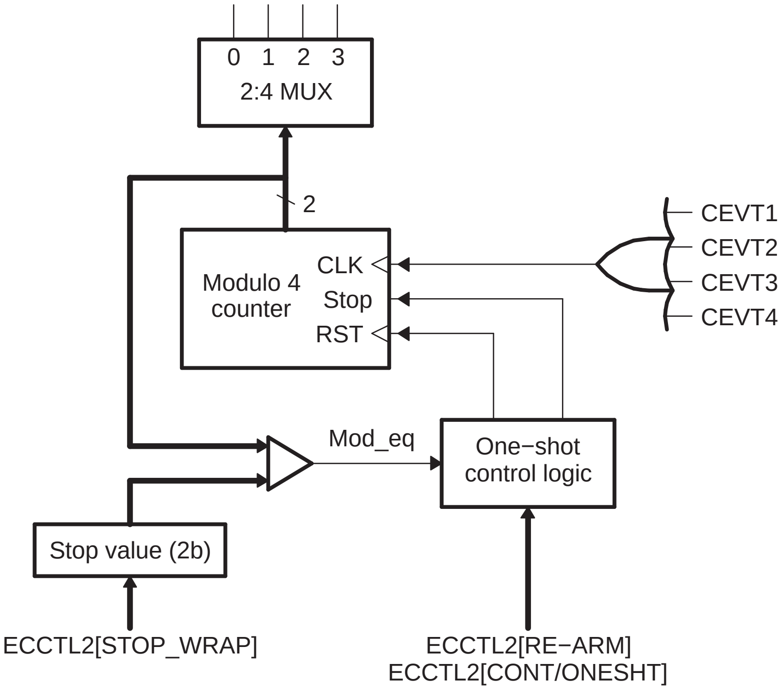
The continuous/one-shot block controls the start/stop and reset (zero) functions of the Mod4 counter via a mono-shot type of action that can be triggered by the stop-value comparator and re-armed via software control.

Once armed, the eCAP module waits for 1-4 (defined by stop-value) capture events before freezing both the Mod4 counter and contents of CAP1-4 registers (time-stamps).

Re-arming prepares the eCAP module for another capture sequence. Also re-arming clears (to zero) the Mod4 counter and permits loading of CAP1-4 registers again, providing the CAPLDEN bit is set.

In continuous mode, the Mod4 counter continues to run (0->1->2->3->0, the one-shot action is ignored, and capture values continue to be written to CAP1-4 in a circular buffer sequence.

Figure 33-5. Continuous/One-shot Block



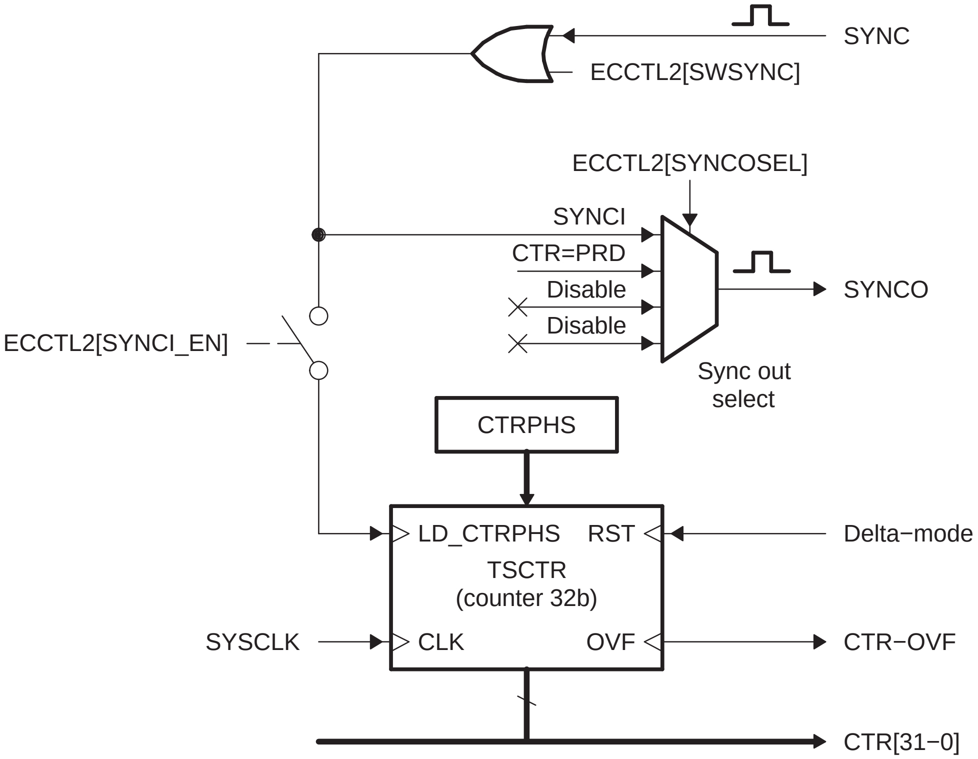
33.2.2.4 32-Bit Counter and Phase Control

This counter provides the time-base for event captures, and is clocked via the system clock.

A phase register is provided to achieve synchronization with other counters, via a hardware and software forced sync. This is useful in APWM mode when a phase offset between modules is needed.

On any of the four event loads, an option to reset the 32-bit counter is given. This is useful for time difference capture. The 32-bit counter value is captured first, then it is reset to 0 by any of the LD1-LD4 signals.

Figure 33-6. Counter and Synchronization Block



33.2.2.5 CAP1-CAP4 Registers

These 32-bit registers are fed by the 32-bit counter timer bus, CTR[0-31] and are loaded (that is, capture a time-stamp) when their respective LD inputs are strobed.

Loading of the capture registers can be inhibited via control bit CAPLDEN. During one-shot operation, this bit is cleared (loading is inhibited) automatically when a stop condition occurs, StopValue = Mod4.

CAP1 and CAP2 registers become the active period and compare registers, respectively, in APWM mode.

CAP3 and CAP4 registers become the respective shadow registers (APRD and ACMP) for CAP1 and CAP2 during APWM operation.

33.2.2.6 Interrupt Control

An Interrupt can be generated on capture events (CEVT1-CEVT4, CTROVF) or APWM events (CTR = PRD, CTR = CMP).

A counter overflow event (FFFFFFFF->00000000) is also provided as an interrupt source (CTROVF).

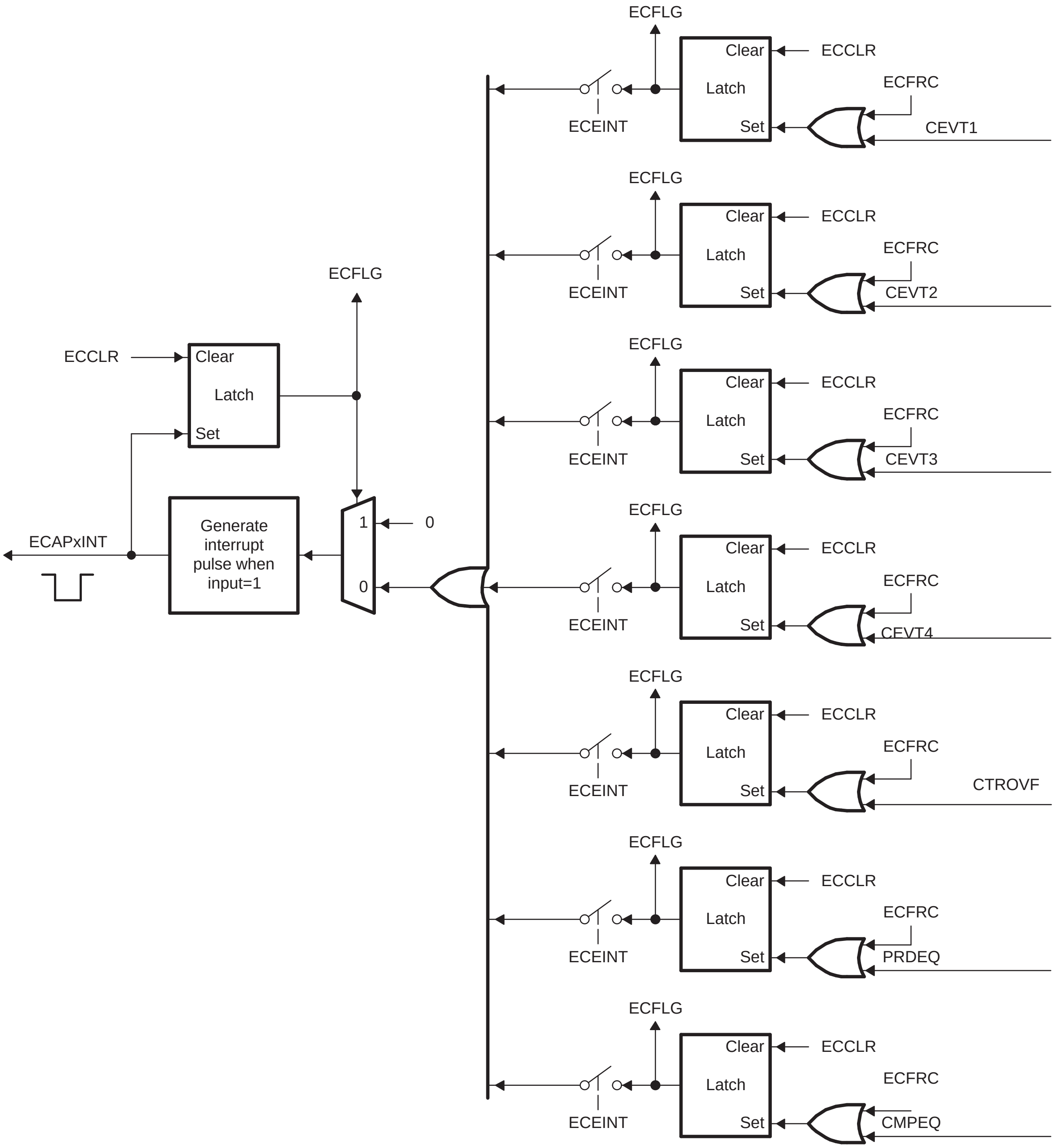
The capture events are edge and sequencer qualified (ordered in time) by the polarity select and Mod4 gating, respectively.

One of these events can be selected as the interrupt source (from the eCAPx module) going to the PIE.

Seven interrupt events (CEVT1, CEVT2, CEVT3, CEVT4, CNTOVF, CTR = PRD, CTR = CMP) can be generated. The interrupt enable register (ECEINT) is used to enable/disable individual interrupt event sources. The interrupt flag register (ECFLG) indicates if any interrupt event has been latched and contains the global interrupt flag bit (INT). An interrupt pulse is generated to the PIE only if any of the interrupt events are enabled, the flag bit is 1, and the INT flag bit is 0. The interrupt service routine must clear the global interrupt flag bit and the serviced event via the interrupt clear register (ECCLR) before any other interrupt pulses are generated. You can force an interrupt event via the interrupt force register (ECFRC). This is useful for test purposes.

Note: The CEVT1, CEVT2, CEVT3, CEVT4 flags are only active in capture mode (ECCTL2[CAP_APWM == 0]). The CTR_PRD and CTR_CMP flags are only valid in APWM mode (ECCTL2[CAP_APWM == 1]). CNTOVF flag is valid in both modes.

Figure 33-7. Interrupts in eCAP Module



33.2.2.7 Shadow Load and Lockout Control

In capture mode, this logic inhibits (locks out) any shadow loading of CAP1 or CAP2 from APRD and ACMP registers, respectively.

In APWM mode, shadow loading is active and two choices are permitted:

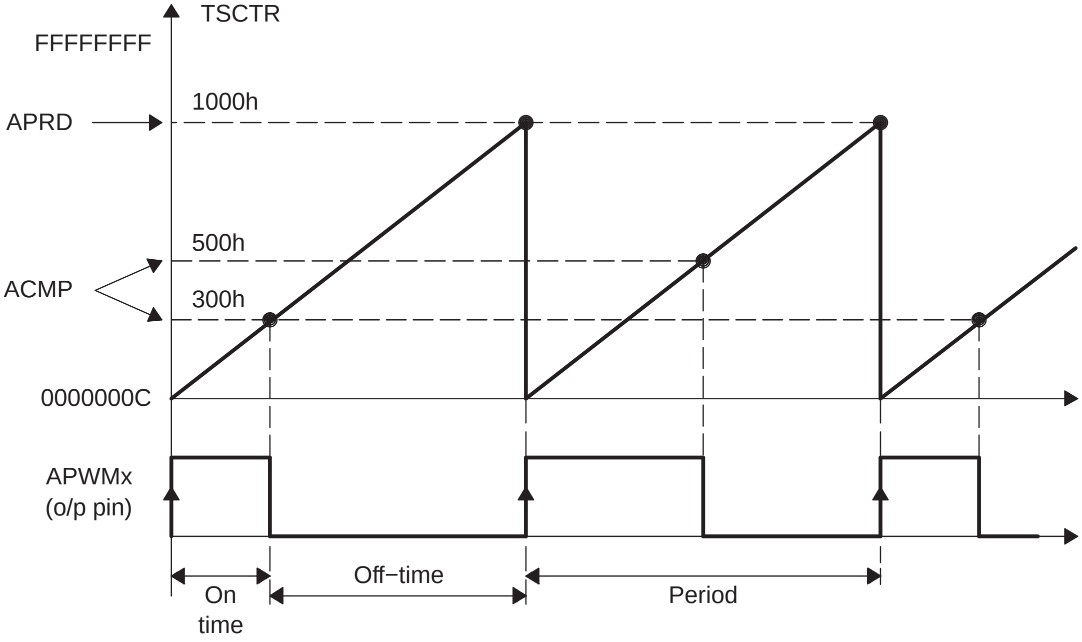
- Immediate - APRD or ACMP are transferred to CAP1 or CAP2 immediately upon writing a new value.
- On period equal, CTR[31:0] = PRD[31:0]

33.2.2.8 APWM Mode Operation

Main operating highlights of the APWM section:

- The time-stamp counter bus is made available for comparison via 2 digital (32-bit) comparators.
- When CAP1/2 registers are not used in capture mode, their contents can be used as Period and Compare values in APWM mode.
- Double buffering is achieved via shadow registers APRD and ACMP (CAP3/4). The shadow register contents are transferred over to CAP1/2 registers either immediately upon a write, or on a CTR = PRD trigger.
- In APWM mode, writing to CAP1/CAP2 active registers will also write the same value to the corresponding shadow registers CAP3/CAP4. This emulates immediate mode. Writing to the shadow registers CAP3/CAP4 will invoke the shadow mode.
- During initialization, you must write to the active registers for both period and compare. This automatically copies the initial values into the shadow values. For subsequent compare updates, during run-time, you only need to use the shadow registers.

Figure 33-8. PWM Waveform Details of APWM Mode Operation



The behavior of APWM active high mode (APWMPOL == 0) is as follows:

```
CMP = 0x00000000, output low for duration of period (0% duty)
CMP = 0x00000001, output high 1 cycle
CMP = 0x00000002, output high 2 cycles
CMP = PERIOD, output high except for 1 cycle (<100% duty)
CMP = PERIOD+1, output high for complete period (100% duty)
CMP > PERIOD+1, output high for complete period
```

The behavior of APWM active low mode (APWMPOL == 1) is as follows:

```
CMP = 0x00000000, output high for duration of period (0% duty)
CMP = 0x00000001, output low 1 cycle
CMP = 0x00000002, output low 2 cycles
CMP = PERIOD, output low except for 1 cycle (<100% duty)
CMP = PERIOD+1, output low for complete period (100% duty)
CMP > PERIOD+1, output low for complete period
```

33.3 Application of the ECAP Module

The following sections will provide Applications examples and code snippets to show how to configure and operate the eCAP module. For clarity and ease of use, the examples use the eCAP “C” header files. Below are useful #defines which will help in the understanding of the examples.

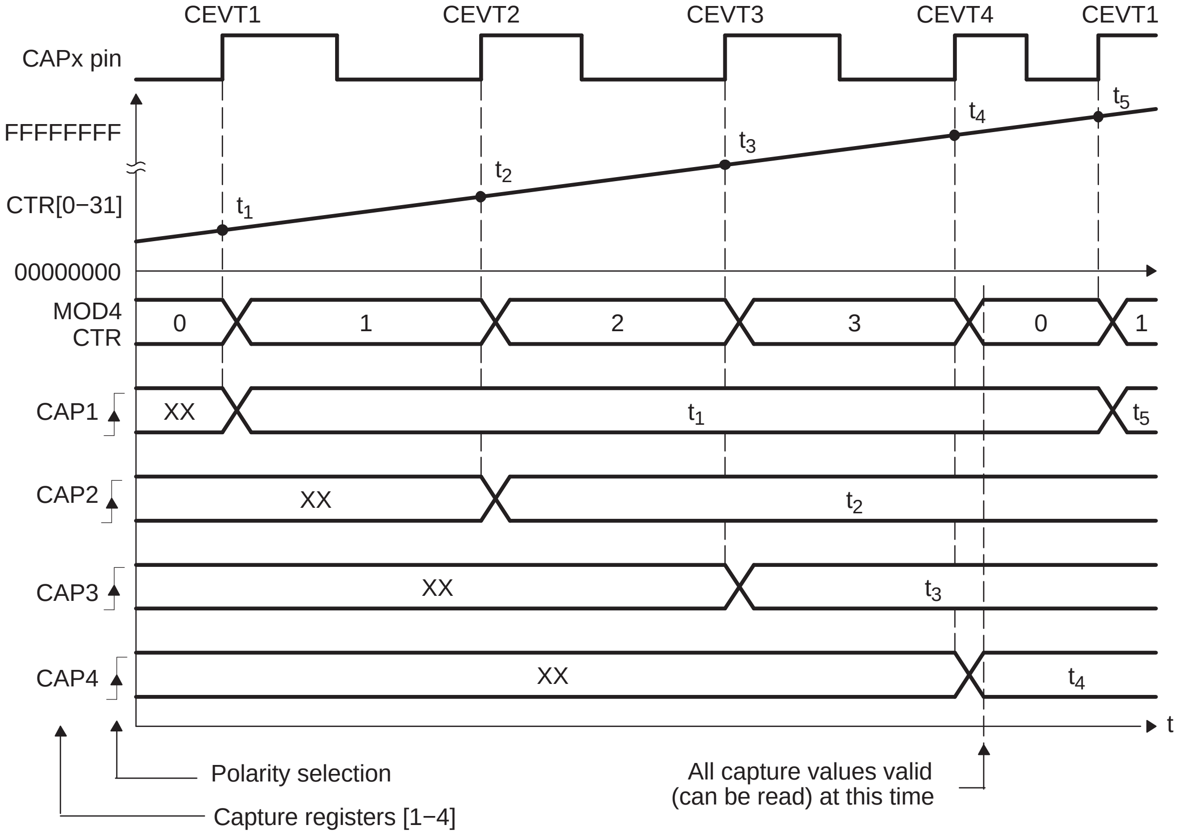
```
// ECCTL1 ( ECAP Control Reg 1)
 //==========================
 // CAPxPOL bits
 #define EC_RISING 0x0
 #define EC_FALLING 0x1
 // CTRRSTx bits
 #define EC_ABS_MODE 0x0
 #define EC_DELTA_MODE 0x1
 // PRESCALE bits
 #define EC_BYPASS 0x0
 #define EC_DIV1 0x0
 #define EC_DIV2 0x1
 #define EC_DIV4 0x2
 #define EC_DIV6 0x3
 #define EC_DIV8 0x4
 #define EC_DIV10 0x5
 // ECCTL2 ( ECAP Control Reg 2)
 //==========================
 // CONT/ONESHOT bit
 #define EC_CONTINUOUS 0x0
 #define EC_ONESHOT 0x1
 // STOPVALUE bit
 #define EC_EVENT1 0x0
 #define EC_EVENT2 0x1
 #define EC_EVENT3 0x2
 #define EC_EVENT4 0x3
 // RE-ARM bit
 #define EC_ARM 0x1
 // TSCTRSTOP bit
 #define EC_FREEZE 0x0
 #define EC_RUN 0x1
 // SYNCO_SEL bit
 #define EC_SYNCIN 0x0
 #define EC_CTR_PRD 0x1
 #define EC_SYNCO_DIS 0x2
 // CAP_APWM mode bit
 #define EC_CAP_MODE 0x0
 #define EC_APWM_MODE 0x1
 // APWMPOL bit
 #define EC_ACTV_HI 0x0
 #define EC_ACTV_LO 0x1
 // Generic
 #define EC_DISABLE 0x0
 #define EC_ENABLE 0x1
 #define EC_FORCE 0x1
```

33.3.1 Example 1 - Absolute Time-Stamp Operation Rising Edge Trigger

Figure 33-9 shows an example of continuous capture operation (Mod4 counter wraps around). In this figure, TSCTR counts-up without resetting and capture events are qualified on the rising edge only, this gives period (and frequency) information.

On an event, the TSCTR contents (time-stamp) is first captured, then Mod4 counter is incremented to the next state. When the TSCTR reaches FFFFFFFF (maximum value), it wraps around to 00000000 (not shown in Figure 33-9), if this occurs, the CTROVF (counter overflow) flag is set, and an interrupt (if enabled) occurs, CTROVF (counter overflow) Flag is set, and an Interrupt (if enabled) occurs. Captured Time-stamps are valid at the point indicated by the diagram, after the 4th event, hence event CEVT4 can conveniently be used to trigger an interrupt and the CPU can read data from the CAPx registers.

Figure 33-9. Capture Sequence for Absolute Time-stamp and Rising Edge Detect



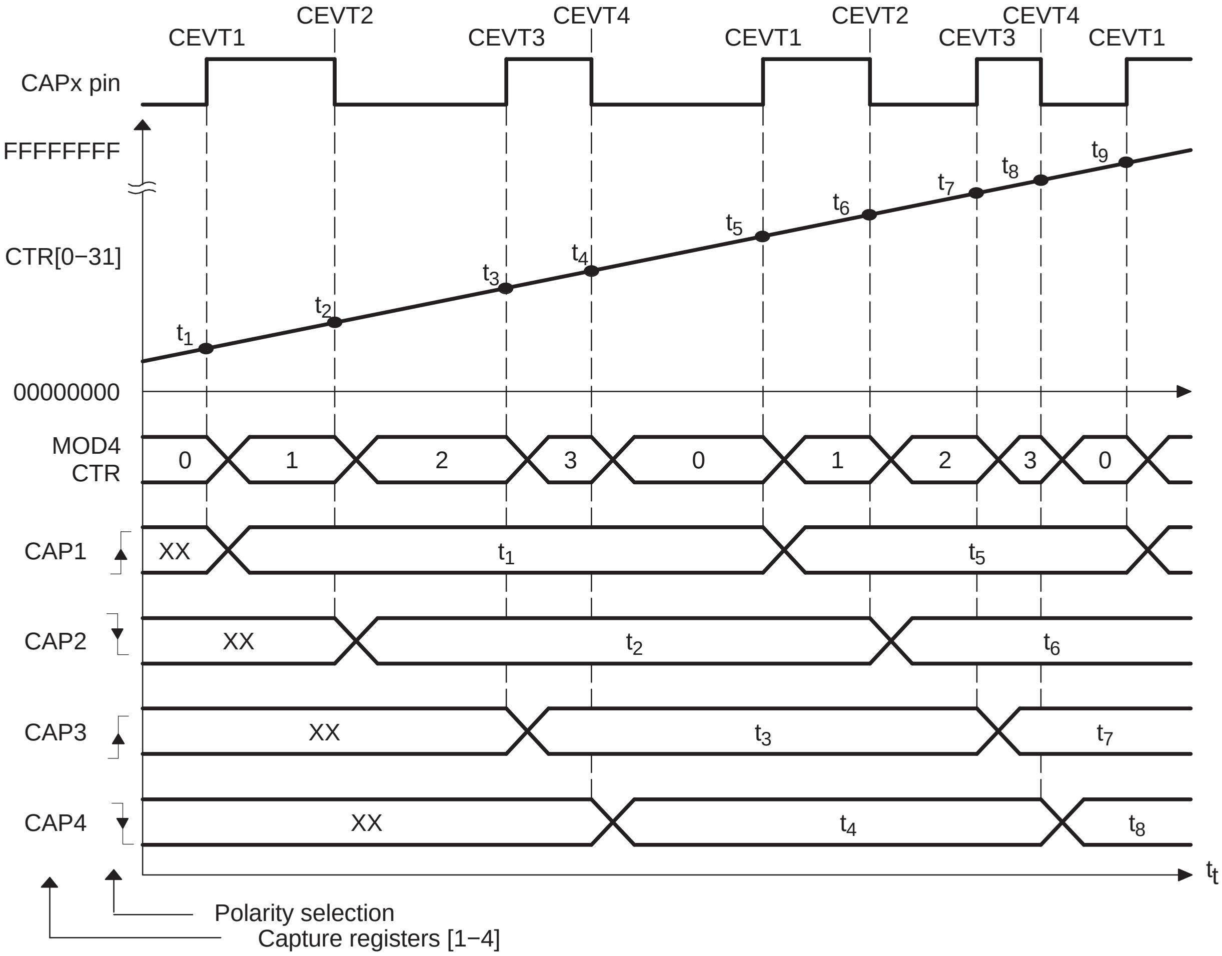
33.3.1.1 Code Snippet for CAP Mode Absolute Time, Rising Edge Trigger

```
// Code snippet for CAP mode Absolute Time, Rising edge trigger
 // Initialization Time
 //=======================
 // ECAP module 1 config
 ECap1Regs.ECCTL1.bit.CAP1POL = EC_RISING;
 ECap1Regs.ECCTL1.bit.CAP2POL = EC_RISING;
 ECap1Regs.ECCTL1.bit.CAP3POL = EC_RISING;
 ECap1Regs.ECCTL1.bit.CAP4POL = EC_RISING;
 ECap1Regs.ECCTL1.bit.CTRRST1 = EC_ABS_MODE;
 ECap1Regs.ECCTL1.bit.CTRRST2 = EC_ABS_MODE;
 ECap1Regs.ECCTL1.bit.CTRRST3 = EC_ABS_MODE;
 ECap1Regs.ECCTL1.bit.CTRRST4 = EC_ABS_MODE;
 ECap1Regs.ECCTL1.bit.CAPLDEN = EC_ENABLE;
 ECap1Regs.ECCTL1.bit.PRESCALE = EC_DIV1;
 ECap1Regs.ECCTL2.bit.CAP_APWM = EC_CAP_MODE;
 ECap1Regs.ECCTL2.bit.CONT_ONESHT = EC_CONTINUOUS;
 ECap1Regs.ECCTL2.bit.SYNCO_SEL = EC_SYNCO_DIS;
 ECap1Regs.ECCTL2.bit.SYNCI_EN = EC_DISABLE;
 ECap1Regs.ECCTL2.bit.TSCTRSTOP = EC_RUN; // Allow TSCTR to run
 // Run Time ( e.g. CEVT4 triggered ISR call)
 //==========================================
 TSt1 = ECap1Regs.CAP1; // Fetch Time-Stamp captured at t1
 TSt2 = ECap1Regs.CAP2; // Fetch Time-Stamp captured at t2
 TSt3 = ECap1Regs.CAP3; // Fetch Time-Stamp captured at t3
 TSt4 = ECap1Regs.CAP4; // Fetch Time-Stamp captured at t4
 Period1 = TSt2-TSt1; // Calculate 1st period
 Period2 = TSt3-TSt2; // Calculate 2nd period
 Period3 = TSt4-TSt3; // Calculate 3rd period
```

33.3.2 Example 2 - Absolute Time-Stamp Operation Rising and Falling Edge Trigger

In Figure 33-10, the eCAP operating mode is almost the same as in the previous section except capture events are qualified as either rising or falling edge, this now gives both period and duty cycle information: Period1 = $t_3 - t_1$, Period2 = $t_5 - t_3$, …etc. Duty Cycle1 (on-time %) = ($t_2 - t_1$) / Period1 x 100%, etc. Duty Cycle1 (off-time %) = ($t_3 - t_2$) / Period1 x 100%, etc.

Figure 33-10. Capture Sequence for Absolute Time-stamp With Rising and Falling Edge Detect



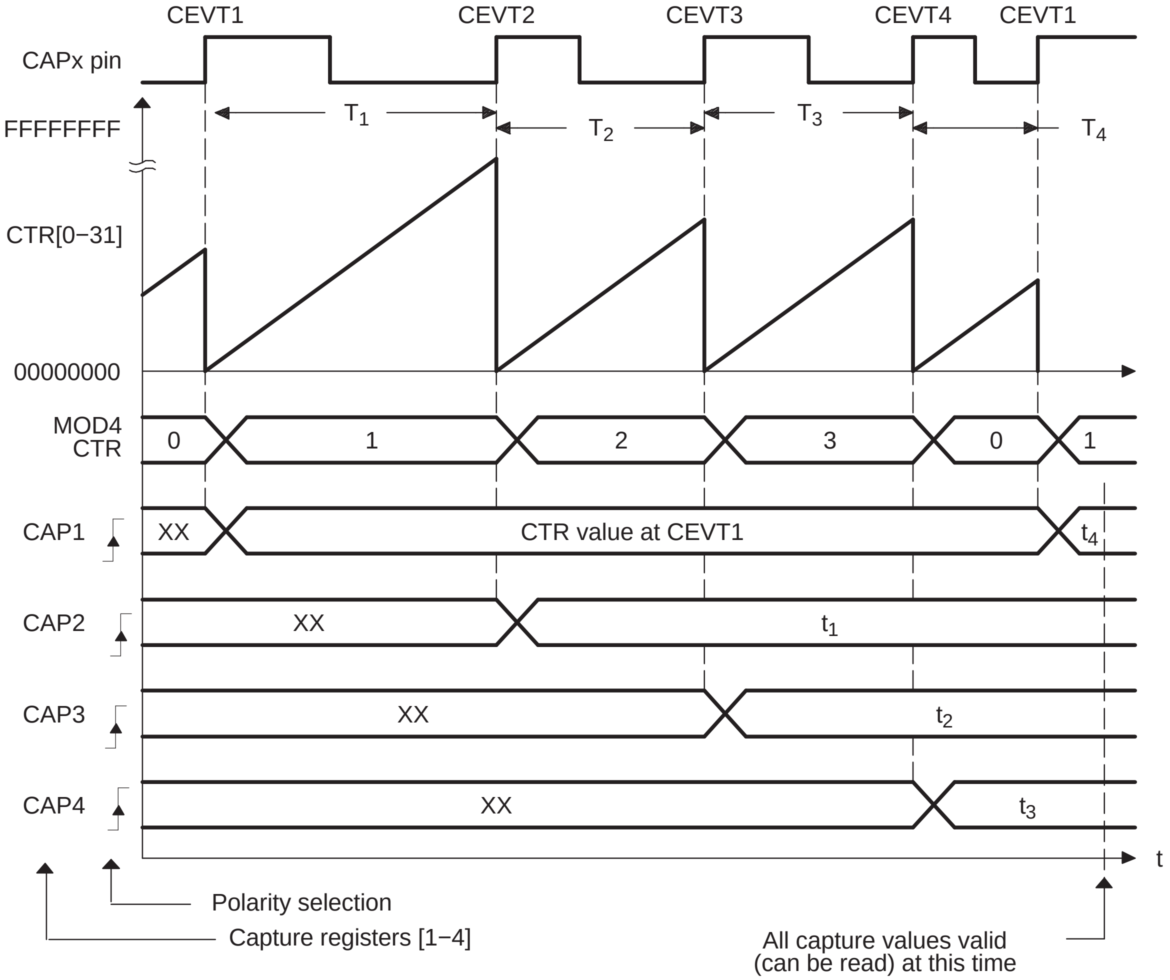
33.3.2.1 Code Snippet for CAP Mode Absolute Time, Rising and Falling Edge Triggers

```
// Code snippet for CAP mode Absolute Time, Rising & Falling edge triggers
 // Initialization Time
 //=======================
 // ECAP module 1 config
 ECap1Regs.ECCTL1.bit.CAP1POL = EC_RISING;
 ECap1Regs.ECCTL1.bit.CAP2POL = EC_FALLING;
 ECap1Regs.ECCTL1.bit.CAP3POL = EC_RISING;
 ECap1Regs.ECCTL1.bit.CAP4POL = EC_FALLING;
 ECap1Regs.ECCTL1.bit.CTRRST1 = EC_ABS_MODE;
 ECap1Regs.ECCTL1.bit.CTRRST2 = EC_ABS_MODE;
 ECap1Regs.ECCTL1.bit.CTRRST3 = EC_ABS_MODE;
 ECap1Regs.ECCTL1.bit.CTRRST4 = EC_ABS_MODE;
 ECap1Regs.ECCTL1.bit.CAPLDEN = EC_ENABLE;
 ECap1Regs.ECCTL1.bit.PRESCALE = EC_DIV1;
 ECap1Regs.ECCTL2.bit.CAP_APWM = EC_CAP_MODE;
 ECap1Regs.ECCTL2.bit.CONT_ONESHT = EC_CONTINUOUS;
 ECap1Regs.ECCTL2.bit.SYNCO_SEL = EC_SYNCO_DIS;
 ECap1Regs.ECCTL2.bit.SYNCI_EN = EC_DISABLE;
 ECap1Regs.ECCTL2.bit.TSCTRSTOP = EC_RUN; // Allow TSCTR to run
 // Run Time ( e.g. CEVT4 triggered ISR call)
 //==========================================
 TSt1 = ECap1Regs.CAP1; // Fetch Time-Stamp captured at t1
 TSt2 = ECap1Regs.CAP2; // Fetch Time-Stamp captured at t2
 TSt3 = ECap1Regs.CAP3; // Fetch Time-Stamp captured at t3
 TSt4 = ECap1Regs.CAP4; // Fetch Time-Stamp captured at t4
 Period1 = TSt3-TSt1; // Calculate 1st period
 DutyOnTime1 = TSt2-TSt1; // Calculate On time
 DutyOffTime1 = TSt3-TSt2; // Calculate Off time
```

33.3.3 Example 3 - Time Difference (Delta) Operation Rising Edge Trigger

Figure 33-11 shows an example of how the eCAP module can be used to collect Delta timing data from pulse train waveforms. Here Continuous Capture mode (TSCTR counts-up without resetting, and Mod4 counter wraps around) is used. In Delta-time mode, TSCTR is Reset back to Zero on every valid event. Here Capture events are qualified as Rising edge only. On an event, TSCTR contents (Time-Stamp) is captured first, and then TSCTR is reset to Zero. The Mod4 counter then increments to the next state. If TSCTR reaches FFFFFFFF (maximum value), before the next event, it wraps around to 00000000 and continues, a CNTOVF (counter overflow) Flag is set, and an Interrupt (if enabled) occurs. The advantage of Delta-time Mode is that the CAPx contents directly give timing data without the need for CPU calculations: Period1 = T_1, Period2 = T_2,…etc. As shown in the diagram, the CEVT1 event is a good trigger point to read the timing data, T_1, T_2, T_3, T_4 are all valid here.

Figure 33-11. Capture Sequence for Delta Mode Time-stamp and Rising Edge Detect



33.3.3.1 Code Snippet for CAP Mode Delta Time, Rising Edge Trigger

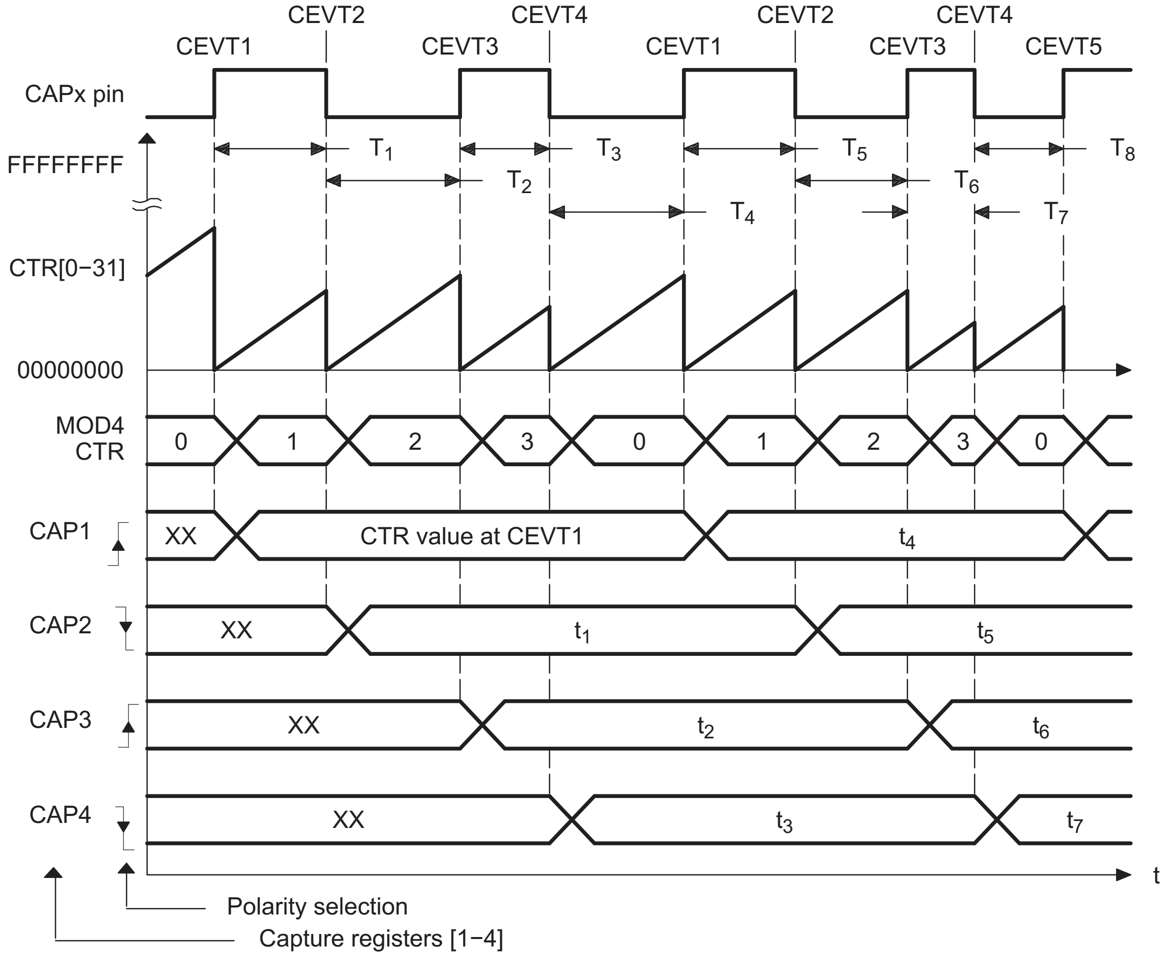
```
// Code snippet for CAP mode Delta Time, Rising edge trigger
 // Initialization Time
 //=======================
 // ECAP module 1 config
 ECap1Regs.ECCTL1.bit.CAP1POL = EC_RISING;
 ECap1Regs.ECCTL1.bit.CAP2POL = EC_RISING;
 ECap1Regs.ECCTL1.bit.CAP3POL = EC_RISING;
 ECap1Regs.ECCTL1.bit.CAP4POL = EC_RISING;
 ECap1Regs.ECCTL1.bit.CTRRST1 = EC_DELTA_MODE;
 ECap1Regs.ECCTL1.bit.CTRRST2 = EC_DELTA_MODE;
 ECap1Regs.ECCTL1.bit.CTRRST3 = EC_DELTA_MODE;
 ECap1Regs.ECCTL1.bit.CTRRST4 = EC_DELTA_MODE;
 ECap1Regs.ECCTL1.bit.CAPLDEN = EC_ENABLE;
 ECap1Regs.ECCTL1.bit.PRESCALE = EC_DIV1;
 ECap1Regs.ECCTL2.bit.CAP_APWM = EC_CAP_MODE;
 ECap1Regs.ECCTL2.bit.CONT_ONESHT = EC_CONTINUOUS;
 ECap1Regs.ECCTL2.bit.SYNCO_SEL = EC_SYNCO_DIS;
 ECap1Regs.ECCTL2.bit.SYNCI_EN = EC_DISABLE;
 ECap1Regs.ECCTL2.bit.TSCTRSTOP = EC_RUN; // Allow TSCTR to run
 // Run Time ( e.g. CEVT1 triggered ISR call)
 //==========================================
 // Note: here Time-stamp directly represents the Period value.
 Period4 = ECap1Regs.CAP1; // Fetch Time-Stamp captured at T1
 Period1 = ECap1Regs.CAP2; // Fetch Time-Stamp captured at T2
 Period2 = ECap1Regs.CAP3; // Fetch Time-Stamp captured at T3
 Period3 = ECap1Regs.CAP4; // Fetch Time-Stamp captured at T4
```

33.3.4 Example 4 - Time Difference (Delta) Operation Rising and Falling Edge Trigger

In Figure 33-12, the eCAP operating mode is almost the same as in previous section except Capture events are qualified as either Rising or Falling edge, this now gives both Period and Duty cycle information: Period1 = T_1+T_2, Period2 = T_3+T_4, …etc Duty Cycle1 (on-time %) = T_1 / Period1 x 100%, etc Duty Cycle1 (off-time %) = T_2 / Period1 x 100%, etc

During initialization, you must write to the active registers for both period and compare. This will then automatically copy the init values into the shadow values. For subsequent compare updates, during run-time, only the shadow registers must be used.

Figure 33-12. Capture Sequence for Delta Mode Time-stamp With Rising and Falling Edge Detect



33.3.4.1 Code Snippet for CAP Mode Delta Time, Rising and Falling Edge Triggers

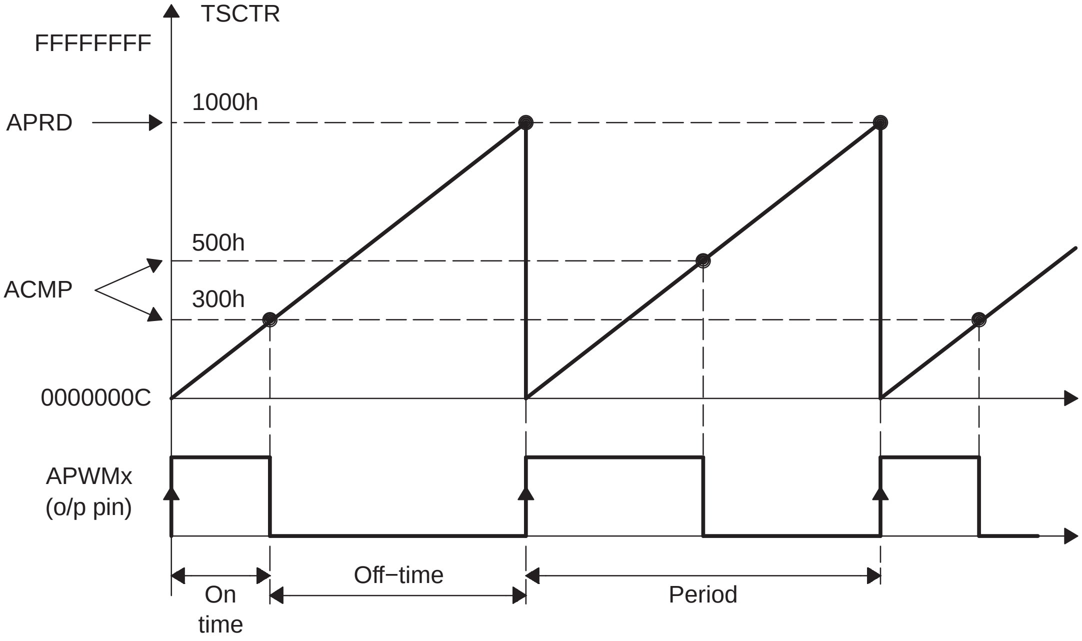
```
// Code snippet for CAP mode Delta Time, Rising and Falling edge triggers
 // Initialization Time
 //=======================
 // ECAP module 1 config
 ECap1Regs.ECCTL1.bit.CAP1POL = EC_RISING;
 ECap1Regs.ECCTL1.bit.CAP2POL = EC_FALLING;
 ECap1Regs.ECCTL1.bit.CAP3POL = EC_RISING;
 ECap1Regs.ECCTL1.bit.CAP4POL = EC_FALLING;
 ECap1Regs.ECCTL1.bit.CTRRST1 = EC_DELTA_MODE;
 ECap1Regs.ECCTL1.bit.CTRRST2 = EC_DELTA_MODE;
 ECap1Regs.ECCTL1.bit.CTRRST3 = EC_DELTA_MODE;
 ECap1Regs.ECCTL1.bit.CTRRST4 = EC_DELTA_MODE;
 ECap1Regs.ECCTL1.bit.CAPLDEN = EC_ENABLE;
 ECap1Regs.ECCTL1.bit.PRESCALE = EC_DIV1;
 ECap1Regs.ECCTL2.bit.CAP_APWM = EC_CAP_MODE;
 ECap1Regs.ECCTL2.bit.CONT_ONESHT = EC_CONTINUOUS;
 ECap1Regs.ECCTL2.bit.SYNCO_SEL = EC_SYNCO_DIS;
 ECap1Regs.ECCTL2.bit.SYNCI_EN = EC_DISABLE;
 ECap1Regs.ECCTL2.bit.TSCTRSTOP = EC_RUN; // Allow TSCTR to run
 // Run Time ( e.g. CEVT1 triggered ISR call)
 //==========================================
 // Note: here Time-stamp directly represents the Duty cycle values.
 DutyOnTime1 = ECap1Regs.CAP2; // Fetch Time-Stamp captured at T2
 DutyOffTime1 = ECap1Regs.CAP3; // Fetch Time-Stamp captured at T3
 DutyOnTime2 = ECap1Regs.CAP4; // Fetch Time-Stamp captured at T4
 DutyOffTime2 = ECap1Regs.CAP1; // Fetch Time-Stamp captured at T1
 Period1 = DutyOnTime1 + DutyOffTime1;
 Period2 = DutyOnTime2 + DutyOffTime2;
```

33.4 Application of the APWM Mode

In this section, the eCAP module is configured to operate as a PWM generator. Here a very simple single channel PWM waveform is generated from output pin APWMx. The PWM polarity is active high, which means that the compare value (CAP2 reg is now a compare register) represents the on-time (high level) of the period. Alternatively, if the APWMPOL bit is configured for active low, then the compare value represents the off-time. Note here values are in hexadecimal (“h”) notation.

33.4.1 Simple PWM Generation (Independent Channel/s)

Figure 33-13. PWM Waveform Details of APWM Mode Operation



33.4.1.1 Code Snippet for APWM Mode

```
// Code snippet for APWM mode Example 1
 // Initialization Time
 //=======================
 // ECAP module 1 config
 ECap1Regs.CAP1 = 0x1000; // Set period value
 ECap1Regs.CTRPHS = 0x0; // make phase zero
 ECap1Regs.ECCTL2.bit.CAP_APWM = EC_APWM_MODE;
 ECap1Regs.ECCTL2.bit.APWMPOL = EC_ACTV_HI; // Active high
 ECap1Regs.ECCTL2.bit.SYNCI_EN = EC_DISABLE; // Synch not used
 ECap1Regs.ECCTL2.bit.SYNCO_SEL = EC_SYNCO_DIS; // Synch not used
 ECap1Regs.ECCTL2.bit.TSCTRSTOP = EC_RUN; // Allow TSCTR to run
 // Run Time (Instant 1, e.g. ISR call)
 //======================
 ECap1Regs.CAP2 = 0x300; // Set Duty cycle i.e. compare value
 // Run Time (Instant 2, e.g. another ISR call)
 //======================
 ECap1Regs.CAP2 = 0x500; // Set Duty cycle i.e. compare value
```

33.5 eCAP Registers

Table 33-1 shows the eCAP module control and status registers. The base address for the control registers is FCF7 9300h for eCAP1, FCF7 9400h for eCAP2, FCF7 9500h for eCAP3, FCF7 9600h for eCAP4, FCF7 9700h for eCAP5, and FCF7 9800h for eCAP6.

Table 33-1. ECAP Control and Status Registers

Address Offset	Acronym	Description	Section
00h	TSCTR	Time-Stamp Counter Register	Section 33.5.1
04h	CTRPHS	Counter Phase Offset Value Register	Section 33.5.2
08h	CAP1	Capture 1 Register	Section 33.5.3
0Ch	CAP2	Capture 2 Register	Section 33.5.4
10h	CAP3	Capture 3 Register	Section 33.5.5
14h	CAP4	Capture 4 Register	Section 33.5.6
28h	ECCTL2	Capture Control Register 2	Section 33.5.7
2Ah	ECCTL1	Capture Control Register 1	Section 33.5.8
2Ch	ECFLG	Capture Interrupt Flag Register	Section 33.5.9
2Eh	ECEINT	Capture Interrupt Enable Register	Section 33.5.10
30h	ECFRC	Capture Interrupt Forcing Register	Section 33.5.11
32h	ECCLR	Capture Interrupt Clear Register	Section 33.5.12

33.5.1 Time-Stamp Counter Register (TSCTR)

Figure 33-14. Time-Stamp Counter Register (TSCTR) [offset = 00h]

31 … 0
TSCTR
R/W-0

LEGEND: R/W = Read/Write; -*n* = value after reset

Table 33-2. Time-Stamp Counter Register (TSCTR) Field Descriptions

Bits	Field	Description
31-0	TSCTR	Active 32-bit counter register that is used as the capture time-base.

33.5.2 Counter Phase Control Register (CTRPHS)

Figure 33-15. Counter Phase Control Register (CTRPHS) [offset = 04h]

31 … 0
CTRPHS
R/W-0

LEGEND: R/W = Read/Write; -*n* = value after reset

Table 33-3. Counter Phase Control Register (CTRPHS) Field Descriptions

Bits	Field	Description
31-0	CTRPHS	Counter phase value register that can be programmed for phase lag/lead. This register shadows TSCTR and is loaded into TSCTR upon either a SYNCI event or S/W force via a control bit. Used to achieve phase control synchronization with respect to other eCAP and EPWM time-bases.

33.5.3 Capture-1 Register (CAP1)

NOTE: In APWM mode, writing to CAP1/CAP2 active registers also writes the same value to the corresponding shadow registers CAP3/CAP4. This emulates immediate mode. Writing to the shadow registers CAP3/CAP4 invokes the shadow mode.

Figure 33-16. Capture-1 Register (CAP1) [offset = 08h]

31 ... 0
CAP1
R/W-0

LEGEND: R/W = Read/Write; -*n* = value after reset

Table 33-4. Capture-1 Register (CAP1) Field Descriptions

Bits	Field	Description
31-0	CAP1	This register can be loaded (written) by : • Time-Stamp (counter value) during a capture event • Software - may be useful for test purposes / initialization • APRD shadow register (CAP3) when used in APWM mode

33.5.4 Capture-2 Register (CAP2)

NOTE: In APWM mode, writing to CAP1/CAP2 active registers also writes the same value to the corresponding shadow registers CAP3/CAP4. This emulates immediate mode. Writing to the shadow registers CAP3/CAP4 invokes the shadow mode.

Figure 33-17. Capture-2 Register (CAP2) [offset = 0Ch]

31 ... 0
CAP2
R/W-0

LEGEND: R/W = Read/Write; -*n* = value after reset

Table 33-5. Capture-2 Register (CAP2) Field Descriptions

Bits	Field	Description
31-0	CAP2	This register can be loaded (written) by: • Time-Stamp (counter value) during a capture event • Software - may be useful for test purposes • APRD shadow register (CAP4) when used in APWM mode

33.5.5 Capture-3 Register (CAP3)

Figure 33-18. Capture-3 Register (CAP3) [offset = 10h]

31	0
CAP3	
R/W-0	

LEGEND: R/W = Read/Write; -*n* = value after reset

Table 33-6. Capture-3 Register (CAP3) Field Descriptions

Bits	Field	Description
31-0	CAP3	In CMP mode, this is a time-stamp capture register. In APWM mode, this is the period shadow (APRD) register. You update the PWM period value through this register. In this mode, CAP3 (APRD) shadows CAP1.

33.5.6 Capture-4 Register (CAP4)

Figure 33-19. Capture-4 Register (CAP4) [offset = 14h]

31	0
CAP4	
R/W-0	

LEGEND: R/W = Read/Write; -*n* = value after reset

Table 33-7. Capture-4 Register (CAP4) Field Descriptions

Bits	Field	Description
31-0	CAP4	In CMP mode, this is a time-stamp capture register. In APWM mode, this is the compare shadow (ACMP) register. You update the PWM compare value via this register. In this mode, CAP4 (ACMP) shadows CAP2.

33.5.7 ECAP Control Register 2 (ECCTL2)

Figure 33-20. ECAP Control Register 2 (ECCTL2) [offset = 28h]

15-11	10	9	8
Reserved	APWMPOL	CAP_APWM	SWSYNC
R-0	R/W-0	R/W-0	R/W-0

7-6	5	4	3	2-1	0
SYNCO_SEL	SYNCI_EN	TSCTRSTOP	REARM	STOP_WRAP	CONT_ONESHT
R/W-0	R/W-0	R/W-0	R/W-0	R/W-3h	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 33-8. ECAP Control Register 2 (ECCTL2) Field Descriptions

Bits	Field	Value	Description
15-11	Reserved	0	Reserved
10	APWMPOL		APWM output polarity select. This is applicable only in APWM operating mode.
		0	Output is active-high (Compare value defines high time).
		1	Output is active-low (Compare value defines low time).
9	CAP_APWM		CAP/APWM operating mode select.
		0	ECAP module operates in capture mode. This mode forces the following configuration: • Inhibits TSCTR resets via CTR = PRD event • Inhibits shadow loads on CAP1 and 2 registers • Permits user to enable CAP1-4 register load • CAPx/APWMx pin operates as a capture input
		1	ECAP module operates in APWM mode. This mode forces the following configuration: • Resets TSCTR on CTR = PRD event (period boundary • Permits shadow loading on CAP1 and 2 registers • Disables loading of time-stamps into CAP1-4 registers • CAPx/APWMx pin operates as a APWM output
8	SWSYNC		Software-forced Counter (TSCTR) Synchronizing. This provides a convenient software method to synchronize some or all ECAP time bases. In APWM mode, the synchronizing can also be done via the CTR = PRD event.
		0	Writing a 0 has no effect. Reading always returns a 0.
		1	Writing a 1 forces a TSCTR shadow load of current ECAP module and any ECAP modules down-stream providing the SYNCO_SEL bits are 0,0. After writing a 1, this bit returns to a 0. Note: Selection CTR = PRD is meaningful only in APWM mode; however, you can choose it in CAP mode if you find doing so useful.
7-6	SYNCO_SEL		Sync-Out select.
		0	Select sync-in event to be the sync-out signal (pass through).
		1h	Select CTR = PRD event to be the sync-out signal.
		2h	Disable sync out signal.
		3h	Disable sync out signal.
5	SYNCI_EN		Counter (TSCTR) Sync-In select mode.
		0	Disable sync-in option.
		1	Enable counter (TSCTR) to be loaded from CTRPHS register upon either a SYNCI signal or a S/W force event.
4	TSCTRSTOP		Time Stamp (TSCTR) Counter Stop (freeze) Control.
		0	TSCTR is stopped.
		1	TSCTR is free-running.

Table 33-8. ECAP Control Register 2 (ECCTL2) Field Descriptions (continued)

Bits	Field	Value	Description
3	REARM		One-Shot Re-Arming Control, wait for stop trigger. Note: The re-arm function is valid in one shot or continuous mode.
		0	Has no effect (reading always returns a 0).
		1	Arms the one-shot sequence as follows: 1) Resets the Mod4 counter to 0 2) Unfreezes the Mod4 counter 3) Enables capture register loads
2-1	STOP_WRAP		Stop value for one-shot mode. This is the number (between 1-4) of captures allowed to occur before the CAP(1-4) registers are frozen, capture sequence is stopped. Wrap value for continuous mode. This is the number (between 1-4) of the capture register in which the circular buffer wraps around and starts again.
		0	Stop after Capture Event 1 in one-shot mode. Wrap after Capture Event 1 in continuous mode.
		1h	Stop after Capture Event 2 in one-shot mode. Wrap after Capture Event 2 in continuous mode.
		2h	Stop after Capture Event 3 in one-shot mode. Wrap after Capture Event 3 in continuous mode.
		3h	Stop after Capture Event 4 in one-shot mode. Wrap after Capture Event 4 in continuous mode.
			Notes: STOP_WRAP is compared to Mod4 counter and, when equal, 2 actions occur: • Mod4 counter is stopped (frozen) • Capture register loads are inhibited In one-shot mode, further interrupt events are blocked until re-armed.
0	CONT_ONESHT		Continuous or one-shot mode control (applicable only in capture mode).
		0	Operate in continuous mode.
		1	Operate in one-shot mode.

33.5.8 ECAP Control Regiser 1 (ECCTL1)

Figure 33-21. ECAP Control Register 1 (ECCTL1) [offset = 2Ah]

15	14	13–9	8
FREE	SOFT	PRESCALE	CAPLDEN
R/W-0	R/W-0	R/W-0	R/W-0

7	6	5	4	3	2	1	0
CTRRST4	CAP4POL	CTRRST3	CAP3POL	CTRRST2	CAP2POL	CTRRST1	CAP1POL
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

LEGEND: R/W = Read/Write; -*n* = value after reset

Table 33-9. ECAP Control Register 1 (ECCTL1) Field Descriptions

Bits	Field	Value	Description
15-14	FREE/SOFT		Emulation Control.
		0	TSCTR counter stops immediately on emulation suspend.
		1h	TSCTR counter runs until = 0.
		2h-3h	TSCTR counter is unaffected by emulation suspend (Run Free).
13-9	PRESCALE		Event Filter prescale select.
		0	Divide by 1 (no prescale, by-pass the prescaler).
		1h	Divide by 2.
		2h	Divide by 4.
		3h	Divide by 6.
		4h	Divide by 8.
		5h	Divide by 10.
		:	:
		1Eh	Divide by 60.
		1Fh	Divide by 62.
8	CAPLDEN		Enable Loading of CAP1-4 registers on a capture event.
		0	Disable CAP1-4 register loads at capture event time.
		1	Enable CAP1-4 register loads at capture event time.
7	CTRRST4		Counter Reset on Capture Event 4.
		0	Do not reset counter on Capture Event 4 (absolute time stamp operation).
		1	Reset counter after Capture Event 4 time-stamp has been captured (used in difference mode operation).
6	CAP4POL		Capture Event 4 Polarity select.
		0	Capture Event 4 triggered on a rising edge (RE).
		1	Capture Event 4 triggered on a falling edge (FE).
5	CTRRST3		Counter Reset on Capture Event 3.
		0	Do not reset counter on Capture Event 3 (absolute time stamp).
		1	Reset counter after Event 3 time-stamp has been captured (used in difference mode operation).
4	CAP3POL		Capture Event 3 Polarity select.
		0	Capture Event 3 triggered on a rising edge (RE).
		1	Capture Event 3 triggered on a falling edge (FE).
3	CTRRST2		Counter Reset on Capture Event 2.
		0	Do not reset counter on Capture Event 2 (absolute time stamp).
		1	Reset counter after Event 2 time-stamp has been captured (used in difference mode operation).

Table 33-9. ECAP Control Register 1 (ECCTL1) Field Descriptions (continued)

Bits	Field	Value	Description
2	CAP2POL		Capture Event 2 Polarity select.
		0	Capture Event 2 triggered on a rising edge (RE).
		1	Capture Event 2 triggered on a falling edge (FE).
1	CTRRST1		Counter Reset on Capture Event 1.
		0	Do not reset counter on Capture Event 1 (absolute time stamp).
		1	Reset counter after Event 1 time-stamp has been captured (used in difference mode operation).
0	CAP1POL		Capture Event 1 Polarity select.
		0	Capture Event 1 triggered on a rising edge (RE).
		1	Capture Event 1 triggered on a falling edge (FE).

33.5.9 ECAP Interrupt Flag Register (ECFLG)

Figure 33-22. ECAP Interrupt Flag Register (ECFLG) [offset = 2Ch]

15	8
Reserved	
R-0	

7	6	5	4	3	2	1	0
CTR_CMP	CTR_PRD	CTROVF	CEVT4	CETV3	CEVT2	CETV1	INT
R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0

LEGEND: R = Read only; -*n* = value after reset

Table 33-10. ECAP Interrupt Flag Register (ECFLG) Field Descriptions

Bits	Field	Value	Description
15-8	Reserved	0	Reserved
7	CTR_CMP		Compare Equal Compare Status Flag. This flag is active only in APWM mode.
		0	Indicates no event occurred.
		1	Indicates the counter (TSCTR) reached the compare register value (ACMP).
6	CTR_PRD		Counter Equal Period Status Flag. This flag is only active in APWM mode.
		0	Indicates no event occurred.
		1	Indicates the counter (TSCTR) reached the period register value (APRD) and was reset.
5	CTROVF		Counter Overflow Status Flag. This flag is active in CAP and APWM mode.
		0	Indicates no event occurred.
		1	Indicates the counter (TSCTR) has made the transition from FFFF FFFFh to 0000 0000h.
4	CEVT4		Capture Event 4 Status Flag This flag is only active in CAP mode.
		0	Indicates no event occurred.
		1	Indicates the fourth event occurred at ECAPx pin.
3	CEVT3		Capture Event 3 Status Flag. This flag is active only in CAP mode.
		0	Indicates no event occurred.
		1	Indicates the third event occurred at ECAPx pin.
2	CEVT2		Capture Event 2 Status Flag. This flag is only active in CAP mode.
		0	Indicates no event occurred.
		1	Indicates the second event occurred at ECAPx pin.
1	CEVT1		Capture Event 1 Status Flag. This flag is only active in CAP mode.
		0	Indicates no event occurred.
		1	Indicates the first event occurred at ECAPx pin.
0	INT		Global Interrupt Status Flag.
		0	Indicates no interrupt generated.
		1	Indicates that an interrupt was generated.

33.5.10 ECAP Interrupt Enable Register (ECEINT)

The interrupt enable bits block any of the selected events from generating an interrupt. Events will still be latched into the flag bit (ECFLG register) and can be forced/cleared via the ECFRC/ECCLR registers.

The proper procedure for configuring peripheral modes and interrupts is as follows:

- Disable global interrupts
- Stop eCAP counter
- Disable eCAP interrupts
- Configure peripheral registers
- Clear spurious eCAP interrupt flags
- Enable eCAP interrupts
- Start eCAP counter
- Enable global interrupts

Figure 33-23. ECAP Interrupt Enable Register (ECEINT) [offset = 2Eh]

15	8
Reserved	
R-0	

7	6	5	4	3	2	1	0
CTR_CMP	CTR_PRD	CTROVF	CEVT4	CEVT3	CEVT2	CETV1	Reserved
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 33-11. ECAP Interrupt Enable Register (ECEINT) Field Descriptions

Bits	Field	Value	Description
15-8	Reserved	0	Reserved
7	CTR_CMP		Counter Equal Compare Interrupt Enable.
		0	Disable Compare Equal as an Interrupt source.
		1	Enable Compare Equal as an Interrupt source.
6	CTR_PRD		Counter Equal Period Interrupt Enable.
		0	Disable Period Equal as an Interrupt source.
		1	Enable Period Equal as an Interrupt source.
5	CTROVF		Counter Overflow Interrupt Enable.
		0	Disabled counter Overflow as an Interrupt source.
		1	Enable counter Overflow as an Interrupt source.
4	CEVT4		Capture Event 4 Interrupt Enable.
		0	Disable Capture Event 4 as an Interrupt source.
		1	Capture Event 4 Interrupt Enable.
3	CEVT3		Capture Event 3 Interrupt Enable.
		0	Disable Capture Event 3 as an Interrupt source.
		1	Enable Capture Event 3 as an Interrupt source.
2	CEVT2		Capture Event 2 Interrupt Enable.
		0	Disable Capture Event 2 as an Interrupt source.
		1	Enable Capture Event 2 as an Interrupt source.
1	CEVT1		Capture Event 1 Interrupt Enable.
		0	Disable Capture Event 1 as an Interrupt source.
		1	Enable Capture Event 1 as an Interrupt source.
0	Reserved	0	Reserved

33.5.11 ECAP Interrupt Forcing Register (ECFRC)

Figure 33-24. ECAP Interrupt Forcing Register (ECFRC) [offset = 30h]

15	8
Reserved	
R-0	

7	6	5	4	3	2	1	0
CTR_CMP	CTR_PRD	CTROVF	CEVT4	CETV3	CETV2	CETV1	Reserved
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 33-12. ECAP Interrupt Forcing Register (ECFRC) Field Descriptions

Bits	Field	Value	Description
15-8	Reserved	0	Any writes to these bit(s) must always have a value of 0.
7	CTR_CMP		Force Counter Equal Compare Interrupt.
		0	No effect. Always reads back a 0.
		1	Writing a 1 sets the CTR_CMP flag bit.
6	CTR_PRD		Force Counter Equal Period Interrupt.
		0	No effect. Always reads back a 0.
		1	Writing a 1 sets the CTR_PRD flag bit.
5	CTROVF		Force Counter Overflow.
		0	No effect. Always reads back a 0.
		1	Writing a 1 to this bit sets the CTROVF flag bit.
4	CEVT4		Force Capture Event 4.
		0	No effect. Always reads back a 0.
		1	Writing a 1 sets the CEVT4 flag bit.
3	CEVT3		Force Capture Event 3.
		0	No effect. Always reads back a 0.
		1	Writing a 1 sets the CEVT3 flag bit.
2	CEVT2		Force Capture Event 2.
		0	No effect. Always reads back a 0.
		1	Writing a 1 sets the CEVT2 flag bit.
1	CEVT1		Force Capture Event 1.
		1	No effect. Always reads back a 0.
		0	Sets the CEVT1 flag bit.
0	Reserved	0	Any writes to these bit(s) must always have a value of 0.

33.5.12 ECAP Interrupt Clear Register (ECCLR)

Figure 33-25. ECAP Interrupt Clear Register (ECCLR) [offset = 32h]

15							8
Reserved							
R-0							

7	6	5	4	3	2	1	0
CTR_CMP	CTR_PRD	CTROVF	CEVT4	CETV3	CETV2	CETV1	INT
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 33-13. ECAP Interrupt Clear Register (ECCLR) Field Descriptions

Bits	Field	Value	Description
15-8	Reserved	0	Any writes to these bit(s) must always have a value of 0.
7	CTR_CMP		Counter Equal Compare Status Flag.
		0	Writing a 0 has no effect. Always reads back a 0.
		1	Writing a 1 clears the CTR_CMP flag condition.
6	CTR_PRD		Counter Equal Period Status Flag.
		0	Writing a 0 has no effect. Always reads back a 0.
		1	Writing a 1 clears the CTR_PRD flag condition.
5	CTROVF		Counter Overflow Status Flag.
		0	Writing a 0 has no effect. Always reads back a 0.
		1	Writing a 1 clears the CTROVF flag condition.
4	CEVT4		Capture Event 4 Status Flag.
		0	Writing a 0 has no effect. Always reads back a 0.
		1	Writing a 1 clears the CEVT4 flag condition.
3	CEVT3		Capture Event 3 Status Flag.
		0	Writing a 0 has no effect. Always reads back a 0.
		1	Writing a 1 clears the CEVT3 flag condition.
2	CEVT2		Capture Event 2 Status Flag.
		1	Writing a 0 has no effect. Always reads back a 0.
		0	Writing a 1 clears the CEVT2 flag condition.
1	CEVT1		Capture Event 1 Status Flag.
		0	Writing a 0 has no effect. Always reads back a 0.
		1	Writing a 1 clears the CEVT1 flag condition.
0	INT		Global Interrupt Clear Flag.
		0	Writing a 0 has no effect. Always reads back a 0.
		1	Writing a 1 clears the INT flag and enable further interrupts to be generated if any of the event flags are set to 1.

Chapter 34
SPNU563A–March 2018

Enhanced Quadrature Encoder Pulse (eQEP) Module

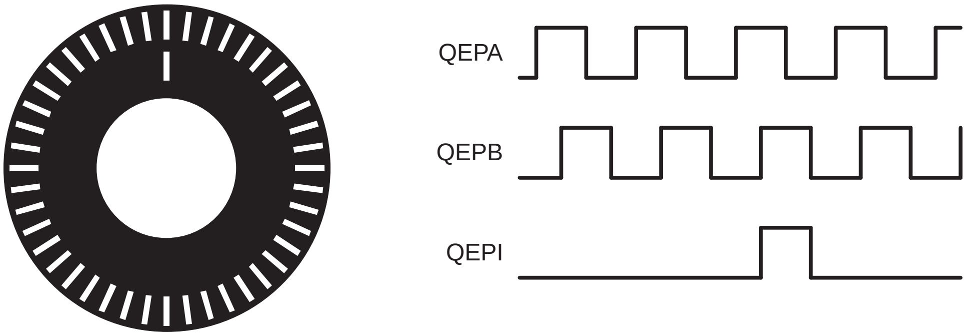
The enhanced quadrature encoder pulse (eQEP) module is used for direct interface with a linear or rotary incremental encoder to get position, direction, and speed information from a rotating machine for use in a high-performance motion and position-control system. This microcontroller implements 2 instances of the eQEP module.

Topic **Page**

34.1 Introduction

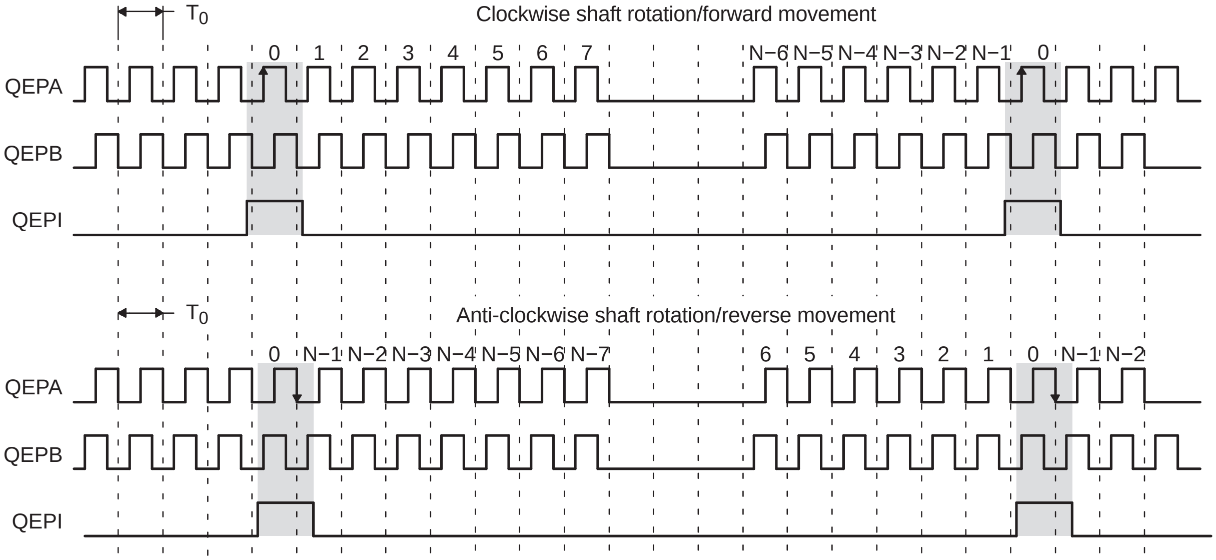
A single track of slots patterns the periphery of an incremental encoder disk, as shown in Figure 34-1. These slots create an alternating pattern of dark and light lines. The disk count is defined as the number of dark/light line pairs that occur per revolution (lines per revolution). As a rule, a second track is added to generate a signal that occurs once per revolution (index signal: QEPI), which can be used to indicate an absolute position. Encoder manufacturers identify the index pulse using different terms such as index, marker, home position, and zero reference.

Figure 34-1. Optical Encoder Disk



To derive direction information, the lines on the disk are read out by two different photo-elements that "look" at the disk pattern with a mechanical shift of 1/4 the pitch of a line pair between them. This shift is realized with a reticle or mask that restricts the view of the photo-element to the desired part of the disk lines. As the disk rotates, the two photo-elements generate signals that are shifted 90° out of phase from each other. These are commonly called the quadrature QEPA and QEPB signals. The clockwise direction for most encoders is defined as the QEPA channel going positive before the QEPB channel and vise versa as shown in Figure 34-2.

Figure 34-2. QEP Encoder Output Signal for Forward/Reverse Movement

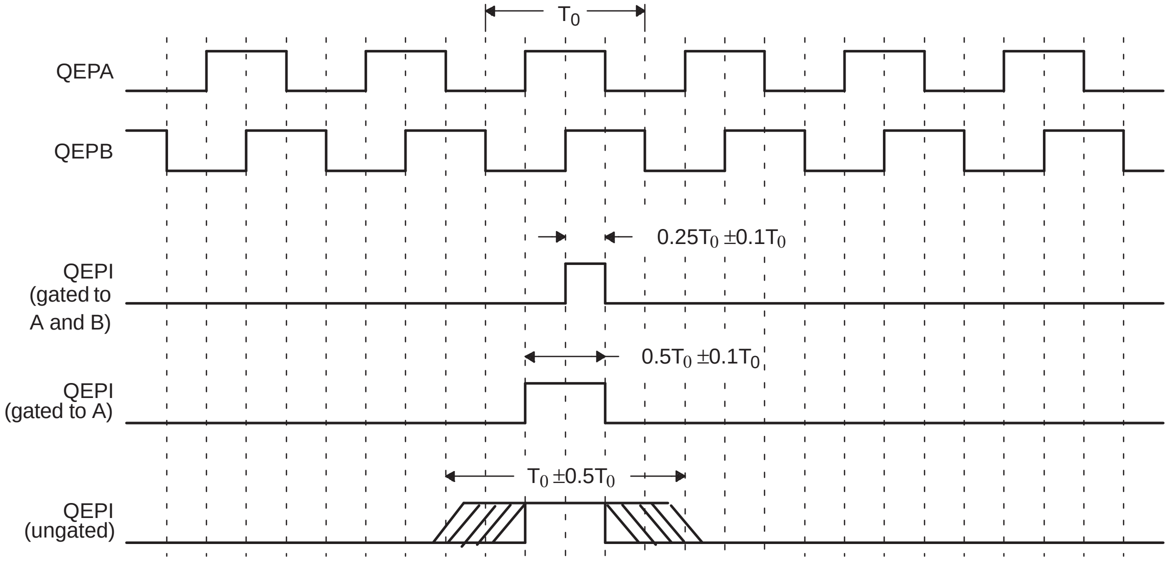


Legend: N = lines per revolution

The encoder wheel typically makes one revolution for every revolution of the motor or the wheel may be at a geared rotation ratio with respect to the motor. Therefore, the frequency of the digital signal coming from the QEPA and QEPB outputs varies proportionally with the velocity of the motor. For example, a 2000-line encoder directly coupled to a motor running at 5000 revolutions per minute (rpm) results in a frequency of 166.6 KHz, so by measuring the frequency of either the QEPA or QEPB output, the processor can determine the velocity of the motor.

Quadrature encoders from different manufacturers come with two forms of index pulse (gated index pulse or ungated index pulse) as shown in Figure 34-3. A nonstandard form of index pulse is ungated. In the ungated configuration, the index edges are not necessarily coincident with A and B signals. The gated index pulse is aligned to any of the four quadrature edges and width of the index pulse and can be equal to a quarter, half, or full period of the quadrature signal.

Figure 34-3. Index Pulse Example



Some typical applications of shaft encoders include robotics and even computer input in the form of a mouse. Inside your mouse you can see where the mouse ball spins a pair of axles (a left/right, and an up/down axle). These axles are connected to optical shaft encoders that effectively tell the computer how fast and in what direction the mouse is moving.

General Issues: Estimating velocity from a digital position sensor is a cost-effective strategy in motor control. Two different first order approximations for velocity may be written as:

$$v(k) \approx \frac{x(k) - x(k-1)}{T} = \frac{\Delta X}{T} \tag{68}$$

$$v(k) \approx \frac{X}{t(k) - t(k-1)} = \frac{X}{\Delta T} \tag{69}$$

where

v(k): Velocity at time instant k

x(k): Position at time instant k

x(k-1): Position at time instant k-1

T: Fixed unit time or inverse of velocity calculation rate

ΔX: Incremental position movement in unit time

t(k): Time instant "k"

t(k-1): Time instant "k-1"

X: Fixed unit position

ΔT: Incremental time elapsed for unit position movement.

Equation 68 is the conventional approach to velocity estimation and it requires a time base to provide unit time event for velocity calculation. Unit time is basically the inverse of the velocity calculation rate.

The encoder count (position) is read once during each unit time event. The quantity [x(k) - x(k-1)] is formed by subtracting the previous reading from the current reading. Then the velocity estimate is computed by multiplying by the known constant 1/T (where T is the constant time between unit time events and is known in advance).

Estimation based on Equation 68 has an inherent accuracy limit directly related to the resolution of the position sensor and the unit time period T. For example, consider a 500-line per revolution quadrature encoder with a velocity calculation rate of 400 Hz. When used for position the quadrature encoder gives a four-fold increase in resolution, in this case, 2000 counts per revolution. The minimum rotation that can be detected is therefore 0.0005 revolutions, which gives a velocity resolution of 12 rpm when sampled at 400 Hz. While this resolution may be satisfactory at moderate or high speeds, e.g. 1% error at 1200 rpm, it would clearly prove inadequate at low speeds. In fact, at speeds below 12 rpm, the speed estimate would erroneously be zero much of the time.

At low speed, Equation 69 provides a more accurate approach. It requires a position sensor that outputs a fixed interval pulse train, such as the aforementioned quadrature encoder. The width of each pulse is defined by motor speed for a given sensor resolution. Equation 69 can be used to calculate motor speed by measuring the elapsed time between successive quadrature pulse edges. However, this method suffers from the opposite limitation, as does Equation 68. A combination of relatively large motor speeds and high sensor resolution makes the time interval ΔT small, and thus more greatly influenced by the timer resolution. This can introduce considerable error into high-speed estimates.

For systems with a large speed range (that is, speed estimation is needed at both low and high speeds), one approach is to use Equation 69 at low speed and have the DSP software switch over to Equation 68 when the motor speed rises above some specified threshold.

34.2 Basic Operation

34.2.1 EQEP Inputs

The eQEP inputs include two pins for quadrature-clock mode or direction-count mode, an index (or 0 marker), and a strobe input.

- *QEPA/XCLK and QEPB/XDIR*

  These two pins can be used in quadrature-clock mode or direction-count mode.

  – *Quadrature-clock Mode*

    The eQEP encoders provide two square wave signals (A and B) 90 electrical degrees out of phase whose phase relationship is used to determine the direction of rotation of the input shaft and number of eQEP pulses from the index position to derive the relative position information. For forward or clockwise rotation, QEPA signal leads QEPB signal and vice versa. The quadrature decoder uses these two inputs to generate quadrature-clock and direction signals.

  – *Direction-count Mode*

    In direction-count mode, direction and clock signals are provided directly from the external source. Some position encoders have this type of output instead of quadrature output. The QEPA pin provides the clock input and the QEPB pin provides the direction input.

- *eQEPI: Index or Zero Marker*

  The eQEP encoder uses an index signal to assign an absolute start position from which position information is incrementally encoded using quadrature pulses. This pin is connected to the index output of the eQEP encoder to optionally reset the position counter for each revolution. This signal can be used to initialize or latch the position counter on the occurrence of a desired event on the index pin.

- QEPS: *Strobe Input*

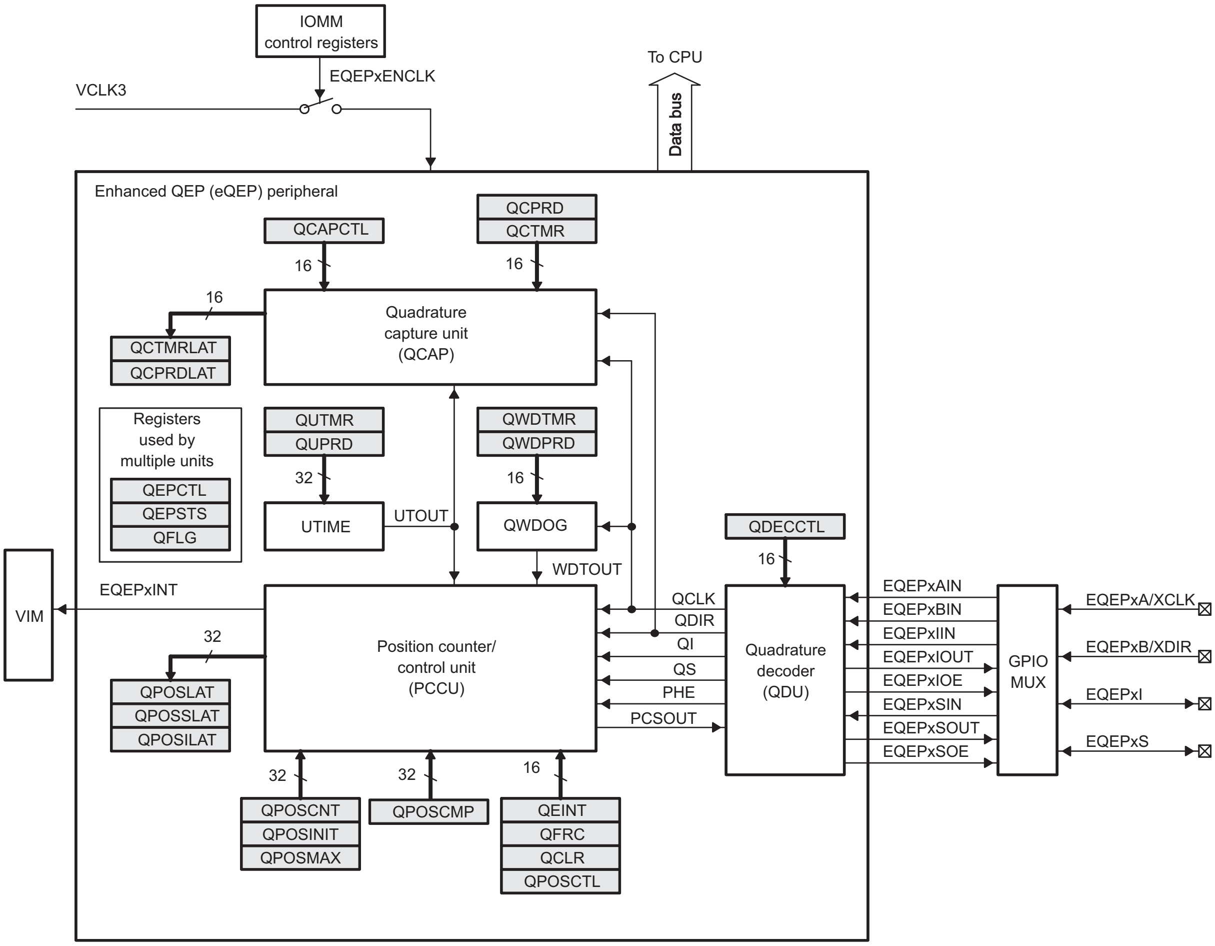
  This general-purpose strobe signal can initialize or latch the position counter on the occurrence of a desired event on the strobe pin. This signal is typically connected to a sensor or limit switch to notify that the motor has reached a defined position.

34.2.2 Functional Description

The eQEP peripheral contains the following major functional units (as shown in Figure 34-4):

- Programmable input qualification for each pin (part of the GPIO MUX)
- Quadrature decoder unit (QDU)
- Position counter and control unit for position measurement (PCCU)
- Quadrature edge-capture unit for low-speed measurement (QCAP)
- Unit time base for speed/frequency measurement (UTIME)
- Watchdog timer for detecting stalls (QWDOG)

Figure 34-4. Functional Block Diagram of the eQEP Peripheral



34.2.2.1 eQEP Memory Map

Table 34-1 lists the registers with their memory locations, sizes, and reset values.

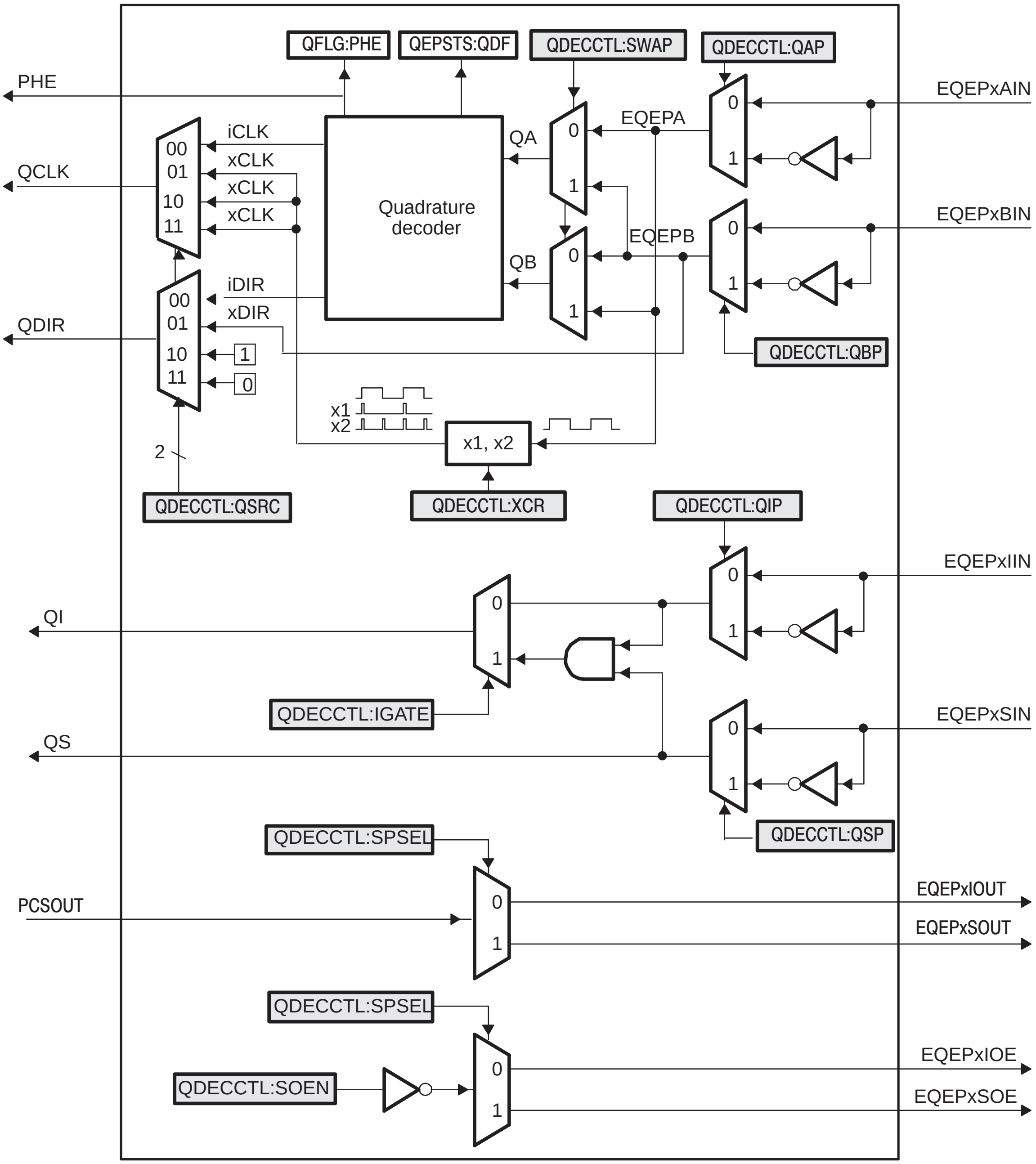
Table 34-1. EQEP Memory Map

Name	Address Offset	Size(x16)/ #shadow	Reset	Register Description
QPOSCNT	0x00	2/0	0x00000000	eQEP Position Counter Register
QPOSINIT	0x04	2/0	0x00000000	eQEP Initialization Position Count Register
QPOSMAX	0x08	2/0	0x00000000	eQEP Maximum Position Count Register
QPOSCMP	0x0C	2/1	0x00000000	eQEP Position-Compare Register
QPOSILAT	0x10	2/0	0x00000000	eQEP Index Position Latch Register
QPOSSLAT	0x14	2/0	0x00000000	eQEP Strobe Position Latch Register
QPOSLAT	0x18	2/0	0x00000000	eQEP Position Latch Register
QUTMR	0x1C	2/0	0x00000000	eQEP Unit Timer Register
QUPRD	0x20	2/0	0x00000000	eQEP Unit Period Register
QWDPRD	0x24	1/0	0x0000	eQEP Watchdog Period Register
QWDTMR	0x26	1/0	0x0000	eQEP Watchdog Timer Register
QEPCTL	0x28	1/0	0x0000	eQEP Control Register
QDECCTL	0x2A	1/0	0x0000	eQEP Decoder Control Register
QPOSCTL	0x2C	1/0	0x00000	eQEP Position-Compare Control Register
QCAPCTL	0x2E	1/0	0x0000	eQEP Capture Control Register
QFLG	0x30	1/0	0x0000	eQEP Interrupt Flag Register
QEINT	0x32	1/0	0x0000	eQEP Interrupt Enable Register
QFRC	0x34	1/0	0x0000	eQEP Interrupt Force Register
QCLR	0x36	1/0	0x0000	eQEP Interrupt Clear Register
QCTMR	0x38	1/0	0x0000	eQEP Capture Timer Register
QEPSTS	0x3A	1/0	0x0000	eQEP Status Register
QCTMRLAT	0x3C	1/0	0x0000	eQEP Capture Timer Latch Register
QCPRD	0x3E	1/0	0x0000	eQEP Capture Period Register
Reserved	0x40	–	–	Reserved
QCPRDLAT	0x42	1/0	0x0000	eQEP Capture Period Latch Register

34.2.2.2 Quadrature Decoder Unit (QDU)

Figure 34-5 shows a functional block diagram of the QDU.

Figure 34-5. Functional Block Diagram of Decoder Unit



34.2.2.2.1 Position Counter Input Modes

Clock and direction input to position counter is selected using QDECCTL[QSRC] bits, based on interface input requirement as follows:

- Quadrature-count mode
- Direction-count mode
- UP-count mode
- DOWN-count mode

34.2.2.2.1.1 Quadrature Count Mode

The quadrature decoder generates the direction and clock to the position counter in quadrature count mode.

Direction Decoding— The direction decoding logic of the eQEP circuit determines which one of the sequences (QEPA, QEPB) is the leading sequence and accordingly updates the direction information in QEPSTS[QDF] bit. Table 34-2 and Figure 34-6 show the direction decoding logic in truth table and state machine form. Both edges of the QEPA and QEPB signals are sensed to generate count pulses for the position counter. Therefore, the frequency of the clock generated by the eQEP logic is four times that of each input sequence. Figure 34-7 shows the direction decoding and clock generation from the eQEP input signals.

Table 34-2. Quadrature Decoder Truth Table

Previous Edge	Present Edge	QDIR	QPOSCNT
QA↑	QB↑	UP	Increment
	QB↓	DOWN	Decrement
	QA↓	TOGGLE	Increment or Decrement
QA↓	QB↓	UP	Increment
	QB↑	DOWN	Decrement
	QA↑	TOGGLE	Increment or Decrement
QB↑	QA↑	DOWN	Increment
	QA↓	UP	Decrement
	QB↓	TOGGLE	Increment or Decrement
QB↓	QA↓	DOWN	Increment
	QA↑	UP	Decrement
	QB↑	TOGGLE	Increment or Decrement

Figure 34-6. Quadrature Decoder State Machine

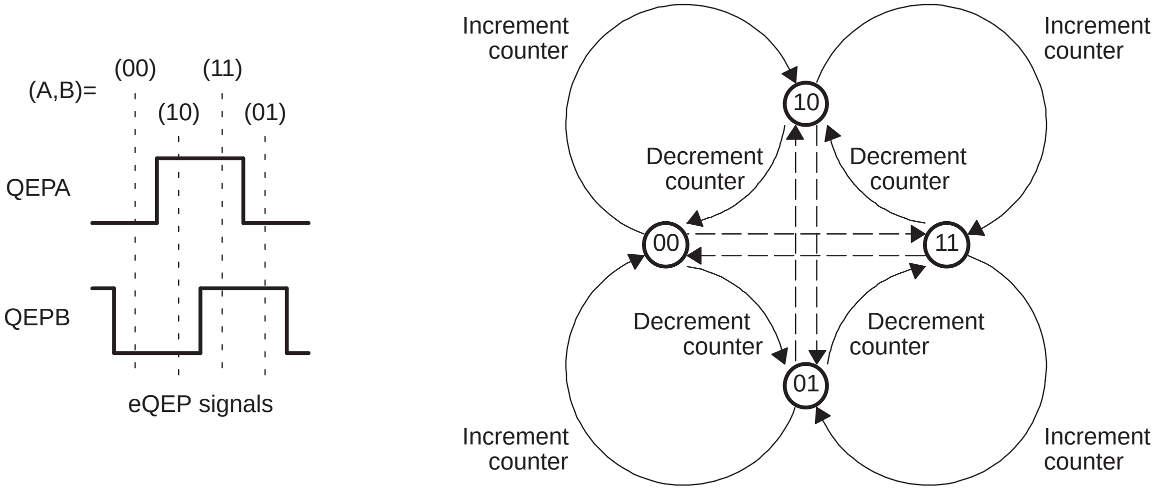
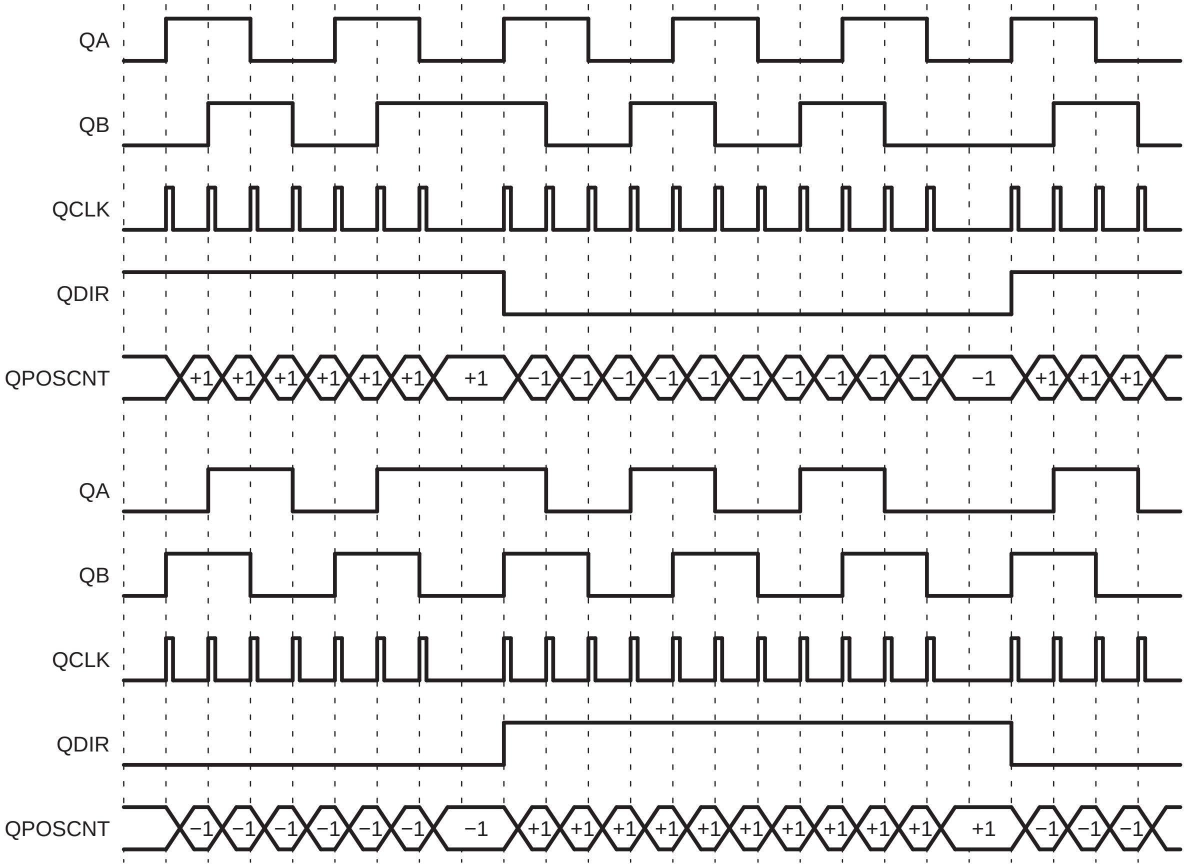


Figure 34-7. Quadrature-clock and Direction Decoding



Phase Error Flag— In normal operating conditions, quadrature inputs QEPA and QEPB will be 90 degrees out of phase. The phase error flag (PHE) is set in the QFLG register when edge transition is detected simultaneously on the QEPA and QEPB signals to optionally generate interrupts. State transitions marked by dashed lines in Figure 34-6 are invalid transitions that generate a phase error.

Count Multiplication— The eQEP position counter provides 4x times the resolution of an input clock by generating a quadrature-clock (QCLK) on the rising/falling edges of both eQEP input clocks (QEPA and QEPB) as shown in Figure 34-7.

Reverse Count— In normal quadrature count operation, QEPA input is fed to the QA input of the quadrature decoder and the QEPB input is fed to the QB input of the quadrature decoder. Reverse counting is enabled by setting the SWAP bit in the QDECCTL register. This will swap the input to the quadrature decoder thereby reversing the counting direction.

34.2.2.2.1.2 Direction-count Mode

Some position encoders provide direction and clock outputs, instead of quadrature outputs. In such cases, direction-count mode can be used. QEPA input will provide the clock for position counter and the QEPB input will have the direction information. The position counter is incremented on every rising edge of a QEPA input when the direction input is high and decremented when the direction input is low.

34.2.2.2.1.3 Up-Count Mode

The counter direction signal is hard-wired for up count and the position counter is used to measure the frequency of the QEPA input. Setting of the QDECCTL[XCR] bit enables clock generation to the position counter on both edges of the QEPA input, thereby increasing the measurement resolution by 2x factor.

34.2.2.2.1.4 Down-Count Mode

The counter direction signal is hardwired for a down count and the position counter is used to measure the frequency of the QEPA input. Setting of the QDECCTL[XCR] bit enables clock generation to the position counter on both edges of a QEPA input, thereby increasing the measurement resolution by 2x factor.

34.2.2.2.2 eQEP Input Polarity Selection

Each eQEP input can be inverted using QDECCTL[8:5] control bits. As an example, setting of QDECCTL[QIP] bit will invert the index input.

34.2.2.2.3 Position-Compare Sync Output

The enhanced eQEP peripheral includes a position-compare unit that is used to generate the position-compare sync signal on compare match between the position counter register (QPOSCNT) and the position-compare register (QPOSCMP). This sync signal can be output using an index pin or strobe pin of the EQEP peripheral.

Setting the QDECCTL[SOEN] bit enables the position-compare sync output and the QDECCTL[SPSEL] bit selects either an eQEP index pin or an eQEP strobe pin.

34.2.2.3 Position Counter and Control Unit (PCCU)

The position counter and control unit provides two configuration registers (QEPCTL and QPOSCTL) for setting up position counter operational modes, position counter initialization/latch modes and position-compare logic for sync signal generation.

34.2.2.3.1 Position Counter Operating Modes

Position counter data may be captured in different manners. In some systems, the position counter is accumulated continuously for multiple revolutions and the position counter value provides the position information with respect to the known reference. An example of this is the quadrature encoder mounted on the motor controlling the print head in the printer. Here the position counter is reset by moving the print head to the home position and then position counter provides absolute position information with respect to home position.

In other systems, the position counter is reset on every revolution using index pulse and position counter provides rotor angle with respect to index pulse position.

Position counter can be configured to operate in following four modes

- Position Counter Reset on Index Event
- Position Counter Reset on Maximum Position
- Position Counter Reset on the first Index Event
- Position Counter Reset on Unit Time Out Event (Frequency Measurement)

In all the above operating modes, position counter is reset to 0 on overflow and to QPOSMAX register value on underflow. Overflow occurs when the position counter counts up after QPOSMAX value. Underflow occurs when position counter counts down after "0". Interrupt flag is set to indicate overflow/underflow in QFLG register.

34.2.2.3.1.1 Position Counter Reset on Index Event (QEPCTL[PCRM] = 00)

If the index event occurs during the forward movement, then position counter is reset to 0 on the next eQEP clock. If the index event occurs during the reverse movement, then the position counter is reset to the value in the QPOSMAX register on the next eQEP clock.

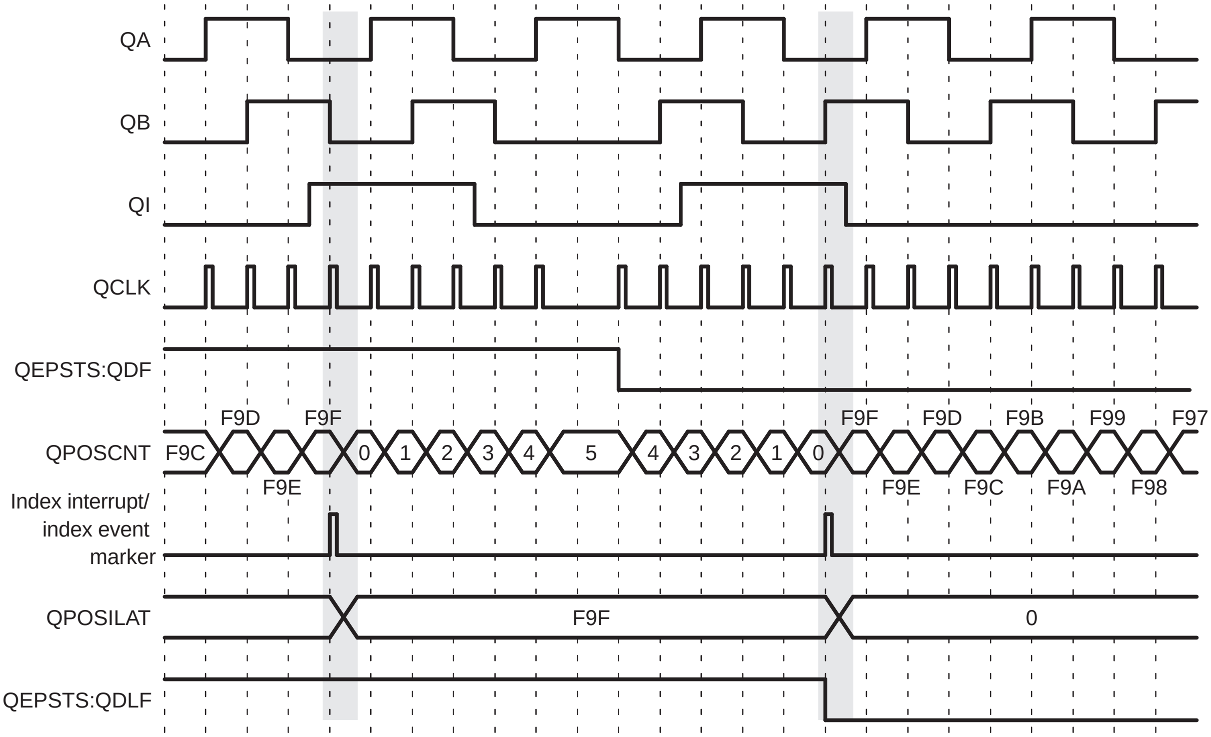
First index marker is defined as the quadrature edge following the first index edge. The eQEP peripheral records the occurrence of the first index marker (QEPSTS[FIMF]) and direction on the first index event marker (QEPSTS[FIDF]) in QEPSTS registers, it also remembers the quadrature edge on the first index marker so that same relative quadrature transition is used for index event reset operation.

For example, if the first reset operation occurs on the falling edge of QEPB during the forward direction, then all the subsequent reset must be aligned with the falling edge of QEPB for the forward rotation and on the rising edge of QEPB for the reverse rotation as shown in Figure 34-8.

The position-counter value is latched to the QPOSILAT register and direction information is recorded in the QEPSTS[QDLF] bit on every index event marker. The position-counter error flag (QEPSTS[PCEF]) and error interrupt flag (QFLG[PCE]) are set if the latched value is not equal to 0 or QPOSMAX. The position-counter error flag (QEPSTS[PCEF]) is updated on every index event marker and an interrupt flag (QFLG[PCE]) will be set on error that can be cleared only through software.

The index event latch configuration QEPCTL[IEL] bits are ignored in this mode and position counter error flag/interrupt flag are generated only in index event reset mode.

Figure 34-8. Position Counter Reset by Index Pulse for 1000 Line Encoder (QPOSMAX = 3999 or F9Fh)

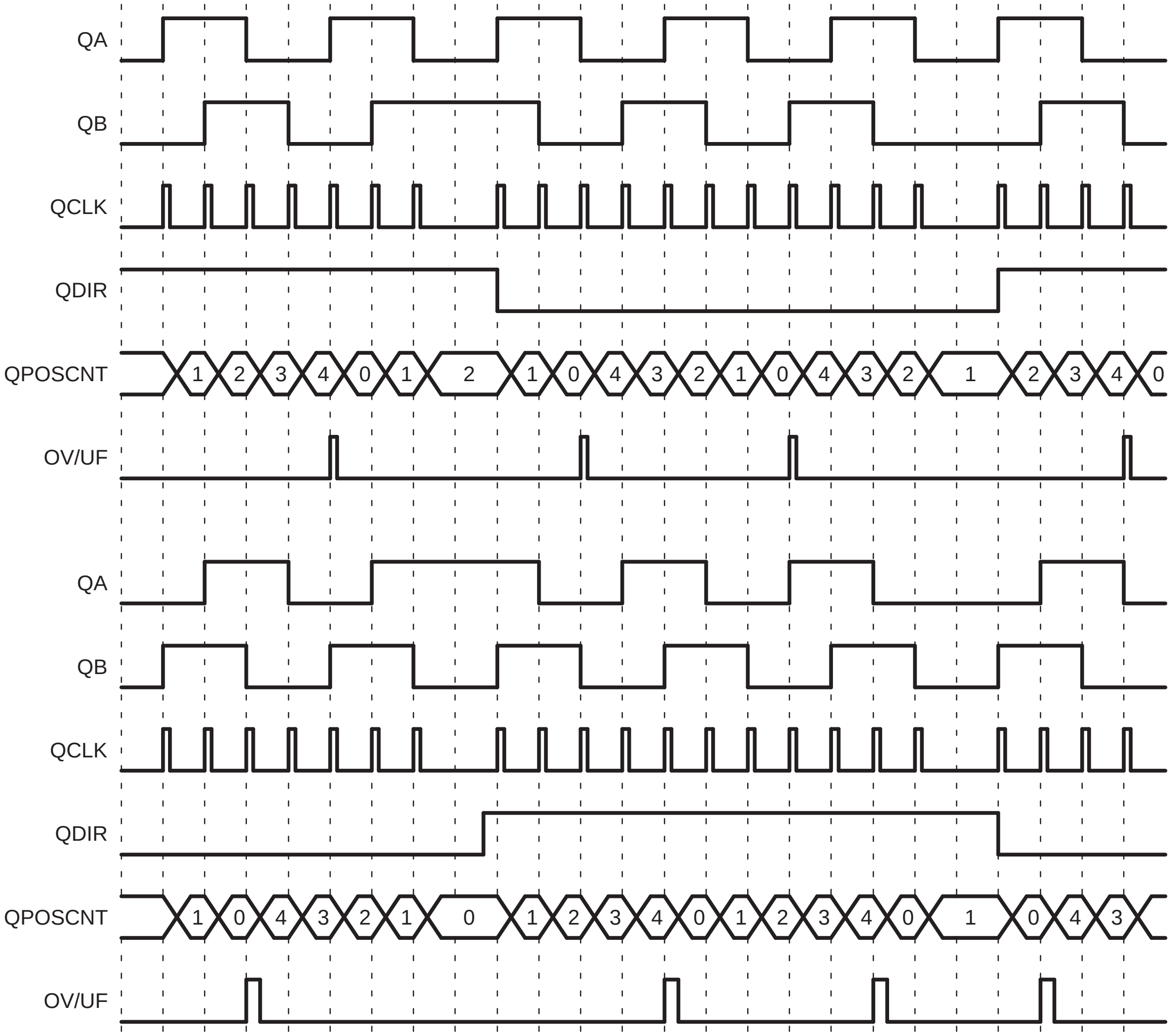


34.2.2.3.1.2 Position Counter Reset on Maximum Position (QEPCTL[PCRM] = 01)

If the position counter is equal to QPOSMAX, then the position counter is reset to 0 on the next eQEP clock for forward movement and position counter overflow flag is set. If the position counter is equal to ZERO, then the position counter is reset to QPOSMAX on the next QEP clock for reverse movement and position counter underflow flag is set. Figure 34-9 shows the position counter reset operation in this mode.

First index marker is defined as the quadrature edge following the first index edge. The eQEP peripheral records the occurrence of the first index marker (QEPSTS[FIMF]) and direction on the first index event marker (QEPSTS[FIDF]) in the QEPSTS registers; it also remembers the quadrature edge on the first index marker so that the same relative quadrature transition is used for the software index marker (QEPCTL[IEL]=11).

Figure 34-9. Position Counter Underflow/Overflow (QPOSMAX = 4)



34.2.2.3.1.3 Position Counter Reset on the First Index Event (QEPCTL[PCRM] = 10)

If the index event occurs during forward movement, then the position counter is reset to 0 on the next eQEP clock. If the index event occurs during the reverse movement, then the position counter is reset to the value in the QPOSMAX register on the next eQEP clock. Note that this is done only on the first occurrence and subsequently the position counter value is not reset on an index event; rather, it is reset based on maximum position as described in Section 34.2.2.3.1.2.

First index marker is defined as the quadrature edge following the first index edge. The eQEP peripheral records the occurrence of the first index marker (QEPSTS[FIMF]) and direction on the first index event marker (QEPSTS[FIDF]) in QEPSTS registers, it also remembers the quadrature edge on the first index marker so that same relative quadrature transition is used for software index marker (QEPCTL[IEL]=11).

34.2.2.3.1.4 Position Counter Reset on Unit Time out Event (QEPCTL[PCRM] = 11)

In this mode, the QPOSCNT value is latched to the QPOSLAT register and then the QPOSCNT is reset (to 0 or QPOSMAX, depending on the direction mode selected by QDECCTL[QSRC] bits on a unit time event). This is useful for frequency measurement.

34.2.2.3.2 Position Counter Latch

The eQEP index and strobe input can be configured to latch the position counter (QPOSCNT) into QPOSILAT and QPOSSLAT, respectively, on occurrence of a definite event on these pins.

34.2.2.3.2.1 Index Event Latch

In some applications, it may not be desirable to reset the position counter on every index event and instead it may be required to operate the position counter in full 32-bit mode (QEPCTL[PCRM] = 01 and QEPCTL[PCRM] = 10 modes).

In such cases, the eQEP position counter can be configured to latch on the following events and direction information is recorded in the QEPSTS[QDLF] bit on every index event marker.

- Latch on Rising edge (QEPCTL[IEL]=01)
- Latch on Falling edge (QEPCTL[IEL]=10)
- Latch on Index Event Marker (QEPCTL[IEL]=11)

This is particularly useful as an error checking mechanism to check if the position counter accumulated the correct number of counts between index events. As an example, the 1000-line encoder must count 4000 times when moving in the same direction between the index events.

The index event latch interrupt flag (QFLG[IEL]) is set when the position counter is latched to the QPOSILAT register. The index event latch configuration bits (QEPCTZ[IEL]) are ignored when QEPCTL[PCRM] = 00.

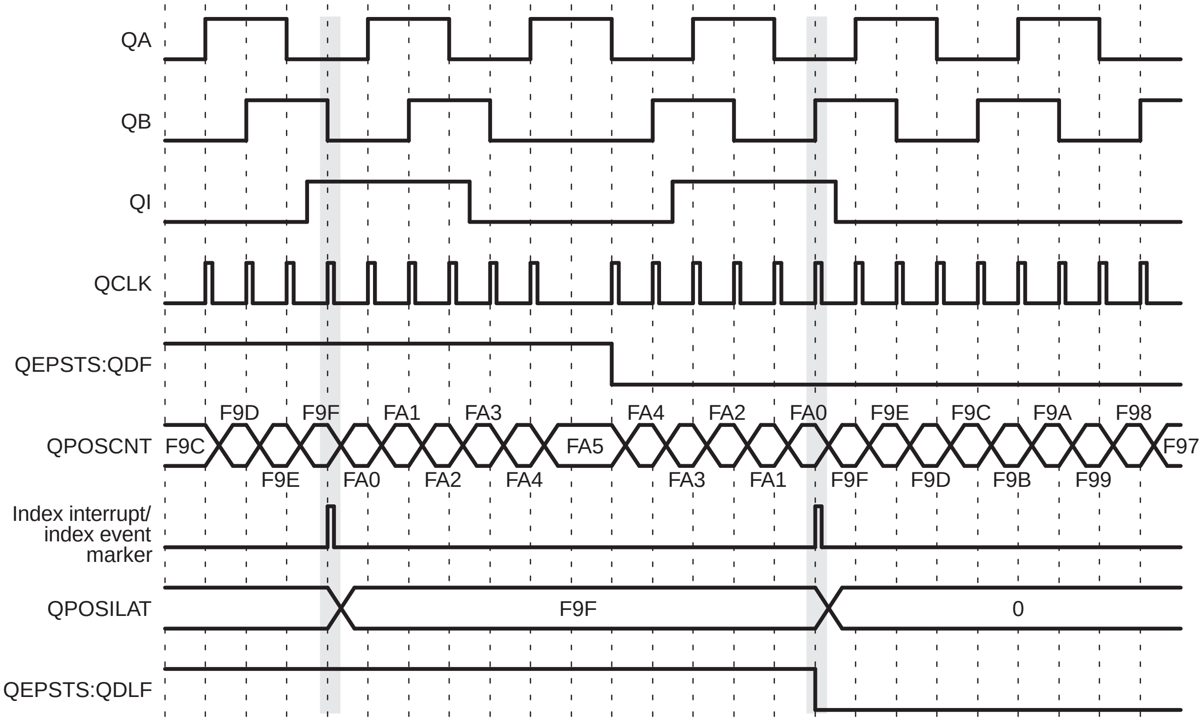
Latch on Rising Edge (QEPCTL[IEL]=01)— The position counter value (QPOSCNT) is latched to the QPOSILAT register on every rising edge of an index input.

Latch on Falling Edge (QEPCTL[IEL] = 10)— The position counter value (QPOSCNT) is latched to the QPOSILAT register on every falling edge of index input.

Latch on Index Event Marker/Software Index Marker (QEPCTL[IEL] = 11— The first index marker is defined as the quadrature edge following the first index edge. The eQEP peripheral records the occurrence of the first index marker (QEPSTS[FIMF]) and direction on the first index event marker (QEPSTS[FIDF]) in the QEPSTS registers. It also remembers the quadrature edge on the first index marker so that same relative quadrature transition is used for latching the position counter (QEPCTL[IEL]=11).

Figure 34-10 shows the position counter latch using an index event marker.

Figure 34-10. Software Index Marker for 1000-line Encoder (QEPCTL[IEL] = 1)



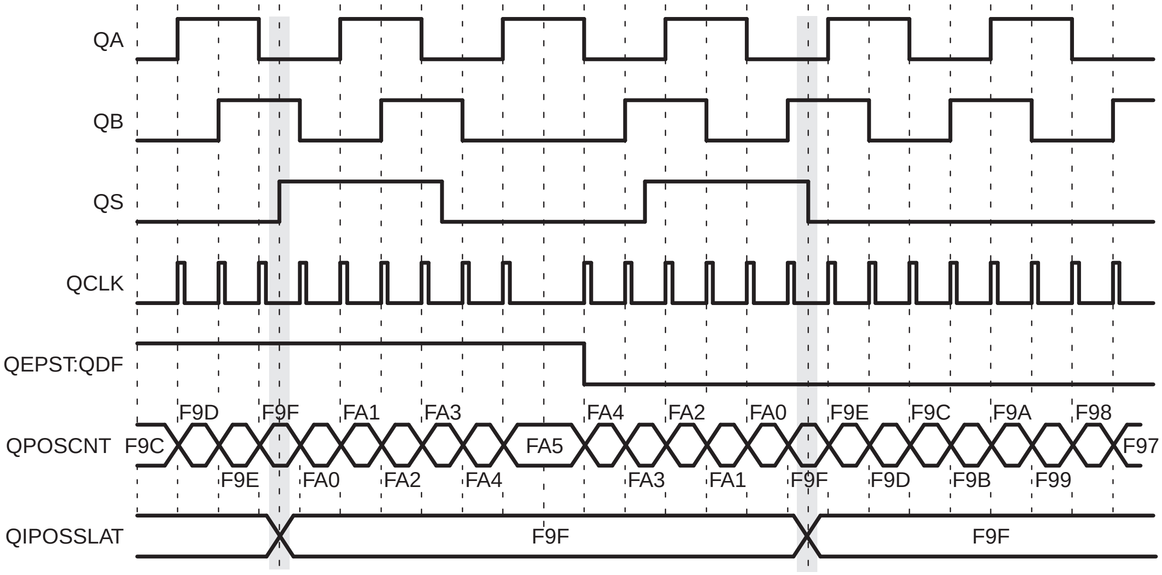
34.2.2.3.2.2 Strobe Event Latch

The position-counter value is latched to the QPOSSLAT register on the rising edge of the strobe input by clearing the QEPCTL[SEL] bit.

If the QEPCTL[SEL] bit is set, then the position counter value is latched to the QPOSSLAT register on the rising edge of the strobe input for forward direction and on the falling edge of the strobe input for reverse direction as shown in Figure 34-11.

The strobe event latch interrupt flag (QFLG[SEL) is set when the position counter is latched to the QPOSSLAT register.

Figure 34-11. Strobe Event Latch (QEPCTL[SEL] = 1)



34.2.2.3.3 Position Counter Initialization

The position counter can be initialized using following events:

- Index event
- Strobe event
- Software initialization

Index Event Initialization (IEI)— The QEPI index input can be used to trigger the initialization of the position counter at the rising or falling edge of the index input. If the QEPCTL[IEI] bits are 10, then the position counter (QPOSCNT) is initialized with a value in the QPOSINIT register on the rising edge of index input. Conversely, if the QEPCTL[IEI] bits are 11, initialization will be on the falling edge of the index input.

Strobe Event Initialization (SEI)— If the QEPCTL[SEI] bits are 10, then the position counter is initialized with a value in the QPOSINIT register on the rising edge of strobe input.

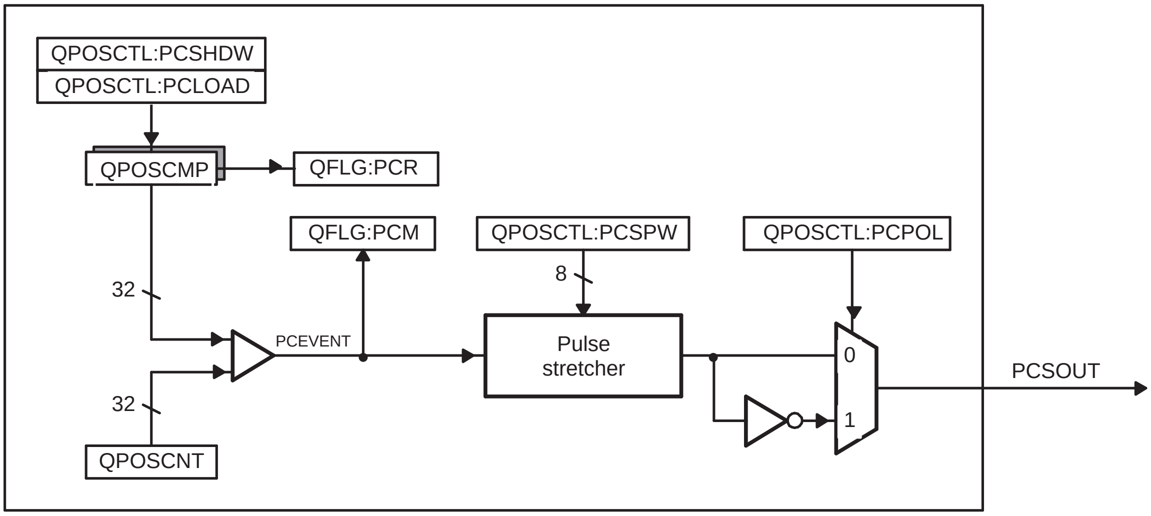
If QEPCTL[SEL] bits are 11, then the position counter is initialized with a value in the QPOSINIT register on the rising edge of strobe input for forward direction and on the falling edge of strobe input for reverse direction.

Software Initialization (SWI)— The position counter can be initialized in software by writing a 1 to the QEPCTL[SWI] bit. This bit is not automatically cleared. While the bit is still set, if a 1 is written to it again, the position counter will be re-initialized.

34.2.2.3.4 eQEP Position-compare Unit

The eQEP peripheral includes a position-compare unit that is used to generate a sync output and/or interrupt on a position-compare match. Figure 34-12 shows a diagram. The position-compare (QPOSCMP) register is shadowed and shadow mode can be enabled or disabled using the QPOSCTL[PSSHDW] bit. If the shadow mode is not enabled, the CPU writes directly to the active position compare register.

Figure 34-12. eQEP Position-compare Unit



In shadow mode, you can configure the position-compare unit (QPOSCTL[PCLOAD]) to load the shadow register value into the active register on the following events and to generate the position-compare ready (QFLG[PCR]) interrupt after loading.

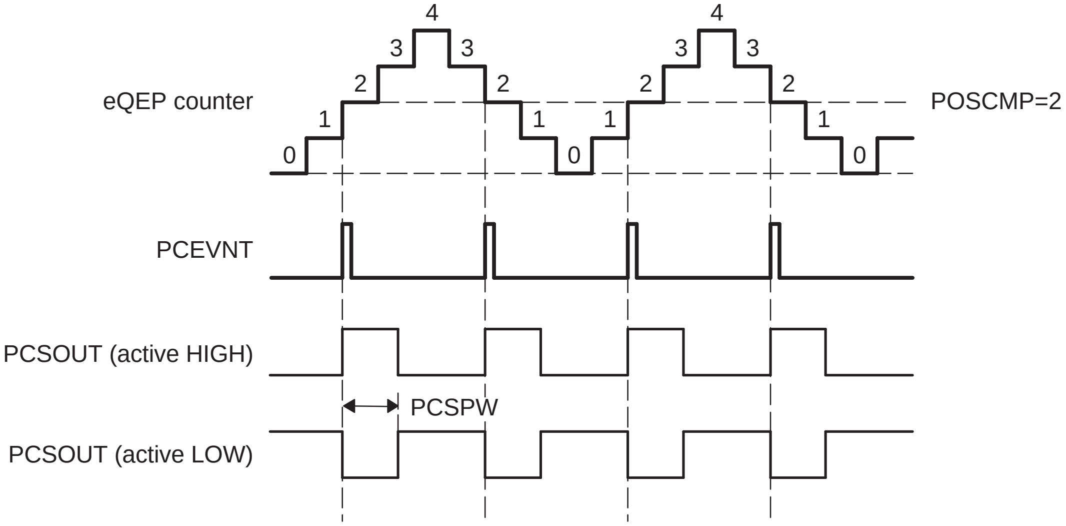
- Load on compare match
- Load on position-counter zero event

The position-compare match (QFLG[PCM]) is set when the position-counter value (QPOSCNT) matches with the active position-compare register (QPOSCMP) and the position-compare sync output of the programmable pulse width is generated on compare match to trigger an external device.

For example, if QPOSCMP = 2, the position-compare unit generates a position-compare event on 1 to 2 transitions of the eQEP position counter for forward counting direction and on 3 to 2 transitions of the eQEP position counter for reverse counting direction (see Figure 34-13).

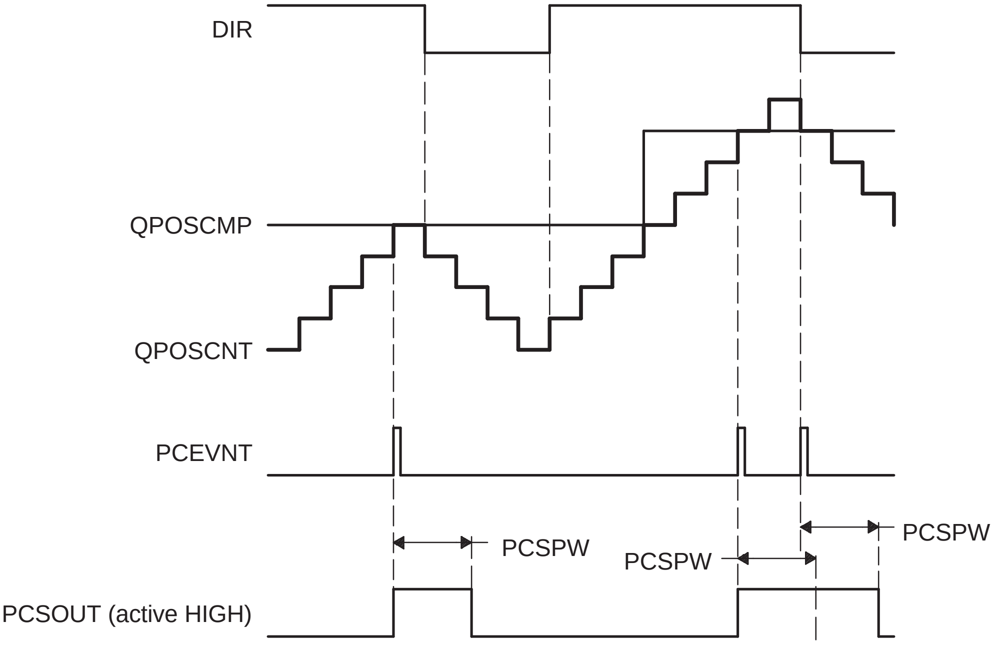
Section 34.3.14 shows the layout of the eQEP Position-Compare Control Register (QPOSCTL) and describes the QPOSCTL bit fields.

Figure 34-13. eQEP Position-compare Event Generation Points



The pulse stretcher logic in the position-compare unit generates a programmable position-compare sync pulse output on the position-compare match. In the event of a new position-compare match while a previous position-compare pulse is still active, then the pulse stretcher generates a pulse of specified duration from the new position-compare event as shown in Figure 34-14.

Figure 34-14. eQEP Position-compare Sync Output Pulse Stretcher



34.2.2.4 eQEP Edge Capture Unit

The eQEP peripheral includes an integrated edge capture unit to measure the elapsed time between the unit position events as shown in Figure 34-15. This feature is typically used for low speed measurement using the following equation:

$$v(k) = \frac{X}{t(k) - t(k-1)} = \frac{X}{\Delta T} \tag{70}$$

where,

- X - Unit position is defined by integer multiple of quadrature edges (see Figure 34-16)
- ΔT - Elapsed time between unit position events
- v(k) - Velocity at time instant "k"

The eQEP capture timer (QCTMR) runs from prescaled VCLK3 and the prescaler is programmed by the QCAPCTL[CCPS] bits. The capture timer (QCTMR) value is latched into the capture period register (QCPRD) on every unit position event and then the capture timer is reset, a flag is set in QEPSTS:UPEVNT to indicate that new value is latched into the QCPRD register. Software can check this status flag before reading the period register for low speed measurement and clear the flag by writing 1.

Time measurement (ΔT) between unit position events will be correct if the following conditions are met:

- No more than 65,535 counts have occurred between unit position events.
- No direction change between unit position events.

The capture unit sets the eQEP overflow error flag (QEPSTS[COEF]) in the event of capture timer overflow between unit position events. If a direction change occurs between the unit position events, then an error flag is set in the status register (QEPSTS[CDEF]).

Capture Timer (QCTMR) and Capture period register (QCPRD) can be configured to latch on following events.

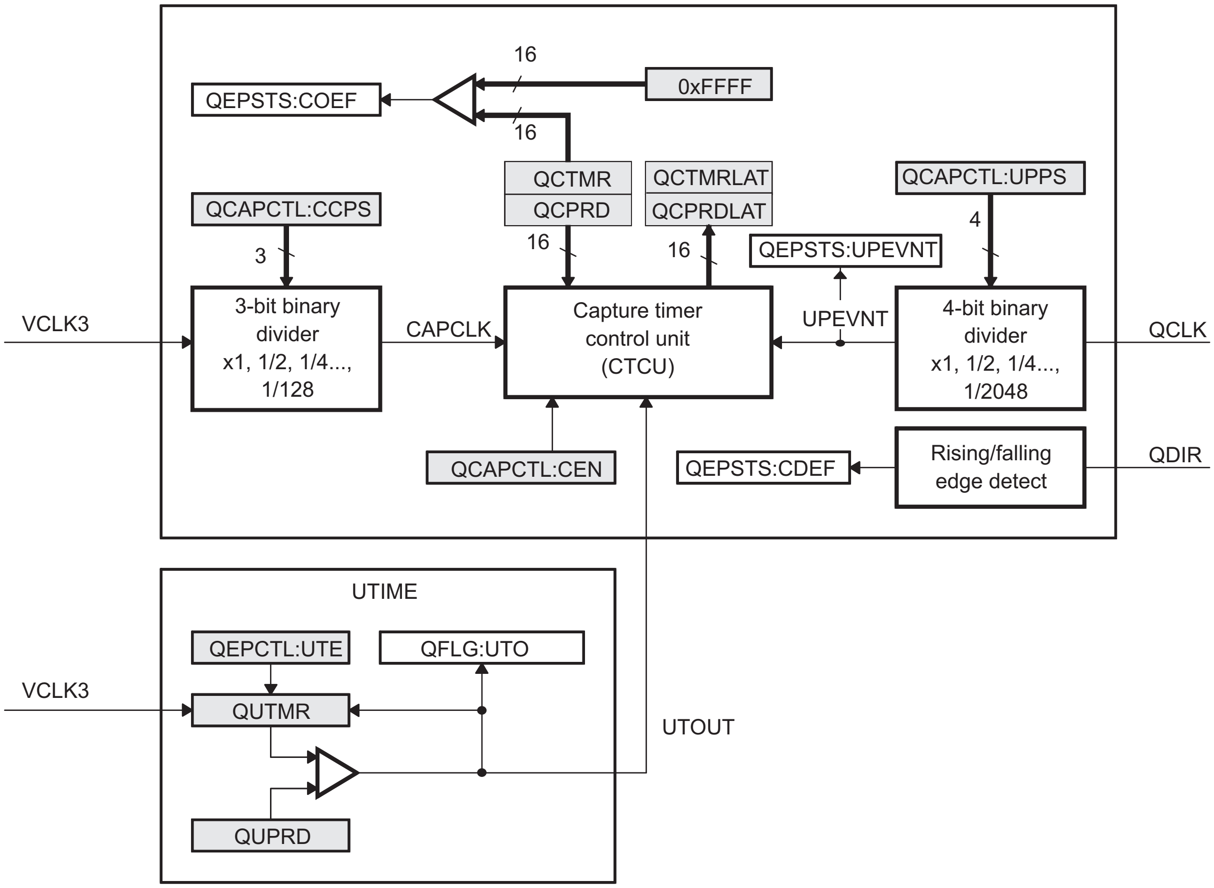
- CPU read of QPOSCNT register
- Unit time-out event

If the QEPCTL[QCLM] bit is cleared, then the capture timer and capture period values are latched into the QCTMRLAT and QCPRDLAT registers, respectively, when the CPU reads the position counter (QPOSCNT).

If the QEPCTL[QCLM] bit is set, then the position counter, capture timer, and capture period values are latched into the QPOSLAT, QCTMRLAT and QCPRDLAT registers, respectively, on unit time out.

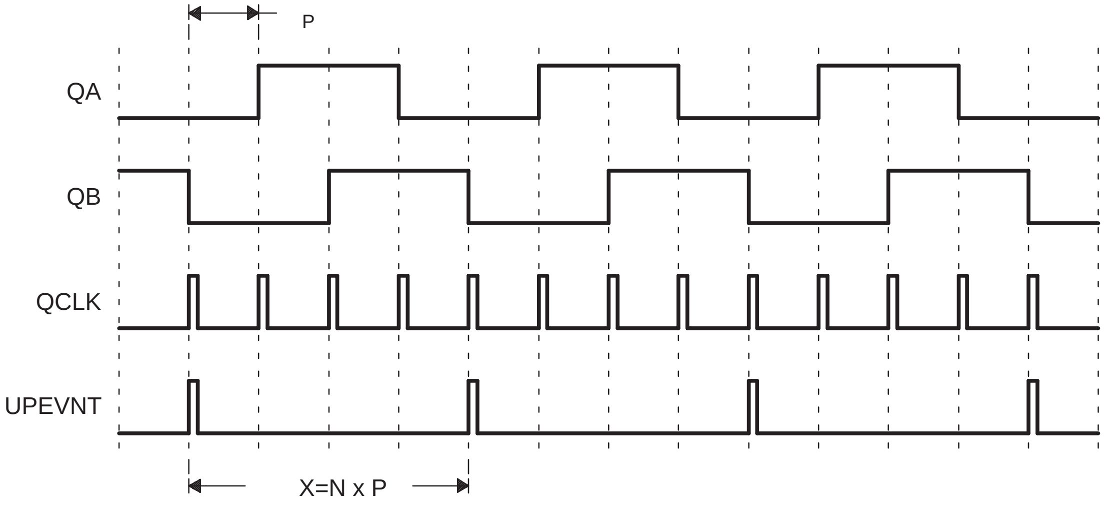
Figure 34-17 shows the capture unit operation along with the position counter.

Figure 34-15. eQEP Edge Capture Unit



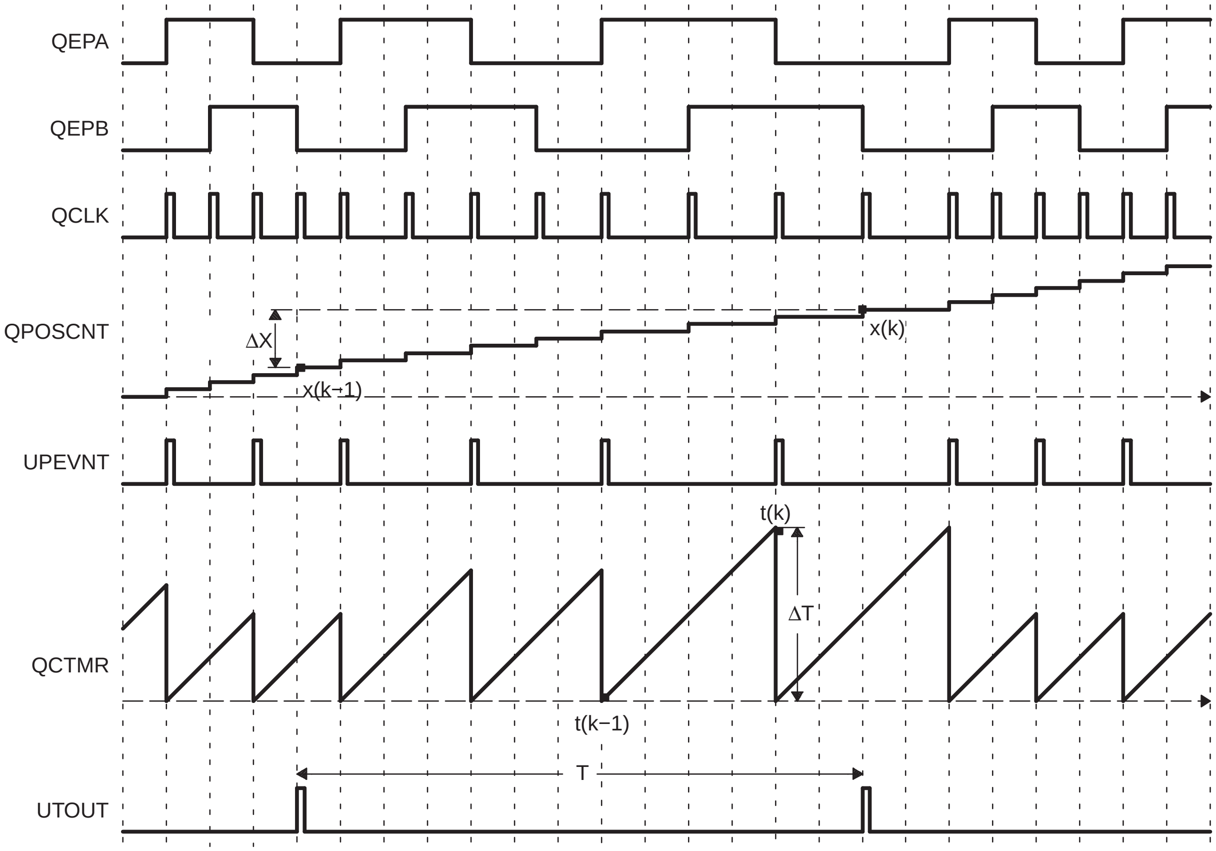
NOTE: The QCAPCTL[UPPS] prescaler should not be modified dynamically (such as switching the unit event prescaler from QCLK/4 to QCLK/8). Doing so may result in undefined behavior. The QCAPCTL[CPPS] prescaler can be modified dynamically (such as switching CAPCLK prescaling mode from SYSCLK/4 to SYSCLK/8) only after the capture unit is disabled.

Figure 34-16. Unit Position Event for Low Speed Measurement (QCAPCTL[UPPS] = 0010)



A N - Number of quadrature periods selected using QCAPCTL[UPPS] bits

Figure 34-17. eQEP Edge Capture Unit - Timing Details



Velocity Calculation Equations:

$$v(k) = \frac{x(k) - x(k-1)}{T} = \frac{\Delta X}{T} \text{ o} \tag{71}$$

where

v(k): Velocity at time instant k

x(k): Position at time instant k

x(k-1): Position at time instant k-1

T: Fixed unit time or inverse of velocity calculation rate

ΔX: Incremental position movement in unit time

X: Fixed unit position

ΔT: Incremental time elapsed for unit position movement

t(k): Time instant "k"

t(k-1): Time instant "k-1"

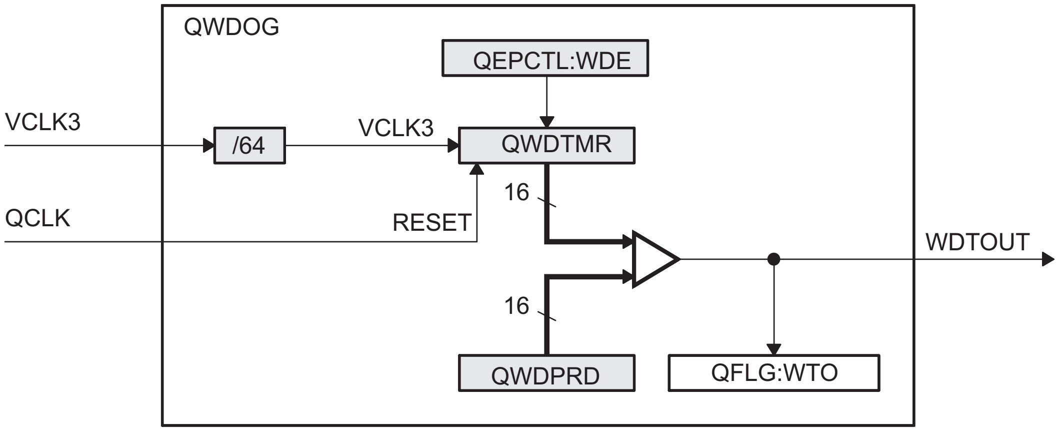
Unit time (T) and unit period(X) are configured using the QUPRD and QCAPCTL[UPPS] registers. Incremental position output and incremental time output is available in the QPOSLAT and QCPRDLAT registers.

Parameter	Relevant Register to Configure or Read the Information
T	Unit Period Register (QUPRD)
ΔX	Incremental Position = QPOSLAT(k) - QPOSLAT(K-1)
X	Fixed unit position defined by sensor resolution and ZCAPCTL[UPPS] bits
ΔT	Capture Period Latch (QCPRDLAT)

34.2.3 eQEP Watchdog

The eQEP peripheral contains a 16-bit watchdog timer that monitors the quadrature-clock to indicate proper operation of the motion-control system. The eQEP watchdog timer is clocked from VCLK3/64 and the quadrate clock event (pulse) resets the watchdog timer. If no quadrature-clock event is detected until a period match (QWDPRD = QWDTMR), then the watchdog timer will time out and the watchdog interrupt flag will be set (QFLG[WTO]). The time-out value is programmable through the watchdog period register (QWDPRD).

Figure 34-18. eQEP Watchdog Timer

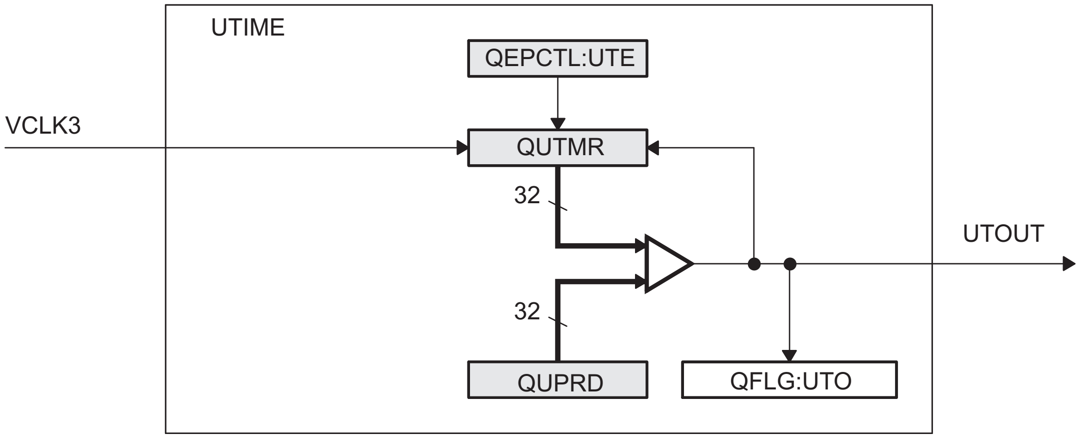


34.2.4 Unit Timer Base

The eQEP peripheral includes a 32-bit timer (QUTMR) that is clocked by VCLK3 to generate periodic interrupts for velocity calculations. The unit time out interrupt is set (QFLG[UTO]) when the unit timer (QUTMR) matches the unit period register (QUPRD).

The eQEP peripheral can be configured to latch the position counter, capture timer, and capture period values on a unit time out event so that latched values are used for velocity calculation as described in Section 34.2.2.4.

Figure 34-19. eQEP Unit Time Base

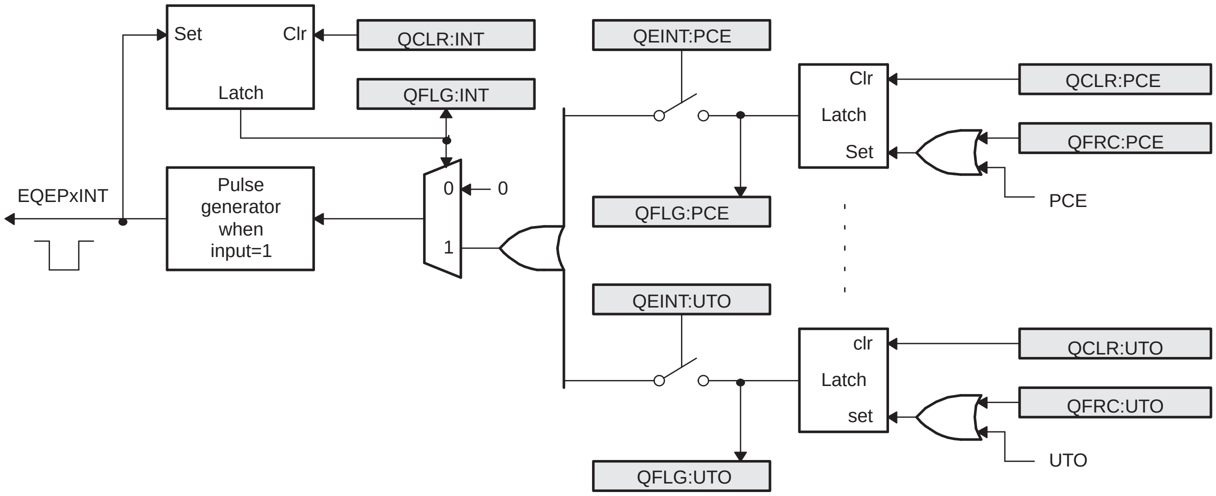


34.2.5 eQEP Interrupt Structure

Figure 34-20 shows how the interrupt mechanism works in the EQEP module.

Eleven interrupt events (PCE, PHE, QDC, WTO, PCU, PCO, PCR, PCM, SEL, IEL and UTO) can be generated. The interrupt control register (QEINT) is used to enable/disable individual interrupt event sources. The interrupt flag register (QFLG) indicates if any interrupt event has been latched and contains the global interrupt flag bit (INT). An interrupt pulse is generated only to the PIE if any of the interrupt events is enabled, the flag bit is 1 and the INT flag bit is 0. The interrupt service routine will need to clear the global interrupt flag bit and the serviced event, via the interrupt clear register (QCLR), before any other interrupt pulses are generated. You can force an interrupt event by way of the interrupt force register (QFRC), which is useful for test purposes.

Figure 34-20. EQEP Interrupt Generation



34.3 eQEP Registers

Table 34-3 lists the registers of the eQEP. The base address for the control registers is FCF7 9900h for eQEP1 and FCF7 9A00h for eQEP2.

Table 34-3. eQEP Registers

Address Offset	Acronym	Register Description	Section
00h	QPOSCNT	eQEP Position Counter Register	Section 34.3.1
04h	QPOSINIT	eQEP Position Counter Initialization Register	Section 34.3.2
08h	QPOSMAX	eQEP Maximum Position Count Register	Section 34.3.3
0Ch	QPOSCMP	eQEP Position-Compare Register	Section 34.3.4
10h	QPOSILAT	eQEP Index Position Latch Register	Section 34.3.5
14h	QPOSSLAT	eQEP Strobe Position Latch Register	Section 34.3.6
18h	QPOSLAT	eQEP Position Counter Latch Register	Section 34.3.7
1Ch	QUTMR	eQEP Unit Timer Register	Section 34.3.8
20h	QUPRD	eQEP Unit Period Register	Section 34.3.9
24h	QWDPRD	eQEP Watchdog Period Register	Section 34.3.10
26h	QWDTMR	eQEP Watchdog Timer Register	Section 34.3.11
28h	QEPCTL	eQEP Control Register	Section 34.3.12
2Ah	QDECCTL	eQEP Decoder Control Register	Section 34.3.13
2Ch	QPOSCTL	eQEP Position-Compare Control Register	Section 34.3.14
2Eh	QCAPCTL	eQEP Capture Control Register	Section 34.3.15
30h	QFLG	eQEP Interrupt Flag Register	Section 34.3.16
32h	QEINT	eQEP Interrupt Enable Register	Section 34.3.17
34h	QFRC	eQEP Interrupt Force Register	Section 34.3.18
36h	QCLR	eQEP Interrupt Clear Register	Section 34.3.19
38h	QCTMR	eQEP Capture Timer Register	Section 34.3.20
3Ah	QEPSTS	eQEP Status Register	Section 34.3.21
3Ch	QCTMRLAT	eQEP Capture Timer Latch Register	Section 34.3.22
3Eh	QCPRD	eQEP Capture Period Register	Section 34.3.23
42h	QCPRDLAT	eQEP Capture Period Latch Register	Section 34.3.24

34.3.1 eQEP Position Counter Register (QPOSCNT)

Figure 34-21. eQEP Position Counter Register (QPOSCNT) [offset = 00h]

31 … 0
QPOSCNT
R/W-0

LEGEND: R/W = Read/Write; -*n* = value after reset

Table 34-4. eQEP Position Counter Register (QPOSCNT) Field Descriptions

Bits	Name	Description
31-0	QPOSCNT	This 32-bit position counter register counts up/down on every eQEP pulse based on direction input. This counter acts as a position integrator whose count value is proportional to position from a give reference point.

34.3.2 eQEP Position Counter Initialization Register (QPOSINIT)

Figure 34-22. eQEP Position Counter Initialization Register (QPOSINIT) [offset = 04h]

31 … 0
QPOSINIT
R/W-0

LEGEND: R/W = Read/Write; -*n* = value after reset

Table 34-5. eQEP Position Counter Initialization Register (QPOSINIT) Field Descriptions

Bits	Name	Description
31-0	QPOSINIT	This register contains the position value that is used to initialize the position counter based on external strobe or index event. The position counter can be initialized through software.

34.3.3 eQEP Maximum Position Count Register (QPOSMAX)

Figure 34-23. eQEP Maximum Position Count Register (QPOSMAX) [offset = 08h]

31 … 0
QPOSMAX
R/W-0

LEGEND: R/W = Read/Write; -*n* = value after reset

Table 34-6. eQEP Maximum Position Count Register (QPOSMAX) Field Descriptions

Bits	Name	Description
31-0	QPOSMAX	This register contains the maximum position counter value.

34.3.4 eQEP Position-Compare Register (QPOSCMP)

Figure 34-24. eQEP Position-Compare Register (QPOSCMP) [offset = 0Ch]

31 … 0
QPOSCMP
R/W-0

LEGEND: R/W = Read/Write; -*n* = value after reset

Table 34-7. eQEP Position-Compare Register (QPOSCMP) Field Descriptions

Bits	Name	Description
31-0	QPOSCMP	The position-compare value in this register is compared with the position counter (QPOSCNT) to generate sync output and/or interrupt on compare match.

34.3.5 eQEP Index Position Latch Register (QPOSILAT)

Figure 34-25. eQEP Index Position Latch Register (QPOSILAT) [offset = 10h]

31 … 0
QPOSILAT
R-0

LEGEND: R = Read only; -*n* = value after reset

Table 34-8. eQEP Index Position Latch Register (QPOSILAT) Field Descriptions

Bits	Name	Description
31-0	QPOSILAT	The position-counter value is latched into this register on an index event as defined by the QEPCTL[IEL] bits.

34.3.6 eQEP Strobe Position Latch Register (QPOSSLAT)

Figure 34-26. eQEP Strobe Position Latch Register (QPOSSLAT) [offset = 14h]

31 … 0
QPOSSLAT
R-0

LEGEND: R = Read only; -*n* = value after reset

Table 34-9. eQEP Strobe Position Latch Register (QPOSSLAT) Field Descriptions

Bits	Name	Description
31-0	QPOSSLAT	The position-counter value is latched into this register on strobe event as defined by the QEPCTL[SEL] bits.

34.3.7 eQEP Position Counter Latch Register (QPOSLAT)

Figure 34-27. eQEP Position Counter Latch Register (QPOSLAT) [offset = 18h]

31	0
QPOSLAT	
R-0	

LEGEND: R = Read only; -*n* = value after reset

Table 34-10. eQEP Position Counter Latch Register (QPOSLAT) Field Descriptions

Bits	Name	Description
31-0	QPOSLAT	The position-counter value is latched into this register on unit time out event.

34.3.8 eQEP Unit Timer Register (QUTMR)

Figure 34-28. eQEP Unit Timer Register (QUTMR) [offset = 1Ch]

31	0
QUTMR	
R/W-0	

LEGEND: R/W = Read/Write; -*n* = value after reset

Table 34-11. eQEP Unit Timer Register (QUTMR) Field Descriptions

Bits	Name	Description
31-0	QUTMR	This register acts as time base for unit time event generation. When this timer value matches with unit time period value, unit time event is generated.

34.3.9 eQEP Unit Period Register (QUPRD)

Figure 34-29. eQEP Unit Period Register (QUPRD) [offset = 20h]

31	0
QUPRD	
R/W-0	

LEGEND: R/W = Read/Write; -*n* = value after reset

Table 34-12. eQEP Unit Period Register (QUPRD) Field Descriptions

Bits	Name	Description
31-0	QUPRD	This register contains the period count for unit timer to generate periodic unit time events to latch the eQEP position information at periodic interval and optionally to generate interrupt.

34.3.10 eQEP Watchdog Period Register (QWDPRD)

Figure 34-30. eQEP Watchdog Period Register (QWDPRD) [offset = 24h]

15	0
QWDPRD	
R/W-0	

LEGEND: R/W = Read/Write; -*n* = value after reset

Table 34-13. eQEP Watchdog Period Register (QWDPRD) Field Description

Bits	Name	Description
15-0	QWDPRD	This register contains the time-out count for the eQEP peripheral watchdog timer. When the watchdog timer value matches the watchdog period value, a watchdog timeout interrupt is generated.

34.3.11 eQEP Watchdog Timer Register (QWDTMR)

Figure 34-31. eQEP Watchdog Timer Register (QWDTMR) [offset = 26h]

15	0
QWDTMR	
R/W-0	

LEGEND: R/W = Read/Write; -*n* = value after reset

Table 34-14. eQEP Watchdog Timer Register (QWDTMR) Field Descriptions

Bits	Name	Description
15-0	QWDTMR	This register acts as time base for watchdog to detect motor stalls. When this timer value matches with watchdog period value, watchdog timeout interrupt is generated. This register is reset upon edge transition in quadrature-clock indicating the motion.

34.3.12 eQEP Control Register (QEPCTL)

Figure 34-32. eQEP Control Register (QEPCTL) [offset = 28h]

15	14	13	12	11	10	9	8
FREE	SOFT	PCRM		SEI		IEI	
R/W-0	R/W-0	R/W-0		R/W-0		R/W-0	

7	6	5	4	3	2	1	0
SWI	SEL	IEL		QPEN	QCLM	UTE	WDE
R/W-0	R/W-0	R/W-0		R/W-0	R/W-0	R/W-0	R/W-0

LEGEND: R/W = Read/Write; -*n* = value after reset

Table 34-15. eQEP Control Register (QEPCTL) Field Descriptions

Bits	Name	Value	Description
15-14	FREE, SOFT		Emulation control bits.
			QPOSCNT behavior:
		0	Position counter stops immediately on emulation suspend.
		1h	Position counter continues to count until the rollover.
		2h-3h	Position counter is unaffected by emulation suspend.
			QWDTMR behavior:
		0	Watchdog counter stops immediately.
		1h	Watchdog counter counts until WD period match roll over.
		2h-3h	Watchdog counter is unaffected by emulation suspend.
			QUTMR behavior:
		0	Unit timer stops immediately.
		1h	Unit timer counts until period rollover.
		2h-3h	Unit timer is unaffected by emulation suspend.
			QCTMR behavior:
		0	Capture timer stops immediately.
		1h	Capture timer counts until next unit period event.
		2h-3h	Capture timer is unaffected by emulation suspend.
13-12	PCRM		Position counter reset mode.
		0	Position counter reset on an index event.
		1h	Position counter reset on the maximum position.
		2h	Position counter reset on the first index event.
		3h	Position counter reset on a unit time event.
11-10	SEI		Strobe event initialization of position counter.
		0	Does nothing (action is disabled).
		1h	Does nothing (action is disabled).
		2h	Initializes the position counter on rising edge of the QEPS signal.
		3h	Clockwise Direction: Initializes the position counter on the rising edge of QEPS strobe. Counter Clockwise Direction: Initializes the position counter on the falling edge of QEPS strobe.
9-8	IEI		Index event initialization of position counter.
		0	Do nothing (action is disabled).
		1h	Do nothing (action is disabled).
		2h	Initializes the position counter on the rising edge of the QEPI signal (QPOSCNT = QPOSINIT).
		3h	Initializes the position counter on the falling edge of QEPI signal (QPOSCNT = QPOSINIT).
7	SWI		Software initialization of position counter.
		0	Do nothing (action is disabled).
		1	Initialize position counter (QPOSCNT=QPOSINIT). This bit is not cleared automatically.

Table 34-15. eQEP Control Register (QEPCTL) Field Descriptions (continued)

Bits	Name	Value	Description
6	SEL		Strobe event latch of position counter.
		0	The position counter is latched on the rising edge of QEPS strobe (QPOSSLAT = POSCCNT). Latching on the falling edge can be done by inverting the strobe input using the QSP bit in the QDECCTL register.
		1	Clockwise Direction: Position counter is latched on rising edge of QEPS strobe.
			Counter Clockwise Direction: Position counter is latched on falling edge of QEPS strobe.
5-4	IEL		Index event latch of position counter (software index marker).
		0	Reserved
		1h	Latches position counter on rising edge of the index signal.
		2h	Latches position counter on falling edge of the index signal.
		3h	Software index marker. Latches the position counter and quadrature direction flag on index event marker. The position counter is latched to the QPOSILAT register and the direction flag is latched in the QEPSTS[QDLF] bit. This mode is useful for software index marking.
3	QPEN		Quadrature position counter enable/software reset.
		0	Reset the eQEP peripheral internal operating flags/read-only registers. Control/configuration registers are not disturbed by a software reset.
		1	eQEP position counter is enabled.
2	QCLM		eQEP capture latch mode.
		0	Latch on position counter read by CPU. Capture timer and capture period values are latched into QCTMRLAT and QCPRDLAT registers when CPU reads the QPOSCNT register.
		1	Latch on unit time out. Position counter, capture timer and capture period values are latched into QPOSLAT, QCTMRLAT and QCPRDLAT registers on unit time out.
1	UTE		eQEP unit timer enable.
		0	eQEP unit timer is disabled.
		1	eQEP unit timer is enabled.
0	WDE		eQEP watchdog enable.
		0	eQEP watchdog timer is disabled.
		1	eQEP watchdog timer is enabled.

34.3.13 eQEP Decoder Control Register (QDECCTL)

Figure 34-33. eQEP Decoder Control Register (QDECCTL) [offset = 2Ah]

15	14	13	12	11	10	9	8
QSRC		SOEN	SPSEL	XCR	SWAP	IGATE	QAP
R/W-0		R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

7	6	5	4				0
QBP	QIP	QSP	Reserved				
R/W-0	R/W-0	R/W-0	R-0				

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 34-16. eQEP Decoder Control Register (QDECCTL) Field Descriptions

Bits	Name	Value	Description
15-14	QSRC		Position-counter source selection.
		0	Quadrature count mode: (QCLK = iCLK, QDIR = iDIR).
		1h	Direction-count mode: (QCLK = xCLK, QDIR = xDIR).
		2h	UP count mode for frequency measurement :(QCLK = xCLK, QDIR = 1).
		3h	DOWN count mode for frequency measurement: (QCLK = xCLK, QDIR = 0).
13	SOEN		Sync output-enable.
		0	Position-compare sync output is disabled.
		1	Position-compare sync output is enabled.
12	SPSEL		Sync output pin selection.
		0	Index pin is used for sync output.
		1	Strobe pin is used for sync output.
11	XCR		External clock rate.
		0	2x resolution: Count the rising/falling edge.
		1	1x resolution: Count the rising edge only.
10	SWAP		Swap quadrature clock inputs. This swaps the input to the quadrature decoder, reversing the counting direction.
		0	Quadrature-clock inputs are not swapped.
		1	Quadrature-clock inputs are swapped.
9	IGATE		Index pulse gating option.
		0	Disable gating of Index pulse.
		1	Gate the index pin with strobe.
8	QAP		QEPA input polarity.
		0	No effect.
		1	Negates QEPA input.
7	QBP		QEPB input polarity.
		0	No effect.
		1	Negates QEPB input.
6	QIP		QEPI input polarity.
		0	No effect.
		1	Negates QEPI input.
5	QSP		QEPS input polarity.
		0	No effect.
		1	Negates QEPS input.
4-0	Reserved	0	Always read as 0.

34.3.14 eQEP Position-Compare Control Register (QPOSCTL)

Figure 34-34. eQEP Position-Compare Control Register (QPOSCTL) [offset = 2Ch]

15	14	13	12	11	8
PCSHDW	PCLOAD	PCPOL	PCE	PCSPW	
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	

7	0
PCSPW	
R/W-0	

LEGEND: R/W = Read/Write; -*n* = value after reset

Table 34-17. eQEP Position-Compare Control Register (QPOSCTL) Field Descriptions

Bit	Name	Value	Description
15	PCSHDW		Position-compare shadow enable.
		0	Shadow is disabled, load Immediate.
		1	Shadow is enabled.
14	PCLOAD		Position-compare shadow load mode.
		0	Load on QPOSCNT = 0.
		1	Load when QPOSCNT = QPOSCMP.
13	PCPOL		Polarity of sync output.
		0	Active HIGH pulse output.
		1	Active LOW pulse output.
12	PCE		Position-compare enable.
		0	Position-compare unit is disabled.
		1	Position-compare unit is enabled.
11-0	PCSPW		Select-position-compare sync output pulse width.
		0	1 × 4 × VCLK3 cycles
		1h	2 × 4 × VCLK3 cycles
		:	:
		FFFh	4096 × 4 × VCLK3 cycles

34.3.15 eQEP Capture Control Register (QCAPCTL)

Figure 34-35. eQEP Capture Control Register (QCAPCTL) [offset = 2Eh]

15	14 7	6 4	3 0
CEN	Reserved	CCPS	UPPS
R/W-0	R-0	R/W-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 34-18. eQEP Capture Control Register (QCAPCTL) Field Descriptions

Bits	Name	Value	Description
15	CEN		Enable eQEP capture.
		0	eQEP capture unit is disabled.
		1	eQEP capture unit is enabled.
14-7	Reserved	0	Always read as 0.
6-4	CCPS		eQEP capture timer clock prescaler.
		0	CAPCLK = VCLK3/1
		1h	CAPCLK = VCLK3/2
		2h	CAPCLK = VCLK3/4
		3h	CAPCLK = VCLK3/8
		4h	CAPCLK = VCLK3/16
		5h	CAPCLK = VCLK3/32
		6h	CAPCLK = VCLK3/64
		7h	CAPCLK = VCLK3/128
3-0	UPPS		Unit position event prescaler.
		0	UPEVNT = QCLK/1
		1h	UPEVNT = QCLK/2
		2h	UPEVNT = QCLK/4
		3h	UPEVNT = QCLK/8
		4h	UPEVNT = QCLK/16
		5h	UPEVNT = QCLK/32
		6h	UPEVNT = QCLK/64
		7h	UPEVNT = QCLK/128
		8h	UPEVNT = QCLK/256
		9h	UPEVNT = QCLK/512
		Ah	UPEVNT = QCLK/1024
		Bh	UPEVNT = QCLK/2048
		Ch-Fh	Reserved

34.3.16 eQEP Interrupt Flag Register (QFLG)

Figure 34-36. eQEP Interrupt Flag Register (QFLG) [offset = 30h]

15–12	11	10	9	8
Reserved	UTO	IEL	SEL	PCM
R-0	R-0	R-0	R-0	R-0

7	6	5	4	3	2	1	0
PCR	PCO	PCU	WTO	QDC	PHE	PCE	INT
R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0

LEGEND: R = Read only; -*n* = value after reset

Table 34-19. eQEP Interrupt Flag Register (QFLG) Field Descriptions

Bits	Name	Value	Description
15-12	Reserved	0	Always read as 0.
11	UTO		Unit time out interrupt flag.
		0	No interrupt is generated.
		1	Set by eQEP unit timer period match.
10	IEL		Index event latch interrupt flag.
		0	No interrupt is generated.
		1	Set after latching the QPOSCNT to QPOSILAT.
9	SEL		Strobe event latch interrupt flag.
		0	No interrupt is generated.
		1	Set after latching the QPOSCNT to QPOSSLAT.
8	PCM		Position-compare match interrupt flag.
		0	No interrupt is generated.
		1	Set on position-compare match.
7	PCR		Position-compare ready interrupt flag.
		0	No interrupt is generated.
		1	Set after transferring the shadow register value to the active position compare register.
6	PCO		Position counter overflow interrupt flag.
		0	No interrupt is generated.
		1	Set on position counter overflow.
5	PCU		Position counter underflow interrupt flag.
		0	No interrupt is generated.
		1	Set on position counter underflow.
4	WTO		Watchdog time out interrupt flag.
		0	No interrupt is generated.
		1	Set by watchdog timeout.
3	QDC		Quadrature direction change interrupt flag.
		0	No interrupt is generated.
		1	Set during change of direction.
2	PHE		Quadrature phase error interrupt flag.
		0	No interrupt is generated.
		1	Set on simultaneous transition of QEPA and QEPB.
1	PCE		Position counter error interrupt flag.
		0	No interrupt is generated.
		1	Position counter error.
0	INT		Global interrupt status flag.
		0	No interrupt is generated.
		1	Interrupt was generated.

34.3.17 eQEP Interrupt Enable Register (QEINT)

Figure 34-37. eQEP Interrupt Enable Register (QEINT) [offset = 32h]

15–12	11	10	9	8
Reserved	UTO	IEL	SEL	PCM
R-0	R/W-0	R/W-0	R/W-0	R/W-0

7	6	5	4	3	2	1	0
PCR	PCO	PCU	WTO	QDC	QPE	PCE	Reserved
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 34-20. eQEP Interrupt Enable Register (QEINT) Field Descriptions

Bits	Name	Value	Description
15-12	Reserved	0	Always read as 0.
11	UTO		Unit time out interrupt enable.
		0	Interrupt is disabled.
		1	Interrupt is enabled.
10	IEL		Index event latch interrupt enable.
		0	Interrupt is disabled.
		1	Interrupt is enabled.
9	SEL		Strobe event latch interrupt enable.
		0	Interrupt is disabled.
		1	Interrupt is enabled.
8	PCM		Position-compare match interrupt enable.
		0	Interrupt is disabled.
		1	Interrupt is enabled.
7	PCR		Position-compare ready interrupt enable.
		0	Interrupt is disabled.
		1	Interrupt is enabled.
6	PCO		Position counter overflow interrupt enable.
		0	Interrupt is disabled.
		1	Interrupt is enabled.
5	PCU		Position counter underflow interrupt enable.
		0	Interrupt is disabled.
		1	Interrupt is enabled.
4	WTO		Watchdog time out interrupt enable.
		0	Interrupt is disabled.
		1	Interrupt is enabled.
3	QDC		Quadrature direction change interrupt enable.
		0	Interrupt is disabled.
		1	Interrupt is enabled.
2	QPE		Quadrature phase error interrupt enable.
		0	Interrupt is disabled.
		1	Interrupt is enabled.
1	PCE		Position counter error interrupt enable.
		0	Interrupt is disabled.
		1	Interrupt is enabled.
0	Reserved	0	Always read as 0.

34.3.18 eQEP Interrupt Force Register (QFRC)

Figure 34-38. eQEP Interrupt Force Register (QFRC) [offset = 34h]

<table>
<tr><td>15</td><td></td><td></td><td>12</td><td>11</td><td>10</td><td>9</td><td>8</td></tr>
<tr><td colspan="4">Reserved</td><td>UTO</td><td>IEL</td><td>SEL</td><td>PCM</td></tr>
<tr><td colspan="4">R-0</td><td>R/W-0</td><td>R/W-0</td><td>R/W-0</td><td>R/W-0</td></tr>
<tr><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td></tr>
<tr><td>PCR</td><td>PCO</td><td>PCU</td><td>WTO</td><td>QDC</td><td>PHE</td><td>PCE</td><td>Reserved</td></tr>
<tr><td>R/W-0</td><td>R/W-0</td><td>R/W-0</td><td>R/W-0</td><td>R/W-0</td><td>R/W-0</td><td>R/W-0</td><td>R-0</td></tr>
</table>

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 34-21. eQEP Interrupt Force Register (QFRC) Field Descriptions

Bit	Field	Value	Description
15-12	Reserved	0	Always read as 0.
11	UTO		Force unit time out interrupt.
		0	No effect.
		1	Force the interrupt.
10	IEL		Force index event latch interrupt.
		0	No effect.
		1	Force the interrupt.
9	SEL		Force strobe event latch interrupt.
		0	No effect.
		1	Force the interrupt.
8	PCM		Force position-compare match interrupt.
		0	No effect.
		1	Force the interrupt.
7	PCR		Force position-compare ready interrupt.
		0	No effect.
		1	Force the interrupt.
6	PCO		Force position counter overflow interrupt.
		0	No effect.
		1	Force the interrupt.
5	PCU		Force position counter underflow interrupt.
		0	No effect.
		1	Force the interrupt.
4	WTO		Force watchdog time out interrupt.
		0	No effect.
		1	Force the interrupt.
3	QDC		Force quadrature direction change interrupt.
		0	No effect.
		1	Force the interrupt.
2	PHE		Force quadrature phase error interrupt.
		0	No effect.
		1	Force the interrupt.
1	PCE		Force position counter error interrupt.
		0	No effect.
		1	Force the interrupt.
0	Reserved	0	Always read as 0.

34.3.19 eQEP Interrupt Clear Register (QCLR)

Figure 34-39. eQEP Interrupt Clear Register (QCLR) [offset = 36h]

15 12	11	10	9	8
Reserved	UTO	IEL	SEL	PCM
R-0	R/W-0	R/W-0	R/W-0	R/W-0

7	6	5	4	3	2	1	0
PCR	PCO	PCU	WTO	QDC	PHE	PCE	INT
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 34-22. eQEP Interrupt Clear Register (QCLR) Field Descriptions

Bit	Field	Value	Description
15-12	Reserved	0	Always read as 0.
11	UTO		Clear unit time out interrupt flag.
		0	No effect.
		1	Clears the interrupt flag.
10	IEL		Clear index event latch interrupt flag.
		0	No effect.
		1	Clears the interrupt flag.
9	SEL		Clear strobe event latch interrupt flag.
		0	No effect.
		1	Clears the interrupt flag.
8	PCM		Clear eQEP compare match event interrupt flag.
		0	No effect.
		1	Clears the interrupt flag.
7	PCR		Clear position-compare ready interrupt flag.
		0	No effect.
		1	Clears the interrupt flag.
6	PCO		Clear position counter overflow interrupt flag.
		0	No effect.
		1	Clears the interrupt flag.
5	PCU		Clear position counter underflow interrupt flag.
		0	No effect.
		1	Clears the interrupt flag.
4	WTO		Clear watchdog timeout interrupt flag.
		0	No effect.
		1	Clears the interrupt flag.
3	QDC		Clear quadrature direction change interrupt flag.
		0	No effect.
		1	Clears the interrupt flag.
2	PHE		Clear quadrature phase error interrupt flag.
		0	No effect.
		1	Clears the interrupt flag.
1	PCE		Clear position counter error interrupt flag.
		0	No effect.
		1	Clears the interrupt flag.
0	INT		Global interrupt clear flag.
		0	No effect.
		1	Clears the interrupt flag and enables further interrupts to be generated if an event flags is set to 1.

34.3.20 eQEP Capture Timer Register (QCTMR)

Figure 34-40. eQEP Capture Timer Register (QCTMR) [offset = 38h]

15 … 0
QCTMR
R/W-0

LEGEND: R/W = Read/Write; -*n* = value after reset

Table 34-23. eQEP Capture Time Register (QCTMR) Field Descriptions

Bits	Name	Description
15-0	QCTMR	This register provides time base for edge capture unit.

34.3.21 eQEP Status Register (QEPSTS)

Figure 34-41. eQEP Status Register (QEPSTS) [offset = 3Ah]

15 … 8
Reserved
R-0

7	6	5	4	3	2	1	0
UPEVNT	FIDF	QDF	QDLF	COEF	CDEF	FIMF	PCEF
R-1	R-0	R-0	R-0	R/W-1	R/W-1	R/W-1	R-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 34-24. eQEP Status Register (QEPSTS) Field Descriptions

Bit	Field	Value	Description
15-8	Reserved	0	Always read as 0.
7	UPEVNT		Unit position event flag.
		0	No unit position event is detected.
		1	Unit position event is detected. Write 1 to clear.
6	FIDF		Direction on the first index marker. Status of the direction is latched on the first index event marker.
		0	Counter-clockwise rotation (or reverse movement) on the first index event.
		1	Clockwise rotation (or forward movement) on the first index event.
5	QDF		Quadrature direction flag.
		0	Counter-clockwise rotation (or reverse movement).
		1	Clockwise rotation (or forward movement).
4	QDLF		eQEP direction latch flag. Status of direction is latched on every index event marker.
		0	Counter-clockwise rotation (or reverse movement) on index event marker.
		1	Clockwise rotation (or forward movement) on index event marker.
3	COEF		Capture overflow error flag.
		0	Sticky bit, cleared by writing 1.
		1	Overflow occurred in eQEP Capture timer (QCTMR).
2	CDEF		Capture direction error flag.
		0	Sticky bit, cleared by writing 1.
		1	Direction change occurred between the capture position event.
1	FIMF		First index marker flag.
		0	Sticky bit, cleared by writing 1.
		1	Set by first occurrence of index pulse.
0	PCEF		Position counter error flag. This bit is not sticky and it is updated for every index event.
		0	No error occurred during the last index transition.
		1	Position counter error.

34.3.22 eQEP Capture Timer Latch Register (QCTMRLAT)

Figure 34-42. eQEP Capture Timer Latch Register (QCTMRLAT) [offset = 3Ch]

15 0
QCTMRLAT
R-0

LEGEND: R = Read only; -*n* = value after reset

Table 34-25. eQEP Capture Timer Latch Register (QCTMRLAT) Field Descriptions

Bits	Name	Description
15-0	QCTMRLAT	The eQEP capture timer value can be latched into this register on two events viz., unit timeout event, reading the eQEP position counter.

34.3.23 eQEP Capture Period Register (QCPRD)

Figure 34-43. eQEP Capture Period Register (QCPRD) [offset = 3Eh]

15 0
QCPRD
R/W-0

LEGEND: R/W = Read/Write; -*n* = value after reset

Table 34-26. eQEP Capture Period Register (QCPRD) Field Descriptions

Bits	Name	Description
15-0	QCPRD	This register holds the period count value between the last successive eQEP position events.

34.3.24 eQEP Capture Period Latch Register (QCPRDLAT)

Figure 34-44. eQEP Capture Period Latch Register (QCPRDLAT) [offset = 42h]

15 0
QCPRDLAT
R/W-0

LEGEND: R/W = Read/Write; -*n* = value after reset

Table 34-27. eQEP Capture Period Latch Register (QCPRDLAT) Field Descriptions

Bits	Name	Description
15-0	QCPRDLAT	eQEP capture period value can be latched into this register on two events viz., unit timeout event, reading the eQEP position counter.

Chapter 35

SPNU563A–March 2018

Enhanced Pulse Width Modulator (ePWM) Module

The enhanced pulse width modulator (ePWM) peripheral is a key element in controlling many of the power electronic systems found in both commercial and industrial equipments. The features supported by the ePWM make it especially suitable for digital motor control.

Topic **Page**

35.1 Introduction

An effective PWM peripheral must be able to generate complex pulse width waveforms with minimal CPU overhead or intervention. It needs to be highly programmable and very flexible while being easy to understand and use. The ePWM unit described here addresses these requirements by allocating all needed timing and control resources on a per PWM channel basis. Cross coupling or sharing of resources has been avoided; instead, the ePWM is built up from smaller single channel modules with separate resources that can operate together as required to form a system. This modular approach results in an orthogonal architecture and provides a more transparent view of the peripheral structure, helping users to understand its operation quickly.

In this document the letter x within a signal or module name is used to indicate a generic ePWM instance on a device. For example output signals EPWMxA and EPWMxB refer to the output signals from the ePWMx instance. Thus, EPWM1A and EPWM1B belong to ePWM1 and likewise EPWM4A and EPWM4B belong to ePWM4.

35.1.1 Submodule Overview

The ePWM module represents one complete PWM channel composed of two PWM outputs: EPWMxA and EPWMxB. Multiple ePWM modules are instanced within a device as shown in Figure 35-1. Each ePWM instance is identical and is indicated by a numerical value starting with 1. For example, ePWM1 is the first instance and ePWM3 is the third instance in the system, and ePWMx indicates any instance.

The ePWM modules are chained together via a clock synchronization scheme that allows them to operate as a single system when required. Additionally, this synchronization scheme can be extended to the capture peripheral modules (eCAP). Modules can also operate stand-alone.

Each ePWM module supports the following features:

- Dedicated 16-bit time-base counter with period and frequency control
- Two PWM outputs (EPWMxA and EPWMxB) that can be used in the following configurations:
  - Two independent PWM outputs with single-edge operation
  - Two independent PWM outputs with dual-edge symmetric operation
  - One independent PWM output with dual-edge asymmetric operation
- Asynchronous override control of PWM signals through software.
- Programmable phase-control support for lag or lead operation relative to other ePWM modules.
- Hardware-locked (synchronized) phase relationship on a cycle-by-cycle basis.
- Dead-band generation with independent rising and falling edge delay control.
- Programmable trip zone allocation of both cycle-by-cycle trip and one-shot trip on fault conditions.
- A trip condition can force either high, low, or high-impedance state logic levels at PWM outputs.
- All events can trigger both CPU interrupts and ADC start of conversion (SOC)
- Programmable event prescaling minimizes CPU overhead on interrupts.
- PWM chopping by high-frequency carrier signal, useful for pulse transformer gate drives.

Each ePWM module is connected to the input/output signals shown in Figure 35-1. The signals are described in detail in subsequent sections.

Each ePWM module consists of eight submodules and is connected within a system via the signals shown in Figure 35-2.

Figure 35-1. Multiple ePWM Modules

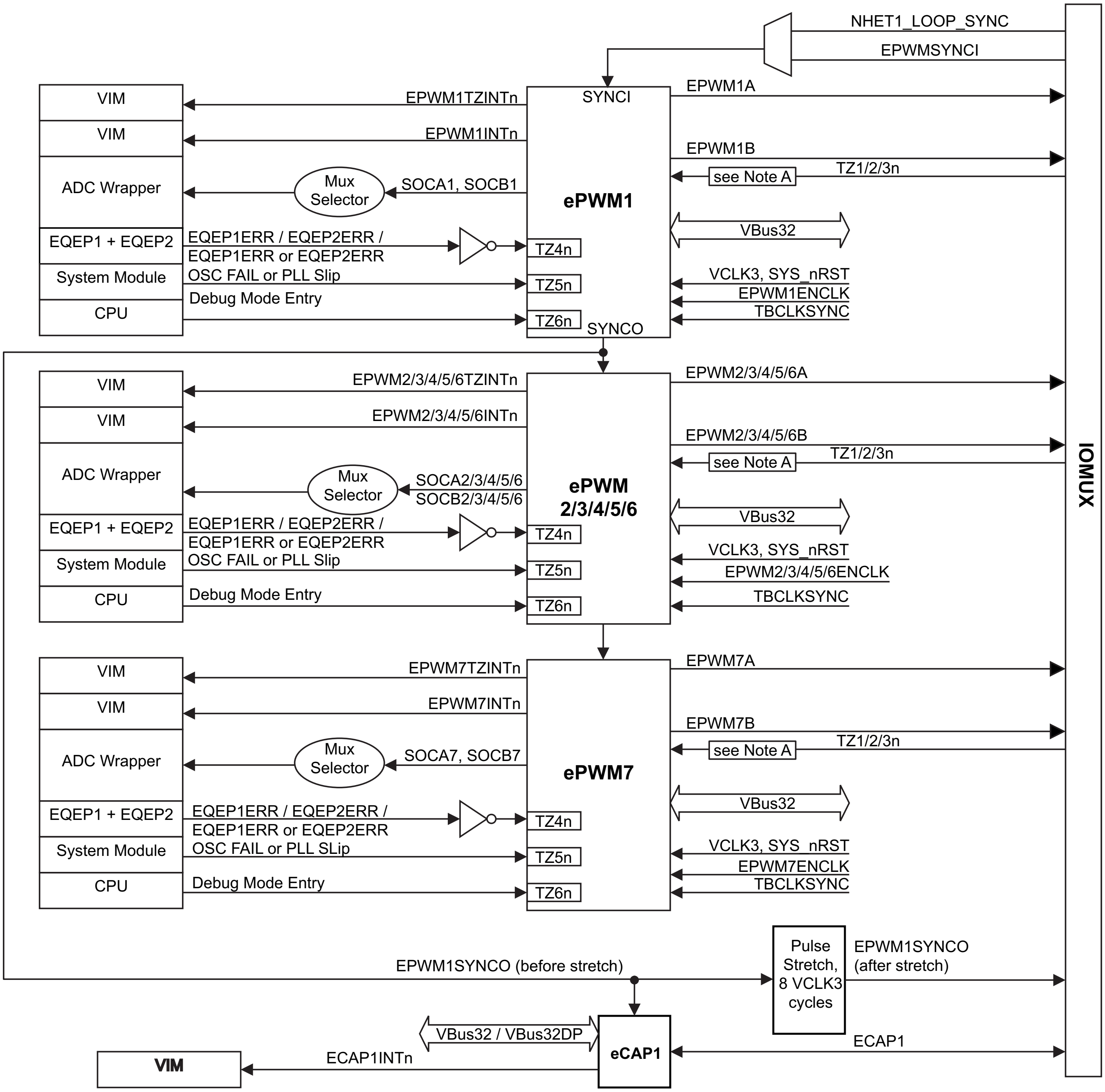
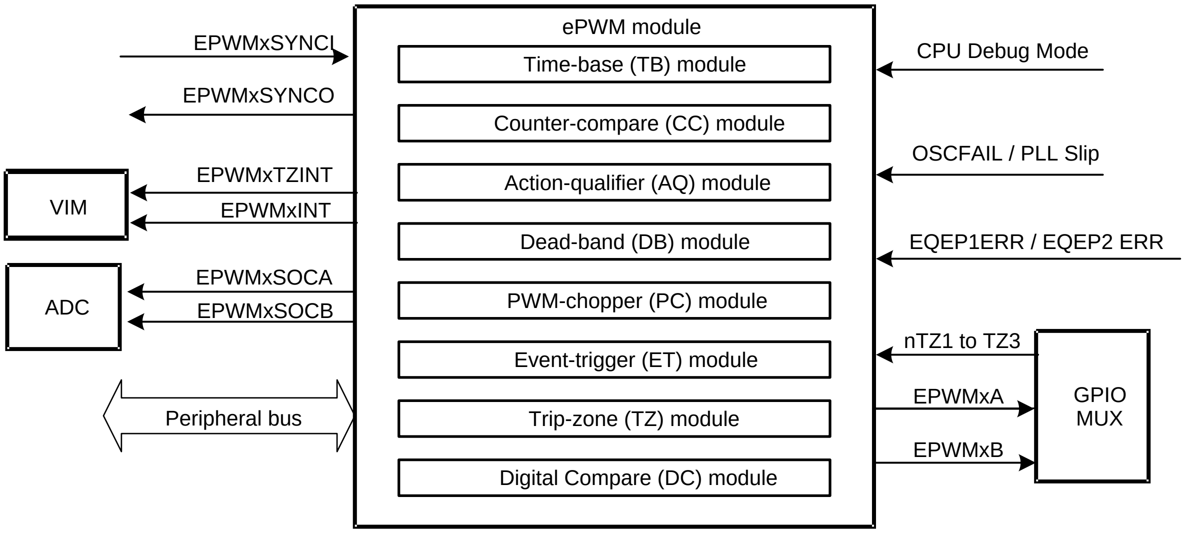


Figure 35-2. Submodules and Signal Connections for an ePWM Module



The main signals used by the ePWM module are:

- **PWM output signals (EPWMxA and EPWMxB).**

  The PWM output signals are made available external to the device through the I/O Multiplexing Module (IOMM) as described in the IOMM chapter.

- **Trip-zone signals ($\overline{TZ1}$ to $\overline{TZ6}$).**

  These input signals alert the ePWM module of fault conditions external to the ePWM module. Each ePWM module can be configured to either use or ignore any of the trip-zone signals. The $\overline{TZ1}$ to $\overline{TZ3}$ trip-zone signals can be configured as asynchronous inputs, or double-synchronized using VCLK3, or double-synchronized and filtered through a 6-VCLK3-cycle counter before connecting to the ePWM modules. This selection is done by configuring registers in the IOMM. $\overline{TZ4}$ is connected to an inverted eQEP1 error signal (EQEP1ERR), or to an inverted eQEP2 error signal (EQEP2ERR), or an OR-combination of EQEP1ERR and EQEP2ERR. This selection is also done via the IOMM registers. $\overline{TZ5}$ is connected to the system clock fail status. This is asserted whenever an oscillator failure is detected, or a PLL slip is detected. $\overline{TZ6}$ is connected to the debug mode entry indicator output from the CPU. This allows you to configure a trip action when the CPU halts.

- **Time-base synchronization input (EPWMxSYNCI) and output (EPWMxSYNCO) signals**.

  The synchronization signals daisy chain the ePWM modules together. Each module can be configured to either use or ignore its synchronization input. The clock synchronization input and output signal are brought out to pins only for ePWM1 (ePWM module #1). The synchronization output for ePWM1 (EPWM1SYNCO) is also connected to the SYNCI of the first enhanced capture module (eCAP1).

- **ADC start-of-conversion signals (EPWMxSOCA and EPWMxSOCB).**

  Each ePWM module has two ADC start of conversion signals. Any ePWM module can trigger a start of conversion. Which event triggers the start of conversion is configured in the Event-Trigger submodule of the ePWM.

- **Peripheral Bus**

  The peripheral bus is 32-bits wide and allows both 16-bit and 32-bit writes to the ePWM register file.

35.1.2 Register Mapping

The complete ePWM module control and status register set is grouped by submodule as shown in Table 35-1. Each register set is duplicated for each instance of the ePWM module.

Table 35-1. ePWM Module Control and Status Register Set Grouped by Submodule

Name	Address Offset[1]	Size (x16)	Shadow	Privileged Mode Write Only?	Description
Time-Base Submodule Registers					
TBSTS	00h	1	No	No	Time-Base Status Register
TBCTL	02h	1	No	No	Time-Base Control Register
TBPHS	04h	1	No	No	Time-Base Phase Register
TBPRD	08h	1	Yes	No	Time-Base Period Register
TBCTR	0Ah	1	No	No	Time-Base Counter Register
Counter-Compare Submodule Registers					
CMPCTL	0Ch	1	No	No	Counter-Compare Control Register
CMPA	10h	1	Yes	No	Counter-Compare A Register
CMPB	16h	1	Yes	No	Counter-Compare B Register
Action-Qualifier Submodule Registers					
AQCTLA	14h	1	No	No	Action-Qualifier Control Register for Output A (EPWMxA)
AQSFRC	18h	1	No	No	Action-Qualifier Software Force Register
AQCTLB	1Ah	1	No	No	Action-Qualifier Control Register for Output B (EPWMxB)
AQCSFRC	1Eh	1	Yes	No	Action-Qualifier Continuous S/W Force Register Set
Dead-Band Generator Submodule Registers					
DBCTL	1Ch	1	No	No	Dead-Band Generator Control Register
DBFED	20h	1	No	No	Dead-Band Generator Falling Edge Delay Count Register
DBRED	22h	1	No	No	Dead-Band Generator Rising Edge Delay Count Register
Trip-Zone Submodule Registers					
TZDCSEL	24h	1	No	Yes	Trip Zone Digital Compare Select Register
TZSEL	26h	1	No	Yes	Trip-Zone Select Register
TZEINT	28h	1	No	Yes	Trip-Zone Enable Interrupt Register
TZCTL	2Ah	1	No	Yes	Trip-Zone Control Register
TZCLR	2Ch	1	No	Yes	Trip-Zone Clear Register
TZFLG	2Eh	1	No	No	Trip-Zone Flag Register
TZFRC	32h	1	No	Yes	Trip-Zone Force Register
Event-Trigger Submodule Registers					
ETSEL	30h	1	No	No	Event-Trigger Selection Register
ETFLG	34h	1	No	No	Event-Trigger Flag Register
ETPS	36h	1	No	No	Event-Trigger Pre-Scale Register
ETFRC	38h	1	No	No	Event-Trigger Force Register
ETCLR	3Ah	1	No	No	Event-Trigger Clear Register
PWM-Chopper Submodule Registers					
PCCTL	3Eh	1	No	No	PWM-Chopper Control Register

[1] Locations not shown are reserved.

Table 35-1. ePWM Module Control and Status Register Set Grouped by Submodule (continued)

Name	Address Offset[1]	Size (x16)	Shadow	Privileged Mode Write Only?	Description
			Digital Compare Event Registers		
DCACTL	60h	1	No	Yes	Digital Compare A Control Register
DCTRIPSEL	62h	1	No	Yes	Digital Compare Trip Select Register
DCFCTL	64h	1	No	Yes	Digital Compare Filter Control Register
DCBCTL	66h	1	No	Yes	Digital Compare B Control Register
DCFOFFSET	68h	1	Writes	No	Digital Compare Filter Offset Register
DCCAPCTL	6Ah	1	No	Yes	Digital Compare Capture Control Register
DCFWINDOW	6Ch	1	No	No	Digital Compare Filter Window Register
DCFOFFSETCNT	6Eh	1	No	No	Digital Compare Filter Offset Counter Register
DCCAP	70h	1	Yes	No	Digital Compare Counter Capture Register
DCFWINDOWCNT	72h	1	No	No	Digital Compare Filter Window Counter Register

35.2 ePWM Submodules

Eight submodules are included in every ePWM peripheral. Each of these submodules performs specific tasks that can be configured by software.

35.2.1 Overview

Table 35-2 lists the eight key submodules together with a list of their main configuration parameters. For example, if you need to adjust or control the duty cycle of a PWM waveform, then you should see the counter-compare submodule in Section 35.2.3 for relevant details.

Table 35-2. Submodule Configuration Parameters

Submodule	Configuration Parameter or Option
Time-base (TB)	• Scale the time-base clock (TBCLK) relative to the system clock (VCLK3). • Configure the PWM time-base counter (TBCTR) frequency or period. • Set the mode for the time-base counter: – count-up mode: used for asymmetric PWM – count-down mode: used for asymmetric PWM – count-up-and-down mode: used for symmetric PWM • Configure the time-base phase relative to another ePWM module. • Synchronize the time-base counter between modules through hardware or software. • Configure the direction (up or down) of the time-base counter after a synchronization event. • Configure how the time-base counter will behave when the device is halted by an emulator. • Specify the source for the synchronization output of the ePWM module: – Synchronization input signal – Time-base counter equal to zero – Time-base counter equal to counter-compare B (CMPB) – No output synchronization signal generated.
Counter-compare (CC)	• Specify the PWM duty cycle for output EPWMxA and/or output EPWMxB • Specify the time at which switching events occur on the EPWMxA or EPWMxB output
Action-qualifier (AQ)	• Specify the type of action taken when a time-base or counter-compare submodule event occurs: – No action taken – Output EPWMxA and/or EPWMxB switched high – Output EPWMxA and/or EPWMxB switched low – Output EPWMxA and/or EPWMxB toggled • Force the PWM output state through software control • Configure and control the PWM dead-band through software

Table 35-2. Submodule Configuration Parameters (continued)

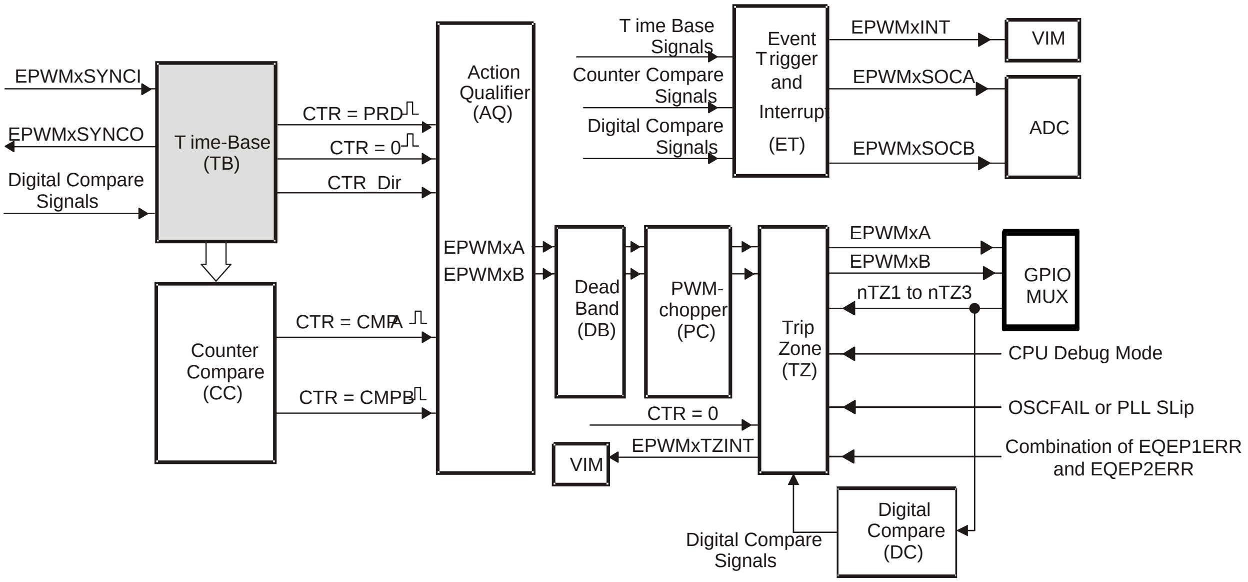
Submodule	Configuration Parameter or Option
Dead-band (DB)	• Control of traditional complementary dead-band relationship between upper and lower switches • Specify the output rising-edge-delay value • Specify the output falling-edge delay value • Bypass the dead-band module entirely. In this case the PWM waveform is passed through without modification. • Option to enable half-cycle clocking for double resolution.
PWM-chopper (PC)	• Create a chopping (carrier) frequency. • Pulse width of the first pulse in the chopped pulse train. • Duty cycle of the second and subsequent pulses. • Bypass the PWM-chopper module entirely. In this case the PWM waveform is passed through without modification.
Trip-zone (TZ)	• Configure the ePWM module to react to one, all, or none of the trip-zone signals or digital compare events. • Specify the tripping action taken when a fault occurs: – Force EPWMxA and/or EPWMxB high – Force EPWMxA and/or EPWMxB low – Force EPWMxA and/or EPWMxB to a high-impedance state – Configure EPWMxA and/or EPWMxB to ignore any trip condition. • Configure how often the ePWM will react to each trip-zone signal: – One-shot – Cycle-by-cycle • Enable the trip-zone to initiate an interrupt. • Bypass the trip-zone module entirely.
Event-trigger (ET)	• Enable the ePWM events that will trigger an interrupt. • Enable ePWM events that will trigger an ADC start-of-conversion event. • Specify the rate at which events cause triggers (every occurrence or every second or third occurrence) • Poll, set, or clear event flags
Digital-compare (DC)	• Enables trip zone signals to create events and filtered events • Specify event-filtering options to capture TBCTR counter or generate blanking window

Code examples are provided in this chapter that show how to implement various ePWM module configurations. These examples use the constant definitions in the device *EPwm_defines.h* file in the device-specific header file and peripheral examples software package.

35.2.2 Time-Base (TB) Submodule

Each ePWM module has its own time-base submodule that determines all of the event timing for the ePWM module. Built-in synchronization logic allows the time-base of multiple ePWM modules to work together as a single system. Figure 35-3 illustrates the time-base module's place within the ePWM.

Figure 35-3. Time-Base Submodule Block Diagram



35.2.2.1 Purpose of the Time-Base Submodule

You can configure the time-base submodule for the following:

- Specify the ePWM time-base counter (TBCTR) frequency or period to control how often events occur.
- Manage time-base synchronization with other ePWM modules.
- Maintain a phase relationship with other ePWM modules.
- Set the time-base counter to count-up, count-down, or count-up-and-down mode.
- Generate the following events:
  - CTR = PRD: Time-base counter equal to the specified period (TBCTR = TBPRD).
  - CTR = Zero: Time-base counter equal to zero (TBCTR = 0x0000).
- Configure the rate of the time-base clock; a prescaled version of the device peripheral clock domain (VCLK3). This allows the time-base counter to increment/decrement at a slower rate.

35.2.2.2 Controlling and Monitoring the Time-Base Submodule

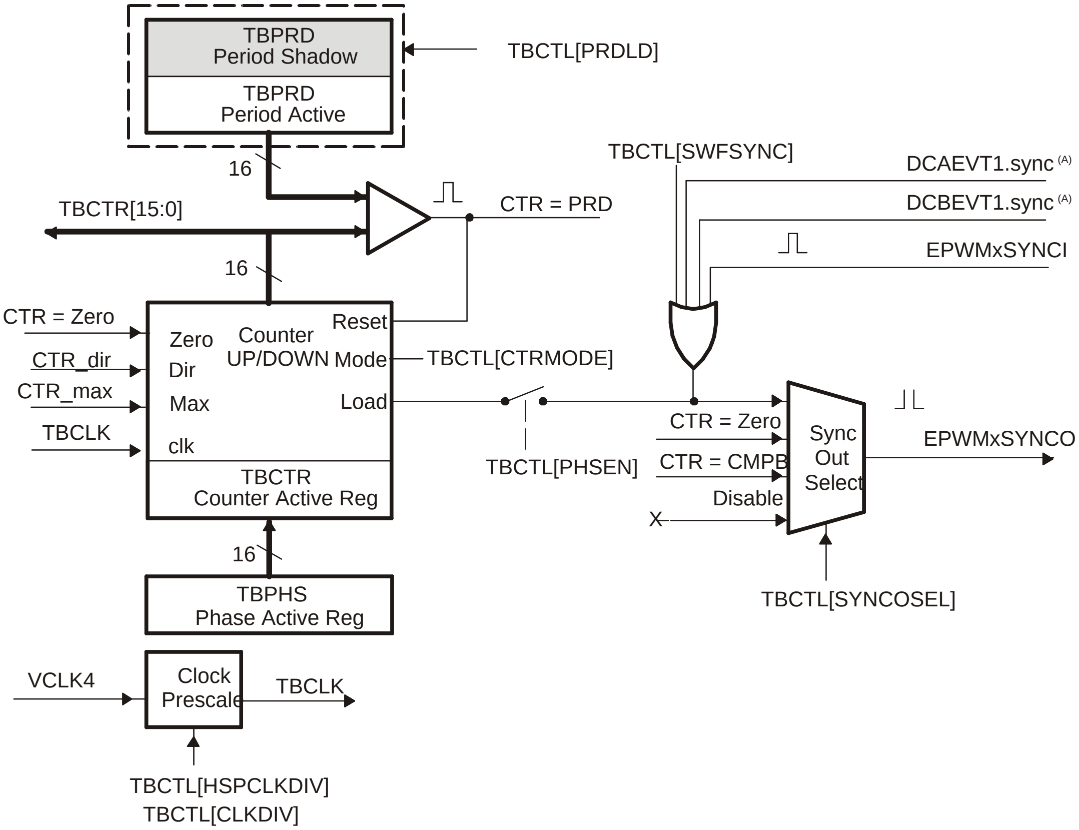
Table 35-3 shows the registers used to control and monitor the time-base submodule.

Table 35-3. Time-Base Submodule Registers

Register	Address Offset	Shadowed	Description
TBSTS	00h	No	Time-Base Status Register
TBCTL	02h	No	Time-Base Control Register
TBPHS	04h	No	Time-Base Phase Register
TBPRD	08h	Yes	Time-Base Period Register
TBCTR	0Ah	No	Time-Base Counter Register

Figure 35-4 shows the critical signals and registers of the time-base submodule. Table 35-4 provides descriptions of the key signals associated with the time-base submodule.

Figure 35-4. Time-Base Submodule Signals and Registers



A. These signals are generated by the digital compare (DC) submodule.

Table 35-4. Key Time-Base Signals

Signal	Description
EPWMxSYNCI	Time-base synchronization input. Input pulse used to synchronize the time-base counter with the counter of ePWM module earlier in the synchronization chain. An ePWM peripheral can be configured to use or ignore this signal. For the first ePWM module (EPWM1) this signal comes from a device pin or from the N2HET1 module. For subsequent ePWM modules this signal is passed from another ePWM peripheral. For example, EPWM2SYNCI is generated by the ePWM1 peripheral, EPWM3SYNCI is generated by ePWM2 and so forth. See Section 35.2.2.3.3 for information on the synchronization order of a particular device.
EPWMxSYNCO	Time-base synchronization output. This output pulse is used to synchronize the counter of an ePWM module later in the synchronization chain. The ePWM module generates this signal from one of three event sources: 1. EPWMxSYNCI (Synchronization input pulse) 2. CTR = Zero: The time-base counter equal to zero (TBCTR = 0x0000). 3. CTR = CMPB: The time-base counter equal to the counter-compare B (TBCTR = CMPB) register.
CTR = PRD	Time-base counter equal to the specified period. This signal is generated whenever the counter value is equal to the active period register value. That is when TBCTR = TBPRD.
CTR = Zero	Time-base counter equal to zero This signal is generated whenever the counter value is zero. That is when TBCTR equals 0x0000.

Table 35-4. Key Time-Base Signals (continued)

Signal	Description
CTR = CMPB	Time-base counter equal to active counter-compare B register (TBCTR = CMPB). This event is generated by the counter-compare submodule and used by the synchronization out logic
CTR_dir	Time-base counter direction. Indicates the current direction of the ePWM's time-base counter. This signal is high when the counter is increasing and low when it is decreasing.
CTR_max	Time-base counter equal max value. (TBCTR = 0xFFFF) Generated event when the TBCTR value reaches its maximum value. This signal is only used only as a status bit
TBCLK	Time-base clock. This is a prescaled version of the system clock (VCLK3) and is used by all submodules within the ePWM. This clock determines the rate at which time-base counter increments or decrements.

35.2.2.3 Calculating PWM Period and Frequency

The frequency of PWM events is controlled by the time-base period (TBPRD) register and the mode of the time-base counter. Figure 35-5 shows the period (T_{pwm}) and frequency (F_{pwm}) relationships for the up-count, down-count, and up-down-count time-base counter modes when the period is set to 4 (TBPRD = 4). The time increment for each step is defined by the time-base clock (TBCLK) which is a prescaled version of the system clock (VCLK3).

The time-base counter has three modes of operation selected by the time-base control register (TBCTL):

- **Up-Down-Count Mode:**

  In up-down-count mode, the time-base counter starts from zero and increments until the period (TBPRD) value is reached. When the period value is reached, the time-base counter then decrements until it reaches zero. At this point the counter repeats the pattern and begins to increment.
- **Up-Count Mode:**

  In this mode, the time-base counter starts from zero and increments until it reaches the value in the period register (TBPRD). When the period value is reached, the time-base counter resets to zero and begins to increment once again.
- **Down-Count Mode:**

  In down-count mode, the time-base counter starts from the period (TBPRD) value and decrements until it reaches zero. When it reaches zero, the time-base counter is reset to the period value and it begins to decrement once again.

35.2.2.3.1 Time-Base Period Shadow Register

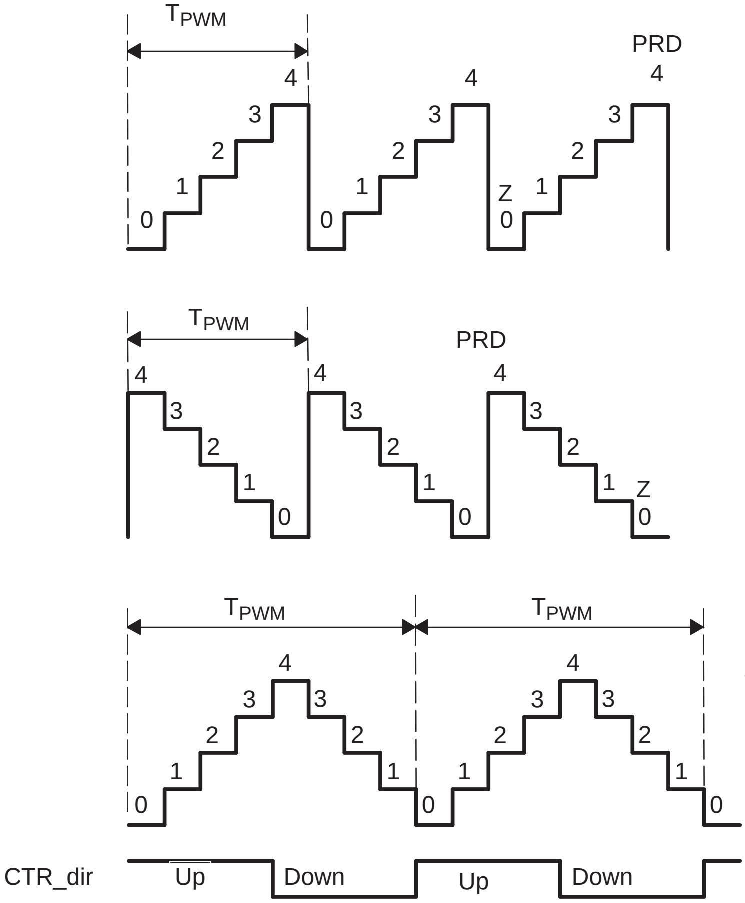
The time-base period register (TBPRD) has a shadow register. Shadowing allows the register update to be synchronized with the hardware. The following definitions are used to describe all shadow registers in the ePWM module:

- **Active Register**

  The active register controls the hardware and is responsible for actions that the hardware causes or invokes.
- **Shadow Register**

  The shadow register buffers or provides a temporary holding location for the active register. It has no direct effect on any control hardware. At a strategic point in time the shadow register's content is transferred to the active register. This prevents corruption or spurious operation due to the register being asynchronously modified by software.

Figure 35-5. Time-Base Frequency and Period



For Up Count and Down Count

$T_{PWM} = (TBPRD + 1) \times T_{TBCLK}$

$F_{PWM} = 1/(T_{PWM})$

For Up and Down Count

$T_{PWM} = 2 \times TBPRD \times T_{TBCLK}$

$F_{PWM} = 1/(T_{PWM})$

The memory address of the shadow period register is the same as the active register. Which register is written to or read from is determined by the TBCTL[PRDLD] bit. This bit enables and disables the TBPRD shadow register as follows:

- **Time-Base Period Shadow Mode:**

  The TBPRD shadow register is enabled when TBCTL[PRDLD] = 0. Reads from and writes to the TBPRD memory address go to the shadow register. The shadow register contents are transferred to the active register (TBPRD (Active) ← TBPRD (shadow)) when the time-base counter equals zero (TBCTR = 0x0000). By default the TBPRD shadow register is enabled.

- **Time-Base Period Immediate Load Mode:**

  If immediate load mode is selected (TBCTL[PRDLD] = 1), then a read from or a write to the TBPRD memory address goes directly to the active register.

35.2.2.3.2 Time-Base Clock Synchronization

Bit 1 of the device-level multiplexing control module (IOMM) register PINMMR166 is defined as the TBCLKSYNC bit. The TBCLKSYNC bit allows users to globally synchronize all enabled ePWM modules to the time-base clock (TBCLK). When set, all enabled ePWM module clocks are started with the first rising edge of TBCLK aligned. For perfectly synchronized TBCLKs, the prescalers for each ePWM module must be set identically.

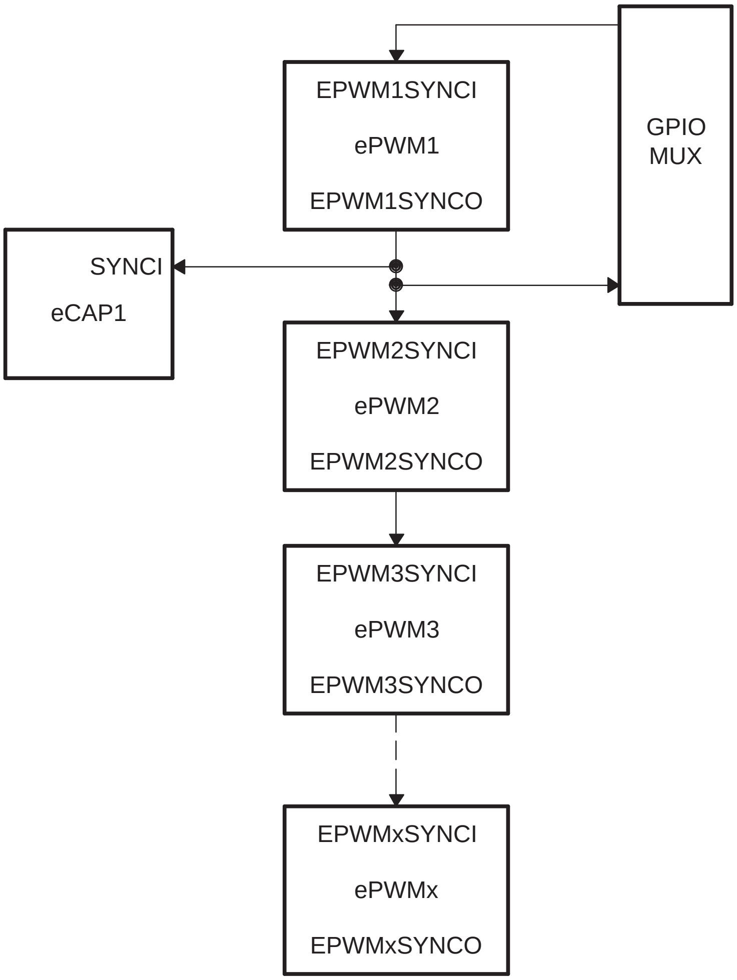
The proper procedure for enabling ePWM clocks is as follows:

1. Enable ePWM module clocks using the IOMM control registers for each ePWM module instance
2. Set TBCLKSYNC = 0. This will stop the time-base clock within any enabled ePWM module.
3. Configure ePWM modules: prescaler values and ePWM modes.
4. Set TBCLKSYNC = 1.

35.2.2.3.3 Time-Base Counter Synchronization

A time-base synchronization scheme connects all of the ePWM modules on a device. Each ePWM module has a synchronization input (EPWMxSYNCI) and a synchronization output (EPWMxSYNCO). The input synchronization for the first instance (ePWM1) comes from an external pin. The synchronization connections for the remaining ePWM modules are shown in Figure 35-6.

Figure 35-6. Time-Base Counter Synchronization Scheme



Each ePWM module can be configured to use or ignore the synchronization input. If the TBCTL[PHSEN] bit is set, then the time-base counter (TBCTR) of the ePWM module will be automatically loaded with the phase register (TBPHS) contents when one of the following conditions occur:

- **EPWMxSYNCI: Synchronization Input Pulse:**

  The value of the phase register is loaded into the counter register when an input synchronization pulse is detected (TBPHS → TBCTR). This operation occurs on the next valid time-base clock (TBCLK) edge.

  The delay from internal master module to slave modules is given by:

  – if (TBCLK = VCLK3): 2 x VCLK3

  – if (TBCLK != VCLK3): 1 TBCLK
- **Software Forced Synchronization Pulse:**

  Writing a 1 to the TBCTL[SWFSYNC] control bit invokes a software forced synchronization. This pulse is ORed with the synchronization input signal, and therefore has the same effect as a pulse on EPWMxSYNCI.
- **Digital Compare Event Synchronization Pulse:**

  DCAEVT1 and DCBEVT1 digital compare events can be configured to generate synchronization pulses which have the same affect as EPWMxSYNCI.

This feature enables the ePWM module to be automatically synchronized to the time base of another ePWM module. Lead or lag phase control can be added to the waveforms generated by different ePWM modules to synchronize them. In up-down-count mode, the TBCTL[PSHDIR] bit configures the direction of the time-base counter immediately after a synchronization event. The new direction is independent of the direction prior to the synchronization event. The PHSDIR bit is ignored in count-up or count-down modes. See Figure 35-7 through Figure 35-10 for examples.

Clearing the TBCTL[PHSEN] bit configures the ePWM to ignore the synchronization input pulse. The synchronization pulse can still be allowed to flow-through to the EPWMxSYNCO and be used to synchronize other ePWM modules. In this way, you can set up a master time-base (for example, ePWM1) and downstream modules (ePWM2 - ePWMx) may elect to run in synchronization with the master.

35.2.2.4 Phase Locking the Time-Base Clocks of Multiple ePWM Modules

The TBCLKSYNC bit can be used to globally synchronize the time-base clocks of all enabled ePWM modules on a device. When TBCLKSYNC = 0, the time-base clock of all ePWM modules is stopped (default). When TBCLKSYNC = 1, all ePWM time-base clocks are started with the rising edge of TBCLK aligned. For perfectly synchronized TBCLKs, the prescaler bits in the TBCTL register of each ePWM module must be set identically. The proper procedure for enabling the ePWM clocks is as follows:

1. Enable ePWM module clocks using the IOMM control registers for each ePWM module instance
2. Set TBCLKSYNC= 0. This will stop the time-base clock within any enabled ePWM module.
3. Configure ePWM modules: prescaler values and ePWM modes.
4. Set TBCLKSYNC=1.

35.2.2.5 Time-base Counter Modes and Timing Waveforms

The time-base counter operates in one of four modes:

- Up-count mode which is asymmetrical.
- Down-count mode which is asymmetrical.
- Up-down-count which is symmetrical
- Frozen where the time-base counter is held constant at the current value

To illustrate the operation of the first three modes, the following timing diagrams show when events are generated and how the time-base responds to an EPWMxSYNCI signal.

Figure 35-7. Time-Base Up-Count Mode Waveforms

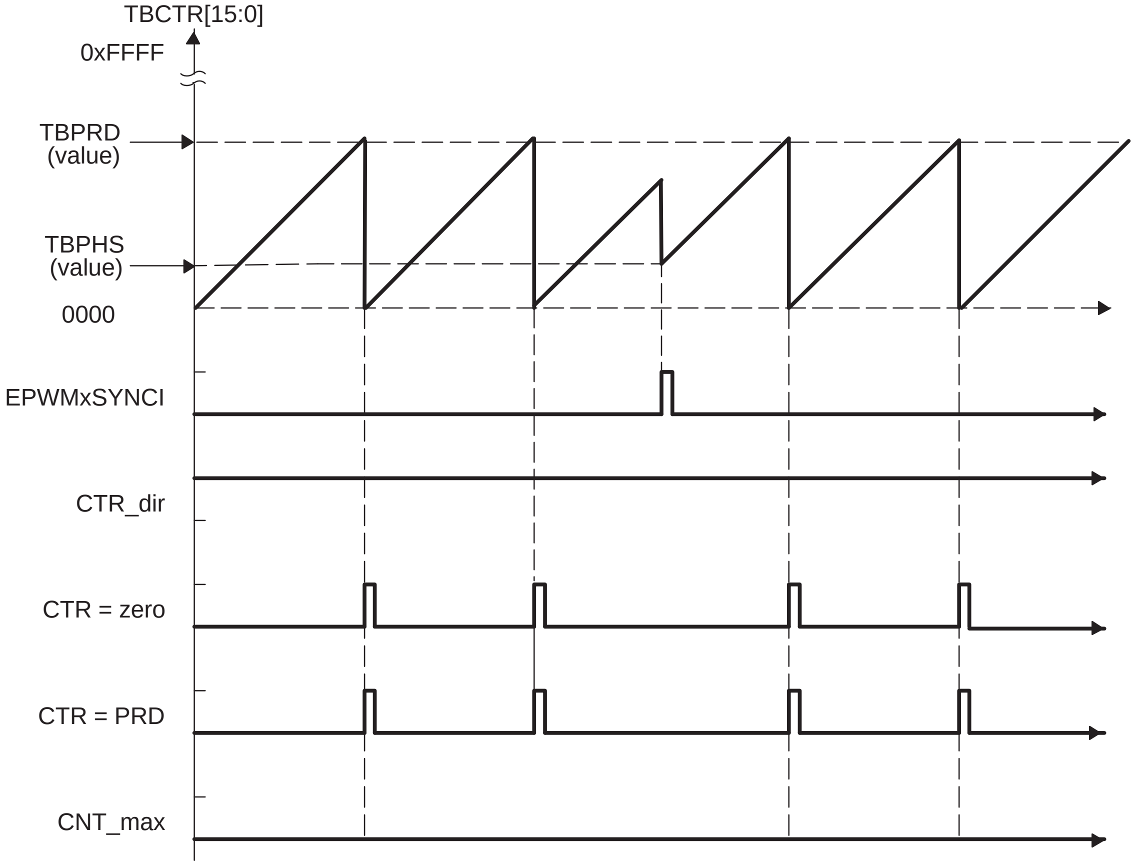


Figure 35-8. Time-Base Down-Count Mode Waveforms

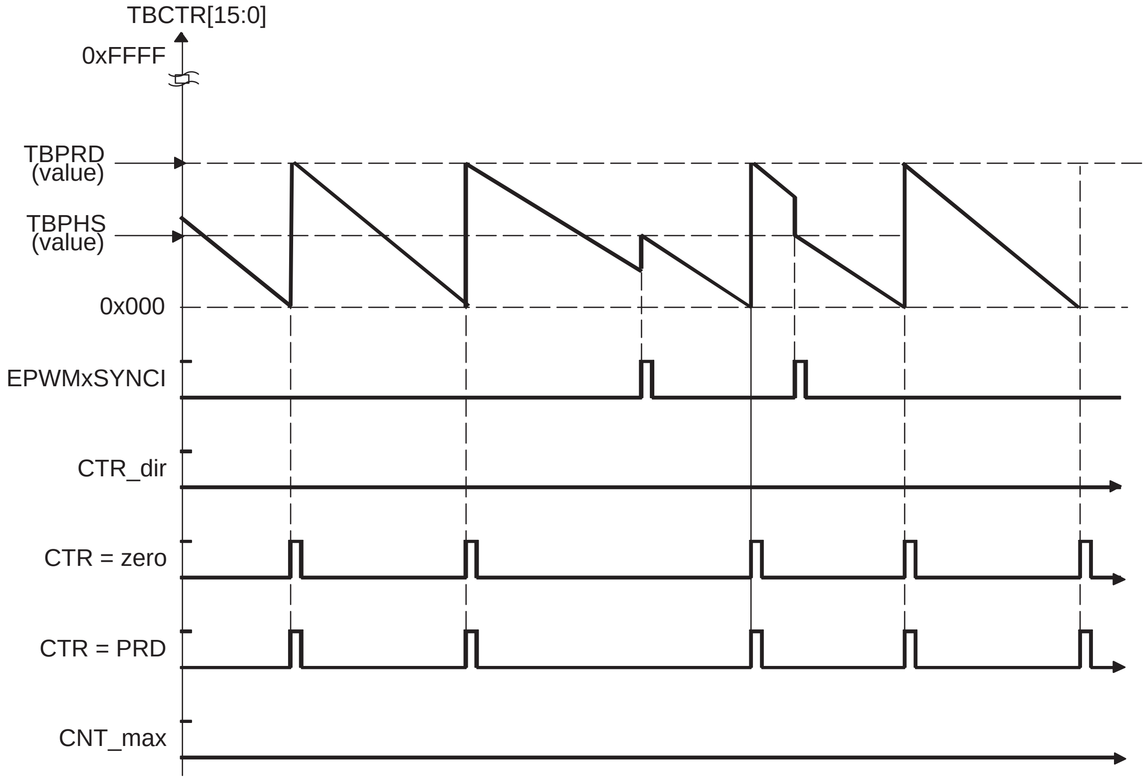


Figure 35-9. Time-Base Up-Down-Count Waveforms, TBCTL[PHSDIR = 0] Count Down On Synchronization Event

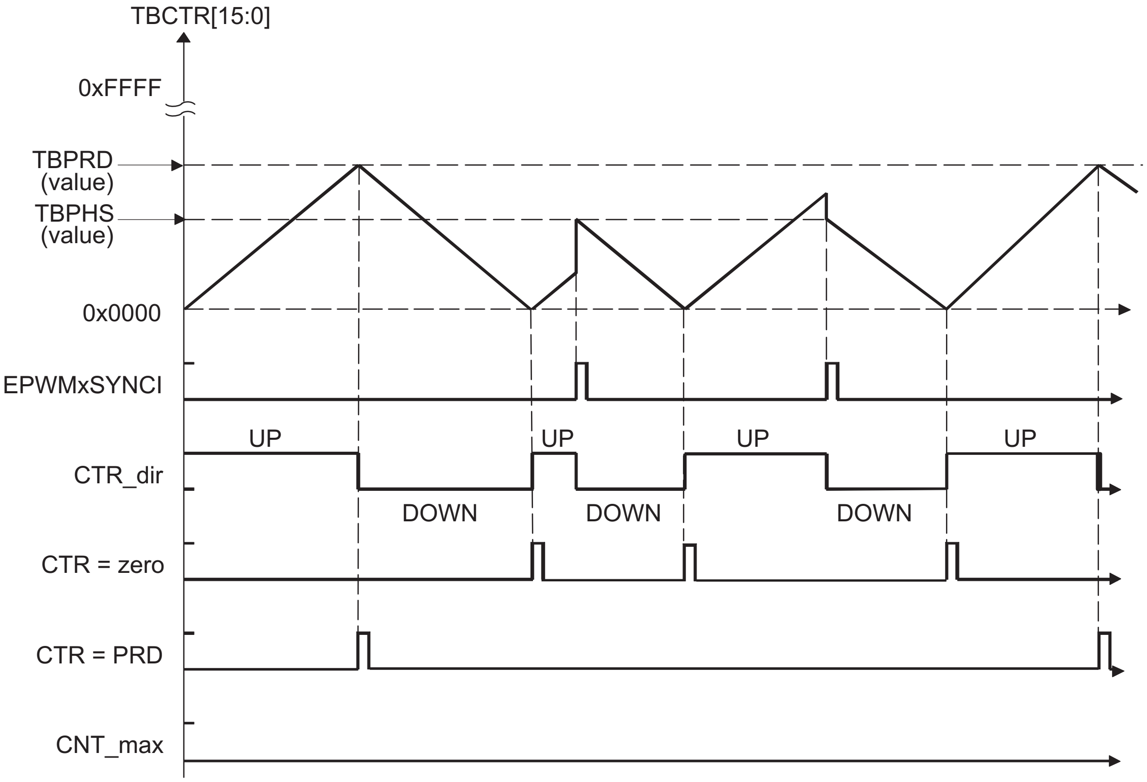
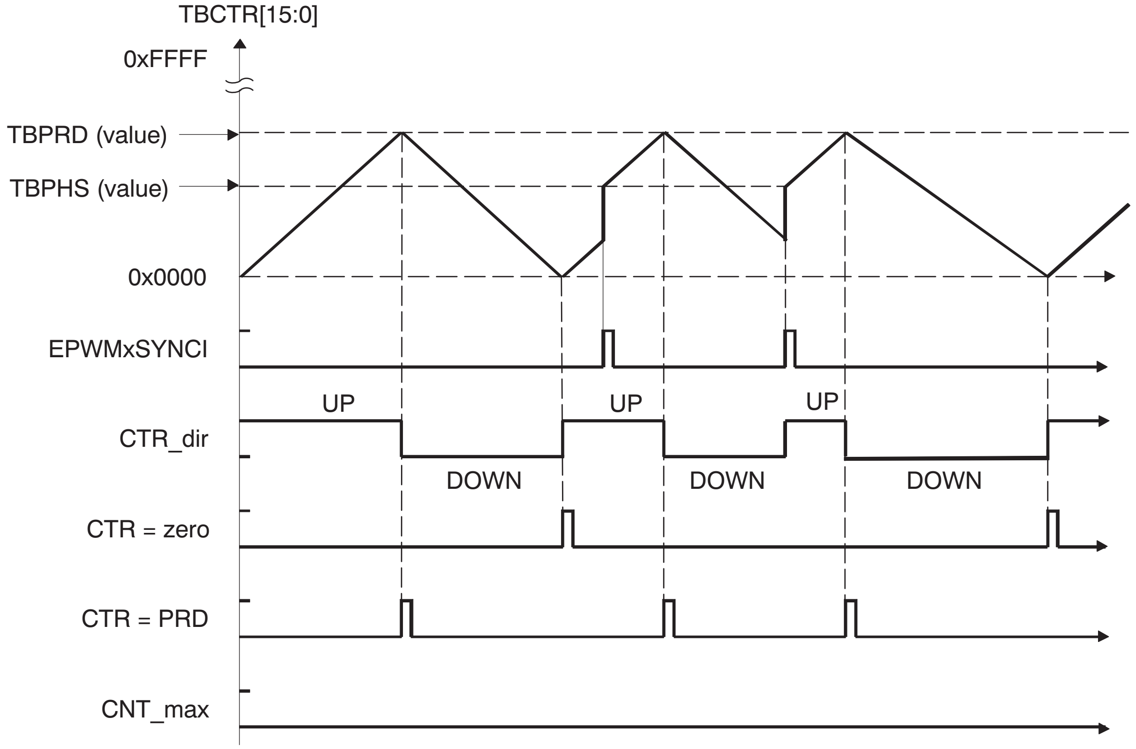


Figure 35-10. Time-Base Up-Down Count Waveforms, TBCTL[PHSDIR = 1] Count Up On Synchronization Event



35.2.3 Counter-Compare (CC) Submodule

Figure 35-11 illustrates the counter-compare submodule within the ePWM.

Figure 35-12 shows the basic structure of the counter-compare submodule.

Figure 35-11. Counter-Compare Submodule

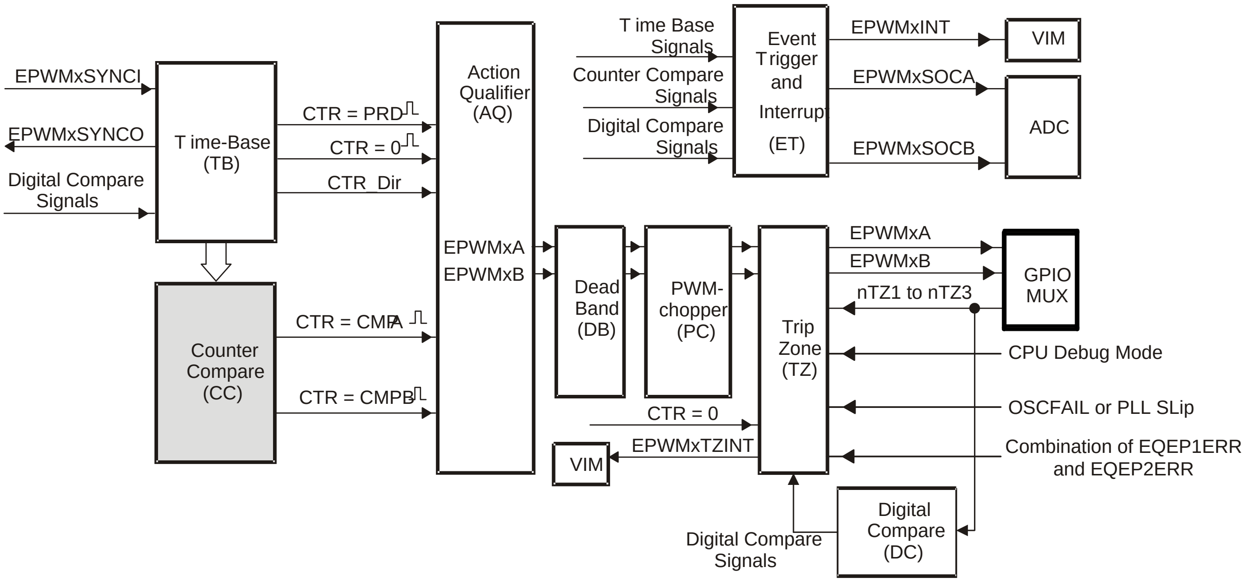
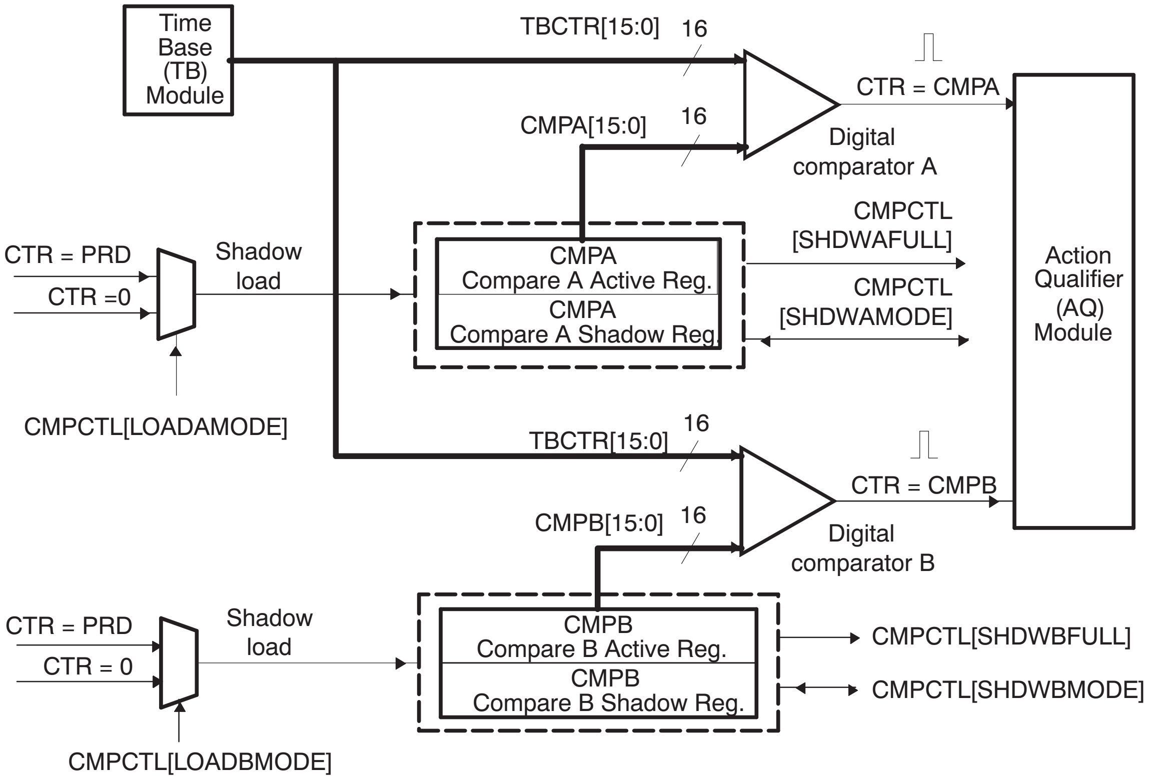


Figure 35-12. Detailed View of the Counter-Compare Submodule



35.2.3.1 Purpose of the Counter-Compare Submodule

The counter-compare submodule takes as input the time-base counter value. This value is continuously compared to the counter-compare A (CMPA) and counter-compare B (CMPB) registers. When the time-base counter is equal to one of the compare registers, the counter-compare unit generates an appropriate event.

The counter-compare:

- Generates events based on programmable time stamps using the CMPA and CMPB registers
  - CTR = CMPA: Time-base counter equals counter-compare A register (TBCTR = CMPA).
  - CTR = CMPB: Time-base counter equals counter-compare B register (TBCTR = CMPB)
- Controls the PWM duty cycle if the action-qualifier submodule is configured appropriately
- Shadows new compare values to prevent corruption or glitches during the active PWM cycle

35.2.3.2 Controlling and Monitoring the Counter-Compare Submodule

The counter-compare submodule operation is controlled and monitored by the registers listed in Table 35-5.

The key signals associated with the counter-compare submodule are described in Table 35-6.

Table 35-5. Counter-Compare Submodule Registers

Register Name	Address Offset	Shadowed	Description
CMPCTL	0Ch	No	Counter-Compare Control Register.
CMPA	10h	Yes	Counter-Compare A Register
CMPB	16h	Yes	Counter-Compare B Register

Table 35-6. Counter-Compare Submodule Key Signals

Signal	Description of Event	Registers Compared
CTR = CMPA	Time-base counter equal to the active counter-compare A value	TBCTR = CMPA
CTR = CMPB	Time-base counter equal to the active counter-compare B value	TBCTR = CMPB
CTR = PRD	Time-base counter equal to the active period. Used to load active counter-compare A and B registers from the shadow register	TBCTR = TBPRD
CTR = ZERO	Time-base counter equal to zero. Used to load active counter-compare A and B registers from the shadow register	TBCTR = 0x0000

35.2.3.3 Operational Highlights for the Counter-Compare Submodule

The counter-compare submodule is responsible for generating two independent compare events based on two compare registers:

1. CTR = CMPA: Time-base counter equal to counter-compare A register (TBCTR = CMPA).
2. CTR = CMPB: Time-base counter equal to counter-compare B register (TBCTR = CMPB).

For up-count or down-count mode, each event occurs only once per cycle. For up-down-count mode each event occurs twice per cycle if the compare value is between 0x0000-TBPRD and once per cycle if the compare value is equal to 0x0000 or equal to TBPRD. These events are fed into the action-qualifier submodule where they are qualified by the counter direction and converted into actions if enabled. Refer to Section 35.2.4.1 for more details.

The counter-compare registers CMPA and CMPB each have an associated shadow register. Shadowing provides a way to keep updates to the registers synchronized with the hardware. When shadowing is used, updates to the active registers only occur at strategic points. This prevents corruption or spurious operation due to the register being asynchronously modified by software. The memory address of the active register and the shadow register is identical. Which register is written to or read from is determined by the CMPCTL[SHDWAMODE] and CMPCTL[SHDWBMODE] bits. These bits enable and disable the CMPA shadow register and CMPB shadow register respectively. The behavior of the two load modes is described below:

Shadow Mode:

The shadow mode for the CMPA is enabled by clearing the CMPCTL[SHDWAMODE] bit and the shadow register for CMPB is enabled by clearing the CMPCTL[SHDWBMODE] bit. Shadow mode is enabled by default for both CMPA and CMPB.

If the shadow register is enabled then the content of the shadow register is transferred to the active register on one of the following events as specified by the CMPCTL[LOADAMODE] and CMPCTL[LOADBMODE] register bits:

- CTR = PRD: Time-base counter equal to the period (TBCTR = TBPRD).
- CTR = Zero: Time-base counter equal to zero (TBCTR = 0x0000)
- Both CTR = PRD and CTR = Zero

Only the active register contents are used by the counter-compare submodule to generate events to be sent to the action-qualifier.

Immediate Load Mode:

If immediate load mode is selected (TBCTL[SHADWAMODE] = 1 or TBCTL[SHADWBMODE] = 1), then a read from or a write to the register will go directly to the active register.

35.2.3.4 Count Mode Timing Waveforms

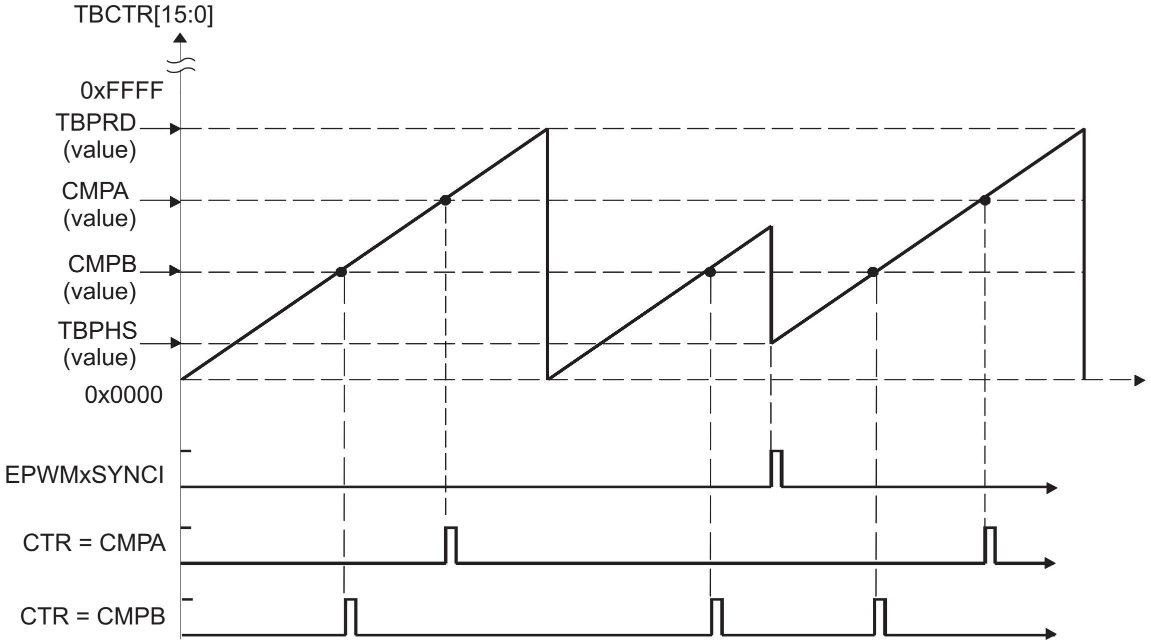
The counter-compare module can generate compare events in all three count modes:

- Up-count mode: used to generate an asymmetrical PWM waveform.
- Down-count mode: used to generate an asymmetrical PWM waveform.

- Up-down-count mode: used to generate a symmetrical PWM waveform.

To illustrate the operation of the first three modes, the timing diagrams in Figure 35-13 through Figure 35-16 show when events are generated and how the EPWMxSYNCI signal interacts.

Figure 35-13. Counter-Compare Event Waveforms in Up-Count Mode



NOTE: An EPWMxSYNCI external synchronization event can cause a discontinuity in the TBCTR count sequence. This can lead to a compare event being skipped. This skipping is considered normal operation and must be taken into account.

Figure 35-14. Counter-Compare Events in Down-Count Mode

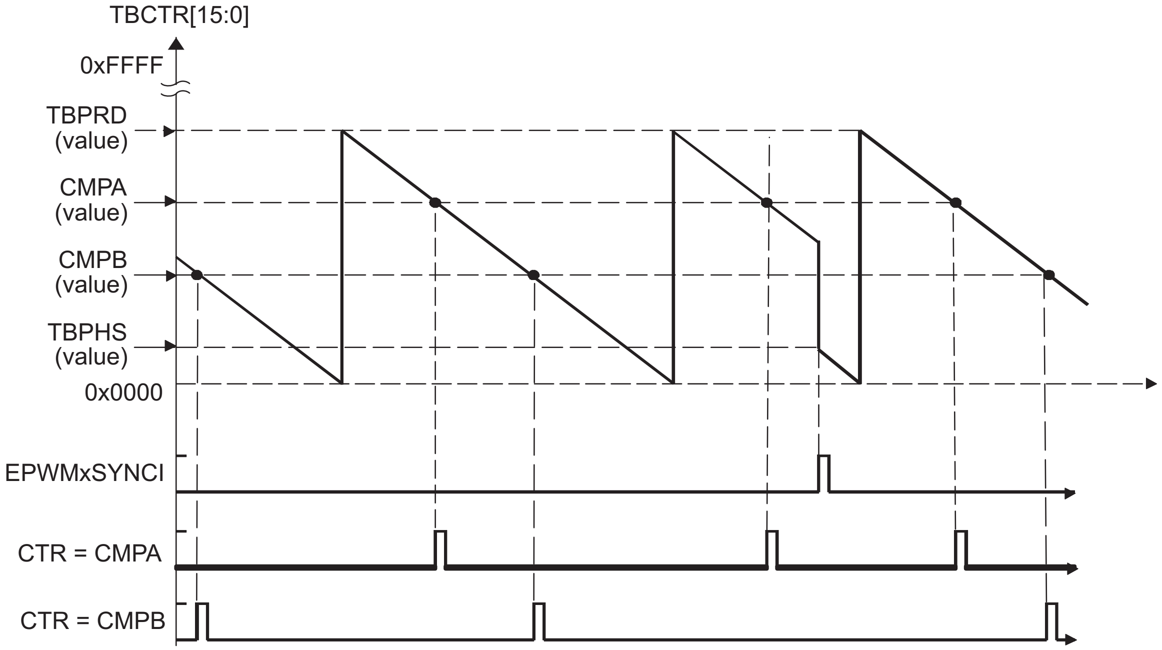


Figure 35-15. Counter-Compare Events In Up-Down-Count Mode, TBCTL[PHSDIR = 0] Count Down On Synchronization Event

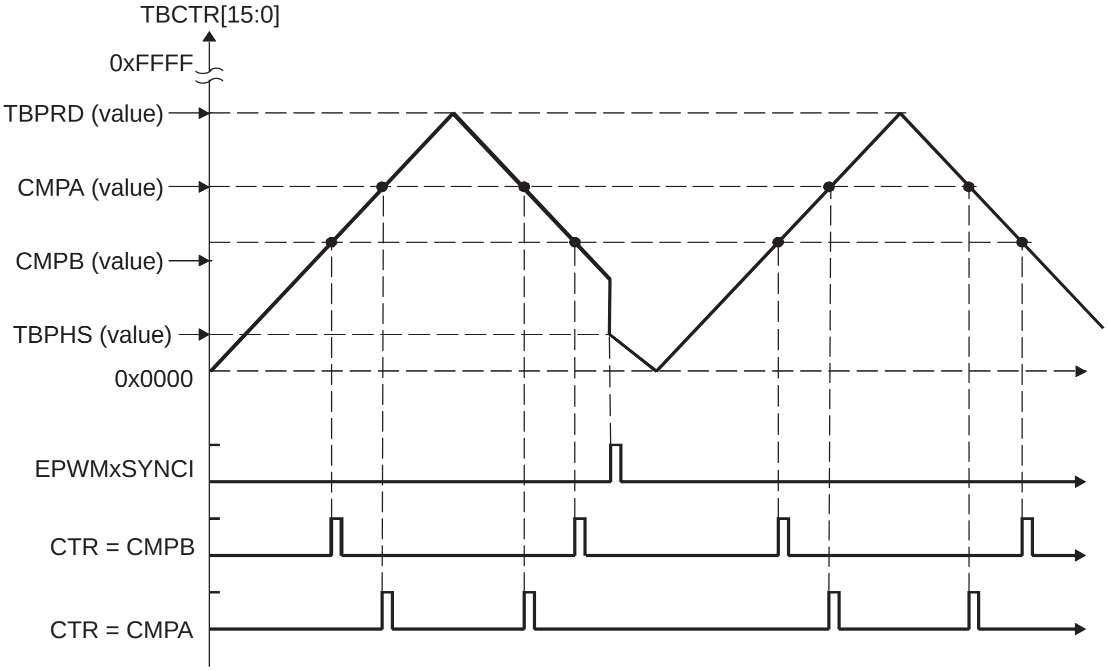
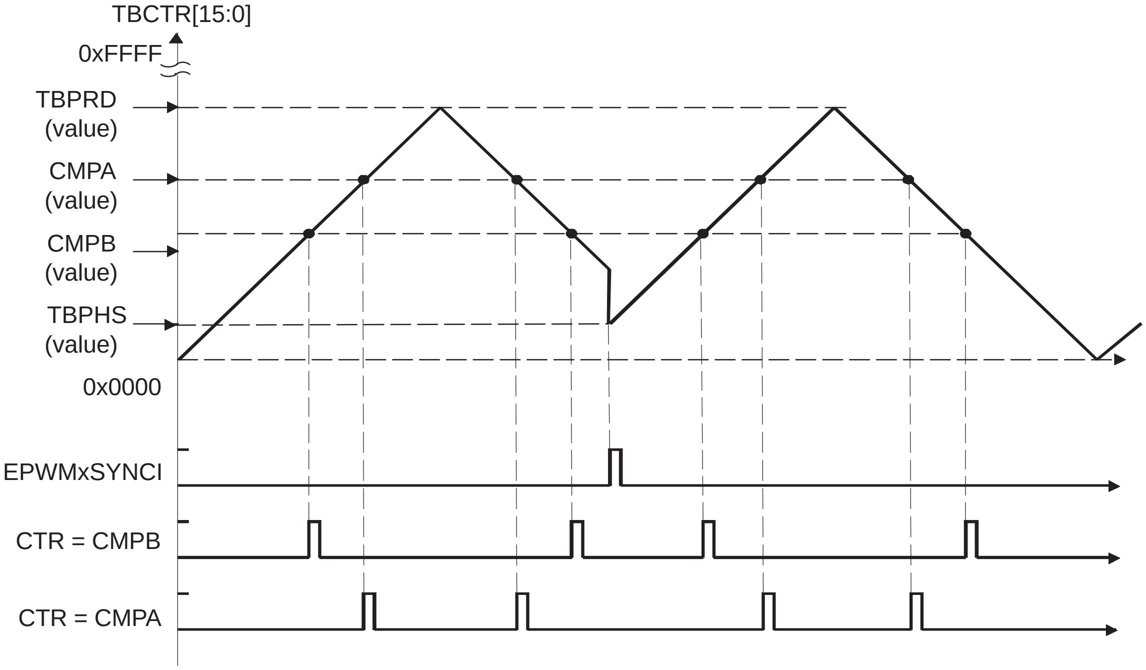


Figure 35-16. Counter-Compare Events In Up-Down-Count Mode, TBCTL[PHSDIR = 1] Count Up On Synchronization Event

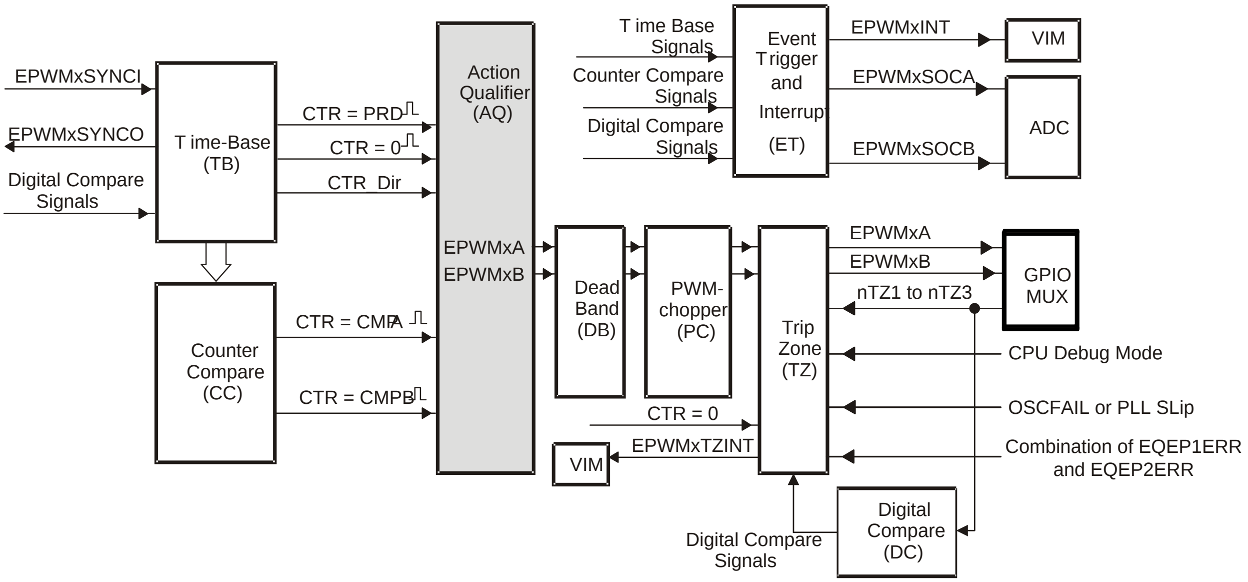


35.2.4 Action-Qualifier (AQ) Submodule

Figure 35-17 shows the action-qualifier (AQ) submodule (see shaded block) in the ePWM system.

The action-qualifier submodule has the most important role in waveform construction and PWM generation. It decides which events are converted into various action types, thereby producing the required switched waveforms at the EPWMxA and EPWMxB outputs.

Figure 35-17. Action-Qualifier Submodule



35.2.4.1 Purpose of the Action-Qualifier Submodule

The action-qualifier submodule is responsible for the following:

- Qualifying and generating actions (set, clear, toggle) based on the following events:
  - CTR = PRD: Time-base counter equal to the period (TBCTR = TBPRD).
  - CTR = Zero: Time-base counter equal to zero (TBCTR = 0x0000)
  - CTR = CMPA: Time-base counter equal to the counter-compare A register (TBCTR = CMPA)
  - CTR = CMPB: Time-base counter equal to the counter-compare B register (TBCTR = CMPB)
- Managing priority when these events occur concurrently
- Providing independent control of events when the time-base counter is increasing and when it is decreasing.

35.2.4.2 Action-Qualifier Submodule Control and Status Register Definitions

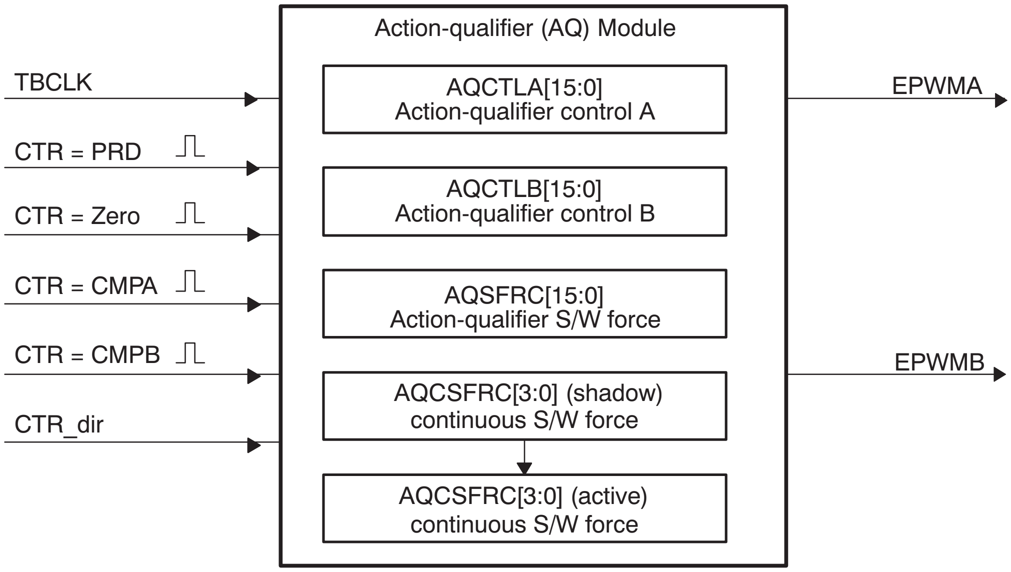
The action-qualifier submodule operation is controlled and monitored via the registers in Table 35-7.

Table 35-7. Action-Qualifier Submodule Registers

Register Name	Address Offset	Shadowed	Description
AQCTLA	14h	No	Action-Qualifier Control Register For Output A (EPWMxA)
AQSFRC	18h	No	Action-Qualifier Software Force Register
AQCTLB	1Ah	No	Action-Qualifier Control Register For Output B (EPWMxB)
AQCSFRC	1Eh	Yes	Action-Qualifier Continuous Software Force

The action-qualifier submodule is based on event-driven logic. It can be thought of as a programmable cross switch with events at the input and actions at the output, all of which are software controlled via the set of registers shown in Table 35-7.

Figure 35-18. Action-Qualifier Submodule Inputs and Outputs



The possible input events are summarized again in Table 35-8.

Table 35-8. Action-Qualifier Submodule Possible Input Events

Signal	Description	Registers Compared
CTR = PRD	Time-base counter equal to the period value	TBCTR = TBPRD
CTR = Zero	Time-base counter equal to zero	TBCTR = 0x0000
CTR = CMPA	Time-base counter equal to the counter-compare A	TBCTR = CMPA
CTR = CMPB	Time-base counter equal to the counter-compare B	TBCTR = CMPB
Software forced event	Asynchronous event initiated by software	

The software forced action is a useful asynchronous event. This control is handled by registers AQSFRC and AQCSFRC.

The action-qualifier submodule controls how the two outputs EPWMxA and EPWMxB behave when a particular event occurs. The event inputs to the action-qualifier submodule are further qualified by the counter direction (up or down). This allows for independent action on outputs on both the count-up and count-down phases.

The possible actions imposed on outputs EPWMxA and EPWMxB are:

- **Set High:**

  Set output EPWMxA or EPWMxB to a high level.
- **Clear Low:**

  Set output EPWMxA or EPWMxB to a low level.
- **Toggle:**

  If EPWMxA or EPWMxB is currently pulled high, then pull the output low. If EPWMxA or EPWMxB is currently pulled low, then pull the output high.

- **Do Nothing:**

  Keep outputs EPWMxA and EPWMxB at same level as currently set. Although the "Do Nothing" option prevents an event from causing an action on the EPWMxA and EPWMxB outputs, this event can still trigger interrupts and ADC start of conversion. See the Event-trigger Submodule description in Section 35.2.8 for details.

Actions are specified independently for either output (EPWMxA or EPWMxB). Any or all events can be configured to generate actions on a given output. For example, both CTR = CMPA and CTR = CMPB can operate on output EPWMxA. All qualifier actions are configured via the control registers found at the end of this section.

For clarity, the drawings in this document use a set of symbolic actions. These symbols are summarized in Figure 35-19. Each symbol represents an action as a marker in time. Some actions are fixed in time (zero and period) while the CMPA and CMPB actions are moveable and their time positions are programmed via the counter-compare A and B registers, respectively. To turn off or disable an action, use the "Do Nothing option"; it is the default at reset.

Figure 35-19. Possible Action-Qualifier Actions for EPWMxA and EPWMxB Outputs

S/W force	TB Counter equals:				Actions
	Zero	Comp A	Comp B	Period	
SW ✕	Z ✕	CA ✕	CB ✕	P ✕	Do Nothing
SW ↓	Z ↓	CA ↓	CB ↓	P ↓	Clear Low
SW ↑	Z ↑	CA ↑	CB ↑	P ↑	Set High
SW T	Z T	CA T	CB T	P T	Toggle

35.2.4.3 Action-Qualifier Event Priority

It is possible for the ePWM action qualifier to receive more than one event at the same time. In this case events are assigned a priority by the hardware. The general rule is events occurring later in time have a higher priority and software forced events always have the highest priority. The event priority levels for up-down-count mode are shown in Table 35-9. A priority level of 1 is the highest priority and level 7 is the lowest. The priority changes slightly depending on the direction of TBCTR.

Table 35-9. Action-Qualifier Event Priority for Up-Down-Count Mode

Priority Level	Event If TBCTR is Incrementing TBCTR = Zero up to TBCTR = TBPRD	Event If TBCTR is Decrementing TBCTR = TBPRD down to TBCTR = 1
1 (Highest)	Software forced event	Software forced event
2	Counter equals CMPB on up-count (CBU)	Counter equals CMPB on down-count (CBD)
3	Counter equals CMPA on up-count (CAU)	Counter equals CMPA on down-count (CAD)
4	Counter equals zero	Counter equals period (TBPRD)
5	Counter equals CMPB on down-count (CBD)	Counter equals CMPB on up-count (CBU)
6 (Lowest)	Counter equals CMPA on down-count (CAD)	Counter equals CMPA on up-count (CBU)

Table 35-10 shows the action-qualifier priority for up-count mode. In this case, the counter direction is always defined as up and thus down-count events will never be taken.

Table 35-10. Action-Qualifier Event Priority for Up-Count Mode

Priority Level	Event
1 (Highest)	Software forced event
2	Counter equal to period (TBPRD)
3	Counter equal to CMPB on up-count (CBU)
4	Counter equal to CMPA on up-count (CAU)
5 (Lowest)	Counter equal to Zero

Table 35-11 shows the action-qualifier priority for down-count mode. In this case, the counter direction is always defined as down and thus up-count events will never be taken.

Table 35-11. Action-Qualifier Event Priority for Down-Count Mode

Priority Level	Event
1 (Highest)	Software forced event
2	Counter equal to Zero
3	Counter equal to CMPB on down-count (CBD)
4	Counter equal to CMPA on down-count (CAD)
5 (Lowest)	Counter equal to period (TBPRD)

It is possible to set the compare value greater than the period. In this case the action will take place as shown in Table 35-12.

Table 35-12. Behavior if CMPA/CMPB is Greater than the Period

Counter Mode	Compare on Up-Count Event CAD/CBD	Compare on Down-Count Event CAD/CBD
Up-Count Mode	If CMPA/CMPB ≤ TBPRD period, then the event occurs on a compare match (TBCTR=CMPA or CMPB). If CMPA/CMPB > TBPRD, then the event will not occur.	Never occurs.

Table 35-12. Behavior if CMPA/CMPB is Greater than the Period (continued)

Counter Mode	Compare on Up-Count Event CAD/CBD	Compare on Down-Count Event CAD/CBD
Down-Count Mode	Never occurs.	If CMPA/CMPB < TBPRD, the event will occur on a compare match (TBCTR=CMPA or CMPB). If CMPA/CMPB ≥ TBPRD, the event will occur on a period match (TBCTR=TBPRD).
Up-Down-Count Mode	If CMPA/CMPB < TBPRD and the counter is incrementing, the event occurs on a compare match (TBCTR=CMPA or CMPB).	If CMPA/CMPB < TBPRD and the counter is decrementing, the event occurs on a compare match (TBCTR=CMPA or CMPB).
	If CMPA/CMPB is ≥ TBPRD, the event will occur on a period match (TBCTR = TBPRD).	If CMPA/CMPB ≥ TBPRD, the event occurs on a period match (TBCTR=TBPRD).

35.2.4.4 Waveforms for Common Configurations

NOTE: The waveforms in this document show the ePWMs behavior for a static compare register value. In a running system, the active compare registers (CMPA and CMPB) are typically updated from their respective shadow registers once every period. The user specifies when the update will take place; either when the time-base counter reaches zero or when the time-base counter reaches period. There are some cases when the action based on the new value can be delayed by one period or the action based on the old value can take effect for an extra period. Some PWM configurations avoid this situation. These include, but are not limited to, the following:

Use up-down-count mode to generate a symmetric PWM:

- If you load CMPA/CMPB on zero, then use CMPA/CMPB values greater than or equal to 1.
- If you load CMPA/CMPB on period, then use CMPA/CMPB values less than or equal to TBPRD-1.

  This means there will always be a pulse of at least one TBCLK cycle in a PWM period which, when very short, tend to be ignored by the system.

Use up-down-count mode to generate an asymmetric PWM:

- To achieve 50%-0% asymmetric PWM use the following configuration: Load CMPA/CMPB on period and use the period action to clear the PWM and a compare-up action to set the PWM. Modulate the compare value from 0 to TBPRD to achieve 50%-0% PWM duty.

When using up-count mode to generate an asymmetric PWM:

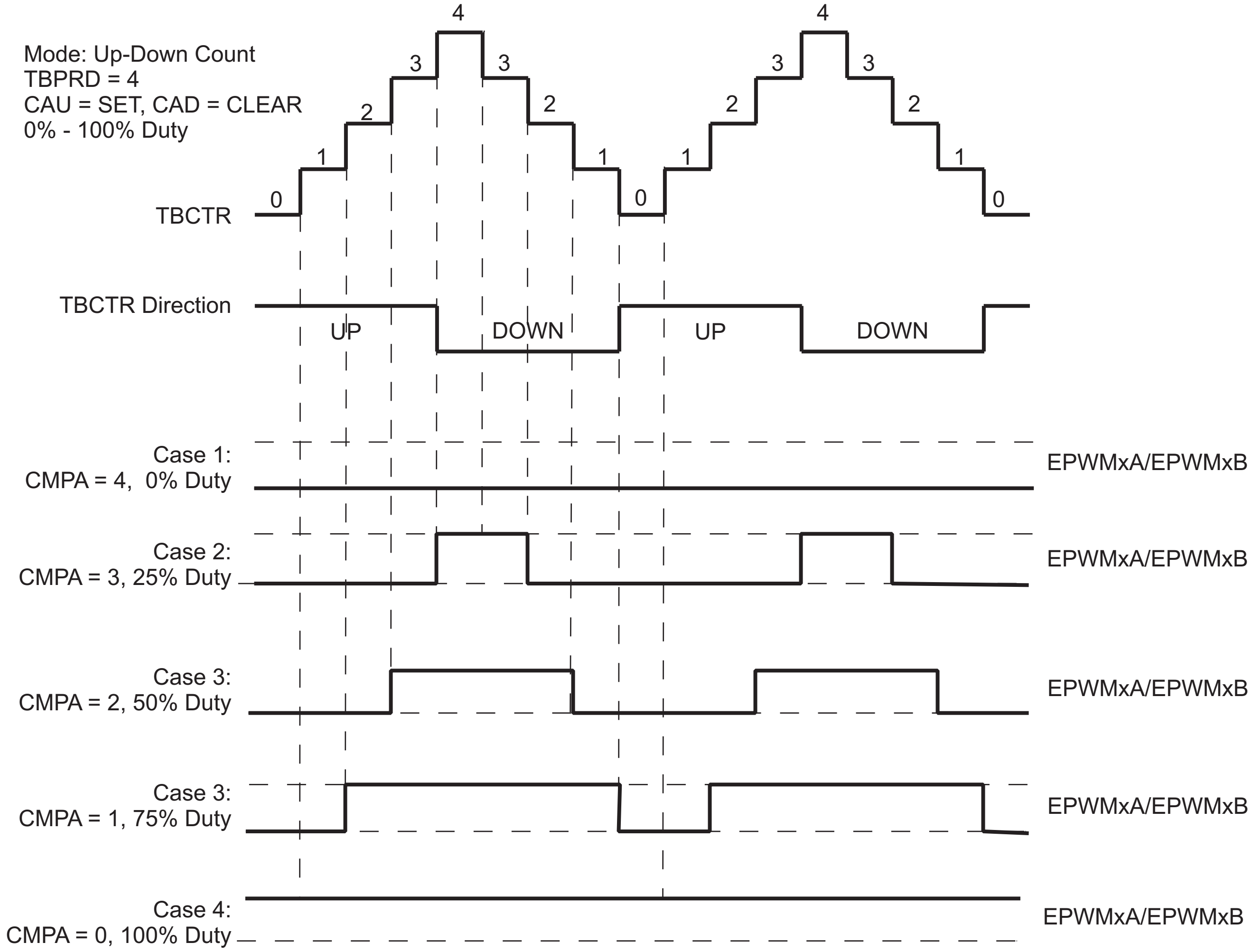
- To achieve 0-100% asymmetric PWM use the following configuration: Load CMPA/CMPB on TBPRD. Use the Zero action to set the PWM and a compare-up action to clear the PWM. Modulate the compare value from 0 to TBPRD+1 to achieve 0-100% PWM duty.

See the *Using Enhanced Pulse Width Modulator (ePWM) Module for 0-100% Duty Cycle Control* Application Report (literature number SPRAAI1)

Figure 35-20 shows how a symmetric PWM waveform can be generated using the up-down-count mode of the TBCTR. In this mode 0%-100% DC modulation is achieved by using equal compare matches on the up count and down count portions of the waveform. In the example shown, CMPA is used to make the comparison. When the counter is incrementing the CMPA match will pull the PWM output high. Likewise, when the counter is decrementing the compare match will pull the PWM signal low. When CMPA = 0, the PWM signal is low for the entire period giving the 0% duty waveform. When CMPA = TBPRD, the PWM signal is high achieving 100% duty.

When using this configuration in practice, if you load CMPA/CMPB on zero, then use CMPA/CMPB values greater than or equal to 1. If you load CMPA/CMPB on period, then use CMPA/CMPB values less than or equal to TBPRD-1. This means there will always be a pulse of at least one TBCLK cycle in a PWM period which, when very short, tend to be ignored by the system.

Figure 35-20. Up-Down-Count Mode Symmetrical Waveform

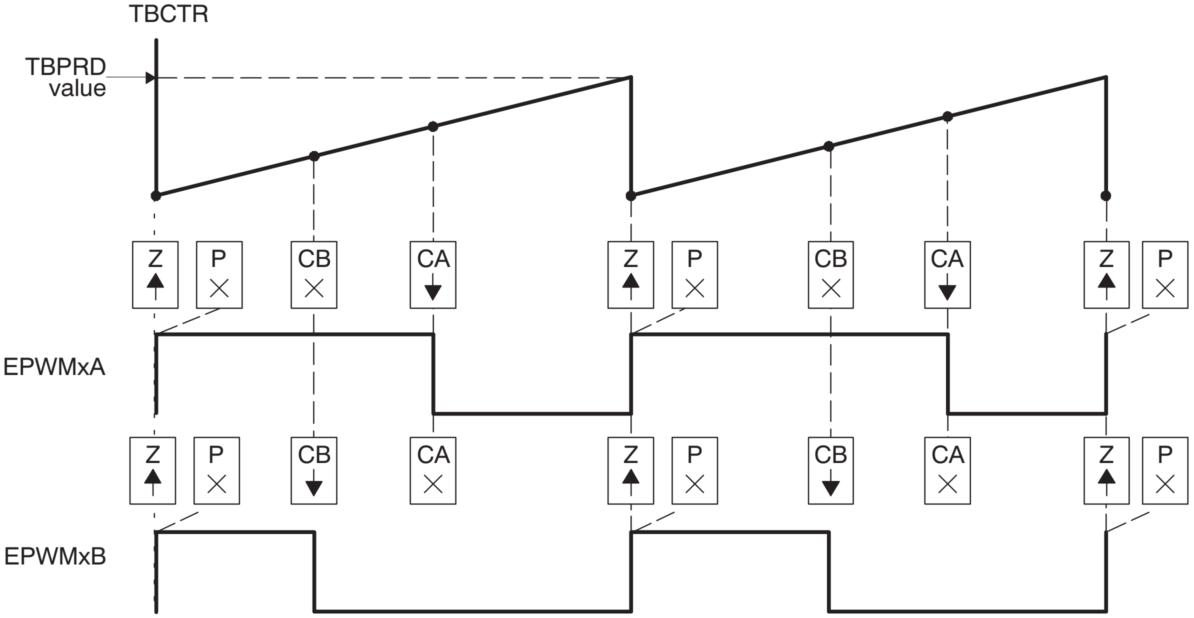


The PWM waveforms in Figure 35-21 through Figure 35-26 show some common action-qualifier configurations. The C-code samples in Example 35-1 through Example 35-6 shows how to configure an ePWM module for each case. Some conventions used in the figures and examples are as follows:

- TBPRD, CMPA, and CMPB refer to the value written in their respective registers. The active register, not the shadow register, is used by the hardware.
- CMPx, refers to either CMPA or CMPB.
- EPWMxA and EPWMxB refer to the output signals from ePWMx
- Up-Down means Count-up-and-down mode, Up means up-count mode and Dwn means down-count mode
- Sym = Symmetric, Asym = Asymmetric

Example 35-1 contains a code sample showing initialization and run time for the waveforms in Figure 35-21.

Figure 35-21. Up, Single Edge Asymmetric Waveform, With Independent Modulation on EPWMxA and EPWMxB—Active High



A PWM period = (TBPRD + 1) × T_{TBCLK}

B Duty modulation for EPWMxA is set by CMPA, and is active high (that is, high time duty proportional to CMPA).

C Duty modulation for EPWMxB is set by CMPB and is active high (that is, high time duty proportional to CMPB).

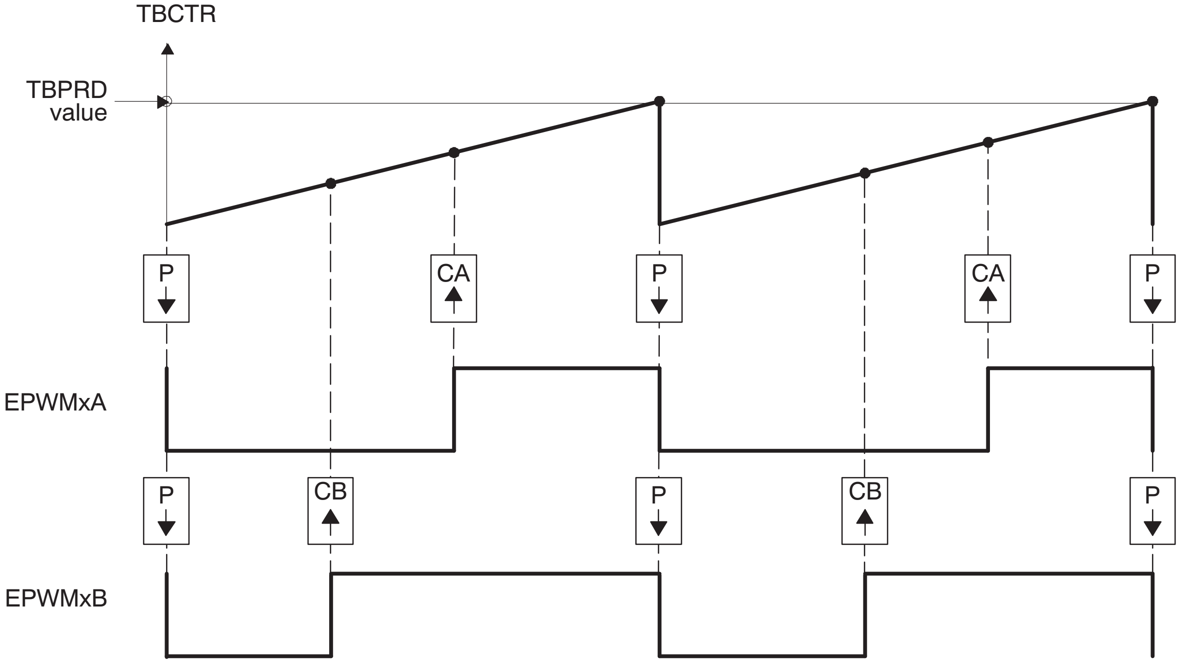
D The "Do Nothing" actions (X) are shown for completeness, but will not be shown on subsequent diagrams.

E Actions at zero and period, although appearing to occur concurrently, are actually separated by one TBCLK period. TBCTR wraps from period to 0000.

Example 35-1. Code Sample for Figure 35-21

```
// Initialization Time
// = = = = = = = = = = = = = = = = = = = = = = = =
EPwm1Regs.TBPRD = 600;                     // Period = 601 TBCLK counts
EPwm1Regs.CMPA.half.CMPA = 350;            // Compare A = 350 TBCLK counts
EPwm1Regs.CMPB = 200;                      // Compare B = 200 TBCLK counts
EPwm1Regs.TBPHS = 0;                       // Set Phase register to zero
EPwm1Regs.TBCTR = 0;                       // clear TB counter
EPwm1Regs.TBCTL.bit.CTRMODE = TB_COUNT_UP;
EPwm1Regs.TBCTL.bit.PHSEN = TB_DISABLE;    // Phase loading disabled
EPwm1Regs.TBCTL.bit.PRDLD = TB_SHADOW;
EPwm1Regs.TBCTL.bit.SYNCOSEL = TB_SYNC_DISABLE;
EPwm1Regs.TBCTL.bit.HSPCLKDIV = TB_DIV1;   // TBCLK = SYSCLK
EPwm1Regs.TBCTL.bit.CLKDIV = TB_DIV1;
EPwm1Regs.CMPCTL.bit.SHDWAMODE = CC_SHADOW;
EPwm1Regs.CMPCTL.bit.SHDWBMODE = CC_SHADOW;
EPwm1Regs.CMPCTL.bit.LOADAMODE = CC_CTR_ZERO; // load on CTR = Zero
EPwm1Regs.CMPCTL.bit.LOADBMODE = CC_CTR_ZERO; // load on CTR = Zero
EPwm1Regs.AQCTLA.bit.ZRO = AQ_SET;
EPwm1Regs.AQCTLA.bit.CAU = AQ_CLEAR;
EPwm1Regs.AQCTLB.bit.ZRO = AQ_SET;
EPwm1Regs.AQCTLB.bit.CBU = AQ_CLEAR;
//
// Run Time
// = = = = = = = = = = = = = = = = = = = = = = = =
EPwm1Regs.CMPA.half.CMPA = Duty1A;         // adjust duty for output EPWM1A
EPwm1Regs.CMPB = Duty1B;                   // adjust duty for output EPWM1B
```

Figure 35-22. Up, Single Edge Asymmetric Waveform With Independent Modulation on EPWMxA and EPWMxB—Active Low



A PWM period = (TBPRD + 1) × T_{TBCLK}

B Duty modulation for EPWMxA is set by CMPA, and is active low (that is, the low time duty is proportional to CMPA).

C Duty modulation for EPWMxB is set by CMPB and is active low (that is, the low time duty is proportional to CMPB).

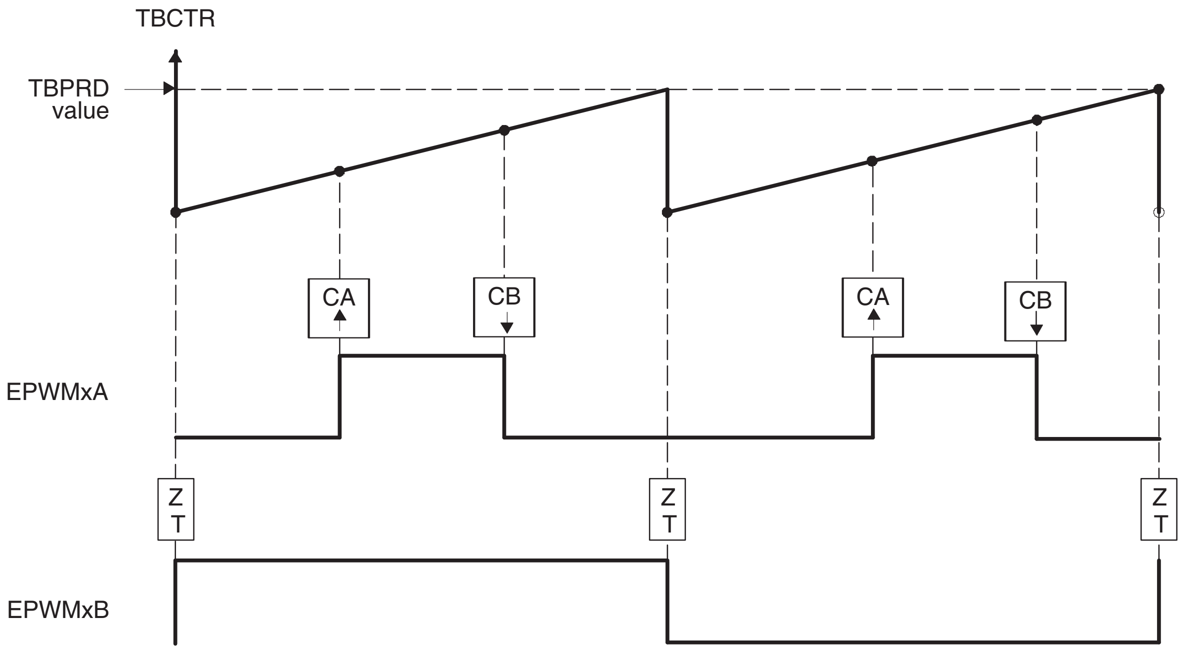
D Actions at zero and period, although appearing to occur concurrently, are actually separated by one TBCLK period. TBCTR wraps from period to 0000.

Example 35-2 contains a code sample showing initialization and run time for the waveforms in Figure 35-22.

Example 35-2. Code Sample for Figure 35-22

```
// Initialization Time
// = = = = = = = = = = = = = = = = = = = = = = = = = =
EPwm1Regs.TBPRD = 600;                         // Period = 601 TBCLK counts
EPwm1Regs.CMPA.half.CMPA = 350;                // Compare A = 350 TBCLK counts
EPwm1Regs.CMPB = 200;                          // Compare B = 200 TBCLK counts
EPwm1Regs.TBPHS = 0;                           // Set Phase register to zero
EPwm1Regs.TBCTR = 0;                           // clear TB counter
EPwm1Regs.TBCTL.bit.CTRMODE = TB_COUNT_UP;
EPwm1Regs.TBCTL.bit.PHSEN = TB_DISABLE;        // Phase loading disabled
EPwm1Regs.TBCTL.bit.PRDLD = TB_SHADOW;
EPwm1Regs.TBCTL.bit.SYNCOSEL = TB_SYNC_DISABLE;
EPwm1Regs.TBCTL.bit.HSPCLKDIV = TB_DIV1;       // TBCLK = VCLK3
EPwm1Regs.TBCTL.bit.CLKDIV = TB_DIV1;
EPwm1Regs.CMPCTL.bit.SHDWAMODE = CC_SHADOW;
EPwm1Regs.CMPCTL.bit.SHDWBMODE = CC_SHADOW;
EPwm1Regs.CMPCTL.bit.LOADAMODE = CC_CTR_ZERO; // load on TBCTR = Zero
EPwm1Regs.CMPCTL.bit.LOADBMODE = CC_CTR_ZERO; // load on TBCTR = Zero
EPwm1Regs.AQCTLA.bit.PRD = AQ_CLEAR;
EPwm1Regs.AQCTLA.bit.CAU = AQ_SET;
EPwm1Regs.AQCTLB.bit.PRD = AQ_CLEAR;
EPwm1Regs.AQCTLB.bit.CBU = AQ_SET;
//
// Run Time
// = = = = = = = = = = = = = = = = = = = = = = = = = =
EPwm1Regs.CMPA.half.CMPA = Duty1A;             // adjust duty for output EPWM1A
EPwm1Regs.CMPB = Duty1B;                       // adjust duty for output EPWM1B
```

Figure 35-23. Up-Count, Pulse Placement Asymmetric Waveform With Independent Modulation on EPWMxA



A PWM frequency = 1/((TBPRD + 1) × T_{TBCLK})

B Pulse can be placed anywhere within the PWM cycle (0000 - TBPRD)

C High time duty proportional to (CMPB - CMPA)

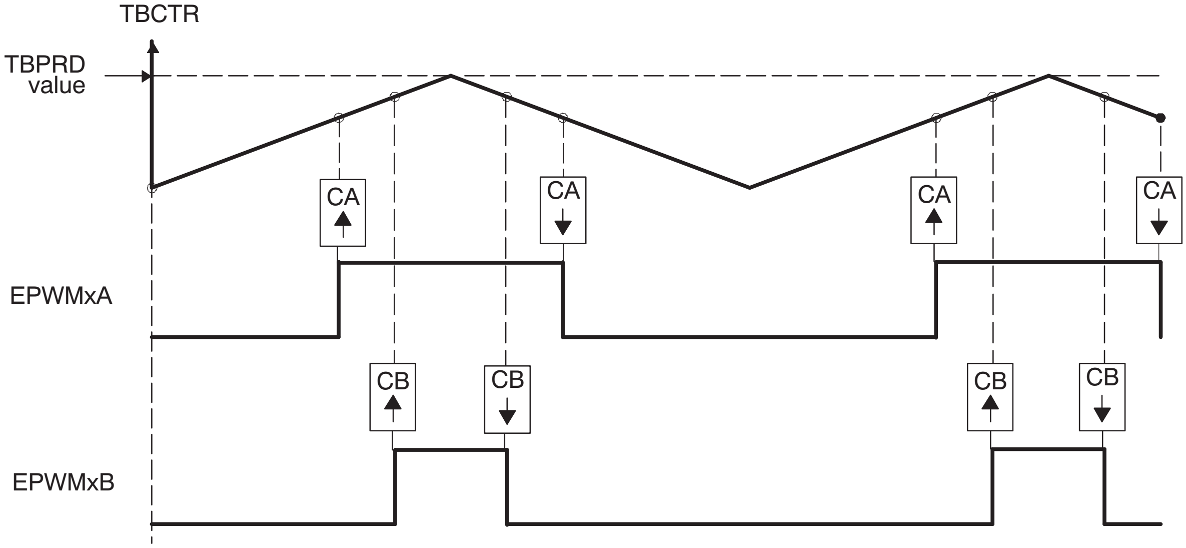
D EPWMxB can be used to generate a 50% duty square wave with frequency = 1/2 × ((TBPRD + 1) × TBCLK)

Example 35-3 contains a code sample showing initialization and run time for the waveforms Figure 35-23.

Example 35-3. Code Sample for Figure 35-23

```
// Initialization Time
// = = = = = = = = = = = = = = = = = = = = = = = = =
EPwm1Regs.TBPRD = 600;                        // Period = 601 TBCLK counts
EPwm1Regs.CMPA.half.CMPA = 200;               // Compare A = 200 TBCLK counts
EPwm1Regs.CMPB = 400;                         // Compare B = 400 TBCLK counts
EPwm1Regs.TBPHS = 0;                          // Set Phase register to zero
EPwm1Regs.TBCTR = 0;                          // clear TB counter
EPwm1Regs.TBCTL.bit.CTRMODE = TB_COUNT_UP;
EPwm1Regs.TBCTL.bit.PHSEN = TB_DISABLE;       // Phase loading disabled
EPwm1Regs.TBCTL.bit.PRDLD = TB_SHADOW;
EPwm1Regs.TBCTL.bit.SYNCOSEL = TB_SYNC_DISABLE;
EPwm1Regs.TBCTL.bit.HSPCLKDIV = TB_DIV1;      // TBCLK = VCLK3
EPwm1Regs.TBCTL.bit.CLKDIV = TB_DIV1;
EPwm1Regs.CMPCTL.bit.SHDWAMODE = CC_SHADOW;
EPwm1Regs.CMPCTL.bit.SHDWBMODE = CC_SHADOW;
EPwm1Regs.CMPCTL.bit.LOADAMODE = CC_CTR_ZERO; // load on TBCTR = Zero
EPwm1Regs.CMPCTL.bit.LOADBMODE = CC_CTR_ZERO; // load on TBCTR = Zero
EPwm1Regs.AQCTLA.bit.CAU = AQ_SET;
EPwm1Regs.AQCTLA.bit.CBU = AQ_CLEAR;
EPwm1Regs.AQCTLB.bit.ZRO = AQ_TOGGLE;
//
// Run Time
// = = = = = = = = = = = = = = = = = = = = = = = = =
EPwm1Regs.CMPA.half.CMPA = EdgePosA;         // adjust duty for output EPWM1A only
EPwm1Regs.CMPB = EdgePosB;
```

Figure 35-24. Up-Down-Count, Dual Edge Symmetric Waveform, With Independent Modulation on EPWMxA and EPWMxB — Active Low



A PWM period = 2 x TBPRD × T_{TBCLK}

B Duty modulation for EPWMxA is set by CMPA, and is active low (that is, the low time duty is proportional to CMPA).

C Duty modulation for EPWMxB is set by CMPB and is active low (that is, the low time duty is proportional to CMPB).

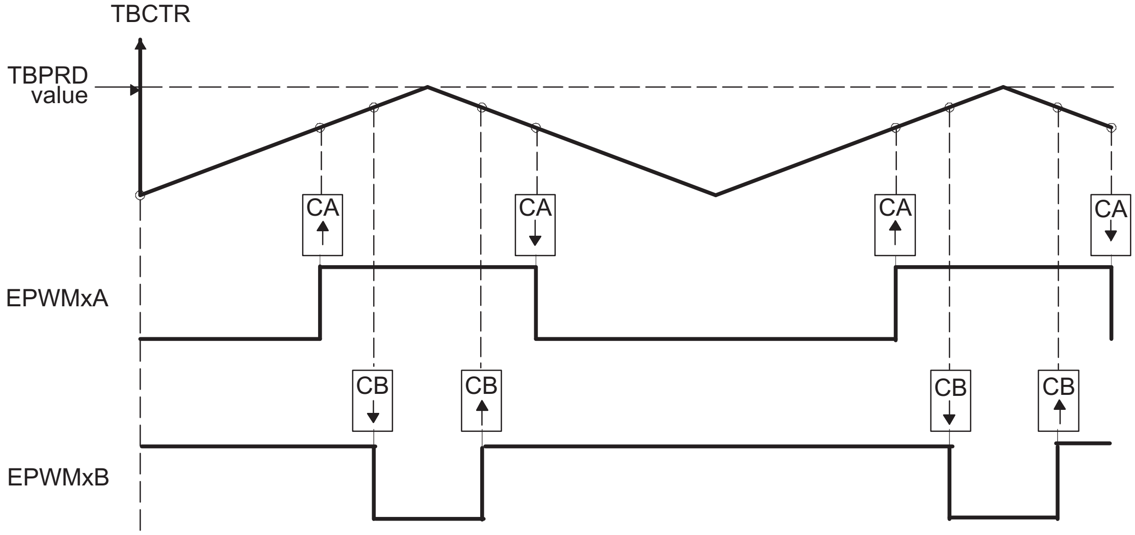
D Outputs EPWMxA and EPWMxB can drive independent power switches

Example 35-4 contains a code sample showing initialization and run time for the waveforms in Figure 35-24.

Example 35-4. Code Sample for Figure 35-24

```
// Initialization Time
// = = = = = = = = = = = = = = = = = = = = = = = = = =
EPwm1Regs.TBPRD = 600;                          // Period = 2´600 TBCLK counts
EPwm1Regs.CMPA.half.CMPA = 400;                 // Compare A = 400 TBCLK counts
EPwm1Regs.CMPB = 500;                           // Compare B = 500 TBCLK counts
EPwm1Regs.TBPHS = 0;                            // Set Phase register to zero
EPwm1Regs.TBCTR = 0;                            // clear TB counter
EPwm1Regs.TBCTL.bit.CTRMODE = TB_COUNT_UPDOWN; // Symmetric
xEPwm1Regs.TBCTL.bit.PHSEN = TB_DISABLE;       // Phase loading disabled
xEPwm1Regs.TBCTL.bit.PRDLD = TB_SHADOW;
EPwm1Regs.TBCTL.bit.SYNCOSEL = TB_SYNC_DISABLE;
EPwm1Regs.TBCTL.bit.HSPCLKDIV = TB_DIV1;       // TBCLK = VCLK3
EPwm1Regs.TBCTL.bit.CLKDIV = TB_DIV1;
EPwm1Regs.CMPCTL.bit.SHDWAMODE = CC_SHADOW;
EPwm1Regs.CMPCTL.bit.SHDWBMODE = CC_SHADOW;
EPwm1Regs.CMPCTL.bit.LOADAMODE = CC_CTR_ZERO;  // load on CTR = Zero
EPwm1Regs.CMPCTL.bit.LOADBMODE = CC_CTR_ZERO;  // load on CTR = Zero
EPwm1Regs.AQCTLA.bit.CAU = AQ_SET;
EPwm1Regs.AQCTLA.bit.CAD = AQ_CLEAR;
EPwm1Regs.AQCTLB.bit.CBU = AQ_SET;
EPwm1Regs.AQCTLB.bit.CBD = AQ_CLEAR;
//
// Run Time
// = = = = = = = = = = = = = = = = = = = = = = = = = =
EPwm1Regs.CMPA.half.CMPA = Duty1A;             // adjust duty for output EPWM1A
EPwm1Regs.CMPB = Duty1B;                       // adjust duty for output EPWM1B
```

Figure 35-25. Up-Down-Count, Dual Edge Symmetric Waveform, With Independent Modulation on EPWMxA and EPWMxB — Complementary



A PWM period = 2 × TBPRD × T_{TBCLK}

B Duty modulation for EPWMxA is set by CMPA, and is active low, i.e., low time duty proportional to CMPA

C Duty modulation for EPWMxB is set by CMPB and is active high, i.e., high time duty proportional to CMPB

D Outputs EPWMx can drive upper/lower (complementary) power switches

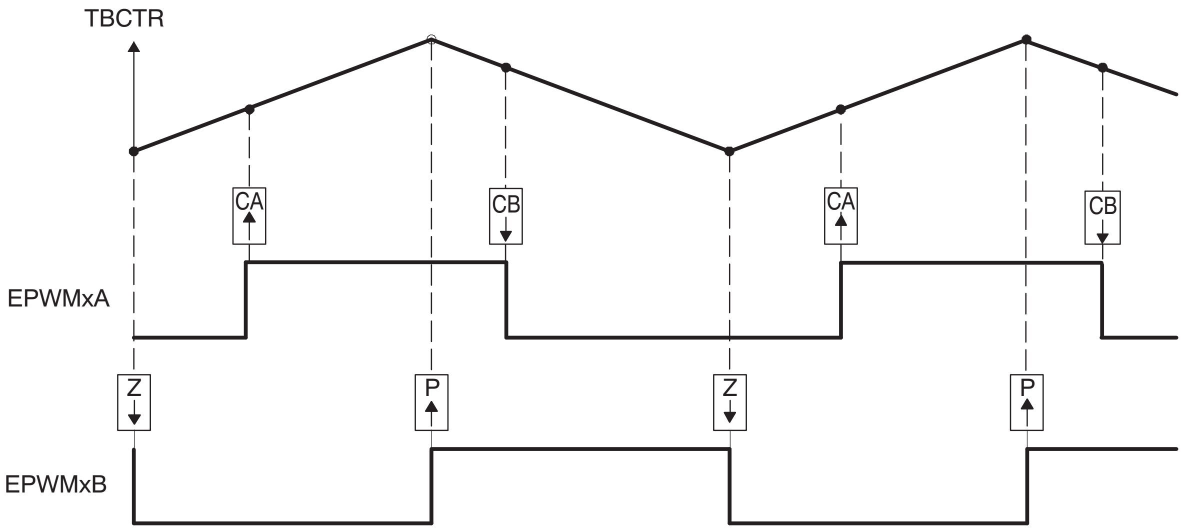
E Dead-band = CMPB - CMPA (fully programmable edge placement by software). Note the dead-band module is also available if the more classical edge delay method is required.

Example 35-5 contains a code sample showing initialization and run time for the waveforms in Figure 35-25.

Example 35-5. Code Sample for Figure 35-25

```
// Initialization Time
// = = = = = = = = = = = = = = = = = = = = = = = = =
EPwm1Regs.TBPRD = 600;                         // Period = 2´600 TBCLK counts
EPwm1Regs.CMPA.half.CMPA = 350;                // Compare A = 350 TBCLK counts
EPwm1Regs.CMPB = 400;                          // Compare B = 400 TBCLK counts
EPwm1Regs.TBPHS = 0;                           // Set Phase register to zero
EPwm1Regs.TBCTR = 0;                           // clear TB counter
EPwm1Regs.TBCTL.bit.CTRMODE = TB_COUNT_UPDOWN; // Symmetric
EPwm1Regs.TBCTL.bit.PHSEN = TB_DISABLE;       // Phase loading disabled
EPwm1Regs.TBCTL.bit.PRDLD = TB_SHADOW;
EPwm1Regs.TBCTL.bit.SYNCOSEL = TB_SYNC_DISABLE;
EPwm1Regs.TBCTL.bit.HSPCLKDIV = TB_DIV1;      // TBCLK = VCLK3
EPwm1Regs.TBCTL.bit.CLKDIV = TB_DIV1;
EPwm1Regs.CMPCTL.bit.SHDWAMODE = CC_SHADOW;
EPwm1Regs.CMPCTL.bit.SHDWBMODE = CC_SHADOW;
EPwm1Regs.CMPCTL.bit.LOADAMODE = CC_CTR_ZERO; // load on CTR = Zero
EPwm1Regs.CMPCTL.bit.LOADBMODE = CC_CTR_ZERO; // load on CTR = Zero
EPwm1Regs.AQCTLA.bit.CAU = AQ_SET;
EPwm1Regs.AQCTLA.bit.CAD = AQ_CLEAR;
EPwm1Regs.AQCTLB.bit.CBU = AQ_CLEAR;
EPwm1Regs.AQCTLB.bit.CBD = AQ_SET;
// Run Time
// = = = = = = = = = = = = = = = = = = = = = = = = =
EPwm1Regs.CMPA.half.CMPA = Duty1A;          // adjust duty for output EPWM1A
EPwm1Regs.CMPB = Duty1B;                    // adjust duty for output EPWM1B
```

Figure 35-26. Up-Down-Count, Dual Edge Asymmetric Waveform, With Independent Modulation on EPWMxA—Active Low



A PWM period = 2 × TBPRD × TBCLK

B Rising edge and falling edge can be asymmetrically positioned within a PWM cycle. This allows for pulse placement techniques.

C Duty modulation for EPWMxA is set by CMPA and CMPB.

D Low time duty for EPWMxA is proportional to (CMPA + CMPB).

E To change this example to active high, CMPA and CMPB actions need to be inverted (i.e., Set ! Clear and Clear Set).

F Duty modulation for EPWMxB is fixed at 50% (utilizes spare action resources for EPWMxB)

Example 35-6 contains a code sample showing initialization and run time for the waveforms in Figure 35-26.

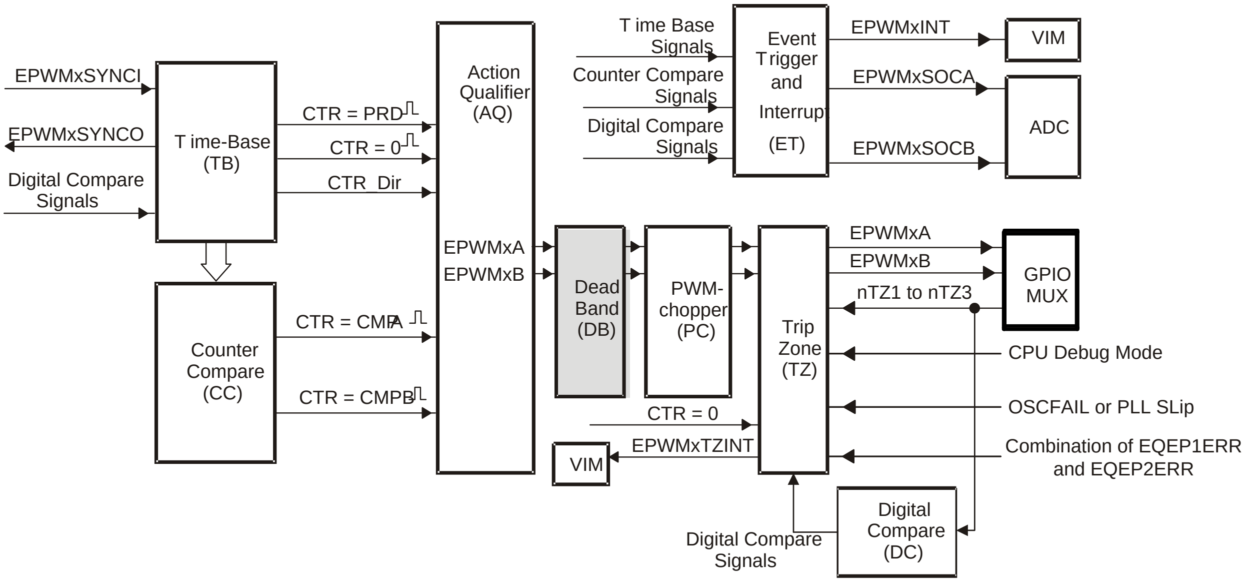
Example 35-6. Code Sample for Figure 35-26

```
// Initialization Time
// = = = = = = = = = = = = = = = = = = = = = = = = =
EPwm1Regs.TBPRD = 600;                          //  Period = 2 ´ 600 TBCLK counts
EPwm1Regs.CMPA.half.CMPA = 250;                 //  Compare A = 250 TBCLK counts
EPwm1Regs.CMPB = 450;                           //  Compare B = 450 TBCLK counts
EPwm1Regs.TBPHS = 0;                            //  Set Phase register to zero
EPwm1Regs.TBCTR = 0;                            //  clear TB counter
EPwm1Regs.TBCTL.bit.CTRMODE = TB_COUNT_UPDOWN; //  Symmetric
EPwm1Regs.TBCTL.bit.PHSEN = TB_DISABLE;         //  Phase loading disabled
EPwm1Regs.TBCTL.bit.PRDLD = TB_SHADOW;
EPwm1Regs.TBCTL.bit.SYNCOSEL = TB_SYNC_DISABLE;
EPwm1Regs.TBCTL.bit.HSPCLKDIV = TB_DIV1;        //  TBCLK = VCLK3
EPwm1Regs.TBCTL.bit.CLKDIV = TB_DIV1;
EPwm1Regs.CMPCTL.bit.SHDWAMODE = CC_SHADOW;
EPwm1Regs.CMPCTL.bit.SHDWBMODE = CC_SHADOW;
EPwm1Regs.CMPCTL.bit.LOADAMODE = CC_CTR_ZERO;  //  load on CTR = Zero
EPwm1Regs.CMPCTL.bit.LOADBMODE = CC_CTR_ZERO;  //  load on CTR = Zero
EPwm1Regs.AQCTLA.bit.CAU = AQ_SET;
EPwm1Regs.AQCTLA.bit.CBD = AQ_CLEAR;
EPwm1Regs.AQCTLB.bit.ZRO = AQ_CLEAR;
EPwm1Regs.AQCTLB.bit.PRD = AQ_SET;
// Run Time
// = = = = = = = = = = = = = = = = = = = = = = = = =
EPwm1Regs.CMPA.half.CMPA = EdgePosA;             // adjust duty for output EPWM1A only
EPwm1Regs.CMPB = EdgePosB;
```

35.2.5 Dead-Band Generator (DB) Submodule

Figure 35-27 illustrates the dead-band submodule within the ePWM module.

Figure 35-27. Dead_Band Submodule



35.2.5.1 Purpose of the Dead-Band Submodule

The "Action-qualifier (AQ) Module" section discussed how it is possible to generate the required dead-band by having full control over edge placement using both the CMPA and CMPB resources of the ePWM module. However, if the more classical edge delay-based dead-band with polarity control is required, then the dead-band submodule described here should be used.

The key functions of the dead-band module are:

- Generating appropriate signal pairs (EPWMxA and EPWMxB) with dead-band relationship from a single EPWMxA input
- Programming signal pairs for:
  - Active high (AH)
  - Active low (AL)
  - Active high complementary (AHC)
  - Active low complementary (ALC)
- Adding programmable delay to rising edges (RED)
- Adding programmable delay to falling edges (FED)
- Can be totally bypassed from the signal path (note dotted lines in diagram)

35.2.5.2 Controlling and Monitoring the Dead-Band Submodule

The dead-band submodule operation is controlled and monitored via the following registers:

Table 35-13. Dead-Band Generator Submodule Registers

Register Name	Address Offset	Shadowed	Description
DBCTL	1Ch	No	Dead-Band Control Register
DBFED	20h	No	Dead-Band Falling Edge Delay Count Register
DBRED	22h	No	Dead-Band Rising Edge Delay Count Register

35.2.5.3 Operational Highlights for the Dead-Band Submodule

The following sections provide the operational highlights.

The dead-band submodule has two groups of independent selection options as shown in Figure 35-28.

- **Input Source Selection:**

  The input signals to the dead-band module are the EPWMxA and EPWMxB output signals from the action-qualifier. In this section they will be referred to as EPWMxA In and EPWMxB In. Using the DBCTL[IN_MODE) control bits, the signal source for each delay, falling-edge or rising-edge, can be selected:

  – EPWMxA In is the source for both falling-edge and rising-edge delay. This is the default mode.

  – EPWMxA In is the source for falling-edge delay, EPWMxB In is the source for rising-edge delay.

  – EPWMxA In is the source for rising edge delay, EPWMxB In is the source for falling-edge delay.

  – EPWMxB In is the source for both falling-edge and rising-edge delay.

- **Half Cycle Clocking:**

  The dead-band submodule can be clocked using half cycle clocking to double the resolution (i.e. counter clocked at 2× TBCLK)

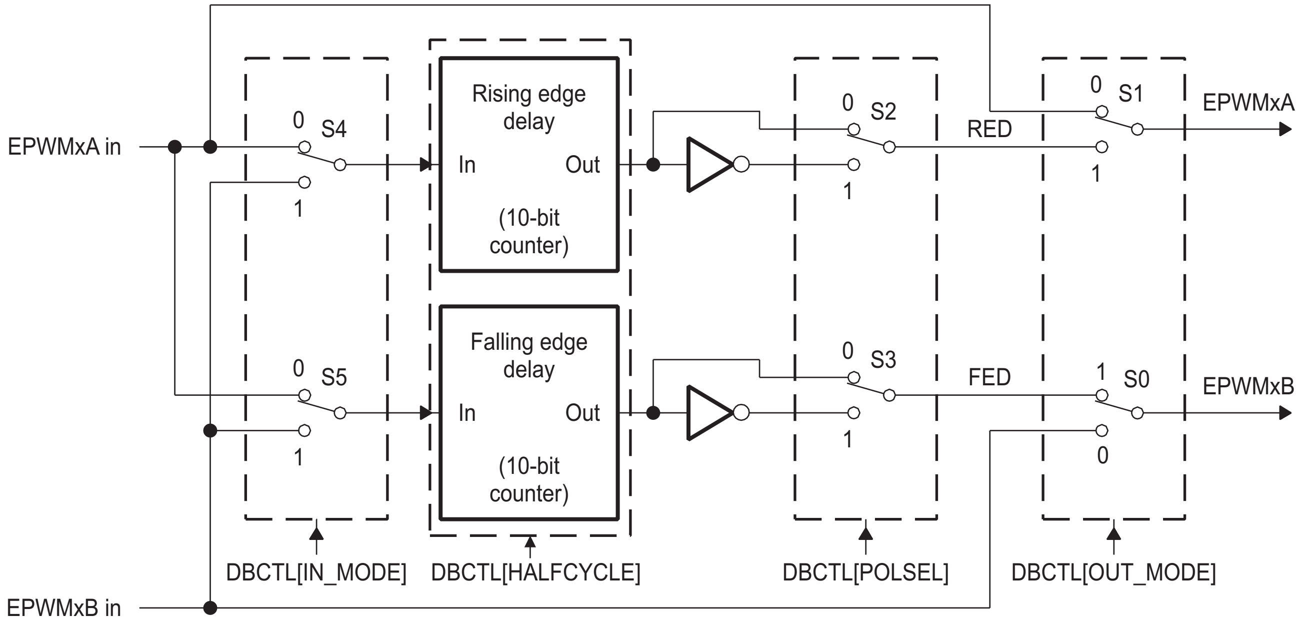
- **Output Mode Control:**

  The output mode is configured by way of the DBCTL[OUT_MODE] bits. These bits determine if the falling-edge delay, rising-edge delay, neither, or both are applied to the input signals.

- **Polarity Control:**

  The polarity control (DBCTL[POLSEL]) allows you to specify whether the rising-edge delayed signal and/or the falling-edge delayed signal is to be inverted before being sent out of the dead-band submodule.

Figure 35-28. Configuration Options for the Dead-Band Submodule



Although all combinations are supported, not all are typical usage modes. Table 35-14 documents some classical dead-band configurations. These modes assume that the DBCTL[IN_MODE] is configured such that EPWMxA In is the source for both falling-edge and rising-edge delay. Enhanced, or non-traditional modes can be achieved by changing the input signal source. The modes shown in Table 35-14 fall into the following categories:

- **Mode 1: Bypass both falling-edge delay (FED) and rising-edge delay (RED)**

  Allows you to fully disable the dead-band submodule from the PWM signal path.
- **Mode 2-5: Classical Dead-Band Polarity Settings:**

  These represent typical polarity configurations that should address all the active high/low modes required by available industry power switch gate drivers. The waveforms for these typical cases are shown in Figure 35-29. Note that to generate equivalent waveforms to Figure 35-29, configure the action-qualifier submodule to generate the signal as shown for EPWMxA.
- **Mode 6: Bypass rising-edge-delay and Mode 7: Bypass falling-edge-delay**

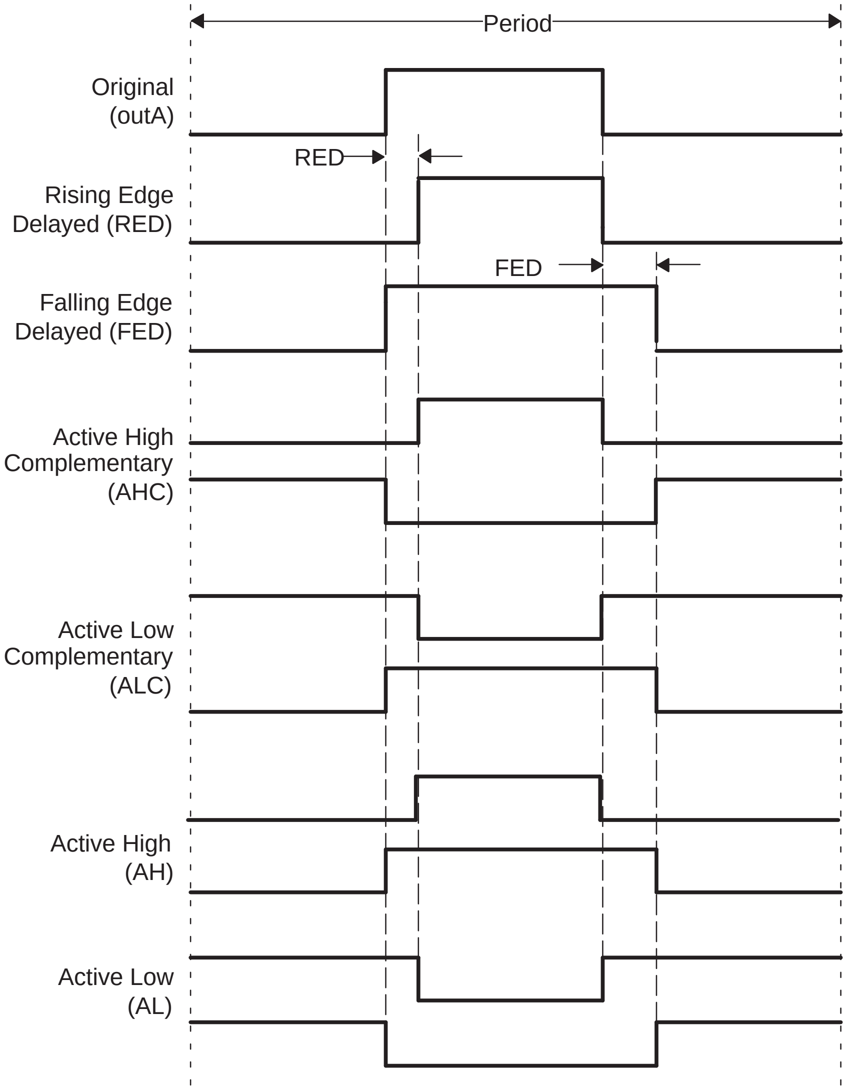
  Finally the last two entries in Table 35-14 show combinations where either the falling-edge-delay (FED) or rising-edge-delay (RED) blocks are bypassed.

Table 35-14. Classical Dead-Band Operating Modes

		DBCTL[POLSEL]		DBCTL[OUT_MODE]	
Mode	**Mode Description**	**S3**	**S2**	**S1**	**S0**
1	EPWMxA and EPWMxB Passed Through (No Delay)	X	X	0	0
2	Active High Complementary (AHC)	1	0	1	1
3	Active Low Complementary (ALC)	0	1	1	1
4	Active High (AH)	0	0	1	1
5	Active Low (AL)	1	1	1	1
6	EPWMxA Out = EPWMxA In (No Delay) EPWMxB Out = EPWMxA In with Falling Edge Delay	0 or 1	0 or 1	0	1
7	EPWMxA Out = EPWMxA In with Rising Edge Delay EPWMxB Out = EPWMxB In with No Delay	0 or 1	0 or 1	1	0

Figure 35-29 shows waveforms for typical cases where 0% < duty < 100%.

Figure 35-29. Dead-Band Waveforms for Typical Cases (0% < Duty < 100%)


Period
Original (outA)
RED
Rising Edge Delayed (RED)
FED
Falling Edge Delayed (FED)
Active High Complementary (AHC)
Active Low Complementary (ALC)
Active High (AH)
Active Low (AL)

The dead-band submodule supports independent values for rising-edge (RED) and falling-edge (FED) delays. The amount of delay is programmed using the DBRED and DBFED registers. These are 10-bit registers and their value represents the number of time-base clock, TBCLK, periods a signal edge is delayed by. For example, the formula to calculate falling-edge-delay and rising-edge-delay are:

FED = DBFED × T_{TBCLK}

RED = DBRED × T_{TBCLK}

Where T_{TBCLK} is the period of TBCLK, the prescaled version of VCLK3.

For convenience, delay values for various TBCLK options are shown in Table 35-15.

Table 35-15. Dead-Band Delay Values in μS as a Function of DBFED and DBRED

Dead-Band Value	Dead-Band Delay in μS		
DBFED, DBRED	TBCLK = VCLK3/1	TBCLK = VCLK3 /2	TBCLK = VCLK3/4
1	0.02 μS	0.03 μS	0.07 μS
5	0.08 μS	0.17 μS	0.33 μS
10	0.17 μS	0.33 μS	0.67 μS
100	1.67 μS	3.33 μS	6.67 μS
200	3.33 μS	6.67 μS	13.33 μS
400	6.67 μS	13.33 μS	26.67 μS
500	8.33 μS	16.67 μS	33.33 μS
600	10.00 μS	20.00 μS	40.00 μS
700	11.67 μS	23.33 μS	46.67 μS
800	13.33 μS	26.67 μS	53.33 μS
900	15.00 μS	30.00 μS	60.00 μS
1000	16.67 μS	33.33 μS	66.67 μS

When half-cycle clocking is enabled, the formula to calculate the falling-edge-delay and rising-edge-delay becomes:

FED = DBFED × $T_{TBCLK}/2$

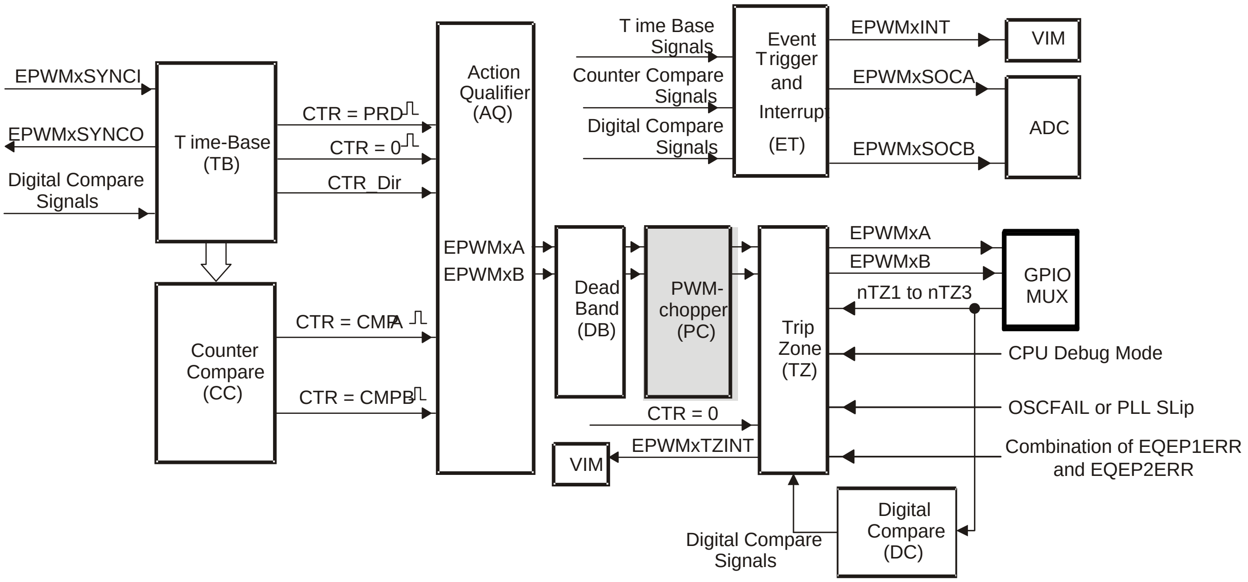
RED = DBRED × $T_{TBCLK}/2$

35.2.6 PWM-Chopper (PC) Submodule

Figure 35-30 illustrates the PWM-chopper (PC) submodule within the ePWM module.

The PWM-chopper submodule allows a high-frequency carrier signal to modulate the PWM waveform generated by the action-qualifier and dead-band submodules. This capability is important if you need pulse transformer-based gate drivers to control the power switching elements.

Figure 35-30. PWM-Chopper Submodule



35.2.6.1 Purpose of the PWM-Chopper Submodule

The key functions of the PWM-chopper submodule are:

- Programmable chopping (carrier) frequency
- Programmable pulse width of first pulse
- Programmable duty cycle of second and subsequent pulses
- Can be fully bypassed if not required

35.2.6.2 Controlling the PWM-Chopper Submodule

The PWM-chopper submodule operation is controlled via the registers in Table 35-16.

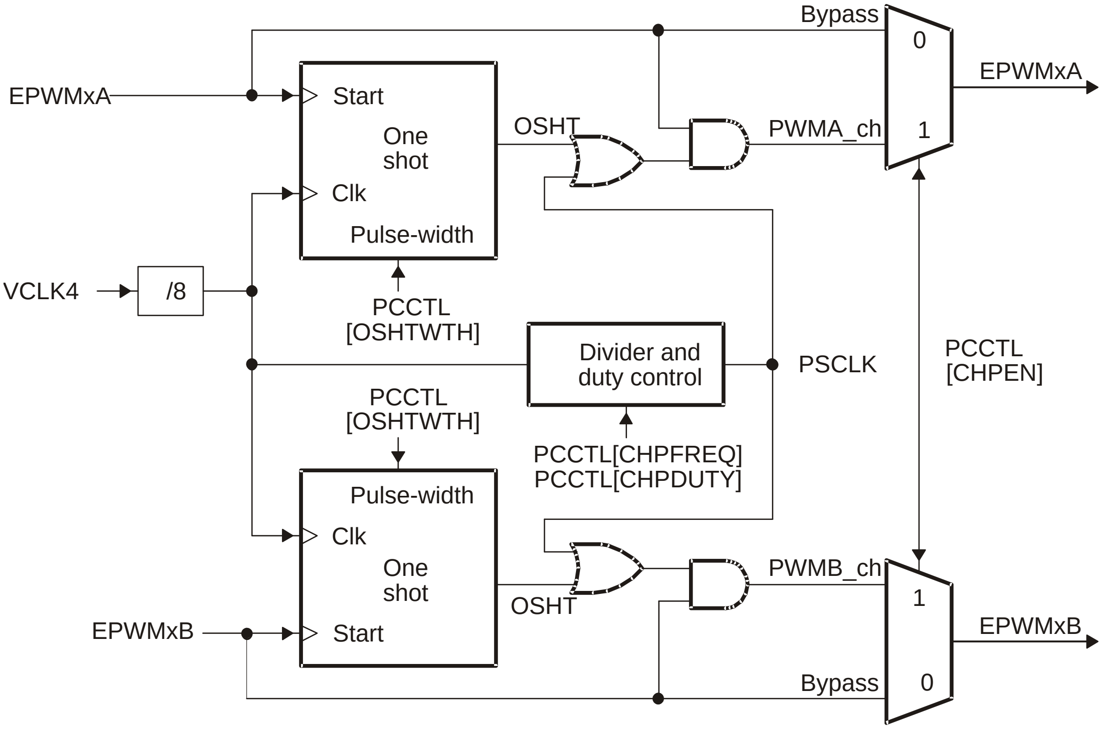
Table 35-16. PWM-Chopper Submodule Registers

Register Name	Address Offset	Shadowed	Description
PCCTL	3Eh	No	PWM-chopper Control Register

35.2.6.3 Operational Highlights for the PWM-Chopper Submodule

Figure 35-31 shows the operational details of the PWM-chopper submodule. The carrier clock is derived from VCLK3. Its frequency and duty cycle are controlled via the CHPFREQ and CHPDUTY bits in the PCCTL register. The one-shot block is a feature that provides a high energy first pulse to ensure hard and fast power switch turn on, while the subsequent pulses sustain pulses, ensuring the power switch remains on. The one-shot width is programmed via the OSHTWTH bits. The PWM-chopper submodule can be fully disabled (bypassed) via the CHPEN bit.

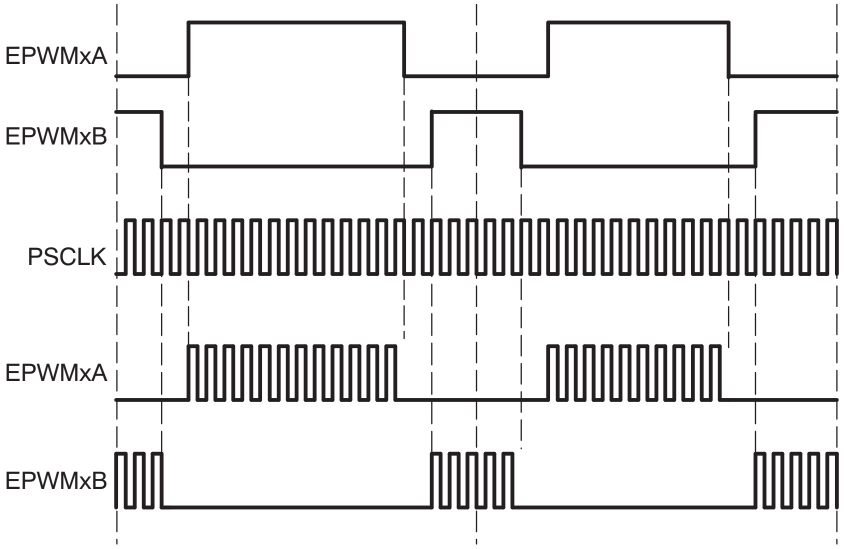
Figure 35-31. PWM-Chopper Submodule Operational Details



35.2.6.4 Waveforms

Figure 35-32 shows simplified waveforms of the chopping action only; one-shot and duty-cycle control are not shown. Details of the one-shot and duty-cycle control are discussed in the following sections.

Figure 35-32. Simple PWM-Chopper Submodule Waveforms Showing Chopping Action Only



35.2.6.4.1 One-Shot Pulse

The width of the first pulse can be programmed to any of 16 possible pulse width values. The width or period of the first pulse is given by:

$T_{1stpulse} = T_{VCLK3} \times 8 \times OSHTWTH$

Where T_{VCLK3} is the period of the system clock (VCLK3) and OSHTWTH is the four control bits (value from 1 to 16)

Figure 35-33 shows the first and subsequent sustaining pulses and Table 35-17 gives the possible pulse width values for a VCLK3 = 100 MHz.

Figure 35-33. PWM-Chopper Submodule Waveforms Showing the First Pulse and Subsequent Sustaining Pulses

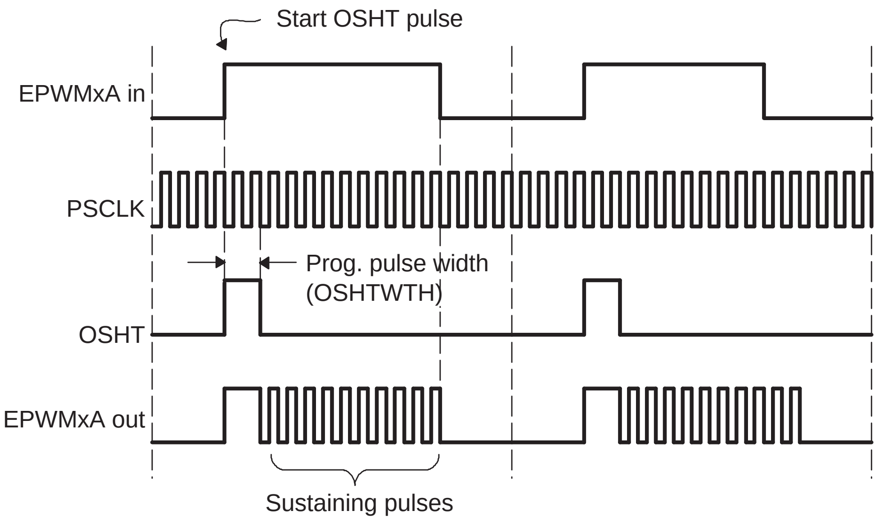


Table 35-17. Possible Pulse Width Values for VCLK3 = 100 MHz

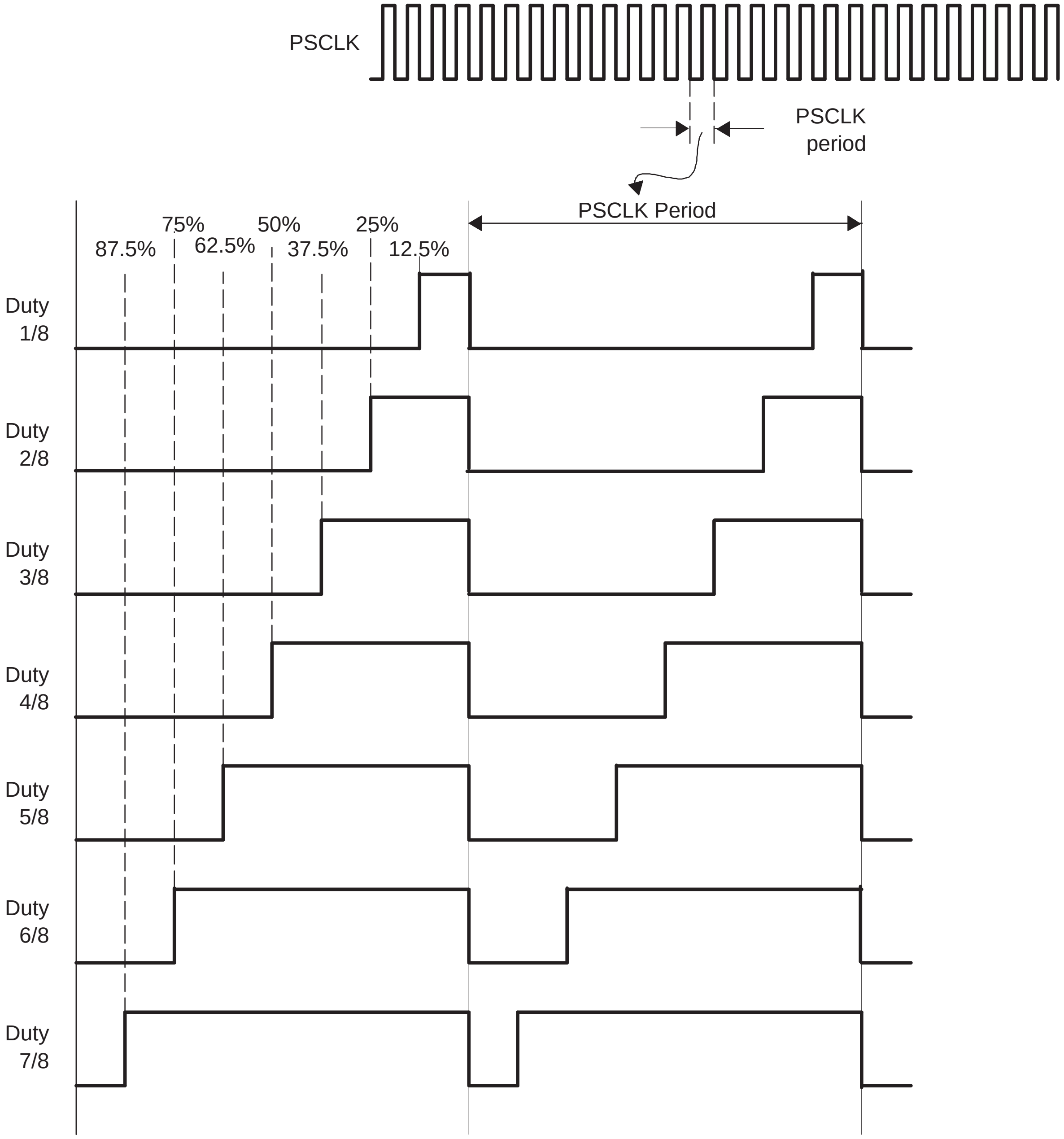
OSHTWTHz (hex)	Pulse Width (nS)
0	100
1	200
2	300
3	400
4	500
5	600
6	700
7	800
8	900
9	1000
A	1100
B	1200
C	1300
D	1400
E	1500
F	1600

35.2.6.4.2 Duty Cycle Control

Pulse transformer-based gate drive designs need to comprehend the magnetic properties or characteristics of the transformer and associated circuitry. Saturation is one such consideration. To assist the gate drive designer, the duty cycles of the second and subsequent pulses have been made programmable. These sustaining pulses ensure the correct drive strength and polarity is maintained on the power switch gate during the on period, and hence a programmable duty cycle allows a design to be tuned or optimized via software control.

Figure 35-34 shows the duty cycle control that is possible by programming the CHPDUTY bits. One of seven possible duty ratios can be selected ranging from 12.5% to 87.5%.

Figure 35-34. PWM-Chopper Submodule Waveforms Showing the Pulse Width (Duty Cycle) Control of Sustaining Pulses

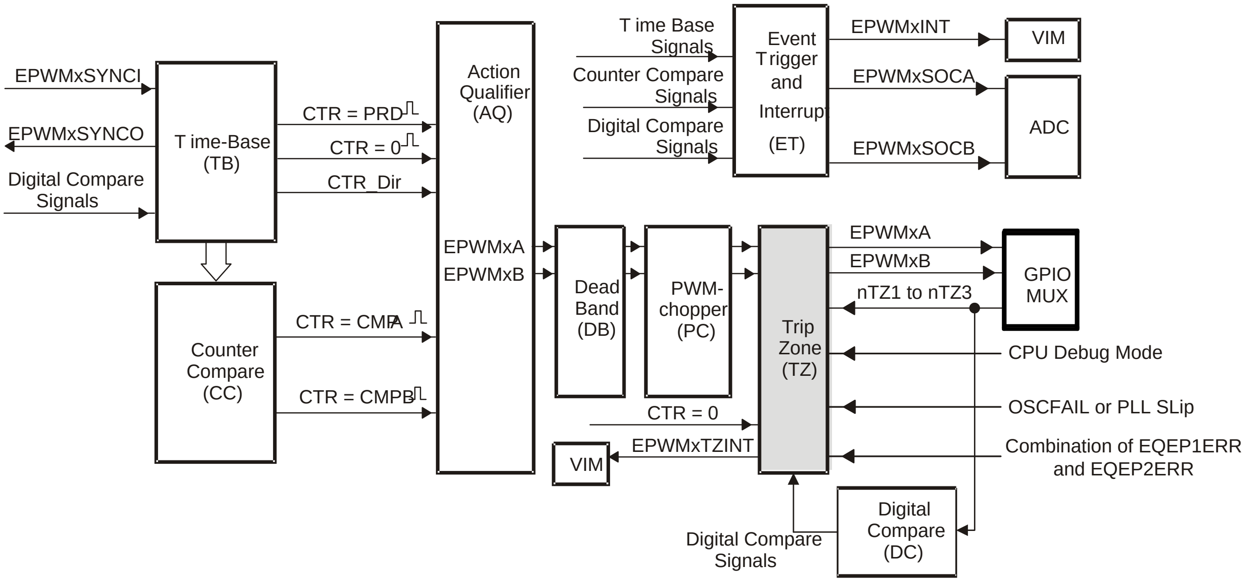


35.2.7 Trip-Zone (TZ) Submodule

Figure 35-35 shows how the trip-zone (TZ) submodule fits within the ePWM module.

Each ePWM module is connected to six $\overline{TZn}$ signals ($\overline{TZ1}$ to $\overline{TZ6}$). $\overline{TZ1}$ to $\overline{TZ3}$ are sourced from the GPIO mux. $\overline{TZ4}$ is sourced from a combination of EQEP1ERR and EQEP2ERR signals. $\overline{TZ5}$ is connected to the system oscillator or PLL clock fail logic, and $\overline{TZ6}$ is sourced from the debug mode halt indication output from the CPU. These signals indicate fault or trip conditions, and the ePWM outputs can be programmed to respond accordingly when faults occur.

Figure 35-35. Trip-Zone Submodule



35.2.7.1 Purpose of the Trip-Zone Submodule

The key functions of the Trip-Zone submodule are:

- Trip inputs $\overline{TZ1}$ to $\overline{TZ6}$ are mapped to all ePWM modules.
- Upon a fault indication, either no action is taken or the ePWM outputs EPWMxA and EPWMxB can be forced to one of the following:
  - High
  - Low
  - High-impedance
- Support for one-shot trip (OSHT) for major short circuits or over-current conditions.
- Support for cycle-by-cycle tripping (CBC) for current limiting operation.
- Support for digital compare tripping (DC) based on state of on-chip analog comparator module outputs and/or $\overline{TZ1}$ to $\overline{TZ3}$ signals.
- Each trip-zone input and digital compare (DC) submodule DCAEVT1/2 or DCBEVT1/2 force event can be allocated to either one-shot or cycle-by-cycle operation.
- Interrupt generation is possible on any trip-zone input.
- Software-forced tripping is also supported.
- The trip-zone submodule can be fully bypassed if it is not required.

35.2.7.2 Controlling and Monitoring the Trip-Zone Submodule

The trip-zone submodule operation is controlled and monitored through the following registers:

Table 35-18. Trip-Zone Submodule Registers

Register Name	Address Offset	Shadowed	Description [1]
TZDCSEL	24h	No	Trip-zone Digital Compare Select Register [2]
TZSEL	26h	No	Trip-Zone Select Register
TZEINT	28h	No	Trip-Zone Enable Interrupt Register
TZCTL	2Ah	No	Trip-Zone Control Register
TZCLR	2Ch	No	Trip-Zone Clear Register
TZFLG	2Eh	No	Trip-Zone Flag Register
TZFRC	32h	No	Trip-Zone Force Register

[1] All trip-zone registers are writable only in privileged mode.

[2] This register is discussed in more detail in Section 35.2.9.

35.2.7.3 Operational Highlights for the Trip-Zone Submodule

The following sections describe the operational highlights and configuration options for the trip-zone submodule.

The trip-zone signals $\overline{TZ1}$ to $\overline{TZ6}$ (also collectively referred to as $\overline{TZn}$) are active low input signals. When one of these signals goes low, or when a DCAEVT1/2 or DCBEVT1/2 force happens based on the TZDCSEL register event selection, it indicates that a trip event has occurred. Each ePWM module can be individually configured to ignore or use each of the trip-zone signals or DC events. Which trip-zone signals or DC events are used by a particular ePWM module is determined by the TZSEL register for that specific ePWM module. The trip-zone signals may or may not be synchronized to the system clock (VCLK3) and digitally filtered within the GPIO MUX block. A minimum of 3 × TBCLK low pulse width on $\overline{TZn}$ inputs is sufficient to trigger a fault condition on the ePWM module. If the pulse width is less than this, the trip condition may not be latched. The asynchronous trip makes sure that if clocks are missing for any reason, the outputs can still be tripped by a valid event present on $\overline{TZn}$ inputs. The GPIOs or peripherals must be appropriately configured. For more information, see the IOMM chapter of the device technical reference manual.

Each $\overline{TZn}$ input can be individually configured to provide either a cycle-by-cycle or one-shot trip event for an ePWM module. DCAEVT1 and DCBEVT1 events can be configured to directly trip an ePWM module or provide a one-shot trip event to the module. Likewise, DCAVET2 and DCBEVT2 events can also be configured to directly trip an ePWM module or provide a cycle-by-cycle trip event to the module. This configuration is determined by the TZSEL[DCAEVT1/2], TZSEL[DCBEVT1/2], TZSEL[CBCn], and TZSEL[OSHTn] control bits (where n corresponds to the trip input) respectively.

- **Cycle-by-Cycle (CBC):**

  When a cycle-by-cycle trip event occurs, the action specified in the TZCTL[TZA] and TZCTL[TZB] bits is carried out immediately on the EPWMxA and/or EPWMxB output. Table 35-19 lists the possible actions. In addition, the cycle-by-cycle trip event flag (TZFLG[CBC]) is set and a EPWMx_TZINT interrupt is generated if it is enabled in the TZEINT register and VIM peripheral.

  If the CBC interrupt is enabled via the TZEINT register, and DCAEVT2 or DCBEVT2 are selected as CBC trip sources via the TZSEL register, it is not necessary to also enable the DCAEVT2 or DCBEVT2 interrupts in the TZEINT register, as the DC events trigger interrupts through the CBC mechanism.

  The specified condition on the inputs is automatically cleared when the ePWM time-base counter reaches zero (TBCTR = 0x0000) if the trip event is no longer present. Therefore, in this mode, the trip event is cleared or reset every PWM cycle. The TZFLG[CBC] flag bit will remain set until it is manually cleared by writing to the TZCLR[CBC] bit. If the cycle-by-cycle trip event is still present when the TZFLG[CBC] bit is cleared, then it will again be immediately set.

- **One-Shot (OSHT):**

  When a one-shot trip event occurs, the action specified in the TZCTL[TZA] and TZCTL[TZB] bits is carried out immediately on the EPWMxA and/or EPWMxB output. Table 35-19 lists the possible actions. In addition, the one-shot trip event flag (TZFLG[OST]) is set and a EPWMx_TZINT interrupt is generated if it is enabled in the TZEINT register and VIM peripheral. The one-shot trip condition must be cleared manually by writing to the TZCLR[OST] bit.

  If the one-shot interrupt is enabled via the TZEINT register, and DCAEVT1 or DCBEVT1 are selected as OSHT trip sources via the TZSEL register, it is not necessary to also enable the DCAEVT1 or DCBEVT1 interrupts in the TZEINT register, as the DC events trigger interrupts through the OSHT mechanism.

- **Digital Compare Events (DCAEVT1/2 and DCBEVT1/2):**

  A digital compare DCAEVT1/2 or DCBEVT1/2 event is generated based on a combination of the DCAH/DCAL and DCBH/DCBL signals as selected by the TZDCSEL register. The signals which source the DCAH/DCAL and DCBH/DCBL signals are selected via the DCTRIPSEL register and can be either trip zone input pins. For more information on the digital compare submodule signals, see Section 35.2.9.

  When a digital compare event occurs, the action specified in the TZCTL[DCAEVT1/2] and TZCTL[DCBEVT1/2] bits is carried out immediately on the EPWMxA and/or EPWMxB output. Table 35-19 lists the possible actions. In addition, the relevant DC trip event flag (TZFLG[DCAEVT1/2] / TZFLG[DCBEVT1/2]) is set and a EPWMx_TZINT interrupt is generated if it is enabled in the TZEINT register and VIM peripheral.

  The specified condition on the pins is automatically cleared when the DC trip event is no longer present. The TZFLG[DCAEVT1/2] or TZFLG[DCBEVT1/2] flag bit will remain set until it is manually cleared by writing to the TZCLR[DCAEVT1/2] or TZCLR[DCBEVT1/2] bit. If the DC trip event is still present when the TZFLG[DCAEVT1/2] or TZFLG[DCBEVT1/2] flag is cleared, then it will again be immediately set.

The action taken when a trip event occurs can be configured individually for each of the ePWM output pins by way of the TZCTL register bit fields. One of four possible actions, shown in Table 35-19, can be taken on a trip event.

Table 35-19. Possible Actions On a Trip Event

TZCTL Register bit-field Settings	EPWMxA and/or EPWMxB	Comment
0,0	High-Impedance	Tripped
0,1	Force to High State	Tripped
1,0	Force to Low State	Tripped
1,1	No Change	Do Nothing. No change is made to the output.

Example 35-7. Trip-Zone Configurations

Scenario A:
A one-shot trip event on $\overline{\text{TZ1}}$ pulls both EPWM1A, EPWM1B low and also forces EPWM2A and EPWM2B high.

- Configure the ePWM1 registers as follows:
  - TZSEL[OSHT1] = 1: enables $\overline{\text{TZ1}}$ as a one-shot event source for ePWM1
  - TZCTL[TZA] = 2: EPWM1A will be forced low on a trip event.
  - TZCTL[TZB] = 2: EPWM1B will be forced low on a trip event.
- Configure the ePWM2 registers as follows:
  - TZSEL[OSHT1] = 1: enables $\overline{\text{TZ1}}$ as a one-shot event source for ePWM2
  - TZCTL[TZA] = 1: EPWM2A will be forced high on a trip event.
  - TZCTL[TZB] = 1: EPWM2B will be forced high on a trip event.

Scenario B:
A cycle-by-cycle event on $\overline{\text{TZ5}}$ pulls both EPWM1A, EPWM1B low.
A one-shot event on $\overline{\text{TZ1}}$ or $\overline{\text{TZ6}}$ puts EPWM2A into a high impedance state.

- Configure the ePWM1 registers as follows:
  - TZSEL[CBC5] = 1: enables $\overline{\text{TZ5}}$ as a one-shot event source for ePWM1
  - TZCTL[TZA] = 2: EPWM1A will be forced low on a trip event.
  - TZCTL[TZB] = 2: EPWM1B will be forced low on a trip event.
- Configure the ePWM2 registers as follows:
  - TZSEL[OSHT1] = 1: enables $\overline{\text{TZ1}}$ as a one-shot event source for ePWM2
  - TZSEL[OSHT6] = 1: enables $\overline{\text{TZ6}}$ as a one-shot event source for ePWM2
  - TZCTL[TZA] = 0: EPWM2A will be put into a high-impedance state on a trip event.
  - TZCTL[TZB] = 3: EPWM2B will ignore the trip event.

35.2.7.4 Generating Trip Event Interrupts

Figure 35-36 and Figure 35-37 illustrate the trip-zone submodule control and interrupt logic, respectively. DCAEVT1/2 and DCBEVT1/2 signals are described in further detail in Section 35.2.9.

Figure 35-36. Trip-Zone Submodule Mode Control Logic

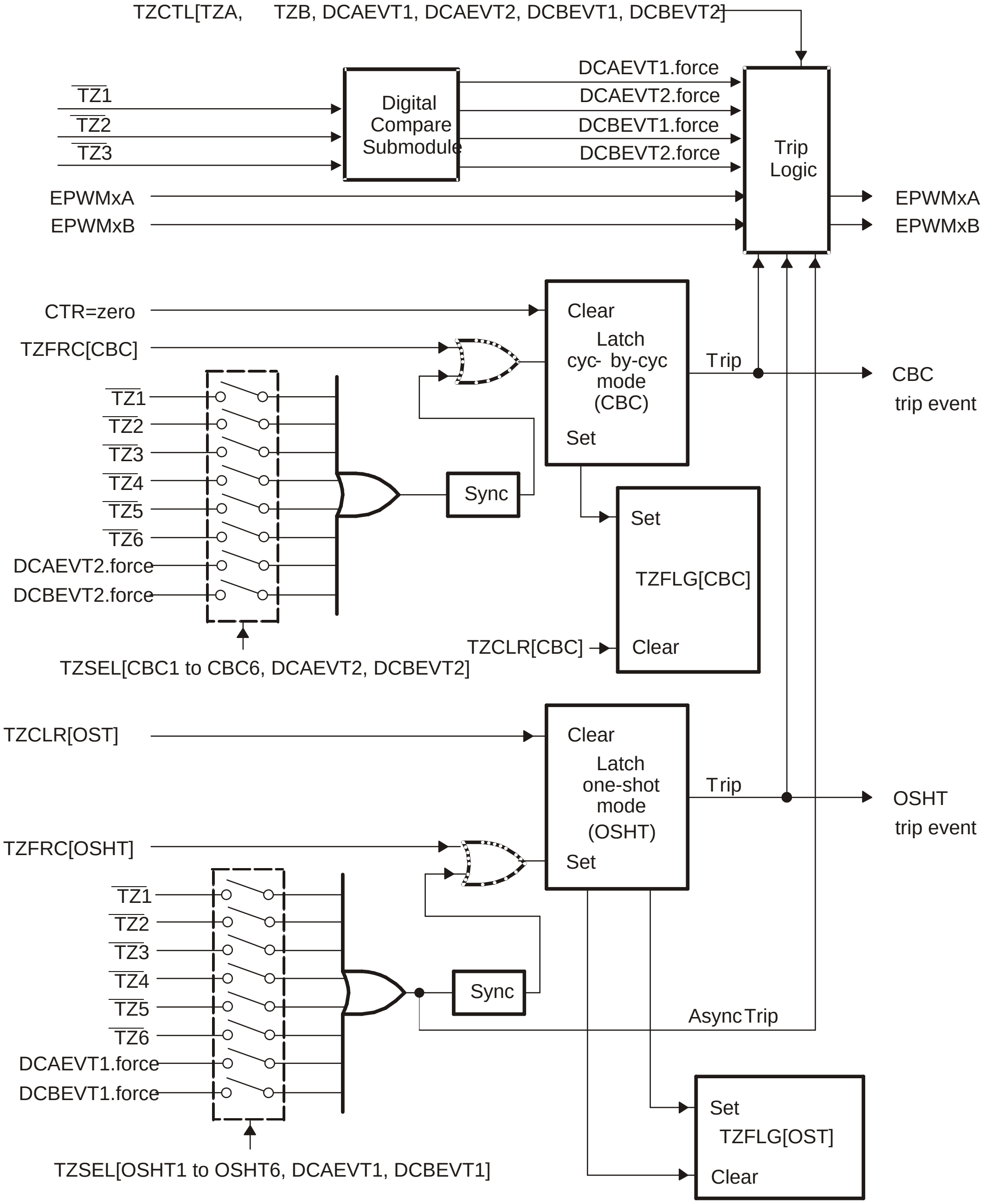
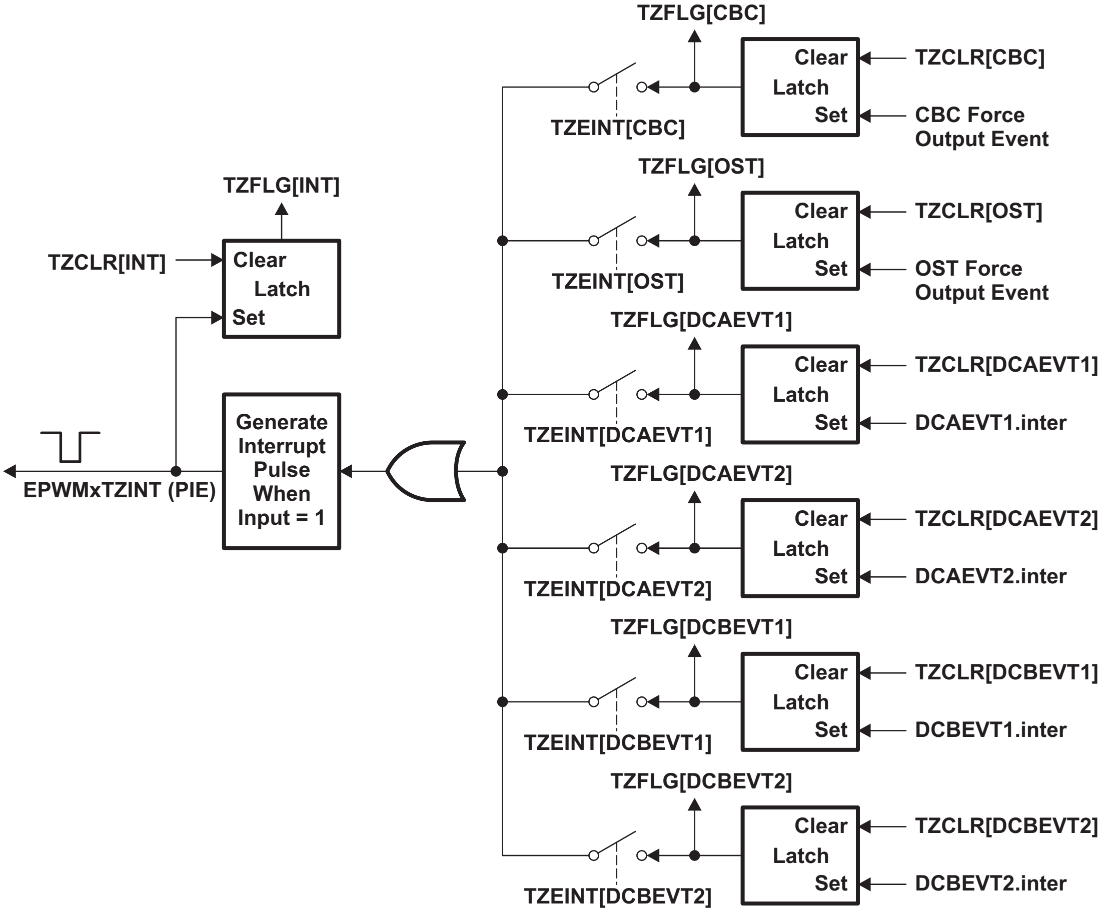


Figure 35-37. Trip-Zone Submodule Interrupt Logic



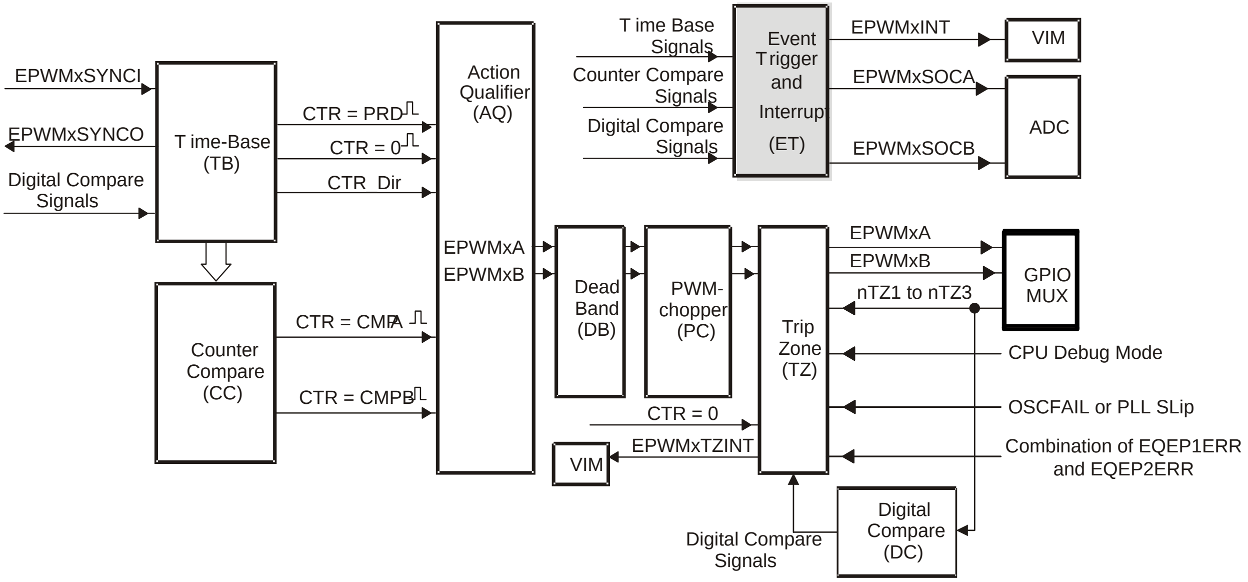
35.2.8 Event-Trigger (ET) Submodule

The key functions of the event-trigger submodule are:

- Receives event inputs generated by the time-base, counter-compare and digital-compare submodules
- Uses the time-base direction information for up/down event qualification
- Uses prescaling logic to issue interrupt requests and ADC start of conversion at:
  - Every event
  - Every second event
  - Every third event
- Provides full visibility of event generation via event counters and flags
- Allows software forcing of Interrupts and ADC start of conversion

The event-trigger submodule manages the events generated by the time-base submodule, the counter-compare submodule, and the digital-compare submodule to generate an interrupt to the CPU and/or a start of conversion pulse to the ADC when a selected event occurs. Figure 35-38 illustrates where the event-trigger submodule fits within the ePWM system.

Figure 35-38. Event-Trigger Submodule

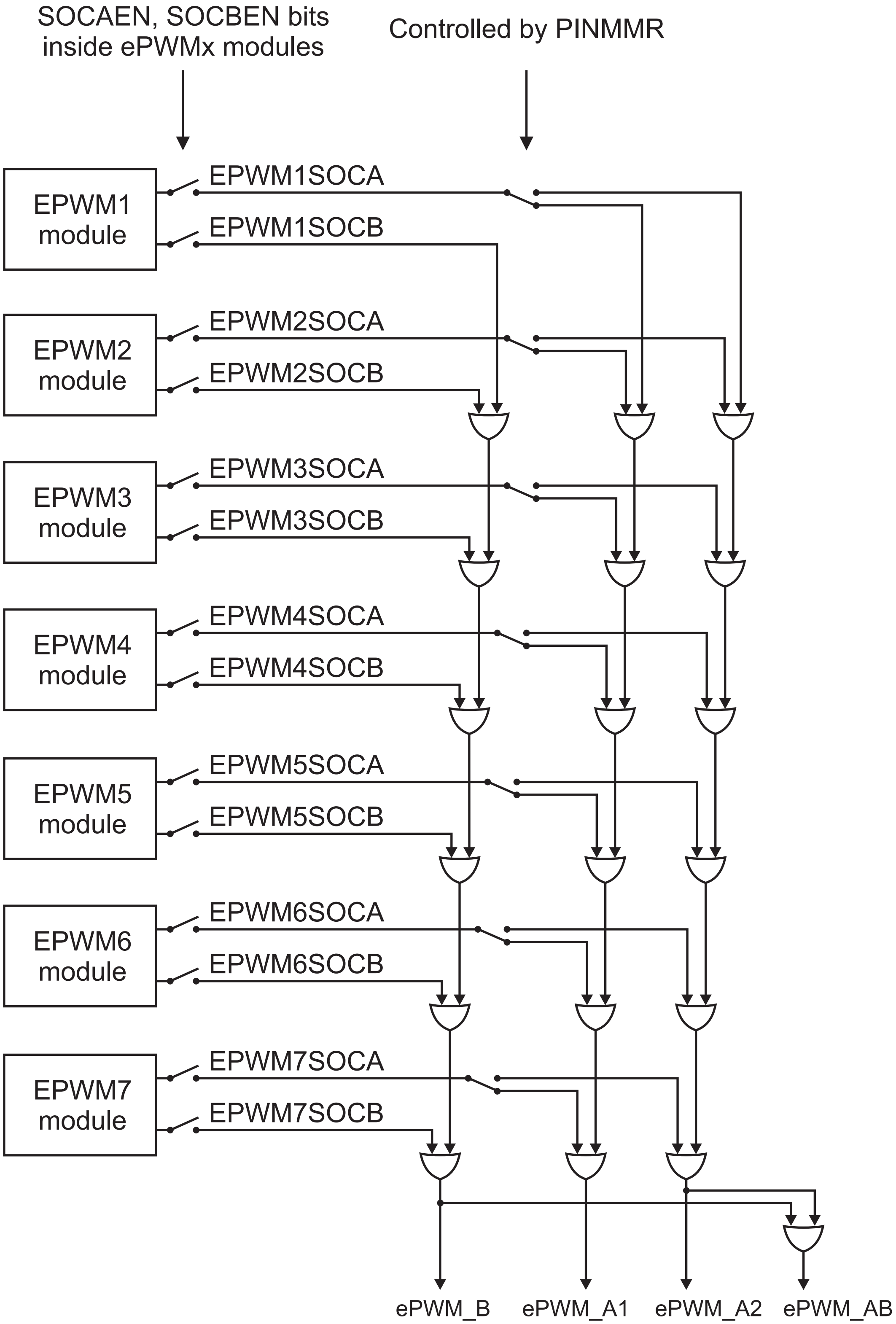


35.2.8.1 Operational Overview of the Event-Trigger Submodule

The following sections describe the event-trigger submodule's operational highlights.

Each ePWM module has one interrupt request line connected to the VIM and two start of conversion signals connected to the ADC module. As shown in Figure 35-39, the ePWMxSOCA and ePWMxSOCB signals are combined to generate four special signals that can be used to trigger an ADC start of conversion, and hence multiple modules can initiate an ADC start of conversion via the ADC trigger inputs.

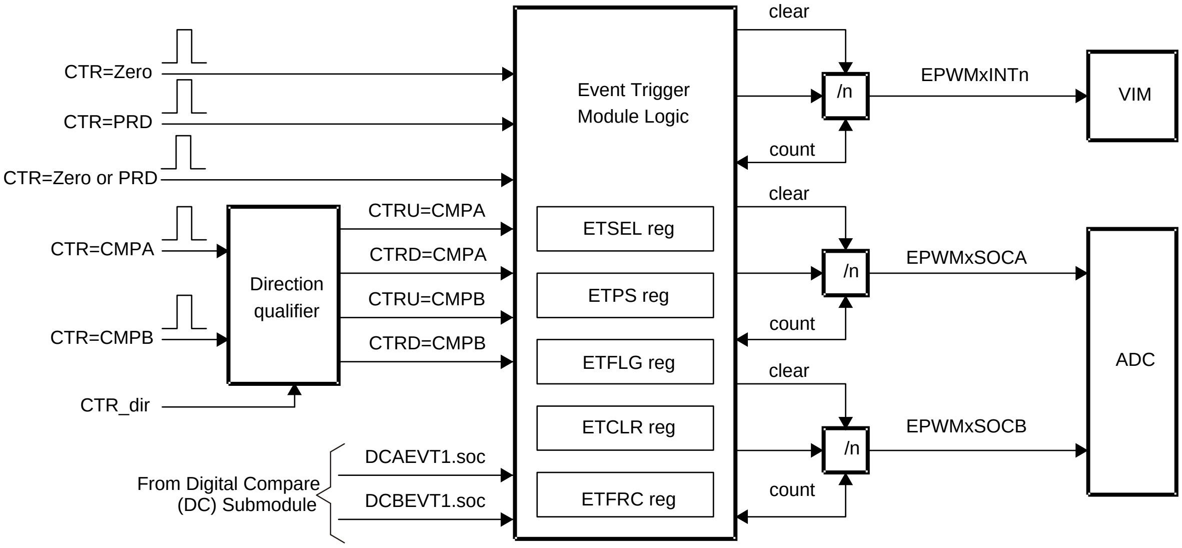
Figure 35-39. Event-Trigger Submodule Inter-Connectivity of ADC Start of Conversion



The event-trigger submodule monitors various event conditions (the left side inputs to event-trigger submodule shown in Figure 35-40) and can be configured to prescale these events before issuing an Interrupt request or an ADC start of conversion. The event-trigger prescaling logic can issue Interrupt requests and ADC start of conversion at:

- Every event
- Every second event
- Every third event

Figure 35-40. Event-Trigger Submodule Showing Event Inputs and Prescaled Outputs



The key registers used to configure the event-trigger submodule are listed in Table 35-20.

Table 35-20. Event-Trigger Submodule Registers

Register Name	Address Offset	Shadowed	Description
ETSEL	30h	No	Event-trigger Selection Register
ETFLG	34h	No	Event-trigger Flag Register
ETPS	36h	No	Event-trigger Prescale Register
ETFRC	38h	No	Event-trigger Force Register
ETCLR	3Ah	No	Event-trigger Clear Register

- ETSEL—This selects which of the possible events will trigger an interrupt or start an ADC conversion
- ETPS—This programs the event prescaling options mentioned above.
- ETFLG—These are flag bits indicating status of the selected and prescaled events.
- ETCLR—These bits allow you to clear the flag bits in the ETFLG register via software.
- ETFRC—These bits allow software forcing of an event. Useful for debugging or s/w intervention.

A more detailed look at how the various register bits interact with the Interrupt and ADC start of conversion logic are shown in Figure 35-41, Figure 35-42, and Figure 35-43.

Figure 35-41 shows the event-trigger's interrupt generation logic. The interrupt-period (ETPS[INTPRD]) bits specify the number of events required to cause an interrupt pulse to be generated. The choices available are:

- Do not generate an interrupt.
- Generate an interrupt on every event
- Generate an interrupt on every second event
- Generate an interrupt on every third event

Which event can cause an interrupt is configured by the interrupt selection (ETSEL[INTSEL]) bits. The event can be one of the following:

- Time-base counter equal to zero (TBCTR = 0x0000).
- Time-base counter equal to period (TBCTR = TBPRD).
- Time-base counter equal to zero or period (TBCTR = 0x0000 || TBCTR = TBPRD)
- Time-base counter equal to the compare A register (CMPA) when the timer is incrementing.
- Time-base counter equal to the compare A register (CMPA) when the timer is decrementing.
- Time-base counter equal to the compare B register (CMPB) when the timer is incrementing.
- Time-base counter equal to the compare B register (CMPB) when the timer is decrementing.

The number of events that have occurred can be read from the interrupt event counter (ETPS[INTCNT]) register bits. That is, when the specified event occurs the ETPS[INTCNT] bits are incremented until they reach the value specified by ETPS[INTPRD]. When ETPS[INTCNT] = ETPS[INTPRD] the counter stops counting and its output is set. The counter is only cleared when an interrupt is sent to the VIM.

When ETPS[INTCNT] reaches ETPS[INTPRD] the following behaviors will occur:

- If interrupts are enabled, ETSEL[INTEN] = 1 and the interrupt flag is clear, ETFLG[INT] = 0, then an interrupt pulse is generated and the interrupt flag is set, ETFLG[INT] = 1, and the event counter is cleared ETPS[INTCNT] = 0. The counter will begin counting events again.
- If interrupts are disabled, ETSEL[INTEN] = 0, or the interrupt flag is set, ETFLG[INT] = 1, the counter stops counting events when it reaches the period value ETPS[INTCNT] = ETPS[INTPRD].
- If interrupts are enabled, but the interrupt flag is already set, then the counter will hold its output high until the ENTFLG[INT] flag is cleared. This allows for one interrupt to be pending while one is serviced.

Writing to the INTPRD bits will automatically clear the counter INTCNT = 0 and the counter output will be reset (so no interrupts are generated). Writing a 1 to the ETFRC[INT] bit will increment the event counter INTCNT. The counter will behave as described above when INTCNT = INTPRD. When INTPRD = 0, the counter is disabled and hence no events will be detected and the ETFRC[INT] bit is also ignored.

The above definition means that you can generate an interrupt on every event, on every second event, or on every third event. An interrupt cannot be generated on every fourth or more events.

Figure 35-41. Event-Trigger Interrupt Generator

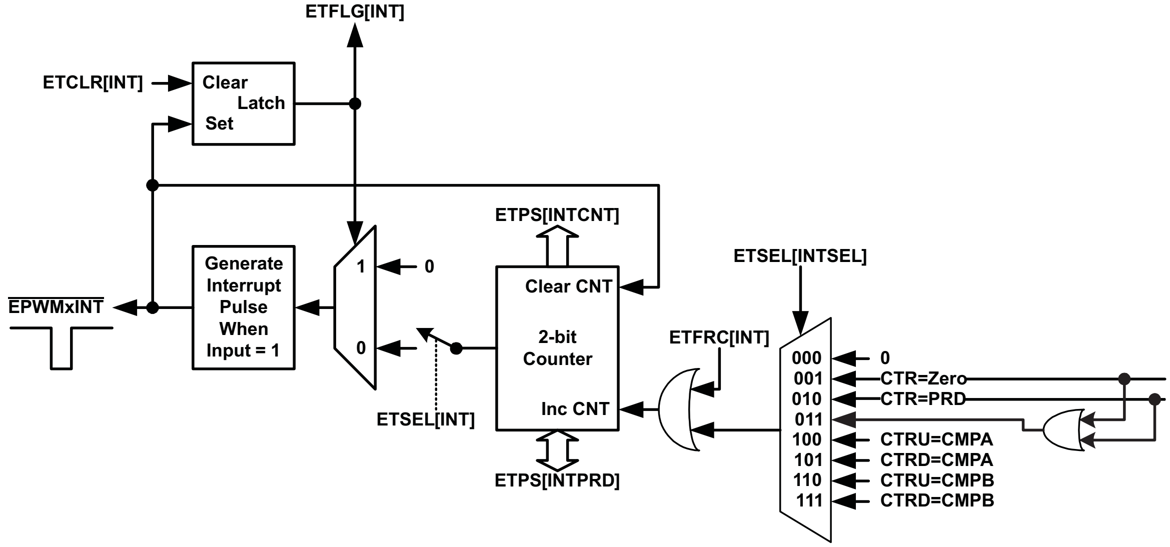
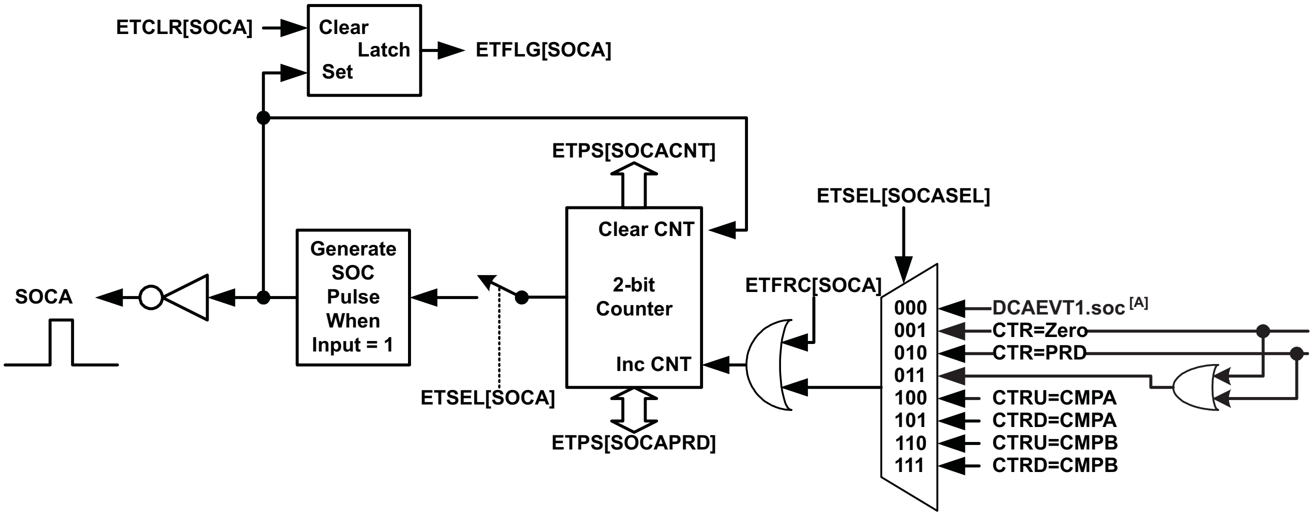


Figure 35-42 shows the operation of the event-trigger's start-of-conversion-A (SOCA) pulse generator. The ETPS[SOCACNT] counter and ETPS[SOCAPRD] period values behave similarly to the interrupt generator except that the pulses are continuously generated. That is, the pulse flag ETFLG[SOCA] is latched when a pulse is generated, but it does not stop further pulse generation. The enable/disable bit ETSEL[SOCAEN] stops pulse generation, but input events can still be counted until the period value is reached as with the interrupt generation logic. The event that will trigger an SOCA and SOCB pulse can be configured separately in the ETSEL[SOCASEL] and ETSEL[SOCBSEL] bits. The possible events are the same events that can be specified for the interrupt generation logic with the addition of the DCAEVT1.soc and DCBEVT1.soc event signals from the digital compare (DC) submodule.

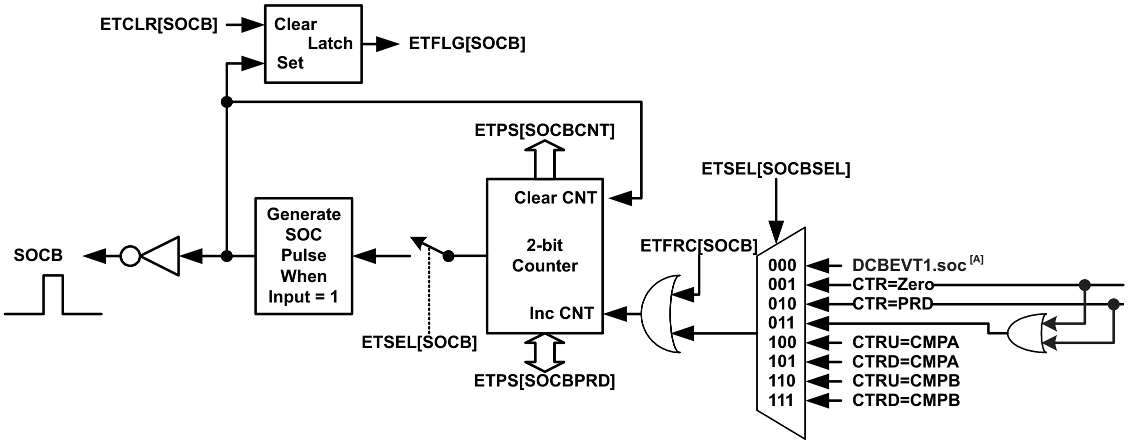
Figure 35-42. Event-Trigger SOCA Pulse Generator



A The DCAEVT1.soc signals are signals generated by the Digital compare (DC) submodule, described in Section 35.2.9

Figure 35-43 shows the operation of the event-trigger's start-of-conversion-B (SOCB) pulse generator. The event-trigger's SOCB pulse generator operates the same way as the SOCA.

Figure 35-43. Event-Trigger SOCB Pulse Generator



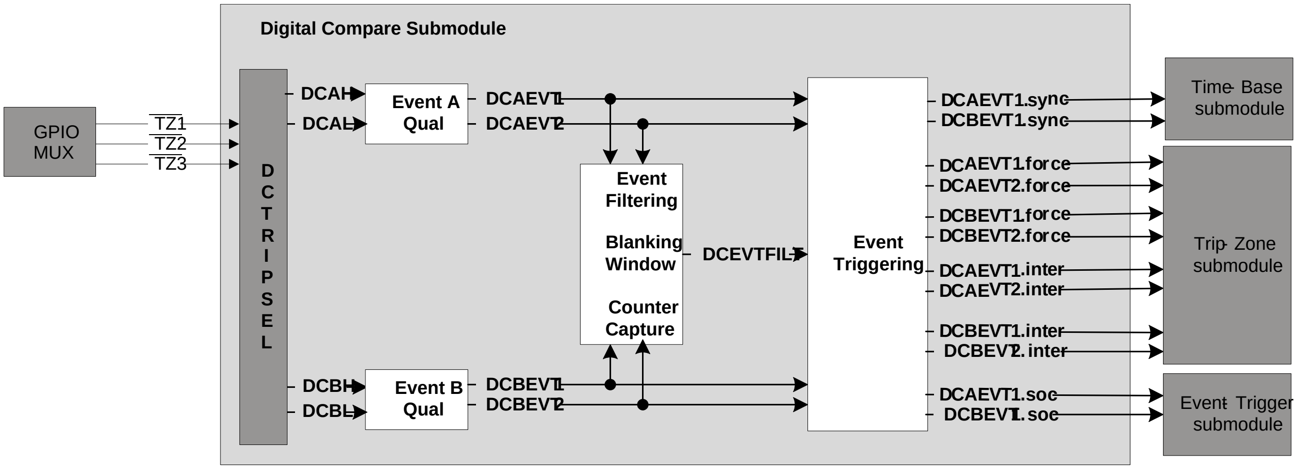
A The DCBEVT1.soc signals are signals generated by the Digital compare (DC) submodule, described in Section 35.2.9

35.2.9 Digital Compare (DC) Submodule

Figure 35-44 illustrates where the digital compare (DC) submodule signals interface to other submodules in the ePWM system.

The digital compare (DC) submodule compares signals external to the ePWM module to directly generate PWM events/actions which then feed to the event-trigger, trip-zone, and time-base submodules. Additionally, blanking window functionality is supported to filter noise or unwanted pulses from the DC event signals.

Figure 35-44. Digital-Compare Submodule High-Level Block Diagram



35.2.9.1 Purpose of the Digital Compare Submodule

The key functions of the digital compare submodule are:

- $\overline{TZ1}$, $\overline{TZ2}$, and $\overline{TZ3}$ inputs generate Digital Compare A High/Low (DCAH, DCAL) and Digital Compare B High/Low (DCBH, DCBL) signals.
- DCAH/L and DCBH/L signals trigger events which can then either be filtered or fed directly to the trip-zone, event-trigger, and time-base submodules to:
  – generate a trip zone interrupt
  – generate an ADC start of conversion
  – force an event
  – generate a synchronization event for synchronizing the ePWM module TBCTR.
- Event filtering (blanking window logic) can optionally blank the input signal to remove noise.

35.2.9.2 Controlling and Monitoring the Digital Compare Submodule

The digital compare submodule operation is controlled and monitored through the following registers:

Table 35-21. Digital Compare Submodule Registers

Register Name	Address Offset	Shadowed	Description
TZDCSEL[1] [2]	24h	No	Trip Zone Digital Compare Select Register
DCACTL[1]	60h	No	Digital Compare A Control Register
DCTRIPSEL[1]	62h	No	Digital Compare Trip Select Register
DCFCTL[1]	64h	No	Digital Compare Filter Control Register
DCBCTL[1]	66h	No	Digital Compare B Control Register
DCFOFFSET	68h	Writes	Digital Compare Filter Offset Register
DCCAPCTL[1]	6Ah	No	Digital Compare Capture Control Register
DCFWINDOW	6Ch	No	Digital Compare Filter Window Register
DCFOFFSETCNT	6Eh	No	Digital Compare Filter Offset Counter Register
DCCAP	70h	Yes	Digital Compare Counter Capture Register
DCFWINDOWCNT	72h	No	Digital Compare Filter Window Counter Register

[1] These registers are writable only in privileged mode.

[2] The TZDCSEL register is part of the trip-zone submodule but is mentioned again here because of its functional significance to the digital compare submodule.

35.2.9.3 Operation Highlights of the Digital Compare Submodule

The following sections describe the operational highlights and configuration options for the digital compare submodule.

35.2.9.3.1 Digital Compare Events

As illustrated in Figure 35-44 earlier in this section, trip zone inputs ($\overline{\text{TZ1}}$, $\overline{\text{TZ2}}$, and $\overline{\text{TZ3}}$) can be selected via the DCTRIPSEL bits to generate the Digital Compare A High and Low (DCAH/L) and Digital Compare B High and Low (DCBH/L) signals. Then, the configuration of the TZDCSEL register qualifies the actions on the selected DCAH/L and DCBH/L signals, which generate the DCAEVT1/2 and DCBEVT1/2 events (Event Qualification A and B).

NOTE: The $\overline{\text{TZn}}$ signals, when used as a DCEVT tripping functions, are treated as a normal input signal and can be defined to be active high or active low inputs. EPWM outputs are asynchronously tripped when either the $\overline{\text{TZn}}$, DCAEVTx.force, or DCBEVTx.force signals are active. For the condition to remain latched, a minimum of 3*TBCLK sync pulse width is required. If pulse width is < 3*TBCLK sync pulse width, the trip condition may or may not get latched by CBC or OST latches.

The DCAEVT1/2 and DCBEVT1/2 events can then be filtered to provide a filtered version of the event signals (DCEVTFILT) or the filtering can be bypassed. Filtering is discussed further in section 2.9.3.2. Either the DCAEVT1/2 and DCBEVT1/2 event signals or the filtered DCEVTFILT event signals can generate a force to the trip zone module, a TZ interrupt, an ADC SOC, or a PWM sync signal.

- **force signal:**

  DCAEVT1/2.force signals force trip zone conditions which either directly influence the output on the EPWMxA pin (via TZCTL[DCAEVT1 or DCAEVT2] configurations) or, if the DCAEVT1/2 signals are selected as one-shot or cycle-by-cycle trip sources (via the TZSEL register), the DCAEVT1/2.force signals can effect the trip action via the TZCTL[TZA] configuration. The DCBEVT1/2.force signals behaves similarly, but affect the EPWMxB output pin instead of the EPWMxA output pin.

  The priority of conflicting actions on the TZCTL register is as follows (highest priority overrides lower priority):

  Output EPWMxA: TZA (highest) -> DCAEVT1 -> DCAEVT2 (lowest)

  Output EPWMxB: TZB (highest) -> DCBEVT1 -> DCBEVT2 (lowest)

- **interrupt signal:**

  DCAEVT1/2.interrupt signals generate trip zone interrupts to the VIM. To enable the interrupt, the user must set the DCAEVT1, DCAEVT2, DCBEVT1, or DCBEVT2 bits in the TZEINT register. Once one of these events occurs, an EPWMxTZINT interrupt is triggered, and the corresponding bit in the TZCLR register must be set in order to clear the interrupt.

- **soc signal:**

  The DCAEVT1.soc signal interfaces with the event-trigger submodule and can be selected as an event which generates an ADC start-of-conversion-A (SOCA) pulse via the ETSEL[SOCASEL] bit. Likewise, the DCBEVT1.soc signal can be selected as an event which generates an ADC start-of-conversion-B (SOCB) pulse via the ETSEL[SOCBSEL] bit.

- **sync signal:**

  The DCAEVT1.sync and DCBEVT1.sync events are ORed with the EPWMxSYNCI input signal and the TBCTL[SWFSYNC] signal to generate a synchronization pulse to the time-base counter.

Figure 35-45 and Figure 35-46 show how the DCAEVT1, DCAEVT2, or DCEVTFILT signals are processed to generate the digital compare A event force, interrupt, soc and sync signals.

Figure 35-45. DCAEVT1 Event Triggering

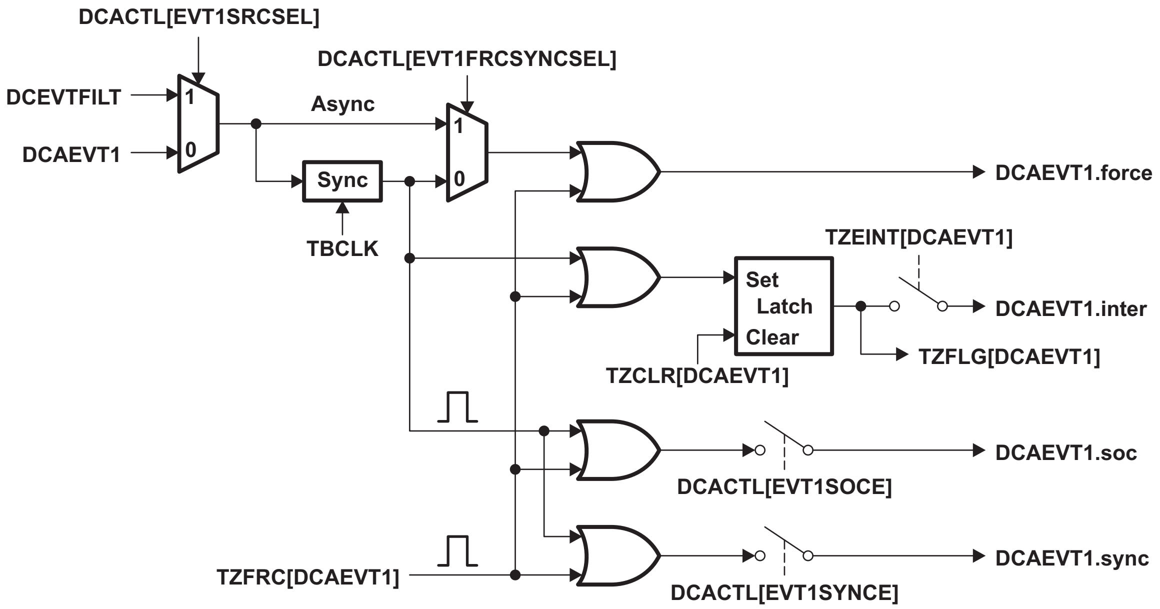


Figure 35-46. DCAEVT2 Event Triggering

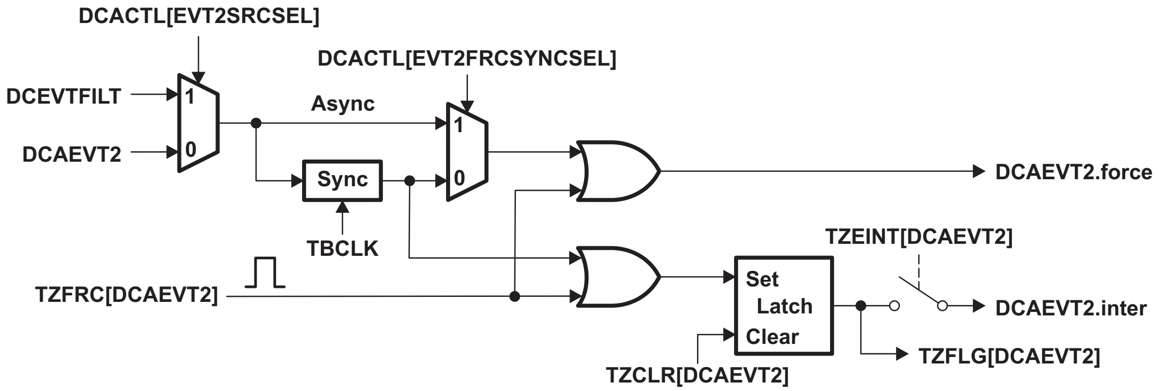


Figure 35-47 and Figure 35-48 show how the DCBEVT1, DCBEVT2, or DCEVTFILT signals are processed to generate the digital compare B event force, interrupt, soc and sync signals.

Figure 35-47. DCBEVT1 Event Triggering

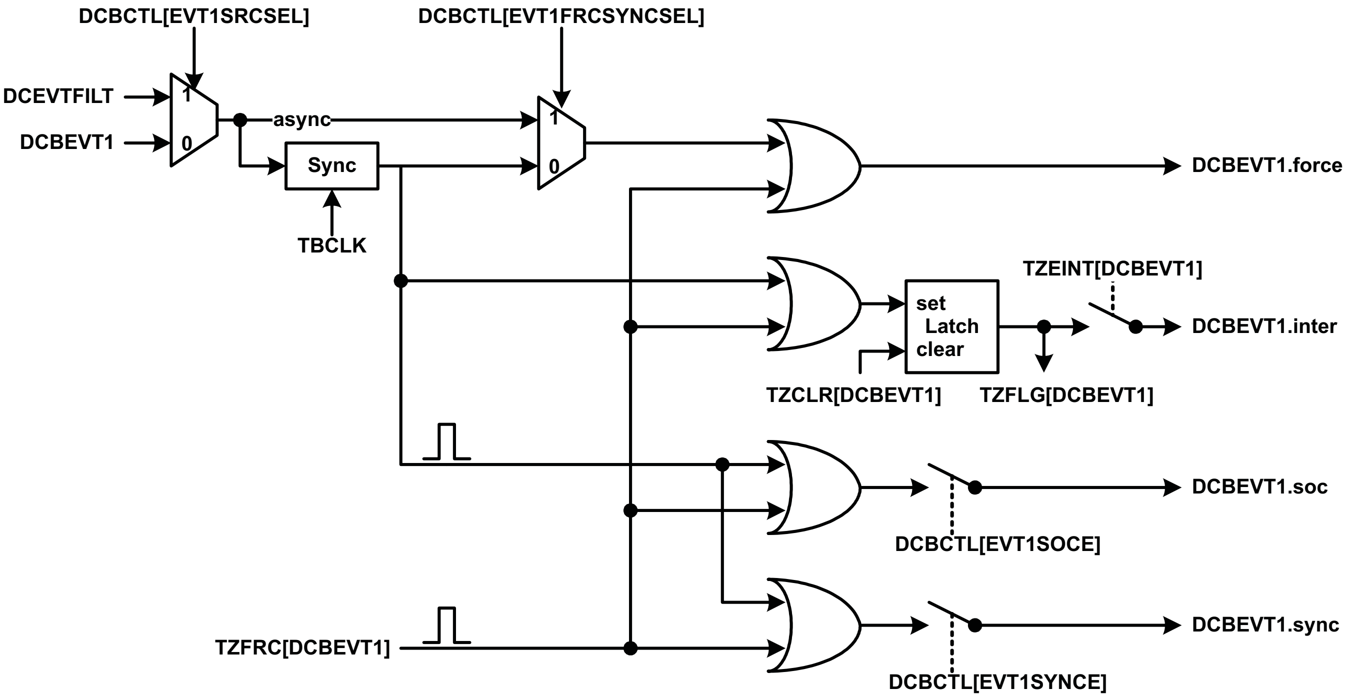
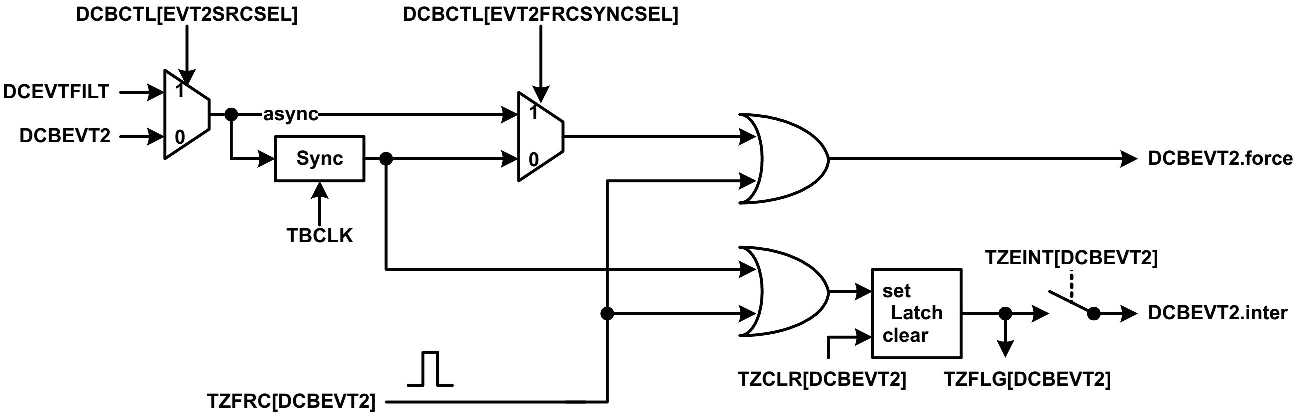


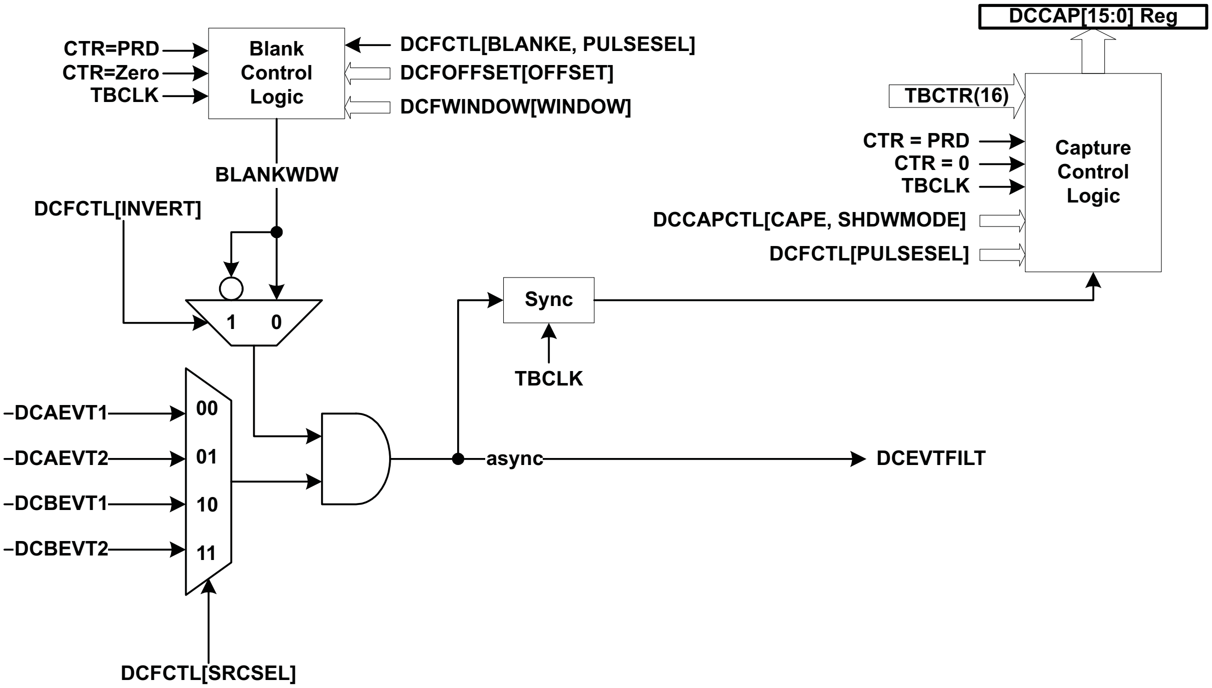
Figure 35-48. DCBEVT2 Event Triggering



35.2.9.3.2 Event Filtering

The DCAEVT1/2 and DCBEVT1/2 events can be filtered via event filtering logic to remove noise by optionally blanking events for a certain period of time. This is useful for cases where the analog comparator outputs may be selected to trigger DCAEVT1/2 and DCBEVT1/2 events, and the blanking logic is used to filter out potential noise on the signal prior to tripping the PWM outputs or generating an interrupt or ADC start-of-conversion. The event filtering can also capture the TBCTR value of the trip event. Figure 35-49 shows the details of the event filtering logic.

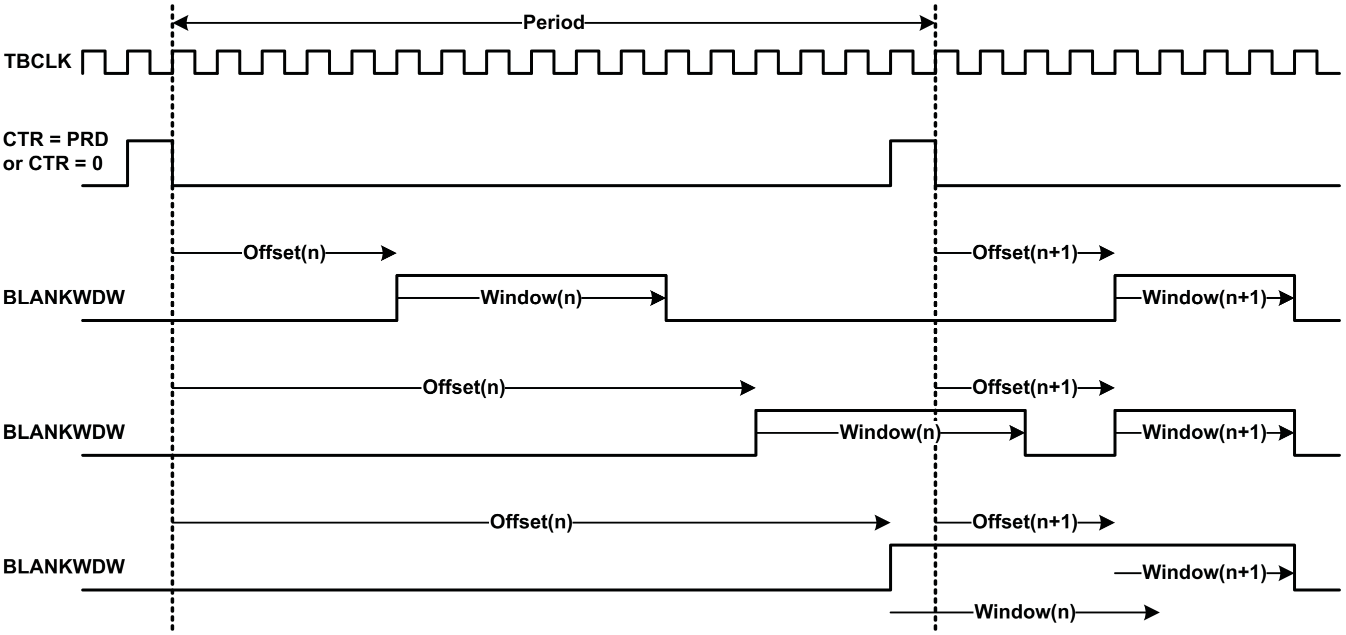
Figure 35-49. Event Filtering



If the blanking logic is enabled, one of the digital compare events – DCAEVT1, DCAEVT2, DCBEVT1, DCBEVT2 – is selected for filtering. The blanking window, which filters out all event occurrences on the signal while it is active, will be aligned to either a CTR = PRD pulse or a CTR = 0 pulse (configured by the DCFCTL[PULSESEL] bits). An offset value in TBCLK counts is programmed into the DCFOFFSET register, which determines at what point after the CTR = PRD or CTR = 0 pulse the blanking window starts. The duration of the blanking window, in number of TBCLK counts after the offset counter expires, is written to the DCFWINDOW register by the application. During the blanking window, all events are ignored. Before and after the blanking window ends, events can generate soc, sync, interrupt, and force signals as before.

Figure 35-50 shows several timing conditions for the offset and blanking window within an ePWM period. Notice that if the blanking window crosses the CTR = 0 or CTR = PRD boundary, the next window still starts at the same offset value after the CTR = 0 or CTR = PRD pulse.

Figure 35-50. Blanking Window Timing Diagram



35.2.10 Proper Interrupt Initialization Procedure

When the ePWM peripheral clock is enabled it may be possible that interrupt flags may be set due to spurious events due to the ePWM registers not being properly initialized. The proper procedure for initializing the ePWM peripheral is as follows:

1. Disable global interrupts (CPU INTM flag)
2. Disable ePWM interrupts
3. Set TBCLKSYNC = 0
4. Initialize peripheral registers
5. Set TBCLKSYNC = 1
6. Clear any spurious ePWM flags (including interrupt flags)
7. Enable ePWM interrupts
8. Enable global interrupts

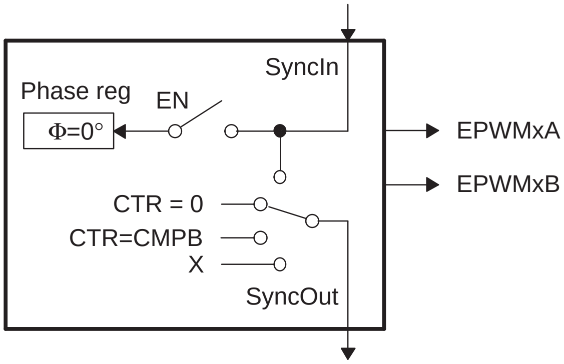
35.3 Application Examples

An ePWM module has all the local resources necessary to operate completely as a standalone module or to operate in synchronization with other identical ePWM modules.

35.3.1 Overview of Multiple Modules

Previously in this user's guide, all discussions have described the operation of a single module. To facilitate the understanding of multiple modules working together in a system, the ePWM module described in reference is represented by the more simplified block diagram shown in Figure 35-51. This simplified ePWM block shows only the key resources needed to explain how a multiswitch power topology is controlled with multiple ePWM modules working together.

Figure 35-51. Simplified ePWM Module



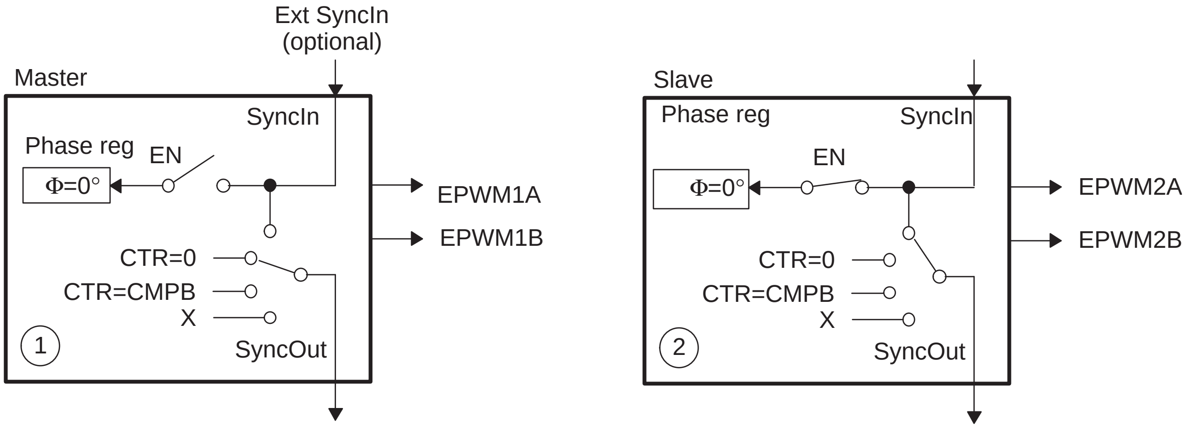
35.3.2 Key Configuration Capabilities

The key configuration choices available to each module are as follows:

- Options for SyncIn
  - Load own counter with phase register on an incoming sync strobe—enable (EN) switch closed
  - Do nothing or ignore incoming sync strobe—enable switch open
  - Sync flow-through - SyncOut connected to SyncIn
  - Master mode, provides a sync at PWM boundaries—SyncOut connected to CTR = PRD
  - Master mode, provides a sync at any programmable point in time—SyncOut connected to CTR = CMPB
  - Module is in standalone mode and provides No sync to other modules—SyncOut connected to X (disabled)
- Options for SyncOut
  - Sync flow-through - SyncOut connected to SyncIn
  - Master mode, provides a sync at PWM boundaries—SyncOut connected to CTR = PRD
  - Master mode, provides a sync at any programmable point in time—SyncOut connected to CTR = CMPB
  - Module is in standalone mode and provides No sync to other modules—SyncOut connected to X (disabled)

For each choice of SyncOut, a module may also choose to load its own counter with a new phase value on a SyncIn strobe input or choose to ignore it, i.e., via the enable switch. Although various combinations are possible, the two most common—master module and slave module modes—are shown in Figure 35-52.

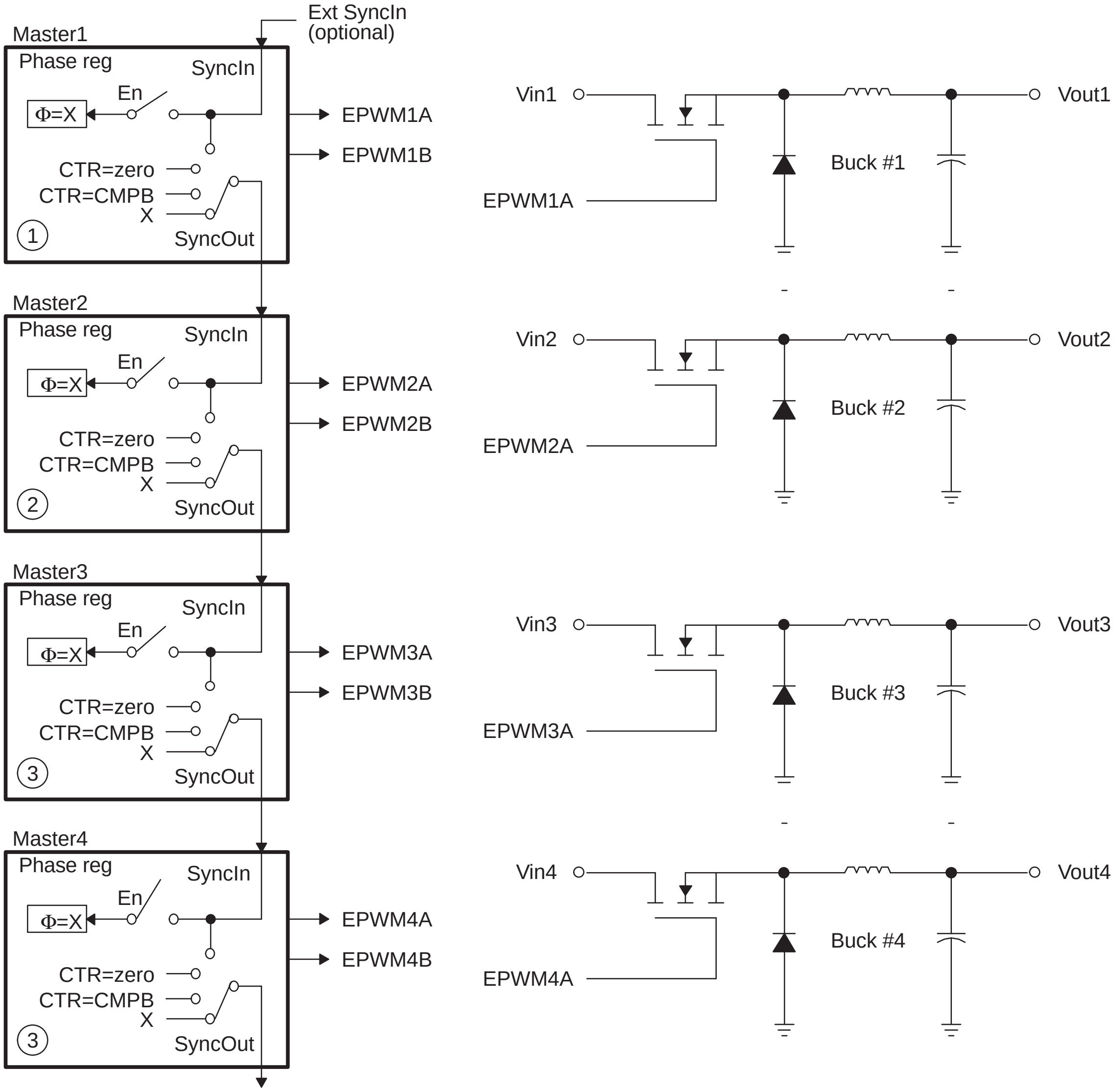
Figure 35-52. EPWM1 Configured as a Typical Master, EPWM2 Configured as a Slave



35.3.3 Controlling Multiple Buck Converters With Independent Frequencies

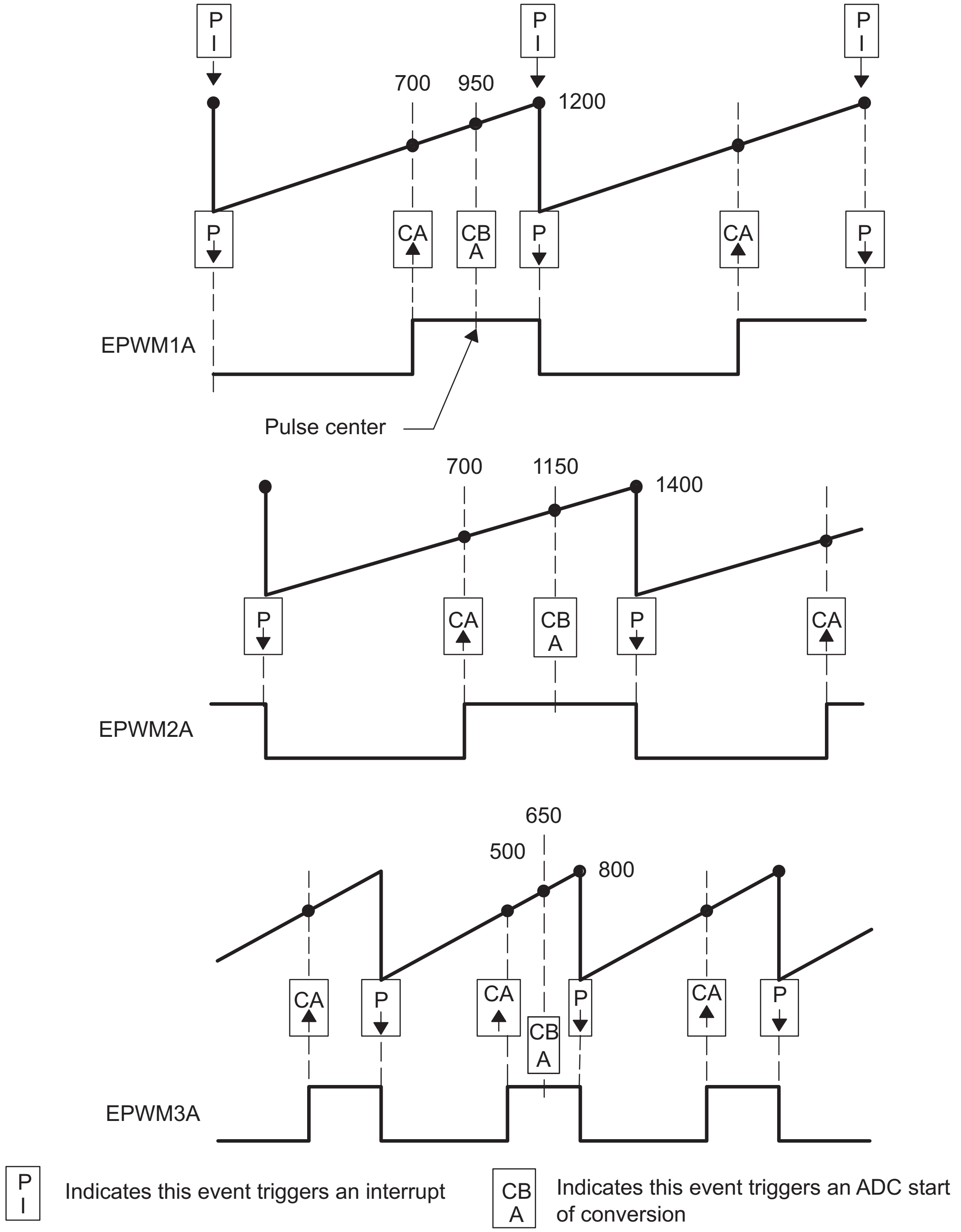
One of the simplest power converter topologies is the buck. A single ePWM module configured as a master can control two buck stages with the same PWM frequency. If independent frequency control is required for each buck converter, then one ePWM module must be allocated for each converter stage. Figure 35-53 shows four buck stages, each running at independent frequencies. In this case, all four ePWM modules are configured as Masters and no synchronization is used. Figure 35-54 shows the waveforms generated by the setup shown in Figure 35-53; note that only three waveforms are shown, although there are four stages.

Figure 35-53. Control of Four Buck Stages. Here $F_{PWM1} \neq F_{PWM2} \neq F_{PWM3} \neq F_{PWM4}$



NOTE: Θ = X indicates value in phase register is a "don't care"

Figure 35-54. Buck Waveforms for Figure 35-53 (Note: Only three bucks shown here)


P
I
700
950
1200
CA
CB
A
EPWM1A
Pulse center
1150
1400
EPWM2A
650
500
800
EPWM3A
Indicates this event triggers an interrupt
Indicates this event triggers an ADC start of conversion

Example 35-8. Configuration for Example in Figure 35-54

```
//=====================================================================
// (Note: code for only 3 modules shown)
// Initialization Time
//========================
// EPWM Module 1 config
   EPwm1Regs.TBPRD = 1200;                         // Period = 1201 TBCLK counts
   EPwm1Regs.TBPHS.half.TBPHS = 0;                 // Set Phase register to zero
   EPwm1Regs.TBCTL.bit.CTRMODE = TB_COUNT_UP;      // Asymmetrical mode
   EPwm1Regs.TBCTL.bit.PHSEN = TB_DISABLE;         // Phase loading disabled
   EPwm1Regs.TBCTL.bit.PRDLD = TB_SHADOW;
   EPwm1Regs.TBCTL.bit.SYNCOSEL = TB_SYNC_DISABLE;
   EPwm1Regs.CMPCTL.bit.SHDWAMODE = CC_SHADOW;
   EPwm1Regs.CMPCTL.bit.SHDWBMODE = CC_SHADOW;
   EPwm1Regs.CMPCTL.bit.LOADAMODE = CC_CTR_ZERO;   // load on CTR=Zero
   EPwm1Regs.CMPCTL.bit.LOADBMODE = CC_CTR_ZERO;   // load on CTR=Zero
   EPwm1Regs.AQCTLA.bit.PRD = AQ_CLEAR;
   EPwm1Regs.AQCTLA.bit.CAU = AQ_SET;
// EPWM Module 2 config
   EPwm2Regs.TBPRD = 1400;                         // Period = 1401 TBCLK counts
   EPwm2Regs.TBPHS.half.TBPHS = 0;                 // Set Phase register to zero
   EPwm2Regs.TBCTL.bit.CTRMODE = TB_COUNT_UP;      // Asymmetrical mode
   EPwm2Regs.TBCTL.bit.PHSEN = TB_DISABLE;         // Phase loading disabled
   EPwm2Regs.TBCTL.bit.PRDLD = TB_SHADOW;
   EPwm2Regs.TBCTL.bit.SYNCOSEL = TB_SYNC_DISABLE;
   EPwm2Regs.CMPCTL.bit.SHDWAMODE = CC_SHADOW;
   EPwm2Regs.CMPCTL.bit.SHDWBMODE = CC_SHADOW;
   EPwm2Regs.CMPCTL.bit.LOADAMODE = CC_CTR_ZERO;   // load on CTR=Zero
   EPwm2Regs.CMPCTL.bit.LOADBMODE = CC_CTR_ZERO;   // load on CTR=Zero
   EPwm2Regs.AQCTLA.bit.PRD = AQ_CLEAR;
   EPwm2Regs.AQCTLA.bit.CAU = AQ_SET;
// EPWM Module 3 config
   EPwm3Regs.TBPRD = 800;                           // Period = 801 TBCLK counts
   EPwm3Regs.TBPHS.half.TBPHS = 0;                  // Set Phase register to zero
   EPwm3Regs.TBCTL.bit.CTRMODE = TB_COUNT_UP;
   EPwm3Regs.TBCTL.bit.PHSEN = TB_DISABLE;          // Phase loading disabled
   EPwm3Regs.TBCTL.bit.PRDLD = TB_SHADOW;
   EPwm3Regs.TBCTL.bit.SYNCOSEL = TB_SYNC_DISABLE;
   EPwm3Regs.CMPCTL.bit.SHDWAMODE = CC_SHADOW;
   EPwm3Regs.CMPCTL.bit.SHDWBMODE = CC_SHADOW;
   EPwm3Regs.CMPCTL.bit.LOADAMODE = CC_CTR_ZERO;    // load on CTR=Zero
   EPwm3Regs.CMPCTL.bit.LOADBMODE = CC_CTR_ZERO;    // load on CTR=Zero
   EPwm3Regs.AQCTLA.bit.PRD = AQ_CLEAR;
   EPwm3Regs.AQCTLA.bit.CAU = AQ_SET;
//
// Run Time (Note: Example execution of one run-time instant)
//=========================================================
   EPwm1Regs.CMPA.half.CMPA = 700;                 // adjust duty for output EPWM1A
   EPwm2Regs.CMPA.half.CMPA = 700;                 // adjust duty for output EPWM2A
   EPwm3Regs.CMPA.half.CMPA = 500;                 // adjust duty for output EPWM3A
```

35.3.4 Controlling Multiple Buck Converters With Same Frequencies

If synchronization is a requirement, ePWM module 2 can be configured as a slave and can operate at integer multiple (N) frequencies of module 1. The sync signal from master to slave ensures these modules remain locked. Figure 35-55 shows such a configuration; Figure 35-56 shows the waveforms generated by the configuration.

Figure 35-55. Control of Four Buck Stages. (Note: F_{PWM2} = N x F_{PWM1})

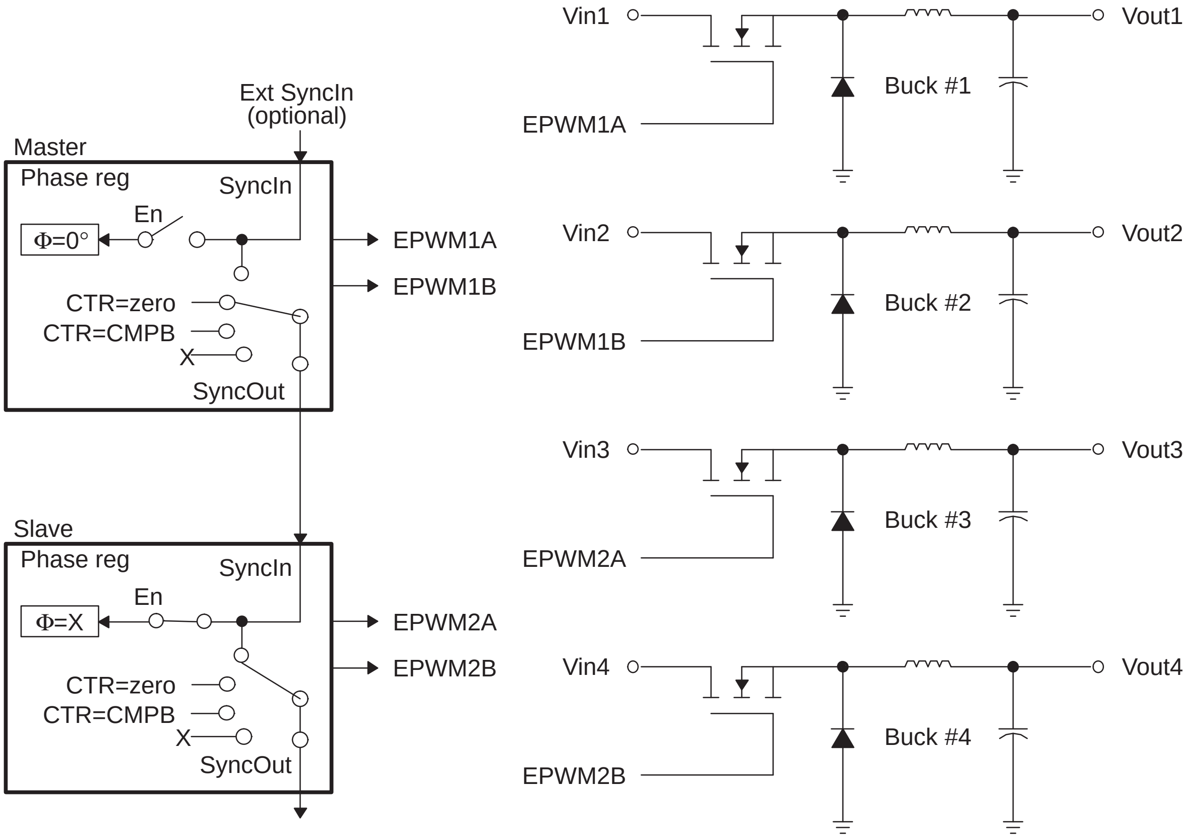
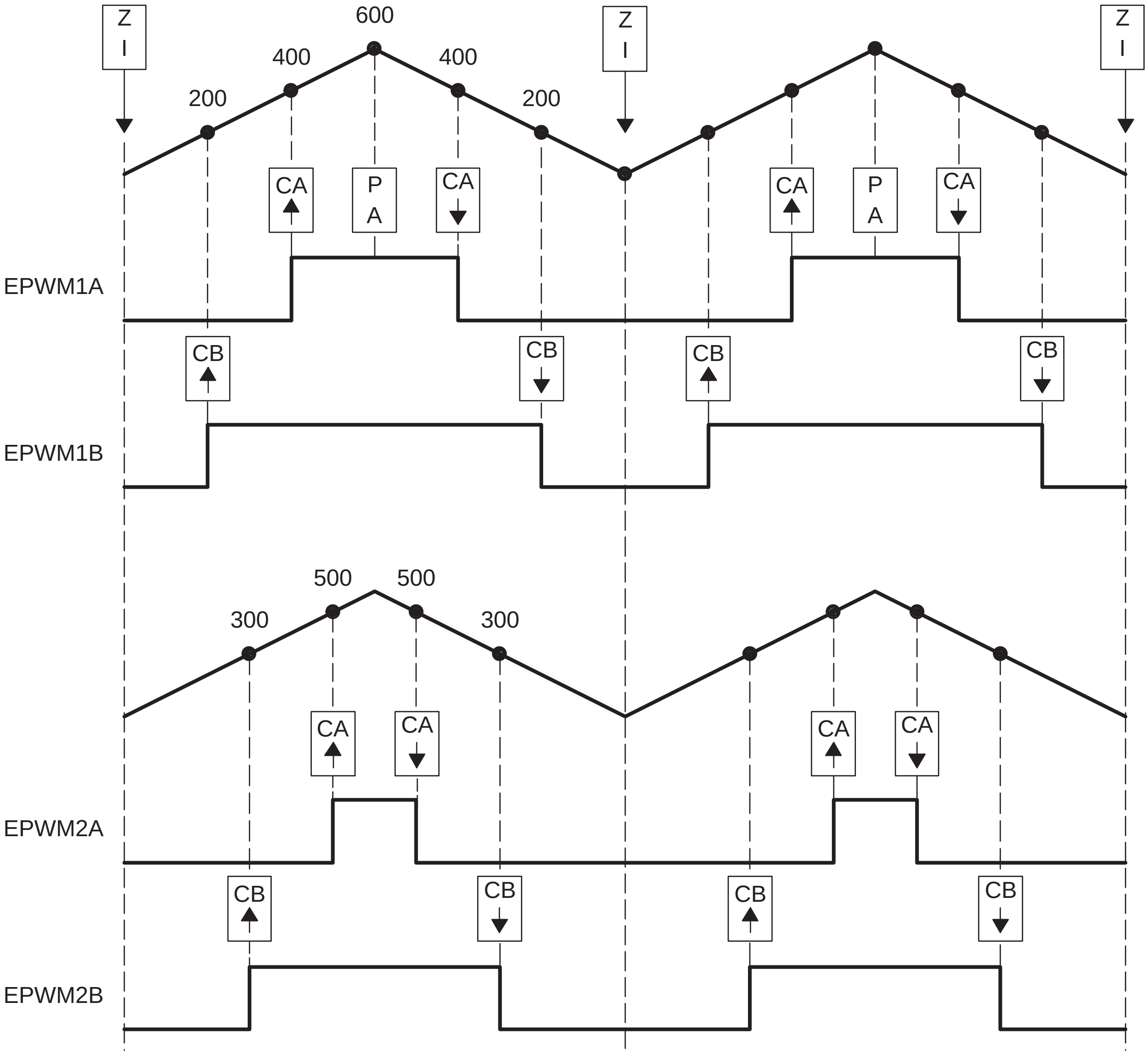


Figure 35-56. Buck Waveforms for Figure 35-55 (Note: $F_{PWM2} = F_{PWM1)}$)



Example 35-9. Code Snippet for Configuration in Figure 35-55

```
//========================
// EPWM Module 1 config
     EPwm1Regs.TBPRD = 600;                         // Period = 1200 TBCLK counts
     EPwm1Regs.TBPHS.half.TBPHS = 0;                // Set Phase register to zero
     EPwm1Regs.TBCTL.bit.CTRMODE = TB_COUNT_UPDOWN; // Symmetrical mode
     EPwm1Regs.TBCTL.bit.PHSEN = TB_DISABLE;        // Master module
     EPwm1Regs.TBCTL.bit.PRDLD = TB_SHADOW;
     EPwm1Regs.TBCTL.bit.SYNCOSEL = TB_CTR_ZERO;    // Sync down-stream module
     EPwm1Regs.CMPCTL.bit.SHDWAMODE = CC_SHADOW;
     EPwm1Regs.CMPCTL.bit.SHDWBMODE = CC_SHADOW;
     EPwm1Regs.CMPCTL.bit.LOADAMODE = CC_CTR_ZERO;  // load on CTR=Zero
     EPwm1Regs.CMPCTL.bit.LOADBMODE = CC_CTR_ZERO;  // load on CTR=Zero
     EPwm1Regs.AQCTLA.bit.CAU = AQ_SET;             // set actions for EPWM1A
     EPwm1Regs.AQCTLA.bit.CAD = AQ_CLEAR;
     EPwm1Regs.AQCTLB.bit.CBU = AQ_SET;             // set actions for EPWM1B
     EPwm1Regs.AQCTLB.bit.CBD = AQ_CLEAR;
// EPWM Module 2 config
     EPwm2Regs.TBPRD = 600;                         // Period = 1200 TBCLK counts
     EPwm2Regs.TBPHS.half.TBPHS = 0;                // Set Phase register to zero
     EPwm2Regs.TBCTL.bit.CTRMODE = TB_COUNT_UPDOWN; // Symmetrical mode
     EPwm2Regs.TBCTL.bit.PHSEN = TB_ENABLE;         // Slave module
     EPwm2Regs.TBCTL.bit.PRDLD = TB_SHADOW;
     EPwm2Regs.TBCTL.bit.SYNCOSEL = TB_SYNC_IN;     // sync flow-through
     EPwm2Regs.CMPCTL.bit.SHDWAMODE = CC_SHADOW;
     EPwm2Regs.CMPCTL.bit.SHDWBMODE = CC_SHADOW;
     EPwm2Regs.CMPCTL.bit.LOADAMODE = CC_CTR_ZERO;  // load on CTR=Zero
     EPwm2Regs.CMPCTL.bit.LOADBMODE = CC_CTR_ZERO;  // load on CTR=Zero
     EPwm2Regs.AQCTLA.bit.CAU = AQ_SET;             // set actions for EPWM2A
     EPwm2Regs.AQCTLA.bit.CAD = AQ_CLEAR;
     EPwm2Regs.AQCTLB.bit.CBU = AQ_SET;             // set actions for EPWM2B
     EPwm2Regs.AQCTLB.bit.CBD = AQ_CLEAR;
//
// Run Time (Note: Example execution of one run-time instance)
//===========================================================
    EPwm1Regs.CMPA.half.CMPA = 400;                // adjust duty for output EPWM1A
    EPwm1Regs.CMPB = 200;                          // adjust duty for output EPWM1B
    EPwm2Regs.CMPA.half.CMPA = 500;                // adjust duty for output EPWM2A
    EPwm2Regs.CMPB = 300;                          // adjust duty for output EPWM2B
```

35.3.5 Controlling Multiple Half H-Bridge (HHB) Converters

Topologies that require control of multiple switching elements can also be addressed with these same ePWM modules. It is possible to control a Half-H bridge stage with a single ePWM module. This control can be extended to multiple stages. Figure 35-57 shows control of two synchronized Half-H bridge stages where stage 2 can operate at integer multiple (N) frequencies of stage 1. Figure 35-58 shows the waveforms generated by the configuration shown in Figure 35-57.

Module 2 (slave) is configured for Sync flow-through; if required, this configuration allows for a third Half-H bridge to be controlled by PWM module 3 and also, most importantly, to remain in synchronization with master module 1.

Figure 35-57. Control of Two Half-H Bridge Stages (F_{PWM2} = N x F_{PWM1})

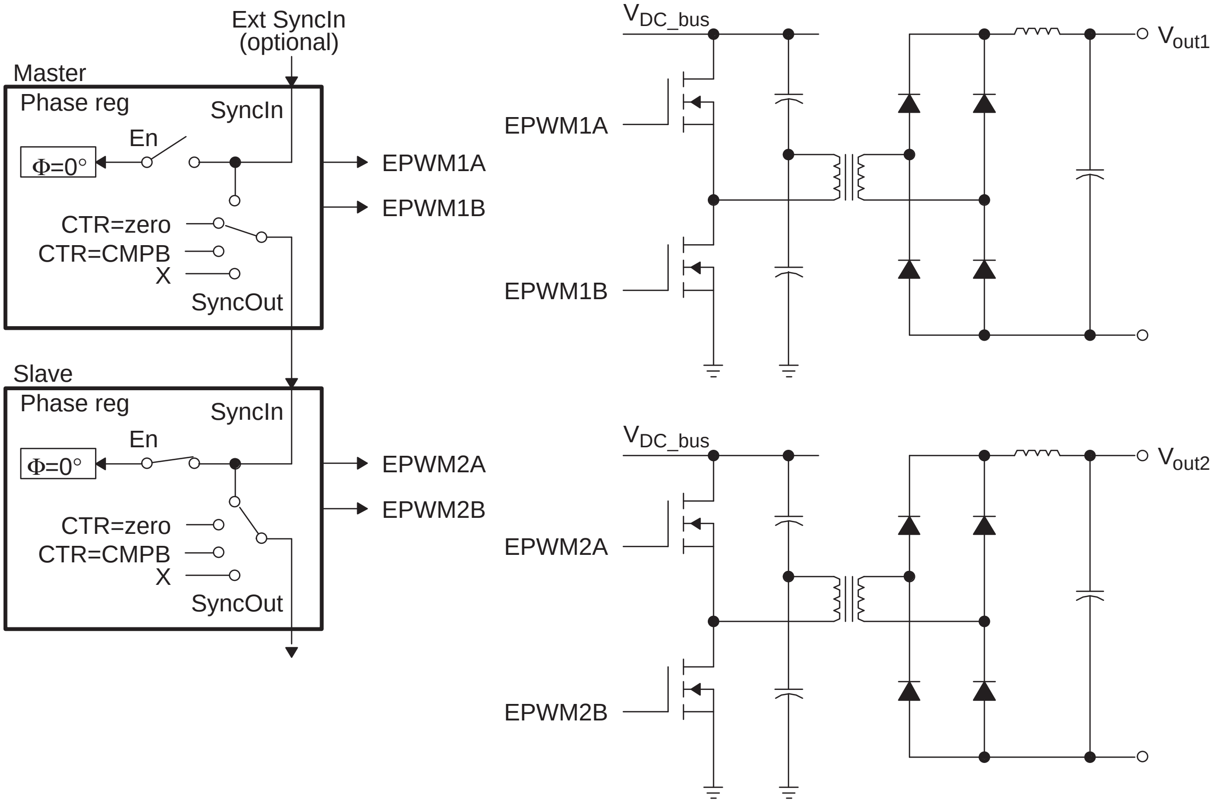
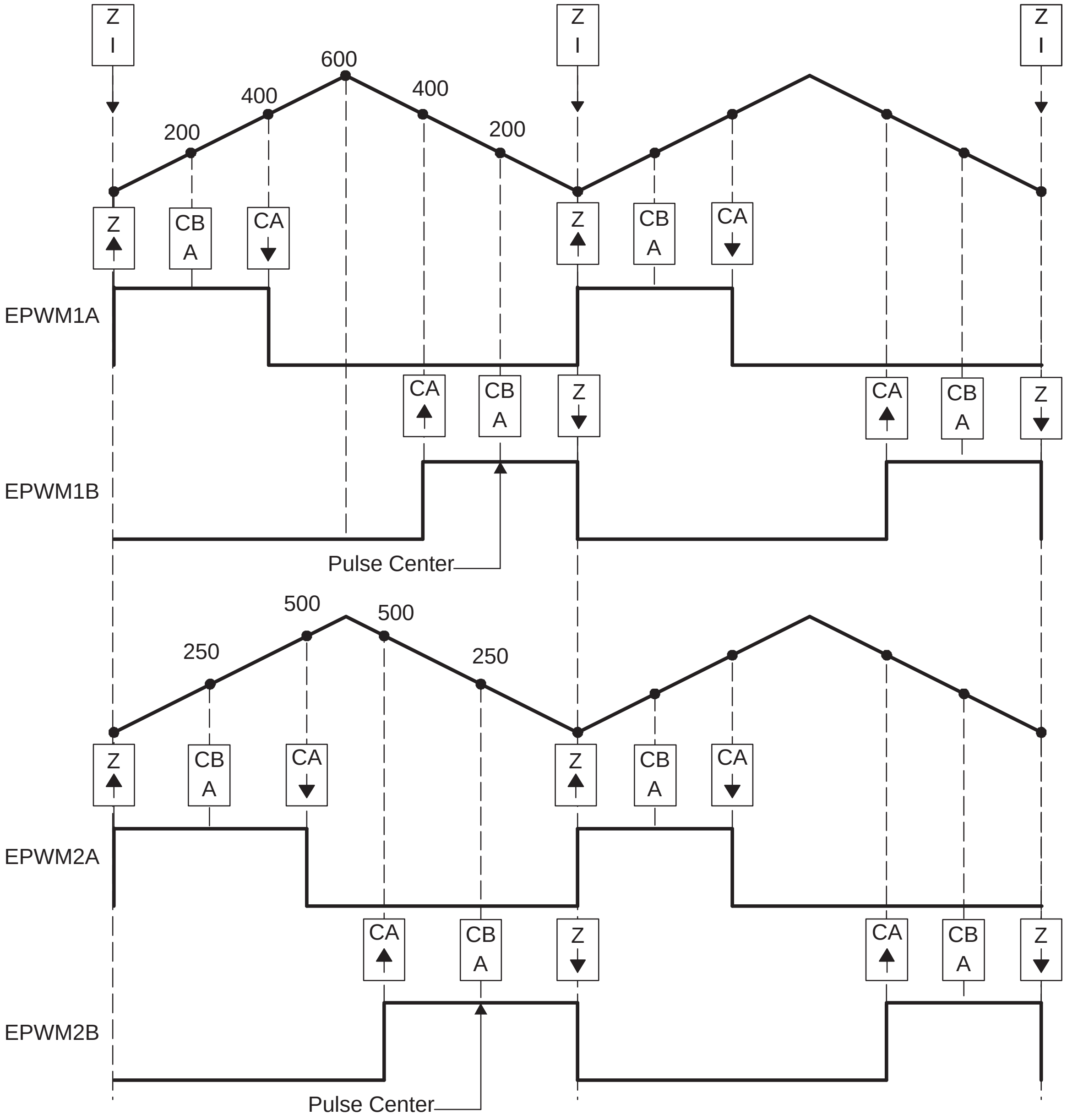


Figure 35-58. Half-H Bridge Waveforms for Figure 35-57 (Note: Here $F_{PWM2} = F_{PWM1}$)



Example 35-10. Code Snippet for Configuration in Figure 35-57

```
//=====================================================================
// Config
//=====================================================================
// Initialization Time
//========================
// EPWM Module 1 config
     EPwm1Regs.TBPRD = 600;                         // Period = 1200 TBCLK counts
     EPwm1Regs.TBPHS.half.TBPHS = 0;                // Set Phase register to zero
     EPwm1Regs.TBCTL.bit.CTRMODE = TB_COUNT_UPDOWN; // Symmetrical mode
     EPwm1Regs.TBCTL.bit.PHSEN = TB_DISABLE;        // Master module
     EPwm1Regs.TBCTL.bit.PRDLD = TB_SHADOW;
     EPwm1Regs.TBCTL.bit.SYNCOSEL = TB_CTR_ZERO;    // Sync down-stream module
     EPwm1Regs.CMPCTL.bit.SHDWAMODE = CC_SHADOW;
     EPwm1Regs.CMPCTL.bit.SHDWBMODE = CC_SHADOW;
     EPwm1Regs.CMPCTL.bit.LOADAMODE = CC_CTR_ZERO;  // load on CTR=Zero
     EPwm1Regs.CMPCTL.bit.LOADBMODE = CC_CTR_ZERO;  // load on CTR=Zero
     EPwm1Regs.AQCTLA.bit.ZRO = AQ_SET;             // set actions for EPWM1A
     EPwm1Regs.AQCTLA.bit.CAU = AQ_CLEAR;
     EPwm1Regs.AQCTLB.bit.ZRO = AQ_CLEAR;           // set actions for EPWM1B
     EPwm1Regs.AQCTLB.bit.CAD = AQ_SET;
// EPWM Module 2 config
     EPwm2Regs.TBPRD = 600;                         // Period = 1200 TBCLK counts
     EPwm2Regs.TBPHS.half.TBPHS = 0;                // Set Phase register to zero
     EPwm2Regs.TBCTL.bit.CTRMODE = TB_COUNT_UPDOWN; // Symmetrical mode
     EPwm2Regs.TBCTL.bit.PHSEN = TB_ENABLE;         // Slave module
     EPwm2Regs.TBCTL.bit.PRDLD = TB_SHADOW;
     EPwm2Regs.TBCTL.bit.SYNCOSEL = TB_SYNC_IN;     // sync flow-through
     EPwm2Regs.CMPCTL.bit.SHDWAMODE = CC_SHADOW;
     EPwm2Regs.CMPCTL.bit.SHDWBMODE = CC_SHADOW;
     EPwm2Regs.CMPCTL.bit.LOADAMODE = CC_CTR_ZERO;  // load on CTR=Zero
     EPwm2Regs.CMPCTL.bit.LOADBMODE = CC_CTR_ZERO;  // load on CTR=Zero
     EPwm2Regs.AQCTLA.bit.ZRO = AQ_SET;             // set actions for EPWM1A
     EPwm2Regs.AQCTLA.bit.CAU = AQ_CLEAR;
     EPwm2Regs.AQCTLB.bit.ZRO = AQ_CLEAR;           // set actions for EPWM1B
     EPwm2Regs.AQCTLB.bit.CAD = AQ_SET;
//============================================================
     EPwm1Regs.CMPA.half.CMPA = 400;                // adjust duty for output EPWM1A & EPWM1B

     EPwm1Regs.CMPB = 200;                          // adjust point-in-time for ADCSOC trigger
     EPwm2Regs.CMPA.half.CMPA = 500;                // adjust duty for output EPWM2A & EPWM2B
     EPwm2Regs.CMPB = 250;                          // adjust point-in-time for ADCSOC trigger
```

35.3.6 Controlling Dual 3-Phase Inverters for Motors (ACI and PMSM)

The idea of multiple modules controlling a single power stage can be extended to the 3-phase Inverter case. In such a case, six switching elements can be controlled using three PWM modules, one for each leg of the inverter. Each leg must switch at the same frequency and all legs must be synchronized. A master + two slaves configuration can easily address this requirement. Figure 35-59 shows how six PWM modules can control two independent 3-phase Inverters; each running a motor.

As in the cases shown in the previous sections, we have a choice of running each inverter at a different frequency (module 1 and module 4 are masters as in Figure 35-59), or both inverters can be synchronized by using one master (module 1) and five slaves. In this case, the frequency of modules 4, 5, and 6 (all equal) can be integer multiples of the frequency for modules 1, 2, 3 (also all equal).

Figure 35-59. Control of Dual 3-Phase Inverter Stages as Is Commonly Used in Motor Control

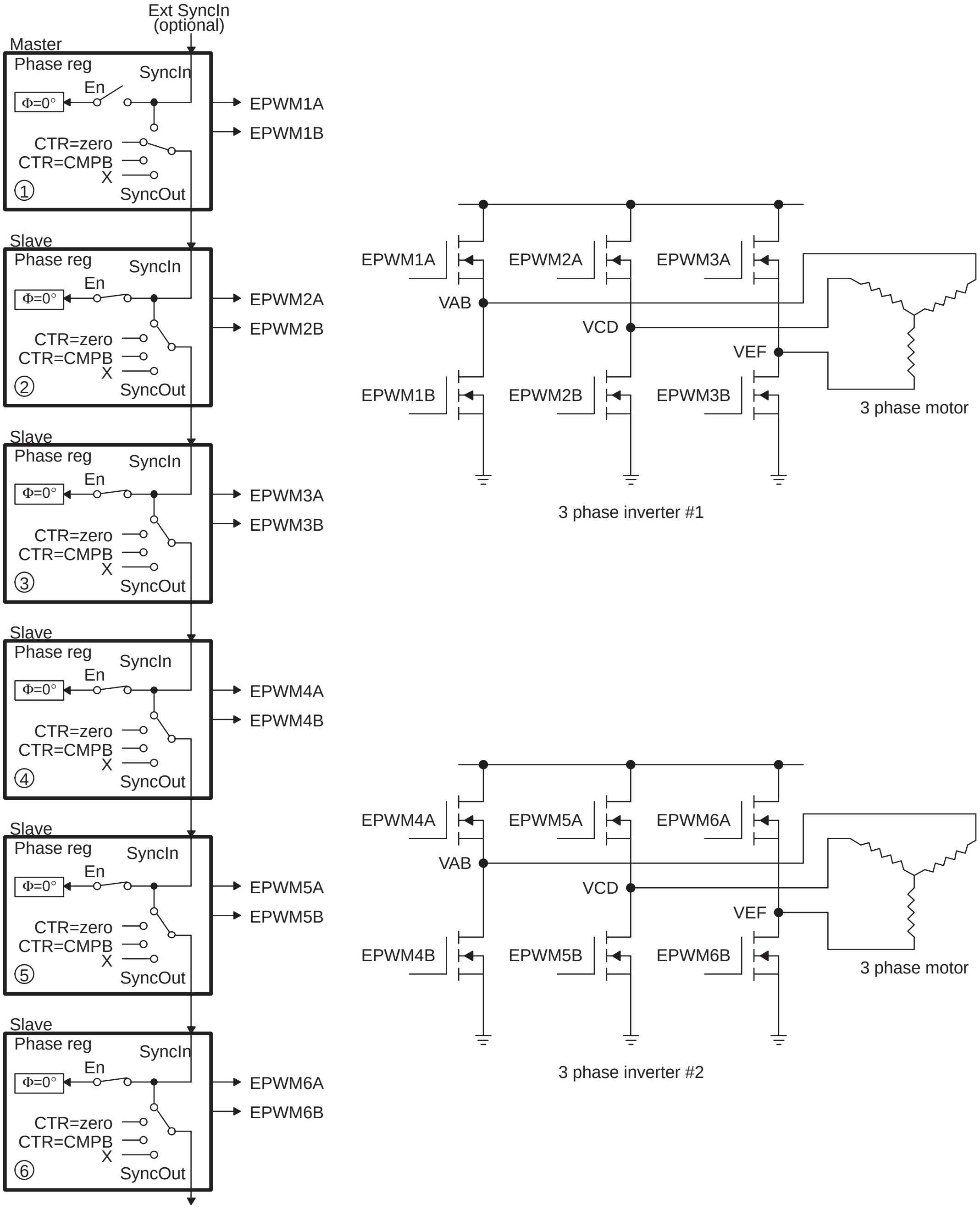
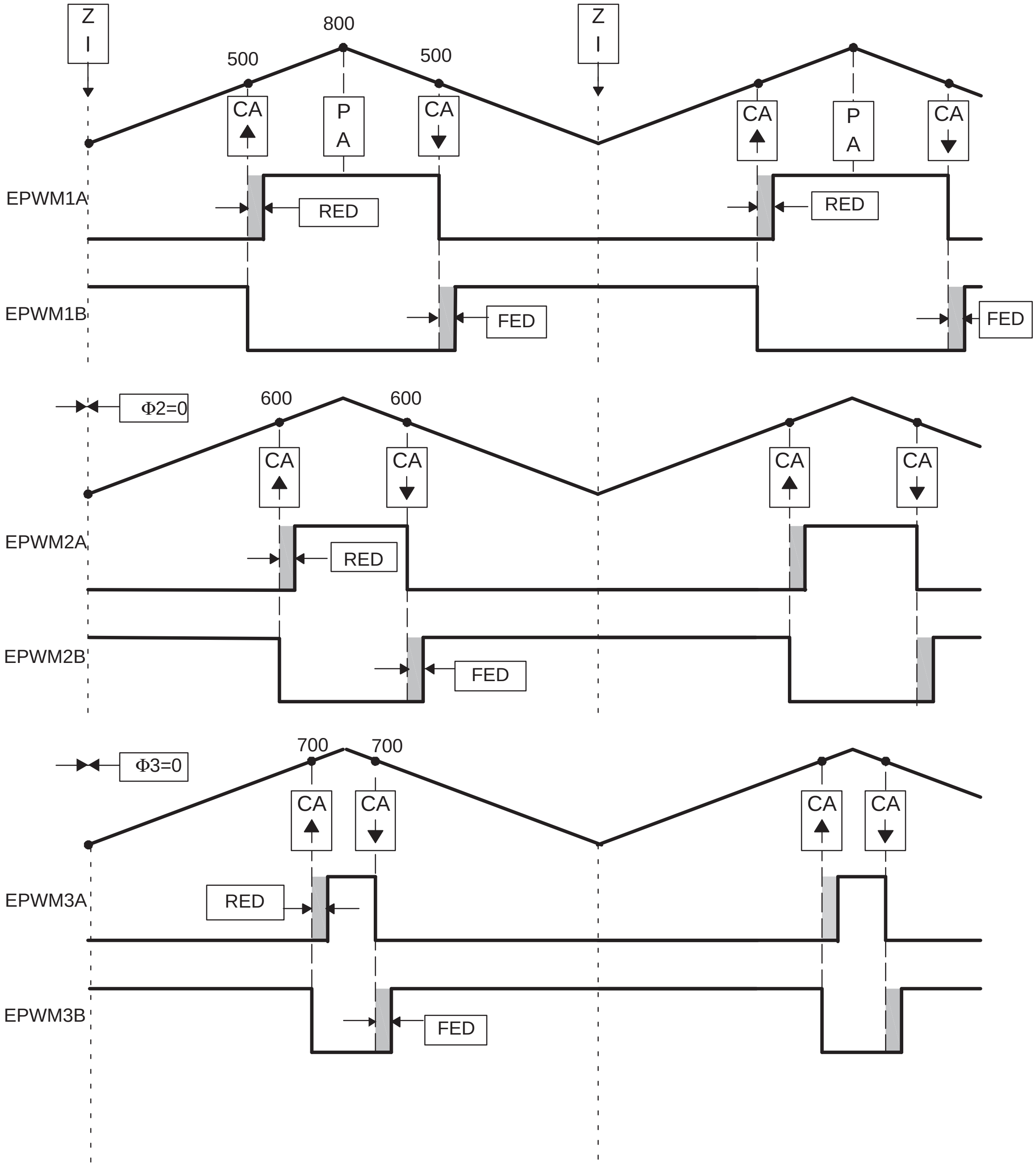


Figure 35-60. 3-Phase Inverter Waveforms for Figure 35-59 (Only One Inverter Shown)



35.3.7 Practical Applications Using Phase Control Between PWM Modules

So far, none of the examples have made use of the phase register (TBPHS). It has either been set to zero or its value has been a don't care. However, by programming appropriate values into TBPHS, multiple PWM modules can address another class of applications that rely on phase relationship between stages for correct operation. As described in the TB module section, a PWM module can be configured to allow a SyncIn pulse to cause the TBPHS register to be loaded into the TBCTR register. To illustrate this concept, Figure 35-61 shows a master and slave module with a phase relationship of 120°, i.e., the slave leads the master.

Figure 35-61. Configuring Two PWM Modules for Phase Control

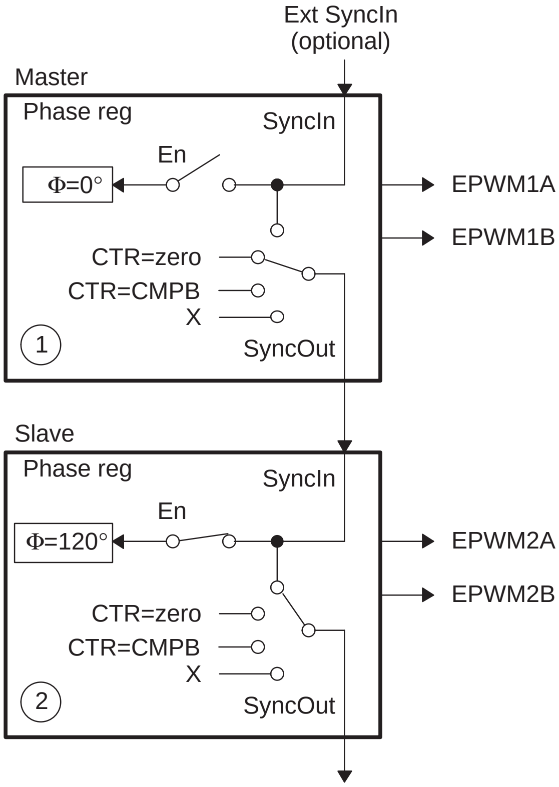
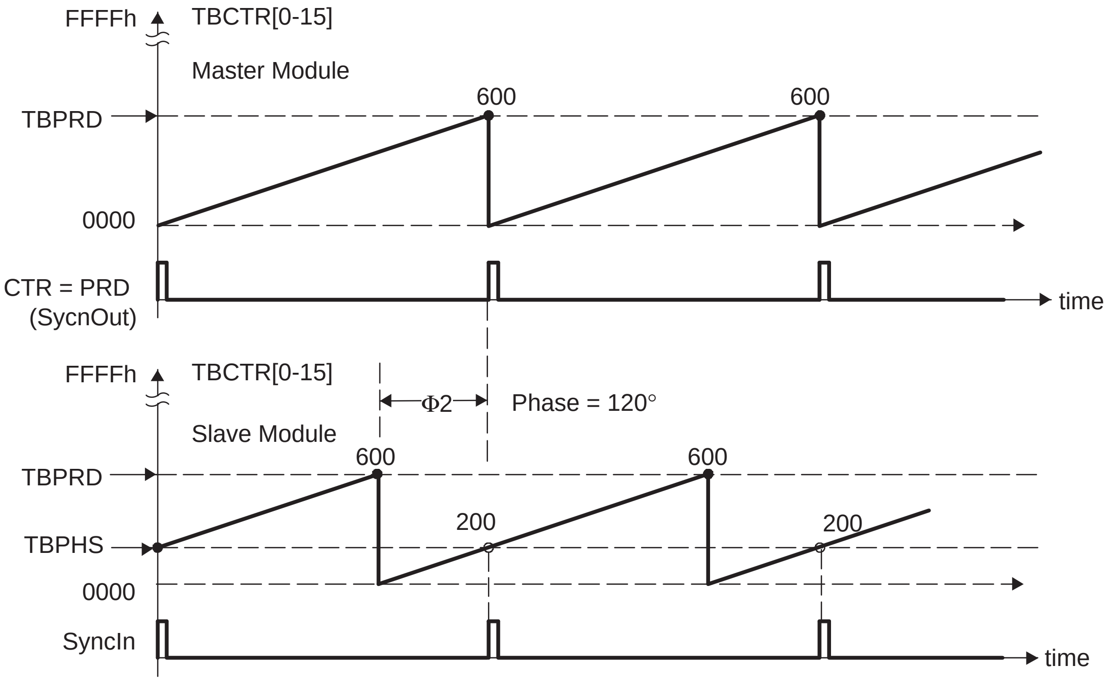


Figure 35-62 shows the associated timing waveforms for this configuration. Here, TBPRD = 600 for both master and slave. For the slave, TBPHS = 200 (200/600 × 360° = 120°). Whenever the master generates a SyncIn pulse (CTR = PRD), the value of TBPHS = 200 is loaded into the slave TBCTR register so the slave time-base is always leading the master's time-base by 120°.

Figure 35-62. Timing Waveforms Associated With Phase Control Between 2 Modules



35.4 ePWM Registers

Table 35-22 lists the complete ePWM module control and status register set grouped by submodule. Each register set is duplicated for each instance of the ePWM module. The base address for the control registers is FCF7 8C00h for ePWM1, FCF7 8D00h for ePWM2, FCF7 8E00h for ePWM3, FCF7 8F00h for ePWM4, FCF7 9000h for ePWM5, FCF7 9100h for ePWM6, and FCF7 9200h for ePWM7.

Table 35-22. ePWM Module Control and Status Register Set Grouped by Submodule

Address Offset	Name	Description	Section
		Time-Base Submodule Registers	
00h	TBSTS	Time-Base Status Register	Section 35.4.1.1
02h	TBCTL	Time-Base Control Register	Section 35.4.1.2
04h	TBPHS	Time-Base Phase Register	Section 35.4.1.3
08h	TBPRD	Time-Base Period Register	Section 35.4.1.4
0Ah	TBCTR	Time-Base Counter Register	Section 35.4.1.5
		Counter-Compare Submodule Registers	
0Ch	CMPCTL	Counter-Compare Control Register	Section 35.4.2.1
10h	CMPA	Counter-Compare A Register	Section 35.4.2.2
16h	CMPB	Counter-Compare B Register	Section 35.4.2.3
		Action-Qualifier Submodule Registers	
14h	AQCTLA	Action-Qualifier Control Register for Output A (EPWMxA)	Section 35.4.3.1
18h	AQSFRC	Action-Qualifier Software Force Register	Section 35.4.3.2
1Ah	AQCTLB	Action-Qualifier Control Register for Output B (EPWMxB)	Section 35.4.3.3
1Eh	AQCSFRC	Action-Qualifier Continuous S/W Force Register Set	Section 35.4.3.4
		Dead-Band Generator Submodule Registers	
1Ch	DBCTL	Dead-Band Generator Control Register	Section 35.4.4.1
20h	DBFED	Dead-Band Generator Falling Edge Delay Count Register	Section 35.4.4.2
22h	DBRED	Dead-Band Generator Rising Edge Delay Count Register	Section 35.4.4.3
		Trip-Zone Submodule Registers	
24h	TZDCSEL	Trip Zone Digital Compare Event Select Register	Section 35.4.5.1
26h	TZSEL	Trip-Zone Select Register	Section 35.4.5.2
28h	TZEINT	Trip-Zone Enable Interrupt Register	Section 35.4.5.3
2Ah	TZCTL	Trip-Zone Control Register	Section 35.4.5.4
2Ch	TZCLR	Trip-Zone Clear Register	Section 35.4.5.5
2Eh	TZFLG	Trip-Zone Flag Register	Section 35.4.5.6
32h	TZFRC	Trip-Zone Force Register	Section 35.4.5.7
		Event-Trigger Submodule Registers	
30h	ETSEL	Event-Trigger Selection Register	Section 35.4.6.1
34h	ETFLG	Event-Trigger Flag Register	Section 35.4.6.2
36h	ETPS	Event-Trigger Pre-Scale Register	Section 35.4.6.3
38h	ETFRC	Event-Trigger Force Register	Section 35.4.6.4
3Ah	ETCLR	Event-Trigger Clear Register	Section 35.4.6.5
		PWM-Chopper Submodule Registers	
3Eh	PCCTL	PWM-Chopper Control Register	Section 35.4.7.1
		Digital Compare Event Registers	
60h	DCACTL	Digital Compare A Control Register	Section 35.4.8.1
62h	DCTRIPSEL	Digital Compare Trip Select Register	Section 35.4.8.2
64h	DCFCTL	Digital Compare Filter Control Register	Section 35.4.8.3
66h	DCBCTL	Digital Compare B Control Register	Section 35.4.8.4
68h	DCFOFFSET	Digital Compare Filter Offset Register	Section 35.4.8.5
6Ah	DCCAPCTL	Digital Compare Capture Control Register	Section 35.4.8.6

Table 35-22. ePWM Module Control and Status Register Set Grouped by Submodule (continued)

Address Offset	Name	Description	Section
6Ch	DCFWINDOW	Digital Compare Filter Window Register	Section 35.4.8.7
6Eh	DCFOFFSETCNT	Digital Compare Filter Offset Counter Register	Section 35.4.8.8
70h	DCCAP	Digital Compare Counter Capture Register	Section 35.4.8.9
72h	DCFWINDOWCNT	Digital Compare Filter Window Counter Register	Section 35.4.8.10

35.4.1 Time-Base Submodule Registers

35.4.1.1 Time-Base Status Register (TBSTS)

Figure 35-63. Time-Base Status Register (TBSTS) [offset = 00h]

15-8
Reserved
R-0

7-3	2	1	0
Reserved	CTRMAX	SYNCI	CTRDIR
R-0	R/W1C-0	R/W1C-0	R-1

LEGEND: R/W = Read/Write; R = Read only; W1C = Write 1 to clear; -*n* = value after reset

Table 35-23. Time-Base Status Register (TBSTS) Field Descriptions

Bit	Field	Value	Description
15-3	Reserved	0	Reserved
2	CTRMAX		Time-Base Counter Max Latched Status Bit.
		0	Read: Indicates the time-base counter never reached its maximum value. Write: No effect.
		1	Read: Indicates that the time-base counter reached the maximum value 0xFFFF. Write: Clears the latched event.
1	SYNCI		Input Synchronization Latched Status Bit.
		0	Read: Indicates no external synchronization event has occurred. Write: No effect.
		1	Read: Indicates that an external synchronization event has occurred (EPWMxSYNCI). Write: Clears the latched event.
0	CTRDIR		Time-Base Counter Direction Status Bit. At reset, the counter is frozen; therefore, this bit has no meaning. To make this bit meaningful, you must first set the appropriate mode via TBCTL[CTRMODE].
		0	Time-Base Counter is currently counting down.
		1	Time-Base Counter is currently counting up.

35.4.1.2 Time-Base Control Register (TBCTL)

Figure 35-64. Time-Base Control Register (TBCTL) [offset = 02h]

15	14	13	12		10	9	8
FREE	SOFT	PHSDIR	CLKDIV			HSPCLKDIV	
R/W-0	R/W-0	R/W-0	R/W-0			R/W-0	

7	6	5	4	3	2	1	0
HSPCLKDIV	SWFSYNC	SYNCOSEL		PRDLD	PHSEN	CTRMODE	
R/W-1	R/W-0	R/W-0		R/W-0	R/W-0	R/W-3h	

LEGEND: R/W = Read/Write; -*n* = value after reset

Table 35-24. Time-Base Control Register (TBCTL) Field Descriptions

Bit	Field	Value	Description
15-14	FREE, SOFT		Emulation Mode Bits. These bits select the behavior of the ePWM time-base counter during emulation events:
		0	Stop after the next time-base counter increment or decrement.
		1h	Stop when counter completes a whole cycle: • Up-count mode: stop when the time-base counter = period (TBCTR = TBPRD) • Down-count mode: stop when the time-base counter = 0x0000 (TBCTR = 0x0000) • Up-down-count mode: stop when the time-base counter = 0x0000 (TBCTR = 0x0000)
		2h-3h	Free run
13	PHSDIR		Phase Direction Bit. This bit is only used when the time-base counter is configured in the up-down-count mode. The PHSDIR bit indicates the direction the time-base counter (TBCTR) will count after a synchronization event occurs and a new phase value is loaded from the phase (TBPHS) register. This is irrespective of the direction of the counter before the synchronization event.. In the up-count and down-count modes this bit is ignored.
		0	Count down after the synchronization event.
		1	Count up after the synchronization event.
12-10	CLKDIV		Time-base Clock Prescale Bits. These bits determine part of the time-base clock prescale value: TBCLK = VCLK3 / (HSPCLKDIV × CLKDIV)
		0	/1 (default on reset)
		1h	/2
		2h	/4
		3h	/8
		4h	/16
		5h	/32
		6h	/64
		7h	/128
9-7	HSPCLKDIV		High Speed Time-base Clock Prescale Bits. These bits determine part of the time-base clock prescale value: TBCLK = VCLK3 / (HSPCLKDIV × CLKDIV)
		0	/1
		1h	/2 (default on reset)
		2h	/4
		3h	/6
		4h	/8
		5h	/10
		6h	/12
		7h	/14

Table 35-24. Time-Base Control Register (TBCTL) Field Descriptions (continued)

Bit	Field	Value	Description
6	SWFSYNC		Software Forced Synchronization Pulse.
		0	Writing a 0 has no effect and reads always return a 0.
		1	Writing a 1 forces a one-time synchronization pulse to be generated.
			This event is ORed with the EPWMxSYNCI input of the ePWM module.
			SWFSYNC is valid (operates) only when EPWMxSYNCI is selected by SYNCOSEL = 00.
5-4	SYNCOSEL		Synchronization Output Select. These bits select the source of the EPWMxSYNCO signal.
		0	EPWMxSYNC
		1h	CTR = zero: Time-base counter equal to zero (TBCTR = 0x0000)
		2h	CTR = CMPB : Time-base counter equal to counter-compare B (TBCTR = CMPB)
		3h	Disable EPWMxSYNCO signal
3	PRDLD		Active Period Register Load From Shadow Register Select.
		0	The period register (TBPRD) is loaded from its shadow register when the time-base counter, TBCTR, is equal to zero.
			A write or read to the TBPRD register accesses the shadow register.
		1	Load the TBPRD register immediately without using a shadow register.
			A write or read to the TBPRD register directly accesses the active register.
2	PHSEN		Counter Register Load From Phase Register Enable.
		0	Do not load the time-base counter (TBCTR) from the time-base phase register (TBPHS).
		1	Load the time-base counter with the phase register when an EPWMxSYNCI input signal occurs or when a software synchronization is forced by the SWFSYNC bit, or when a digital compare sync event occurs.
1-0	CTRMODE		Counter Mode.
			The time-base counter mode is normally configured once and not changed during normal operation. If you change the mode of the counter, the change will take effect at the next TBCLK edge and the current counter value shall increment or decrement from the value before the mode change.
			These bits set the time-base counter mode of operation as follows:
		0	Up-count mode
		1h	Down-count mode
		2h	Up-down-count mode
		3h	Stop-freeze counter operation (default on reset)

35.4.1.3 Time-Base Phase Register (TBPHS)

Figure 35-65. Time-Base Phase Register (TBPHS) [offset = 04h]

15 … 0
TBPHS
R/W-0

LEGEND: R/W = Read/Write; -*n* = value after reset

Table 35-25. Time-Base Phase Register (TBPHS) Field Descriptions

Bits	Name	Description
15-0	TBPHS	These bits set time-base counter phase of the selected ePWM relative to the time-base that is supplying the synchronization input signal. • If TBCTL[PHSEN] = 0, then the synchronization event is ignored and the time-base counter is not loaded with the phase. • If TBCTL[PHSEN] = 1, then the time-base counter (TBCTR) will be loaded with the phase (TBPHS) when a synchronization event occurs. The synchronization event can be initiated by the input synchronization signal (EPWMxSYNCI) or by a software forced synchronization.

35.4.1.4 Time-Base Period Register (TBPRD)

Figure 35-66. Time-Base Period Register (TBPRD) [offset = 08h]

15 … 0
TBPRD
R/W-0

LEGEND: R/W = Read/Write; -*n* = value after reset

Table 35-26. Time-Base Period Register (TBPRD) Field Descriptions

Bits	Name	Description
15-0	TBPRD	These bits determine the period of the time-base counter. This sets the PWM frequency. Shadowing of this register is enabled and disabled by the TBCTL[PRDLD] bit. By default this register is shadowed. • If TBCTL[PRDLD] = 0, then the shadow is enabled and any write or read will automatically go to the shadow register. In this case, the active register will be loaded from the shadow register when the time-base counter equals 0. • If TBCTL[PRDLD] = 1, then the shadow is disabled and any write or read will go directly to the active register, that is the register actively controlling the hardware. • The active and shadow registers share the same memory map address.

35.4.1.5 Time-Base Counter Register (TBCTR)

Figure 35-67. Time-Base Counter Register (TBCTR) [offset = 0Ah]

15 … 0
TBCTR
R/W-0

LEGEND: R/W = Read/Write; -*n* = value after reset

Table 35-27. Time-Base Counter Register (TBCTR) Field Descriptions

Bits	Name	Description
15-0	TBCTR	Reading these bits gives the current time-base counter value. Writing to these bits sets the current time-base counter value. The update happens as soon as the write occurs; the write is NOT synchronized to the time-base clock (TBCLK) and the register is not shadowed.

35.4.2 Counter-Compare Submodule Registers

35.4.2.1 Counter-Compare Control Register (CMPCTL)

Figure 35-68. Counter-Compare Control Register (CMPCTL) [offset = 0Ch]

15	10	9	8
Reserved		SHDWBFULL	SHDWAFULL
R-0		R-0	R-0

7	6	5	4	3	2	1	0
Reserved	SHDWBMODE	Reserved	SHDWAMODE	LOADBMODE		LOADAMODE	
R-0	R/W-0	R-0	R/W-0	R/W-0		R/W-0	

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 35-28. Counter-Compare Control Register (CMPCTL) Field Descriptions

Bits	Name	Value	Description
15-10	Reserved	0	Reserved
9	SHDWBFULL		Counter-compare B (CMPB) Shadow Register Full Status Flag.
			This bit self clears once a load-strobe occurs.
		0	CMPB shadow FIFO not full yet.
		1	Indicates the CMPB shadow FIFO is full; a CPU write will overwrite current shadow value.
8	SHDWAFULL		Counter-compare A (CMPA) Shadow Register Full Status Flag.
			The flag bit is set when a 32-bit write to CMPA:CMPAHR register or a 16-bit write to CMPA register is made. A 16-bit write to CMPAHR register will not affect the flag.
			This bit self clears once a load-strobe occurs.
		0	CMPA shadow FIFO not full yet.
		1	Indicates the CMPA shadow FIFO is full, a CPU write will overwrite the current shadow value.
7	Reserved	0	Reserved
6	SHDWBMODE		Counter-compare B (CMPB) Register Operating Mode.
		0	Shadow mode. Operates as a double buffer. All writes via the CPU access the shadow register.
		1	Immediate mode. Only the active compare B register is used. All writes and reads directly access the active register for immediate compare action.
5	Reserved	0	Reserved
4	SHDWAMODE		Counter-compare A (CMPA) Register Operating Mode.
		0	Shadow mode. Operates as a double buffer. All writes via the CPU access the shadow register.
		1	Immediate mode. Only the active compare register is used. All writes and reads directly access the active register for immediate compare action.
3-2	LOADBMODE		Active Counter-Compare B (CMPB) Load From Shadow Select Mode. This bit has no effect in immediate mode (CMPCTL[SHDWBMODE] = 1).
		0	Load on CTR = Zero: Time-base counter equal to zero (TBCTR = 0x0000)
		1h	Load on CTR = PRD: Time-base counter equal to period (TBCTR = TBPRD)
		2h	Load on either CTR = Zero or CTR = PRD
		3h	Freeze (no loads possible)
1-0	LOADAMODE		Active Counter-Compare A (CMPA) Load From Shadow Select Mode. This bit has no effect in immediate mode (CMPCTL[SHDWAMODE] = 1).
		0	Load on CTR = Zero: Time-base counter equal to zero (TBCTR = 0x0000)
		1h	Load on CTR = PRD: Time-base counter equal to period (TBCTR = TBPRD)
		2h	Load on either CTR = Zero or CTR = PRD
		3h	Freeze (no loads possible)

35.4.2.2 Counter-Compare A Register (CMPA)

Figure 35-69. Counter-Compare A Register (CMPA) [offset = 10h]

15	0
CMPA	
R/W-0	

LEGEND: R/W = Read/Write; -n = value after reset

Table 35-29. Counter-Compare A Register (CMPA) Field Descriptions

Bits	Name	Description
15-0	CMPA	The value in the active CMPA register is continuously compared to the time-base counter (TBCTR). When the values are equal, the counter-compare module generates a "time-base counter equal to counter compare A" event. This event is sent to the action-qualifier where it is qualified and converted it into one or more actions. These actions can be applied to either the EPWMxA or the EPWMxB output depending on the configuration of the AQCTLA and AQCTLB registers. The actions that can be defined in the AQCTLA and AQCTLB registers include: • Do nothing; the event is ignored. • Clear: Pull the EPWMxA and/or EPWMxB signal low. • Set: Pull the EPWMxA and/or EPWMxB signal high. • Toggle the EPWMxA and/or EPWMxB signal. Shadowing of this register is enabled and disabled by the CMPCTL[SHDWAMODE] bit. By default this register is shadowed. • If CMPCTL[SHDWAMODE] = 0, then the shadow is enabled and any write or read will automatically go to the shadow register. In this case, the CMPCTL[LOADAMODE] bit field determines which event will load the active register from the shadow register. • Before a write, the CMPCTL[SHDWAFULL] bit can be read to determine if the shadow register is currently full. • If CMPCTL[SHDWAMODE] = 1, then the shadow register is disabled and any write or read will go directly to the active register, that is the register actively controlling the hardware. • In either mode, the active and shadow registers share the same memory map address.

35.4.2.3 Counter-Compare B Register (CMPB)

Figure 35-70. Counter-Compare B Register (CMPB) [offset = 16h]

15	0
CMPB	
R/W-0	

LEGEND: R/W = Read/Write; -*n* = value after reset

Table 35-30. Counter-Compare B Register (CMPB) Field Descriptions

Bits	Name	Description
15-0	CMPB	The value in the active CMPB register is continuously compared to the time-base counter (TBCTR). When the values are equal, the counter-compare module generates a "time-base counter equal to counter compare B" event. This event is sent to the action-qualifier where it is qualified and converted it into one or more actions. These actions can be applied to either the EPWMxA or the EPWMxB output depending on the configuration of the AQCTLA and AQCTLB registers. The actions that can be defined in the AQCTLA and AQCTLB registers include: • Do nothing. event is ignored. • Clear: Pull the EPWMxA and/or EPWMxB signal low. • Set: Pull the EPWMxA and/or EPWMxB signal high. • Toggle the EPWMxA and/or EPWMxB signal. Shadowing of this register is enabled and disabled by the CMPCTL[SHDWBMODE] bit. By default this register is shadowed. • If CMPCTL[SHDWBMODE] = 0, then the shadow is enabled and any write or read will automatically go to the shadow register. In this case, the CMPCTL[LOADBMODE] bit field determines which event will load the active register from the shadow register: • Before a write, the CMPCTL[SHDWBFULL] bit can be read to determine if the shadow register is currently full. • If CMPCTL[SHDWBMODE] = 1, then the shadow register is disabled and any write or read will go directly to the active register, that is the register actively controlling the hardware. • In either mode, the active and shadow registers share the same memory map address.

35.4.3 Action-Qualifier Submodule Registers

35.4.3.1 Action-Qualifier Output A Control Register (AQCTLA)

Figure 35-71. Action-Qualifier Output A Control Register (AQCTLA) [offset = 14h]

15-12	11-10	9-8
Reserved	CBD	CBU
R-0	R/W-0	R/W-0

7-6	5-4	3-2	1-0
CAD	CAU	PRD	ZRO
R/W-0	R/W-0	R/W-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 35-31. Action-Qualifier Output A Control Register (AQCTLA) Field Descriptions

Bits	Name	Value	Description
15-12	Reserved	0	Reserved
11-10	CBD		Action when the time-base counter equals the active CMPB register and the counter is decrementing.
		0	Do nothing (action is disabled).
		1h	Clear: force EPWMxA output low.
		2h	Set: force EPWMxA output high.
		3h	Toggle EPWMxA output: low output signal will be forced high, and a high signal will be forced low.
9-8	CBU		Action when the counter equals the active CMPB register and the counter is incrementing.
		0	Do nothing (action is disabled).
		1h	Clear: force EPWMxA output low.
		2h	Set: force EPWMxA output high.
		3h	Toggle EPWMxA output: low output signal will be forced high, and a high signal will be forced low.
7-6	CAD		Action when the counter equals the active CMPA register and the counter is decrementing.
		0	Do nothing (action is disabled).
		1h	Clear: force EPWMxA output low.
		2h	Set: force EPWMxA output high.
		3h	Toggle EPWMxA output: low output signal will be forced high, and a high signal will be forced low.
5-4	CAU		Action when the counter equals the active CMPA register and the counter is incrementing.
		0	Do nothing (action is disabled).
		1h	Clear: force EPWMxA output low.
		2h	Set: force EPWMxA output high.
		3h	Toggle EPWMxA output: low output signal will be forced high, and a high signal will be forced low.
3-2	PRD		Action when the counter equals the period.
			Note: By definition, in count up-down mode when the counter equals period the direction is defined as 0 or counting down.
		0	Do nothing (action is disabled).
		1h	Clear: force EPWMxA output low.
		2h	Set: force EPWMxA output high.
		3h	Toggle EPWMxA output: low output signal will be forced high, and a high signal will be forced low.
1-0	ZRO		Action when counter equals zero.
			Note: By definition, in count up-down mode when the counter equals 0 the direction is defined as 1 or counting up.
		0	Do nothing (action is disabled).
		1h	Clear: force EPWMxA output low.
		2h	Set: force EPWMxA output high.
		3h	Toggle EPWMxA output: low output signal will be forced high, and a high signal will be forced low.

35.4.3.2 Action-Qualifier Software Force Register (AQSFRC)

Figure 35-72. Action-Qualifier Software Force Register (AQSFRC) [offset = 18h]

15	8
Reserved	
R-0	

7	6	5	4	3	2	1	0
RLDCSF		OTSFB	ACTSFB		OTSFA	ACTSFA	
R/W-0		R/W-0	R/W-0		R/W-0	R/W-0	

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 35-32. Action-Qualifier Software Force Register (AQSFRC) Field Descriptions

Bit	Field	Value	Description
15-8	Reserved	0	Reserved
7-6	RLDCSF		AQCSFRC Active Register Reload From Shadow Options.
		0	Load on event counter equals zero.
		1h	Load on event counter equals period.
		2h	Load on event counter equals zero or counter equals period.
		3h	Load immediately (the active register is directly accessed by the CPU and is not loaded from the shadow register).
5	OTSFB		One-Time Software Forced Event on Output B.
		0	Writing a 0 has no effect. Always reads back a 0.
			This bit is auto cleared once a write to this register is complete (that is, a forced event is initiated.)
			This is a one-shot forced event. It can be overridden by another subsequent event on output B.
		1	Initiates a single s/w forced event.
4-3	ACTSFB		Action when One-Time Software Force B Is invoked.
		0	Does nothing (action is disabled).
		1h	Clear (low)
		2h	Set (high)
		3h	Toggle (Low -> High, High -> Low)
			Note: This action is not qualified by counter direction (CNT_dir).
2	OTSFA		One-Time Software Forced Event on Output A.
		0	Writing a 0 has no effect. Always reads back a 0.
			This bit is auto cleared once a write to this register is complete (that is, a forced event is initiated).
		1	Initiates a single software forced event.
1-0	ACTSFA		Action When One-Time Software Force A Is Invoked.
		0	Does nothing (action is disabled).
		1h	Clear (low)
		2h	Set (high)
		3h	Toggle (Low → High, High → Low)
			Note: This action is not qualified by counter direction (CNT_dir).

35.4.3.3 Action-Qualifier Output B Control Register (AQCTLB)

Figure 35-73. Action-Qualifier Output B Control Register (AQCTLB) [offset = 1Ah]

15	12	11	10	9	8
Reserved		CBD		CBU	
R-0		R/W-0		R/W-0	

7	6	5	4	3	2	1	0
CAD		CAU		PRD		ZRO	
R/W-0		R/W-0		R/W-0		R/W-0	

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 35-33. Action-Qualifier Output B Control Register (AQCTLB) Field Descriptions

Bits	Name	Value	Description
15-12	Reserved	0	Reserved
11-10	CBD		Action when the counter equals the active CMPB register and the counter is decrementing.
		0	Do nothing (action is disabled).
		1h	Clear: force EPWMxB output low.
		2h	Set: force EPWMxB output high.
		3h	Toggle EPWMxB output: low output signal will be forced high, and a high signal will be forced low.
9-8	CBU		Action when the counter equals the active CMPB register and the counter is incrementing.
		0	Do nothing (action is disabled).
		1h	Clear: force EPWMxB output low.
		2h	Set: force EPWMxB output high.
		3h	Toggle EPWMxB output: low output signal will be forced high, and a high signal will be forced low.
7-6	CAD		Action when the counter equals the active CMPA register and the counter is decrementing.
		0	Do nothing (action is disabled).
		1h	Clear: force EPWMxB output low.
		2h	Set: force EPWMxB output high.
		3h	Toggle EPWMxB output: low output signal will be forced high, and a high signal will be forced low.
5-4	CAU		Action when the counter equals the active CMPA register and the counter is incrementing.
		0	Do nothing (action is disabled).
		1h	Clear: force EPWMxB output low.
		2h	Set: force EPWMxB output high.
		3h	Toggle EPWMxB output: low output signal will be forced high, and a high signal will be forced low.
3-2	PRD		Action when the counter equals the period. Note: By definition, in count up-down mode when the counter equals period the direction is defined as 0 or counting down.
		0	Do nothing (action is disabled).
		1h	Clear: force EPWMxB output low.
		2h	Set: force EPWMxB output high.
		3h	Toggle EPWMxB output: low output signal will be forced high, and a high signal will be forced low.
1-0	ZRO		Action when counter equals zero. Note: By definition, in count up-down mode when the counter equals 0 the direction is defined as 1 or counting up.
		0	Do nothing (action is disabled).
		1h	Clear: force EPWMxB output low.
		2h	Set: force EPWMxB output high.
		3h	Toggle EPWMxB output: low output signal will be forced high, and a high signal will be forced low.

35.4.3.4 Action-Qualifier Continuous Force Register (AQCSFRC)

Figure 35-74. Action-Qualifier Continuous Software Force Register (AQCSFRC) [offset = 1Eh]

15 – 8
Reserved
R-0

7 – 4	3 – 2	1 – 0
Reserved	CSFB	CSFA
R-0	R/W-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 35-34. Action-qualifier Continuous Software Force Register (AQCSFRC) Field Descriptions

Bits	Name	Value	Description
15-4	Reserved	0	Reserved
3-2	CSFB		Continuous Software Force on Output B. In immediate mode, a continuous force takes effect on the next TBCLK edge. In shadow mode, a continuous force takes effect on the next TBCLK edge after a shadow load into the active register. To configure shadow mode, use AQSFRC[RLDCSF].
		0	Forcing disabled, that is, has no effect.
		1h	Forces a continuous low on output B.
		2h	Forces a continuous high on output.
		3h	Software forcing is disabled and has no effect.
1-0	CSFA		Continuous Software Force on Output A. In immediate mode, a continuous force takes effect on the next TBCLK edge. In shadow mode, a continuous force takes effect on the next TBCLK edge after a shadow load into the active register.
		0	Forcing disabled, that is, has no effect.
		1h	Forces a continuous low on output A.
		2h	Forces a continuous high on output A.
		3h	Software forcing is disabled and has no effect.

35.4.4 Dead-Band Submodule Registers

35.4.4.1 Dead-Band Generator Control Register (DBCTL)

Figure 35-75. Dead-Band Generator Control Register (DBCTL) [offset = 1Ch]

15	14	8
HALFCYCLE	Reserved	
R/W-0	R-0	

7	6	5	4	3	2	1	0
Reserved		IN_MODE		POLSEL		OUT_MODE	
R-0		R/W-0		R/W-0		R/W-0	

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 35-35. Dead-Band Generator Control Register (DBCTL) Field Descriptions

Bits	Name	Value	Description
15	HALFCYCLE		Half Cycle Clocking Enable Bit.
		0	Full cycle clocking enabled. The dead-band counters are clocked at the TBCLK rate.
		1	Half cycle clocking enabled. The dead-band counters are clocked at TBCLK × 2.
14-6	Reserved	0	Reserved
5-4	IN_MODE		Dead Band Input Mode Control. Bit 5 controls the S5 switch and bit 4 controls the S4 switch shown in Figure 35-28. This allows you to select the input source to the falling-edge and rising-edge delay. To produce classical dead-band waveforms the default is EPWMxA In is the source for both falling and rising-edge delays.
		0	EPWMxA In (from the action-qualifier) is the source for both falling-edge and rising-edge delay.
		1h	EPWMxB In (from the action-qualifier) is the source for rising-edge delayed signal. EPWMxA In (from the action-qualifier) is the source for falling-edge delayed signal.
		2h	EPWMxA In (from the action-qualifier) is the source for rising-edge delayed signal. EPWMxB In (from the action-qualifier) is the source for falling-edge delayed signal.
		3h	EPWMxB In (from the action-qualifier) is the source for both rising-edge delay and falling-edge delayed signal.
3-2	POLSEL		Polarity Select Control. Bit 3 controls the S3 switch and bit 2 controls the S2 switch shown in Figure 35-28. This allows you to selectively invert one of the delayed signals before it is sent out of the dead-band submodule. The following descriptions correspond to classical upper/lower switch control as found in one leg of a digital motor control inverter. These assume that DBCTL[OUT_MODE] = 1,1 and DBCTL[IN_MODE] = 0,0. Other enhanced modes are also possible, but not regarded as typical usage modes.
		0	Active high (AH) mode. Neither EPWMxA nor EPWMxB is inverted (default).
		1h	Active low complementary (ALC) mode. EPWMxA is inverted.
		2h	Active high complementary (AHC). EPWMxB is inverted.
		3h	Active low (AL) mode. Both EPWMxA and EPWMxB are inverted.

Table 35-35. Dead-Band Generator Control Register (DBCTL) Field Descriptions (continued)

Bits	Name	Value	Description
1-0	OUT_MODE		Dead-band Output Mode Control. Bit 1 controls the S1 switch and bit 0 controls the S0 switch shown in Figure 35-28. This allows you to selectively enable or bypass the dead-band generation for the falling-edge and rising-edge delay.
		0	Dead-band generation is bypassed for both output signals. In this mode, both the EPWMxA and EPWMxB output signals from the action-qualifier are passed directly to the PWM-chopper submodule. In this mode, the POLSEL and IN_MODE bits have no effect.
		1h	Disable rising-edge delay. The EPWMxA signal from the action-qualifier is passed straight through to the EPWMxA input of the PWM-chopper submodule. The falling-edge delayed signal is seen on output EPWMxB. The input signal for the delay is determined by DBCTL[IN_MODE].
		2h	The rising-edge delayed signal is seen on output EPWMxA. The input signal for the delay is determined by DBCTL[IN_MODE]. Disable falling-edge delay. The EPWMxB signal from the action-qualifier is passed straight through to the EPWMxB input of the PWM-chopper submodule.
		3h	Dead-band is fully enabled for both rising-edge delay on output EPWMxA and falling-edge delay on output EPWMxB. The input signal for the delay is determined by DBCTL[IN_MODE].

35.4.4.2 Dead-Band Generator Falling Edge Delay Register (DBFED)

Figure 35-76. Dead-Band Generator Falling Edge Delay Register (DBFED) [offset = 20h]

15 – 10	9 – 8
Reserved	DEL
R-0	R/W-0

7 – 0
DEL
R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 35-36. Dead-Band Generator Falling Edge Delay Register (DBFED) Field Descriptions

Bits	Name	Description
15-10	Reserved	Reserved
9-0	DEL	Falling Edge Delay Count. 10-bit counter.

35.4.4.3 Dead-Band Generator Rising Edge Delay Register (DBRED)

Figure 35-77. Dead-Band Generator Rising Edge Delay Register (DBRED) [offset = 22h]

15 – 10	9 – 8
Reserved	DEL
R-0	R/W-0

7 – 0
DEL
R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 35-37. Dead-Band Generator Rising Edge Delay Register (DBRED) Field Descriptions

Bits	Name	Description
15-10	Reserved	Reserved
9-0	DEL	Rising Edge Delay Count. 10-bit counter.

35.4.5 Trip-Zone Submodule Registers

35.4.5.1 Trip-Zone Digital Compare Event Select Register (TZDCSEL)

Figure 35-78. Trip Zone Digital Compare Event Select Register (TZDCSEL) [offset = 24h]

15 12	11 9	8 6	5 3	2 0
Reserved	DCBEVT2	DCBEVT1	DCAEVT2	DCAEVT1
R-0	R/W-0	R/W-0	R/W-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 35-38. Trip Zone Digital Compare Event Select Register (TZDCSEL) Field Descriptions

Bit	Field	Value	Description
15-12	Reserved	0	Reserved
11-9	DCBEVT2		Digital Compare Output B Event 2 Selection.
		0	Event is disabled.
		1h	DCBH = low, DCBL = don't care
		2h	DCBH = high, DCBL = don't care
		3h	DCBL = low, DCBH = don't care
		4h	DCBL = high, DCBH = don't care
		5h	DCBL = high, DCBH = low
		6h-7h	Reserved
8-6	DCBEVT1		Digital Compare Output B Event 1 Selection.
		0	Event is disabled.
		1h	DCBH = low, DCBL = don't care
		2h	DCBH = high, DCBL = don't care
		3h	DCBL = low, DCBH = don't care
		4h	DCBL = high, DCBH = don't care
		5h	DCBL = high, DCBH = low
		6h-7h	Reserved
5-3	DCAEVT2		Digital Compare Output A Event 2 Selection.
		0	Event is disabled.
		1h	DCAH = low, DCAL = don't care
		2h	DCAH = high, DCAL = don't care
		3h	DCAL = low, DCAH = don't care
		4h	DCAL = high, DCAH = don't care
		5h	DCAL = high, DCAH = low
		6h-7h	Reserved
2-0	DCAEVT1		Digital Compare Output A Event 1 Selection.
		0	Event is disabled.
		1h	DCAH = low, DCAL = don't care
		2h	DCAH = high, DCAL = don't care
		3h	DCAL = low, DCAH = don't care
		4h	DCAL = high, DCAH = don't care
		5h	DCAL = high, DCAH = low
		6h-7h	Reserved

35.4.5.2 Trip-Zone Select Register (TZSEL)

Figure 35-79. Trip-Zone Select Register (TZSEL) [offset = 26h]

15	14	13	12	11	10	9	8
DCBEVT1	DCAEVT1	OSHT6	OSHT5	OSHT4	OSHT3	OSHT2	OSHT1
R-0	R-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

7	6	5	4	3	2	1	0
DCBEVT2	DCAEVT2	CBC6	CBC5	CBC4	CBC3	CBC2	CBC1
R-0	R-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 35-39. Trip-Zone Submodule Select Register (TZSEL) Field Descriptions

Bits	Name	Value	Description
One-Shot (OSHT) Trip-zone enable/disable. When any of the enabled pins go low, a one-shot trip event occurs for this ePWM module. When the event occurs, the action defined in the TZCTL register is taken on the EPWMxA and EPWMxB outputs. The one-shot trip condition remains latched until the user clears the condition via the TZCLR register.			
15	DCBEVT1		Digital Compare Output B Event 1 Select.
		0	Disable DCBEVT1 as one-shot-trip source for this ePWM module.
		1	Enable DCBEVT1 as one-shot-trip source for this ePWM module.
14	DCAEVT1		Digital Compare Output A Event 1 Select.
		0	Disable DCAEVT1 as one-shot-trip source for this ePWM module.
		1	Enable DCAEVT1 as one-shot-trip source for this ePWM module.
13	OSHT6		Trip-zone 6 ($\overline{\text{TZ6}}$) Select.
		0	Disable $\overline{\text{TZ6}}$ as a one-shot trip source for this ePWM module.
		1	Enable $\overline{\text{TZ6}}$ as a one-shot trip source for this ePWM module.
12	OSHT5		Trip-zone 5 ($\overline{\text{TZ5}}$) Select.
		0	Disable $\overline{\text{TZ5}}$ as a one-shot trip source for this ePWM module.
		1	Enable $\overline{\text{TZ5}}$ as a one-shot trip source for this ePWM module.
11	OSHT4		Trip-zone 4 ($\overline{\text{TZ4}}$) Select.
		0	Disable $\overline{\text{TZ4}}$ as a one-shot trip source for this ePWM module.
		1	Enable $\overline{\text{TZ4}}$ as a one-shot trip source for this ePWM module.
10	OSHT3		Trip-zone 3 ($\overline{\text{TZ3}}$) Select.
		0	Disable $\overline{\text{TZ3}}$ as a one-shot trip source for this ePWM module.
		1	Enable $\overline{\text{TZ3}}$ as a one-shot trip source for this ePWM module.
9	OSHT2		Trip-zone 2 ($\overline{\text{TZ2}}$) Select.
		0	Disable $\overline{\text{TZ2}}$ as a one-shot trip source for this ePWM module.
		1	Enable $\overline{\text{TZ2}}$ as a one-shot trip source for this ePWM module.
8	OSHT1		Trip-zone 1 ($\overline{\text{TZ1}}$) Select.
		0	Disable $\overline{\text{TZ1}}$ as a one-shot trip source for this ePWM module.
		1	Enable $\overline{\text{TZ1}}$ as a one-shot trip source for this ePWM module.
Cycle-by-Cycle (CBC) Trip-zone enable/disable. When any of the enabled pins go low, a cycle-by-cycle trip event occurs for this ePWM module. When the event occurs, the action defined in the TZCTL register is taken on the EPWMxA and EPWMxB outputs. A cycle-by-cycle trip condition is automatically cleared when the time-base counter reaches zero.			
7	DCBEVT2		Digital Compare Output B Event 2 Select.
		0	Disable DCBEVT2 as a CBC trip source for this ePWM module.
		1	Enable DCBEVT2 as a CBC trip source for this ePWM module.
6	DCAEVT2		Digital Compare Output A Event 2 Select.
		0	Disable DCAEVT2 as a CBC trip source for this ePWM module.
		1	Enable DCAEVT2 as a CBC trip source for this ePWM module.

Table 35-39. Trip-Zone Submodule Select Register (TZSEL) Field Descriptions (continued)

Bits	Name	Value	Description
5	CBC6		Trip-zone 6 ($\overline{\text{TZ6}}$) Select.
		0	Disable $\overline{\text{TZ6}}$ as a CBC trip source for this ePWM module.
		1	Enable $\overline{\text{TZ6}}$ as a CBC trip source for this ePWM module.
4	CBC5		Trip-zone 5 ($\overline{\text{TZ5}}$) Select.
		0	Disable $\overline{\text{TZ5}}$ as a CBC trip source for this ePWM module.
		1	Enable $\overline{\text{TZ5}}$ as a CBC trip source for this ePWM module.
3	CBC4		Trip-zone 4 ($\overline{\text{TZ4}}$) Select.
		0	Disable $\overline{\text{TZ4}}$ as a CBC trip source for this ePWM module.
		1	Enable $\overline{\text{TZ4}}$ as a CBC trip source for this ePWM module.
2	CBC3		Trip-zone 3 ($\overline{\text{TZ3}}$) Select.
		0	Disable $\overline{\text{TZ3}}$ as a CBC trip source for this ePWM module.
		1	Enable $\overline{\text{TZ3}}$ as a CBC trip source for this ePWM module.
1	CBC2		Trip-zone 2 ($\overline{\text{TZ2}}$) Select.
		0	Disable $\overline{\text{TZ2}}$ as a CBC trip source for this ePWM module.
		1	Enable $\overline{\text{TZ2}}$ as a CBC trip source for this ePWM module.
0	CBC1		Trip-zone 1 ($\overline{\text{TZ1}}$) Select.
		0	Disable $\overline{\text{TZ1}}$ as a CBC trip source for this ePWM module.
		1	Enable $\overline{\text{TZ1}}$ as a CBC trip source for this ePWM module.

35.4.5.3 Trip-Zone Enable Interrupt Register (TZEINT)

Figure 35-80. Trip-Zone Enable Interrupt Register (TZEINT) [offset = 28h]

15 … 8
Reserved
R -0

7	6	5	4	3	2	1	0
Reserved	DCBEVT2	DCBEVT1	DCAEVT2	DCAEVT1	OST	CBC	Reserved
R-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 35-40. Trip-Zone Enable Interrupt Register (TZEINT) Field Descriptions

Bits	Name	Value	Description
15-3	Reserved	0	Reserved
6	DCBEVT2		Digital Comparator Output B Event 2 Interrupt Enable.
		0	Disabled
		1	Enabled
5	DCBEVT1		Digital Comparator Output B Event 1 Interrupt Enable.
		0	Disabled
		1	Enabled
4	DCAEVT2		Digital Comparator Output A Event 2 Interrupt Enable.
		0	Disabled
		1	Enabled
3	DCAEVT1		Digital Comparator Output A Event 1 Interrupt Enable.
		0	Disabled
		1	Enabled
2	OST		Trip-zone One-Shot Interrupt Enable.
		0	Disable one-shot interrupt generation.
		1	Enable Interrupt generation; a one-shot trip event will cause a EPWMx_TZINT VIM interrupt.
1	CBC		Trip-zone Cycle-by-Cycle Interrupt Enable.
		0	Disable cycle-by-cycle interrupt generation.
		1	Enable interrupt generation; a cycle-by-cycle trip event will cause an EPWMx_TZINT VIM interrupt.
0	Reserved	0	Reserved

35.4.5.4 Trip-Zone Control Register (TZCTL)

Figure 35-81. Trip-Zone Control Register (TZCTL) [offset = 2Ah]

15	14	13	12	11	10	9	8
Reserved				DCBEVT2		DCBEVT1	
R-0				R/W-0		R/W-0	

7	6	5	4	3	2	1	0
DCAEVT2		DCAEVT1		TZB		TZA	
R/W-0		R/W-0		R/W-0		R/W-0	

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 35-41. Trip-Zone Control Register (TZCTL) Field Descriptions

Bit	Field	Value	Description
15-12	Reserved	0	Reserved
11-10	DCBEVT2		Digital Compare Output B Event 2 Action On EPWMxB.
		0	High-impedance (EPWMxB = High-impedance state).
		1h	Force EPWMxB to a high state.
		2h	Force EPWMxB to a low state.
		3h	Do Nothing, trip action is disabled.
9-8	DCBEVT1		Digital Compare Output B Event 1 Action On EPWMxB.
		0	High-impedance (EPWMxB = High-impedance state).
		1h	Force EPWMxB to a high state.
		2h	Force EPWMxB to a low state.
		3h	Do Nothing, trip action is disabled.
7-6	DCAEVT2		Digital Compare Output A Event 2 Action On EPWMxA.
		0	High-impedance (EPWMxA = High-impedance state).
		1h	Force EPWMxA to a high state.
		2h	Force EPWMxA to a low state.
		3h	Do Nothing, trip action is disabled.
5-4	DCAEVT1		Digital Compare Output A Event 1 Action On EPWMxA.
		0	High-impedance (EPWMxA = High-impedance state).
		1h	Force EPWMxA to a high state.
		2h	Force EPWMxA to a low state.
		3h	Do Nothing, trip action is disabled.
3-2	TZB		When a trip event occurs the following action is taken on output EPWMxB. Which trip-zone pins can cause an event is defined in the TZSEL register.
		0	High-impedance (EPWMxB = High-impedance state).
		1h	Force EPWMxB to a high state.
		2h	Force EPWMxB to a low state.
		3h	Do nothing, no action is taken on EPWMxB.
1-0	TZA		When a trip event occurs the following action is taken on output EPWMxA. Which trip-zone pins can cause an event is defined in the TZSEL register.
		0	High-impedance (EPWMxA = High-impedance state).
		1h	Force EPWMxA to a high state.
		2h	Force EPWMxA to a low state.
		3h	Do nothing, no action is taken on EPWMxA.

35.4.5.5 Trip-Zone Clear Register (TZCLR)

Figure 35-82. Trip-Zone Clear Register (TZCLR) [offset = 2Ch]

15–8
Reserved
R-0

7	6	5	4	3	2	1	0
Reserved	DCBEVT2	DCBEVT1	DCAEVT2	DCAEVT1	OST	CBC	INT
R-0	R/W1C-0	R/W1C-0	R/W1C-0	R/W1C-0	R/W-0	R/W-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; W1C = Write 1 to clear; -*n* = value after reset

Table 35-42. Trip-Zone Clear Register (TZCLR) Field Descriptions

Bit	Field	Value	Description
15-7	Reserved	0	Reserved
6	DCBEVT2		Clear Flag for Digital Compare Output B Event 2.
		0	Writing 0 has no effect. This bit always reads back 0.
		1	Writing 1 clears the DCBEVT2 event trip condition.
5	DCBEVT1		Clear Flag for Digital Compare Output B Event 1.
		0	Writing 0 has no effect. This bit always reads back 0.
		1	Writing 1 clears the DCBEVT1 event trip condition.
4	DCAEVT2		Clear Flag for Digital Compare Output A Event 2.
		0	Writing 0 has no effect. This bit always reads back 0.
		1	Writing 1 clears the DCAEVT2 event trip condition.
3	DCAEVT1		Clear Flag for Digital Compare Output A Event 1.
		0	Writing 0 has no effect. This bit always reads back 0.
		1	Writing 1 clears the DCAEVT1 event trip condition.
2	OST		Clear Flag for One-Shot Trip (OST) Latch.
		0	Has no effect. Always reads back a 0.
		1	Clears this Trip (set) condition.
1	CBC		Clear Flag for Cycle-By-Cycle (CBC) Trip Latch.
		0	Has no effect. Always reads back a 0.
		1	Clears this Trip (set) condition.
0	INT		Global Interrupt Clear Flag.
		0	Has no effect. Always reads back a 0.
		1	Clears the trip-interrupt flag for this ePWM module (TZFLG[INT]).
			NOTE: No further EPWMx_TZINT VIM interrupts will be generated until the flag is cleared. If the TZFLG[INT] bit is cleared and any of the other flag bits are set, then another interrupt pulse will be generated. Clearing all flag bits will prevent further interrupts.

35.4.5.6 Trip-Zone Flag Register (TZFLG)

Figure 35-83. Trip-Zone Flag Register (TZFLG) [offset = 2Eh]

15 … 8
Reserved
R-0

7	6	5	4	3	2	1	0
Reserved	DCBEVT2	DCBEVT1	DCAEVT2	DCAEVT1	OST	CBC	INT
R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0

LEGEND: R = Read only; -n = value after reset

Table 35-43. Trip-Zone Flag Register (TZFLG) Field Descriptions

Bit	Field	Value	Description
15-7	Reserved	0	Reserved
6	DCBEVT2		Latched Status Flag for Digital Compare Output B Event 2.
		0	No trip event has occurred on DCBEVT2.
		1	A trip event has occurred for the event defined for DCBEVT2.
5	DCBEVT1		Latched Status Flag for Digital Compare Output B Event 1.
		0	No trip event has occurred on DCBEVT1.
		1	A trip event has occurred for the event defined for DCBEVT1.
4	DCAEVT2		Latched Status Flag for Digital Compare Output A Event 2.
		0	No trip event has occurred on DCAEVT2.
		1	A trip event has occurred for the event defined for DCAEVT2.
3	DCAEVT1		Latched Status Flag for Digital Compare Output A Event 1.
		0	No trip event has occurred on DCAEVT1.
		1	A trip event has occurred for the event defined for DCAEVT1.
2	OST		Latched Status Flag for A One-Shot Trip Event.
		0	No one-shot trip event has occurred.
		1	A trip event has occurred on a pin selected as a one-shot trip source.
			This bit is cleared by writing the appropriate value to the TZCLR register.
1	CBC		Latched Status Flag for Cycle-By-Cycle Trip Event.
		0	No cycle-by-cycle trip event has occurred.
		1	A trip event has occurred on a signal selected as a cycle-by-cycle trip source. The TZFLG[CBC] bit will remain set until it is manually cleared by the user. If the cycle-by-cycle trip event is still present when the CBC bit is cleared, then CBC will be immediately set again. The specified condition on the signal is automatically cleared when the ePWM time-base counter reaches zero (TBCTR = 0x0000) if the trip condition is no longer present. The condition on the signal is only cleared when the TBCTR = 0x0000 no matter where in the cycle the CBC flag is cleared.
			This bit is cleared by writing the appropriate value to the TZCLR register.
0	INT		Latched Trip Interrupt Status Flag.
		0	No interrupt has been generated.
		1	An EPWMx_TZINT VIM interrupt was generated because of a trip condition.
			No further EPWMx_TZINT VIM interrupts will be generated until this flag is cleared. If the interrupt flag is cleared when either CBC or OST is set, then another interrupt pulse will be generated. Clearing all flag bits will prevent further interrupts.
			This bit is cleared by writing the appropriate value to the TZCLR register.

35.4.5.7 Trip-Zone Force Register (TZFRC)

Figure 35-84. Trip-Zone Force Register (TZFRC) [offset = 32h]

15	8
Reserved	
R-0	

7	6	5	4	3	2	1	0
Reserved	DCBEVT2	DCBEVT1	DCAEVT2	DCAEVT1	OST	CBC	Reserved
R-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R- 0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 35-44. Trip-Zone Force Register (TZFRC) Field Descriptions

Bits	Name	Value	Description
15-7	Reserved	0	Reserved
6	DCBEVT2		Force Flag for Digital Compare Output B Event 2.
		0	Writing 0 has no effect. This bit always reads back 0.
		1	Writing 1 forces the DCBEVT2 event trip condition and sets the TZFLG[DCBEVT2] bit.
5	DCBEVT1		Force Flag for Digital Compare Output B Event 1.
		0	Writing 0 has no effect. This bit always reads back 0.
		1	Writing 1 forces the DCBEVT1 event trip condition and sets the TZFLG[DCBEVT1] bit.
4	DCAEVT2		Force Flag for Digital Compare Output A Event 2.
		0	Writing 0 has no effect. This bit always reads back 0.
		1	Writing 1 forces the DCAEVT2 event trip condition and sets the TZFLG[DCAEVT2] bit.
3	DCAEVT1		Force Flag for Digital Compare Output A Event 1.
		0	Writing 0 has no effect. This bit always reads back 0
		1	Writing 1 forces the DCAEVT1 event trip condition and sets the TZFLG[DCAEVT1] bit.
2	OST		Force a One-Shot Trip Event via Software.
		0	Writing of 0 is ignored. Always reads back a 0.
		1	Forces a one-shot trip event and sets the TZFLG[OST] bit.
1	CBC		Force a Cycle-by-Cycle Trip Event via Software.
		0	Writing of 0 is ignored. Always reads back a 0.
		1	Forces a cycle-by-cycle trip event and sets the TZFLG[CBC] bit.
0	Reserved	0	Reserved

35.4.6 Event-Trigger Submodule Registers

35.4.6.1 Event-Trigger Selection Register (ETSEL)

Figure 35-85. Event-Trigger Selection Register (ETSEL) [offset = 30h]

15	14	12	11	10	8
SOCBEN	SOCBSEL		SOCAEN	SOCASEL	
R/W-0	R/W-0		R/W-0	R/W-0	

7	4	3	2	0
Reserved		INTEN	INTSEL	
R-0		R/W-0	R/W-0	

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 35-45. Event-Trigger Selection Register (ETSEL) Field Descriptions

Bits	Name	Value	Description
15	SOCBEN		Enable the ADC Start of Conversion B (EPWMxSOCB) Pulse.
		0	Disable EPWMxSOCB.
		1	Enable EPWMxSOCB pulse.
14-12	SOCBSEL		EPWMxSOCB Selection Options.
			These bits determine when a EPWMxSOCB pulse will be generated.
		0	Enable DCBEVT1.soc event.
		1h	Enable event time-base counter equal to zero. (TBCTR = 0x0000).
		2h	Enable event time-base counter equal to period (TBCTR = TBPRD).
		3h	Enable event time-base counter equal to zero or period (TBCTR = 0x0000 or TBCTR = TBPRD). This mode is useful in up-down count mode.
		4h	Enable event time-base counter equal to CMPA when the timer is incrementing.
		5h	Enable event time-base counter equal to CMPA when the timer is decrementing.
		6h	Enable event: time-base counter equal to CMPB when the timer is incrementing.
		7h	Enable event: time-base counter equal to CMPB when the timer is decrementing.
11	SOCAEN		Enable the ADC Start of Conversion A (EPWMxSOCA) Pulse.
		0	Disable EPWMxSOCA.
		1	Enable EPWMxSOCA pulse.
10-8	SOCASEL		EPWMxSOCA Selection Options.
			These bits determine when a EPWMxSOCA pulse will be generated.
		0	Enable DCAEVT1.soc event.
		1h	Enable event time-base counter equal to zero. (TBCTR = 0x0000).
		2h	Enable event time-base counter equal to period (TBCTR = TBPRD).
		3h	Enable event time-base counter equal to zero or period (TBCTR = 0x0000 or TBCTR = TBPRD). This mode is useful in up-down count mode.
		4h	Enable event time-base counter equal to CMPA when the timer is incrementing.
		5h	Enable event time-base counter equal to CMPA when the timer is decrementing.
		6h	Enable event: time-base counter equal to CMPB when the timer is incrementing.
		7h	Enable event: time-base counter equal to CMPB when the timer is decrementing.
7-4	Reserved	0	Reserved
3	INTEN		Enable ePWM Interrupt (EPWMx_INT) Generation.
		0	Disable EPWMx_INT generation.
		1	Enable EPWMx_INT generation.

Table 35-45. Event-Trigger Selection Register (ETSEL) Field Descriptions (continued)

Bits	Name	Value	Description
2-0	INTSEL		ePWM Interrupt (EPWMx_INT) Selection Options.
		0	Reserved
		1h	Enable event time-base counter equal to zero. (TBCTR = 0x0000).
		2h	Enable event time-base counter equal to period (TBCTR = TBPRD).
		3h	Enable event time-base counter equal to zero or period (TBCTR = 0x0000 or TBCTR = TBPRD). This mode is useful in up-down count mode.
		4h	Enable event time-base counter equal to CMPA when the timer is incrementing.
		5h	Enable event time-base counter equal to CMPA when the timer is decrementing.
		6h	Enable event: time-base counter equal to CMPB when the timer is incrementing.
		7h	Enable event: time-base counter equal to CMPB when the timer is decrementing.

35.4.6.2 Event-Trigger Flag Register (ETFLG)

Figure 35-86. Event-Trigger Flag Register (ETFLG) [offset = 34h]

15	8
Reserved	
R-0	

7	4	3	2	1	0
Reserved		SOCB	SOCA	Reserved	INT
R-0		R-0	R-0	R-0	R-0

LEGEND: R = Read only; -*n* = value after reset

Table 35-46. Event-Trigger Flag Register (ETFLG) Field Descriptions

Bits	Name	Value	Description
15-4	Reserved	0	Reserved
3	SOCB		Latched ePWM ADC Start-of-Conversion B (EPWMxSOCB) Status Flag.
		0	No EPWMxSOCB event occurred.
		1	A start of conversion pulse was generated on EPWMxSOCB. The EPWMxSOCB output will continue to be generated even if the flag bit is set.
2	SOCA		Latched ePWM ADC Start-of-Conversion A (EPWMxSOCA) Status Flag. Unlike the ETFLG[INT] flag, the EPWMxSOCA output will continue to pulse even if the flag bit is set.
		0	No event occurred.
		1	A start of conversion pulse was generated on EPWMxSOCA. The EPWMxSOCA output will continue to be generated even if the flag bit is set.
1	Reserved	0	Reserved
0	INT		Latched ePWM Interrupt (EPWMx_INT) Status Flag.
		0	No event occurred.
		1	An ePWMx interrupt (EWPMx_INT) was generated. No further interrupts will be generated until the flag bit is cleared. Up to one interrupt can be pending while the ETFLG[INT] bit is still set. If an interrupt is pending, it will not be generated until after the ETFLG[INT] bit is cleared. Refer to Figure 35-41.

35.4.6.3 Event-Trigger Prescale Register (ETPS)

Figure 35-87. Event-Trigger Prescale Register (ETPS) [offset = 36h]

15	14	13	12	11	10	9	8
SOCBCNT		SOCBPRD		SOCACNT		SOCAPRD	
R-0		R/W-0		R-0		R/W-0	

7			4	3	2	1	0
Reserved				INTCNT		INTPRD	
R-0				R-0		R/W-0	

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 35-47. Event-Trigger Prescale Register (ETPS) Field Descriptions

Bits	Name	Value	Description
15-14	SOCBCNT		ePWM ADC Start-of-Conversion B Event (EPWMxSOCB) Counter Register.
			These bits indicate how many selected ETSEL[SOCBSEL] events have occurred.
		0	No events have occurred.
		1h	1 event has occurred.
		2h	2 events have occurred.
		3h	3 events have occurred.
13-12	SOCBPRD		ePWM ADC Start-of-Conversion B Event (EPWMxSOCB) Period Select.
			These bits determine how many selected ETSEL[SOCBSEL] events need to occur before an EPWMxSOCB pulse is generated. To be generated, the pulse must be enabled (ETSEL[SOCBEN] = 1). The SOCB pulse will be generated even if the status flag is set from a previous start of conversion (ETFLG[SOCB] = 1). Once the SOCB pulse is generated, the ETPS[SOCBCNT] bits will automatically be cleared.
		0	Disable the SOCB event counter. No EPWMxSOCB pulse will be generated.
		1h	Generate the EPWMxSOCB pulse on the first event: ETPS[SOCBCNT] = 0,1.
		2h	Generate the EPWMxSOCB pulse on the second event: ETPS[SOCBCNT] = 1,0.
		3h	Generate the EPWMxSOCB pulse on the third event: ETPS[SOCBCNT] = 1,1.
11-10	SOCACNT		ePWM ADC Start-of-Conversion A Event (EPWMxSOCA) Counter Register.
			These bits indicate how many selected ETSEL[SOCASEL] events have occurred.
		0	No events have occurred.
		1h	1 event has occurred.
		2h	2 events have occurred.
		3h	3 events have occurred.
9-8	SOCAPRD		ePWM ADC Start-of-Conversion A Event (EPWMxSOCA) Period Select.
			These bits determine how many selected ETSEL[SOCASEL] events need to occur before an EPWMxSOCA pulse is generated. To be generated, the pulse must be enabled (ETSEL[SOCAEN] = 1). The SOCA pulse will be generated even if the status flag is set from a previous start of conversion (ETFLG[SOCA] = 1). Once the SOCA pulse is generated, the ETPS[SOCACNT] bits will automatically be cleared.
		0	Disable the SOCA event counter. No EPWMxSOCA pulse will be generated.
		1h	Generate the EPWMxSOCA pulse on the first event: ETPS[SOCACNT] = 0,1.
		2h	Generate the EPWMxSOCA pulse on the second event: ETPS[SOCACNT] = 1,0.
		3h	Generate the EPWMxSOCA pulse on the third event: ETPS[SOCACNT] = 1,1.
7-4	Reserved	0	Reserved

Table 35-47. Event-Trigger Prescale Register (ETPS) Field Descriptions (continued)

Bits	Name	Value	Description
3-2	INTCNT		ePWM Interrupt Event (EPWMx_INT) Counter Register.
			These bits indicate how many selected ETSEL[INTSEL] events have occurred. These bits are automatically cleared when an interrupt pulse is generated. If interrupts are disabled, ETSEL[INT] = 0 or the interrupt flag is set, ETFLG[INT] = 1, the counter will stop counting events when it reaches the period value ETPS[INTCNT] = ETPS[INTPRD].
		0	No events have occurred.
		1h	1 event has occurred.
		2h	2 events have occurred.
		3h	3 events have occurred.
1-0	INTPRD		ePWM Interrupt (EPWMx_INT) Period Select.
			These bits determine how many selected ETSEL[INTSEL] events need to occur before an interrupt is generated. To be generated, the interrupt must be enabled (ETSEL[INT] = 1). If the interrupt status flag is set from a previous interrupt (ETFLG[INT] = 1) then no interrupt will be generated until the flag is cleared via the ETCLR[INT] bit. This allows for one interrupt to be pending while another is still being serviced. Once the interrupt is generated, the ETPS[INTCNT] bits will automatically be cleared.
			Writing a INTPRD value that is the same as the current counter value will trigger an interrupt if it is enabled and the status flag is clear.
			Writing a INTPRD value that is less than the current counter value will result in an undefined state.
			If a counter event occurs at the same instant as a new zero or non-zero INTPRD value is written, the counter is incremented.
		0	Disable the interrupt event counter. No interrupt will be generated and ETFRC[INT] is ignored.
		1h	Generate an interrupt on the first event INTCNT = 01 (first event).
		2h	Generate interrupt on ETPS[INTCNT] = 1,0 (second event).
		3h	Generate interrupt on ETPS[INTCNT] = 1,1 (third event).

35.4.6.4 Event-Trigger Force Register (ETFRC)

Figure 35-88. Event-Trigger Force Register (ETFRC) [offset = 38h]

15	8
Reserved	
R-0	

7	4	3	2	1	0
Reserved		SOCB	SOCA	Reserved	INT
R-0		R/W-0	R/W-0	R-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 35-48. Event-Trigger Force Register (ETFRC) Field Descriptions

Bits	Name	Value	Description
15-4	Reserved	0	Reserved
3	SOCB		SOCB Force Bit. The SOCB pulse will only be generated if the event is enabled in the ETSEL register. The ETFLG[SOCB] flag bit will be set regardless.
		0	Has no effect. Always reads back a 0.
		1	Generates a pulse on EPWMxSOCB and sets the SOCBFLG bit. This bit is used for test purposes.
2	SOCA		SOCA Force Bit. The SOCA pulse will only be generated if the event is enabled in the ETSEL register. The ETFLG[SOCA] flag bit will be set regardless.
		0	Writing 0 to this bit will be ignored. Always reads back a 0.
		1	Generates a pulse on EPWMxSOCA and set the SOCAFLG bit. This bit is used for test purposes.
1	Reserved	0	Reserved
0	INT		INT Force Bit. The interrupt will only be generated if the event is enabled in the ETSEL register. The INT flag bit will be set regardless.
		0	Writing 0 to this bit will be ignored. Always reads back a 0.
		1	Generates an interrupt on $\overline{\text{EPWMxINT}}$ and set the INT flag bit. This bit is used for test purposes.

35.4.6.5 Event-Trigger Clear Register (ETCLR)

Figure 35-89. Event-Trigger Clear Register (ETCLR) [offset = 3Ah]

15-8
Reserved
R-0

7-4	3	2	1	0
Reserved	SOCB	SOCA	Reserved	INT
R-0	R/W-0	R/W-0	R-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 35-49. Event-Trigger Clear Register (ETCLR) Field Descriptions

Bits	Name	Value	Description
15-4	Reserved	0	Reserved
3	SOCB		ePWM ADC Start-of-Conversion B (EPWMxSOCB) Flag Clear Bit.
		0	Writing a 0 has no effect. Always reads back a 0.
		1	Clears the ETFLG[SOCB] flag bit.
2	SOCA		ePWM ADC Start-of-Conversion A (EPWMxSOCA) Flag Clear Bit.
		0	Writing a 0 has no effect. Always reads back a 0.
		1	Clears the ETFLG[SOCA] flag bit.
1	Reserved	0	Reserved
0	INT		ePWM Interrupt (EPWMx_INT) Flag Clear Bit.
		0	Writing a 0 has no effect. Always reads back a 0.
		1	Clears the ETFLG[INT] flag bit and enable further interrupts pulses to be generated.

35.4.7 PWM-Chopper Submodule Register

35.4.7.1 PWM-Chopper Control Register (PCCTL)

Figure 35-90. PWM-Chopper Control Register (PCCTL) [offset = 3Eh]

15 … 11	10 … 8
Reserved	CHPDUTY
R-0	R/W-0

7 … 5	4 … 1	0
CHPFREQ	OSHTWTH	CHPEN
R/W-0	R/W-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 35-50. PWM-Chopper Control Register (PCCTL) Bit Descriptions

Bits	Name	Value	Description
15-11	Reserved	0	Reserved
10-8	CHPDUTY		Chopping Clock Duty Cycle.
		0	Duty = 1/8 (12.5%)
		1h	Duty = 2/8 (25.0%)
		2h	Duty = 3/8 (37.5%)
		3h	Duty = 4/8 (50.0%)
		4h	Duty = 5/8 (62.5%)
		5h	Duty = 6/8 (75.0%)
		6h	Duty = 7/8 (87.5%)
		7h	Reserved
7-5	CHPFREQ		Chopping Clock Frequency.
		0	Divide by 1 (no prescale, = 12.5 MHz at 100 MHz VCLK3)
		1h	Divide by 2 (6.25 MHz at 100 MHz VCLK3)
		2h	Divide by 3 (4.16 MHz at 100 MHz VCLK3)
		3h	Divide by 4 (3.12 MHz at 100 MHz VCLK3)
		4h	Divide by 5 (2.50 MHz at 100 MHz VCLK3)
		5h	Divide by 6 (2.08 MHz at 100 MHz VCLK3)
		6h	Divide by 7 (1.78 MHz at 100 MHz VCLK3)
		7h	Divide by 8 (1.56 MHz at 100 MHz VCLK3)

Table 35-50. PWM-Chopper Control Register (PCCTL) Bit Descriptions (continued)

Bits	Name	Value	Description
4-1	OSHTWTH		One-Shot Pulse Width.
		0	1 x VCLK3 / 8 wide (= 80 nS at 100 MHz VCLK3)
		1h	2 x VCLK3 / 8 wide (= 160 nS at 100 MHz VCLK3)
		2h	3 x VCLK3 / 8 wide (= 240 nS at 100 MHz VCLK3)
		3h	4 x VCLK3 / 8 wide (= 320 nS at 100 MHz VCLK3)
		4h	5 x VCLK3 / 8 wide (= 400 nS at 100 MHz VCLK3)
		5h	6 x VCLK3 / 8 wide (= 480 nS at 100 MHz VCLK3)
		6h	7 x VCLK3 / 8 wide (= 560 nS at 100 MHz VCLK3)
		7h	8 x VCLK3 / 8 wide (= 640 nS at 100 MHz VCLK3)
		8h	9 x VCLK3 / 8 wide (= 720 nS at 100 MHz VCLK3)
		9h	10 x VCLK3 / 8 wide (= 800 nS at 100 MHz VCLK3)
		Ah	11 x VCLK3 / 8 wide (= 880 nS at 100 MHz VCLK3)
		Bh	12 x VCLK3 / 8 wide (= 960 nS at 100 MHz VCLK3)
		Ch	13 x VCLK3 / 8 wide (= 1040 nS at 100 MHz VCLK3)
		Dh	14 x VCLK3 / 8 wide (= 1120 nS at 100 MHz VCLK3)
		Eh	15 x VCLK3 / 8 wide (= 1200 nS at 100 MHz VCLK3)
		Fh	16 x VCLK3 / 8 wide (= 1280 nS at 100 MHz VCLK3)
0	CHPEN		PWM-chopping Enable.
		0	Disable (bypass) PWM chopping function.
		1	Enable chopping function.

35.4.8 Digital Compare Submodule Registers

35.4.8.1 Digital Compare A Control Register (DCACTL)

Figure 35-91. Digital Compare A Control Register (DCACTL) [offset = 60h]

15	10	9	8
Reserved		EVT2FRC SYNCSEL	EVT2SRCSEL
R-0		R/W-0	R/W-0

7	4	3	2	1	0
Reserved		EVT1SYNCE	EVT1SOCE	EVT1FRC SYNCSEL	EVT1SRCSEL
R-0		R/W-0	R/W-0	R/W-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 35-51. Digital Compare A Control Register (DCACTL) Field Descriptions

Bit	Field	Value	Description
15-10	Reserved	0	Reserved
9	EVT2FRC SYNCSEL		DCAEVT2 Force Synchronization Signal Select.
		0	Source Is Synchronous Signal.
		1	Source Is Asynchronous Signal.
8	EVT2SRCSEL		DCAEVT2 Source Signal Select.
		0	Source Is DCAEVT2 Signal.
		1	Source Is DCEVTFILT Signal.
7-4	Reserved	0	Reserved
3	EVT1SYNCE		DCAEVT1 SYNC Enable.
		0	SYNC Generation is disabled.
		1	SYNC Generation is enabled.
2	EVT1SOCE		DCAEVT1 SOC Enable.
		0	SOC Generation is disabled.
		1	SOC Generation is enabled .
1	EVT1FRC SYNCSEL		DCAEVT1 Force Synchronization Signal Select.
		0	Source Is Synchronous Signal.
		1	Source Is Asynchronous Signal.
0	EVT1SRCSEL		DCAEVT1 Source Signal Select.
		0	Source Is DCAEVT1 Signal.
		1	Source Is DCEVTFILT Signal.

35.4.8.2 Digital Compare Trip Select (DCTRIPSEL)

Figure 35-92. Digital Compare Trip Select (DCTRIPSEL) [offset = 62h]

15	12	11	8
DCBLCOMPSEL		DCBHCOMPSEL	
R/W-0		R/W-0	

7	4	3	0
DCALCOMPSEL		DCAHCOMPSEL	
R/W-0		R/W-0	

LEGEND: R/W = Read/Write; -*n* = value after reset

Table 35-52. Digital Compare Trip Select (DCTRIPSEL) Field Descriptions

Bit	Field	Value	Description
15-12	DCBLCOMPSEL		Digital Compare B Low Input Select. Defines the source for the DCBL input. The TZ signals, when used as trip signals, are treated as normal inputs and can be defined as active high or active low.
		0	$\overline{\text{TZ1}}$ input
		1h	$\overline{\text{TZ2}}$ input
		2h	$\overline{\text{TZ3}}$ input
		All other values	Values not shown are reserved. If a device does not have a particular comparitor, then that option is reserved.
11-8	DCBHCOMPSEL		Digital Compare B High Input Select. Defines the source for the DCBH input. The TZ signals, when used as trip signals, are treated as normal inputs and can be defined as active high or active low.
		0	$\overline{\text{TZ1}}$ input
		1h	$\overline{\text{TZ2}}$ input
		2h	$\overline{\text{TZ3}}$ input
		All other values	Values not shown are reserved. If a device does not have a particular comparitor, then that option is reserved.
7-4	DCALCOMPSEL		Digital Compare A Low Input Select. Defines the source for the DCAL input. The TZ signals, when used as trip signals, are treated as normal inputs and can be defined as active high or active low.
		0	$\overline{\text{TZ1}}$ input
		1h	$\overline{\text{TZ2}}$ input
		2h	$\overline{\text{TZ3}}$ input
		All other values	Values not shown are reserved. If a device does not have a particular comparitor, then that option is reserved.
3-0	DCAHCOMPSEL		Digital Compare A High Input Select. Defines the source for the DCAH input. The TZ signals, when used as trip signals, are treated as normal inputs and can be defined as active high or active low.
		0	$\overline{\text{TZ1}}$ input
		1h	$\overline{\text{TZ2}}$ input
		2h	$\overline{\text{TZ3}}$ input
		All other values	Values not shown are reserved. If a device does not have a particular comparitor, then that option is reserved.

35.4.8.3 Digital Compare Filter Control Register (DCFCTL)

Figure 35-93. Digital Compare Filter Control Register (DCFCTL) [offset = 64h]

15 … 8
Reserved
R-0

7	6	5	4	3	2	1	0
Reserved		PULSESEL		BLANKINV	BLANKE	SRCSEL	
R-0		R/W-0		R/W-0	R/W-0	R/W-0	

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 35-53. Digital Compare Filter Control Register (DCFCTL) Field Descriptions

Bit	Field	Value	Description
15-6	Reserved	0	Reserved
5-4	PULSESEL		Pulse Select For Blanking & Capture Alignment.
		0	Time-base counter equal to period (TBCTR = TBPRD)
		1h	Time-base counter equal to zero (TBCTR = 0x0000)
		2h-3h	Reserved
3	BLANKINV		Blanking Window Inversion.
		0	Blanking window is not inverted.
		1	Blanking window is inverted.
2	BLANKE		Blanking Window Enable.
		0	Blanking window is disabled.
		1	Blanking window is enabled.
1-0	SRCSEL		Filter Block Signal Source Select.
		0	Source Is DCAEVT1 Signal.
		1h	Source Is DCAEVT2 Signal.
		2h	Source Is DCBEVT1 Signal.
		3h	Source Is DCBEVT2 Signal.

35.4.8.4 Digital Compare B Control Register (DCBCTL)

Figure 35-94. Digital Compare B Control Register (DCBCTL) [offset = 66h]

15-10	9	8
Reserved	EVT2FRC SYNCSEL	EVT2SRCSEL
R-0	R/W-0	R/W-0

7-4	3	2	1	0
Reserved	EVT1SYNCE	EVT1SOCE	EVT1FRC SYNCSEL	EVT1SRCSEL
R-0	R/W-0	R/W-0	R/W-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 35-54. Digital Compare B Control Register (DCBCTL) Field Descriptions

Bit	Field	Value	Description
15-10	Reserved	0	Reserved
9	EVT2FRC SYNCSEL		DCBEVT2 Force Synchronization Signal Select.
		0	Source Is Synchronous Signal.
		1	Source Is Asynchronous Signal.
8	EVT2SRCSEL		DCBEVT2 Source Signal Select.
		0	Source Is DCBEVT2 Signal.
		1	Source Is DCEVTFILT Signal.
7-4	Reserved	0	Reserved
3	EVT1SYNCE		DCBEVT1 SYNC, Enable.
		0	SYNC Generation is disabled.
		1	SYNC Generation is enabled.
2	EVT1SOCE		DCBEVT1 SOC, Enable.
		0	SOC Generation is disabled.
		1	SOC Generation is enabled.
1	EVT1FRC SYNCSEL		DCBEVT1 Force Synchronization Signal Select.
		0	Source Is Synchronous Signal.
		1	Source Is Asynchronous Signal.
0	EVT1SRCSEL		DCBEVT1 Source Signal Select.
		0	Source Is DCBEVT1 Signal.
		1	Source Is DCEVTFILT Signal.

35.4.8.5 Digital Compare Filter Offset Register (DCFOFFSET)

Figure 35-95. Digital Compare Filter Offset Register (DCFOFFSET) [offset = 68h]

15 — 0
DCOFFSET
R-0

LEGEND: R = Read only; -*n* = value after reset

Table 35-55. Digital Compare Filter Offset Register (DCFOFFSET) Field Descriptions

Bit	Field	Description
15-0	OFFSET	Blanking Window Offset These 16-bits specify the number of TBCLK cycles from the blanking window reference to the point when the blanking window is applied. The blanking window reference is either period or zero as defined by the DCFCTL[PULSESEL] bit. This offset register is shadowed and the active register is loaded at the reference point defined by DCFCTL[PULSESEL]. The offset counter is also initialized and begins to count down when the active register is loaded. When the counter expires, the blanking window is applied. If the blanking window is currently active, then the blanking window counter is restarted.

35.4.8.6 Digital Compare Capture Control Register (DCCAPCTL)

Figure 35-96. Digital Compare Capture Control Register (DCCAPCTL) [offset = 6Ah]

15 — 8
Reserved
R-0

7 — 2	1	0
Reserved	SHDWMODE	CAPE
R-0	R/W-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 35-56. Digital Compare Capture Control Register (DCCAPCTL) Field Descriptions

Bit	Field	Value	Description
15-2	Reserved	0	Reserved
1	SHDWMODE		TBCTR Counter Capture Shadow Select Mode.
		0	Enable shadow mode. The DCCAP active register is copied to shadow register on a TBCTR = TBPRD or TBCTR = zero event as defined by the DCFCTL[PULSESEL] bit. CPU reads of the DCCAP register will return the shadow register contents.
		1	Active Mode. In this mode the shadow register is disabled. CPU reads from the DCCAP register will always return the active register contents.
0	CAPE		TBCTR Counter Capture Enable.
		0	Time-base counter capture is disabled.
		1	Time-base counter capture is enabled.

35.4.8.7 Digital Compare Filter Window Register (DCFWINDOW)

Figure 35-97. Digital Compare Filter Window Register (DCFWINDOW) [offset = 6Ch]

15	8
Reserved	
R-0	

7	0
WINDOW	
R/W-0	

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 35-57. Digital Compare Filter Window Register (DCFWINDOW) Field Descriptions

Bit	Field	Value	Description
15-8	Reserved	0	Reserved
7-0	WINDOW		Blanking Window Width.
		0	No blanking window is generated.
		1h-FFh	Specifies the width of the blanking window in TBCLK cycles. The blanking window begins when the offset counter expires. When this occurs, the window counter is loaded and begins to count down. If the blanking window is currently active and the offset counter expires, the blanking window counter is restarted. The blanking window can cross a PWM period boundary.

35.4.8.8 Digital Compare Filter Offset Counter Register (DCFOFFSETCNT)

Figure 35-98. Digital Compare Filter Offset Counter Register (DCFOFFSETCNT) [offset = 6Eh]

15	0
OFFSETCNT	
R-0	

LEGEND: R = Read only; -*n* = value after reset

Table 35-58. Digital Compare Filter Offset Counter Register (DCFOFFSETCNT) Field Descriptions

Bit	Field	Description
15-0	OFFSETCNT	Blanking Offset Counter. These 16-bits are read only and indicate the current value of the offset counter. The counter counts down to zero and then stops until it is re-loaded on the next period or zero event as defined by the DCFCTL[PULSESEL] bit. The offset counter is not affected by the free/soft emulation bits. That is, it will always continue to count down if the device is halted by a emulation stop.

35.4.8.9 Digital Compare Counter Capture Register (DCCAP)

Figure 35-99. Digital Compare Counter Capture Register (DCCAP) [offset = 70h]

15 … 0
DCCAP
R-0

LEGEND: R = Read only; -*n* = value after reset

Table 35-59. Digital Compare Counter Capture Register (DCCAP) Field Descriptions

Bit	Field	Description
15-0	DCCAP	Digital Compare Time-Base Counter Capture. To enable time-base counter capture, set the DCCAPCLT[CAPE] bit to 1. If enabled, reflects the value of the time-base counter (TBCTR) on the low-to-high edge transition of a filtered (DCEVTFLT) event. Further capture events are ignored until the next period or zero as selected by the DCFCTL[PULSESEL] bit. Shadowing of DCCAP is enabled and disabled by the DCCAPCTL[SHDWMODE] bit. By default this register is shadowed. • If DCCAPCTL[SHDWMODE] = 0, then the shadow is enabled. In this mode, the active register is copied to the shadow register on the TBCTR = TBPRD or TBCTR = zero as defined by the DCFCTL[PULSESEL] bit. CPU reads of this register will return the shadow register value. • If DCCAPCTL[SHDWMODE] = 1, then the shadow register is disabled. In this mode, CPU reads will return the active register value. The active and shadow registers share the same memory-map address.

35.4.8.10 Digital Compare Filter Window Counter Register (DCFWINDOWCNT)

Figure 35-100. Digital Compare Filter Window Counter Register (DCFWINDOWCNT) [offset = 72h]

15 … 8
Reserved
R-0

7 … 0
WINDOWCNT
R-0

LEGEND: R = Read only; -*n* = value after reset

Table 35-60. Digital Compare Filter Window Counter Register (DCFWINDOWCNT) Field Descriptions

Bit	Field	Value	Description
15-8	Reserved	0	Any writes to these bit(s) must always have a value of 0.
7-0	WINDOWCNT	0-FFh	Blanking Window Counter. These 8 bits are read-only and indicate the current value of the window counter. The counter counts down to zero and then stops until it is re-loaded when the offset counter reaches zero again.

Chapter 36
SPNU563A–March 2018

Data Modification Module (DMM)

This chapter describes the functionality of the Data Modification Module (DMM), which provides the capability to modify data in the entire 4 GB address space of the device from an external peripheral, with minimal interruption of the application.

Topic **Page**

36.1 Overview

36.1.1 Features

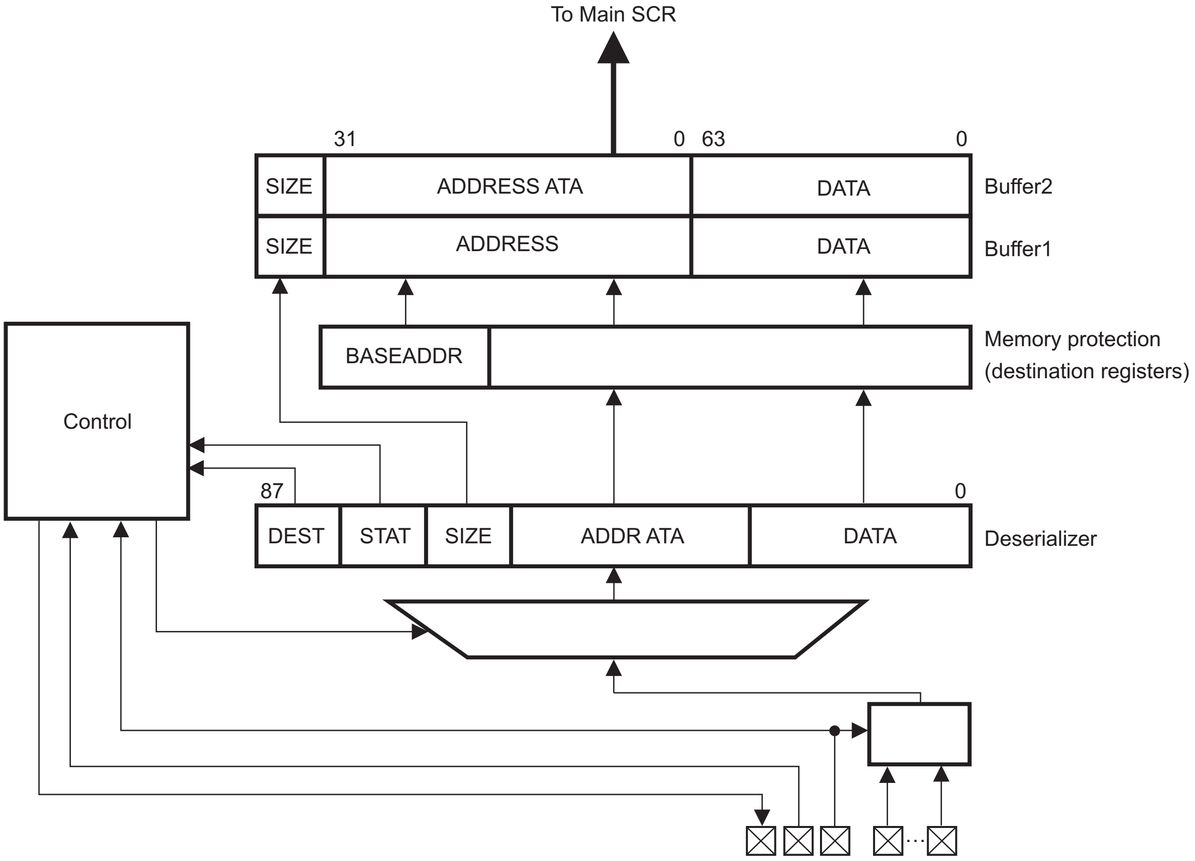
The DMM module has the following features:

- Acts as a bus master, thus enabling direct writes to the 4GB address space without CPU intervention
- Writes to memory locations specified in the received packet (leverages packets defined by trace mode of the RAM trace port (RTP) module
- Writes received data to consecutive addresses, which are specified by the DMM module (leverages packets defined by direct data mode of RTP module)
- Configurable port width (1, 2, 4, 8, 16 pins)
- Up to 100 Mbit/s pin data rate
- Unused pins configurable as GIO pins

36.1.2 Block Diagram

Figure 36-1 shows the block diagram for the DMM.

Figure 36-1. DMM Block Diagram



36.2 Module Operation

The DMM receives data over the DMM pins from external systems and writes the received data directly to the base address programmed in the module plus offset address given in the packet or into a buffer specified by start address and length. It leverages the protocol defined by the RAM Trace Port (RTP) module to have a common interface definition for external systems. It can also be used to connect an RTP and DMM module together for fast processor intercommunication.

The DMM module provides two modes of operation:

- **Trace Mode:** In this mode, the DMM writes the received data directly to an address that is calculated from the base address programmed into the destination register (Section 36.3.12; Section 36.3.14) plus the offset address contained in the received packet. An interrupt can be generated when data is written the lowest address of a programmed region. This capability enables the sender to raise an interrupt at the receiver while sending specific information.
- **Direct Data Mode:** In this mode, the DMM writes the received data into an address range of the 4GB address space. The buffer start address (Section 36.3.8) and blocksize (Section 36.3.9) is programmable in the DMM module. When the buffer reaches its end address, the buffer pointer wraps around and points to the beginning of the buffer again. The EO_BUFF flag (Section 36.3.5) will be set and if enabled, an interrupt will be generated to indicate a buffer-full condition. Another interrupt, can be configured to indicate different buffer fill levels. This can be accomplished by programming a certain fill level into the DMMINTPT register (Section 36.3.11). The PROG_BUFF flag (Section 36.3.5) indicates that this level has been reached.

Data will be captured by the input buffer and moved to the appropriate bit field in the deserializer. When the deserializer is completely full, the data will be moved to the output buffer register. A two-level buffer is implemented to avoid overflow conditions if the internal bus is occupied by other transactions. In addition the $\overline{\text{DMMENA}}$ signal can be used to signal the external hardware that an overflow might occur if more data is sent. The automatic generation of the $\overline{\text{DMMENA}}$ signal can be configured by setting the ENAFUNC bit (Section 36.3.16). While the $\overline{\text{DMMENA}}$ signal is active, the DMM module will not receive any new data.

The DMM is a bus master and forwards the received data to the bus system. The write operation will be minimally intrusive to the program flow, because the CPU/DMA access will only be blocked if the CPU/DMA accesses the same resource as the DMM.

To prevent an external system from overwriting critical data in the memory while configured in Trace Mode, a memory protection mechanism is implemented via a programmable start address and block size of a region. A maximum of four destinations with two regions each are supported.

For proper operation, at least DMMCLK, DMMSYNC and DMMDATA[0] need to be programmed in functional mode (Section 36.3.16). If a large amount of data should be transmitted in a short time, more data pins should be used in functional mode. The module supports 1, 2, 4, 8, or 16-pin configurations.

The module can be configured to handle a free running clock provided on DMMCLK (Section 36.3.1). Clock pulses between two DMMSYNC pulses that exceed the number of valid clock pulses for a packet will be ignored.

36.2.1 Data Format

Below is a description of the packet and frame format.

36.2.1.1 Clocking Scheme

The DMM supports both continuous and noncontinuous clocking. The clock received on DMMCLK in the continuous clocking scheme is a free-running clock. In noncontinuous clocking scheme, the clock will stop after each packet and will start with the reception of a DMMSYNC signal.

36.2.1.2 Trace Mode Packet

Figure 36-2 illustrates the trace mode packet format. One packet consists of 2 bits (DEST) denoting the destination in which the data is stored, 2 status bits (STAT), the 2-bit SIZE of the data, the 18-bit address of where the data should be written to, and a variable data field.

The DEST bits (Table 36-1) will be used to determine which destination register applies to the transmitted data and the received address determines if the packet falls into a valid region of the destination area. If the address is valid, the base address, programmed in one of the destination registers (Section 36.3.12; Section 36.3.14) of this particular region will be applied to create the complete 32-bit address for the destination. The DMM module only takes action on a "11" setting of the STAT bits (Table 36-2). This signals that an overflow in the transmitting hardware module has occurred. If this is the case the SRC_OVF flag (Section 36.3.5) will be set and the received data will be written to the address specified in the packet. The size information of the data transmitted in the packet is denoted in the SIZE bits (Table 36-3) of the packet. Depending on the SIZE information, the module expects to receive only this amount of data.

Figure 36-2. Trace Mode Packet Format

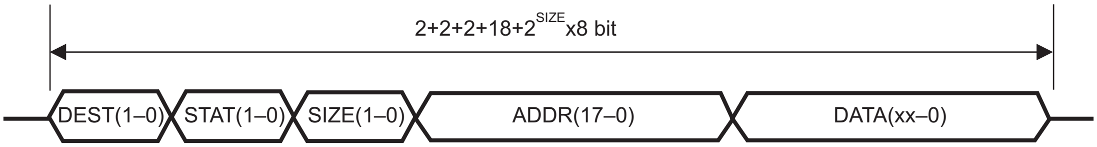


Table 36-1 through Table 36-3 illustrate the encoding of packet format in trace mode.

Table 36-1. Encoding of Destination Bits in Trace Mode Packet Format

DEST[1:0]	Destination
00	Dest 0
01	Dest 1
10	Dest 2
11	Dest 3

Table 36-2. Encoding of Status Bits in Trace Mode Packet Format

STAT[1:0]	Status
00	don't care
01	don't care
10	don't care
11	overflow

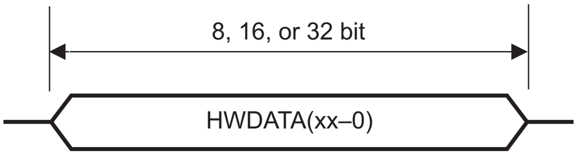
Table 36-3. Encoding of Write Size in Packet Format

SIZE[1:0]	Write Size
00	8 bit
01	16 bit
10	32 bit
11	64 bit

36.2.1.3 Direct Data Mode Packet

Figure 36-3 illustrates the direct data mode packet format.

Figure 36-3. Direct Data Mode Packet Format



The packet consists only of data bits and no header information. It can be 8-, 16- or 32-bit wide. A variable packet width is not supported because the DMM module will check the number of incoming bits (DMMCLK cycles) for error detection. The DMM will write the received data to the destination once the programmed number of bits has been received.

If the programmed word width does not correspond to the received data, the following actions will be taken:

- If the received data is greater than the programmed width, only the configured number of bits are transferred into the RAM buffer, the additional bits are discarded.
- If the received number of bits is smaller than the programmed width, no data will be written to the buffer, because a new DMMSYNC signal has been received before the expected number of bits.

36.2.2 Data Port

The packet will be received in several subpackets, depending on the width of the external data bus (DMMDATA[y:0]) and the amount of data to be transmitted. Table 36-4 illustrates the number of clock cycles required for a complete packet.

Table 36-4. Number of Clock Cycles per Packet

Port Width/ Pins	Write Size in Bits			
	8	16	32	64
1	32	40	56	88
2	16	20	28	44
4	8	10	14	22
8	4	5	7	11
16	2	3	4	6

The user can program the port width in the DMMPC0 register (Section 36.3.16). This feature allows pins that are not used for DMM functionality to be used as GIO pins. Only the pins shown in Table 36-5 can be used for a desired port width.

Table 36-5. Pins Used for Data Communication

Port Width	Pins Used
1	DMMDATA[0]
2	DMMDATA[1:0]
4	DMMDATA[3:0]
8	DMMDATA[7:0]
16	DMMDATA[15:0]

NOTE: If pins other than the ones specified in Table 36-5 are programmed as functional pins for a desired port width, the received data will be corrupted and will not be transferred to the deserializer.

NOTE: If DMMCLK or DMMSYNC are programmed as nonfunctional pins, functional operation will not occur.

36.2.2.1 Signal Description

DMMSYNC — This signal has to be provided by external hardware. It signals the start of a new packet. It has to be active (high) for one full DMMCLK cycle, starting with the rising edge of DMMCLK. If the DMMSYNC pulse is longer than a single DMMCLK cycle and two falling edges of DMMCLK see a high pulse on DMMSYNC, the module will treat the second DMMSYNC pulse as the start of a packet and will flag a PACKET_ERR_INT (Section 36.3.5).

DMMCLK — The clock is externally generated and can be suspended between two packets. For this feature, CONTCLK must be set to 0 (Section 36.3.1). If the clock is not stopped between two packets, CONTCLK must be set to 1. Data will be latched on the falling edge of the DMMCLK signal.

$\overline{\text{DMMENA}}$ — This signal is pulled high if no new data should be received via the data pins, because of a potential overflow situation.

DMMDATA[15:0] — These pins receive the packet information transmitted by the external hardware. Data is latched on the falling edge of DMMCLK.

Figure 36-4 shows an example of multiple packets received during trace mode, in noncontinuous clock configuration.

Figure 36-4. Packet Sync Signal Example

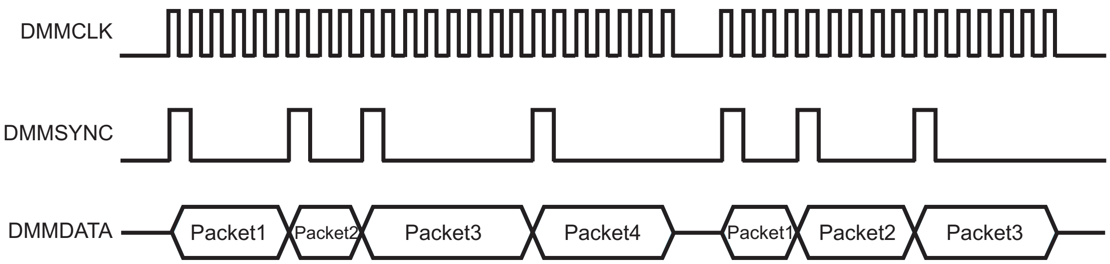
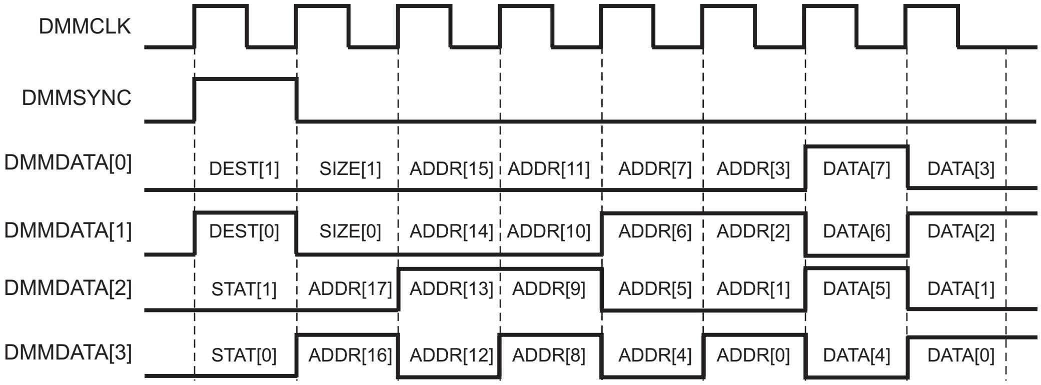


Figure 36-5 shows an example of a 4-bit data port with 8-bit receive data (A5h) to be written into DEST1 (address 0001 2345h) on a trace mode packet.

Figure 36-5. Example Single Packet Transmission



36.2.3 Error Handling

The module will generate two different kind of errors. Once an error condition is recognized, an interrupt will be generated if enabled.

36.2.3.1 Overflow Error

This error is signaled when the module has received new data before the previous data was written to the destination address. If the internal buffers are full, the $\overline{\text{DMMENA}}$ signal will go high. If the sending module does not evaluate the $\overline{\text{DMMENA}}$ signal and keeps on sending new frames, the data that was previously received might be overwritten, thus resulting in setting the BUFF_OVF flag (Section 36.3.5).

36.2.3.2 Packet Error

Noncontinuous Clock Mode

The size of the incoming packet is defined by the SIZE information of a trace mode packet or the programmed size of a direct data mode packet. If too many or less than the number of bits are received before the next sync signal, the PACKET_ERR_INT flag will be set (Section 36.3.5). In case of receiving a DMMCLK signal without a corresponding DMMSYNC signal, a packet error will also be generated.

Continuous Clock Mode

If less than the expected number of bits are received, the PACKET_ERR_INT flag will be set (Section 36.3.5) when the next DMMSYNC signal is received. Packets with more than the expected number of bits cannot be detected.

The check for packet error is done only after the detection of the first DMMSYNC signal after the DMM is turned on or comes out of suspend mode (with COS = 0; Section 36.3.1), that is, before the reception of first DMMSYNC, the toggling of DMMCLK would be ignored.

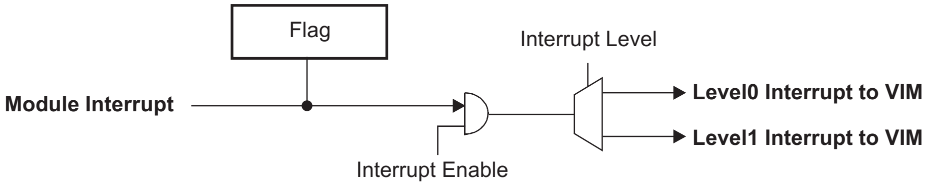
36.2.3.3 Bus Error

If an error occurs on the microcontroller internal bus system while transferring the data from the DMM to the destination, the BUSERROR flag will be set.

36.2.4 Interrupts

The module provides different interrupts. These can be programmed to different interrupt levels independently using DMMINTLVL (Section 36.3.4).

Figure 36-6. Interrupt Structure



Interrupts can be divided into error interrupts and functional interrupts. The error handling is described in Section 36.2.3. Functional interrupts depend on the mode (Trace Mode, Direct Data Mode) the DMM module is used in.

Trace Mode: An interrupt can be enabled whenever an access to the lowest address of a defined region is performed. This address is the starting address programmed in the DMMDESTxREGy register. An interrupt for each of the region can be generated by setting the individual interrupt enable bits.

Direct Data Mode: There are two interrupts that can be individually controlled. One is generated when the buffer pointer reaches the end of the defined buffer and wraps around (EO_BUFF; Section 36.3.2). The other one is generated when the buffer pointer matches the programmed interrupt threshold (PROG_BUFF; Section 36.3.2). The buffer pointer points to the next address to be written, therefor there are (interrupt threshold - 1) values stored in the buffer. The interrupt threshold can be programmed in the DMMINTPT register (Section 36.3.11).

36.3 Control Registers

This section describes the DMM registers. The registers support 8, 16, and 32-bit writes. The offset is relative to the associated peripheral select. Table 36-6 provides a summary of the registers and their bits. The base address of the DMM module registers is FFFF F700h.

Table 36-6. DMM Registers

Offset	Acronym	Register Description	Section
0h	DMMGLBCTRL	DMM Global Control Register	Section 36.3.1
4h	DMMINTSET	DMM Interrupt Set Register	Section 36.3.2
8h	DMMINTCLR	DMM Interrupt Clear Register	Section 36.3.3
0Ch	DMMINTLVL	DMM Interrupt Level Register	Section 36.3.4
10h	DMMINTFLG	DMM Interrupt Flag Register	Section 36.3.5
14h	DMMOFF1	DMM Interrupt Offset 1 Register	Section 36.3.6
18h	DMMOFF2	DMM Interrupt Offset 2 Register	Section 36.3.7
1Ch	DMMDDMDEST	DMM Direct Data Mode Destination Register	Section 36.3.8
20h	DMMDDMBL	DMM Direct Data Mode Blocksize Register	Section 36.3.9
24h	DMMDDMPT	DMM Direct Data Mode Pointer Register	Section 36.3.10
28h	DMMINTPT	DMM Direct Data Mode Interrupt Pointer Register	Section 36.3.11
2Ch, 3Ch, 4Ch, 5Ch	DMMDESTxREG1	DMM Destination x Region 1	Section 36.3.12
30h, 40h, 50h, 60h	DMMDESTxBL1	DMM Destination x Blocksize 1	Section 36.3.13
34h, 44h, 54h, 64h	DMMDESTxREG2	DMM Destination x Region 2	Section 36.3.14
38h, 48h, 58h, 68h	DMMDESTxBL2	DMM Destination x Blocksize 2	Section 36.3.15
6Ch	DMMPC0	DMM Pin Control 0	Section 36.3.16
70h	DMMPC1	DMM Pin Control 1	Section 36.3.17
74h	DMMPC2	DMM Pin Control 2	Section 36.3.18
78h	DMMPC3	DMM Pin Control 3	Section 36.3.19
7Ch	DMMPC4	DMM Pin Control 4	Section 36.3.20
80h	DMMPC5	DMM Pin Control 5	Section 36.3.21
84h	DMMPC6	DMM Pin Control 6	Section 36.3.22
88h	DMMPC7	DMM Pin Control 7	Section 36.3.23
8Ch	DMMPC8	DMM Pin Control 8	Section 36.3.24

36.3.1 DMM Global Control Register (DMMGLBCTRL)

With this register the basic operation of the module is selected.

Figure 36-7. DMM Global Control Register (DMMGLBCTRL) [offset = 00h]

31 25	24
Reserved	BUSY
R-0	R-0

23 19	18	17	16
Reserved	CONTCLK	COS	RESET
R-0	R/WP-0	R/WP-0	R/WP-0

15 11	10 9	8
Reserved	DDM_WIDTH	TM_DDM
R-0	R/WP-0	R/WP-0

7 4	3 0
Reserved	ON/OFF
R-0	R/WP-5h

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privileged mode only; -*n* = value after reset

Table 36-7. DMM Global Control Register (DMMGLBCTRL) Field Descriptions

Bit	Field	Value	Description
31-25	Reserved	0	Reads returns 0. Writes have no effect.
24	BUSY		Busy indicator.
		0	The DMM does not currently receive data and has no data in its internal buffers, which needs to be transferred.
		1	The module is currently receiving data, or has data in its internal buffers.
23-19	Reserved	0	Reads returns 0. Writes have no effect.
18	CONTCLK		Continuous DMMCLK input.
			User and privilege mode read, privilege mode write:
		0	DMMCLK is expected to be suspended between two packets.
		1	DMMCLK is expected to be free running between packets.
17	COS		Continue on suspend. Influences behavior of module while in debug mode. In all cases the corresponding interrupt will be set.
			User and privilege mode (read):
		0	Packets will not be received during debug mode. Before entering debug mode, the ongoing reception of a packet will be finished and the value will be written to the destination.
		1	Continue receiving packets and update destination, while in debug mode.
			Privilege mode (write):
		0	Disable data reception while in debug mode.
		1	Enable data reception while in debug mode.
16	RESET		Reset. This bit resets the state machine and the registers to its reset value, except the RESET bit itself. It must be cleared by writing to it.
			User and privilege mode (read):
		0	No reset of DMM module.
		1	Reset of DMM module.
			Privilege mode (write):
		0	No reset of DMM module.
		1	Reset DMM module to its reset state.
15-11	Reserved	0	Reads returns 0. Writes have no effect.

Table 36-7. DMM Global Control Register (DMMGLBCTRL) Field Descriptions (continued)

Bit	Field	Value	Description
10-9	DDM_WIDTH		Packet Width in direct data mode.
			User and privilege mode read and privilege mode write operation:
		Bit Encoding	Transfer Size
		0	8 bit
		1h	16 bit
		2h	32 bit
		3h	Reserved
8	TM_DMM		Packet Format.
			User and privilege mode (read):
		0	The DMM module assumes packets in trace mode definition.
		1	The DMM module assumes packets in direct data mode definition.
			Privilege mode (write):
		0	Enable trace mode.
		1	Enable direct data mode.
7-4	Reserved	0	Reads returns 0. Writes have no effect.
3-0	ON/OFF		Switch module on or off
			User and privilege mode (read):
		All other	The DMM module does not receive data.
		Ah	The DMM module receives data and writes it to the buffer.
			Privilege mode (write):
		All other	Disable receive/write operations. Packets in reception, will still be finished.
		Ah	Enable receive/write operations. Packets will be received 1 HCLK cycle after enabling the module.

NOTE: It is recommended to write 5h to ON/OFF to avoid having a soft error inadvertently enabling the module when a single bit flips.

NOTE: Registers that affect the operation of the module, should be only programmed when the BUSY bit is 0 and the ON/OFF bits are not Ah.

NOTE: If the module was in operation, turned off (ON/OFF = all other than Ah) and then turned on (ON/OFF = Ah) again, it is recommended to perform a reset (RESET = 1) of the module before switching it on. This avoids that the state machine is held in an unrecoverable state.

NOTE: A write to these register bits while receiving a packet will not have any effect on the received packet. The mode change will be performed after the packet is received

36.3.2 DMM Interrupt Set Register (DMMINTSET)

This register contains the interrupt set bits for error interrupts and functional interrupts. Only the bits which are relevant for the particular mode (trace mode or direct data mode) will be taken into account for the interrupt generation.

Figure 36-8. DMM Interrupt Set Register (DMMINTSET) [offset = 04h]

31–24
Reserved
R-0

23–18	17	16
Reserved	PROG_BUFF	EO_BUFF
R-0	R/WP-0	R/WP-0

15	14	13	12	11	10	9	8
DEST3REG2	DEST3REG1	DEST2REG2	DEST2REG1	DEST1REG2	DEST1REG1	DEST0REG2	DEST0REG1
R/WP-0	R/WP-0	R/WP-0	R/WP-0	R/WP-0	R/WP-0	R/WP-0	R/WP-0

7	6	5	4	3	2	1	0
BUSERROR	BUFF_OVF	SRC_OVF	DEST3_ERR	DEST2_ERR	DEST1_ERR	DEST0_ERR	PACKET_ ERR_INT
R/WP-0	R/WP-0	R/WP-0	R/WP-0	R/WP-0	R/WP-0	R/WP-0	R/WP-0

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privileged mode only; -*n* = value after reset

Table 36-8. DMM Interrupt Set Register (DMMINTSET) Field Descriptions

Bit	Field	Value	Description
31-18	Reserved	0	Reads returns 0. Writes have no effect.
17	PROG_BUFF		**Programmable Buffer Interrupt Set.** This enables the interrupt generation in case the buffer pointer equals the programmed value in the DMMINTPT register (Section 36.3.11). This bit is only relevant in Direct Data Mode.
			User and privilege mode (read):
		0	No interrupt will be generated.
		1	An interrupt will be generated on pointer match.
			Privilege mode (write):
		0	No influence on bit.
		1	Enable interrupt (sets corresponding bit in DMMINTCLR; DMMINTLVL).
16	EO_BUFF		**End of Buffer Interrupt Set.** This enables the interrupt generation in case data was written to the last entry in the buffer and the pointer wrapped around to the beginning of the buffer. This bit is only relevant in Direct Data Mode.
			User and privilege mode (read):
		0	No interrupt will be generated.
		1	An interrupt will be generated on writing to the last entry.
			Privilege mode (write):
		0	No influence on bit.
		1	Enable interrupt (sets corresponding bit in DMMINTCLR; DMMINTLVL).
15	DEST3REG2		**Destination 3 Region 2 Interrupt Set.** This enables the interrupt generation in case data was accessed at the start address of Destination 3 Region 2. This bit is only relevant in Trace Mode.
			User and privilege mode (read):
		0	No interrupt will be generated.
		1	An interrupt will be generated on a write to the start address of this region.
			Privilege mode (write):
		0	No influence on bit.
		1	Enable interrupt (sets corresponding bit in DMMINTCLR; DMMINTLVL).

Table 36-8. DMM Interrupt Set Register (DMMINTSET) Field Descriptions (continued)

Bit	Field	Value	Description
14	DEST3REG1		**Destination 3 Region 1 Interrupt Set.** This enables the interrupt generation in case data was accessed at the start address of Destination 3 Region 1. This bit is only relevant in Trace Mode.
			User and privilege mode (read):
		0	No interrupt will be generated.
		1	An interrupt will be generated on a write to the start address of this region.
			Privilege mode (write):
		0	No influence on bit.
		1	Enable interrupt (sets corresponding bit in DMMINTCLR; DMMINTLVL).
13	DEST2REG2		**Destination 2 Region 2 Interrupt Set.** This enables the interrupt generation in case data was accessed at the start address of Destination 2 Region 2. This bit is only relevant in Trace Mode.
			User and privilege mode (read):
		0	No interrupt will be generated.
		1	An interrupt will be generated on a write to the start address of this region.
			Privilege mode (write):
		0	No influence on bit.
		1	Enable interrupt (sets corresponding bit in DMMINTCLR; DMMINTLVL).
12	DEST2REG1		**Destination 2 Region 1 Interrupt Set.** This enables the interrupt generation in case data was accessed at the start address of Destination 2 Region 1. This bit is only relevant in Trace Mode.
			User and privilege mode (read):
		0	No interrupt will be generated.
		1	An interrupt will be generated on a write to the start address of this region.
			Privilege mode (write):
		0	No influence on bit.
		1	Enable interrupt (sets corresponding bit in DMMINTCLR; DMMINTLVL).
11	DEST1REG2		**Destination 1 Region 2 Interrupt Set.** This enables the interrupt generation in case data was accessed at the start address of Destination 1 Region 2. This bit is only relevant in Trace Mode.
			User and privilege mode (read):
		0	No interrupt will be generated.
		1	An interrupt will be generated on a write to the start address of this region.
			Privilege mode (write):
		0	No influence on bit.
		1	Enable interrupt (sets corresponding bit in DMMINTCLR; DMMINTLVL).
10	DEST1REG1		**Destination 1 Region 1 Interrupt Set.** This enables the interrupt generation in case data was accessed at the start address of Destination 1 Region 1. This bit is only relevant in Trace Mode.
			User and privilege mode (read):
		0	No interrupt will be generated.
		1	An interrupt will be generated on a write to the start address of this region.
			Privilege mode (write):
		0	No influence on bit.
		1	Enable interrupt (sets corresponding bit in DMMINTCLR; DMMINTLVL).
9	DEST0REG2		**Destination 0 Region 2 Interrupt Set.** This enables the interrupt generation in case data was accessed at the start address of Destination 0 Region 2. This bit is only relevant in Trace Mode.
			User and privilege mode (read):
		0	No interrupt will be generated.
		1	An interrupt will be generated on a write to the start address of this region.
			Privilege mode (write):
		0	No influence on bit.
		1	Enable interrupt (sets corresponding bit in DMMINTCLR; DMMINTLVL).

Table 36-8. DMM Interrupt Set Register (DMMINTSET) Field Descriptions (continued)

Bit	Field	Value	Description
8	DEST0REG1		**Destination 0 Region 1 Interrupt Set.** This enables the interrupt generation in case data was accessed at the start address of Destination 0 Region 1. This bit is only relevant in Trace Mode.
			User and privilege mode (read):
		0	No interrupt will be generated
		1	An interrupt will be generated on a write to the start address of this region
			Privilege mode (write):
		0	No influence on bit.
		1	Enable interrupt (sets corresponding bit in DMMINTCLR; DMMINTLVL).
7	BUSERROR		Bus Error Response for errors generated when doing internal bus transfers.
			User and privilege mode (read):
		0	No interrupt will be generated.
		1	An interrupt will be generated.
			Privilege mode (write):
		0	No influence on bit.
		1	Enable interrupt (sets corresponding bit in DMMINTCLR; DMMINTLVL).
6	BUFF_OVF		**Buffer Overflow.** This enables the interrupt generation in case new data is received, while the previous data still has not been transmitted.
			User and privilege mode (read):
		0	No interrupt will be generated.
		1	An interrupt will be generated.
			Privilege mode (write):
		0	No influence on bit.
		1	Enable interrupt (sets corresponding bit in DMMINTCLR; DMMINTLVL).
5	SRC_OVF		**Source Overflow.** This enables an interrupt if the external system experienced and overflow that was signaled in the Trace Mode packet.
			User and privilege mode (read):
		0	No interrupt will be generated.
		1	An interrupt will be generated.
			Privilege mode (write):
		0	No influence on bit.
		1	Enable interrupt (sets corresponding bit in DMMINTCLR; DMMINTLVL).
4	DEST3_ERR		**Destination 3 Error.** This enables the interrupt generation in case data should be written into a address not specified by DMMDEST3REG1/DMMDEST3BL1 or DMMDEST3REG2/DMMDEST3BL2. If both blocksizes are programmed to 0 or a reserved value, the interrupt will still be generated, the write to the internal RAM however will not take place.
			User and privilege mode (read):
		0	No interrupt will be generated.
		1	An interrupt will be generated.
			Privilege mode (write):
		0	No influence on bit.
		1	Enable interrupt (sets corresponding bit in DMMINTCLR; DMMINTLVL).

Table 36-8. DMM Interrupt Set Register (DMMINTSET) Field Descriptions (continued)

Bit	Field	Value	Description
3	DEST2_ERR		**Destination 2 Error Interrupt Set.** This enables the interrupt generation in case data should be written into a address not specified by DMMDEST2REG1/DMMDEST2BL1 or DMMDEST2REG2/DMMDEST2BL2. If both blocksizes are programmed to 0 or a reserved value, the interrupt will still be generated, the write to the internal RAM however will not take place.
			User and privilege mode (read):
		0	No interrupt will be generated.
		1	An interrupt will be generated.
			Privilege mode (write):
		0	No influence on bit.
		1	Enable interrupt (sets corresponding bit in DMMINTCLR; DMMINTLVL).
2	DEST1_ERR		**Destination 1 Error Interrupt Set.** This enables the interrupt generation in case data should be written into a address not specified by DMMDEST1REG1/DMMDEST1BL1 or DMMDEST1REG2/DMMDEST1BL2. If both blocksizes are programmed to 0 or a reserved value, the interrupt will still be generated, the write to the internal RAM however will not take place.
			User and privilege mode (read):
		0	No interrupt will be generated.
		1	An interrupt will be generated.
			Privilege mode (write):
		0	No influence on bit.
		1	Enable interrupt (sets corresponding bit in DMMINTCLR; DMMINTLVL).
1	DEST0_ERR		**Destination 0 Error Interrupt Set.** This enables the interrupt generation in case data should be written into a address not specified by DMMDEST0REG1/DMMDEST0BL1 or DMMDEST0REG2/DMMDEST0BL2. If both blocksizes are programmed to 0 or a reserved value, the interrupt will still be generated, the write to the internal RAM however will not take place.
			User and privilege mode (read):
		0	No interrupt will be generated.
		1	An interrupt will be generated.
			Privilege mode (write):
		0	No influence on bit.
		1	Enable interrupt (sets corresponding bit in DMMINTCLR; DMMINTLVL).
0	PACKET_ERR_INT		**Packet Error.** This enables the interrupt generation in case of an error condition in the packet reception. Please refer to Section 36.2.3 for the error conditions.
			User and privilege mode (read):
		0	No interrupt will be generated.
		1	An interrupt will be generated.
			Privilege mode (write):
		0	No influence on bit.
		1	Enable interrupt (sets corresponding bit in DMMINTCLR; DMMINTLVL).

36.3.3 DMM Interrupt Clear Register (DMMINTCLR)

This register contains the interrupt clear bits for error interrupts and functional interrupts. Only the bits which are relevant for the particular mode (trace mode or direct data mode) will be taken into account for the interrupt generation

Figure 36-9. DMM Interrupt Clear Register (DMMINTCLR) [offset = 08h]

31							24
Reserved							
R-0							
23					18	17	16
Reserved						PROG_BUFF	EO_BUFF
R-0						R/WP-0	R/WP-0
15	14	13	12	11	10	9	8
DEST3REG2	DEST3REG1	DEST2REG2	DEST2REG1	DEST1REG2	DEST1REG1	DEST0REG2	DEST0REG1
R/WP-0	R/WP-0	R/WP-0	R/WP-0	R/WP-0	R/WP-0	R/WP-0	R/WP-0
7	6	5	4	3	2	1	0
BUSERROR	BUFF_OVF	SRC_OVF	DEST3_ERR	DEST2_ERR	DEST1_ERR	DEST0_ERR	PACKET_ ERR_INT
R/WP-0	R/WP-0	R/WP-0	R/WP-0	R/WP-0	R/WP-0	R/WP-0	R/WP-0

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privileged mode only; -*n* = value after reset

Table 36-9. DMM Interrupt Clear Register (DMMINTCLR) Field Descriptions

Bit	Field	Value	Description
31-18	Reserved	0	Reads returns 0. Writes have no effect.
17	PROG_BUFF		**Programmable Buffer Interrupt Set.**This disables the interrupt generation in case the buffer pointer equals the programmed value in the DMMINTPT register (Section 36.3.11). This bit is only relevant in Direct Data Mode.
			User and privilege mode (read):
		0	No interrupt will be generated.
		1	An interrupt will be generated on pointer match.
			Privilege mode (write):
		0	No influence on bit.
		1	Disable interrupt (clears corresponding bit in DMMINTCLR; DMM Interrupt Level Register (DMMINTLVL)).
16	EO_BUFF		**End of Buffer Interrupt Set.**This disables the interrupt generation in case data was written to the last entry in the buffer and the pointer wrapped around to the beginning of the buffer. This bit is only relevant in Direct Data Mode.
			User and privilege mode (read):
		0	No interrupt will be generated.
		1	An interrupt will be generated on writing to the last entry.
			Privilege mode (write):
		0	No influence on bit.
		1	Disable interrupt (clears corresponding bit in DMMINTCLR; DMM Interrupt Level Register (DMMINTLVL)).

Table 36-9. DMM Interrupt Clear Register (DMMINTCLR) Field Descriptions (continued)

Bit	Field	Value	Description
15	DEST3REG2		**Destination 3 Region 2 Interrupt Set.**This disables the interrupt generation in case data was accessed at the start address of Destination 3 Region 2. This bit is only relevant in Trace Mode.
			User and privilege mode (read):
		0	No interrupt will be generated.
		1	An interrupt will be generated on a write to the start address of this region.
			Privilege mode (write):
		0	No influence on bit.
		1	Disable interrupt (clears corresponding bit in DMMINTCLR; DMM Interrupt Level Register (DMMINTLVL)).
14	DEST3REG1		**Destination 3 Region 1 Interrupt Set.**This disables the interrupt generation in case data was accessed at the start address of Destination 3 Region 1. This bit is only relevant in Trace Mode.
			User and privilege mode (read):
		0	No interrupt will be generated.
		1	An interrupt will be generated on a write to the start address of this region.
			Privilege mode (write):
		0	No influence on bit.
		1	Disable interrupt (clears corresponding bit in DMMINTCLR; DMM Interrupt Level Register (DMMINTLVL)).
13	DEST2REG2		**Destination 2 Region 2 Interrupt Set.**This disables the interrupt generation in case data was accessed at the start address of Destination 2 Region 2. This bit is only relevant in Trace Mode.
			User and privilege mode (read):
		0	No interrupt will be generated.
		1	An interrupt will be generated on a write to the start address of this region.
			Privilege mode (write):
		0	No influence on bit.
		1	Disable interrupt (clears corresponding bit in DMMINTCLR; DMM Interrupt Level Register (DMMINTLVL)).
12	DEST2REG1		**Destination 2 Region 1 Interrupt Set.**This disables the interrupt generation in case data was accessed at the start address of Destination 2 Region 1. This bit is only relevant in Trace Mode.
			User and privilege mode (read):
		0	No interrupt will be generated.
		1	An interrupt will be generated on a write to the start address of this region.
			Privilege mode (write):
		0	No influence on bit.
		1	Disable interrupt (clears corresponding bit in DMMINTCLR; DMM Interrupt Level Register (DMMINTLVL)).
11	DEST1REG2		**Destination 1 Region 2 Interrupt Set.**This disables the interrupt generation in case data was accessed at the start address of Destination 1 Region 2. This bit is only relevant in Trace Mode.
			User and privilege mode (read):
		0	No interrupt will be generated.
		1	An interrupt will be generated on a write to the start address of this region.
			Privilege mode (write):
		0	No influence on bit.
		1	Disable interrupt (clears corresponding bit in DMMINTCLR; DMM Interrupt Level Register (DMMINTLVL)).

Table 36-9. DMM Interrupt Clear Register (DMMINTCLR) Field Descriptions (continued)

Bit	Field	Value	Description
10	DEST1REG1		**Destination 1 Region 1 Interrupt Set.**This enables the interrupt generation in case data was accessed at the start address of Destination 1 Region 1. This bit is only relevant in Trace Mode.
			User and privilege mode (read):
		0	No interrupt will be generated.
		1	An interrupt will be generated on a write to the start address of this region.
			Privilege mode (write):
		0	No influence on bit.
		1	Disable interrupt (clears corresponding bit in DMMINTCLR; DMM Interrupt Level Register (DMMINTLVL)).
9	DEST0REG2		**Destination 0 Region 2 Interrupt Set.**This disables the interrupt generation in case data was accessed at the start address of Destination 0 Region 2. This bit is only relevant in Trace Mode.
			User and privilege mode (read):
		0	No interrupt will be generated.
		1	An interrupt will be generated on a write to the start address of this region.
			Privilege mode (write):
		0	No influence on bit.
		1	Disable interrupt (clears corresponding bit in DMMINTCLR; DMM Interrupt Level Register (DMMINTLVL)).
8	DEST0REG1		**Destination 0 Region 1 Interrupt Set.**This disables the interrupt generation in case data was accessed at the start address of Destination 0 Region 1. This bit is only relevant in Trace Mode.
			User and privilege mode (read):
		0	No interrupt will be generated.
		1	An interrupt will be generated on a write to the start address of this region.
			Privilege mode (write):
		0	No influence on bit.
		1	Disable interrupt (clears corresponding bit in DMMINTCLR; DMM Interrupt Level Register (DMMINTLVL)).
7	BUSERROR		**Bus Error Response** for errors generated when doing internal bus transfers.
			User and privilege mode (read):
		0	No interrupt will be generated.
		1	An interrupt will be generated.
			Privilege mode (write):
		0	No influence on bit.
		1	Disable interrupt (clears corresponding bit in DMMINTCLR; DMM Interrupt Level Register (DMMINTLVL)).
6	BUFF_OVF		**Buffer Overflow.**This disables the interrupt generation in case new data is received, while the previous data still has not been transmitted.
			User and privilege mode (read):
		0	No interrupt will be generated.
		1	An interrupt will be generated.
			Privilege mode (write):
		0	No influence on bit.
		1	Disable interrupt (clears corresponding bit in DMMINTCLR; DMM Interrupt Level Register (DMMINTLVL)).

Table 36-9. DMM Interrupt Clear Register (DMMINTCLR) Field Descriptions (continued)

Bit	Field	Value	Description
5	SRC_OVF		**Source Overflow.** This disables an interrupt if the external system experienced and overflow which was signaled in the Trace Mode packet.
			User and privilege mode (read):
		0	No interrupt will be generated.
		1	An interrupt will be generated.
			Privilege mode (write):
		0	No influence on bit.
		1	Disable interrupt (clears corresponding bit in DMMINTCLR; DMM Interrupt Level Register (DMMINTLVL)).
4	DEST3_ERR		**Destination 3 Error.**This disables the interrupt generation in case data should be written into a address not specified by DMMDEST3REG1/DMMDEST3BL1 or DMMDEST3REG2/DMMDEST3BL2. If both blocksizes are programmed to 0 or a reserved value, the interrupt will still be generated, the write to the internal RAM however will not take place.
			User and privilege mode (read):
		0	No interrupt will be generated.
		1	An interrupt will be generated.
			Privilege mode (write):
		0	No influence on bit.
		1	Disable interrupt (clears corresponding bit in DMMINTCLR; DMM Interrupt Level Register (DMMINTLVL)).
3	DEST2_ERR		**Destination 2 Error Interrupt Set.**This disables the interrupt generation in case data should be written into a address not specified by DMMDEST2REG1/DMMDEST2BL1 or DMMDEST2REG2/DMMDEST2BL2. If both blocksizes are programmed to 0 or a reserved value, the interrupt will still be generated, the write to the internal RAM however will not take place.
			User and privilege mode (read):
		0	No interrupt will be generated.
		1	An interrupt will be generated.
			Privilege mode (write):
		0	No influence on bit.
		1	Disable interrupt (clears corresponding bit in DMMINTCLR; DMM Interrupt Level Register (DMMINTLVL)).
2	DEST1_ERR		**Destination 1 Error Interrupt Set.**This disables the interrupt generation in case data should be written into a address not specified by DMMDEST1REG1/DMMDEST1BL1 or DMMDEST1REG2/DMMDEST1BL2. If both blocksizes are programmed to 0 or a reserved value, the interrupt will still be generated, the write to the internal RAM however will not take place.
			User and privilege mode (read):
		0	No interrupt will be generated.
		1	An interrupt will be generated.
			Privilege mode (write):
		0	No influence on bit.
		1	Disable interrupt (clears corresponding bit in DMMINTCLR; DMM Interrupt Level Register (DMMINTLVL)).

Table 36-9. DMM Interrupt Clear Register (DMMINTCLR) Field Descriptions (continued)

Bit	Field	Value	Description
1	DEST0_ERR		**Destination 0 Error Interrupt Set.**This disables the interrupt generation in case data should be written into a address not specified by DMMDEST0REG1/DMMDEST0BL1 or DMMDEST0REG2/DMMDEST0BL2. If both blocksizes are programmed to 0 or a reserved value, the interrupt will still be generated, the write to the internal RAM however will not take place.
			User and privilege mode (read):
		0	No interrupt will be generated.
		1	An interrupt will be generated.
			Privilege mode (write):
		0	No influence on bit.
		1	Disable interrupt (clears corresponding bit in DMMINTCLR; DMM Interrupt Level Register (DMMINTLVL)).
0	PACKET_ERR_INT		**Packet Error.**This disables the interrupt generation in case of an error condition in the packet reception. Please refer to Section 36.2.3for the error conditions.
			User and privilege mode (read):
		0	No interrupt will be generated.
		1	An interrupt will be generated.
			Privilege mode (write):
		0	No influence on bit.
		1	Disable interrupt (clears corresponding bit in DMMINTCLR; DMM Interrupt Level Register (DMMINTLVL)).

36.3.4 DMM Interrupt Level Register (DMMINTLVL)

This register contains the interrupt level bits for error interrupts and normal interrupts.

Figure 36-10. DMM Interrupt Level Register (DMMINTLVL) [offset = 0Ch]

31	24
Reserved	
R-0	

23	18	17	16
Reserved		PROG_BUFF	EO_BUFF
R-0		R/WP-0	R/WP-0

15	14	13	12	11	10	9	8
DEST3REG2	DEST3REG1	DEST2REG2	DEST2REG1	DEST1REG2	DEST1REG1	DEST0REG2	DEST0REG1
R/WP-0	R/WP-0	R/WP-0	R/WP-0	R/WP-0	R/WP-0	R/WP-0	R/WP-0

7	6	5	4	3	2	1	0
BUSERROR	BUFF_OVF	SRC_OVF	DEST3_ERR	DEST2_ERR	DEST1_ERR	DEST0_ERR	PACKET_ ERR_INT
R/WP-0	R/WP-0	R/WP-0	R/WP-0	R/WP-0	R/WP-0	R/WP-0	R/WP-0

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privilege mode only; -*n* = value after reset

Table 36-10. DMM Interrupt Level Register (DMMINTLVL) Field Descriptions

Bit	Field	Value	Description
31-18	Reserved	0	Reads returns 0. Writes have no effect.
17	PROG_BUFF		**Programmable Buffer Interrupt Level**
			User and privilege mode read, privilege mode write:
		0	Interrupt mapped to level 0.
		1	Interrupt mapped to level 1.
16	EO_BUFF		**End of Buffer Interrupt Level**
			User and privilege mode read, privilege mode write:
		0	Interrupt mapped to level 0.
		1	Interrupt mapped to level 1.
15	DEST3REG2		**Destination 3 Region 2 Interrupt Level**
			User and privilege mode read, privilege mode write:
		0	Interrupt mapped to level 0.
		1	Interrupt mapped to level 1.
14	DEST3REG1		**Destination 3 Region 1 Interrupt Level**
			User and privilege mode read, privilege mode write:
		0	Interrupt mapped to level 0.
		1	Interrupt mapped to level 1.
13	DEST2REG2		**Destination 2 Region 2 Interrupt Level**
			User and privilege mode read, privilege mode write:
		0	Interrupt mapped to level 0.
		1	Interrupt mapped to level 1.
12	DEST2REG1		**Destination 2 Region 1 Interrupt Level**
			User and privilege mode read, privilege mode write:
		0	Interrupt mapped to level 0.
		1	Interrupt mapped to level 1.
11	DEST1REG2		**Destination 1 Region 2 Interrupt Level**
			User and privilege mode read, privilege mode write:
		0	Interrupt mapped to level 0.
		1	Interrupt mapped to level 1.

Table 36-10. DMM Interrupt Level Register (DMMINTLVL) Field Descriptions (continued)

Bit	Field	Value	Description
10	DEST1REG1		**Destination 1 Region 1 Interrupt Level**
			User and privilege mode read, privilege mode write:
		0	Interrupt mapped to level 0.
		1	Interrupt mapped to level 1.
9	DEST0REG2		**Destination 0 Region 2 Interrupt Level**
			User and privilege mode read, privilege mode write:
		0	Interrupt mapped to level 0.
		1	Interrupt mapped to level 1.
8	DEST0REG1		**Destination 0 Region 1 Interrupt Level**
			User and privilege mode read, privilege mode write:
		0	Interrupt mapped to level 0.
		1	Interrupt mapped to level 1.
7	BUSERROR		**BMM Bus Error Response**
			User and privilege mode read, privilege mode write:
		0	Interrupt mapped to level 0.
		1	Interrupt mapped to level 1.
6	BUFF_OVF		**Write Buffer Overflow Interrupt Level**
			User and privilege mode read, privilege mode write:
		0	Interrupt mapped to level 0.
		1	Interrupt mapped to level 1.
5	SRC_OVF		**Source Overflow Interrupt Level**
			User and privilege mode read, privilege mode write:
		0	Interrupt mapped to level 0.
		1	Interrupt mapped to level 1.
4	DEST3_ERR		**Destination 3 Error Interrupt Level**
			User and privilege mode read, privilege mode write:
		0	Interrupt mapped to level 0.
		1	Interrupt mapped to level 1.
3	DEST2_ERR		**Destination 2 Error Interrupt Level**
			User and privilege mode read, privilege mode write:
		0	Interrupt mapped to level 0.
		1	Interrupt mapped to level 1.
2	DEST1_ERR		**Destination 1 Error Interrupt Level**
			User and privilege mode read, privilege mode write:
		0	Interrupt mapped to level 0.
		1	Interrupt mapped to level 1.
1	DEST0_ERR		**Destination 0 Error Interrupt Level**
			User and privilege mode read, privilege mode write:
		0	Interrupt mapped to level 0.
		1	Interrupt mapped to level 1.
0	PACKET_ERR_INT		**Packet Error Interrupt Level**
			User and privilege mode read, privilege mode write:
		0	Interrupt mapped to level 0.
		1	Interrupt mapped to level 1.

36.3.5 DMM Interrupt Flag Register (DMMINTFLG)

This register contains the interrupt level bits for error interrupts and normal interrupts.

Figure 36-11. DMM Interrupt Flag Register (DMMINTFLG) [offset = 10h]

31-24
Reserved
R-0

23-18	17	16
Reserved	PROG_BUFF	EO_BUFF
R-0	R/WPC-0	R/WPC-0

15	14	13	12	11	10	9	8
DEST3REG2	DEST3REG1	DEST2REG2	DEST2REG1	DEST1REG2	DEST1REG1	DEST0REG2	DEST0REG1
R/WPC-0	R/WPC-0	R/WPC-0	R/WPC-0	R/WPC-0	R/WPC-0	R/WPC-0	R/WPC-0

7	6	5	4	3	2	1	0
BUSERROR	BUFF_OVF	SRC_OVF	DEST3_ERR	DEST2_ERR	DEST1_ERR	DEST0_ERR	PACKET_ ERR_INT
R/WPC-0	R/WPC-0	R/WPC-0	R/WPC-0	R/WPC-0	R/WPC-0	R/WPC-0	R/WPC-0

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privilege mode only; C = Clear; -*n* = value after reset

Table 36-11. DMM Interrupt Flag Register (DMMINTFLG) Field Descriptions

Bit	Field	Value	Description
31-18	Reserved	0	Reads returns 0. Writes have no effect.
17	PROG_BUFF		**Programmable Buffer Interrupt Flag**
			User and privilege mode (read):
		0	No interrupt occurred.
		1	Interrupt occurred.
			Privilege mode (write):
		0	No influence on bit.
		1	Bit will be cleared.
16	EO_BUFF		**End of Buffer Interrupt Flag**
			User and privilege mode (read):
		0	No interrupt occurred.
		1	Interrupt occurred.
			Privilege mode (write):
		0	No influence on bit.
		1	Bit will be cleared.
15	DEST3REG2		**Destination 3 Region 2 Interrupt Flag**
			User and privilege mode (read):
		0	No interrupt occurred.
		1	Interrupt occurred.
			Privilege mode (write):
		0	No influence on bit.
		1	Bit will be cleared.
14	DEST3REG1		**Destination 3 Region 1 Interrupt Flag**
			User and privilege mode (read):
		0	No interrupt occurred.
		1	Interrupt occurred.
			Privilege mode (write):
		0	No influence on bit.
		1	Bit will be cleared.

Table 36-11. DMM Interrupt Flag Register (DMMINTFLG) Field Descriptions (continued)

Bit	Field	Value	Description
13	DEST2REG2		**Destination 2 Region 2 Interrupt Flag**
			User and privilege mode (read):
		0	No interrupt occurred.
		1	Interrupt occurred.
			Privilege mode (write):
		0	No influence on bit.
		1	Bit will be cleared.
12	DEST2REG1		**Destination 2 Region 1 Interrupt Flag**
			User and privilege mode (read):
		0	No interrupt occurred.
		1	Interrupt occurred.
			Privilege mode (write):
		0	No influence on bit.
		1	Bit will be cleared.
11	DEST1REG2		**Destination 1 Region 2 Interrupt Flag**
			User and privilege mode (read):
		0	No interrupt occurred.
		1	Interrupt occurred.
			Privilege mode (write):
		0	No influence on bit.
		1	Bit will be cleared.
10	DEST1REG1		**Destination 1 Region 1 Interrupt Flag**
			User and privilege mode (read):
		0	No interrupt occurred.
		1	Interrupt occurred.
			Privilege mode (write):
		0	No influence on bit.
		1	Bit will be cleared.
9	DEST0REG2		**Destination 0 Region 2 Interrupt Flag**
			User and privilege mode (read):
		0	No interrupt occurred.
		1	Interrupt occurred.
			Privilege mode (write):
		0	No influence on bit.
		1	Bit will be cleared.
8	DEST0REG1		**Destination 0 Region 1 Interrupt Flag**
			User and privilege mode (read):
		0	No interrupt occurred.
		1	Interrupt occurred.
			Privilege mode (write):
		0	No influence on bit.
		1	Bit will be cleared.

Table 36-11. DMM Interrupt Flag Register (DMMINTFLG) Field Descriptions (continued)

Bit	Field	Value	Description
7	BUSERROR		**BMM Bus Error Response.**
			User and privilege mode (read):
		0	No interrupt occurred.
		1	Interrupt occurred.
			Privilege mode (write):
		0	No influence on bit.
		1	Bit will be cleared.
6	BUFF_OVF		**Write Buffer Overflow Interrupt Flag**
			User and privilege mode (read):
		0	No interrupt occurred.
		1	Interrupt occurred.
			Privilege mode (write):
		0	No influence on bit.
		1	Bit will be cleared.
5	SRC_OVF		**Source Overflow Interrupt Flag**
			User and privilege mode (read):
		0	No interrupt occurred.
		1	Interrupt occurred.
			Privilege mode (write):
		0	No influence on bit.
		1	Bit will be cleared.
4	DEST3_ERR		**Destination 3 Error Interrupt Flag**
			User and privilege mode (read):
		0	No interrupt occurred.
		1	Interrupt occurred.
			Privilege mode (write):
		0	No influence on bit.
		1	Bit will be cleared.
3	DEST2_ERR		**Destination 2 Error Interrupt Flag**
			User and privilege mode (read):
		0	No interrupt occurred.
		1	Interrupt occurred.
			Privilege mode (write):
		0	No influence on bit.
		1	Bit will be cleared.
2	DEST1_ERR		**Destination 1 Error Interrupt Flag**
			User and privilege mode (read):
		0	No interrupt occurred.
		1	Interrupt occurred.
			Privilege mode (write):
		0	No influence on bit.
		1	Bit will be cleared.

Table 36-11. DMM Interrupt Flag Register (DMMINTFLG) Field Descriptions (continued)

Bit	Field	Value	Description
1	DEST0_ERR		**Destination 0 Error Interrupt Flag**
			User and privilege mode (read):
		0	No interrupt occurred.
		1	Interrupt occurred.
			Privilege mode (write):
		0	No influence on bit.
		1	Bit will be cleared.
0	PACKET_ERR_INT		**Packet Error Interrupt Flag**
			User and privilege mode (read):
		0	No interrupt occurred.
		1	Interrupt occurred.
			Privilege mode (write):
		0	No influence on bit.
		1	Bit will be cleared.

36.3.6 DMM Interrupt Offset 1 Register (DMMOFF1)

This register holds the offset indicating which interrupt occurred on interrupt level 0. The CPU can read this register to determine the source of the interrupt without having to test individual interrupt flags.

Figure 36-12. DMM Interrupt Offset 1 Register (DMMOFF1) [offset = 14h]

31–16
Reserved
R-0

15–5	4–0
Reserved	OFFSET
R-0	R-0

LEGEND: R = Read only; -*n* = value after reset

Table 36-12. DMM Interrupt Offset 1 Register (DMMOFF1) Field Descriptions

Bit	Field	Value	Description
31-5	Reserved	0	Read returns 0. Writes have no effect.
4-0	OFFSET		User and privilege mode (read):
		Bit Encoding	Interrupt
		0	Phantom. All interrupt flags have been cleared before the offset register has been read.
		1h	Packet Error
		2h	Destination 0 Error
		3h	Destination 1 Error
		4h	Destination 2 Error
		5h	Destination 3 Error
		6h	Source Overflow
		7h	Buffer Overflow
		8h	Bus Error
		9h	Destination 0 Region 1
		Ah	Destination 0 Region 2
		Bh	Destination 1 Region 1
		Ch	Destination 1 Region 2
		Dh	Destination 2 Region 1
		Eh	Destination 2 Region 2
		Fh	Destination 3 Region 1
		10h	Destination 3 Region 2
		11h	End of Buffer
		12h	Programmable Buffer
		13h-1Fh	Reserved
			Reading the offset will clear the corresponding flag in DMMINTFLG (Section 36.3.5).
			Privilege and user mode writes have no effect

36.3.7 DMM Interrupt Offset 2 Register (DMMOFF2)

This register holds the offset indicating which interrupt occurred on interrupt level 1. The CPU can read this register to determine the source of the interrupt without having to test individual interrupt flags.

Figure 36-13. DMM Interrupt Offset 2 Register (DMMOFF2) [offset = 18h]

31	16
Reserved	
R-0	

15	5	4	0
Reserved		OFFSET	
R-0		R-0	

LEGEND: R = Read only; -*n* = value after reset

Table 36-13. DMM Interrupt Offset 2 Register (DMMOFF1) Field Descriptions

Bit	Field	Value	Description
31-5	Reserved	0	Read returns 0. Writes have no effect.
4-0	OFFSET		User and privilege mode (read):
		Bit Encoding	Interrupt
		0	Phantom. All interrupt flags have been cleared before the offset register has been read.
		1h	Packet Error
		2h	Destination 0 Error
		3h	Destination 1 Error
		4h	Destination 2 Error
		5h	Destination 3 Error
		6h	Source Overflow
		7h	Buffer Overflow
		8h	Bus Error
		9h	Destination 0 Region 1
		Ah	Destination 0 Region 2
		Bh	Destination 1 Region 1
		Ch	Destination 1 Region 2
		Dh	Destination 2 Region 1
		Eh	Destination 2 Region 2
		Fh	Destination 3 Region 1
		10h	Destination 3 Region 2
		11h	End of Buffer
		12h	Programmable Buffer
		13h-1Fh	Reserved
			Reading the offset will clear the corresponding flag in DMMINTFLG (Section 36.3.5).
			Privilege and user mode writes have no effect

36.3.8 DMM Direct Data Mode Destination Register (DMMDDMDEST)

This register defines the starting address of the buffer used to store the received data in Direct Data Mode. By writing to this register, the DMMDDMPT register (Section 36.3.10) will be set to 0x0000.

Figure 36-14. DMM Direct Data Mode Destination Register (DMMDDMDEST) [offset = 1Ch]

31 ... 0
STARTADDR
R/WP-0

LEGEND: R/W = Read/Write; WP = Write in privilege mode only; -*n* = value after reset

Table 36-14. DMM Direct Data Mode Destination Register (DMMDDMDEST) Field Descriptions

Bit	Field	Description
31-0	STARTADDR	These bits define the starting address of the buffer. The starting address has to be a multiple of the blocksize chosen in DMMDDMBL (Section 36.3.9). User and privilege mode (read): current start address Privilege mode (write): sets start address to value written

36.3.9 DMM Direct Data Mode Blocksize Register (DMMDDMBL)

This register defines the blocksize of the buffer used to store the received data in Direct Data Mode.

Figure 36-15. DMM Direct Data Mode Blocksize Register (DMMDDMBL) [offset = 20h]

31 ... 16
Reserved
R-0

15 ... 4	3 ... 0
Reserved	BLOCKSIZE
R-0	R/WP-0

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privilege mode only; -*n* = value after reset

Table 36-15. DMM Direct Data Mode Blocksize Register (DMMDDMBL) Field Descriptions

Bit	Field	Value	Description
31-4	Reserved	0	Read returns 0. Writes have no effect.
3-0	BLOCKSIZE		These bits define the size of the buffer region
			User and privilege mode (read): current block size
			Privilege mode (write):
		0	Buffer disabled. No data will be stored.
		1h	32 Byte
		2h	64 Byte
		3h	128 Byte
		4h	256 Byte
		5h	512 Byte
		6h	1 KByte
		7h	2 KByte
		8h	4 KByte
		9h	8 KByte
		Ah	16 KByte
		Bh	32 KByte
		Ch-Fh	Reserved

36.3.10 DMM Direct Data Mode Pointer Register (DMMDDMPT)

This register shows the pointer into the buffer programmed by DMMDDMDEST (Section 36.3.8) and DMMDDMBL (Section 36.3.9).

Figure 36-16. DMM Direct Data Mode Pointer Register (DMMDDMPT) [offset = 24h]

31	16
Reserved	
R-0	

15	14	0
Rsvd	POINTER	
R-0	R-0	

LEGEND: R = Read only; -*n* = value after reset

Table 36-16. DMM Direct Data Mode Pointer Register (DMMDDMPT) Field Descriptions

Bit	Field	Value	Description
31-15	Reserved	0	Read returns 0. Writes have no effect.
14-0	POINTER		These bits hold the pointer to the next entry to be written in the buffer. The pointer points to the byte aligned address. If in 16-bit DDM mode, bit 0 will be 0. If in 32-bit DDM mode, bit 0 and 1 will be 0. User and privilege mode (read): next data entry Privilege mode (write): writes have no effect

36.3.11 DMM Direct Data Mode Interrupt Pointer Register (DMMINTPT)

This register can be programmed to hold a threshold to which the DMMDDMPT register (Section 36.3.10) is compared. An interrupt can be generated when both match.

Figure 36-17. DMM Direct Data Mode Interrupt Pointer Register (DMMINTPT) [offset = 28h]

31	16
Reserved	
R-0	

15	14	0
Rsvd	INTPT	
R-0	R/WP-0	

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privilege mode only; -*n* = value after reset

Table 36-17. DMM Direct Data Mode Interrupt Pointer Register (DMMINTPT) Field Descriptions

Bit	Field	Value	Description
31-15	Reserved	0	Read returns 0. Writes have no effect.
14-0	INTPT		Interrupt Pointer. When the buffer pointer (Section 36.3.10) matches the programmed value in DMMINTPT and the PROG_BUF interrupt (Section 36.3.2) is set, an interrupt is generated. User and privilege mode (read): current interrupt threshold Privilege mode (write): new interrupt threshold

36.3.12 DMM Destination x Region 1 (DMMDESTxREG1)

This register defines the starting address of the buffer used to store the received data in Trace Mode. If the received data does not fall into the address range defined by DMMDESTxREG1 and DMMDESTxBL1, an interrupt (DESTx_ERR) can be generated. The description below is valid for following registers: DMMDEST0REG1, DMMDEST1REG1, DMMDEST2REG1, DMMDEST3REG1.

Figure 36-18. DMM Destination x Region 1 (DMMDESTxREG1) [offset = 2Ch, 3Ch, 4Ch, 5Ch]

31	18	17	16
BASEADDR		BLOCKADDR	
R/WP-0		R/WP-0	

15	0
BLOCKADDR	
R/WP-0	

LEGEND: R/W = Read/Write; WP = Write in privilege mode only; -*n* = value after reset

Table 36-18. DMM Destination x Region 1 (DMMDESTxREG1) Field Descriptions

Bit	Field	Description
31-18	BASEADDR	These bits define the base address of the 256kB region where the buffer is located. User and privilege mode (read): current start address Privilege mode (write): sets base address to value written
17-0	BLOCKADDR	These bits define the starting address of the buffer in the 256kB page. The starting address has to be a multiple of the blocksize chosen in DMMDESTxBL1 (Section 36.3.13). User and privilege mode (read): current start address Privilege mode (write): sets start address to value written

36.3.13 DMM Destination x Blocksize 1 (DMMDESTxBL1)

This register defines the blocksize of the buffer used to store the received data in Trace Mode. If the received data does not fall into the address range defined by DMMDESTxREG1 and DMMDESTxBL1, an interrupt (DESTx_ERR) can be generated. The description below is valid for following registers: DMMDEST0BL1, DMMDEST1BL1, DMMDEST2BL1, DMMDEST3BL1.

Figure 36-19. DMM Destination x Blocksize 1 (DMMDESTxBL1) [offset = 30h, 40h, 50h, 60h]

31-16
Reserved
R-0

15-4	3-0
Reserved	BLOCKSIZE
R-0	R/WP-0

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privilege mode only; -*n* = value after reset

Table 36-19. DMM Destination x Blocksize 1 (DMMDESTxBL1) Field Descriptions

Bit	Field	Value	Description
31-4	Reserved	0	Read returns 0. Writes have no effect.
3-0	BLOCKSIZE		These bits define the length of the buffer region. If all bits are 0, the region is disabled and no data will be stored.
			User and privilege mode (read): current block size
			Privilege mode (write):
		0	Region disabled
		1h	1 KByte
		2h	2 KByte
		3h	4 KByte
		4h	8 KByte
		5h	16 KByte
		6h	32 KByte
		7h	64 KByte
		8h	128 KByte
		9h	256 KByte
		Ah-Fh	Reserved

36.3.14 DMM Destination x Region 2 (DMMDESTxREG2)

This register defines the starting address of the buffer used to store the received data in Trace Mode. If the received data does not fall into the address range defined by DMMDESTxREG2 and DMMDESTxBL2, an interrupt (DESTx_ERR) can be generated. The description below is valid for following registers: DMMDEST0REG2, DMMDEST1REG2, DMMDEST2REG2, DMMDEST3REG2.

Figure 36-20. DMM Destination x Region 2 (DMMDESTxREG2) [offset = 34h, 44h, 54h, 64h]

31	18	17	16
BASEADDR		BLOCKADDR	
R/WP-0		R/WP-0	

15	0
BLOCKADDR	
R/WP-0	

LEGEND: R/W = Read/Write; WP = Write in privilege mode only; -*n* = value after reset

Table 36-20. DMM Destination x Region 2 (DMMDESTxREG2) Field Descriptions

Bit	Field	Description
31-18	BASEADDR	These bits define the base address of the 256kB region where the buffer is located. User and privilege mode (read): current start address Privilege mode (write): sets base address to value written
17-0	BLOCKADDR	These bits define the starting address of the buffer in the 256kB page. The starting address has to be a multiple of the blocksize chosen in DMMDESTxBL1 (Section 36.3.15). User and privilege mode (read): current start address Privilege mode (write): sets start address to value written

36.3.15 DMM Destination x Blocksize 2 (DMMDESTxBL2)

This register defines the blocksize of the buffer used to store the received data in Trace Mode. If the received data does not fall into the address range defined by DMMDESTxREG2 and DMMDESTxBL2, an interrupt (DESTx_ERR) can be generated. The description below is valid for following registers: DMMDEST0BL2, DMMDEST1BL2, DMMDEST2BL2, DMMDEST3BL2.

Figure 36-21. DMM Destination x Blocksize 2 (DMMDESTxBL2) [offset = 38h, 48h, 58h, 68h]

31-16
Reserved
R-0

15-4	3-0
Reserved	BLOCKSIZE
R-0	R/WP-0

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privilege mode only; -*n* = value after reset

Table 36-21. DMM Destination x Blocksize 2 (DMMDESTxBL2) Field Descriptions

Bit	Field	Value	Description
31-4	Reserved	0	Read returns 0. Writes have no effect.
3-0	BLOCKSIZE		These bits define the length of the buffer region. If all bits are 0, the region is disabled and no data will be stored.
			User and privilege mode (read): current block size
			Privilege mode (write):
		0	Region disabled
		1h	1 KByte
		2h	2 KByte
		3h	4 KByte
		4h	8 KByte
		5h	16 KByte
		6h	32 KByte
		7h	64 KByte
		8h	128 KByte
		9h	256 KByte
		Ah-Fh	Reserved

36.3.16 DMM Pin Control 0 (DMMPC0)

This register defines if the DMM pins are used in functional or GIO mode. It should only be written when ON/OFF = 0101 and the BUSY bit = 0 (Section 36.3.1). If pins other than the pins specified in Table 36-5 are configured, or DMMCLK and DMMSYNC are programmed as non-functional pins, no operation in trace mode or direct data mode is possible.

Figure 36-22. DMM Pin Control 0 (DMMPC0) [offset = 6Ch]

31	24
Reserved	
R-0	

23	19	18	17	16
Reserved		ENAFUNC	DATA15FUNC	DATA14FUNC
R-0		R/WP-0	R/WP-0	R/WP-0

15	14	13	12	11	10	9	8
DATA13FUNC	DATA12FUNC	DATA11FUNC	DATA10FUNC	DATA9FUNC	DATA8FUNC	DATA7FUNC	DATA6FUNC
R/WP-0	R/WP-0	R/WP-0	R/WP-0	R/WP-0	R/WP-0	R/WP-0	R/WP-0

7	6	5	4	3	2	1	0
DATA5FUNC	DATA4FUNC	DATA3FUNC	DATA2FUNC	DATA1FUNC	DATA0FUNC	CLKFUNC	SYNCFUNC
R/WP-0	R/WP-0	R/WP-0	R/WP-0	R/WP-0	R/WP-0	R/WP-0	R/WP-0

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privileged mode only; -*n* = value after reset

Table 36-22. DMM Pin Control 0 (DMMPC0) Field Descriptions

Bit	Field	Value	Description
31-19	Reserved	0	Reads returns 0. Writes have no effect.
18	ENAFUNC		**Functional mode of $\overline{\text{DMMENA}}$ pin.** This bit defines whether the pin is used in functional mode or in GIO mode.
			User and privilege mode (read):
		0	Pin is used in GIO mode.
		1	Pin is used in Functional mode.
			Privilege mode (write):
		0	Pin is used in GIO mode.
		1	Pin is used in Functional mode.
17-2	DATAxFUNC		**Functional mode of DMMDATA[x] pin.** This bit defines whether the pin is used in functional mode or in GIO mode. If pins are configured in functional mode, only pins defined in Table 36-5 have to be used for proper operation.
			User and privilege mode (read):
		0	Pin is used in GIO mode.
		1	Pin is used in Functional mode.
			Privilege mode (write):
		0	Pin is used in GIO mode.
		1	Pin is used in Functional mode.
1	CLKFUNC		**Functional mode of DMMCLK pin.** This bit defines whether the pin is used in functional mode or in GIO mode.
			User and privilege mode (read):
		0	Pin is used in GIO mode.
		1	Pin is used in Functional mode.
			Privilege mode (write):
		0	Pin is used in GIO mode.
		1	Pin is used in Functional mode.

Table 36-22. DMM Pin Control 0 (DMMPC0) Field Descriptions (continued)

Bit	Field	Value	Description
0	SYNCFUNC		**Functional mode of DMMSYNC pin.** This bit defines whether the pin is used in functional mode or in GIO mode.
			User and privilege mode (read):
		0	Pin is used in GIO mode.
		1	Pin is used in Functional mode.
			Privilege mode (write):
		0	Pin is used in GIO mode.
		1	Pin is used in Functional mode.

36.3.17 DMM Pin Control 1 (DMMPC1)

The bits in this register define the direction of the individual module pins when in GIO mode.

Figure 36-23. DMM Pin Control 1 (DMMPC1) [offset = 70h]

31-24
Reserved
R-0

23-19	18	17	16
Reserved	ENADIR	DATA15DIR	DATA14DIR
R-0	R/WP-0	R/WP-0	R/WP-0

15	14	13	12	11	10	9	8
DATA13DIR	DATA12DIR	DATA11DIR	DATA10DIR	DATA9DIR	DATA8DIR	DATA7DIR	DATA6DIR
R/WP-0	R/WP-0	R/WP-0	R/WP-0	R/WP-0	R/WP-0	R/WP-0	R/WP-0

7	6	5	4	3	2	1	0
DATA5DIR	DATA4DIR	DATA3DIR	DATA2DIR	DATA1DIR	DATA0DIR	CLKDIR	SYNCDIR
R/WP-0	R/WP-0	R/WP-0	R/WP-0	R/WP-0	R/WP-0	R/WP-0	R/WP-0

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privileged mode only; -*n* = value after reset

Table 36-23. DMM Pin Control 1 (DMMPC1) Field Descriptions

Bit	Field	Value	Description
31-19	Reserved	0	Reads returns 0. Writes have no effect.
18	ENADIR		**Direction of $\overline{\text{DMMENA}}$ pin**.
			User and privilege mode (read):
		0	Pin is used as input.
		1	Pin is used as output.
			Privilege mode (write):
		0	Pin is set to input.
		1	Pin is set to output.
17-2	DATAxDIR		**Direction of DMMDATA[x] pin.** This bit defines whether the pin is used as input or output in GIO mode.
			User and privilege mode (read):
		0	Pin is used as input.
		1	Pin is used as output.
			Privilege mode (write):
		0	Pin is set to input.
		1	Pin is set to output.

Table 36-23. DMM Pin Control 1 (DMMPC1) Field Descriptions (continued)

Bit	Field	Value	Description
1	CLKDIR		**Direction of DMMCLK pin.** This bit defines whether the pin is used as input or output in GIO mode.
			User and privilege mode (read):
		0	Pin is used as input.
		1	Pin is used as output.
			Privilege mode (write):
		0	Pin is set to input.
		1	Pin is set to output.
0	SYNCDIR		**Direction of DMMSYNC pin.** This bit defines whether the pin is used as input or output in GIO mode.
			User and privilege mode (read):
		0	Pin is used as input.
		1	Pin is used as output.
			Privilege mode (write):
		0	Pin is set to input.
		1	Pin is set to output.

36.3.18 DMM Pin Control 2 (DMMPC2)

The bits in this register reflect the digital representation of the voltage level at the module pins. Even if a pin is configured to be an output pin, the level can be read back via this register.

Figure 36-24. DMM Pin Control 2 (DMMPC2) [offset = 74h]

31							24
Reserved							
R-0							

23				19	18	17	16
Reserved					ENAIN	DATA15IN	DATA14IN
R-0					R/WP-0	R/WP-0	R/WP-0

15	14	13	12	11	10	9	8
DATA13IN	DATA12IN	DATA11IN	DATA10IN	DATA9IN	DATA8IN	DATA7IN	DATA6IN
R/WP-0	R/WP-0	R/WP-0	R/WP-0	R/WP-0	R/WP-0	R/WP-0	R/WP-0

7	6	5	4	3	2	1	0
DATA5IN	DATA4IN	DATA3IN	DATA2IN	DATA1IN	DATA0IN	CLKIN	SYNCIN
R/WP-0	R/WP-0	R/WP-0	R/WP-0	R/WP-0	R/WP-0	R/WP-0	R/WP-0

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privileged mode only; -*n* = value after reset

Table 36-24. DMM Pin Control 2 (DMMPC2) Field Descriptions

Bit	Field	Value	Description
31-19	Reserved	0	Reads returns 0. Writes have no effect.
18	ENAIN		**$\overline{\text{DMMENA}}$ input.** This bit reflects the state of the pin in all modes.
			User and privilege mode (read):
		0	Logic low (input voltage is V_{IL} or lower).
		1	Logic high (input voltage is V_{IH} or higher).
			Privilege mode (write): writes to this bit have no effect.
17-2	DATAxIN		**DMMDATA[x] input.** This bit reflects the state of the pin in all modes.
			User and privilege mode (read):
		0	Logic low (input voltage is V_{IL} or lower).
		1	Logic high (input voltage is V_{IH} or higher).
			Privilege mode (write): writes to this bit have no effect.
1	CLKIN		**DMMCLK input.** This bit reflects the state of the pin in all modes.
			User and privilege mode (read):
		0	Logic low (input voltage is V_{IL} or lower).
		1	Logic high (input voltage is V_{IH} or higher).
			Privilege mode (write): writes to this bit have no effect.
0	SYNCIN		**DMMSYNC input.** This bit reflects the state of the pin in all modes.
			User and privilege mode (read):
		0	Logic low (input voltage is V_{IL} or lower).
		1	Logic high (input voltage is V_{IH} or higher).
			Privilege mode (write): writes to this bit have no effect.

36.3.19 DMM Pin Control 3 (DMMPC3)

The bits in this register set the pin to logic low or high level if the pin is configured as output (Section 36.3.17).

Figure 36-25. DMM Pin Control 3 (DMMPC3) [offset = 78h]

31							24
Reserved							
R-0							

23				19	18	17	16
Reserved					ENAOUT	DATA15OUT	DATA14OUT
R-0					R/WP-0	R/WP-0	R/WP-0

15	14	13	12	11	10	9	8
DATA13OUT	DATA12OUT	DATA11OUT	DATA10OUT	DATA9OUT	DATA8OUT	DATA7OUT	DATA6OUT
R/WP-0	R/WP-0	R/WP-0	R/WP-0	R/WP-0	R/WP-0	R/WP-0	R/WP-0

7	6	5	4	3	2	1	0
DATA5OUT	DATA4OUT	DATA3OUT	DATA2OUT	DATA1OUT	DATA0OUT	CLKOUT	SYNCOUT
R/WP-0	R/WP-0	R/WP-0	R/WP-0	R/WP-0	R/WP-0	R/WP-0	R/WP-0

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privileged mode only; -*n* = value after reset

Table 36-25. DMM Pin Control 3 (DMMPC3) Field Descriptions

Bit	Field	Value	Description
31-19	Reserved	0	Reads returns 0. Writes have no effect.
18	ENAOUT		**Output state of $\overline{\text{DMMENA}}$ pin.** This bit sets the pin to logic low or high level.
			User and privilege mode (read):
		0	Logic low (output voltage is V_{OL} or lower).
		1	Logic high (output voltage is V_{OH} or higher).
			Privilege mode (write):
		0	Logic low (output voltage is set to V_{OL} or lower).
		1	Logic high (output voltage is set to V_{OH} or higher).
17-2	DATAxOUT		**Output state of DMMDATA[x] pin.** This bit sets the pin to logic low or high level.
			User and privilege mode (read):
		0	Logic low (output voltage is V_{OL} or lower).
		1	Logic high (output voltage is V_{OH} or higher).
			Privilege mode (write):
		0	Logic low (output voltage is set to V_{OL} or lower).
		1	Logic high (output voltage is set to V_{OH} or higher).
1	CLKOUT		**Output state of DMMCLK pin.** This bit sets the pin to logic low or high level.
			User and privilege mode (read):
		0	Logic low (output voltage is V_{OL} or lower).
		1	Logic high (output voltage is V_{OH} or higher).
			Privilege mode (write):
		0	Logic low (output voltage is set to V_{OL} or lower).
		1	Logic high (output voltage is set to V_{OH} or higher).

Table 36-25. DMM Pin Control 3 (DMMPC3) Field Descriptions (continued)

Bit	Field	Value	Description
0	SYNCOUT		**Output state of DMMSYNC pin.** This bit sets the pin to logic low or high level.
			User and privilege mode (read):
		0	Logic low (output voltage is V_{OL} or lower).
		1	Logic high (output voltage is V_{OH} or higher).
			Privilege mode (write):
		0	Logic low (output voltage is set to V_{OL} or lower).
		1	Logic high (output voltage is set to V_{OH} or higher).

36.3.20 DMM Pin Control 4 (DMMPC4)

This register allows to set individual pins to a logic high level without having to do a read-modify-write operation as would be the case with the DMMPC3 register (Section 36.3.19). Writing a zero to a bit will not change the state of the pin.

Figure 36-26. DMM Pin Control 4 (DMMPC4) [offset = 7Ch]

31-24
Reserved
R-0

23-19	18	17	16
Reserved	ENASET	DATA15SET	DATA14SET
R-0	R/WP-0	R/WP-0	R/WP-0

15	14	13	12	11	10	9	8
DATA13SET	DATA12SET	DATA11SET	DATA10SET	DATA9SET	DATA8SET	DATA7SET	DATA6SET
R/WP-0	R/WP-0	R/WP-0	R/WP-0	R/WP-0	R/WP-0	R/WP-0	R/WP-0

7	6	5	4	3	2	1	0
DATA5SET	DATA4SET	DATA3SET	DATA2SET	DATA1SET	DATA0SET	CLKSET	SYNCSET
R/WP-0	R/WP-0	R/WP-0	R/WP-0	R/WP-0	R/WP-0	R/WP-0	R/WP-0

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privileged mode only; -*n* = value after reset

Table 36-26. DMM Pin Control 4 (DMMPC4) Field Descriptions

Bit	Field	Value	Description
31-19	Reserved	0	Reads returns 0. Writes have no effect.
18	ENASET		**Sets output state of $\overline{\text{DMMENA}}$ pin to logic high.** Value in the ENASET bit sets the data output control register bit to 1 regardless of the current value in the ENAOUT bit.
			User and privilege mode (read):
		0	Logic low (output voltage is V_{OL} or lower).
		1	Logic high (output voltage is V_{OH} or higher).
			Privilege mode (write):
		0	State of the pin is unchanged.
		1	Logic high (output voltage is set to V_{OH} or higher).
17-2	DATAxSET		**Sets output state of DMMDATA[x] pin to logic high.** Value in the DATAxSET bit sets the data output control register bit to 1 regardless of the current value in the DATAxOUT bit.
			User and privilege mode (read):
		0	Logic low (output voltage is V_{OL} or lower).
		1	Logic high (output voltage is V_{OH} or higher).
			Privilege mode (write):
		0	State of the pin is unchanged.
		1	Logic high (output voltage is set to V_{OH} or higher).

Table 36-26. DMM Pin Control 4 (DMMPC4) Field Descriptions (continued)

Bit	Field	Value	Description
1	CLKSET		**Sets output state of DMMCLK pin to logic high.** Value in the CLKSET bit sets the data output control register bit to 1 regardless of the current value in the CLKOUT bit.
			User and privilege mode (read):
		0	Logic low (output voltage is V_{OL} or lower).
		1	Logic high (output voltage is V_{OH} or higher).
			Privilege mode (write):
		0	State of the pin is unchanged.
		1	Logic high (output voltage is set to V_{OH} or higher).
0	SYNCSET		**Sets output state of DMMSYNC pin logic high.** Value in the SYNCSET bit sets the data output control register bit to 1 regardless of the current value in the SYNCOUT bit.
			User and privilege mode (read):
		0	Logic low (output voltage is V_{OL} or lower).
		1	Logic high (output voltage is V_{OH} or higher).
			Privilege mode (write):
		0	State of the pin is unchanged.
		1	Logic high (output voltage is set to V_{OH} or higher).

36.3.21 DMM Pin Control 5 (DMMPC5)

This register allows to set individual pins to a logic low level without having to do a read-modify-write operation as would be the case with the DMMPC3 register (Section 36.3.19). Writing a one to a bit will change the output to a logic low level, writing a zero will not change the state of the pin.

Figure 36-27. DMM Pin Control 5 (DMMPC5) [offset = 80h]

31							24
Reserved							
R-0							

23				19	18	17	16
Reserved					ENACLR	DATA15CLR	DATA14CLR
R-0					R/WP-0	R/WP-0	R/WP-0

15	14	13	12	11	10	9	8
DATA13CLR	DATA12CLR	DATA11CLR	DATA10CLR	DATA9CLR	DATA8CLR	DATA7CLR	DATA6CLR
R/WP-0	R/WP-0	R/WP-0	R/WP-0	R/WP-0	R/WP-0	R/WP-0	R/WP-0

7	6	5	4	3	2	1	0
DATA5CLR	DATA4CLR	DATA3CLR	DATA2CLR	DATA1CLR	DATA0CLR	CLKCLR	SYNCCLR
R/WP-0	R/WP-0	R/WP-0	R/WP-0	R/WP-0	R/WP-0	R/WP-0	R/WP-0

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privileged mode only; -*n* = value after reset

Table 36-27. DMM Pin Control 5 (DMMPC5) Field Descriptions

Bit	Field	Value	Description
31-19	Reserved	0	Reads returns 0. Writes have no effect.
18	ENACLR		**Sets output state of $\overline{\text{DMMENA}}$ pin to logic low.** Value in the ENACLR bit clears the data output control register bit to 0, regardless of the current value in the ENAOUT bit.
			User and privilege mode (read):
		0	Logic low (output voltage is V_{OL} or lower).
		1	Logic high (output voltage is V_{OH} or higher).
			Privilege mode (write):
		0	State of the pin is unchanged.
		1	Clears the pin to logic low (output voltage is set to V_{OL} or lower).
17-2	DATAxCLR		**Sets output state of DMMDATA[x] pin to logic low.** Value in the DATAxCLR bit clears the data output control register DATAxOUT bit to 0, regardless of the current value in the DATAxOUT bit.
			User and privilege mode (read):
		0	Logic low (output voltage is V_{OL} or lower).
		1	Logic high (output voltage is V_{OH} or higher).
			Privilege mode (write):
		0	State of the pin is unchanged.
		1	Clears the pin to logic low (output voltage is set to V_{OL} or lower).
1	CLKCLR		**Sets output state of DMMCLK pin to logic low.** Value in the CLKCLR bit clears the data output control register CLKOUT bit to 0, regardless of the current value in the CLKOUT bit.
			User and privilege mode (read):
		0	Logic low (output voltage is V_{OL} or lower).
		1	Logic high (output voltage is V_{OH} or higher).
			Privilege mode (write):
		0	State of the pin is unchanged.
		1	Clears the pin to logic low (output voltage is set to V_{OL} or lower).

Table 36-27. DMM Pin Control 5 (DMMPC5) Field Descriptions (continued)

Bit	Field	Value	Description
0	SYNCCLR		**Sets output state of DMMSYNC pin to logic low.** Value in the SYNCCLR bit clears the data output control register SYNCOUT bit to 0, regardless of the current value in the SYNCOUT bit.
			User and privilege mode (read):
		0	Logic low (output voltage is V_{OL} or lower).
		1	Logic high (output voltage is V_{OH} or higher).
			Privilege mode (write):
		0	State of the pin is unchanged.
		1	Clears the pin to logic low (output voltage is set to V_{OL} or lower).

36.3.22 DMM Pin Control 6 (DMMPC6)

These bits configure the pins in push-pull or open-drain functionality. If configured to be open-drain, the module only drives a logic-low level on the pin. An external pull-up resistor needs to be connected to the pin to pull it high, when the pin is in high-impedance mode.

Figure 36-28. DMM Pin Control 6 (DMMPC6) [offset = 84h]

31-24
Reserved
R-0

23-19	18	17	16
Reserved	ENAPDR	DATA15PDR	DATA14PDR
R-0	R/WP-0	R/WP-0	R/WP-0

15	14	13	12	11	10	9	8
DATA13PDR	DATA12PDR	DATA11PDR	DATA10PDR	DATA9PDR	DATA8PDR	DATA7PDR	DATA6PDR
R/WP-0	R/WP-0	R/WP-0	R/WP-0	R/WP-0	R/WP-0	R/WP-0	R/WP-0

7	6	5	4	3	2	1	0
DATA5PDR	DATA4PDR	DATA3PDR	DATA2PDR	DATA1PDR	DATA0PDR	CLKPDR	SYNCPDR
R/WP-0	R/WP-0	R/WP-0	R/WP-0	R/WP-0	R/WP-0	R/WP-0	R/WP-0

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privileged mode only; -*n* = value after reset

Table 36-28. DMM Pin Control 6 (DMMPC6) Field Descriptions

Bit	Field	Value	Description
31-19	Reserved	0	Reads returns 0. Writes have no effect.
18	ENAPDR		**Open Drain enable.** Enables open-drain functionality, if the pin is configured as GIO output (DMMPC0[18] = 0; DMMPC1[18] = 1). If the pin is configured as a functional pin (DMMPC0[18] = 1), the open-drain functionality is disabled.
			User and privilege mode (read):
		0	Pin behaves as normal push/pull pin.
		1	Pin operates in open-drain mode.
			Privilege mode (write):
		0	Configures pin as push/pull.
		1	Configures pin as open drain.

Table 36-28. DMM Pin Control 6 (DMMPC6) Field Descriptions (continued)

Bit	Field	Value	Description
17-2	DATAxPDR		**Open Drain enable.** Enables open-drain functionality on pin, if pin is configured as GIO output (DMMPC0[x] = 0; DMMPC1[x] = 1). If the pin is configured as a functional pin (DMMPC0[x] = 1), the open-drain functionality is disabled.
			User and privilege mode (read):
		0	Pin behaves as normal push/pull pin.
		1	Pin operates in open-drain mode.
			Privilege mode (write):
		0	Configures the pin as push/pull.
		1	Configures the pin as open drain.
1	CLKPDR		**Open Drain enable.** Enables open-drain functionality on pin, if pin is configured as GIO output (DMMPC0[1] = 0; DMMPC1[1] = 1). If the pin is configured as a functional pin (DMMPC0[1] = 1), the open-drain functionality is disabled.
			User and privilege mode (read):
		0	Pin behaves as normal push/pull pin.
		1	Pin operates in open-drain mode.
			Privilege mode (write):
		0	Configures the pin as push/pull.
		1	Configures the pin as open drain.
0	SYNCPDR		**Open Drain enable.** Enables open-drain functionality on pin, if pin is configured as GIO output (DMMPC0[0] = 0; DMMPC1[0] = 1). If the pin is configured as a functional pin (DMMPC0[0] = 1), the open-drain functionality is disabled.
			User and privilege mode (read):
		0	Pin behaves as normal push/pull pin.
		1	Pin operates in open-drain mode.
			Privilege mode (write):
		0	Configures the pin as push/pull.
		1	Configures the pin as open drain.

36.3.23 DMM Pin Control 7 (DMMPC7)

The bits in register control the pullup/down functionality of a pin. The internal pullup/down can be enabled or disabled by this register. The reset configuration of these bits is device implementation dependent. Please consult the device datasheet for this information.

Figure 36-29. DMM Pin Control 7 (DMMPC7) [offset = 88h]

31-24
Reserved
R-0

23-19	18	17	16
Reserved	ENAPDIS	DATA15PDIS	DATA14PDIS
R-0	R/WP-x	R/WP-x	R/WP-x

15	14	13	12	11	10	9	8
DATA13PDIS	DATA12PDIS	DATA11PDIS	DATA10PDIS	DATA9PDIS	DATA8PDIS	DATA7PDIS	DATA6PDIS
R/WP-x	R/WP-x	R/WP-x	R/WP-x	R/WP-x	R/WP-x	R/WP-x	R/WP-x

7	6	5	4	3	2	1	0
DATA5PDIS	DATA4PDIS	DATA3PDIS	DATA2PDIS	DATA1PDIS	DATA0PDIS	CLKPDIS	SYNCPDIS
R/WP-x	R/WP-x	R/WP-x	R/WP-x	R/WP-x	R/WP-x	R/WP-x	R/WP-x

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privileged mode only; -*n* = value after reset

Table 36-29. DMM Pin Control 7 (DMMPC7) Field Descriptions

Bit	Field	Value	Description
31-19	Reserved	0	Reads returns 0. Writes have no effect.
18	ENAPDIS		**Pull disable.** Removes internal pullup/pulldown functionality from pin when configured as input pin (DMMPC1[18] = 0).
			User and privilege mode (read):
		0	Pullup/pulldown functionality is enabled.
		1	Pullup/pulldown functionality is disabled.
			Privilege mode (write):
		0	Enables pullup/pulldown functionality.
		1	Disables pullup/pulldown functionality.
17-2	DATAxPDIS		**Pull disable.** Removes internal pullup/pulldown functionality from pin when configured as input pin (DMMPC1[x] = 0).
			User and privilege mode (read):
		0	Pullup/pulldown functionality is enabled.
		1	Pullup/pulldown functionality is disabled.
			Privilege mode (write):
		0	Enables pullup/pulldown functionality.
		1	Disables pullup/pulldown functionality.
1	CLKPDIS		**Pull disable.** Removes internal pullup/pulldown functionality from pin when configured as input pin (DMMPC1[1] = 0).
			User and privilege mode (read):
		0	Pullup/pulldown functionality is enabled.
		1	Pullup/pulldown functionality is disabled.
			Privilege mode (write):
		0	Enables pullup/pulldown functionality.
		1	Disables pullup/pulldown functionality.

Table 36-29. DMM Pin Control 7 (DMMPC7) Field Descriptions (continued)

Bit	Field	Value	Description
0	SYNCPDIS		**Pull disable.** Removes internal pullup/pulldown functionality from pin when configured as input pin (DMMPC1[0] = 0).
			User and privilege mode (read):
		0	Pullup/pulldown functionality is enabled.
		1	Pullup/pulldown functionality is disabled.
			Privilege mode (write):
		0	Enables pullup/pulldown functionality.
		1	Disables pullup/pulldown functionality.

36.3.24 DMM Pin Control 8 (DMMPC8)

These bits control if the internal pullup or pulldown is configured on the input pin.

Figure 36-30. DMM Pin Control 8 (DMMPC8) [offset = 8Ch]

31-24
Reserved
R-0

23-19	18	17	16
Reserved	ENAPSEL	DATA15PSEL	DATA14PSEL
R-0	R/WP-1	R/WP-1	R/WP-1

15	14	13	12	11	10	9	8
DATA13PSEL	DATA12PSEL	DATA11PSEL	DATA10PSEL	DATA9PSEL	DATA8PSEL	DATA7PSEL	DATA6PSEL
R/WP-1	R/WP-1	R/WP-1	R/WP-1	R/WP-1	R/WP-1	R/WP-1	R/WP-1

7	6	5	4	3	2	1	0
DATA5PSEL	DATA4PSEL	DATA3PSEL	DATA2PSEL	DATA1PSEL	DATA0PSEL	CLKPSEL	SYNCPSEL
R/WP-1	R/WP-1	R/WP-1	R/WP-1	R/WP-1	R/WP-1	R/WP-1	R/WP-1

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privileged mode only; -*n* = value after reset

Table 36-30. DMM Pin Control 8 (DMMPC8) Field Descriptions

Bit	Field	Value	Description
31-19	Reserved	0	Reads returns 0. Writes have no effect.
18	ENAPSEL		**Pull select.** Configures pullup or pulldown functionality if DMMPC7[18] = 0.
			User and privilege mode (read):
		0	Pulldown functionality is enabled.
		1	Pullup functionality is enabled.
			Privilege mode (write):
		0	Enables pulldown functionality.
		1	Enables pullup functionality.
17-2	DATAxPSEL		**Pull select.** Configures pullup or pulldown functionality if DMMPC7[x] = 0.
			User and privilege mode (read):
		0	Pulldown functionality is enabled.
		1	Pullup functionality is enabled.
			Privilege mode (write):
		0	Enables pulldown functionality.
		1	Enables pullup functionality.

Table 36-30. DMM Pin Control 8 (DMMPC8) Field Descriptions (continued)

Bit	Field	Value	Description
1	CLKPSEL		**Pull select.** Configures pullup or pulldown functionality if DMMPC7[1] = 0.
			User and privilege mode (read):
		0	Pulldown functionality is enabled.
		1	Pullup functionality is enabled.
			Privilege mode (write):
		0	Enables pulldown functionality.
		1	Enables pullup functionality.
0	SYNCPSEL		**Pull select.** Configures pullup or pulldown functionality if DMMPC7[0] = 0.
			User and privilege mode (read):
		0	Pulldown functionality is enabled.
		1	Pullup functionality is enabled.
			Privilege mode (write):
		0	Enables pulldown functionality.
		1	Enables pullup functionality.

Chapter 37
SPNU563A–March 2018

RAM Trace Port (RTP)

This chapter describes the functionality of the RAM trace port (RTP) module. It allows the capability to perform data trace of a CPU or other master accesses to the internal RAM and peripherals.

Topic **Page**

37.1 Overview

This document describes the functionality of the RAM trace port (RTP) module, which provides the features to datalog the RAM contents of the devices or accesses to peripherals without program intrusion. It can trace all data write or read accesses to internal RAM. In addition, it provides the capability to directly transfer data to a FIFO to support a CPU-controlled transmission of the data. The trace data is transmitted over a dedicated external interface.

37.1.1 Features

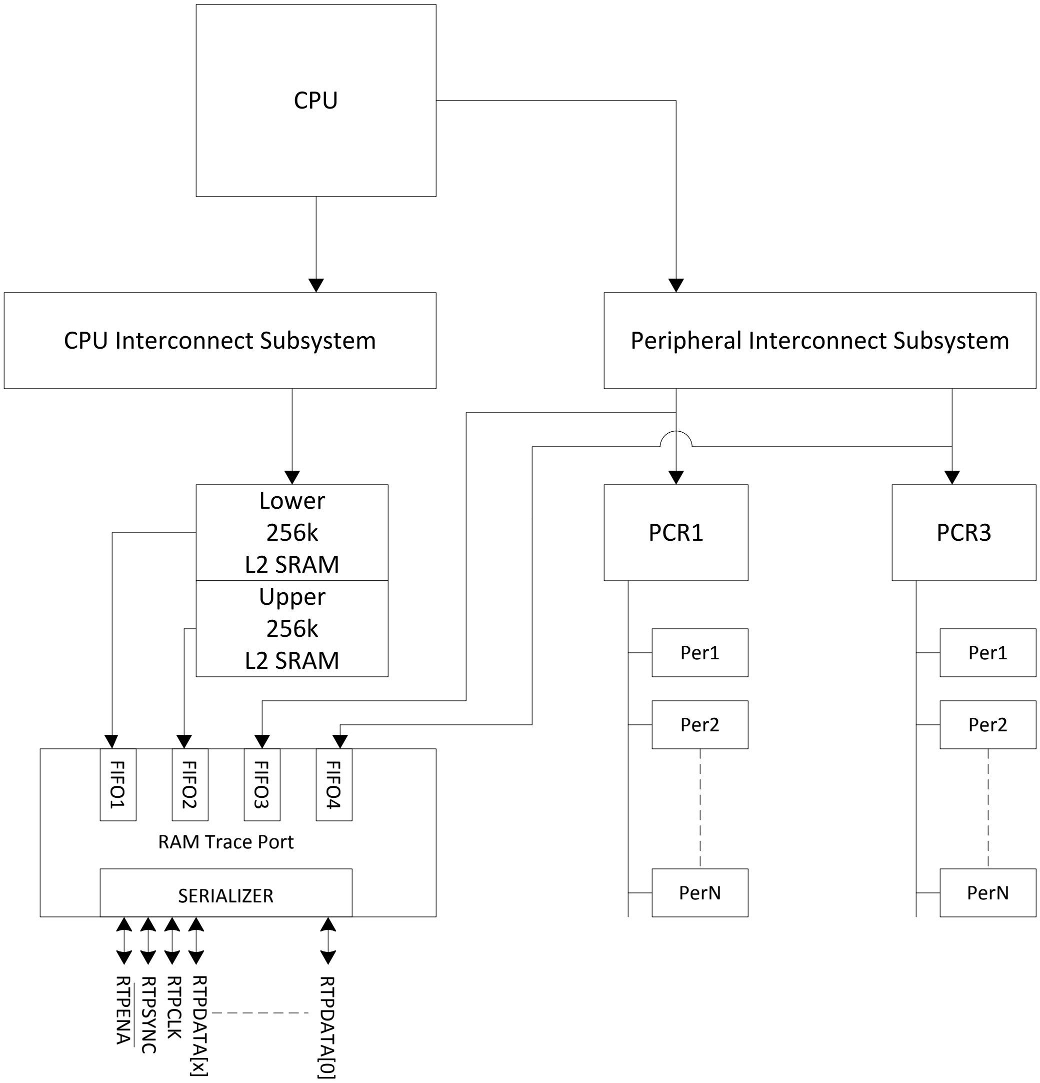
The RTP offers the following features:

- Two modes of operation - Trace Mode and Direct Data Mode
  - Trace Mode (Section 37.2.1)
    - Non-intrusive data trace on write or read operation
    - Visibility of RAM content at any time on external capture hardware
    - Trace of peripheral accesses
    - 2 configurable trace regions for each RAM module to limit amount of data to be traced
    - FIFO to store data and address of data of multiple read/write operations
    - Trace of CPU and/or DMA accesses with indication of the master in the transmitted data packet
  - Direct Data Mode (Section 37.2.2)
    - Directly write data with the CPU or trace read operations to a FIFO, without transmitting header and address information
- Dedicated synchronous interface to transmit data to external devices
- Free-running clock generation or clock stop mode between transmissions
- up to 100 Mbit per sec/pin transfer rate for transmitting data (up to 100MB/s; see device datasheet for maximum transmission clock frequency)
- Pins not used in functional mode can be used as GIOs

37.1.2 Block Diagram

Figure 37-1 is a block diagram of the RTP.

Figure 37-1. RAM Trace Port Module Block Diagram



37.2 Module Operation

The RTP module has two modes of operation: Trace Mode and Direct Data Mode.

37.2.1 Trace Mode

This mode traces all write or read accesses of CPU and/or a different master to the internal RAMs and the peripheral bus, if the access falls into one of the programmed trace regions. The trace regions allow to restrict the amount of data which is traced. This is done by specifying the start address and the size of the region to be traced. It is not possible to trace write and read operations in the same region at the same time.

Whenever a write or read access occurs, the address, data, size of the access (8, 16, 32, 64 bit), and which module initiated the write or read operation is captured into the FIFO of the corresponding RAM frame. Once new data is in the FIFO and the serializer is empty, the RTP transmits the data into the serializer and starts transmitting it.

The FIFOs are shifting data into the serializer in a round-robin scheme. This means if data is available in multiple FIFOs, the sequence for shifting data into the serializer is FIFO1, FIFO2, FIFO3 and then FIFO4. Only one entry in the respective FIFO is provided to the serializer before switching to the next FIFO. If a FIFO does not hold new data, it will be skipped. This scheme ensures that the FIFOs are drained uniformly.

NOTE: This device implements Level 1 cache memory. Reading and writing from/to Level 2 RAMs which is declared Cacheable can result in RAM traces that do not correspond to the software's original intent. Reading from Level 2 RAMs which is declared Cacheable will not result in any load transaction if the address is a hit in the level 1 cache memory. If a write-through with allocate on reads policy is selected and a cache miss happens, the cache controller will also allocate (load) the matching cache line from level 2 RAM after the data is written to the Level 2 RAM. This load due to the allocate on reads policy will result in the read data being traced.

37.2.1.1 Packet Format in Trace Mode

Figure 37-2 and Figure 37-3 illustrate this format.

Figure 37-2. Packet Format Trace Mode for RAM Locations

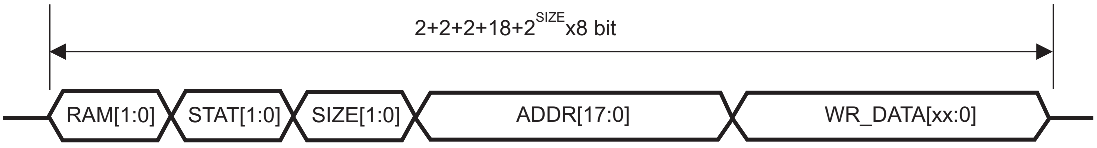
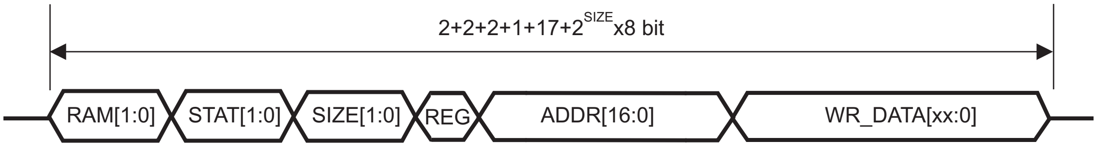


Figure 37-3. Packet Format Trace Mode for Peripheral Locations



When RAM locations are traced, one packet consists of two bits denoting the RAM block in which the data is stored or if the access has been to a peripheral location (Table 37-1), two status bits showing the access initiator or if there was a FIFO overflow (Table 37-2), two bit size (8, 16, 32, or 64 bit) information of the data (Table 37-3), the 18-bit address for RAM accesses and 2^{SIZE} × 8 bits of data. If a peripheral location is traced, then the effective address reduces to 17 bits(ADDR[16:0]) and a separate bit (REG) between the SIZE information and the address denotes which programmable region has traced this peripheral access (Table 37-4). With the region identifier, the external hardware can determine which peripheral was traced.

Table 37-1. Encoding of RAM Bits in Trace Mode Packet Format

RAM[1:0]	RAM
0	Level 2 lower 256kB RAM
1h	Level 2 upper 256kB RAM
2h	Peripherals under PCR1
3h	Peripherals under PCR3

Table 37-2. Encoding of Status Bits in Trace Mode Packet Format

STAT[1:0]	Status
0	Normal entry CPU1
1h	Normal entry other Master
2h	Normal entry CPU2
3h	Overflow of the dedicated FIFO

In the event of a FIFO overflow, an overflow will be signaled in the status bits of the next transmitted packet of that particular FIFO. The last entry in the FIFO will not be overwritten by the new data.

Table 37-3. Encoding of SIZE bits in Trace Mode Packet Format

SIZE[1:0]	Write/Read Size
0	8 bit
1h	16 bit
2h	32 bit
3h	64 bit

Table 37-4. Encoding of REG in Trace Mode Packet Format

REG	Region
0	1
1	2

The packet will be split up into several subpackets when transmitted over the RTP port pins depending on the port width configured. The port width is configured with bits PW[1:0] in the RTPGLBCTRL register (Section 37.3.1). For certain port width configurations and write/read sizes, the number of bits in a packet does not exactly match the port width for the last subpacket. The remaining bits will be filled with zeros.

Table 37-5. Number of Transfers/Packet

	Write/Read Size in bits			
Port Width	8	16	32	64
2	16 --> 16	20 --> 20	28 --> 28	44 --> 44
4	8 --> 8	10 --> 10	14 --> 14	22 --> 22
8	4 --> 4	5 --> 5	7 --> 7	11 --> 11
16	2 --> 2	2.5 --> 3	3.5 --> 4	5.5 --> 6

Example: For a 16-bit port and with data of 16-bit, the last transfer has to be padded with eight 0s. This effectively results in a transfer of 48 bits instead of 40. However the whole transfer is completed in 3 RTPCLK cycles.

For a detailed description of the representation of the packet on the RTP port pins, refer to Section 37.2.5.

37.2.2 Direct Data Mode (DDM)

In this mode, data is written directly by the CPU or other master to a dedicated capture register (RTPDDMW). The data is then transferred from the capture register to the FIFO. In a different configuration the module traces the data on read operations on the RAM directly into the FIFOs. In Direct Data Mode, no information other than the actual data is transmitted. The address of the written data can only be determined by the order of writes or reads by the CPU or other master. This mode is especially useful if a block of data on consecutive addresses has to be transmitted.

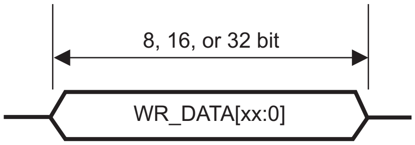
The transfer size (8, 16, or 32 bit) is programmable, but cannot be dynamically changed. Data not written/read in the correct transfer size will be truncated/extended. For example, if the transfer size is programmed to 16 bits and a 32-bit write operation is performed, the data written to the FIFO will be 32-bit wide, however only the upper 16 bits of the FIFO will be transmitted. If an 8-bit operation is performed, bits 8-15 of the FIFO will be indeterminate, so the upper 8 bits of the data transmitted are dependent on the previous contents of the FIFO RAM.

When the module is configured in Direct Data Mode (TM_DDM = 1) to trace write operations (DDM_RW = 1) to the RTPDDMW register, the programming of the trace regions for all FIFOs will be ignored and data tracing, when accessing the addresses defined by the regions, will not occur. If the module is configured in read mode (DDM_RW = 0), and if the read access to a RAM block falls into a valid trace region, the data will be traced into the corresponding FIFO for this RAM block. Since no address information is transmitted in Direct Data Mode, the executing program has to make sure that one FIFO is completely empty (RTPGSR register), before new data is traced into the next FIFO.

37.2.2.1 Packet Format in Direct Data Mode

In Direct Data Mode write or read operations, only the data written to the RTP direct data mode write register (RTPDDMW) or the data read from RAM, and therefore captured into the FIFO, will be transmitted. The packet length is programmable (8, 16, or 32 bits). Figure 37-4 illustrates this format.

Figure 37-4. Packet Format in Direct Data Mode



37.2.3 Trace Regions

To limit the amount of data to be trace, two trace regions per RAM or peripheral are implemented. These can be programmed to specific start addresses and block sizes. Depending on the device configuration (number of RAM blocks), not all regions might be implemented. Trace regions are used in Trace Mode for read or write trace and in Direct Data Mode for read trace. In Direct Data Mode write configuration, the data has to be written directly to the RTP direct data mode write register (RTPDDMW).

The RAM and peripherals start at fixed addresses in the devices memory map. With this the start address of a region does not need to be specified with its full 32-bit address. For RAM regions, only the lower 18-bit need to be programmed. The peripheral address frame covers a wider range and the start address needs to be programmed with the lower 24-bit.

The trace regions do not support a programmable end address; however, a block size needs to be specified for each region. The block size can be chosen from as low as 256 Bytes up to 256 kBytes (128 kBytes for peripherals).

37.2.3.1 Inverse Trace Regions

The RTP can be configured to trace accesses which fall into, or are made outside of the specified regions. This can be accomplished by the INV_RGN bit. If this bit is 0, all access which are made inside a region are traced. If the bit is 1, all accesses outside the region are traced. The INV_RGN bit affects all regions of the RTP, see the RTP global control register (RTPGLBCTRL).

There are certain restrictions when using INV_RGN = 1:

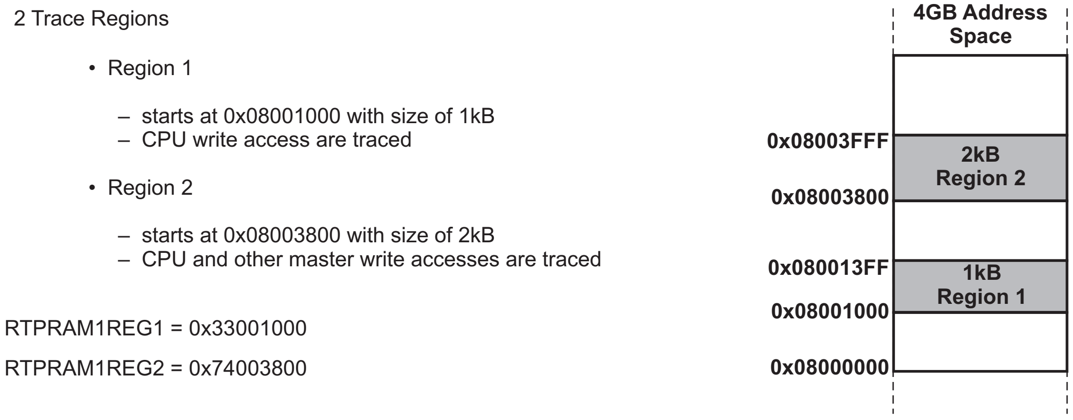
- In this mode up to 2 regions can be excluded from tracing accesses to a particular RAM.
- Inverse trace regions with one or both regions of a RAM programmed with blocksize = 0 is not supported. If only one address range should be excluded from the trace, either the address range has to be covered by both regions (e.g. excluding 1kB range with two 512B regions), or both regions have to be programmed with the same start address and region size. If the whole RAM should be traced, inverse region mode should not be used, instead the 2 regions could be programmed to cover the entire address range with INV_RGN = 0.
- Both regions have to define the same access rights (bits CPU_DMA and RW; see Section 37.3.4) for accesses outside of the region of each RAM block, otherwise the result is undefined.
- Peripheral trace in inverse region mode is not supported. The 16 MByte peripheral address range cannot be covered entirely by the 17 bit address definition of the RTP protocol.

37.2.3.2 Overlapping Trace Regions

When in INV_RGN = 0 mode with both regions overlapping and an access is done into the overlapping address range, both regions will be checked for their access rights and if one or both is satisfied, the access will be traced. In the case that both regions would allow the data to be traced, there will still be only one entry into the FIFO.

If accesses to peripherals are done within overlapping regions, the REG bit in the protocol will be 0, denoting Region 1 (see Section 37.2.1.1).

Figure 37-5. Example for Trace Region Setup



2 Trace Regions

- Region 1
  - starts at 0x08001000 with size of 1kB
  - CPU write access are traced
- Region 2
  - starts at 0x08003800 with size of 2kB
  - CPU and other master write accesses are traced

RTPRAM1REG1 = 0x33001000

RTPRAM1REG2 = 0x74003800

37.2.3.3 Cortex-R5 Specifics

Considerations/Restrictions

- Reading and writing from/to Level 2 RAMs which is declared Cacheable can result in RAM traces that do not correspond to the software's original intent.
- A store instruction to Non-cacheable, or write-through Normal memory might not result in an AXI transfer to the Level 2 RAMs or peripherals because of the merging of store in the internal buffers.

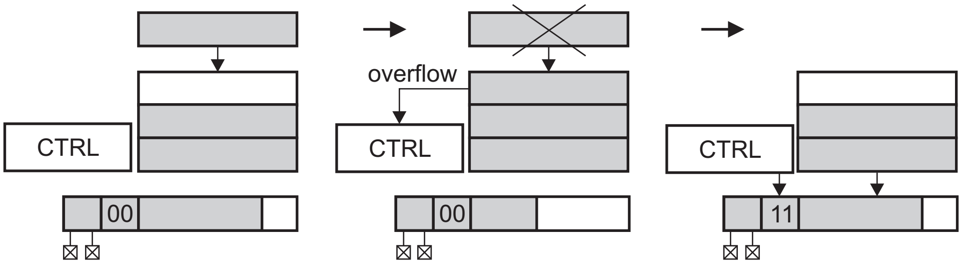
37.2.4 Overflow/Empty Handling

In case the application does RAM accesses faster than the FIFO can be emptied via the external pin interface, the FIFO can overflow. You can choose whether the program execution/data transfer should be suspended, or an overflow should be signaled in the status bits of the next, to be transmitted, message of this particular FIFO. If program execution is resumed, the data will be lost. The overflow will not be signaled in the message that is already in the serializer and being transmitted when the overflow occurs.

NOTE: The status information will only be transmitted in Trace Mode, since the Direct Data Mode packet does not contain any status information.

When an overflow in a FIFO occurs, the corresponding bit in the RTP global status register (RTPGSR) will be set.

Figure 37-6. FIFO Overflow Handling



37.2.5 Signal Description

Table 37-6 lists the signals of the RTP.

Table 37-6. RTP Signals

Signal	Description
RTPCLK	This clock signal is used to clock out the data of the serializer. Depending on the CONTCLK bit, the clock can be suspended between packets or it can be free running. The RTPCLK frequency can be adjusted by the PRESCALER bits (see RTP global control register (RTPGLBCTRL).
RTPSYNC	The module provides a packet-sync signal. This signal will go high on the rising edge of RTPCLK and will be valid for one RTPCLK cycle to synchronize external hardware to the data stream. The RTPSYNC pulse will be generated for each new packet.
$\overline{\text{RTPENA}}$	This signal is an input and can be used by external hardware to stop the data transmission between packets. When the $\overline{\text{RTPENA}}$ signal goes high, the RTP will finish the current packet transmission and then stop. Once the signal is pulled low again, the RTP will resume the transfer if data is still present in the serializer or FIFOs. The $\overline{\text{RTPENA}}$ signal does not have to be used for proper module operation. It can be used in GIO mode if the external hardware cannot generate this signal. Overflows of the external system cannot be handled in this case.
RTPDATA[15:0]	These pins are used to do the actual data transfer. Data changes with the rising edge of RTPCLK. The port can be configured for different widths (PW[1:0]). The minimum port width supported is 2 pins. See Table 37-10 which pins are used for the port.

Figure 37-7 shows an example of multiple packet transmissions in Trace Mode with an interruption between packets because of $\overline{\text{RTPENA}}$ pulled high.

Figure 37-7. RTP Packet Transfer with Sync Signal

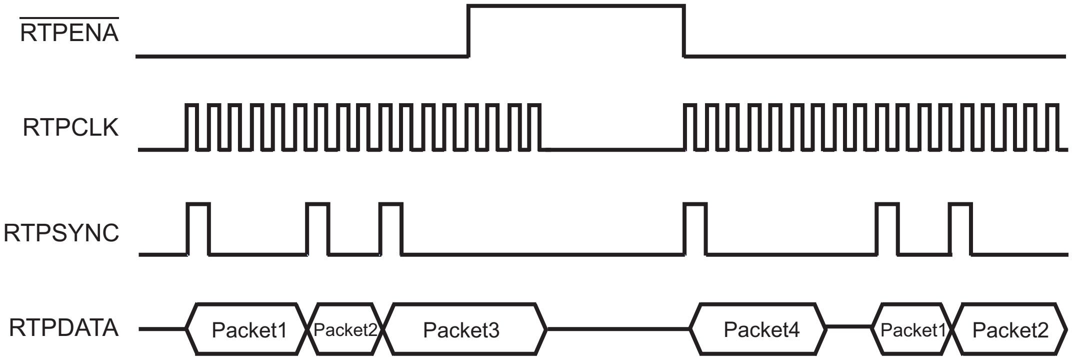
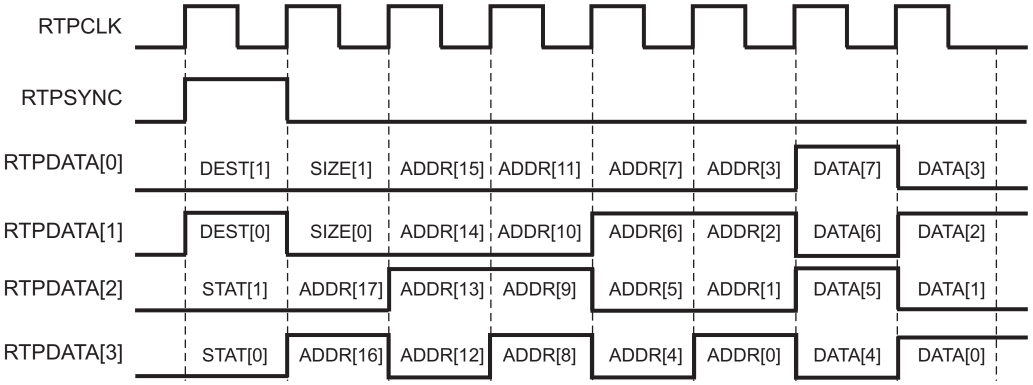


Figure 37-8 shows an example of a 4-bit data port with 8-bit write data (A5h) written into RAM1 (address 12345h) with no overflow in trace mode.

Figure 37-8. Packet Format in Trace Mode



37.2.6 Data Rate

The module is configurable to support different RTPCLK frequencies. See the device datasheet for the maximum supported frequency. HCLK will be prescaled to achieve the desired RTPCLK frequency. The prescaler supports prescale values from 1 to 8, using the RTP global control register (RTPGLBCTRL).

The effective bandwidth depends on the configuration of the module and the average data width transmitted in the packets.

37.2.7 GIO Function

Pins which are not used for RTP functionality can be used as normal GIO pins. If pins should be used in functional mode or GIO mode, they can be programmed in the RTP pin control 0 register (RTPPC0). The direction of the pins can be chosen in the RTP pin control 1 register (RTPPC1).

Module pins can have either an internal pullup or active pulldown that makes it possible to leave the pins unconnected externally when configured as inputs. The pins can be programmed to have the active pull capability by writing a 0 to the corresponding bit in the RTP pin control 7 register (RTPPC7). Writing a 1 to the corresponding bit disables the active pull functionality of the pin. A pull up can be configured by writing 1 to the corresponding bit in the RTP pin control 8 register (RTPPC8). Writing 0 will activate the pulldown capability. The pullup/pulldown is deactivated when a bidirectional pin is configured as an output. If the pullup/down capability is disabled (RTPPC7) and the pull is configured as pulldown (RTPPC8), the input buffer will be disabled.

The GIO pin can be configured to include an open drain functionality when they are configured as output pins. This is done by writing a 1 into the corresponding bit of the RTP pin control 6 register (RTPPC6). When the open drain functionality is enabled, a zero written to the data output register (RTPPC3) forces the pin to a low output voltage (V_{OL} or lower), whereas writing a 1 to the data output register (RTPPC3) forces the pin to a high impedance state. The open drain functionality is disabled when the pin is configured as an input pin.

37.3 RTP Control Registers

Table 37-7 lists the RTP module registers. The registers support 8-, 16-, and 32-bit writes. The base address of the RTP module is FFFF FA00h.

Table 37-7. RTP Control Registers

Offset	Acronym	Register Description	Section
00h	RTPGLBCTRL	RTP Global Control Register	Section 37.3.1
04h	RTPTRENA	RTP Trace Enable Register	Section 37.3.2
08h	RTPGSR	RTP Global Status Register	Section 37.3.3
0Ch	RTPRAM1REG1	RTP RAM 1 Trace Region 1 Register	Section 37.3.4
10h	RTPRAM1REG2	RTP RAM 1 Trace Region 2 Register	Section 37.3.4
14h	RTPRAM2REG1	RTP RAM 2 Trace Region 1 Register	Section 37.3.5
18h	RTPRAM2REG2	RTP RAM 2 Trace Region 2 Register	Section 37.3.5
1Ch	RTPRAM3REG1	RTP RAM 3 Trace Region 1 Register	Section 37.3.6
20h	RTPRAM3REG2	RTP RAM 3 Trace Region 2 Register	Section 37.3.6
24h	RTPPERREG1	RTP Peripheral Trace Region 1 Register	Section 37.3.7
28h	RTPPERREG2	RTP Peripheral Trace Region 2 Register	Section 37.3.7
2Ch	RTPDDMW	RTP Direct Data Mode Write Register	Section 37.3.8
34h	RTPPC0	RTP Pin Control 0 Register	Section 37.3.9
38h	RTPPC1	RTP Pin Control 1 Register	Section 37.3.10
3Ch	RTPPC2	RTP Pin Control 2 Register	Section 37.3.11
40h	RTPPC3	RTP Pin Control 3 Register	Section 37.3.12
44h	RTPPC4	RTP Pin Control 4 Register	Section 37.3.13
48h	RTPPC5	RTP Pin Control 5 Register	Section 37.3.14
4Ch	RTPPC6	RTP Pin Control 6 Register	Section 37.3.15
50h	RTPPC7	RTP Pin Control 7 Register	Section 37.3.16
54h	RTPPC8	RTP Pin Control 8 Register	Section 37.3.17

37.3.1 RTP Global Control Register (RTPGLBCTRL)

The configuration of the module can be changed with this register. Figure 37-9 and Table 37-8 describe this register.

Figure 37-9. RTP Global Control Register (RTPGLBCTRL) (offset = 00h)

31 25	24	23 19	18 16
Reserved	TEST	Reserved	PRESCALER
R-0	R/WP-0	R-0	R/WP-7h

15 14	13 12	11	10	9 8
Reserved	DDM_WIDTH	DDM_RW	TM_DDM	PW
R-0	R/WP-0	R/WP-0	R/WP-0	R/WP-0

7	6	5	4	3 0
RESET	CONTCLK	HOVF	INV_RGN	ON/OFF
R/WP-0	R/WP-0	R/WP-0	R/WP-0	R/WP-5h

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privileged mode only; -*n* = value after reset

Table 37-8. RTP Global Control Register (RTPGLBCTRL) Field Descriptions

Bit	Field	Value	Description
31-25	Reserved	0	Reads return 0. Writes have no effect.
24	TEST		By setting the bit, the FIFO RAM will be mapped into the SYSTEM Peripheral frame starting at address 0xFFF83000. Each FIFO starts at a 1-k boundary. Each FIFO entry is aligned to a 128-bit boundary. See Table 37-9 for a listing of the FIFOs and their corresponding addresses.
			Read:
		0	FIFO RAM is not accessible in the memory-map.
		1	FIFO RAM is mapped to address FFF8 3000h.
			Write in Privilege:
		0	Disables mapping of the FIFO RAM.
		1	Enables mapping of the FIFO RAM into address FFF8 3000h.
18-16	PRESCALER		The prescaler divides HCLK down to the desired RTPCLK frequency. The maximum RTPCLK frequency specified in the device datasheet must not be exceeded. No dynamic change of RTPCLK is supported. The module should be switched off by the ON/OFF bits in this register before changing the prescaler.
			User and privilege mode read, privilege mode write:
		0	Prescaler is HCLK/1.
		1h	Prescaler is HCLK/2.
		2h	Prescaler is HCLK/3.
		3h	Prescaler is HCLK/4.
		4h	Prescaler is HCLK/5.
		5h	Prescaler is HCLK/6.
		6h	Prescaler is HCLK/7.
		7h	Prescaler is HCLK/8.
15-14	Reserved	0	Reads return 0. Writes have no effect.
13-12	DDM_WIDTH		Direct data mode word size width. This bit field configures the number of bits that will be transmitted in Direct Data Mode.
			User and privilege mode read, privilege mode write:
		0	Word size width is 8 bits.
		1h	Word size width is 16 bits.
		2h	Word size width is 32 bits.
		3h	Reserved

Table 37-8. RTP Global Control Register (RTPGLBCTRL) Field Descriptions (continued)

Bit	Field	Value	Description
11	DDM_RW		Direct data mode.
			Read:
		0	Read tracing in Direct Data Mode is enabled.
		1	Write tracing in Direct Data Mode to DDMW register is enabled.
			Write in Privilege:
		0	Enable read tracing in Direct Data Mode. The RW bits in the RTPRAMxREGy registers to be ignored.
		1	Write tracing in Direct Data Mode to DDMW register is enabled. The RW bits in the RTPRAMxREGy registers are to be ignored.
10	TM_DDM		Trace Mode or Direct Data Mode.
			Read:
		0	Module is configured in Trace Mode.
		1	Module is configured in Direct Data Mode.
			Write in Privilege:
		0	Configure module to Trace Mode.
		1	Configure module to Direct Data Mode.
9-8	PW		Port width. This bit field configures the RTP to the desired port width. Pins that are not used for functional mode can be used as GIO pins. See Table 37-10 for which pins are used for the port.
		0	RTP is 2 pins wide.
		1h	RTP is 4 pins wide.
		2h	RTP is 8 pins wide.
		3h	RTP is 16 pins wide.
7	RESET		This bit resets the state machine and the registers to their reset value. This reset ensures that no data left in the FIFOs is shifted out after switching on the module with the ON/OFF bit.
			Read:
		0	RTP module is out of reset.
		1	RTP module is in reset.
			Write in Privilege:
		0	Do not reset the module.
		1	Reset the module.
6	CONTCLK		Continuous RTPCLK enable.
			Read:
		0	RTPCLK is stopped between transmissions.
		1	RTPCLK is free running.
			Write in Privilege:
		0	Stop RTPCLK between transmissions.
		1	Configure RTPCLK as free running.
5	HOVF		Halt on overflow. This bit indicates whether the CPU or DMA is halted while only one location in the FIFO is empty in Trace Mode or Direct Data Mode (read).
			Read:
		0	Current data transfer to the FIFO will not be suspended in case of a full FIFO.
		1	Current data transfer to the FIFO will be suspended in case of a full FIFO.
			Write in Privilege:
		0	The halt on FIFO overflow will be disabled. The data transfer will not be suspended and will be discarded. Data written to the RTPDDMW register will overwrite the RTPDDMW register.
		1	The halt on FIFO overflow will be enabled. Data written to the already full FIFO will be written once the FIFO is emptied again. The data transfer to the FIFO will be suspended and signaled to the CPU or other master while there is still data to be shifted out. When there is an empty FIFO location again, the transfer of the data to the FIFO will be finished.

Table 37-8. RTP Global Control Register (RTPGLBCTRL) Field Descriptions (continued)

Bit	Field	Value	Description
4	INV_RGN		Trace inside or outside of defined trace regions.
			Read:
		0	Accesses inside the trace regions are traced.
		1	Accesses outside the trace regions are traced.
			Write in Privilege:
		0	Allow tracing of accesses inside the regions set in RTPRAMxREGy.
		1	Allow tracing of accesses outside the regions set in RTPRAMxREGy.
3-0	ON/OFF		ON/Off switch.
			Read:
		Ah	Tracing of data is enabled.
		All other values	Tracing of data is disabled.
			Write in Privilege:
		Ah	Enable Tracing of data. If there is any previous captured data remaining, it will be shifted out.
		All other values	Disable tracing of data. If there is still data left in the shift register, it will be shifted out before disabling the shift operations. The data captured in the FIFO remains there until the ON/OFF bits are set to Ah.
			NOTE: It is recommended to write 5h to disable the module to prevent a soft error from enabling the module inadvertently by a single bit flip.

Table 37-9. FIFO Corresponding Addresses

FIFO	Address
1	FFF8 3000h
2	FFF8 3400h
3	FFF8 3800h
4	FFF8 3C00h

Table 37-10. Pins Used for Data Communication

Port Width (PW)	Pins Used
00	RTPDATA[1:0]
01	RTPDATA[3:0]
10	RTPDATA[7:0]
11	RTPDATA[15:0]

37.3.2 RTP Trace Enable Register (RTPTRENA)

This register enables/disables tracing of the different RAM blocks or the peripherals individually. Figure 37-10 and Table 37-11 describe this register.

Figure 37-10. RTP Trace Enable Register (RTPTRENA) (offset = 04h)

31 25	24	23 17	16
Reserved	ENA4	Reserved	ENA3
R-0	R/WP-0	R-0	R/WP-0

15 9	8	7 1	0
Reserved	ENA2	Reserved	ENA1
R-0	R/WP-0	R-0	R/WP-0

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privileged mode only; -*n* = value after reset

Table 37-11. RTP Trace Enable Register (RTPTRENA) Field Descriptions

Bit	Field	Value	Description
31-25	Reserved	0	Reads return 0. Writes have no effect.
24	ENA4		Enable tracing for peripherals under PCR3. This bit enables tracing into FIFO4 in trace mode (read/write) or direct data mode (read) operations. In Direct Data Mode write operations, this bit is ignored and tracing into FIFO4 is disabled.
			Read:
		0	Tracing is disabled.
		1	Tracing is enabled.
			Write in Privilege:
		0	Disable tracing. If RTPGLBCTRL.ON/OFF = Ah, data already captured in FIFO4 is still transmitted (RTPGLBCTRL).
		1	Enable tracing.
23-17	Reserved	0	Reads return 0. Writes have no effect.
16	ENA3		Enable tracing for peripherals under PCR1. This bit enables tracing into FIFO3 in Trace Mode (read/write) or Direct Data Mode (read) operations. In Direct Data Mode write operations, this bit is ignored and tracing into FIFO3 is disabled.
			Read:
		0	Tracing is disabled.
		1	Tracing is enabled.
			Write in Privilege:
		0	Disable tracing. If RTPGLBCTRL.ON/OFF = Ah, data already captured in FIFO3 is still transmitted (RTPGLBCTRL).
		1	Enable tracing.
15-9	Reserved	0	Reads return 0. Writes have no effect.
8	ENA2		Enable tracing for RAM block 2. This bit enables tracing into FIFO2 in Trace Mode (read/write) or Direct Data Mode (read) operations. In Direct Data Mode write operations, this bit is ignored and tracing into FIFO2 is disabled.
			Read:
		0	Tracing is disabled.
		1	Tracing is enabled.
			Write in Privilege:
		0	Disable tracing. If RTPGLBCTRL.ON/OFF = Ah, data already captured in FIFO2 is still transmitted.
		1	Enable tracing.
7-1	Reserved	0	Reads return 0. Writes have no effect.

Table 37-11. RTP Trace Enable Register (RTPTRENA) Field Descriptions (continued)

Bit	Field	Value	Description
0	ENA1		Enable tracing for RAM block 1. This bit enables tracing into FIFO1 in Trace Mode (read/write) or Direct Data Mode (read) operations. In Direct Data Mode write operations, this bit is ignored and tracing into FIFO1 is disabled.
			Read:
		0	Tracing is disabled.
		1	Tracing is enabled.
			Write in Privilege:
		0	Disable tracing. If RTPGLBCTRL.ON/OFF = Ah, data already captured in FIFO1 is still transmitted.
		1	Enable tracing.

37.3.3 RTP Global Status Register (RTPGSR)

This register provides status information of the module. Figure 37-11 and Table 37-12 describe this register.

Figure 37-11. RTP Global Status Register (RTPGSR) (offset = 08h)

31–16
Reserved
R-0

15–13	12	11	10	9	8
Reserved	EMPTYSER	EMPTYPER2	EMPTYPER1	EMPTY2	EMPTY1
R-0	R-1	R-1	R-1	R-1	R-1

7–4	3	2	1	0
Reserved	OVFPER2	OVFPER1	OVF2	OVF1
R-0	R/W1CP-0	R/W1CP-0	R/W1CP-0	R/W1CP-0

LEGEND: R/W = Read/Write; R = Read only; W1CP = Write 1 to clear in privilege mode only; -*n* = value after reset

Table 37-12. RTP Global Status Register (RTPGSR) Field Descriptions

Bit	Field	Value	Description
31-13	Reserved	0	Reads return 0. Writes have no effect.
12	EMPTYSER		Serializer empty. This bit determines if there is data left in the serializer.
		0	Serializer holds data that is shifted out.
		1	Serializer is empty.
11	EMPTYPER2		Peripheral FIFO empty. This bit determines if there are entries left in the FIFO. FIFO4 is used for tracing peripherals under PCR3.
		0	FIFO4 contains entries.
		1	FIFO4 is empty.
10	EMPTYPER1		Peripheral FIFO empty. This bit determines if there are entries left in the FIFO. FIFO3 is used for tracing peripherals under PCR1.
		0	FIFO3 contains entries.
		1	FIFO3 is empty.
9	EMPTY2		RAM block 2 FIFO empty. This bit determines if there are entries left in the FIFO. FIFO2 is used for tracing the upper 256kB RAM.
		0	FIFO2 contains entries.
		1	FIFO2 is empty.
8	EMPTY1		RAM block 1 FIFO empty. This bit determines if there are entries left in the FIFO. FIFO1 is used for tracing the lower 256kB RAM.
		0	FIFO1 contains entries.
		1	FIFO1 is empty.
7-4	Reserved	0	Reads return 0. Writes have no effect.
3	OVFPER		Overflow peripheral FIFO. This flag indicates that FIFO4 had all locations full and another attempt to write data to it occurred. The bit will not be cleared automatically if the FIFO is emptied again. The bit will stay set until the CPU clears it.
			Read:
		0	No overflow occurred.
		1	An overflow occurred.
			Write in Privilege:
		0	No effect.
		1	Clears the bit.
2	Reserved	0	Reads return 0. Writes have no effect.

Table 37-12. RTP Global Status Register (RTPGSR) Field Descriptions (continued)

Bit	Field	Value	Description
1	OVF2		Overflow RAM block 2 FIFO. This flag indicates that FIFO2 had all locations full and another attempt to write data to it occurred. The bit will not be cleared automatically if the FIFO is emptied again. The bit will stay set until the CPU clears it.
			Read:
		0	No overflow occurred.
		1	An overflow occurred.
			Write in Privilege:
		0	No effect.
		1	Clears the bit.
0	OVF1		Overflow RAM block 1 FIFO. This flag indicates that FIFO1 had all locations full and another attempt to write data to it occurred. The bit will not be cleared automatically if the FIFO is emptied again. The bit will stay set until the CPU clears it.
			Read:
		0	No overflow occurred.
		1	An overflow occurred.
			Write in Privilege:
		0	No effect.
		1	Clears the bit.

37.3.4 RTP RAM 1 Trace Region Registers (RTPRAM1REG[1:2])

Figure 37-12 and Table 37-13 illustrate these registers.

Figure 37-12. RTP RAM 1 Trace Region Registers (RTPRAM1REGn) (offset = 0Ch and 10h)

31 29	28	27 24	23 18	17 16
CPU_DMA	RW	BLOCKSIZE	Reserved	STARTADDR
R/WP-0	R/WP-0	R/WP-0	R-0	R/WP-0

15 0
STARTADDR
R/WP-0

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privileged mode only; -*n* = value after reset

Table 37-13. RTP RAM 1 Trace Region Registers (RTPRAM1REGn) Field Descriptions

Bit	Field	Value	Description
31-29	CPU_DMA		When the device is configured in lock-step mode, bit 31 will return 0 and a write has no effect. This bit field indicates if read or write operations are traced either coming from the CPU and/or from the other master.
			User and privilege mode read, privilege mode write:
		0	Read or write operations are traced when coming from the CPU and the other master.
		1h	Read or write operations are traced only when coming from the CPU.
		2h	Read or write operations are traced only when coming from the other non-CPU master.
		3h	Reserved
28	RW		Read/Write. This bit indicates if read or write operations are traced in Trace Mode or Direct Data Mode (read operation). If configured for write in Direct Data Mode (RTPGLBCTRL), the data captured will be discarded. A write operation in Direct Data Mode has to be directly to the RTP direct data mode write register (RTPDDMW) instead of to RAM. Depending on the INV_RGN bit setting, accesses into or outside the region will be traced.
			Read:
		0	Read operations will be captured.
		1	Write operations will be captured.
			Write in Privilege:
		0	Trace read accesses.
		1	Trace write accesses.
27-24	BLOCKSIZE		These bits define the length of the trace region. Depending on the setting of INV_RGN (RTPGLBCTRL), accesses inside or outside the region defined by the start address and blocksize will be traced. If all bits of BLOCKSIZE are 0, the region is disabled and no data will be captured.
			Region size (in bytes):
		0	0
		1h	256
		2h	512
		3h	1K
		4h	2K
		Ah	128K
		Bh	256K
		Ch-Fh	Reserved
23-18	Reserved	0	Reads return 0. Writes have no effect.
17-0	STARTADDR	0-3 FFFFh	These bits define the starting address of the address region that should be traced. The start address has to be a multiple of the block size chosen. If the start address is not a multiple of the block size, the start of the region will begin at the next lower block size boundary.

37.3.5 RTP RAM 2 Trace Region Registers (RTPRAM2REG[1:2])

Figure 37-13 and Table 37-14 illustrate these registers.

Figure 37-13. RTP RAM 2 Trace Region Registers (RTPRAM2REGn) (offset = 14h and 18h)

31-29	28	27-24	23-18	17-16
CPU_DMA	RW	BLOCKSIZE	Reserved	STARTADDR
R/WP-0	R/WP-0	R/WP-0	R-0	R/WP-0

15-0
STARTADDR
R/WP-0

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privileged mode only; -*n* = value after reset

Table 37-14. RTP RAM 2 Trace Region Registers (RTPRAM2REGn) Field Descriptions

Bit	Field	Value	Description
31-29	CPU_DMA		When the device is configured in lock-step mode, bit 31 will return 0 and a write has no effect. This bit field indicates if read or write operations are traced either coming from the CPU and/or from the other master.
			User and privilege mode read, privilege mode write:
		0	Read or write operations are traced when coming from the CPU and the other master.
		1h	Read or write operations are traced only when coming from the CPU.
		2h	Read or write operations are traced only when coming from the other master.
		3h	Reserved
28	RW		Read/Write. This bit indicates if read or write operations are traced in Trace Mode or Direct Data Mode (read operation). If configured for write in Direct Data Mode (RTPGLBCTRL), the data captured will be discarded. A write operation in Direct Data Mode has to be directly to the RTP direct data mode write register (RTPDDMW) instead of to RAM. Depending on the INV_RGN bit setting, accesses into or outside the region will be traced.
			Read:
		0	Read operations will be captured.
		1	Write operations will be captured.
			Write in Privilege:
		0	Trace read accesses.
		1	Trace write accesses.
27-24	BLOCKSIZE		These bits define the length of the trace region. Depending on the setting of INV_RGN (RTPGLBCTRL), accesses inside or outside the region defined by the start address and blocksize will be traced. If all bits of BLOCKSIZE are 0, the region is disabled and no data will be captured.
			Region size (in bytes):
		0	0
		1h	256
		2h	512
		3h	1K
		4h	2K
		Ah	128K
		Bh	256K
		Ch-Fh	Reserved
23-18	Reserved	0	Reads return 0. Writes have no effect.
17-0	STARTADDR	0-3 FFFFh	These bits define the starting address of the address region that should be traced. The start address has to be a multiple of the block size chosen. If the start address is not a multiple of the block size, the start of the region will begin at the next lower block size boundary.

37.3.6 RTP RAM 3 Trace Region Registers (RTPRAM3REG[1:2])

FIFO3 was originally designed to support RAM trace limiting to a maximum trace range of 256kB. In this device, FIFO3 is dedicated for tracing the PCR1 peripheral accesses. Peripherals in PCR1 occupy a total address range of 512kB of space. Therefore, it is not possible to trace the entire range of 512kB since there are only 18 bits of address being traced out in the packet. You can use the two trace regions to trace any two areas in the lower half of the PCR1's space from 0xFFF80000 to 0xFFFBFFFF provided there is not an intentional or un-intentional access to the upper half of the space. If you want to trace the upper half of PCR1's space from 0xFFFC0000 to 0xFFFFFFFF then the external hardware/software must reconstruct the full 32-bit address by forcing address bit 18 high and also ensures that there is no intentional or un-intentional accesses to the lower half of the space. Since the external hardware is unable to distinguish between the lower half and the upper half of PCR1, you can not trace both of the halves at the same time.

NOTE: Bit REG (Section 37.2.1.1) in the protocol for peripheral trace will be not be applicable to the PCR1 trace. PCR1 trace follows the RAM trace protocol with 18 bits of address trace out.

Figure 37-14 and Table 37-15 illustrate these registers.

Figure 37-14. RTP RAM 3 Trace Region Registers (RTPRAM3REGn) (offset = 1Ch and 20h)

31 29	28	27 24	23 18	17 16
CPU_DMA	RW	BLOCKSIZE	Reserved	STARTADDR
R/WP-0	R/WP-0	R/WP-0	R-0	R/WP-0

15 0
STARTADDR
R/WP-0

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privileged mode only; -*n* = value after reset

Table 37-15. RTP RAM 3 Trace Region Registers (RTPRAM3REGn) Field Descriptions

Bit	Field	Value	Description
31-29	CPU_DMA		When the device is configured in lock-step mode, bit 31 will return 0 and a write has no effect. This bit field indicates if read or write operations are traced either coming from the CPU and/or from the other master.
			User and privilege mode read, privilege mode write:
		0	Read or write operations are traced when coming from the CPU and the other master.
		1h	Read or write operations are traced only when coming from the CPU.
		2h	Read or write operations are traced only when coming from the other master.
		3h	Reserved
28	RW		Read/Write. This bit indicates if read or write operations are traced in Trace Mode or Direct Data Mode (read operation). If configured for write in Direct Data Mode (RTPGLBCTRL), the data captured will be discarded. A write operation in Direct Data Mode has to be directly to the RTP direct data mode write register (RTPDDMW) instead of to RAM. Depending on the INV_RGN bit setting, accesses into or outside the region will be traced.
			Read:
		0	Read operations will be captured.
		1	Write operations will be captured.
			Write in Privilege:
		0	Trace read accesses.
		1	Trace write accesses.

Table 37-15. RTP RAM 3 Trace Region Registers (RTPRAM3REGn) Field Descriptions (continued)

Bit	Field	Value	Description
27-24	BLOCKSIZE		These bits define the length of the trace region. Depending on the setting of INV_RGN (RTPGLBCTRL), accesses inside or outside the region defined by the start address and blocksize will be traced. If all bits of BLOCKSIZE are 0, the region is disabled and no data will be captured.
			Region size (in bytes):
		0	0
		1h	256
		2h	512
		3h	1K
		4h	2K
		Ah	128K
		Bh	256K
		Ch-Fh	Reserved
23-18	Reserved	0	Reads return 0. Writes have no effect.
17-0	STARTADDR	0-3 FFFFh	These bits define the starting address of the address region that should be traced. The start address has to be a multiple of the block size chosen. If the start address is not a multiple of the block size, the start of the region will begin at the next lower block size boundary.

37.3.7 RTP Peripheral Trace Region Registers (RTPPERREG[1:2])

FIFO4 is dedicated for tracing the PCR3 peripheral accesses. Since the peripheral frame is 16 Mbytes, the start address has to be defined as a 24-bit value. However, only bits 16 to 0 will be transmitted in the protocol. Bit REG (Section 37.2.1.1) in the protocol will be 0 if there was an access to the range defined by RTPPERREG1. REG will be 1 if the access was into the range defined by RTPPERREG2. Figure 37-15 and Table 37-16 illustrate these registers.

Figure 37-15. RTP Peripheral Trace Region Registers (RTPPERREGn) (offset = 24h and 28h)

31-29	28	27-24	23-16
CPU_DMA	RW	BLOCKSIZE	STARTADDR
R/WP-0	R/WP-0	R/WP-0	R/WP-0

15-0
STARTADDR
R/WP-0

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privileged mode only; -*n* = value after reset

Table 37-16. RTP Peripheral Trace Region Registers (RTPPERREGn) Field Descriptions

Bit	Field	Value	Description
31-29	CPU_DMA		When the device is configured in lock-step mode, bit 31 will return 0 and a write has no effect. This bit field indicates if read or write operations are traced either coming from the CPU and/or from the other master. User and privilege mode read, privilege mode write:
		0	Read or write operations are traced when coming from the CPU and the other master.
		1h	Read or write operations are traced only when coming from the CPU.
		2h	Read or write operations are traced only when coming from the other master.
		3h	Reserved
28	RW		Read/Write. This bit indicates if read or write operations are traced in Trace Mode or Direct Data Mode (read operation). If configured for write in Direct Data Mode (RTPGLBCTRL), the data captured will be discarded. A write operation in Direct Data Mode has to be directly to the RTP direct data mode write register (RTPDDMW) instead of to RAM. Depending on the INV_RGN bit setting, accesses into or outside the region will be traced. **Read:**
		0	Read operations will be captured.
		1	Write operations will be captured.
			Write in Privilege:
		0	Trace read accesses.
		1	Trace write accesses.
27-24	BLOCKSIZE		These bits define the length of the trace region. Depending on the setting of INV_RGN (RTPGLBCTRL), accesses inside or outside the region defined by the start address and blocksize will be traced. If all bits of BLOCKSIZE are 0, the region is disabled and no data will be captured. Region size (in bytes):
		0	0
		1h	256
		2h	512
		3h	1K
		4h	2K
		Ah	128K
		Bh	256K
		Ch-Fh	Reserved
23-0	STARTADDR	0-FF FFFFh	These bits define the starting address of the address region that should be traced. The start address has to be a multiple of the block size chosen. If the start address is not a multiple of the block size, the start of the region will begin at the next lower block size boundary.

37.3.8 RTP Direct Data Mode Write Register (RTPDDMW)

The CPU has to write data to this register if the module is used in Direct Data Mode write configuration. Figure 37-16 and Table 37-17 describe this register.

Figure 37-16. RTP Direct Data Mode Write Register (RTPDDMW) (offset = 2Ch)

31	0
DATA	
R/W-0	

LEGEND: R/W = Read/Write; -*n* = value after reset

Table 37-17. RTP Direct Data Mode Write Register (RTPDDMW) Field Descriptions

Bit	Field	Description
31-0	DATA	This register must be written to in a Direct Data Mode write operation to store the data into FIFO1. Data written must be right-aligned. If the FIFO is full, the reaction depends on the setting of the HOVF bit (RTPGLBCTRL). If the bit is set, the master writing the data will be wait-stated. If the bit is cleared, previous data written to the register will be overwritten. Reads of this register always return 0.

37.3.9 RTP Pin Control 0 Register (RTPPC0)

This register configures the RTP pins as functional or GIO pins. Once the pin is configured in functional mode, it overrides the settings in the RTPPC1 register. Writing to the RTPPC3, RTPPC4, and RTPPC5 registers will have no effect for pins configured as functional pins. Figure 37-17 and Table 37-18 describe this register.

Figure 37-17. RTP Pin Control 0 Register (RTPPC0) (offset = 34h)

31 19	18	17	16
Reserved	ENAFUNC	CLKFUNC	SYNCFUNC
R-0	R/W-0	R/W-0	R/W-0

15 0
DATAFUNC[15:0]
R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 37-18. RTP Pin Control 0 Register (RTPPC0) Field Descriptions

Bit	Field	Value	Description
31-19	Reserved	0	Reads return 0. Writes have no effect.
18	ENAFUNC		Functional mode of $\overline{\text{RTPENA}}$ pin.
			Read:
		0	Pin is used in GIO mode.
		1	Pin is used in functional mode.
			Write:
		0	Configure pin to GIO mode.
		1	Configure pin to functional mode.
17	CLKFUNC		Functional mode of RTPCLK pin.
			Read:
		0	Pin is used in GIO mode.
		1	Pin is used in functional mode.
			Write:
		0	Configure pin to GIO mode.
		1	Configure pin to functional mode.
16	SYNCFUNC		Functional mode of RTPSYNC pin.
			Read:
		0	Pin is used in GIO mode.
		1	Pin is used in functional mode.
			Write:
		0	Configure pin to GIO mode.
		1	Configure pin to functional mode.
15-0	DATAFUNC[*n*]		Functional mode of RTPDATA[15:0] pins. These bits define whether the pins are used in functional mode or in GIO mode. Each bit [*n*] represents a single pin.
			Read:
		0	Pin is used in GIO mode.
		1	Pin is used in functional mode.
			Write:
		0	Configure pin to GIO mode.
		1	Configure pin to functional mode.

37.3.10 RTP Pin Control 1 Register (RTPPC1)

Once the pin is configured in functional mode (using RTPPC0 register), configuring the corresponding bit in RTPPC1 to 0 will not disable the output driver. Figure 37-18 and Table 37-19 describe this register.

Figure 37-18. RTP Pin Control 1 Register (RTPPC1) (offset = 38h)

31-19	18	17	16
Reserved	ENADIR	CLKDIR	SYNCDIR
R-0	R/W-0	R/W-0	R/W-0

15-0
DATADIR[15:0]
R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 37-19. RTP Pin Control 1 Register (RTPPC1) Field Descriptions

Bit	Field	Value	Description
31-19	Reserved	0	Reads return 0. Writes have no effect.
18	ENADIR		Direction of $\overline{\text{RTPENA}}$ pin. This bit defines whether the pin is used as input or output in GIO mode. This bit has no effect when the pin is configured in functional mode.
			Read:
		0	Pin is used as input.
		1	Pin is used as output.
			Write:
		0	Configure pin to input mode.
		1	Configure pin to output mode.
17	CLKDIR		Direction of RTPCLK pin. This bit defines whether the pin is used as input or output in GIO mode. This bit has no effect when the pin is configured in functional mode.
			Read:
		0	Pin is used as input.
		1	Pin is used as output.
			Write:
		0	Configure pin to input mode.
		1	Configure pin to output mode.
16	SYNCDIR		Direction of RTPSYNC pin. This bit defines whether the pin is used as input or output in GIO mode. This bit has no effect when the pin is configured in functional mode.
			Read:
		0	Pin is used as input.
		1	Pin is used as output.
			Write:
		0	Configure pin to input mode.
		1	Configure pin to output mode.
15-0	DATADIR[*n*]		Direction of RTPDATA[15:0] pins. These bits define whether the pins are used as input or output in GIO mode. These bits have no effect when the pins are configured in functional mode. Each bit [*n*] represents a single pin.
			Read:
		0	Pin is used as input.
		1	Pin is used as output.
			Write:
		0	Configure pin to input mode.
		1	Configure pin to output mode.

37.3.11 RTP Pin Control 2 Register (RTPPC2)

This register represents the input value of the pins when in GIO or functional mode. Figure 37-19 and Table 37-20 describe this register.

Figure 37-19. RTP Pin Control 2 Register (RTPPC2) (offset = 3Ch)

31 19	18	17	16
Reserved	ENAIN	CLKIN	SYNCIN
R-0	R-x	R-x	R-x

15 0
DATAIN[15:0]
R-x

LEGEND: R = Read only; -*n* = value after reset; -x = value is indeterminate

Table 37-20. RTP Pin Control 2 Register (RTPPC2) Field Descriptions

Bit	Field	Value	Description
31-19	Reserved	0	Reads return 0. Writes have no effect.
18	ENAIN		$\overline{\text{RTPENA}}$ input. This bit reflects the state of the pin in all modes. Writes to this bit have no effect.
		0	The pin is at logic low (0) (input voltage is V_{IL} or lower).
		1	The pin is at logic high (1) (input voltage is V_{IH} or higher).
17	CLKIN		RTPCLK input. This bit reflects the state of the pin in all modes. Writes to this bit have no effect.
		0	The pin is at logic low (0) (input voltage is V_{IL} or lower).
		1	The pin is at logic high (1) (input voltage is V_{IH} or higher).
16	SYNCIN		RTPSYNC input. This bit reflects the state of the pin in all modes. Writes to this bit have no effect.
		0	The pin is at logic low (0) (input voltage is V_{IL} or lower).
		1	The pin is at logic high (1) (input voltage is V_{IH} or higher).
15-0	DATAIN[*n*]		RTPDATA[15:0] input. These bits reflect the state of the pins in all modes. Each bit [*n*] represents a single pin. Writes to this bit have no effect.
		0	The pin is at logic low (0) (input voltage is V_{IL} or lower).
		1	The pin is at logic high (1) (input voltage is V_{IH} or higher).

37.3.12 RTP Pin Control 3 Register (RTPPC3)

This register defines the state of the pins when configured in GIO mode as output pins. Once a pin is configured in functional mode (using RTPPC0 register), changing the state of the corresponding bit in RTPPC3 will not affect the pin's state. Figure 37-20 and Table 37-21 describe this register.

Figure 37-20. RTP Pin Control 3 Register (RTPPC3) (offset = 40h)

31 19	18	17	16
Reserved	ENAOUT	CLKOUT	SYNCOUT
R-0	R/W-0	R/W-0	R/W-0

15 0
DATAOUT[15:0]
R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 37-21. RTP Pin Control 3 Register (RTPPC3) Field Descriptions

Bit	Field	Value	Description
31-19	Reserved	0	Reads return 0. Writes have no effect.
18	ENAOUT		$\overline{\text{RTPENA}}$ output. This pin sets the output state of the RTPENA pin.
			Read:
		0	The pin is configured to output logic low (0) (output voltage is V_{OL} or lower).
		1	The pin is configured to output logic high (1) (output voltage is V_{OH} or higher).
			Write:
		0	Set pin to logic low (0) (output voltage is V_{OL} or lower).
		1	Set pin to logic high (1) (output voltage is V_{OH} or higher).
17	CLKOUT		RTPCLK output. This pin sets the output state of the RTPCLK pin.
			Read:
		0	The pin is configured to output logic low (0) (output voltage is V_{OL} or lower).
		1	The pin is configured to output logic high (1) (output voltage is V_{OH} or higher).
			Write:
		0	Set pin to logic low (0) (output voltage is V_{OL} or lower).
		1	Set pin to logic high (1) (output voltage is V_{OH} or higher).
16	SYNCOUT		RTPSYNC output. This pin sets the output state of the RTPSYNC pin.
			Read:
		0	The pin is configured to output logic low (0) (output voltage is V_{OL} or lower).
		1	The pin is configured to output logic high (1) (output voltage is V_{OH} or higher).
			Write:
		0	Set pin to logic low (0) (output voltage is V_{OL} or lower).
		1	Set pin to logic high (1) (output voltage is V_{OH} or higher).
15-0	DATAOUT[*n*]		RTPDATA[15:0] output. These bits set the output state of the RTPDATA[15:0] pins. Each bit [*n*] represents a single pin.
			Read:
		0	The pin is configured to output logic low (0) (output voltage is V_{OL} or lower).
		1	The pin is configured to output logic high (1) (output voltage is V_{OH} or higher).
			Write:
		0	Set pin to logic low (0) (output voltage is V_{OL} or lower).
		1	Set pin to logic high (1) (output voltage is V_{OH} or higher).

37.3.13 RTP Pin Control 4 Register (RTPPC4)

This register provides the option to set pins to a logic 1 level without influencing the state of other pins. It eliminates the read-modify-write operation necessary with the RTPPC3 register. Once the pin is configured in functional mode (using RTPPC0 register), setting the corresponding bit to 1 in RTPPC4 will not affect the pin's state. Figure 37-21 and Table 37-22 describe this register.

Figure 37-21. RTP Pin Control 4 Register (RTPPC4) (offset = 44h)

31-19	18	17	16
Reserved	ENASET	CLKSET	SYNCSET
R-0	R/W-0	R/W-0	R/W-0

15-0
DATASET[15:0]
R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 37-22. RTP Pin Control 4 Register (RTPPC4) Field Descriptions

Bit	Field	Value	Description
31-19	Reserved	0	Reads return 0. Writes have no effect.
18	ENASET		Sets the output state of $\overline{\text{RTPENA}}$ pin to logic high. Value in the ENASET bit sets the data output control register bit to 1 regardless of the current value in the ENAOUT bit .
			Read:
		0	The pin is configured to output a logic low (0) (output voltage is V_{OL} or lower).
		1	The pin is configured to output logic high (1) (output voltage is V_{OH} or higher).
			Write:
		0	No effect.
		1	Set pin to logic high (1) (output voltage is V_{OH} or higher).
17	CLKSET		Sets the output state of RTPCLK pin to logic high. Value in the CLKSET bit sets the data output control register bit to 1 regardless of the current value in the CLKOUT bit.
			Read:
		0	The pin is configured to output a logic low (0) (output voltage is V_{OL} or lower).
		1	The pin is configured to output logic high (1) (output voltage is V_{OH} or higher).
			Write:
		0	No effect.
		1	Set pin to logic high (1) (output voltage is V_{OH} or higher).
16	SYNCSET		Sets output state of RTPSYNC pin logic high. Value in the SYNCSET bit sets the data output control register bit to 1 regardless of the current value in the SYNCOUT bit.
			Read:
		0	The pin is configured to output a logic low (0) (output voltage is V_{OL} or lower).
		1	The pin is configured to output logic high (1) (output voltage is V_{OH} or higher).
			Write:
		0	No effect.
		1	Set pin to logic high (1) (output voltage is V_{OH} or higher).
15-0	DATASET[*n*]		Sets output state of RTPDATA[15:0] pins to logic high. Value in the DATAxSET bit sets the data output control register bit to 1 regardless of the current value in the DATAxOUT bit. Each bit [*n*] represents a single pin.
			Read:
		0	The pin is configured to output a logic low (0) (output voltage is V_{OL} or lower).
		1	The pin is configured to output logic high (1) (output voltage is V_{OH} or higher).
			Write:
		0	No effect.
		1	Set pin to logic high (1) (output voltage is V_{OH} or higher).

37.3.14 RTP Pin Control 5 Register (RTPPC5)

This register provides the option to set pins to a logic 0 level without influencing the state of other pins. It eliminates the read-modify-write operation necessary with the RTPPC3 register. Once the pin is configured in functional mode (using RTPPC0 register), setting the corresponding bit to 1 in RTPPC5 will not affect the pin state. Figure 37-22 and Table 37-23 describe this register.

Figure 37-22. RTP Pin Control 5 Register (RTPPC5) (offset = 48h)

31-19	18	17	16
Reserved	ENACLR	CLKCLR	SYNCCLR
R-0	R/W-0	R/W-0	R/W-0

15-0
DATACLR[15:0]
R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 37-23. RTP Pin Control 5 Register (RTPPC5) Field Descriptions

Bit	Field	Value	Description
31-19	Reserved	0	Reads return 0. Writes have no effect.
18	ENACLR		Sets the output state of $\overline{\text{RTPENA}}$ pin to logic high. Value in the ENASET bit sets the data output control register bit to 1 regardless of the current value in the ENAOUT bit.
			Read:
		0	The pin is configured to output a logic low (0) (output voltage is V_{OL} or lower).
		1	The pin is configured to output logic high (1) (output voltage is V_{OH} or higher).
			Write:
		0	No effect.
		1	Set pin to logic low (0) (output voltage is V_{OL} or lower).
17	CLKCLR		Sets output state of RTPCLK pin to logic low. Value in the CLKCLR bit sets the data output control register bit to 0 regardless of the current value in the CLKOUT bit.
			Read:
		0	The pin is configured to output a logic low (0) (output voltage is V_{OL} or lower).
		1	The pin is configured to output logic high (1) (output voltage is V_{OH} or higher).
			Write:
		0	No effect.
		1	Set pin to logic low (0) (output voltage is V_{OL} or lower).
16	SYNCCLR		Sets output state of RTPSYNC pin logic low. Value in the SYNCCLR bit clears the data output control register bit to 0 regardless of the current value in the SYNCOUT bit.
			Read:
		0	The pin is configured to output a logic low (0) (output voltage is V_{OL} or lower).
		1	The pin is configured to output logic high (1) (output voltage is V_{OH} or higher).
			Write:
		0	No effect.
		1	Set pin to logic low (0) (output voltage is V_{OL} or lower).
15-0	DATACLR[*n*]		Sets output state of RTPDATA[15:0] pins to logic low. Value in the DATAxCLR bit clears the data output control register bit to 0 regardless of the current value in the DATAxOUT bit. Each bit [*n*] represents a single pin.
			Read:
		0	The pin is configured to output a logic low (0) (output voltage is V_{OL} or lower).
		1	The pin is configured to output logic high (1) (output voltage is V_{OH} or higher).
			Write:
		0	No effect.
		1	Set pin to logic low (0) (output voltage is V_{OL} or lower).

37.3.15 RTP Pin Control 6 Register (RTPPC6)

This register configures the pins in push-pull or open-drain functionality. If configured to be open-drain, the module only drives a logic low level on the pin. An external pull-up resistor needs to be connected to the pin to pull it high when the pin is in high-impedance mode. Figure 37-23 and Table 37-24 describe this register.

Figure 37-23. RTP Pin Control 6 Register (RTPPC6) (offset = 4Ch)

31	19	18	17	16
Reserved		ENAPDR	CLKPDR	SYNCPDR
R-0		R/W-0	R/W-0	R/W-0

15	0
DATAPDR[15:0]	
R/W-0	

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 37-24. RTP Pin Control 6 Register (RTPPC6) Field Descriptions

Bit	Field	Value	Description
31-19	Reserved	0	Reads return 0. Writes have no effect.
18	ENAPDR		$\overline{\text{RTPENA}}$ Open drain enable. This bit enables open drain functionality on the pin if it is configured as a GIO output (RTPPC0[18] = 0; RTPPC1[18] = 1). If the pin is configured as a functional pin (RTPPC0[18] = 1), the open drain functionality is disabled.
			Read:
		0	Pin behaves as normal push/pull pin.
		1	Pin operates in open drain mode.
			Write:
		0	Configures the pin as push/pull.
		1	Configures the pin as open drain.
17	CLKPDR		RTPCLK Open drain enable. This bit enables open drain functionality on the pin if it is configured as GIO output (RTPPC0[17] = 0; RTPPC1[17] = 1). If the pin is configured as functional pin (RTPPC0[17] = 1), the open drain functionality is disabled.
			Read:
		0	Pin behaves as normal push/pull pin.
		1	Pin operates in open drain mode.
			Write:
		0	Configures the pin as push/pull.
		1	Configures the pin as open drain.
16	SYNCPDR		RTPSYNC Open drain enable. This bit enables open drain functionality on the pin if it is configured as a GIO output (RTPPC0[16] = 0; RTPPC1[16] = 1). If pin is configured as functional pin (RTPPC0[16] = 1), the open drain functionality is disabled.
			Read:
		0	Pin behaves as normal push/pull pin.
		1	Pin operates in open drain mode.
			Write:
		0	Configures the pin as push/pull.
		1	Configures the pin as open drain.

Table 37-24. RTP Pin Control 6 Register (RTPPC6) Field Descriptions (continued)

Bit	Field	Value	Description
15-0	DATAPDR[*n*]		RTPDATA[15:0] Open drain enable. These bits enable open drain functionality on the pins if they are configured as a GIO output (RTPPC0[15:0] = 0; RTPPC1[15:0] = 1). If the pins are configured as a functional pins (RTPPC0[15:0] = 1), the open drain functionality is disabled. Each bit [*n*] represents a single pin.
			Read:
		0	Pin behaves as normal push/pull pin.
		1	Pin operates in open drain mode.
			Write:
		0	Configures the pin as push/pull.
		1	Configures the pin as open drain.

37.3.16 RTP Pin Control 7 Register (RTPPC7)

This register controls the pullup/down functionality of a pin. The internal pullup/down can be enabled or disabled by this register. The reset configuration of these bits is device dependent, consult the device datasheet for this information. Figure 37-24 and Table 37-25 describe this register.

Figure 37-24. RTP Pin Control 7 Register (RTPPC7) (offset = 50h)

31 - 19	18	17	16
Reserved	ENADIS	CLKDIS	SYNCDIS
R-0	R/W-x	R/W-x	R/W-x

15 - 0
DATADIS[15:0]
R/W-x

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset; -x = value is indeterminate

Table 37-25. RTP Pin Control 7 Register (RTPPC7) Field Descriptions

Bit	Field	Value	Description
31-19	Reserved	0	Reads return 0. Writes have no effect.
18	ENADIS		$\overline{\text{RTPENA}}$ Pull disable. This bit removes internal pullup/pulldown functionality from the pin when it is configured as an input pin (RTPPC1[18] = 0).
			Read:
		0	Pullup/pulldown functionality is enabled.
		1	Pullup/pulldown functionality is disabled.
			Write:
		0	Enables pullup/pulldown functionality.
		1	Disables pullup/pulldown functionality.
17	CLKDIS		RTPCLK Pull disable. This bit removes the internal pullup/pulldown functionality from the pin when it is configured as an input pin (RTPPC1[17] = 0).
			Read:
		0	Pullup/pulldown functionality is enabled.
		1	Pullup/pulldown functionality is disabled.
			Write:
		0	Enables pullup/pulldown functionality.
		1	Disables pullup/pulldown functionality.
16	SYNCDIS		RTPSYNC Pull disable. Removes internal pullup/pulldown functionality from the pin when configured as an input pin (RTPPC1[16] = 0).
			Read:
		0	Pullup/pulldown functionality is enabled.
		1	Pullup/pulldown functionality is disabled.
			Write:
		0	Enables pullup/pulldown functionality.
		1	Disables pullup/pulldown functionality.
15-0	DATADIS[*n*]		RTPDATA[15:0] Pull disable. Removes internal pullup/pulldown functionality from the pins when configured as input pins (RTPPC1[15:0] = 0). Each bit [*n*] represents a single pin.
			Read:
		0	Pullup/pulldown functionality is enabled.
		1	Pullup/pulldown functionality is disabled.
			Write:
		0	Enables pullup/pulldown functionality.
		1	Disables pullup/pulldown functionality.

37.3.17 RTP Pin Control 8 Register (RTPPC8)

This register configures the internal pullup or pulldown on the input pin. A secondary function exists when the pull configuration is disabled and a pulldown is selected. This will disable the input buffer. Figure 37-25 and Table 37-26 describe this register.

NOTE: If the pullup/down is disabled in the RTPPC7 register and configured as pulldown in RTPPC8, then the input buffer is disabled.

Figure 37-25. RTP Pin Control 8 Register (RTPPC8) (offset = 54h)

31 … 19	18	17	16
Reserved	ENAPSEL	CLKPSEL	SYNCPSEL
R-0	R/W-1	R/W-1	R/W-1

15 … 0
DATAPSEL[15:0]
R/W-1

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 37-26. RTP Pin Control 8 Register (RTPPC8) Field Descriptions

Bit	Field	Value	Description
31-19	Reserved	0	Reads return 0. Writes have no effect.
18	ENAPSEL		$\overline{\text{RTPENA}}$ Pull select. This bit configures pullup or pulldown functionality if RTPPC7[18] = 0.
			Read:
		0	Pulldown functionality is enabled.
		1	Pullup functionality is enabled.
			Write:
		0	Enables pulldown functionality.
		1	Enables pullup functionality.
17	CLKPSEL		RTPCLK Pull select. This bit configures pullup or pulldown functionality if RTPPC7[17] = 0.
			Read:
		0	Pulldown functionality is enabled.
		1	Pullup functionality is enabled.
			Write:
		0	Enables pulldown functionality.
		1	Enables pullup functionality.
16	SYNCPSEL		RTPSYNC Pull select. This bit configures pullup or pulldown functionality if RTPPC7[16] = 0.
			Read:
		0	Pulldown functionality is enabled.
		1	Pullup functionality is enabled.
			Write:
		0	Enables pulldown functionality.
		1	Enables pullup functionality.
15-0	DATAPSEL[*n*]		RTPDATA[15:0] Pull select. These bits configure pullup or pulldown functionality if RTPPC7[15:0] = 0. Each bit [*n*] represents a single pin.
			Read:
		0	Pulldown functionality is enabled.
		1	Pullup functionality is enabled.
			Write:
		0	Enables pulldown functionality.
		1	Enables pullup functionality.

eFuse Controller

This chapter describes the eFuse controller.

Topic **Page**

38.1 Overview

Electrically programmable fuses (eFuses) are used to configure the device after deassertion of $\overline{\text{PORRST}}$. The eFuse values are read and loaded into internal registers as part of the power-on-reset sequence. The eFuse values are protected with single bit error correction, double bit error detection (SECDED) codes. These fuses are programmed during the initial factory test of the device. The eFuse controller is designed so that the state of the eFuses cannot be changed once the device is packaged.

38.2 Introduction

The eFuse controller automatically reads the values of the eFuses and shifts them into registers during the power-on reset sequence. No action is required from the application code. However, in a safety critical application, the user code should check to see if a correctable or an uncorrectable error was detected during the reset sequence and then preform a self-test on the eFuse controller ECC logic.

38.3 eFuse Controller Testing

38.3.1 eFuse Controller Connections to ESM

There are three connections from the eFuse controller to the Error Signaling Module (ESM). If an uncorrectable error occurs during the loading of the eFuse values after reset, a group three, channel one error and a group one channel 40 error are sent to the ESM. The group three error will cause the $\overline{\text{ERROR}}$ pin to go low. If during the eFuse loading a correctable error occurs, only a group one channel 40 error is sent to the ESM. If an error occurs during the eFuse controller self test, then a group one channel 41 error and a group one channel 40 error are sent to the ESM. After reset, by default, the group one errors do not affect the $\overline{\text{ERROR}}$ pin. If the software enables the appropriate bit in the appropriate ESM Influence Error Pin Set/Status Register (ESMIEPSRn) while the group one error is set, the $\overline{\text{ERROR}}$ pin will go low.

Table 38-1. ESM Signals Set by eFuse Controller

ESM Signal	Uncorrected Load Failure	Correctable Load Error	Self Test		
			eFuse Self Test	eFuse stuck at 0 Test	
				Version a: with Error pin	Version b: without Error pin
Group 3 Channel 1	X			X	
Group 1 Channel 40	X	X	X		
Group 1 Channel 41			X	X	X

38.3.2 Checking for eFuse Errors After Power Up

For safety critical systems, it is required that you check the status of the eFuse controller after a device reset. A suggested flow chart for checking the eFuse controller after device reset is shown in Figure 38-1. Failures during the eFuse self test can be grouped into three levels of severity. Depending on the safety critical application, the error handling for each error type may be different.

38.3.2.1 Class 1 Error

A class 1 error of the eFuse controller means that there was a failure during the autoload sequence. The values read from the eFuses cannot be relied on. All device operation is suspect. A class 1 error is indicated by a signal to group 3 channel 1 of the ESM. This will cause the $\overline{\text{ERROR}}$ pin to go active low.

38.3.2.2 Class 2 Errors

A class 2 error is an indication that the safety checks of the eFuse controller did not work. These are also serious errors because you can no longer guarantee that a more severe error did not occur.

38.3.2.3 Class 3 Error

A class 3 error indicates that there was a single bit failure reading the eFuses that was corrected by ECC bits. Proper operation is still likely, but the system is now at a higher risk for a future non-correctable error. When a correctable error occurs, ESM group 1, channel 40 will be set. In the suggested flow chart shown in Figure 38-1 below, the single bit error is determined by directly reading the eFuse error status register, and not depending on the integrity of the connections between the eFuse controller and the ESM.

38.3.2.4 Stuck at Zero Test

The purpose of the stuck at zero test is to verify that the eFuse controller could signal the ESM if an autoload error did occur. It basically verifies the path through the eFuse controller and to the ESM. This is done by writing a special instruction to the eFuse controller boundary register, then verifying that the proper bits are set in the eFuse controller pins register. Upon successful completion of this test ESM group 1 channel 41 and ESM group 3 channel 1 will be set. This will force the $\overline{\text{ERROR}}$ pin low.

- Version A
  - Write boundary register (address 0xFFF8C01C) with 0x003FC000 to set the error signals.
  - Read pins register (address 0xFFF8C02C) and verify that bits 14, 12, 11 and 10 are set.
  - Write boundary register (address 0xFFF8C01C) with 0x003C0000, to clear the error signals.
  - Verify that ESM group 1 channel 41 and group 3 channel 1 are set, then clear them.

If the system cannot support a test which causes the ERROR pin to go low, then the stuck at zero test can be modified as follows:

- Version B
  - Write boundary register (address 0xFFF8C01C) with 0x003BC000.
  - Read pins register (address 0xFFF8C02C) and verify that bits 14, 12, and 11 are set.
  - Write boundary register (address 0xFFF8C01C) with 0x003C0000, to clear the error signals.
  - Verify that ESM group 1 channel 41 is set, then clear it.

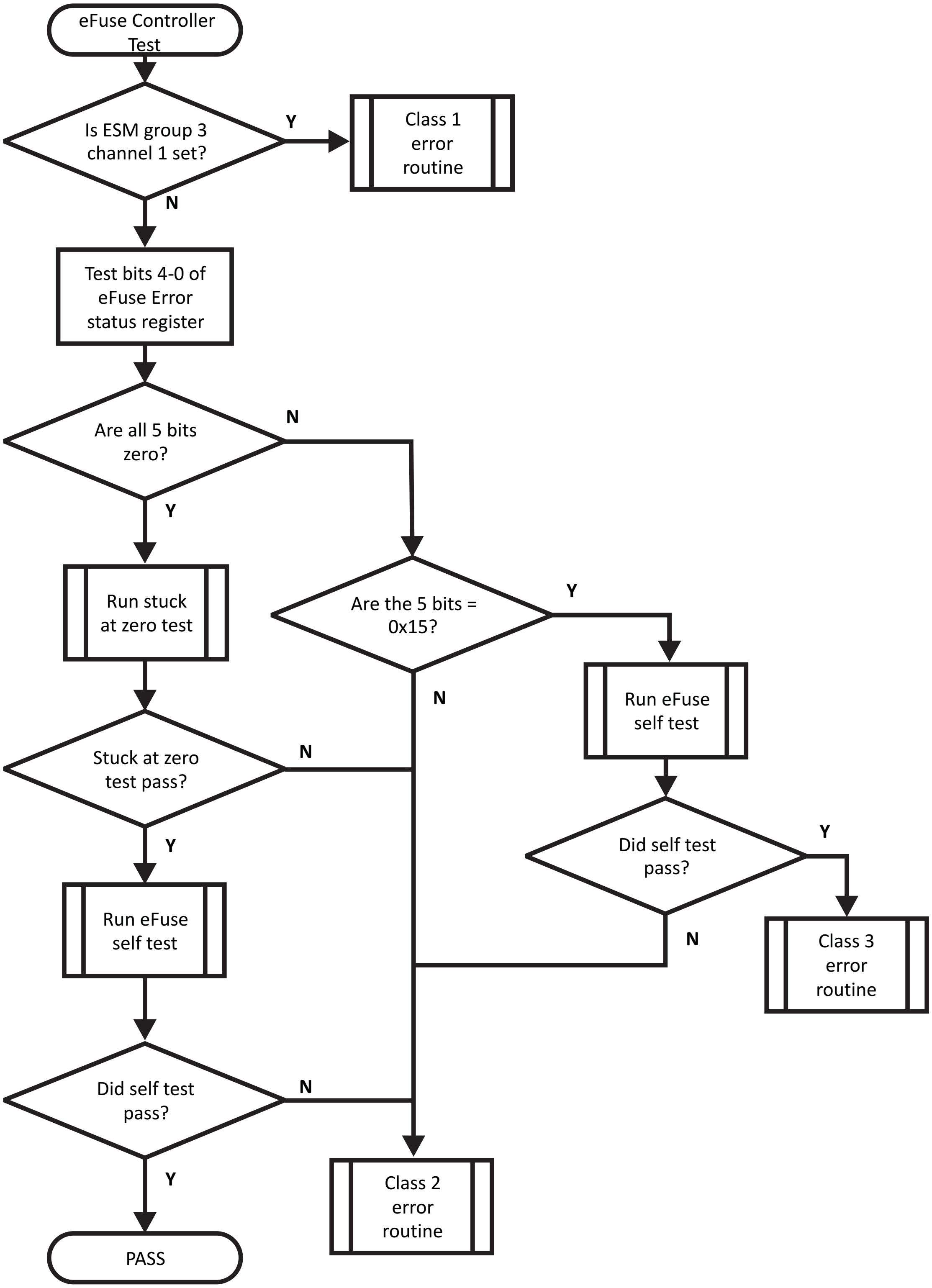
This alternate method provides less test coverage because the path from the uncorrectable error signal from the eFuse controller to the ESM is not specifically tested. However, even if this path is broken, reading the five eFuse error status bits will indicate that an error occurred.

38.3.2.5 eFuse ECC Logic Self Test

The eFuse controller self test performs extensive validation of the ECC logic in the eFuse controller. This test should only be performed once for every device $\overline{\text{PORRST}}$ cycle. Perform the self test by following these steps:

- Write 0x00000258 to the self test cycles register (EFCSTCY) at address 0xFFF8C048.
- Write 0x5362F97F to the self test signature register (EFCSTSIG) at address 0xFFF8C04C.
- Write 0x0000200F to the boundary register at address 0xFFF8C01C. This triggers the self test. The test takes 610 VCLK cycles to complete. The application can poll bit 15 of the pins register at address 0xFFF8C02C to wait for the test to complete.
- Check ESM group 1 channels 40 and 41 for any errors, neither should be set.
- Verify that bits 4 to 0 of the eFuse Error Status register at address 0xFFF8C03C are zero.

Figure 38-1. eFuse Self Test Flow Chart



38.4 eFuse Controller Registers

All registers in the eFuse Controller module are 32-bit, word-aligned; 8-bit, 16-bit and 32-bit accesses are allowed. Table 38-2 provides a quick reference to each of these registers. Specific bit descriptions are discussed in the following subsections. The base address for the control registers is FFF8 C000h.

Table 38-2. eFuse Controller Registers

Offset	Acronym	Register Description	Section
1Ch	EFCBOUND	EFC Boundary Control Register	Section 38.4.1
2Ch	EFCPINS	EFC Pins Register	Section 38.4.2
3Ch	EFCERRSTAT	EFC Error Status Register	Section 38.4.3
48h	EFCSTCY	EFC Self Test Cycles Register	Section 38.4.4
4Ch	EFCSTSIG	EFC Self Test Signature Register	Section 38.4.5

38.4.1 EFC Boundary Control Register (EFCBOUND)

Figure 38-2 and Table 38-3 describe the EFCBOUND register. The eFuse Boundary Control Register is used to test the connections between the eFuse controller and the ESM module. The eFuse Boundary Control Register is also used to initiate an eFuse controller ECC self-test.

Figure 38-2. EFC Boundary Control Register (EFCBOUND) [offset = 1Ch]

31-24
Reserved
R-0

23-22	21	20	19	18	17	16
Reserved	EFC Self Test Error	EFC Single Bit Error	EFC Instruction Error	EFC Autoload Error	Self Test Error OE	Single Bit Error OE
R-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

15	14	13	12-8
Instruction Error OE	Autoload Error OE	EFC ECC Selftest Enable	Reserved
R/W-0	R/W-0	R/W-0	R-0

7-4	3-0
Reserved	Input Enable
R-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after power-on reset (nPORRST)

Table 38-3. EFC Boundary Register (EFCBOUND) Field Descriptions

Bit	Field	Value	Description
31-22	Reserved	0	Read returns 0. Writes have no effect.
21	EFC Self Test Error		This bit drives the self test error signal when bit 17 (Self Test Error OE) is high. This signal is attached to ESM error Group 1, Channel 41.
		0	Drives the self test error signal low, if Self Test OE is high.
		1	Drives the self test error signal high, if Self Test OE is high.
20	EFC Single Bit Error		This bit drives the single bit error signal when bit 16 (Single bit Error OE) is high. This signal is attached to ESM error Group 1, Channel 40.
		0	Drives the self test error signal low, if Single Bit Error OE is high.
		1	Drives the self test error signal high, if Single Bit Error OE is high.

Table 38-3. EFC Boundary Register (EFCBOUND) Field Descriptions (continued)

Bit	Field	Value	Description
19	EFC Instruction Error		This bit drives the instruction error signal when bit 15 (Instruction Error OE) is high. This signal is used to denote an error occurred during e-fuse programming. This signal is not attached to the ESM.
		0	Drives the Instruction Error signal low, if Instruction Error OE is high.
		1	Drives the Instruction Error signal high, if Instruction Error OE is high.
18	EFC Autoload Error		This bit drives the Autoload Error signal when bit 14 (Autoload Error OE) is high. This signal is attached to ESM error Group 3, Channel 1.
		0	Drives the Autoload Error signal low, if Autoload Error OE is high.
		1	Drives the Autoload Error signal high, if Autoload Error OE is high.
17	Self Test Error OE		The Self Test Error Output Enable bit determines if the EFC Self Test signal comes from the eFuse controller or from bit 21 of the boundary register.
		0	EFC Self Test Error comes from eFuse controller.
		1	EFC Self Test Error comes from the boundary register.
16	Single Bit Error OE		The single bit error output enable signal determines if the EFC Single Bit Error signal comes from the eFuse controller or from bit 20 of the boundary register.
		0	EFC Single Bit Error comes from eFuse controller.
		1	EFC Single Bit Error comes from the boundary register.
15	Instruction Error OE		The instruction error output enable signal determines if the EFC Instruction Error signal comes from the eFuse controller or from bit 19 of the boundary register.
		0	EFC Instruction Error comes from eFuse controller.
		1	EFC Instruction Error comes from the boundary register.
14	Autoload Error OE		The autoload error output enable signal determines if the EFC Autoload Error signal comes from the eFuse controller or from bit 18 of the boundary register.
		0	EFC Autoload Error comes from eFuse controller.
		1	EFC Autoload Error comes from the boundary register.
13	EFC ECC Selftest Enable		The eFuse Controller ECC Selftest Enable bit starts the selftest of the ECC logic if the four input enable bits (EFCBOUND[3:0) are all 1s.
		0	No action
		1	Start ECC selftest if EFCBOUND[3:0] are Fh.
12-4	Reserved	0	Read returns 0. Writes have no effect.
3-0	Input Enable		The eFuse Controller ECC Selftest Enable bit starts the selftest of the ECC logic if the four input enable bits (EFCBOUND[3:0) are all 1s.
		Fh	ECC selftest can be started if EFC ECC Selftest Enable, bit 13, is set
		All others	ECC selftest cannot be started.

38.4.2 EFC Pins Register (EFCPINS)

Figure 38-3 and Figure 38-3 describe the EFCPINS register.

Figure 38-3. EFC Pins Register (EFCPINS) [offset = 2Ch]

31–16
Reserved
R-0

15	14	13	12	11	10	9–8
EFC Selftest Done	EFC Selftest Error	Reserved	EFC Single Bit Error	EFC Instruction Error	EFC Autoload Error	Reserved
R-0	R-0	R-0	R-x	R-0	R-x	R-x

7–0
Reserved
R-x

LEGEND: R = Read only; -*n* = value after power-on reset (nPORRST); x = Indeterminate

Table 38-4. EFC Pins Register (EFCPINS) Field Descriptions

Bit	Name	Value	Description
31–16	Reserved	0	Reads return zeros, writes have no effect.
15	EFC Selftest Done		This bit can be polled to determine when the EFC ECC selftest is complete
		0	EFC ECC selftest is not complete.
		1	EFC ECC selftest is complete.
14	EFC Selftest Error		This bit indicates the pass/fail status of the EFC ECC Selftest once the EFC Selftest Done bit (bit 15) is set.
		0	EFC ECC Selftest passed.
		1	EFC ECC Selftest failed.
13	Reserved	0	Reads return zeros. Do NOT write a 1 to this bit.
12	EFC Single Bit Error		This bit indicates if a single bit error was corrected by the ECC logic during the autoload after reset.
		0	No single bit error was detected.
		1	A single bit error was detected and corrected.
11	EFC Instruction Error		This bit indicates an error occurred during a factory test or program operation. This bit should not be set from normal use.
		0	No instruction error detected.
		1	An error occurred during a factory test or program operation.
10	EFC Autoload Error		This bit indicates that some non-correctable error occurred during the autoload sequence after reset. This bit also sets ESM group 3, channel 1.
		0	The autoload function completed successfully.
		1	There were non-correctable errors during the autoload sequence.
9-0	Reserved	0-1	After reset, these bits are indeterminate and reads return either a 1 or 0.

38.4.3 EFC Error Status Register (EFCERRSTAT)

Figure 38-4 and Table 38-5 describe the EFCERRSTAT register.

Figure 38-4. EFC Error Status Register (EFCERRSTAT) [offset = 3Ch]

31	8
Reserved	
R-0	

7	6	5	4	0
Reserved		Instruc Done	Error Code	
R-0		R/W-0	R/W-0	

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after power-on reset (nPORRST)

Table 38-5. EFC Error Status Register (EFCERRSTAT) Field Descriptions

Bit	Name	Value	Description
31–6	Reserved	0	Reads return zeros, writes have no effect.
5	Instruc Done		Instruction done. Used to indicate that the eFuse self test has completed
		0	The eFuse controller is still executing.
		1	The eFuse controller has completed executing.
4-0	Error Code		The error status of the last instruction executed by the eFuse Controller
		0	No error.
		5h	An uncorrectable (multibit) error was detected during the power-on autoload sequence.
		15h	At least one single bit error was detected and corrected during the power-on autoload sequence.
		18h	The signature generated by the ECC self-test logic did not match the golden signature written in the EFCSTSIG register. The EDAC circuitry might have a fault.
		All other values	All other values are reserved for e-fuse system tests and are not expected to occur in normal system use.

38.4.4 EFC Self Test Cycles Register (EFCSTCY)

Figure 38-5 and Table 38-6 describe the EFCSTCY register.

Figure 38-5. EFC Self Test Cycles Register (EFCSTCY) [offset = 48h]

31	16
Cycles	
R/W-0	

15	0
Cycles	
R/W-0	

LEGEND: R/W = Read/Write; -*n* = value after power-on reset (nPORRST)

Table 38-6. EFC Self Test Cycles Register (EFCSTCY) Field Descriptions

Bit	Name	Description
31–0	Cycles	This register is used to determine the number of cycles to run the eFuse controller ECC logic self test. It is recommended to use a value of 600 (0x00000258).

38.4.5 EFC Self Test Signature Register (EFCSTSIG)

Figure 38-6 and Table 38-7 describe the EFCSTSIG register.

Figure 38-6. EFC Self Test Cycles Register (EFCSTSIG) [offset = 4Ch]

31	16
Signature	
R/W-0	

15	0
Signature	
R/W-0	

LEGEND: R/W = Read/Write; -*n* = value after power-on reset (nPORRST)

Table 38-7. EFC Self Test Cycles Register (EFCSTSIG) Field Descriptions

Bit	Name	Description
31–0	Signature	This register is used to hold the expected signature for the eFuse ECC logic self test. It is recommended to write a value of 0x5362F97F to this register and a value of 600 (0x00000258) to the EFCSTCY register. If after running the eFuse ECC logic self test, the calculated signature does not match the expected signature in the EFCSTSIG register, then a value of 18h is stored in the EFCERRSTAT register.

Revision History

Changes from May 20, 2014 to February 28, 2018 **Page**

- **Chapter 1: Introduction** 106
- **Chapter 2: Architecture** 112
- Table 2-1: Rearranged sequence of terms 115
- Figure 2-3: Updated figure. Changed R5F-1 Cache to RESERVED 121
- Table 2-6: Changed table. Corrected values in Valid RAM Groups and Valid RINFOL/RINFOU Register Value columns 136
- Table 2-10: Changed VCLKA4_S to VCLKA4 143
- Table 2-10: Changed RTICLK to RTICLK1 143
- Table 2-10: Changed Special Considerations description of VCLKA1, VCLKA2, and VCLKA4. Deleted Frequency can be as fast as HCLK frequency 143
- Table 2-10: Changed Special Considerations description of VCLKA4_DIVR. Changed VCLKA4_S to VCLKA4 143
- Table 2-12: Updated signal names 147
- Table 2-31: Changed Description of bits for Value = 0 (Read) to enabled 160
- Section 2.5.1.13: Added second paragraph to NOTE 161
- Table 2-35: Changed Description of GHVSRC bit. Removed "on wakeup" 167
- Table 2-41: Updated MSTGENA and MINITGENA values to Ah for MSIENA = 1 173
- Table 2-44: Corrected Description of PLLMUL bit. Value = 0h is ×1, Value = 100h is ×2 176
- Figure 2-35: Corrected register bit fields 179
- Table 2-47: Changed table to reflect updated register bit fields 179
- Figure 2-36: Corrected register bit fields 179
- Table 2-48: Changed table to reflect updated register bit fields 179
- Table 2-49: Changed Description of OSCFRQCONFIGCNT bit. Writes have no effect 180
- Figure 2-38: Changed Reserved bits to 7-5 and SEL_ECP_PIN bits to 4-0 183
- Table 2-50: Changed Reserved bits to 7-5 and SEL_ECP_PIN bits to 4-0 183
- Table 2-50: Changed Description of SEL_GIO_PIN and SEL_ECP_PIN bits 183
- Table 2-53: Changed Description of PLL1_FBSLIP_FILTER_ COUNT and PLL1_FBSLIP_FILTER_ KEY bits 187
- Section 2.5.1.41: Changed paragraph 194
- Section 2.5.1.41: Added NOTE 194
- Section 2.5.1.42: Changed NOTE 195
- Table 2-61: Added Note to VCLK2R and VCLKR bits 195
- Section 2.5.1.45: Added NOTE 197
- Figure 2-53: Changed bit 12 to Reserved 198
- Figure 2-53: Changed Reserved bits to 2-0 198
- Figure 2-53: Deleted MPMODE bit 198
- Table 2-65: Changed Description of WDRST bit 198
- Table 2-65: Changed bit 12 to Reserved 198
- Table 2-65: Changed Reserved bits to 2-0 198
- Table 2-65: Deleted MPMODE bit 198
- Figure 2-63: Changed bits 10-8 and 4-0 to Reserved 210
- Table 2-76: Changed bits 10-8 and 4-0 to Reserved 210
- Table 2-76: Changed Value column of Reserved bits 15-0 to 109h 210
- Table 2-76: Changed Description of VCLKA4S bit for Value = 8h-Fh 210
- Table 2-78: Changed Description of PLL1_RFSLIP_FILTER_COUNT and PLL1_RFSLIP_FILTER_KEY bits 212
- Section 2.5.2.10: Changed paragraph 214
- Figure 2-68: Corrected register bit fields 214
- Table 2-81: Changed table to reflect updated register bit fields 214
- Section 2.5.2.11: Changed paragraph 215
- Figure 2-69: Corrected register bit fields 215
- Table 2-82: Changed table to reflect updated register bit fields 215

- Table 2-83: Changed Description of DIEIDL2 bit. Added last sentence 215
- Table 2-84: Changed Description of DIEIDH2 bit. Added last sentence 216
- Figure 2-76: Changed register bit name to PS[7-0]QUAD[3-0]PROTSET 225
- Table 2-90: Changed register bit name to PS[7-0]QUAD[3-0]PROTSET 225
- Table 2-90: Corrected register names in Description of PROTSET bit for Value = 1 (Write) 225
- Figure 2-77: Changed register bit name to PS[15-8]QUAD[3-0]PROTSET 226
- Table 2-91: Changed register bit name to PS[15-8]QUAD[3-0]PROTSET 226
- Table 2-91: Corrected register names in Description of PROTSET bit for Value = 1 (Write) 226
- Figure 2-78: Changed register bit name to PS[23-16]QUAD[3-0]PROTSET 226
- Table 2-92: Changed register bit name to PS[23-16]QUAD[3-0]PROTSET 226
- Table 2-92: Corrected register names in Description of PROTSET bit for Value = 1 (Write) 226
- Figure 2-79: Changed register bit name to PS[31-24]QUAD[3-0]PROTSET 227
- Table 2-93: Changed register bit name to PS[31-24]QUAD[3-0]PROTSET 227
- Table 2-93: Corrected register names in Description of PROTSET bit for Value = 1 (Write) 227
- Figure 2-80: Changed register bit name to PS[7-0]QUAD[3-0]PROTCLR 227
- Table 2-94: Changed register bit name to PS[7-0]QUAD[3-0]PROTCLR 227
- Table 2-94: Corrected register names in Description of PROTCLR bit for Value = 1 (Write) 227
- Figure 2-81: Changed register bit name to PS[15-8]QUAD[3-0]PROTCLR 228
- Table 2-95: Changed register bit name to PS[15-8]QUAD[3-0]PROTCLR 228
- Table 2-95: Corrected register names in Description of PROTCLR bit for Value = 1 (Write) 228
- Figure 2-82: Changed register bit name to PS[23-16]QUAD[3-0]PROTCLR 228
- Table 2-96: Changed register bit name to PS[23-16]QUAD[3-0]PROTCLR 228
- Table 2-96: Corrected register names in Description of PROTCLR bit for Value = 1 (Write) 228
- Figure 2-83: Changed register bit name to PS[31-24]QUAD[3-0]PROTCLR 229
- Table 2-97: Changed register bit name to PS[31-24]QUAD[3-0]PROTCLR 229
- Table 2-97: Corrected register names in Description of PROTCLR bit for Value = 1 (Write) 229
- Section 2.5.3.30: Added paragraphs 240
- Section 2.5.3.30: Added NOTE 240
- Section 2.5.3.31: Added paragraph 242
- Section 2.5.3.32: Added paragraph 243
- **Chapter 3: SCR Control Module (SCM)** 252
- Figure 3-1: Added footnote 254
- Figure 3-2: Added footnote 255
- **Chapter 4: Interconnect** 265
- Table 4-1: Changed table. Added Access Mode column 266
- Table 4-3: Changed table. Added Access Mode column 268
- Table 4-3: Added footnote 268
- **Chapter 5: Power Management Module (PMM)** 279
- Figure 5-16: Updated Read/Write value of LCMPE bits to R/W1CP-0 298
- **Chapter 6: I/O Multiplexing and Control Module (IOMM)** 301
- Section 6.2: Deleted second bullet 302
- Section 6.3: Changed last sentence of third paragraph 303
- Table 6-1: Added N2HET1_NDIS at address 198h, ball D8, for Alternate Function 1 and 34[25] for Selection Bit 304
- Table 6-1: Added N2HET2_NDIS at address 19Ch, ball D7, for Alternate Function 1 and 35[1] for Selection Bit 304
- Table 6-1: Added footnote 304
- Table 6-5: Corrected PINMMR163 bit number in Control Option B column for Events 6, 7, and 8 315
- Section 6.5.6: Corrected terminal names in first two sentences of fourth paragraph 319
- Section 6.5.6: Corrected first sentence of fifth paragraph. GIO module has four sources 319
- Figure 6-6: Corrected terminal names of first two terminals 319
- Table 6-9: Corrected bit value in middle column to = 1 323
- Section 6.5.13: Added NOTE 326
- Section 6.6.2: Deleted subsection Master ID Check. Subsequent subsection renumbered 327
- Section 6.7.3: Addd paragraph 330

- Figure 6-12: Updated Read/Write value of KICK0 bits to R/W-0 330
- Figure 6-12: Changed LEGEND 330
- Section 6.7.4: Added paragraph 330
- Figure 6-13: Updated Read/Write value of KICK1 bits to R/W-0 330
- Figure 6-13: Changed LEGEND 330
- **Chapter 7: F021 Level 2 Flash Module Controller (L2FMC)** 338
- Table 7-1: Corrected for big endian 342
- Section 7.5.1: Changed sentence 345
- Figure 7-1: Changed figure 345
- Figure 7-2: Added figure. Subsequent figures renumbered 345
- Figure 7-7: Switched the order of the temperature sensor ADC reading and the temperature in the OTP 348
- Figure 7-8: Switched the order of the temperature sensor ADC reading and the temperature in the OTP 348
- Figure 7-9: Switched the order of the temperature sensor ADC reading and the temperature in the OTP 348
- Table 7-8: Switched the order of the temperature sensor ADC reading and the temperature in the OTP 349
- Table 7-8: Changed Description of SxTEMP1, SxTEMP2, and SxTEMP3 fields 349
- Section 7.5.2.6: Moved subsection into OTP Memory subsection. Subsequent subsections renumbered 349
- Section 7.7.2: Updated second paragraph 350
- Section 7.7.2.1: Updated second paragraph 351
- Table 7-10: Deleted DIAG MODE 6 352
- Table 7-10: Updated Name of test modes 352
- Table 7-12: Added Read Margin Control Register (FSPRD) 355
- Section 7.10.2: Added subsection. Subsequent subsections, figures, and tables renumbered 357
- Figure 7-26: Changed default value of PSLEEP bit to C8h 368
- Table 7-45: Updated Description of SECT_ERASED bit 381
- Table 7-46: Updated Description of SECT_ERASED bit 381
- **Chapter 8: Level 2 RAM (L2RAMW) Module** 387
- Table 8-4: Corrected Description of DWDE bit to double-bit error 395
- Table 8-8: Changed Description of TEST_MODE and TEST_ENABLE bits 400
- Figure 8-8: Changed bits 15-0 to RAM_CHIP_SELECT 401
- **Chapter 9: Programmable Built-In Self-Test (PBIST) Module** 405
- Figure 9-2: Changed figure. Deleted FSRF1 408
- Section 9.3.1: Changed code example 409
- Section 9.3.1: Changed step 6. Deleted ROM interface clock 409
- Section 9.3.1: Changed step 12. Deleted FSRF1 and ROM clock 409
- Section 9.3.1: Added last sentence with link to last paragraph 410
- Section 9.5: Changed second paragraph (write 1h) 412
- Table 9-1: Deleted FSRF1 412
- Table 9-2: Added Note to RDS bit 413
- Section 9.5.3: Updated paragraph to remove bit [1] 415
- Section 9.5.3: Deleted PACT1 bullet 415
- Figure 9-5: Deleted bit [1] 415
- Table 9-4: Deleted bit [1] 415
- Section 9.5.6: Changed subsection title. Deleted FSRF1 418
- Section 9.5.6: Changed paragraph 418
- Section 9.5.6: Deleted Fail Status Fail Register 1 (FSRF1) figure. Subsequent figures renumbered 418
- Section 9.5.6: Deleted Fail Status Fail Register 1 (FSRF1) Field Descriptions table. Subsequent tables renumbered.. 418
- Section 9.6.1: Changed step 5. Deleted ROM interface clock 426
- Section 9.6.1: Changed step 12. Deleted FSRF1 and ROM clock 426
- Section 9.6.2: Changed step 5. Deleted ROM interface clock 427
- Section 9.6.2: Changed step 11. Deleted FSRF1 and ROM clock 427
- **Chapter 10: Self-Test Controller (STC) Module** 428
- Table 10-2: Changed format 441
- Table 10-4: Changed format 444

- Table 10-6: Changed format 444
- Table 10-9: Changed Description of RS_CNT bit. Added values 2h-3h 447
- Section 10.8.2: Added second sentence to NOTE 448
- **Chapter 11: System Memory Protection Unit (NMPU)** 460
- Table 11-3: Deleted MPU Input Address Mask Register (MPUIAM) 471
- Figure 11-8: Updated Read/Write value of ERRFLAG bit to R/W1CP-0 474
- **Chapter 12: Error Profiling Controller (EPC)** 483
- Figure 12-4: Updated Read/Write value of bits to R/W1CP-0 491
- Figure 12-5: Updated Read/Write value of bits to R/W1CP-0 492
- Figure 12-6: Updated Read/Write value of bits to R/W1CP-0 493
- Figure 12-7: Updated Read/Write value of bits to R/W1CP-0 494
- **Chapter 13: CPU Compare Module for Cortex-R5F (CCM-R5F)** 497
- **Chapter 14: Oscillator and PLL** 517
- Section 14.5: Updated third paragraph. Changed $f_{(HCLK)}$ to $f_{(GCLK)}$ 525
- Table 14-1: Updated Frequency Limit value to $f_{(GCLK)}$for $f_{PLL\ CLK}$ 525
- Table 14-2: Updated formula for NF 526
- Section 14.5.2.1: Changed table of step 2. Updated lock phase time formula to (512 × T_{OSCIN}) 528
- Section 14.5.2.2: Added formula to last sentence of second paragraph: T_{Enable} = 150 × T_{OSCIN} 529
- Section 14.5.2.2: Deleted table 529
- Section 14.5.2.3: Added formula to last sentence of paragraph: T_{ODPLL} = 3 × T_{OSCIN} 529
- Section 14.5.2.3: Deleted table 529
- Table 14-3: Changed table title. Changed table format 530
- Table 14-3: Updated lock phase time formula to (512 × T_{OSCIN}) 530
- Section 14.5.4: Added last sentence to step 3 in both paragraphs 532
- Figure 14-11: Corrected symbols in figure 539
- Figure 14-11: Changed PF block to CP 539
- Section 14.8: Changed step 3, second sentence 540
- **Chapter 15: Dual-Clock Comparator (DCC) Module** 542
- Figure 15-12: Updated Read/Write value of DONE and ERR bits to R/W1CP-0 553
- Figure 15-12: Updated LEGEND to include W1CP 553
- Table 15-8: Corrected table title 554
- Table 15-11: Corrected table title 556
- Table 15-11: Changed Description of KEY bit for Value = Any other value 556
- **Chapter 16: Error Signaling Module (ESM)** 558
- Section 16.1.2: Changed first paragraph 559
- Figure 16-17: Updated Read/Write value of bits to R/W1CP-X/0 569
- Figure 16-18: Updated Read/Write value of bits to R/W1CP-0 569
- Figure 16-19: Updated Read/Write value of bits to R/W1CP-X/0 570
- Figure 16-26: Updated Read/Write value of bits to R/W1CP-X/0 574
- Figure 16-33: Updated Read/Write value of bits to R/W1CP-X/0 578
- Figure 16-40: Updated Read/Write value of bits to R/W1CP-X/0 582
- **Chapter 17: Real-Time Interrupt (RTI) Module** 583
- Equation 24: Corrected first eqution (if RTICPUCy ≠ 0) 587
- Section 17.2.5.1: Changed first paragraph 592
- Figure 17-18: Updated Read/Write value of bits to R/WP-0 601
- Figure 17-18: Updated LEGEND to include WP 601
- Figure 17-38: Updated Read/Write value of bits to R/W1CP-0 615
- Figure 17-41: Updated Read/Write value of bits to R/W1CP-0 618
- **Chapter 18: Cyclic Redundancy Check (CRC) Controller Module** 625
- Figure 18-3: Changed MCRC Controller to CRC Controller 632
- Figure 18-4: Changed MCRC Controller to CRC Controller 632
- **Chapter 19: Vectored Interrupt Manager (VIM) Module** 662
- Section 19.5: Added NOTE 672

- Figure 19-14: Corrected reset value of Interrupt Vector Table offset bits 15-9 683
- Table 19-8: Corrected Description of Interrupt Vector Table offset bits. Reads are always FFF8 2xxxh 683
- Section 19.9.11: Updated LEGEND to include WP 688
- Figure 19-31: Updated LEGEND to include WP 689
- Table 19-17: Corrected table title 689
- **Chapter 20: Direct Memory Access Controller (DMA) Module** 696
- Chapter 20: Global: Changed index pointer to offset value 696
- Section 20.1.1: Updated eighth bullet 697
- Section 20.2.2: Changed second bullet 700
- Section 20.2.3: Deleted last sentence in second paragraph 701
- Figure 20-5: Updated figure (changed Index Pointer to Offset Value) 702
- Section 20.2.7: Corrected second bullet. in first paragraph. The DMA controller can handle up to 48 DMA Request lines 710
- Section 20.2.7: Added last two paragraphs 710
- Table 20-3: Added table. Subsequent tables renumbered 710
- Section 20.2.9: Deleted fifth bullet (Bus error (BER) interrupt) 712
- Section 20.2.9: Added fifth bullet (External imprecise error on read) 712
- Section 20.2.9: Added sixth bullet (External imprecise error on write) 712
- Section 20.2.9: Changed NOTE. Deleted BER references 712
- Figure 20-14: Deleted BERA error signal. Added SCR block 713
- Figure 20-15: Changed output of OR gate to FTCA 713
- Figure 20-15: Changed footnote. Deleted BER reference 713
- Section 20.2.12Changed fifth paragraph 715
- Section 20.2.18: Changed second paragraph 719
- Table 20-7: Deleted BER Interrupt Mapping Register (BERMAP), BERA Interrupt Channel Offset Register (BERAOFFSET), and BERB Interrupt Channel Offset Register (BERBOFFSET). Subsequent subsections, figures, and tables renumbered 721
- Table 20-9: Updated Description of DMA_RES bit. (Writing a zero to this bit has no effect.) 724
- Figure 20-23: Updated Read/Write value of HWCHENA bit to R/WP-0 727
- Figure 20-24: Updated Read/Write value of HWCHDIS bit to R/WP-0 727
- Figure 20-25: Updated Read/Write value of SWCHENA bit to R/WP-0 728
- Figure 20-26: Updated Read/Write value of SWCHDIS bit to R/WP-0 728
- Figure 20-27: Updated Read/Write value of CPS bit to R/WP-0 729
- Figure 20-28: Updated Read/Write value of CPR bit to R/WP-0 729
- Figure 20-29: Updated Read/Write value of GCHIE bit to R/WP-0 730
- Figure 20-30: Updated Read/Write value of GCHID bit to R/WP-0 730
- Table 20-21: Changed Description of all bits for Value = 2Fh. DMA request line 47 triggers channel 731
- Table 20-21: Updated Value column for all bits. Added 30h-3Fh = Reserved 731
- Table 20-22: Changed Description of all bits for Value = 2Fh. DMA request line 47 triggers channel 732
- Table 20-22: Updated Value column for all bits. Added 30h-3Fh = Reserved 732
- Table 20-23: Changed Description of all bits for Value = 2Fh. DMA request line 47 triggers channel 733
- Table 20-23: Updated Value column for all bits. Added 30h-3Fh = Reserved 733
- Table 20-24: Changed Description of all bits for Value = 2Fh. DMA request line 47 triggers channel 734
- Table 20-24: Updated Value column for all bits. Added 30h-3Fh = Reserved 734
- Table 20-25: Changed Description of all bits for Value = 2Fh. DMA request line 47 triggers channel 735
- Table 20-25: Updated Value column for all bits. Added 30h-3Fh = Reserved 735
- Table 20-26: Changed Description of all bits. Corrected channel number 736
- Table 20-26: Changed Description of all bits for Value = 2Fh. DMA request line 47 triggers channel 736
- Table 20-26: Updated Value column for all bits. Added 30h-3Fh = Reserved 736
- Table 20-27: Changed Description of all bits. Corrected channel number 737
- Table 20-27: Changed Description of all bits for Value = 2Fh. DMA request line 47 triggers channel 737
- Table 20-27: Updated Value column for all bits. Added 30h-3Fh = Reserved 737
- Table 20-28: Changed Description of all bits for Value = 2Fh. DMA request line 47 triggers channel 738
- Table 20-28: Updated Value column for all bits. Added 30h-3Fh = Reserved 738

- Table 20-29: Changed Description of all bits. Refer to CH0PA for bit value descriptions 739
- Table 20-30: Changed Description of all bits. Refer to CH8PA for bit value descriptions 740
- Table 20-31: Changed Description of all bits. Refer to CH16PA for bit value descriptions 741
- Table 20-32: Changed Description of all bits. Refer to CH24PA for bit value descriptions 742
- Table 20-44: Changed Description of BTCINTDIS bit for Value = 0 (Read). Channel is disabled 748
- Table 20-44: Changed Description of BTCINTDIS bit for Value = 1 (Read). Channel is enabled 748
- Table 20-45: Updated Description of GINT bit. Deleted BER references 749
- Figure 20-56: Updated Read/Write value of FTC bit to R/W1CP-0 749
- Figure 20-57: Updated Read/Write value of LFS bit to R/W1CP-0 750
- Figure 20-58: Updated Read/Write value of HBC bit to R/W1CP-0 750
- Figure 20-59: Updated Read/Write value of BTC bit to R/W1CP-0 751
- Section 20.3.1.42: Added paragraph 751
- Section 20.3.1.42: Deleted BER Interrupt Flag Register (BERFLAG) figure and BER Interrupt Flag Register (BERFLAG) Field Descriptions table. Subsequent figures and tables renumbered 751
- Table 20-50: Updated Value column of FTCA bit. Added 21h-3Fh = Reserved 752
- Table 20-51: Updated Value column of LFSA bit. Added 21h-3Fh = Reserved 753
- Table 20-52: Updated Value column of HBCA bit. Added 21h-3Fh = Reserved 754
- Table 20-53: Updated Value column of BTCA bit. Added 21h-3Fh = Reserved 755
- Table 20-54: Updated Value column of FTCB bit. Added 21h-3Fh = Reserved 756
- Table 20-55: Updated Value column of LFSB bit. Added 21h-3Fh = Reserved 757
- Table 20-56: Updated Value column of HBCB bit. Added 21h-3Fh = Reserved 758
- Table 20-57: Updated Value column of BTCB bit. Added 21h-3Fh = Reserved 759
- Figure 20-70: Updated Read/Write value of DMADBGS bit to R/W1C-0 762
- Figure 20-80: Updated Read/Write value of EDFLAG bit to R/W1C-0 767
- Figure 20-82: Updated Read/Write value of bits to R/W1C-0 770
- Table 20-72: Changed Description of bits. Added 0 (Write) = No effect 770
- Table 20-73: Updated Description of STARTADDRESS bit. Added last sentence 771
- Table 20-74: Updated Description of ENDADDRESS bit. Added last two sentences 771
- Table 20-74: Added Note 771
- Table 20-75: Updated Description of STARTADDRESS bit. Added last sentence 772
- Table 20-76: Updated Description of ENDADDRESS bit. Added last two sentences 772
- Table 20-76: Added Note 772
- Table 20-77: Updated Description of STARTADDRESS bit. Added last sentence 773
- Table 20-78: Updated Description of ENDADDRESS bit. Added last two sentences 773
- Table 20-78: Added Note 773
- Table 20-79: Updated Description of STARTADDRESS bit. Added last sentence 774
- Table 20-80: Updated Description of ENDADDRESS bit. Added last two sentences 774
- Table 20-80: Added Note 774
- Figure 20-92: Updated Read/Write value of bits to R/W1C-0 777
- Table 20-82: Changed Description of bits. Added 0 (Write) = No effect 777
- Table 20-83: Updated Description of STARTADDRESS bit. Added last sentence 778
- Table 20-84: Updated Description of ENDADDRESS bit. Added last two sentences 778
- Table 20-84: Added Note 778
- Table 20-85: Updated Description of STARTADDRESS bit. Added last sentence 779
- Table 20-86: Updated Description of ENDADDRESS bit. Added last two sentences 779
- Table 20-86: Added Note 779
- Table 20-87: Updated Description of STARTADDRESS bit. Added last sentence 780
- Table 20-88: Updated Description of ENDADDRESS bit. Added last two sentences 780
- Table 20-88: Added Note 780
- Table 20-89: Updated Description of STARTADDRESS bit. Added last sentence 781
- Table 20-90: Updated Description of ENDADDRESS bit. Added last two sentences 781
- Table 20-90: Added Note 781
- Figure 20-101: Updated Read/Write value of SBERR bit to R/W1CP-0 782
- Table 20-91: Changed Description of SBERR bit. Added 0 (Write) = No effect 782

- Figure 20-107: Updated Read/Write value of TER_ERR bit to R/W1C-0 786
- Figure 20-108: Updated Read/Write value of TERE bit to R/W1CP-0 786
- Table 20-99: Updated Value column of TER_OFF bit. Added 21h-3Fh = Reserved 787
- Table 20-103: Changed Description of TTYPE bit. (A request triggers) 790
- **Chapter 21: External Memory Interface (EMIF)** 793
- Figure 21-1: Corrected pin names 795
- Figure 21-1: Deleted EMIF_RNW pin 795
- Table 21-3: Deleted EMIF_RNW pin 797
- Figure 21-18: Updated reset value of RR bit to 80h 831
- Figure 21-24: Changed bit 1 to LT_MASK_SET 837
- Figure 21-25: Changed bit 1 to LT_MASK_CLR 838
- Figure 21-27: Corrected pin names 841
- **Chapter 22: Analog To Digital Converter (ADC) Module** 848
- Figure 22-1: Corrected EXT_SEL and EXT_ENA signals 850
- Figure 22-2: Corrected Event Trigger Generation signal 851
- Figure 22-2: Corrected EXT_SEL and EXT_ENA signals 851
- Figure 22-2: Deleted EXT_nENA signal 851
- Section 22.1.1.3: Updated third and fifth sentences in third paragraph 852
- Section 22.2.1.3: Added last sentence (reference) to fourth paragraph 854
- Figure 22-10: Corrected bit range of the EV_CURRENT_COUNT, EV_MAX_COUNT, G1_CURRENT_COUNT, G1_MAX_COUNT, G2_CURRENT_COUNT, and G2_MAX_COUNT bits to 4-0 860
- Section 22.2.5.3: Corrected register names in first sentence. (ADMAGINTENASET and ADMAGINTENACLR) 870
- Table 22-11: Changed Description of FRZ_EV bit. (The Event Group conversion is kept frozen while the Group1 or Group2 conversion is active,) 888
- Table 22-12: Changed Description of FRZ_G1 bit. (The Group1 conversion is kept frozen while the Event Group or Group2 conversion is active,) 891
- Table 22-13: Changed Description of FRZ_G2 bit. (The Group2 conversion is kept frozen while the Event Group or Group1 conversion is active,) 894
- Table 22-28: Changed Description for DMA_G2_END bit. Corrected group number to 2 911
- Figure 22-53: Deleted Reserved bits. Changed EV_SEL bits to 31-0 920
- Table 22-37: Deleted Reserved bits. Changed EV_SEL bits to 31-0 920
- Figure 22-54: Deleted Reserved bits. Changed G1_SEL bits to 31-0 921
- Table 22-38: Deleted Reserved bits. Changed G1_SEL bits to 31-0 921
- Figure 22-55: Deleted Reserved bits. Changed G2_SEL bits to 31-0 922
- Table 22-39: Deleted Reserved bits. Changed G2_SEL bits to 31-0 922
- Table 22-42: Changed Description of LAST_CONV bit for Value = 1. (A level higher than or equal to the midpoint reference voltage) 924
- **Chapter 23: High-End Timer (N2HET) Module** 953
- Section 23.2.5.4: Changed first sentence 973
- Table 23-9: Changed Pull Control and Input Buffer = Enabled when device is under reset 985
- Section 23.2.9: Updated first paragraph 990
- Section 23.3.2: Added Hardware Angle Generator (HWAG) subsection. Subsequent figures and tables renumbered .. 995
- Figure 23-56: Changed format 1018
- Table 23-16: Updated Description of PPF and TO bits 1018
- Table 23-17: Updated Description of LRPFC and HRPFC bits 1020
- Table 23-26: Changed Description of HETPRY bit 1026
- Section 23.5: Added HWAG Registers section. Subsequent section, figures, and tables renumbered 1044
- Table 23-73: Added cross references to instruction descriptions 1060
- Table 23-73: Added OR instruction 1060
- Table 23-73: Corrected sub-opcodes for ADC, ADD, and XOR instructions 1060
- Table 23-74: Added SUB to Set/Reset column for Zero flag (Z) 1061
- Section 23.6.3.8: Updated Description of CNT instruction. The data field [D31:7] is incremented unconditionally on each execution of the instruction 1086
- Table 23-87: Changed registers in Source and Destination(s) columns to register A, B, R, S, or T 1103
- Section 23.6.3.19: Updated Description of RCNT instruction. For example, choosing M = 100 allows the input period to be

expressed as a percentage (%) of the reference period 1118
- **Chapter 24: High-End Timer Transfer Unit (HTU) Module** 1131
- Section 24.2: Updated third bullet in fifth paragraph. Added per request 1134
- Section 24.2: Added fourth bullet in fifth paragraph 1134
- Section 24.2.4.1: Updated first sentence in third paragraph 1141
- Section 24.2.4.1: Updated fourth sentence in seventh paragraph. If the signal frequency would increase, then a wrong pair [22,23] could be read 1142
- Section 24.2.5: Updated second paragraph 1143
- Figure 24-13: Changed position of third rising edge in waveform 1145
- Table 24-24: Corrected bit range of the Reserved bit to 31-10 and the INTTYPE0 bit to 9-8 1158
- Table 24-25: Corrected bit range of the Reserved bit to 31-10 and the INTTYPE1 bit to 9-8 1159
- Figure 24-31: Changed 2 LSBs to 0 1164
- Table 24-31: Updated Description of STARTADDRESS1 bit. Added last sentence 1164
- Figure 24-32: Changed 2 LSBs to 0 1164
- Table 24-32: Updated Description of ENDADDRESS1 bit. Added last sentences 1164
- Figure 24-40: Changed 2 LSBs to 0 1173
- Table 24-40: Updated Description of STARTADDRESS0 bit. Added last sentence 1173
- Figure 24-41: Changed 2 LSBs to 0 1173
- Table 24-41: Updated Description of ENDADDRESS0 bit. Added last sentences 1173
- Section 24.5.7: Updated paragraph 1180
- Table 24-49: Corrected table title 1180
- **Chapter 25: General-Purpose Input/Output (GIO) Module** 1183
- Section 25.2: Corrected register names in second paragraph to GIOOFF1 and GIOOFF2 1185
- Figure 25-3: Moved figure location. Subsequent figures renumbered 1187
- Section 25.3.1: Changed description of Data direction in first bullet 1187
- Section 25.3.1: Changed description of Open drain in sixth bullet 1187
- Section 25.3.1: Changed description of Pull select in eighth bullet 1188
- Section 25.4.2: Changed last sentence 1190
- Section 25.5.1: Chnaged third sentence 1192
- Section 25.5.2: Changed paragraph 1193
- Section 25.5.3: Changed paragraph 1194
- Section 25.5.4: Changed paragraph 1195
- Section 25.5.4.1: Changed NOTE. Deleted first sentence 1195
- Section 25.5.5: Changed paragraph 1197
- Section 25.5.5: Added NOTE 1197
- Section 25.5.6: Added second sentence 1200
- Figure 25-12: Updated Read/Write value of bits to R/W1C-0 1200
- Figure 25-12: Updated LEGEND to include W1C 1200
- Table 25-22: Changed Pull Control = Enabled when device is under reset 1209
- **Chapter 26: FlexRay Module** 1210
- Figure 26-30: Updated figure to include where interrupts are sent 1270
- Figure 26-31: Updated title 1272
- Section 26.3.1.12: Updated register bit names to reflect corresponding message buffer number 1288
- Table 26-30: Combined Transfer to System Memory Occurred (TSMO[1-4]) Field Descriptions tables into a single table 1289
- Section 26.3.1.13: Updated register bit names to reflect corresponding message buffer number 1290
- Table 26-31: Combined Transfer to Communication Controller Occurred (TCCO[1-4]) Field Descriptions tables into a single table. 1291
- Section 26.3.1.16: Corrected NOTE 1294
- Figure 26-56: Corrected register bit name for bits 10-8 and bits 6-4 1297
- Figure 26-57: Corrected register bit name for bits 10-8 and bits 6-4 1298
- Section 26.3.1.19: Updated register bit names to reflect corresponding message buffer number 1299
- Section 26.3.1.20: Updated register bit names to reflect corresponding message buffer number 1303
- Section 26.3.1.21: Updated register bit names to reflect corresponding message buffer number 1307

- Section 26.3.1.22: Updated register bit names to reflect corresponding message buffer number 1311
- Section 26.3.1.23: Updated register bit names to reflect corresponding message buffer number 1315
- Section 26.3.1.24: Updated register bit names to reflect corresponding message buffer number 1319
- Section 26.3.2.2.5: Updated paragraph 1350
- Section 26.3.2.2.6: Updated paragraph 1352
- **Chapter 27: Controller Area Network (DCAN) Module** 1417
- Equation 38: Updated equation 1424
- Equation 39: Updated equation 1424
- Section 27.3.2.4: Changed t_q = 1 µs 1425
- Equation 40: Updated equation 1425
- Equation 41: Updated equation 1425
- Section 27.16: Changed subsection 1454
- Table 27-6: Added Core Release Register 1454
- Figure 27-21: Updated Read/Write value of PER bit to R/C-0 1459
- Table 27-9: Updated Value column of REC bit to 0-7Fh 1461
- Table 27-9: Updated Description of REC bit (Values from 0 to 127) 1461
- Table 27-11: Corrected Value column range of Int1ID and Int0ID bits to 1h-40h 1463
- Table 27-13: Corrected Value column range of Message Number bit to 1h-FFh 1465
- Table 27-13: Updated Description of Message Number bit. Only values 1h-40h are valid. Values 41h-FFh are invalid 1465
- Section 27.17.8: Added subsection. Subsequent subsections, figures, and tables renumbered 1465
- Figure 27-29: Changed Read/Write value of DEFLG_DIAG and SEFLG_DIAG bits to R/W1C-0 1466
- Figure 27-30: Changed Read/Write value of DEFLG and SEFLG bits to R/W1C-0 1467
- Table 27-18: Corrected Value column range of Message Number bit to 1h-FFh 1468
- Table 27-18: Updated Description of Message Number bit. Only values 1h-40h are valid. Values 41h-FFh are invalid 1468
- Figure 27-32: Corrected register bit name to ABO_TIME 1469
- Table 27-19: Corrected register bit name to ABO_TIME 1469
- Table 27-25: Corrected Value column range of Message Number bit to 1h-40h 1479
- Table 27-28: Updated Description of EoB bit 1485
- Section 27.17.28: Changed sixth paragraph 1487
- Table 27-35: Changed the Func Bit description for Value = 1. (as an input to receive CAN data) 1495
- **Chapter 28: Multi-Buffered Serial Peripheral Interface Module (MibSPI) with Parallel Pin Option (MibSPIP)** 1497
- Chapter 28: Global: Updated all VBUSPCLK signals to VCLK 1497
- Chapter 28: Global: Changed $\overline{\text{SPISCS}}$ to $\overline{\text{SPICS}}$ 1497
- Table 28-1: Changed SPIENA enabled description in Slave Mode 1499
- Section 28.2.1: Changed first sentence in second paragraph 1500
- Figure 28-10: Corrected figure title 1511
- Figure 28-10: Corrected bits D11-D8 to 1110 1511
- Figure 28-11: Corrected figure title 1511
- Section 28.2.6.3: Updated first two paragraphs 1515
- Section 28.2.9.1: Changed sixth sentence. Added T2EDELAY 1529
- Section 28.2.9.2: Changed second sentence. Added C2EDELAY 1530
- Section 28.2.11.1: Changed second sentence 1533
- Section 28.2.11.2: Changed second sentence 1534
- Table 28-8: Added SPIPC9 at address offset 68h 1535
- Table 28-10: Changed Description of CLKMOD bit for Value = 1. ($\overline{\text{SPIENA}}$ is an input.) 1537
- Table 28-15: Corrected Description of SIMODIR0 bit 1545
- Table 28-21: Updated Description of all bits to clarify that bit is a pull control disable 1554
- Table 28-23: Changed Description of TXDATA bit. Added last Note 1556
- Section 28.3.16: Added NOTE 1557
- Table 28-24: Updated Description of CSNR and TXDATA bits 1557
- Table 28-25: Added table. Subsequent tables renumbered 1559
- Table 28-26: Corrected Description of RXEMPTY bit. (SPIBUF to RXDATA) 1560

- Table 28-28: Corrected Value column for all bits to 0-FFh 1562
- Table 28-28: Updated Description of C2TDELAY bit. Deleted first NOTE 1562
- Table 28-28: Updated Description of T2CDELAY bit. Deleted first NOTE 1562
- Figure 28-55: Updated reset value of CSDEF bit to FFh 1565
- Section 28.3.24: Added subsection. Subsequent subsections, figures, and tables renumbered 1571
- Table 28-36: Updated bit Descriptions to clarify register bit number corresponding to TG number 1576
- Table 28-37: Updated bit Descriptions to clarify register bit number corresponding to TG number 1577
- Table 28-37: Changed Description of CLRINTENRDY and CLRINTENSUS bits for Value = 1 (Write). The interrupt does not get generated 1577
- Table 28-38: Updated bit Descriptions to clarify register bit number corresponding to TG number 1578
- Table 28-39: Updated bit Descriptions to clarify register bit number corresponding to TG number 1579
- Table 28-39: Changed Description of CLRINTLVLRDY and CLRINTLVLSUS bits for Value = 1 (Write). Clear the TGx interrupt INT0 1579
- Table 28-40: Updated bit Descriptions to clarify register bit number corresponding to TG number 1580
- Table 28-41: Changed Description of TICKENA bit for Value = 1. Deleted second sentence 1581
- Figure 28-69: Changed LPEND bits to 15-8 1582
- Table 28-42: Changed Reserved bits to 23-16 and LPEND bits to 15-8 1582
- Table 28-42: Changed Value column of LPEND bit to 0-FFh 1582
- Table 28-42: Updated Description of LPEND bit 1582
- Figure 28-70: Changed PSTART bits to 15-8 and PCURRENT bits to 7-0 1583
- Figure 28-70: Changed bit 15 to PSTART 1583
- Figure 28-70: Changed bit 7 to PCURRENT 1583
- Table 28-43: Changed PSTART bits to 15-8 and PCURRENT bits to 7-0 1583
- Table 28-43: Changed Value column of PSTART bits and PCURRENT bits to 0-FFh 1583
- Table 28-43: Updated Description of PSTART bits and PCURRENT bits. Added Note 1583
- Table 28-44: Changed Description of ONESHOT bit. Added Note 1586
- Table 28-44: Changed Description of ICOUNTx bit. Added last sentence to second paragraph 1586
- Table 28-56: Corrected Description of EPRESCALE_FMT3 and EPRESCALE_FMT2 bits 1600
- Section 28.4.3: Added NOTE 1607
- Table 28-62: Updated Description of CSHOLD and TXDATA bits 1607
- Table 28-62: Updated Description of CSNR bit 1607
- Table 28-63: Added table. Subsequent tables renumbered 1609
- Table 28-64: Corrected bits descriptions (SPIBUF to RXRAM) 1610
- Section 28.5: Changed second paragraph 1612
- **Chapter 29: Serial Communication Interface (SCI)/Local Interconnect Network (LIN) Module 1621**
- Section 29.2.3.1: Changed third paragraph 1638
- Section 29.2.3.2: Changed second sentence in second paragraph 1638
- Section 29.2.4: Updated both paragraphs 1639
- Section 29.2.4: Updated procedure in second paragraph 1639
- Section 29.2.4.1.1: Updated first and last paragraphs 1639
- Section 29.2.4.1.2: Updated paragraph 1640
- Section 29.2.4.2.1: Updated first and last paragraphs 1640
- Section 29.2.4.2.1: Changed number 2 in second paragraph to Transmit Interrupt 1640
- Section 29.2.4.2.2: Updated paragraph 1640
- Section 29.3.4: Updated paragraph 1660
- Section 29.3.4: Updated procedure 1660
- Section 29.3.4.1.1: Updated first and last paragraphs 1660
- Section 29.3.4.1.2: Updated paragraph 1661
- Section 29.3.4.2.1: Updated first and last paragraphs 1661
- Section 29.3.4.2.1: Changed number 2 in second paragraph to Transmit Interrupt 1661
- Section 29.3.4.2.2: Updated paragraph 1662
- Section 29.6.2: Changed first bullet 1665
- Table 29-9: Changed Pull Control = Enabled when device is under reset 1666
- Table 29-16: Updated Description of SET RX DMA bit 1675

- Figure 29-32: Corrected SCICLEARINT register bit name for bit 30. (CLR PBE INT) 1678
- Table 29-28: Corrected bit name in Note to SET TX INT 1699
- **Chapter 30: Serial Communication Interface (SCI) Module** 1717
- Section 30.1.2: Changed Baud Clock Generator bullet to VCLK 1718
- Equation 57: Updated equation. Changed VBUSPCLK to VCLK 1722
- Equation 58: Updated equation. Changed VBUSPCLK to VCLK 1722
- Section 30.5: Updated both paragraphs 1729
- Section 30.5: Updated procedure in second paragraph 1729
- Section 30.5.1: Updated last paragraph 1730
- Section 30.5.2: Updated last paragraph 1730
- Section 30.5.2: Changed number 2 in second paragraph to Transmit Interrupt 1730
- Equation 59: Updated equation. Changed VBUSPLCK to VCLK 1751
- Equation 60: Updated equation. Changed VBUSPLCK to VCLK 1751
- Section 30.8.2: Changed first bullet 1763
- Table 30-33: Changed Pull Control = Enabled when device is under reset 1764
- **Chapter 31: Inter-Integrated Circuit (I2C) Module** 1765
- Table 31-7: Changed fourth paragraph in Description of XSMT bit 1784
- Section 31.6.21: Changed paragraph 1798
- Section 31.6.22: Changed paragraph 1799
- Section 31.6.23: Changed paragraph 1799
- Table 31-32: Updated Description of SDAPDR and SCLPDR bits. 0 = enabled; 1 = disabled 1799
- Section 31.6.24: Changed paragraph 1800
- Section 31.6.25: Changed paragraph 1800
- Table 31-35: Changed Pull Control = Enabled when device is under reset 1801
- **Chapter 32: EMAC/MDIO Module** 1803
- Figure 32-7: Added figure. Subsequent figures renumbered 1815
- Figure 32-8: Added figure. Subsequent figures renumbered 1816
- Figure 32-9: Added figure. Subsequent figures renumbered 1817
- Figure 32-28: Changed default value of REV bit to 0007 0105h 1867
- Table 32-25: Changed Value column of REV bit to 0007 0105h 1867
- Table 32-68: Changed Description of GMIIEN bit 1904
- **Chapter 33: Enhanced Capture (eCAP) Module** 1927
- **Chapter 34: Enhanced Quadrature Encoder Pulse (eQEP) Module** 1957
- **Chapter 35: Enhanced Pulse Width Modulator (ePWM) Module** 1995
- **Chapter 36: Data Modification Module (DMM)** 2108
- Figure 36-22: Updated Read/Write value of all bits to R/WP-0 2141
- Figure 36-22: Updated LEGEND to includ e WP 2141
- Table 36-22: Changed Description of all bits to Privilege mode (write) 2141
- Figure 36-23: Updated Read/Write value of all bits to R/WP-0 2142
- Figure 36-23: Updated LEGEND to include WP 2142
- Table 36-23: Changed Description of all bits to Privilege mode (write) 2142
- Figure 36-24: Updated Read/Write value of all bits to R/WP-0 2144
- Figure 36-24: Updated LEGEND to include WP 2144
- Figure 36-25: Updated Read/Write value of all bits to R/WP-0 2145
- Figure 36-25: Updated LEGEND to include WP 2145
- Table 36-25: Changed Description of all bits to Privilege mode (write) 2145
- Figure 36-26: Updated Read/Write value of all bits to R/WP-0 2146
- Figure 36-26: Updated LEGEND to include WP 2146
- Table 36-26: Changed Description of all bits to Privilege mode (write) 2146
- Figure 36-27: Updated Read/Write value of all bits to R/WP-0 2148
- Figure 36-27: Updated LEGEND to include WP 2148
- Table 36-27: Changed Description of all bits to Privilege mode (write) 2148
- Table 36-27: Changed Description of CLKCLR and SYNCCLR bits. Corrected bits to CLKOUT and SYNCOUT 2148

- Figure 36-28: Updated Read/Write value of all bits to R/WP-0 2149
- Figure 36-28: Updated LEGEND to include WP 2149
- Table 36-28: Changed Description of all bits to Privilege mode (write) 2149
- Figure 36-29: Updated Read/Write value of all bits to R/WP-x 2151
- Figure 36-29: Updated LEGEND to include WP 2151
- Table 36-29: Changed Description of all bits to Privilege mode (write) 2151
- Figure 36-30: Updated Read/Write value of all bits to R/WP-1 2152
- Figure 36-30: Updated LEGEND to include WP 2152
- Table 36-30: Changed Description of all bits to Privilege mode (write) 2152
- **Chapter 37: RAM Trace Port (RTP)** 2154
- **Chapter 38: eFuse Controller** 2187

IMPORTANT NOTICE FOR TI DESIGN INFORMATION AND RESOURCES

Texas Instruments Incorporated ('TI") technical, application or other design advice, services or information, including, but not limited to, reference designs and materials relating to evaluation modules, (collectively, "TI Resources") are intended to assist designers who are developing applications that incorporate TI products; by downloading, accessing or using any particular TI Resource in any way, you (individually or, if you are acting on behalf of a company, your company) agree to use it solely for this purpose and subject to the terms of this Notice.

TI's provision of TI Resources does not expand or otherwise alter TI's applicable published warranties or warranty disclaimers for TI products, and no additional obligations or liabilities arise from TI providing such TI Resources. TI reserves the right to make corrections, enhancements, improvements and other changes to its TI Resources.

You understand and agree that you remain responsible for using your independent analysis, evaluation and judgment in designing your applications and that you have full and exclusive responsibility to assure the safety of your applications and compliance of your applications (and of all TI products used in or for your applications) with all applicable regulations, laws and other applicable requirements. You represent that, with respect to your applications, you have all the necessary expertise to create and implement safeguards that (1) anticipate dangerous consequences of failures, (2) monitor failures and their consequences, and (3) lessen the likelihood of failures that might cause harm and take appropriate actions. You agree that prior to using or distributing any applications that include TI products, you will thoroughly test such applications and the functionality of such TI products as used in such applications. TI has not conducted any testing other than that specifically described in the published documentation for a particular TI Resource.

You are authorized to use, copy and modify any individual TI Resource only in connection with the development of applications that include the TI product(s) identified in such TI Resource. NO OTHER LICENSE, EXPRESS OR IMPLIED, BY ESTOPPEL OR OTHERWISE TO ANY OTHER TI INTELLECTUAL PROPERTY RIGHT, AND NO LICENSE TO ANY TECHNOLOGY OR INTELLECTUAL PROPERTY RIGHT OF TI OR ANY THIRD PARTY IS GRANTED HEREIN, including but not limited to any patent right, copyright, mask work right, or other intellectual property right relating to any combination, machine, or process in which TI products or services are used. Information regarding or referencing third-party products or services does not constitute a license to use such products or services, or a warranty or endorsement thereof. Use of TI Resources may require a license from a third party under the patents or other intellectual property of the third party, or a license from TI under the patents or other intellectual property of TI.

TI RESOURCES ARE PROVIDED "AS IS" AND WITH ALL FAULTS. TI DISCLAIMS ALL OTHER WARRANTIES OR REPRESENTATIONS, EXPRESS OR IMPLIED, REGARDING TI RESOURCES OR USE THEREOF, INCLUDING BUT NOT LIMITED TO ACCURACY OR COMPLETENESS, TITLE, ANY EPIDEMIC FAILURE WARRANTY AND ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, AND NON-INFRINGEMENT OF ANY THIRD PARTY INTELLECTUAL PROPERTY RIGHTS.

TI SHALL NOT BE LIABLE FOR AND SHALL NOT DEFEND OR INDEMNIFY YOU AGAINST ANY CLAIM, INCLUDING BUT NOT LIMITED TO ANY INFRINGEMENT CLAIM THAT RELATES TO OR IS BASED ON ANY COMBINATION OF PRODUCTS EVEN IF DESCRIBED IN TI RESOURCES OR OTHERWISE. IN NO EVENT SHALL TI BE LIABLE FOR ANY ACTUAL, DIRECT, SPECIAL, COLLATERAL, INDIRECT, PUNITIVE, INCIDENTAL, CONSEQUENTIAL OR EXEMPLARY DAMAGES IN CONNECTION WITH OR ARISING OUT OF TI RESOURCES OR USE THEREOF, AND REGARDLESS OF WHETHER TI HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

You agree to fully indemnify TI and its representatives against any damages, costs, losses, and/or liabilities arising out of your non-compliance with the terms and provisions of this Notice.

This Notice applies to TI Resources. Additional terms apply to the use and purchase of certain types of materials, TI products and services. These include; without limitation, TI's standard terms for semiconductor products http://www.ti.com/sc/docs/stdterms.htm), evaluation modules, and samples (http://www.ti.com/sc/docs/sampterms.htm).

Mailing Address: Texas Instruments, Post Office Box 655303, Dallas, Texas 75265