VSC8489-17 Datasheet
Dual Channel WAN/LAN/Backplane RXAUI/XAUI to SFP+/KR 10 GbE SerDes PHY with VeriTime™

Microsemi Headquarters
One Enterprise, Aliso Viejo,
CA 92656 USA
Within the USA: +1 (800) 713-4113
Outside the USA: +1 (949) 380-6100
Sales: +1 (949) 380-6136
Fax: +1 (949) 215-4996
Email: sales.support@microsemi.com
www.microsemi.com

©2018 Microsemi, a wholly owned subsidiary of Microchip Technology Inc. All rights reserved. Microsemi and the Microsemi logo are registered trademarks of Microsemi Corporation. All other trademarks and service marks are the property of their respective owners.

Microsemi makes no warranty, representation, or guarantee regarding the information contained herein or the suitability of its products and services for any particular purpose, nor does Microsemi assume any liability whatsoever arising out of the application or use of any product or circuit. The products sold hereunder and any other products sold by Microsemi have been subject to limited testing and should not be used in conjunction with mission-critical equipment or applications. Any performance specifications are believed to be reliable but are not verified, and Buyer must conduct and complete all performance and other testing of the products, alone and together with, or installed in, any end-products. Buyer shall not rely on any data and performance specifications or parameters provided by Microsemi. It is the Buyer's responsibility to independently determine suitability of any products and to test and verify the same. The information provided by Microsemi hereunder is provided "as is, where is" and with all faults, and the entire risk associated with such information is entirely with the Buyer. Microsemi does not grant, explicitly or implicitly, to any party any patent rights, licenses, or any other IP rights, whether with regard to such information itself or anything described by such information. Information provided in this document is proprietary to Microsemi, and Microsemi reserves the right to make any changes to the information in this document or to any products and services at any time without notice.

About Microsemi

Microsemi, a wholly owned subsidiary of Microchip Technology Inc. (Nasdaq: MCHP), offers a comprehensive portfolio of semiconductor and system solutions for aerospace & defense, communications, data center and industrial markets. Products include high-performance and radiation-hardened analog mixed-signal integrated circuits, FPGAs, SoCs and ASICs; power management products; timing and synchronization devices and precise time solutions, setting the world's standard for time; voice processing devices; RF solutions; discrete components; enterprise storage and communication solutions, security technologies and scalable anti-tamper products; Ethernet solutions; Power-over-Ethernet ICs and midspans; as well as custom design capabilities and services. Learn more at www.microsemi.com.

VMDS-10504. 4.1 11/18

Contents

Figures

Tables

1 Revision History

The revision history describes the changes that were implemented in the document. The changes are listed by revision, starting with the most current publication.

1.1 Revision 4.1

Revision 4.1 was published in September 2018. In revision 4.1 of this document, the registers were attached. For more information, see Registers, page 118.

1.2 Revision 4.0

Revision 4.0 was published in November 2017. The following is a summary of the changes in revision 4.0 of this document.

- Low-voltage transistor-to-transistor logic (LVTTL) updated to low-voltage transistor-to-transistor logic with open-drain output (LVTTLOD) where appropriate.
- The two-wire serial slave interface register address illustrations and 24-bit addressing scheme details were updated. For more information, see Two-Wire Serial (Slave) Interface, page 111.
- Line-side 10G receiver input AC characteristics were updated. For more information, see Table 52, page 120.
- Conditions for transmitter SFP+ direct attach copper output AC characteristics were updated. For more information, see Table 59, page 125.
- Reference clock AC characteristics were updated. For more information, see Table 67, page 128.
- The SPI interface timing diagram was updated. For more information, see Figure 97, page 132.
- Some pin description information was updated. For more information, see Pins by Function, page 136.
- Moisture sensitivity level (MSL) was corrected from 2 to 4. For more information, see Moisture Sensitivity, page 159.

1.3 Revision 2.0

Revision 2.0 was published in September 2017. It was the first publication of this document.

2 Overview

The VSC8489-17 device is a dual-port 10G/1G WAN/LAN/Backplane RXAUI/XAUI to SFP+/KR 10 GbE SerDes PHY with VeriTime™. It supports IEEE 802.3ae and IEEE 1588v2.

The VSC8489-17 is an IEEE 1588v2-compliant dual-channel device for timing-critical applications. It is also well suited for optical module, copper Twinax cable, and backplane applications with support for a wide variety of protocols, including 10 GbE LAN, 10 Gb WAN, and 1 Gb Legacy Ethernet.

VeriTime™ is Microsemi's patent-pending timing technology that delivers the industry's most accurate IEEE 1588v2 timing implementation for Ethernet transceivers. The IEEE 1588v2 timing integrated in the VSC8489-17 device is the most reliable and lowest-cost method of implementing the timing accuracy to maintain existing timing-critical capabilities during the migration from TDM to packet-based architectures. Complete Y.1731 OAM performance monitoring capabilities, master, slave, boundary, and transparent clock configurations, and sophisticated classifications (including UDP, IPv4, IPv6 packets and VLAN, and MPLS-TP encapsulation) are also supported.

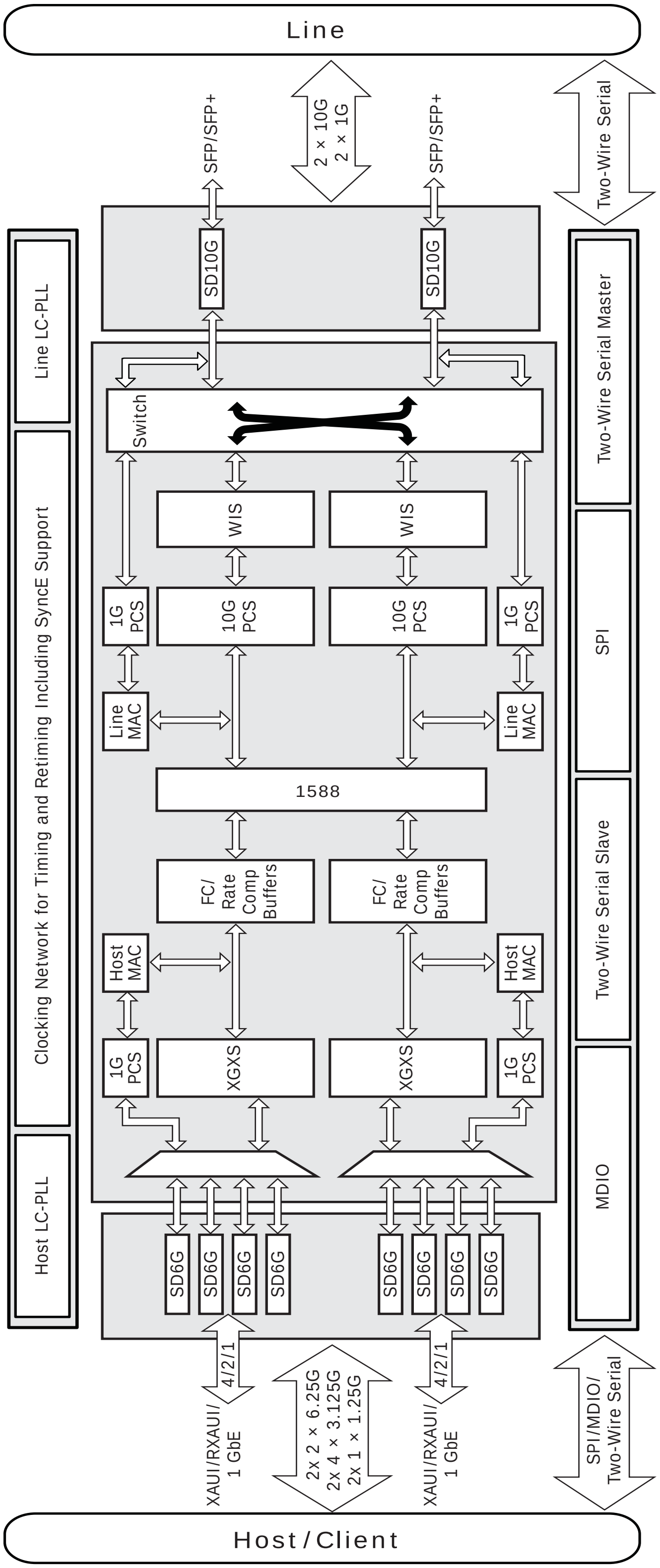
The VSC8489-17 device meets the SFP+ limiting and linear SR/LR/ER/ZR/220MMF host requirements in accordance with the SFF-8431 specifications. It also compensates for electrical and optical impairments in SFP+ applications, along with imperfections of the PCB and connectors.

The VSC8489-17 device provides a complete suite of BIST functionality, including line and client loopbacks, along with pattern generation and error detection. Highly flexible clocking options support LAN and WAN operation using single 156.25 MHz reference clock rate inputs for seamless Synchronous Ethernet support. The VSC8489-17 device also includes a failover switching capability for protection routing, along with selectable lane ordering.

The serial side supports 1.25 Gbps and various 10 Gbps modes. Each channel consists of a receiver (Rx) and a transmitter (Tx) subsection. Three programmable reference clock inputs (XREFCK, SREFCK, and WREFCK) support the various modes along with clock and data recovery (CDR) in the Rx and Tx subsections of all channels.

The following illustration shows a high-level block diagram for the VSC8489-17 device.

Figure 1 • VSC8489-17 Block Diagram



2.1 Major Applications

- Multiple-port RXAUI/XAUI to SFI/SFP+ line cards or network interface controllers

- Carrier Ethernet networks requiring 1588v2 timing
- Secure data center-to-data center interconnects
- 10 GbE switch cards, router cards, and NICs

The following illustrations show the various applications for the VSC8489-17 device.

Figure 2 • **SFP/SFP+ Application**

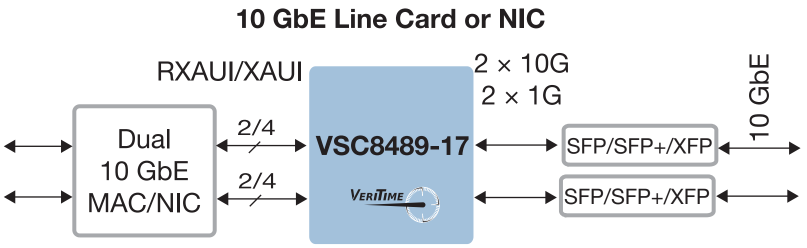


Figure 3 • **Backplane Equalization Application**

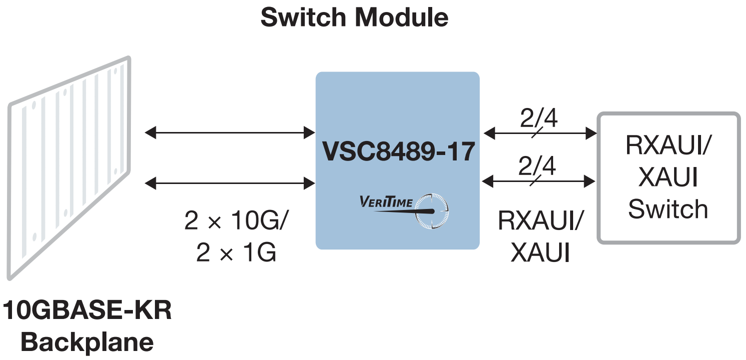


Figure 4 • **1588 Transparent Clock Line Card End-to-End PHY Application**

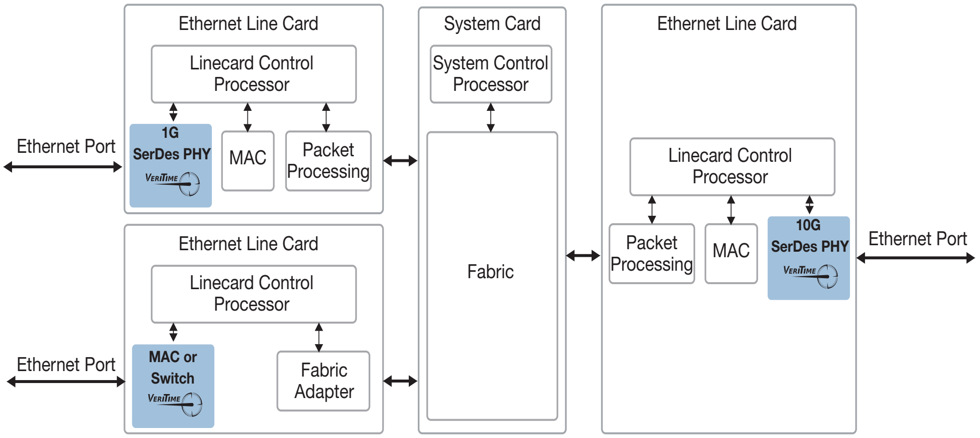
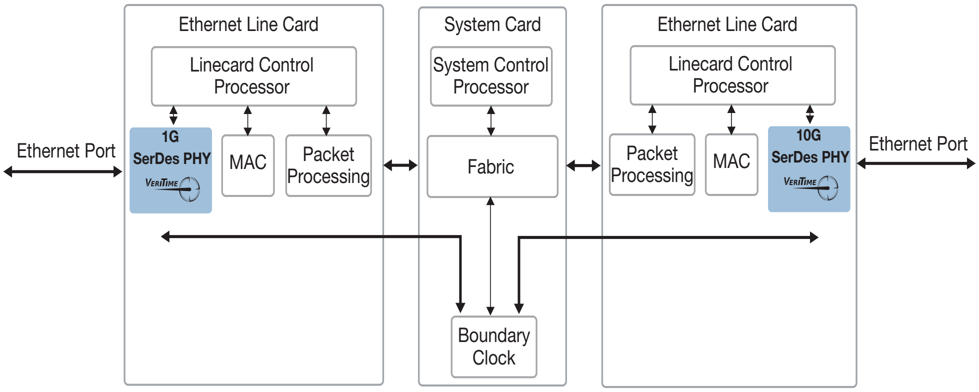


Figure 5 • **1588 Boundary Clock Line Card Application**



2.2 Features and Benefits

The main features of the VSC8489-17 device are as follows:

- IEEE 1588v2/1731 OAM precision timing support at 1G and 10G
- Compliant to IEEE 802.3ae and SFF-8431 electrical (SFI) specifications
- 9.95 Gbps WAN, 10.3125 Gbps LAN, and 1.25 Gbps Ethernet support
- Supports all standard SFP+ applications
- Adaptive receive equalization with programmable, multitap transmit pre-emphasis
- Extended WIS support
- MDIO, SPI, and two-wire serial slave management interfaces
- Failover switching for protection routing, along with selectable lane ordering (non-hitless switching)
- VScope™ input signal monitoring integrated circuit
- Host-side and line-side loopbacks with BIST functions
- I/O programmability for lane swap, invert, amplitude, slew, pre-emphasis, and equalization
- Optional forward error correction (FEC)
- Flexible clocking options for Synchronous Ethernet support
- Passive copper cable compliant to SFF-8431 is supported for minimum transmission cost
- Pin-friendly with VSC8488-15

3 Functional Descriptions

This section describes the functional aspects of the VSC8489-17 device, including the functional block diagram, operating modes, and major functional blocks.

The VSC8489-17 device host-side interface is either four-lane XAUI, two-lane RXAUI, or one-lane 1 GbE. The line-side interface is 10G SFP+ or 1 GbE SFP.

Each lane has the following main sections:

- PMA
  The PMA section contains the high-speed serial I/O interfaces, an input equalization circuit, a KR-compliant output buffer, and a SerDes. Additionally, the PMA also generates all the line-side clocks, including the clocks required for Synchronous Ethernet applications.
- WIS
  The WIS section contains the framing and de-framing circuits and control and status registers to convert the data to be IEEE 802.3ae WIS-compliant.
- 10G PCS
  The 10G PCS section is composed of the PCS transmit, PCS receive, block synchronization, and BER monitor processes. The PCS functions can be further broken down into encode or decode, scramble or descramble, and gearbox functions, as well as various test and loopback modes.
- 1G PCS
  The 1G PCS section describes the 1000BASE-X/SGMII coding and auto-negotiation processes. There are two instances per channel, one for the host and one for the line.
- IEEE 1588
  The IEEE 1588 section contains the local time counter, analyzer, time stamp, FIFO, and rewriter to support both 1-step and 2-step clock timing, and to perform 1588 frame detection, time stamp appending, header removal, and frame processing.
- MAC
  The MAC block frames data for transmission over the network before passing the frame to the physical layer interface, where it is transmitted as a stream of bits. In 1G mode, MAC can be enabled.
- FIFO
  The FIFO section contains a rate-compensating FIFO between the line rate and the host rate.
- Cross Connect
  The cross connect connects one port to the adjacent port to enable routing data/clock to and from port 1 and 0. This cross connect only supports broadcasting from PMA to XAUI but NOT from XAUI to PMA. The failover supported by this cross connect is not hitless.
- XGXS
  The XGXS implements the PHY XGXS referenced in IEEE 802.3 Clause 47, and contains a 10GBASE-X PCS as defined in Clause 48. It provides the necessary translation between the external XAUI interface and the on-chip XGMII interface. In addition to standard 4-lane XAUI, it also supports 2-lane RXAUI/DDR-XAUI.
- XAUI/RXAUI
  The XAUI and RXAUI section contains the parallel XAUI/RXAUI I/O interface and a SerDes.
- KR
  The KR driver includes programmable equalization accomplished by a three-tap finite impulse response (FIR) structure. Three-tap delays are achieved by three flip-flops clocked by a high-speed serial clock (10 GHz in 10G mode; 1 GHz in 1G mode).
- Loopback
  The loopback sections describe the different loopbacks available in the VSC8489-17 device, including system and network loopbacks. The various loopbacks enhance the engineering debugging and manufacturing testing capability.
- Management
  The management section contains the status and configuration registers and the serial management interface logic to access them.

3.1 Data Path Overview

The following sections provide data path information for the VSC8489-17 device. Ingress and egress data flow is relative to the line-side interface.

3.1.1 Ingress Operation

Data is received by the line-side interface (SFP+/1 GbE), processed by core logic, and transmitted from the host-side interface (XAUI/RXAUI/1 GbE) in the ingress (or line-side receive) data path.

High-speed serial data is received by the PMA. Data can be equalized and is delivered to the clock recovery unit (CRU). The received serial data must be a 66B/64B encoded ethernet frame at 10.3125 Gbps in 10G LAN mode, a SONET/SDH STS-192c frame at 9.953 Gbps in 10G WAN mode, or 8B/10B encoded data at 1.25 Gbps in 1 GbE mode.

In 10G WAN mode, the CRU data is processed by the WIS where 66B/64B encoded ethernet data is extracted from SONET/SDH STS-192c frames and overhead bytes are processed. The extracted payload data is then processed by the 10G PCS. In 10G LAN mode, the CRU data is processed by a 10G PCS. In 1G mode, the CRU data is processed by the line-side 1G PCS. The 1G PCS data can be optionally processed by the IEEE 1588 and two MAC logic blocks.

In 10G LAN and WAN modes, data from the core is 8B/10B encoded by the XGXS logic and serialized in the host-side SerDes. The host interface can be configured as a XAUI interface where four lanes of 3.125 Gbps data is transmitted, or as a RXAUI interface where two lanes of 6.25 Gbps data is transmitted. Data is transmitted on XAUI lanes 0 and 2 when the host interface is configured to be RXAUI.

In 1 GbE mode, data from the core is 8B/10B encoded by the host-side 1G PCS logic and serialized in the host-side SerDes. 1.25 Gbps data is transmitted from the host interface on either XAUI lane 0 or 3. When 1 GbE data is transmitted from XAUI lane 0, data received by the host interface must enter on lane 0. When 1 GbE data is transmitted from XAUI lane 3, data received by the host interface must enter on lane 3.

3.1.2 Egress Operation

Data is received by the host-side interface (XAUI/RXAUI/1 GbE), processed by core logic, and transmitted from the line-side interface (SFP+/1 GbE) in the egress (or line-side transmit) data path.

The host-side interface can be configured to receive XAUI or RXAUI data when in 10G LAN or 10G WAN modes. Data enters the part on XAUI lanes 0 and 2 when using the RXAUI interface. The host-side interface receives 1 GbE data when the VSC8489-17 device is in the 1G operating mode. XAUI lane 0 or lane 3 may be selected to receive the 1.25 Gbps data at the host interface. When receiving data on XAUI lane 0, 1 GbE data will be transmitted from XAUI lane 0 in the ingress data path. When receiving data on XAUI lane 3, 1 GbE data will be transmitted from XAUI lane 3 in the ingress data path.

In 10G mode, a clock is recovered from each lane of XAUI/RXAUI data in the host-side SerDes. The data is 8B/10B decoded and lane aligned in the XGXS logic, then optionally processed by the IEEE 1588 and two MAC logic blocks. The data is then 66B/64B encoded by the 10G PCS logic. The data is serialized by the PMA in 10G LAN mode and transmitted from the line interface at 10.3125 Gbps. When the WIS logic is enabled in 10G WAN mode, a SONET/SDH STS-192c frame is created using the 66B/64B encoded data as the frame's payload. The WIS data is serialized by the PMA and transmitted from the line interface at 9.953 Gbps.

In 1G mode, a clock is recovered from 1 GbE data in the host-side SerDes. The data is 8B/10B decoded by the host-side 1G PCS, then optionally processed by the IEEE 1588 The data is 8B/10B encoded by the line-side 1G PCS logic, serialized by the PMA, and transmitted from the line interface at 1.25 Gbps.

3.1.3 Interface Data Rates

The following table shows the interface data rates supported by the VSC8489-17 device.

Table 1 • **Interface Data Rates**

Operating Mode	Line-Side Datarate (Gbps)	Host-Side Interface	Host-Side Datarate (Gbps)
10G LAN	1 × 10.3125	XAUI	4 × 3.125
10G LAN	1 × 10.3125	RXAUI	2 × 6.25
10G WAN	1 × 9.95328	XAUI	4 × 3.125
10G WAN	1 × 9.95328	RXAUI	2 × 6.25
1 GbE	1 × 1.25	1 GbE	1 × 1.25

3.2 Physical Medium Attachment (PMA)

The VSC8489-17 PMA section consists of a receiver (Rx) and a transmitter (Tx) subsection. The receiver accepts data from the serial data input RXIN and sends the parallel data to the WIS, 10G PCS, or 1G PCS block. A data rate clock also accompanies the parallel data. The transmitter accepts parallel data from the WIS or PCS block and transmits at serial data output TXOUT. A loopback at the data path is also provided, connecting the Rx and the Tx subsection.

Serial data is pre-equalized in the input buffer, and clock and data are recovered in the deserializer, which provides 32-bit data. A demux then deserializes the data into a parallel core data interface. A PLL in the Rx subsection is used as reference for clock and data recovery. Locked to the incoming datastream, a lane sync signal is derived from the PLL clock, which may be used for source synchronous data transmission to one or multiple transmitters.

The Tx subsection is made up of the serializer, the output buffer, and the PLL. The high-speed serial stream is forwarded to a 3-tap filter output buffer. The PLL in the Tx subsection is used to generate the high-speed clock used in the serializer.

To support different data rates, a frequency synthesizer inside the Rx and Tx subsection takes the reference clock input XREFCK and generates all necessary clock rates.

The PMA also has two fully programmable clock outputs, TXCKOUT and RXCKOUT, that may be used to output various clock domains from the PMA. For more information about the reference clock, see Reference Clock, page 103.

3.2.1 VScope Input Signal Monitoring Integrated Circuit

The VScope™ input signal monitoring integrated circuit displays the input signal before it is digitized by the CDR. The two primary configurations are as follows:

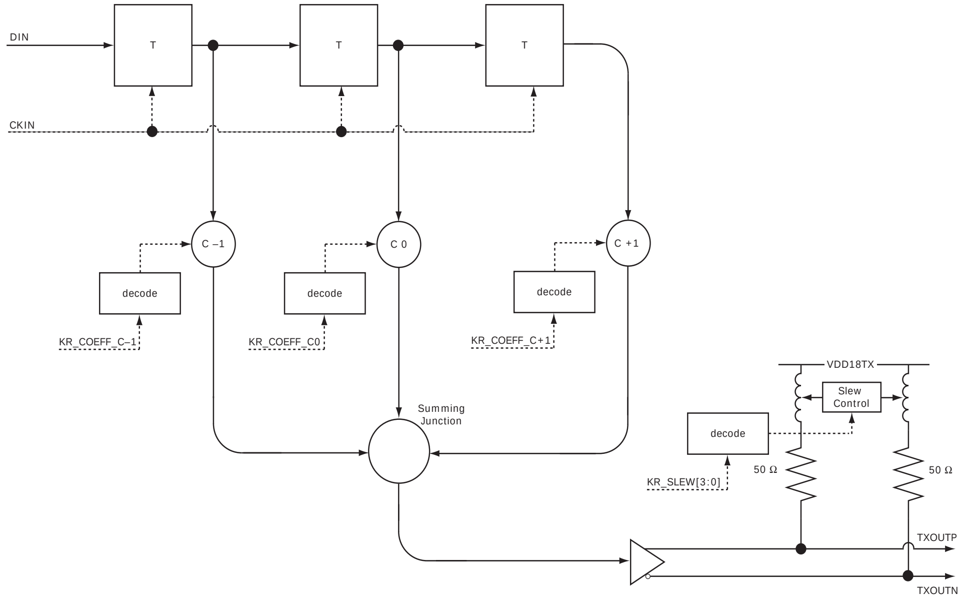
- Unity Gain Amplifier monitors the 10 Gbps input signals before signal processing and equalization. VScope input signal monitoring integrated circuit acts as a virtual scope to effectively observe the received data signal before it has been processed. The autonomous adaptive filter taps must first be disabled and the front-end receiver must be set for operation as a linear, unity gain amplifier. In this mode, all DFE taps are set to zero. This mode does not require an adaptive algorithm.
- Link Monitor provides the link margin.
  VScope input signal monitoring integrated circuit enables design engineers and system developers to monitor signals remotely without disrupting the data integrity of a live data path. By monitoring the health of a given link (optical or electrical), various types of signal degradation can be identified and corrected.

Note: The VScope input signal monitoring integrated circuit feature is only available in the 10G operation mode.

3.2.1.1 10GBASE-KR Output Driver

The high-speed output driver includes programmable equalization accomplished by a three-tap finite impulse response (FIR) structure. The three-tap delays are achieved by three flip-flops clocked by a high-speed serial clock, as shown in the following illustration. Coefficients C(–1), C(0) and C(+1) adjust the pre-cursor, main-cursor, and post-cursor of the output waveform. The coefficients are independently adjusted by control bits. The bits for each coefficient are decoded in a thermometer fashion to achieve linear coefficient adjustment. The three delayed data streams, after being properly strength adjusted by their coefficients, are summed by a summing amplifier. The output driver meets the requirements defined in IEEE 802.3ap Clause 72.

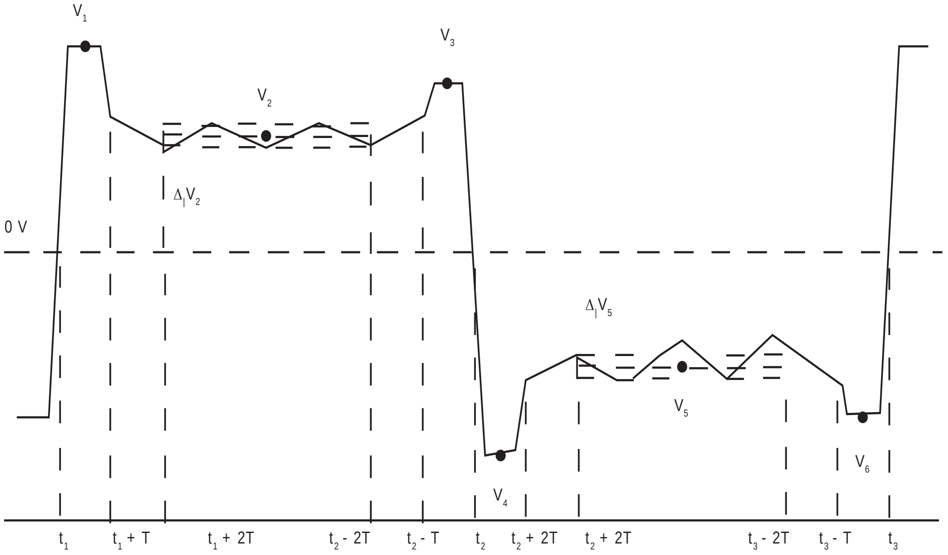
Figure 6 • **10GBASE-KR Output Driver**



The final output stage has 50 Ω back-termination with inductor peaking. The output slew rate is controlled by adjusting the effectiveness of the inductors.

The test pattern for the transmitter output waveform is the square wave test pattern with at least eight consecutive 1s. The following illustration shows the transmitter output waveform test, based on voltages V_1 through V_6, ΔV_2, and ΔV_5.

Figure 7 • **KR Test Pattern**



The output waveform is manipulated through the state of the coefficient C(-1), C(0), and C(+1).

3.3 WAN Interface Sublayer (WIS)

The WAN interface sublayer (WIS) is defined in IEEE 802.3ae Clause 50. The VSC8489-17 WIS block is fully compliant with this specification. The VSC8489-17 offers additional controls, ports, and registers to allow integration into a wider array of SONET/SDH equipment.

In addition to the SONET/SDH features addressed by WIS as defined by IEEE, most SONET/SDH framers/mappers contain additional circuitry for implementing operation, administration, maintenance, and provisioning (OAM&P). These framers/mappers also support special features to enable compatibility with legacy SONET/SDH solutions. Because the VSC8489-17 WIS leverages Microsemi’s industry leading framer/mapper technology, it contains suitable features for standard SONET/SDH equipment. This includes the transmit/receive overhead serial interfaces (TOSI/ROSI) commonly used for network customization and OAM&P support for SONET/SDH errors not contained in the WIS standard, support for common legacy SONET/SDH implementations, and SONET/SDH jitter and timing quality.

3.3.1 Operation

WAN mode is enabled by asserting 2x0007.0 (SPI/MDIO/TWS) or wis_ctrl2.wan_mode. Status register bit 1xA101.3 (SPI/MDIO/TWS) or Vendor_Specific_PMA_Status_2.WAN_ENABLED_status indicates whether WAN mode is enabled or not. The Rx and Tx paths both have WAN mode enabled or disabled. It is not possible to have WAN mode in the Tx path enabled while the Rx path is disabled, or vice versa.

The transmit portion of the WIS does the following:

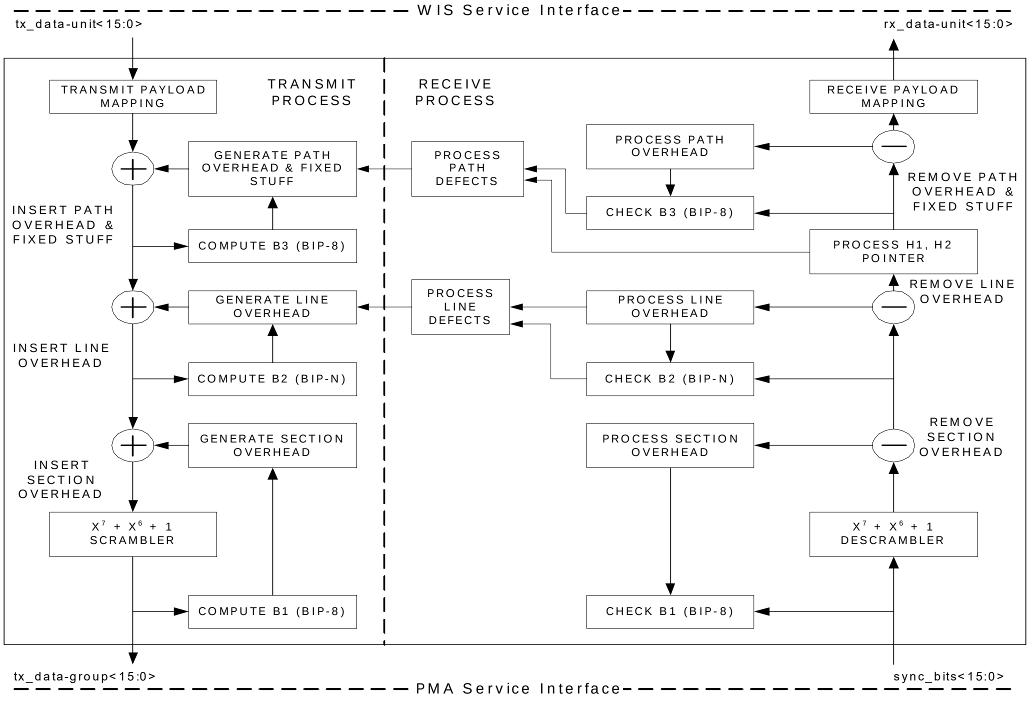
- Maps data from the PCS through the WIS service interface and to the SONET/SDH synchronous payload envelope (SPE)
- Generates path, line, and section overhead octets
- Scrambles the frame
- Transmits the frame to the PMA service interface

The receive portion of the WIS does the following:

- Receives data from the PMA service interface
- Delineates octet and frame boundaries
- Descrambles the frame
- Processes section, line, and path overhead information that contain alarms and parity errors
- Interprets the pointer field
- Extracts the payload for transmittal to the PCS through the WIS service interface

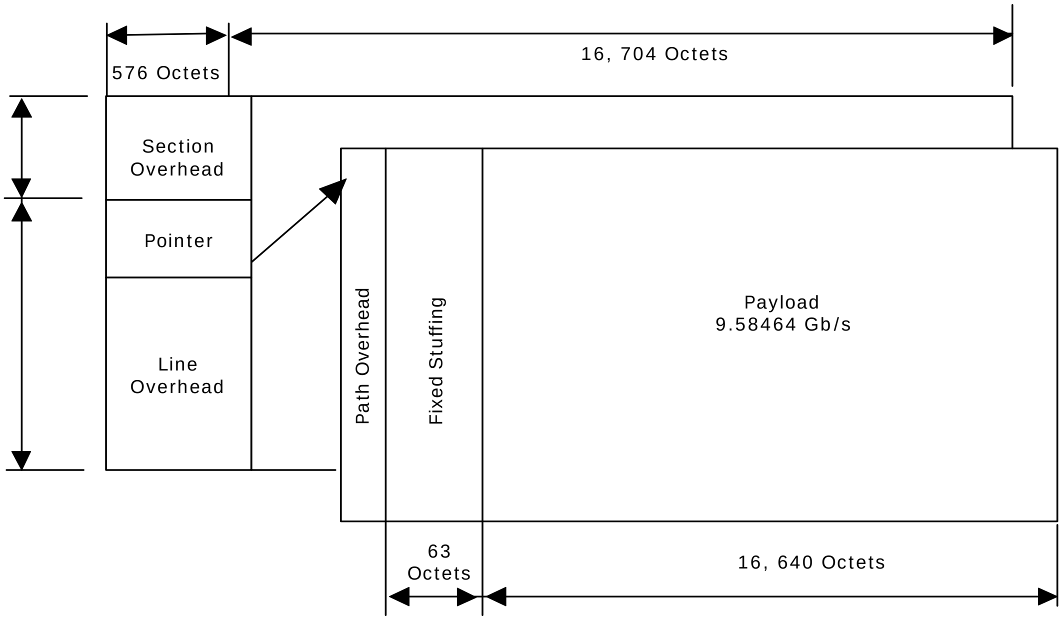
The following illustration shows the WIS block diagram.

Figure 8 • **WIS Transmit and Receive Functions**



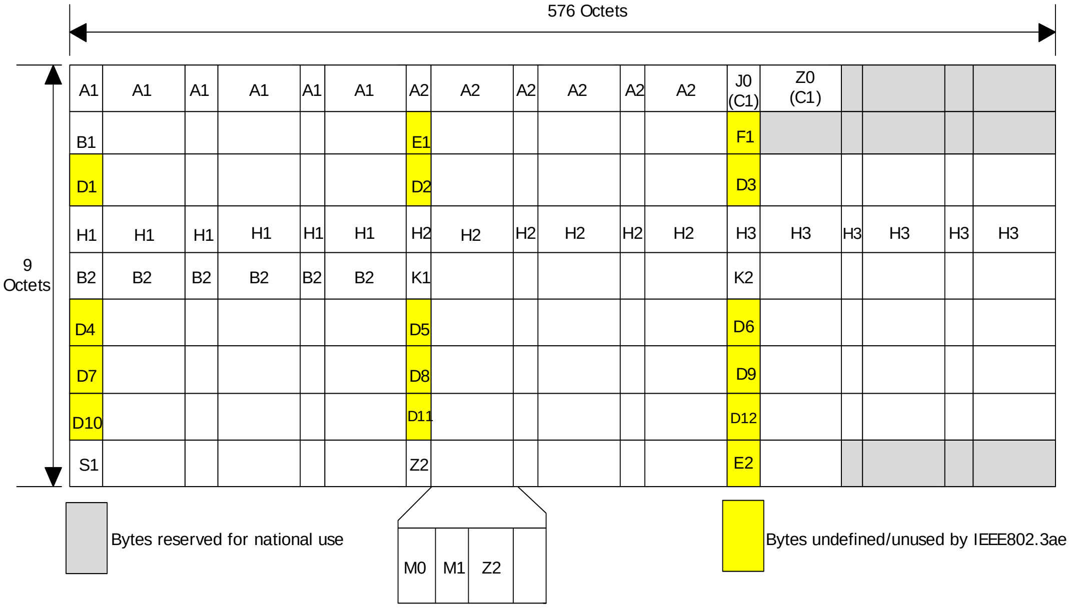
The following illustration shows the WIS frame structure.

Figure 9 • **WIS Frame Structure**



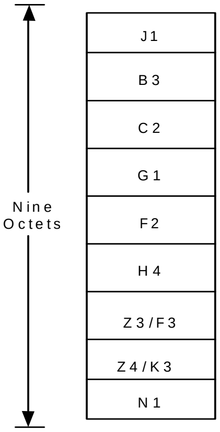
The following illustration shows the positions of the section and line overhead octets within the WIS frame.

Figure 10 • **STS-192c/STM-64 Section and Line Overhead Structure**



The following illustration shows the path overhead octet positions.

Figure 11 • **Path Overhead Octets**



3.3.2 Section Overhead

The section overhead portion of the SONET/SDH frame supports frame synchronization, a tandem connection monitor (TCM) known as the Section Trace, a high-level parity check, and some OAM&P octets. The following table lists each of the octets, including their function, specification, and related information.

The VSC8489-17 device provides a mechanism to transmit a static value as programmed by the MDIO interface. However, by definition, MDIO is not fast enough to alter the octet on a frame-by-frame basis.

Table 2 • **Section Overhead**

Overhead Octet	Function	IEEE 802.3ae WIS Usage	Recommended Value	WIS Extension
A1	Frame alignment	Supported	0xF6	Register (EWIS_TX_A1_A2) TOSI and ROSI access.

Table 2 • **Section Overhead *(continued)***

Overhead Octet	Function	IEEE 802.3ae WIS Usage	Recommended Value	WIS Extension
A2	Frame alignment	Supported	0x28	Register (EWIS_TX_A1_A2) TOSI and ROSI access.
J0	Section trace	Specified value	For more information, see Section Trace (J0), page 17	A 1-byte, 16-byte, or 64-byte trace message can be sent using registers WIS_Tx_J0_Octets_1_0 to WIS_Tx_J0_Octets_15_14, EWIS_TX_MSGLEN, or EWIS_Tx_J0_Octets_17_16 to EWIS_Tx_J0_Octets_63_62 and received using registers WIS_Rx_J0_Octets_1_0 to WIS_Rx_J0_Octets_15_14, EWIS_RX_MSGLEN, and EWIS_Rx_J0_Octets_17_16 to EWIS_Rx_J0_Octets_63_62. TOSI and ROSI access.
Z0	Reserved for section growth	Unsupported	0xCC	Register EWIS_TX_Z0_E1 TOSI and ROSI access.
B1	Section error monitoring (Section BIP-8)	Supported	Bit interleaved parity - 8 bits, as specified in T1.416	Using the TOSI, the B1 byte can be masked for test purposes. For each B1 mask bit that is cleared to 0 on the TOSI interface, the transmitted bit is left unchanged. For each B1 mask bit that is set to 1 on the TOSI interface, the transmitted bit is inverted. Using the ROSI, the B1 error locations can be extracted. Periodically latched counter (EWIS_B1_ERR_CNT1-EWIS_B1_ERR_CNT0) is available.
E1	Orderwire	Unsupported	0x00	Register EWIS_TX_Z0_E1 TOSI and ROSI access.
F1	Section user channel	Unsupported	0x00	Register EWIS_TX_F1_D1 TOSI and ROSI access.
D1-D3	Section data communications channel (DCC)	Unsupported	0x00	Register EWIS_TX_F1_D1 to EWIS_TX_D2_D3 TOSI and ROSI access.

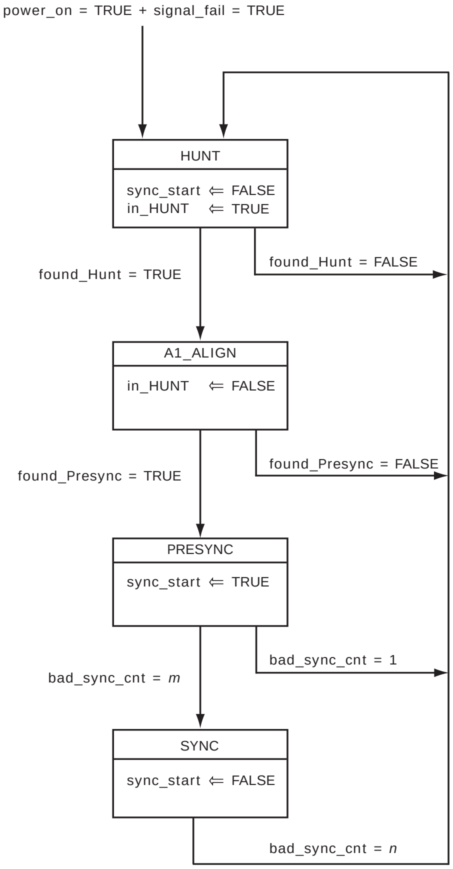
3.3.2.1 Frame Alignment (A1, A2)

The SONET/SDH protocol is based upon a frame structure that is delineated by the framing octets, A1 and A2. The framing octets are defined to be 0xF6 and 0x28 respectively. In the transmit direction, all 192 A1 octets are sourced from the TX_A1 (EWIS_TX_A1_A2.TX_A1) register while the A2 octets are sourced from the TX_A2 (EWIS_TX_A1_A2.TX_A2) register.

In the receive direction, the frame aligner monitors the input bus from the PMA and performs word alignment. The frame alignment architecture is composed of a primary and secondary state machine. The selected frame alignment and synchronization pattern have implications on the tolerated input BER. The higher the input BER, the less likely the frame boundary can be found. The chances of finding the frame boundary are improved by reducing the number of A1/A2 bytes required to be detected (using a smaller pattern width). According to the WIS specification, the minimum for all parameters allows a signal with an error tolerance of 10^{-12} to be framed.

The following illustration shows the primary synchronization state diagram.

Figure 12 • **Primary Synchronization State Diagram**



The following table lists the variables for the primary state diagram. The variables are reflected in registers EWIS_RX_FRM_CTRL1 and EWIS_RX_FRM_CTRL2 that can be alternately reconfigured.

Table 3 • **Framing Parameter Description and Values**

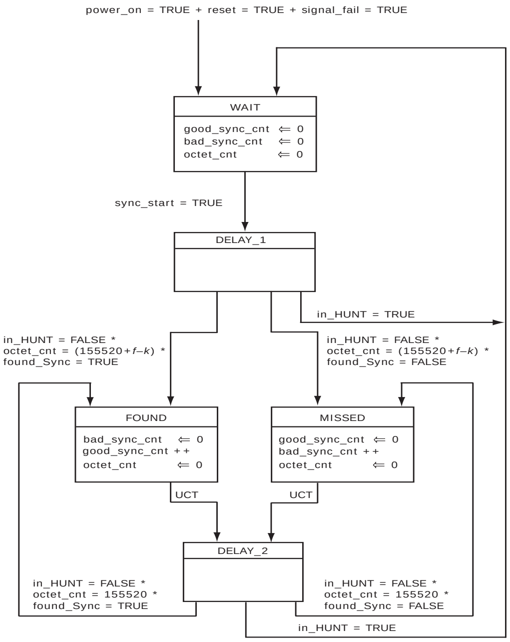
Name	Description	IEEE 802.3ae Parameter	IEEE 802.3ae Range	Range	Default
Sync_Pattern width	Sequence of f consecutive A1s followed immediately by a sequence of f consecutive A2s. If f = 2, Sync_Pattern is A1A1A2A2.	f	2 to 192	0 to 16 Exceptions: If f = 0, Sync_Pattern is A1 + 4 MSBs of A2. If f = 1, Sync_Pattern is A1A1A2.	2
Hunt_Pattern width	Sequence of i consecutive A1s.	i	1 to 192	1 to 16.	4
Presync_Pattern A1 width	Presync_Pattern consists of a sequence of j consecutive A1s followed immediately by a sequence of k consecutive A2s.	j	16 to 190	1 to 16 If set to 0, behaves as if set to 1. If set to 17 to 31, behaves as if set to 16.	16

Table 3 • Framing Parameter Description and Values *(continued)*

Name	Description	IEEE 802.3ae Parameter	IEEE 802.3ae Range	Range	Default
Presync_Pattern A2 width	Presync_Pattern consists of a sequence of j consecutive A1s followed immediately by a sequence of k consecutive A2s.	k	16 to 192	0 to 16 0 means only 4 MSB of A2 are used. If set to 17 to 31, behaves as if set to 16.	16
SYNC state entry	Number of consecutive frame boundaries needed to be found after entering the PRESYNC state in order to enter the SYNC state.	m	4 to 8	1 to 15 If set to 0, behaves as if set to 1.	4
SYNC state exit	Number of consecutive frame boundary location errors detected before exiting the SYNC state.	n	1 to 8	1 to 15 If set to 0, behaves as if set to 1.	4

The following illustration shows the secondary synchronization state diagram.

Figure 13 • Secondary Synchronization State Diagram



3.3.2.2 Loss of Signal (LOS)

WIS_STAT3.LOS alarm status is a latch-high register; back-to-back reads provide both the event as well as status information. The LOS event also asserts register EWIS_INTR_PEND1.LOS_PEND until read. This event can propagate an interrupt to either WIS_INTA or WIS_INTB based upon mask enable bits EWIS_INTR_MASKA_1.LOS_MASKA and EWIS_INTR_MASKB_1.LOS_MASK.

There is no hysteresis on the LOS detection, and so it is recommended to have the system software to implement a sliding window to check on the LOS before qualifying the presence of a signal. As an alternative, Rx_LOS can be used from the optical module (through LOPC) to qualify the input signal. In addition to using analog detection, digital detection such as PCS_Rx_Fault is recommended to determine if the input signal is good.

When the near-end device experiences LOS, it is possible to automatically transmit a remote defect indication (RDI-L) to the far-end for notification purposes. The EWIS_RXTX_CTRL.TXRDIL_ON_LOS, if asserted, overwrites the outgoing K2 bits with the RDI-L code. In the receive path, it is possible to trigger an AIS-L state (alarm assertion plus forcing the payload to an all ones state) upon a detection of an LOS condition. This is accomplished by asserting EWIS_RXTX_CTRL.RXAISL_ON_LOS.

3.3.2.3 Loss of Optical Carrier (LOPC)

The input pin LOPC can be used by external optic components to directly assert the loss of optical power to the physical media device. Any change in level on the LOPC input asserts register EWIS_INTR_PEND2.LOPC_PEND until read. The current status of the LOPC input pin can be read in register EWIS_INTR_STAT2.LOPC_STAT. The LOPC input can be active high or active low by setting the Vendor_Specific_LOPC_Control.LOPC_state_inversion_select bit appropriately. The LOPC_PEND bit can propagate an interrupt to either WIS_INTA or WIS_INTB based upon mask enable bits EWIS_INTR_MASKA_2.LOPC_MASKA and EWIS_INTR_MASKB_2.LOPC_MASKB.

When the near-end device experiences LOPC, it is possible to automatically transmit a remote defect indication (RDI-L) to the far-end to notify it of a problem. The EWIS_RXTX_CTRL.TXRDIL_ON_LOPC register bit, if asserted, overwrites the outgoing K2 bits with the RDI-L code. In the receive path, it is possible to force the receive framer into an LOF state, thereby squelching subsequent alarms and invalid payload data processing. This is accomplished by asserting EWIS_RX_ERR_FRC1.RXLOF_ON_LOPC. Similar to the LOF condition forced upon an LOPC, the EWIS_RXTX_CTRL.RXAISL_ON_LOPC can force the AIS-L alarm assertion, plus force the payload to an all ones state to indicate to the PCS the lack of valid data upon an LOPC condition.

3.3.2.4 Severely Errored Frame (SEF)

Upon reset, the VSC8489-17 device Rx WIS enters the out-of-frame (OOF) state with both the severely errored frame (SEF) and loss of frame (LOF) alarms active. The SEF state is terminated when the framer enters the SYNC state. The framer enters the SYNC state after EWIS_RX_FRM_CTRL2.SYNC_ENTRY_CNT plus 1 consecutive frame boundaries are identified. An SEF state is declared when the framer enters the out-of-frame (OOF) state. The frame changes from the SYNC state to the OOF state when EWIS_RX_FRM_CTRL2.SYNC_EXIT_CNT consecutive frames with errored frame alignment words are detected. The SEF alarm condition is reported in WIS_STAT3.SEF. This register latches high providing a combination of interrupt pending and status information within consecutive reads.

An additional bi-stable interrupt pending bit SEF_PEND (EWIS_INTR_PEND1.SEF_PEND) is provided to propagate an interrupt to either WIS_INTA or WIS_INTB based upon mask enable bits SEF_MASKA (EWIS_INTR_MASKA_1.SEF_MASKA) and SEF_MASKB (EWIS_INTR_MASKB_1.SEF_MASKB).

3.3.2.5 Loss of Frame (LOF)

An LOF occurs when an out-of-frame state persists for an integrating period of EWIS_LOF_CTRL1.LOF_T1 frames. To provide for the case of intermittent OOFs when not in the LOF state, the integrating timer is not reset to zero until an in-frame condition persists continuously for EWIS_LOF_CTRL1.LOF_T2 frames. The LOF state is exited when the in-frame state persists continuously for EWIS_LOF_CTRL2.LOF_T3 frames. The LOF state is indicated by the WIS_STAT3.LOF register being asserted. This register latches high, providing a combination of pending and status information over consecutive reads.

An additional bi-stable interrupt pending bit, EWIS_INTR_PEND1.LOF_PEND, is provided to propagate an interrupt to either WIS_INTA or WIS_INTB based upon mask enable bits EWIS_INTR_MASKA_1.LOF_MASKA and EWIS_INTR_MASKB_1.LOF_MASKB.

When the near-end device experiences an LOF condition, it is possible to automatically transmit a remote defect indication (RDI-L) to the far end to notify it of a problem. The EWIS_RXTX_CTRL.TXRDIL_ON_LOF, if asserted, overwrites the outgoing K2 bits with the RDI-L code.

In the receive path, it is possible to force a AIS-L state (alarm assertion plus forcing the payload to an all ones state) upon a detection of an LOF condition. This is accomplished by asserting EWIS_RXTX_CTRL.RXAISL_ON_LOF.

3.3.2.6 Section Trace (J0)

The J0 octet often carries a repeating message called the Section Trace message. The default transmitted message length is 16 octets whose contents are defined in WIS_TXJ0 (WIS_Tx_J0_Octets_1_0-WIS_Tx_J0_Octets_15_14). If no active message is being broadcast, a default section trace message is transmitted. This section trace message consists of 15 octets of zeros and a header octet formatted according to Section 5 of ANSI T1.269-2000. The header octet for the 15-octets of zero is 0x89. The default values of WIS_TXJ0 (WIS_Tx_J0_Octets_1_0-WIS_Tx_J0_Octets_15_14) do not contain the 0x89 value of the header octet, so software must write this value.

The J0 octet in the receive direction is assumed to be carrying a 16-octet continuously-repeating section trace message. The message is extracted from the incoming WIS frames and stored in WIS_RXJ0 (WIS_Rx_J0_Octets_1_0-WIS_Rx_J0_Octets_15_14). The WIS receive process does not delineate the message boundaries, thus the message might appear rotated between new frame alignment events.

The VSC8489-17 device supports two alternate message types, a single repeating octet and a 64-octet message. The message type can be independently selected for the transmit and receive direction. The transmit direction is configured using EWIS_TX_MSGLEN.J0_TXLEN, while EWIS_RX_MSGLEN.J0_RX_LEN configures the receive path.

When the transmit direction is configured for a 64-octet message, the first 16 octets are programmed in WIS_TXJ0 (WIS_Tx_J0_Octets_1_0-WIS_Tx_J0_Octets_15_14), while the 48 remaining octets are programmed in EWIS_TXJ0 (EWIS_Tx_J0_Octets_17_16-EWIS_Tx_J0_Octets_63_62). Likewise, the first 16 octets of the receive message are stored in WIS_RXJ0 (WIS_Rx_J0_Octets_1_0-WIS_Rx_J0_Octets_15_14), while the other 48 octets are stored in EWIS_RXJ0 (EWIS_Rx_J0_Octets_17_16-EWIS_Rx_J0_Octets_63_62). The receive message is updated every 125 µs with the recently received octet. Any persistency or message matching is expected to take place within the station manager.

3.3.2.7 Reserved for Section Growth (Z0)

The WIS standard does not support the Z0 octet and requires transmission of 0xCC in the octet locations. A different Z0 value can be transmitted by configuring EWIS_TX_Z0_E1.TX_Z0. The TX_Z0 default is 0xCC.

3.3.2.8 Scrambling/Descrambling

The transmit signal (except for row 1 of the section overhead) is scrambled according to the standards when register bit EWIS_TXCTRL2.SCR is asserted, which is the default state. When deasserted, the scrambler is disabled.

The receive signal descrambler is enabled by default. The descrambler can be bypassed by deasserting register bit EWIS_RX_CTRL1.DSCR_ENA.

Enabling loopback H4 and turning off the WIS scrambler and descrambler may yield an interesting data point when debugging board setups. The CRU in the ingress PMA path would not have enough edge transitions in the data to reliably recover the clock if the chip were receiving non-scrambled data. The same would be true for any far-end device connected to the egress PMA if the scrambler were turned off. The WIS scrambler and descrambler should be left on under normal operating conditions.

3.3.2.9 Section Error Monitoring (B1)

The B1 octet is a bit interleaved parity-8 (BIP-8) code using even parity calculated over the previous STS-192c frame, post scrambling. The computed BIP-8 is placed in the following outgoing SONET frame before scrambling.

In the receive direction, the incoming frame is processed, and a BIP-8 is calculated. The calculated value is then compared with the B1 value received in the following frame. The difference between the calculated and received octets are accumulated into the WIS_B1_CNT register. This counter rolls over after the maximum count. This counter is cleared upon device reset.

The EWIS_B1_ERR_CNT1 and EWIS_B1_ERR_CNT0 registers provide a count of the number of received B1 parity errors. This register is updated with the internal count value upon a PMTICK condition, after which the internal counter is reset to zero. When the counter is nonzero, the EWIS_INTR_PEND2.B1_NZ_PEND event register is asserted until read. A non-latch high version of this event, EWIS_INTR_STAT2.B1_NZ_STAT, is also available. This event can propagate an interrupt to either WIS_INTA or WIS_INTB based upon mask enable bits EWIS_INTR_MASKA_2.B1_NZ_MASKA and EWIS_INTR_MASKB_2.B1_NZ_MASKB.

The B1_ERR_CNT can optionally be configured to increment on a block count basis, a maximum increment of 1 per errored frame regardless of the number of errors received. This mode is enabled by asserting EWIS_CNT_CFG.B1_BLK_MODE.

3.3.2.10 Section Orderwire (E1)

The WIS standard does not support the E1 octet and requires transmission of 0x00 in the octet location. A different E1 value can be transmitted by configuring EWIS_TX_Z0_E1.TX_E1 (whose default is 0x00).

3.3.2.11 Section User Channel (F1)

The WIS standard does not support the F1 octet and requires transmission of 0x00 in the octet location. A different F1 value can be transmitted by configuring EWIS_TX_F1_D1.TX_F1 (whose default is 0x00).

3.3.2.12 Section Data Communication Channel (DCC-S)

The WIS standard does not support the DCC-S octets and requires transmission of 0x00 in the octet locations. Different DCC-S values can be transmitted by configuring EWIS_TX_F1_D1.TX_D1, EWIS_TX_D2_D3.TX_D2, and EWIS_TX_D2_D3.TX_D3 (all of which default to 0x00).

3.3.2.13 Reserved, National, and Unused Octets

The VSC8489-17 device transmits 0x00 for all reserved, national, and unused overhead octets.

3.3.3 Line Overhead

The line overhead portion of the SONET/SDH frame supports pointer interpretation, a per channel parity check, protection switching information, synchronization status messaging, far-end error reporting, and some OAM&P octets.

The VSC8489-17 device provides a mechanism to transmit a static value as programmed by the MDIO interface. However, by definition, MDIO is not fast enough to alter the octet on a frame-by-frame basis. The following table lists each of the octets including their function, specification, and related information.

Table 4 • **Line Overhead Octets**

Overhead Octet	Function	IEEE 802.3ae WIS Usage	Recommended Value	WIS Extension
H1-H2	Pointer	Specified value	SONET mode: STS-1: 0x62, 0x0A STS-n: 0x93, 0xFF SDH mode: STS-1: 0x6A, 0x0A STS-n: 0x9B, 0xFF	Registers EWIS_TX_C2_H1.TX_H1 and EWIS_TX_H2_H3.TX_H2 TOSI and ROSI access.
H3	Pointer action	Specified value	0x00	Register EWIS_TX_H2_H3.TX_H3 TOSI and ROSI access.

Table 4 • **Line Overhead Octets *(continued)***

Overhead Octet	Function	IEEE 802.3ae WIS Usage	Recommended Value	WIS Extension
B2	Line error monitoring (line BIP-1536)	Supported	BIP-8, as specified in T1.416	Using the TOSI, the B2 bytes can be masked for test purposes. For each B2 mask bit that is cleared to 0 on the TOSI interface, the transmitted bit is left unchanged. For each B2 mask bit that is set to 1 on the TOSI interface, the transmitted bit is inverted. Using the ROSI, the B2 error locations can be extracted. Periodically latched counter (EWIS_B1_ERR_CNT1-EWIS_B1_ERR_CNT0) is available.
K1, K2	Automatic protection switch (APS) channel and line remote defect identifier (RDI-L)	Specified value	For more information about K2 coding, see Table 5, page 21	Register Registers EWIS_TX_G1_K1.TX_K1 and EWIS_TX_K2_F2.TX_K2 TOSI and ROSI access.
D4-D12	Line data communications channel (DCC)	Unsupported	0x00	Registers EWIS_TX_D4_D5 and EWIS_TX_D6_H4 TOSI and ROSI access.
S1	Synchronization messaging	Unsupported	0x0F	Register EWIS_TX_S1_Z1.TX_S1 TOSI and ROSI access.
Z1	Reserved for Line growth	Unsupported	0x00	Register EWIS_TX_S1_Z1.TX_Z1 TOSI and ROSI access.
M0/M1	STS-1/N line remote error indication (REI)	M0 unsupported, M1 supported	0x00/number of detected B2 errors in the receive path, as specified in T1.416	TOSI and ROSI access. The VSC8489-17 device supports a mode that uses only M1 to back report REI-L (EWIS_MODE_CTR.REI_MODE = 0) and another mode which uses both M0 and M1 to back report REI-L (EWIS_MODE_CTR.REI_MODE = 1). For more information, see Line Error Monitoring (B2), page 20.
E2	Orderwire	Unsupported	0x00	Register EWIS_TX_Z2_E2.TX_E2 TOSI and ROSI access.
Z2	Reserved for Line growth	Unsupported	0x00	Register EWIS_TX_Z2_E2.TX_Z2 TOSI and ROSI access.

3.3.3.1 Line Error Monitoring (B2)

The B2 octet is a BIP-8 value calculated over each of the previous STS-1 channels excluding the section overhead and pre-scrambling. As the B2 octet is calculated on an STS-1 basis, there are 192 B2 octets within an STS-192/STM-64 frame. Each of the 192 calculated BIP-8 octets are then placed in the outgoing SONET/SDH frame.

Note: For SONET mode, when the number of errors detected in the B2 octet of a receive frame is greater than 255, the total count of detected errors is transmitted in more than one frame. Even when no B2 errors are detected in subsequent frames, the number of detected B2 errors going into an accumulator will be limited to 255 if more than 255 errors are detected in a frame. The Tx framer pulls the REI-L count out of the accumulator when REI-L is transmitted to be compliant with T1-105.

In the receive direction, the incoming frame is processed, a per STS-1 BIP-8 is calculated (excluding section overhead and after descrambling), and then compared to the B2 value in the following frame. Errors are accumulated in the WIS_B2_CNT1 and WIS_B2_CNT0 registers. This counter is non-saturating and so rolls over after its maximum count. The counter is cleared only on device reset.

An additional 32-bit B2 error counter is provided in B2_ERR_CNT (EWIS_B2_ERR_CNT1 and EWIS_B2_ERR_CNT0), which is a saturating counter and is latched and cleared based upon a PMTICK event. Errors are accumulated from the previous PMTICK event. When the counter is nonzero, the EWIS_INTR_PEND2.B2_NZ_PEND event register is asserted until read. A non-latch high version of this event is available in EWIS_INTR_STAT2.B2_NZ_STAT. This event can propagate an interrupt to either WIS_INTA or WIS_INTB, based on mask enable bits EWIS_INTR_MASKA_2.B2_NZ_MASKA and EWIS_INTR_MASKB_2.B2_NZ_MASKB.

The B2_ERR_CNT can optionally be configured to increment on a block count basis, a maximum increment of 1 per errored frame regardless of the number of errors received. This mode is enabled by asserting EWIS_CNT_CFG.B2_BLK_MODE.

It is possible that two sets of B2 bytes (from two SONET/SDH frames) are received by the Rx WIS logic in a period of time when only one M0/M1 octet is transmitted. In this situation, one of the two B2 error counts delivered to the Tx WIS logic is discarded. This situation occurs when the receive data rate is faster than the transmit data rate. Similarly, when the transmit data rate is faster than the receive data rate, a B2 error count is not available for REI-L insertion into the M0/M1 octets of the transmitted SONET/SDH frame. A value of zero is transmitted in this case. This behavior is achieved by using a FIFO to transfer the detected B2 error count from the receive to transmit domains.

A FIFO overflow or underflow condition is not considered an error. Instead, it is recovered from gracefully, as described above. A FIFO overflow or underflow eventually occurs unless the transmit and receive interfaces are running at the same average data rate. Because the received and transmitted frames can differ by, at most, 40 ppm (±20 ppm) and still meet the industry standards, this “slip” can happen no more often than once every 3.1 seconds.

3.3.3.2 APS Channel and Line Remote Defect Identifier (K1, K2)

The K1 and K2 octets carry information regarding automatic protection switching (APS) and line remote defect identifier (RDI-L). The K1 octet and the most significant five bits of the K2 octet contain the APS channel information. The transmitted values can be configured at EWIS_TX_G1_K1.TX_K1 and EWIS_TX_K2_F2.TX_K2. The default values of all zeros are compliant with the WIS standard.

The three least significant bits within the K2 octet carry the RDI-L encoding, as defined by section 7.4.1 of ANSI T1.416-1999 and as shown in the following table.

Table 5 • **K2 Encodings**

Indicator	K2 Value for Bits 6, 7, 8	Interpretation
RDI-L	110	Remote error indication. For the receive process, an RDI-L defect occurs after a programmable number of RDI-L signals are received in contiguous frames and is terminated when no RDI-L is received for the same number of contiguous frames. An RDI-L can be forced by asserting EWIS_RX_ERR_FRC1.FRC_RX_RDIL. For the transmit process, the WIS standard does not indicate when or how to transmit RDI-L. VSC8489-17 provides the option of transmitting K2 by programming it through the TOSI, by programming it using the K2_TX MDIO register, or by programming it based on the contents of the K2_TX register with bits 6, 7, and 8 modified depending on the status of the following: LOPC, LOS, LOF, AIS-L and their associated transmit enable bits enable bits TXRDIL_ON_LOPC, TXRDIL_ON_LOS, TXRDIL_ON_LOF and TXRDIL_ON_AISL in register EWIS_RXTX_CTRL.
AIS-L	111	Alarm indication signal (line). For the receive process, this is detected based on the settings of the K2 byte. When AIS-L is detected, the WIS link status is down and WIS_STAT3.AISL is set high. This also contributes to errored second (ES) and severally errored second (SES) reports. For standard WIS operation, this is never transmitted.
Idle (normal)	000	Unless RDI-L exists, the standard WIS transmits idle.

Although the transmission of RDI-L is not explicitly defined within the WIS standard, the VSC8489-17 device allows the automatic transmission of RDI-L upon the detection of LOPC, LOS, LOF, or AIS-L conditions. These features are enabled by asserting TXRDIL_ON_LOPC, TXRDIL_ON_LOS, TXRDIL_ON_LOF and TXRDIL_ON_AISL in register EWIS_RXTX_CTRL.

Note: The RDI-L code of 110 is transmitted by the DUT only when Rx AIS-L is asserted. For example, if AIS-L is detected by the DUT for five continuous frames in the Rx direction, then the RDI-L code is transmitted for five frames in the Tx direction (not 20 frames as stated in the ANSI T1.105 specification).

The VSC8489-17 device can force a RDI-L condition independent of the K2 transmit value by asserting EWIS_TXCTRL2.FRC_TX_RDI. Likewise, a AIS-L condition can be forced by asserting EWIS_TXCTRL2.FRC_TX_AISL. If both conditions are forced, the AIS-L value is transmitted.

In the receive direction, the RDI-L alarm (K2[6:8] = 110, using SONET nomenclature) and the AIS-L alarm (K2[6:8] = 111, using SONET nomenclature) are not asserted until the condition persists for a programmable number of contiguous frames. This value is programmable at EWIS_RX_ERR_FRC1.APS_THRES and is typically set to values of 5 or 10. The AIS-L is detected by the receiver after the programmable number of frames is received, and results in the reporting of AIS-P.

The WIS standard defines WIS_STAT3.RDIL and WIS_STAT3.AISL as a read-only latch-high register, so a read of a one in this register indicates that an error condition occurred since the last read. A second read of the register provides the current status of the event as to whether the alarm is currently asserted. EWIS_INTR_PEND1.RDIL_PEND and EWIS_INTR_PEND1.AISL_PEND assert whenever the RDI-L or AIS-L state changes (assert or deassert). These interrupts have associated mask enable bits (EWIS_INTR_MASKA_1.RDIL_MASKA, EWIS_INTR_MASKB_1.RDIL_MASKB, EWIS_INTR_MASKA_1.AISL_MASKA and EWIS_INTR_MASKB_1.AISL_MASKB), which, if enabled, propagate an interrupt to the WIS_INTA/B pins.

For test purposes, the VSC8489-17 device can induce a RDI-L condition in the receive direction independent of the received K2 value by asserting EWIS_RX_ERR_FRC1.FRC_RX_RDIL. Likewise, a AIS-L condition can be forced in the receive direction by asserting EWIS_RX_ERR_FRC1.FRC_RX_AISL.

3.3.3.3 Line Data Communications Channel (D4 to D12)

The WIS standard does not support Line Data Communications Channel (L-DCC) octets (D4-D12) and recommends transmitting 0x00 within these octets. The D4-D12 transmitted values can be programmed in registers EWIS_TX_D4_D5 - EWIS_TX_D12_Z4. The register defaults are all 0x00. The receive L-DCC octets are only accessible through the ROSI port.

3.3.3.4 STS-1/N Line Remote Error Indication (M0 and M1)

The M0 and M1 octets are used for back reporting the number of B2 errors received, known as remote error indication (REI-L). The value in this octet comes from the B2 error FIFO, as discussed with the B2 octet. The WIS standard does not support the M0 octet and recommends transmitting 0x00 in place of the M0 octet. However, the WIS standard supports the M1 octet in accordance with T1.416.

Two methods for back-reporting exist and are controlled by EWIS_TXCTRL2.SDH_TX_MODE. Because a single frame can contain up to 1536 B2 errors while the M1 byte alone can only back report a maximum of 255 errors, a discrepancy exists. When G707_2000_REIL is deasserted, only the M1 byte is used and a maximum of 255 errors are back-reported. When G707_2000_REIL is asserted, two octets per frame are used for back reporting- the M1 octet and the M0 octet (not the first STS-1 octet, but the second STS-1 octet). In this mode, a total of 1536 errors can be back-reported per frame.

In the receive direction, the VSC8489-17 device detects and accumulates errors according to the EWIS_MODE_CTRL.REI_MODE setting. The VSC8489-17 device deviates from the G.707 standard by not interpreting REI-L values greater than 1536 as zero. The WIS standard defines a 32-bit REI-L counter in registers WIS_REIL_CNT1 and WIS_REIL_CNT0. This counter is non-saturating and so rolls over after its maximum count. The counter is cleared only on device reset.

An additional 32-bit REI-L counter is provided in registers EWIS_REIL_CNT1 and EWIS_REIL_CNT0, which is a saturating counter and is latched and cleared based upon a PMTICK event. Errors are accumulated since the previous PMTICK event. When the counter is nonzero, the EWIS_INTR_PEND2.REIL_NZ_PEND event register is asserted until read. A non-latch high version of this event (EWIS_INTR_STAT2.REIL_NZ_STAT) is also available. This event can propagate an interrupt to either WIS_INTA or WIS_INTB based upon mask enable bits EWIS_INTR_MASKA_2.REIL_NZ_MASKA and EWIS_INTR_MASKB_2.REIL_NZ_MASKB.

The REIL_ERR_CNT can optionally be configured to increment on a block count basis, a maximum increment of 1 per errored frame regardless of the number of errors received. This mode is enabled by asserting EWIS_CNT_CFG.REIL_BLK_MODE.

3.3.3.5 Synchronization Messaging (S1)

The S1 octet carries the synchronization status message and provides synchronization quality measures of the transmission link in the least significant 4 bits. The WIS standard does not support the S1 octet and requires the transmission of a 0x0F within the S1 octet. A value other than 0x0F can be programmed in TX_S1 (2xE61F).

3.3.3.6 Reserved for Line Growth (Z1 and Z2)

The WIS standard does not support the Z1 or Z2 octets and requires the transmission of 0x00 in their locations. Different Z1 and Z2 values can be transmitted by programming the values at EWIS_TX_S1_Z1.TX_Z1 and EWIS_TX_Z2_E2.TX_Z2 respectively.

3.3.3.7 Orderwire (E2)

The WIS standard does not support the E2 octet and recommends transmitting 0x00 in place of the E2 octet. A value other than 0x00 can be transmitted by programming the intended value at EWIS_TX_Z2_E2.TX_E2.

3.3.4 SPE Pointer

The H1 and H2 octets are used as a pointer within the SONET/SDH frame to locate the beginning of the path overhead and the beginning of the synchronous payload envelope (SPE). Within SONET/SDH the SPE can begin anywhere within the payload area. However, IEEE 802.3ae specifies that a transmitted SPE must always be positioned solely within a single SONET/SDH frame. The constant pointer value of 522 decimal (0x20A) must be contained in the first channel's H1 and H2 octets. Together these conditions result in the H1 and H2 octets being 0x62 and 0x0A, respectively. These are the default values of EWIS_TX_C2_H1.TX_H1 and EWIS_TX_H2_H3.TX_H2. Programming these registers with alternate values does not alter the positioning of the SPE, but it might induce a loss of pointer (LOP-P) at the far-end, or at least prevent the far-end from extracting the proper payload. Furthermore, the WIS standard specifies the frame structure be a concatenated payload. For this reason, the H1 and H2 octets in channels 2 through 192 contain the concatenation indicator.

The VSC8489-17 device supports forcing the loss of pointer (LOP-P) and path alarm indication signal (AIS-P) state.

The WIS standard specifies that a 0×00 be transmitted in the H3 octet. An alternate value can be transmitted by programming EWIS_TX_H2_H3.TX_H3.

The WIS specification does not limit the pointer position within the receive SONET/SDH frame to allow interoperability to other SONET/SDH equipment. In addition to supporting the required SONET pointer rules, the VSC8489-17 device pointer interpreter optionally supports SDH pointers. This is selectable using the EWIS_MODE_CTRL.RX_SS_MODE bit. The following table shows the differences between SONET and SDH modes.

Table 6 • **SONET/SDH Pointer Mode Differences**

SONET	SDH
SS bits are ignored by the device pointer interpreter and not used	SS bits are set to 10 and are checked by the device pointer interpreter to determine the pointer type
All 192 bytes of H1 and H2 are checked by the pointer interpreter to determine the pointer type	The first 64 bytes are checked by the pointer interpreter to determine the pointer type (first Au-4 of an AU-4-64c)
Uses ‘8 out of 10’ GR-253-core objective increment/decrement rule	Uses majority detect increment/decrement rule

The H1 and H2 octets combine to form a word with several fields, as shown in Figure 14, page 24.

3.3.4.1 Bit Designations within Payload Pointer

The N bits [15:12] carry a new data flag (NDF). This mechanism allows an arbitrary change in the location of the payload. NDF is indicated by at least three out of the four N bits matching the code ‘1001’ (NDF enabled). Normal operation is indicated by three out of the four N bits matching the code ‘0110’ (normal NDF).

The last ten bits of the pointer word (D bits and I bits) carry the pointer value. The pointer value has a range from 0 to 782 that indicates the offset between the first byte after the H3 byte and the first byte of the SPE.

The SS bits are located in bits 11 and 10, and are unused in SONET mode. In SDH mode, these bits are compared with pattern ‘10’, and the pointer is considered invalid if it does not match.

Because the VSC8489-17 device only supports concatenated frames, only the first pair of bytes (H1, H2) are called the primary pointer and have a normal format. The

rest of the H1/H2 bytes contain the concatenation indication (CI). The format for the CI is NDF enabled with a pointer value of all ones.

Figure 14 • **16-bit Designations within Payload Pointer**

H1								H2							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
N	N	N	N	S	S	I	D	I	D	I	D	I	D	I	D

3.3.4.2 Pointer Types

The VSC8489-17 device supports five different pointer types, as described in the following table. A normal pointer indicates the current pointer, a new data flag indicates a new pointer location, and an AIS pointer indicates AIS. The pointer increment and pointer decrement mechanism adjusts the frequency offset between the frame overhead and SPE. A pointer increment is indicated by a normal NDF that has the currently accepted pointer with the I bits inverted. A pointer decrement is indicated by a normal NDF that has the currently accepted pointer with D bits inverted.

Table 7 • **H1/H2 Pointer Types**

Pointer Type	nnnn Value	Pointer Value	SS bits
Normal	Three out of the four bits matching 0110	0 to 782	Matching in SDH mode, ignored in SONET mode
New data flag (NDF)	Three out of the four bits matching 1001	0 to 782	Matching in SDH mode, ignored in SONET mode
AIS pointer	1111	1111 1111 11	11
Pointer increment	Three out of the four bits matching 0110	Current pointer with I bits inverted	Matching in SDH mode, ignored in SONET mode
Pointer decrement	Three out of the four bits matching 0110	Current pointer with D bits inverted	Matching in SDH mode, ignored in SONET mode

Table 8 • **Concatenation Indication Types**

Pointer Type	*nnnn* Value	Pointer Value	SS bits
Normal concatenation indication	Three out of the four bits matching 1001	1111 1111 11	Matching in SDH mode, ignored in SONET mode
AIS concatenation indication	Pointer value, *nnnn* value, and SS bits are the same as the AIS pointer		
Invalid concatenation indication	Any other concatenation indication other than normal CI or AIS CI		

3.3.4.3 Pointer Adjustment Rule

The VSC8489-17 device pointer interpreter adjusts the current pointer value according to rules listed in Section 9.1.6 of ANSI T1.105-1995. In addition, no increment/decrement is accepted for at least three frames following an increment/decrement or NDF operation.

3.3.4.4 Pointer Increment/Decrement Majority Rules

In SONET mode, the pointer interpreter uses more restrictive GR-253-CORE objective rules, as follows:

- An increment is indicated by eight or more bits matching non-inverted D bits and inverted I bits.
- A decrement is indicated by eight or more bits matching non-inverted I bits and inverted D bits.

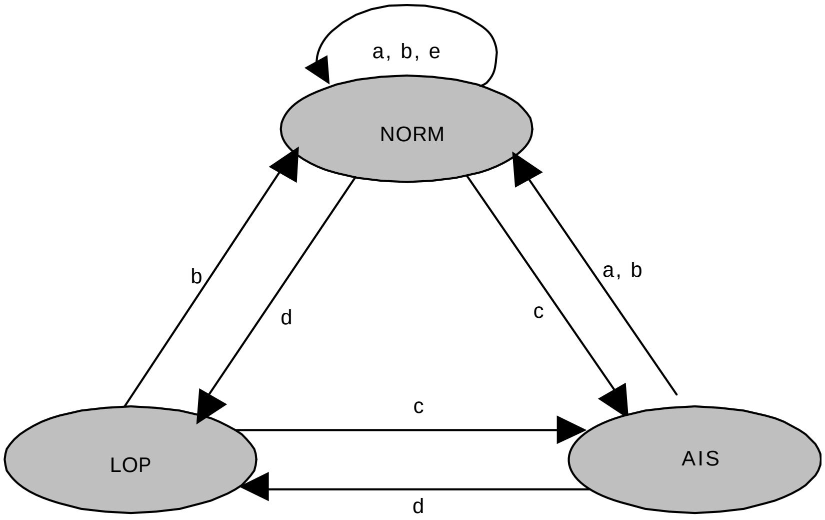
In SDH mode, the majority rules are:

- An increment is indicated by three or more inverted I bits and two or fewer inverted D bits.
- A decrement is indicated by three or more inverted D bits and two or fewer inverted I bits.
- If three or more D bits are inverted and three or more I bits are inverted, no action is taken.

3.3.4.5 Pointer Interpretation States

The pointer interpreter algorithm for state transitions can be modeled as a finite state machine with three states, as shown in the following illustration. The three states are normal (NORM), loss of pointer (LOP), and alarm indication state (AIS).

Figure 15 • **Pointer Interpreter State Diagram**



The conditions for transitions between these states are summarized in the following table

Table 9 • **Pointer Interpreter State Diagram Transitions**

Transitions	States	Description	Required Persistence
a	NORM → NORM AIS → NORM	<H1><H2>=<EEEESSPP><PPPPPPPP>. NDF enabled with pointer in range (0 to 782). SS bit match (if enabled).	1 frame
b	NORM → NORM LOP → NORM AIS → NORM	<H1><H2>=<DDDDSSPP><PPPPPPPP>. NDF disabled (NORM pointer) with the same pointer value in range (0 to 782). SS bit match (if enabled).	3 frames
c	NORM → AIS LOP → AIS	<H1><H2>=<11111111><11111111>. AIS pointer (0xFFFF).	3 frames
d	NORM → LOP AIS → LOP	Anything other than transitions b and c or NDF enabled (transition a) or AIS pointer when not in AIS state or NORM pointer when not in NORM state or NORM pointer with pointer value not equal to current or increment/decrement or CONC pointer or SS bit mismatch (if comparison is enabled).	8 frames
e	Justification	Valid increment or decrement indication.	1 frame

3.3.4.6 Valid Pointer Definition for Interpreter State Diagram Transitions

During an AIS state, only an AIS pointer is a valid pointer. In NORM state, several definitions of “valid pointer” for purpose of LOP detection are possible, according to GR-253-CORE. The VSC8489-17 device follows the GR-253-CORE intended definition, but adds a single normal pointer that exactly matches the current valid pointer value.

Any change in the AIS state is reflected in the alarm bit WIS_STAT3.AISP. This latch-high register reports both the event and status information in consecutive reads. The EWIS_INTR_PEND1.AISP_PEND bit remains asserted until read. This event can propagate an interrupt to either WIS_INTA or WIS_INTB, based on mask enable bits EWIS_INTR_MASKA_1.AISP_MASKA and EWIS_INTR_MASKB_1.AISP_MASKB.

Similarly, any change in the LOP state is reflected in the alarm bit WIS_STAT3.LOPP. This latch-high register reports both the event and status information in consecutive reads. The EWIS_INTR_PEND1.LOPP_PEND bit remains asserted until read. This event can propagate an interrupt to either WIS_INTA or WIS_INTB, based upon the mask enable bits EWIS_INTR_MASKA_1.LOPP_MASKA and EWIS_INTR_MASKB_1.LOPP_MASKB.

3.3.5 Path Overhead

The path overhead portion of the SONET/SDH frame supports an end-to-end trace identifier, a payload parity check, a payload type indicator, a status indicator, and a user channel. The following table lists each of the octets, including their function.

Note: The VSC8489-17 device provides a mechanism to transmit a static value as programmed by the MDIO interface. However, by definition, MDIO is not fast enough to alter the octet on a frame-by-frame basis. Extended WIS TOSI and ROSI do not support path overhead.

Table 10 • **STS Path Overhead Octets**

Overhead Octet	Function	IEEE 802.3ae WIS Usage	Recommended Value	WIS Extension
J1	Path trace message	Specified value	For more information, see Overhead Octet (J1), page 27	A 1-, 16-, or 64-byte trace message can be sent using registers (EWIS_TX_MSGLEN.J1_TXLEN, WIS_Tx_J1_Octets_1_0-WIS_Tx_J1_Octets_15_14 and EWIS_Tx_J1_Octets_17_16-EWIS_Tx_J1_Octets_63_62) and received using registers (EWIS_RX_MSGLEN.J1_RX_LEN, WIS_Rx_J1_Octets_1_0-WIS_Rx_J1_Octets_15_14, EWIS_Rx_J1_Octets_17_16-EWIS_Rx_J1_Octets_63_62).TOSI and ROSI access.
B3	Path error monitoring (path BIP-8)	Supported	Bit interleaved parity - 8 bits, as specified in T1.416	Both SONET and SDH mode B3 calculation is supported.
C2	Path signal label	Specified value	0x1A	Register (EWIS_TX_C2_H1.TX_C2). Supports persistency and mismatch detection (EWIS_MODE_CTRL.C2_EXP).
G1	Path status	Supported	As specified in T1.416	Ability to select between RDI-P and ERDI-P formats.
F2	Path user channel	Unsupported	0x00	Register (EWIS_TX_K2_F2.TX_F2).
H4	Multiframe indicator	Unsupported	0x00	Register (EWIS_TX_D6_H4.TX_H4).
Z3-Z4	Reserved for path growth	Unsupported	0x00	Register (EWIS_TX_D9_Z3.TX_Z3, EWIS_TX_D12_Z4.TX_Z4).
N1	Tandem connection maintenance and path data channel	Unsupported	0x00	Register (EWIS_TX_N1.TX_N1). TOSI and ROSI access.

3.3.5.1 Overhead Octet (J1)

The J1 transmitted octet contains a 16-octet repeating path trace message whose contents are defined in WIS Tx J1s (WIS_Tx_J1_Octets_1_0-WIS_Tx_J1_Octets_15_14). If no active message is being broadcast, a default path trace message is transmitted, consisting of 15 octets of zeros and a header octet formatted according to Section 5 of ANSI T1.269-2000. The header octet for the 15-octets of zero is 0x89. The default values of WIS Tx J1s do not contain the 0x89 value of the header octet, thus software must write this value.

By default, the J1 octet in the receive direction is assumed to be carrying a 16-octet continuously repeating path trace message. The message is extracted from the incoming WIS frames and presented in WIS Rx J1s (WIS_Rx_J1_Octets_1_0-WIS_Rx_J1_Octets_15_14). The WIS receive process does not delineate the message boundaries, thus the message might appear rotated between new frame alignment events.

The VSC8489-17 device supports two alternate message types, a single repeating octet and a 64-octet message. The message type can be independently selected for the transmit and receive direction. The transmit direction is configured using EWIS_TX_MSGLEN.J1_TXLEN while EWIS_RX_MSGLEN.J1_RX_LEN configures the receive path.

When the transmit direction is configured for a 64-octet message, the first 16 octets are programmed in WIS_Tx_J1_Octets_1_0-WIS_Tx_J1_Octets_15_14, while the 48 remaining octets are programmed in EWIS_Tx_J1_Octets_17_16-EWIS_Tx_J1_Octets_63_62. Likewise, the first 16-octets of the receive message are stored in J1_RXMSG (WIS_Rx_J1_Octets_1_0-WIS_Rx_J1_Octets_15_14), while the other 48 octets are stored in EWIS_Rx_J1_Octets_17_16-EWIS_Rx_J1_Octets_63_62. The receive message is updated every 125 µs with the recently received octet. Any persistence or message matching is expected to take place within the station manager.

3.3.5.2 STS Path Error Monitoring (B3)

The B3 octet is a bit interleaved parity-8 (BIP-8) code, using even parity, calculated over the previous STS-192c SPE before scrambling. The computed BIP-8 is placed in the B3 byte of the following frame before scrambling.

In the receive direction, the incoming frame is processed and a B3 octet is calculated over the received frame. The calculated value is then compared with the B3 value received in the following frame. The difference between the calculated and received octets are accumulated in block (maximum increment of 1 per errored frame) fashion into a B3 error register, WIS_B3_CNT. This counter is non-saturating and so rolls over. The counter is cleared upon a device reset.

An additional 32-bit B3 error counter is provided at B3_ERR_CNT (EWIS_B3_ERR_CNT1 and EWIS_B3_ERR_CNT0), a saturating counter that is latched and cleared based upon a PMTICK event. Errors are accumulated starting from the previous PMTICK event. When the counter is nonzero, the EWIS_INTR_PEND2.B3_NZ_PEND event register is asserted until read. A non-latch high version of this event EWIS_INTR_STAT2.B3_NZ_STAT is also available. This event may propagate an interrupt to either WIS_INTA or WIS_INTB, based on the mask enable bits EWIS_INTR_MASKA_2.B3_NZ_MASKA and EWIS_INTR_MASKB_2.B3_NZ_MASKB.

The B3_ERR_CNT may optionally be configured to increment on a block count basis, a maximum increment of 1 per errored frame regardless of the number of errors received. The EWIS_CNT_CFG.B3_BLK_MODE control bit, if asserted, places the B3_ERR_CNT counter in block increment mode.

It is possible that two sets of B3 bytes (from two SONET/SDH frames) are received by the Rx WIS logic in a period of time when only one G1 octet is transmitted. In this situation, one of the two B3 error counts delivered to the Tx WIS logic is discarded. This situation occurs when the receive data rate is faster than the transmit data rate. Similarly, when the transmit data rate is faster than the receive data rate, a B3 error count is not available for REI-P insertion into the G1 octets of the transmitted SONET/SDH frame. A value of zero is transmitted in this case. This behavior is achieved by using a FIFO to transfer the detected B3 error count from the receive to transmit domains.

3.3.5.3 STS Path Signal Label and Path Label Mismatch (C2)

The C2 octet contains a value intended to describe the type of payload carried within the SONET/SDH frame. The WIS standard calls for a 0x1A to be transmitted. This is the default value of EWIS_TX_C2_H1.TX_C2.

As specified in T1.416, a path label mismatch (PLM-P), register WIS_STAT3.PLMP, event occurs when the C2 octet in five consecutive frames contain a value other than the expected one. The expected value is set in EWIS_MODE_CTRL.C2_EXP, whose default value 0x1A is compliant with the WIS standard.

When a value of 0x00 is accepted (received for five or more consecutive frames) the unequipped path pending (EWIS_INTR_PEND2.UNEQP_PEND) event is asserted until read. A non-latch high version of this event (EWIS_INTR_STAT2.UNEQP_STAT) is also available. This event can propagate an interrupt to either WIS_INTA or WIS_INTB, based on the mask enable bits EWIS_INTR_MASKA_2.UNEQP_MASKA and EWIS_INTR_MASKB_2.UNEQP_MASKB.

If the accepted value is not an unequipped label (0x00) and it differs from the programmed expected value, EWIS_MODE_CTRL.C2_EXP, then a path label mismatch (WIS_STAT3.PLMP) is asserted. Similarly the EWIS_INTR_PEND1.PLMP_PEND event is asserted until read. This event can propagate an interrupt to either WIS_INTA or WIS_INTB, based on the mask enable bits EWIS_INTR_MASKA_1.PLMP_MASKA and EWIS_INTR_MASKB_1.PLMP_MASKB.

Although PLMP is not a path level defect, it does cause a change in the setting of one of the ERDI-P codes. For more information, see Table 13, page 29.

3.3.5.4 Remote Path Error Indication (G1)

The most significant four bits of the G1 octet are used for back reporting the number of B3 block errors received at the near-end. This is typically known as path remote error indication (REI-P). The value in this octet comes from the B3 error FIFO. The WIS standard defines a 16-bit REI-P counter, register WIS_REIP_CNT. The WIS standard defines this counter to operate as a block counter as opposed to an individual errored bit counter. This counter is non-saturating and so rolls over after its maximum count. The counter does not clear upon a read, but instead only upon reset as defined in the WIS specification. When the counter is nonzero, the EWIS_INTR_PEND2.REIP_PEND event register is asserted until read. A non-latch high version of this event EWIS_INTR_STAT2.REIP_STAT is also available. This event may propagate an interrupt to either WIS_INTA or WIS_INTB, based on the mask enable bits EWIS_INTR_MASKA_2.REIP_MASKA and EWIS_INTR_MASKB_2.REIP_MASKB, respectively.

An additional 32-bit REI-P counter is provided at REIP_ERR_CNT (EWIS_REIP_CNT1 and EWIS_REIP_CNT0), which is a saturating counter, and is latched and cleared based upon a PMTICK event. Errors are accumulated since the previous PMTICK event. When the counter is nonzero, the EWIS_INTR_PEND2.REIP_NZ_PEND event register is asserted until read. A non-latch high version of this event (EWIS_INTR_STAT2.REIP_NZ_STAT) is also available. This event may propagate an interrupt to either WIS_INTA or WIS_INTB, based on the mask enable bits EWIS_INTR_MASKA_2.REIP_NZ_MASKA and EWIS_INTR_MASKB_2.REIP_NZ_MASKB, respectively.

The REIP_ERR_CNT may optionally be configured to increment on a block count basis, a maximum increment of 1 per errored frame, regardless of the number of errors received. This mode is enabled by asserting EWIS_CNT_CFG.REIP_BLK_MODE.

3.3.5.5 Path Status (G1)

In addition to back-reporting the far-end B3 BIP-8 error count, the G1 octet carries status information from the far-end device known as path remote defect indicator (RDI-P). T1.416 allows either support of 1-bit RDI-P or 3-bit ERDI-P, but indicates ERDI-P is preferred. The VSC8489-17 device supports both modes and may be independently configured for the Rx and Tx directions by configuring EWIS_MODE_CTRL.RX_ERDI_MODE and EWIS_TXCTRL2.ERDI_TX_MODE. ERDI-P is the default for both directions.

The following tables show the different structures for this octet.

Table 11 • Path Status (G1) Byte for RDI-P Mode

G1 REI (B3)				RDI-P	Reserved		Spare
1	2	3	4	5	6	7	8
Remote Error Indicator count from B3 (0–8 value)				Remote Defect indicator	Set to 00 by transmitter		Ignored by receiver

Table 12 • Path Status (G1) Byte for ERDI-P Mode

G1 REI (B3)				ERDI-P			Spare
1	2	3	4	5	6	7	8
Remote Error Indicator count from B3 (0–8 value)				Enhanced Remote Defect Indicator (see following table)			Ignored by receiver

Enhanced RDI is defined for SONET-based systems as listed in GR-253-CORE (Issue 3), reproduced here in the following table, and as a possible enhancement of SDH-based systems (G.707/Y.1322 (10/2000) Appendix VII (not an integral part of that recommendation)).

Table 13 • RDI-P and ERDI-P Bit Settings and Interpretation

G1 Bits 5, 6, and 7	Priority of ERDI-P Codes	Trigger	Interpretation
000/011	Not applicable	No defects.	No RDI-P defect
100/111	Not applicable	Path alarm indication signal (AIS-P). The remote device sends all ones for H1, H2, H3, and the entire STS SPE. Path loss of pointer (LOP-P).	One-bit RDI-P defect
001	4	No defects.	No ERDI-P defect
010	3	Path label mismatch (PLM-P). Path loss of code group delineation (LCD-P).	ERDI-P payload defect
101	1	Path alarm indication signal (AIS-P). The remote device sends all ones for H1, H2, H3 and entire STS SPE. Path loss of pointer (LOP-P).	ERDI-P server defect
110	2	Path unequipped (UNEQ-P). The received C2 byte is 0x00. Path trace identifier mismatch (TIM-P). This error is not automatically generated, but can be forced using MDIO.	ERDI-P connectivity defect

In the receive direction, with EWIS_MODE_CTRL.RX_ERDI_MODE = 0, an RDI-P defect is the occurrence of the RDI-P signal in ten contiguous frames. An RDI-P defect terminates when no RDI-P signal is detected in ten contiguous frames. An RDI-P event asserts EWIS_INTR_PEND2.FERDIP_PEND until read. A non-latch high version of the far-end RDI-P status can be found in EWIS_INTR_STAT2.FERDIP_STAT. This event may propagate an interrupt to either WIS_INTA or WIS_INTB, based on the mask enable bits EWIS_INTR_MASKA_2.FERDIP_MASKA and EWIS_INTR_MASKB_2.FERDIP_MASKB.

When EWIS_MODE_CTRL.RX_ERDI_MODE = 1, an ERDI-P defect is the occurrence of any one of three ERDI-P signals in ten contiguous frames. An ERDI-P defect terminates when no ERDI-P signal is detected in ten contiguous frames.

The 010 code triggers the latch high register bit WIS_STAT3.FEPLMP_LCDP. It also asserts EWIS_INTR_PEND1.FEPLMP_LCDP_PEND until read. This event may propagate an interrupt to either WIS_INTA or WIS_INTB, based on the mask enable bits EWIS_INTR_MASKA_1.FEPLMP_LCDP_MASKA and EWIS_INTR_MASKB_1.FEPLMP_LCDP_MASKB, respectively.

The 101 code triggers the latch high register bit WIS_STAT3.FEAISP_LOPP. It also asserts EWIS_INTR_PEND1.FEAISP_LOPP_PEND until read. This event may propagate an interrupt to either WIS_INTA or WIS_INTB, based on the mask enable bits EWIS_INTR_MASKA_1.FEAISP_LOPP_MASKA and EWIS_INTR_MASKB_1.FEAISP_LOPP_MASKB, respectively.

The 110 code asserts the EWIS_INTR_PEND2.FEUNEQP_PEND until read. A non-latch-high version of this register (EWIS_INTR_STAT2.FEUNEQP_STAT) is also available. This event may propagate an interrupt to either WIS_INTA or WIS_INTB, based on the mask enable bits EWIS_INTR_MASKA_2.FERDIP_MASKA and EWIS_INTR_MASKB_2.FERDIP_MASKB, respectively.

3.3.5.6 Path User Channel (F2)

The WIS standard does not support the F2 octet and recommends transmitting 0x00 in place of the F2 octet. A value other than 0x00 may be transmitted by programming the intended value at EWIS_TX_K2_F2.TX_F2.

3.3.5.7 Multi-frame Indicator (H4)

The WIS standard does not support the H4 multi-frame octet and recommends transmitting 0x00 in place of the H4 octet. A value other than 0x00 may be transmitted by programming the intended value at EWIS_TX_D6_H4.TX_H4.

3.3.5.8 Reserved for Path Growth (Z3-Z4)

The WIS standard does not support the Z3-Z4 octets and recommends transmitting 0x00 in their place. A value other than 0x00 may be transmitted by programming the intended value at EWIS_TX_D9_Z3.TX_Z3 and EWIS_TX_D12_Z4.TX_Z4 respectively.

3.3.5.9 Tandem Connection Maintenance/Path Data Channel (N1)

The WIS standard does not support the N1 octet and recommends transmitting 0x00 in place of the N1 octet. A value other than 0x00 may be transmitted by programming the intended value at EWIS_TX_N1.TX_N1.

3.3.5.10 Loss of Code Group Delineation

After the overhead is stripped, the payload is passed to the PCS. If the PCS block loses synchronization and cannot delineate valid code groups, the PCS passes a loss of code group delineation (LCD-P) alarm to the WIS. This alarm triggers the latch high register bit WIS_STAT3.LCDP. It also asserts EWIS_INTR_PEND1.LCDP_PEND until read. This event may propagate an interrupt to either WIS_INTA or WIS_INTB, based on the mask enable bits EWIS_INTR_MASKA_1.LCDP_MASKA and EWIS_INTR_MASKB_1.LCDP_MASKB, respectively.

The WIS specification calls for a LCD-P defect persisting continuously for more than 3 ms to be back reported to the far-end. Upon device reset, a LCD-P is back reported until the PCS signals that valid code groups are being delineated. The LCD-P defect deasserts (and is not back reported) after the condition is absent continuously for at least 1 ms.

3.3.5.11 Reading Statistical Counters

The VSC8489-17 device contains several counters that may be read using the MDIO interface. For each error count, there are two sets of counters. The first set is the standard WIS counter implemented according to IEEE 802.3ae, and the second set is for statistical counts using PMTICK.

To read the IEEE 802.3ae counters, the station manager must read the most significant register of the 32-bit counter first. This read action latches the internal error counter value into the MDIO readable registers. A subsequent read of the least significant register does not latch new values, but returns the value latched at the time of the most significant register read.

It may be difficult to get a clear picture of the timeframes in which errors were received because the IEEE 802.3ae counters are independently latched. The PMTICK counters are all latched together, thereby providing a complete snapshot in time. When PMTICK is asserted, the internal error counter values are copied into their associated registers and the internal counters are reset.

There are three methods of asserting PMTICK.

- The station manager may asynchronously assert EWIS_PMTICK_CTRL.PMTICK_FRC to latch the values at a given time, regardless of the EWIS_PMTICK_CTRL.PMTICK_ENA setting.
- The VSC8489-17 device may be configured to latch and clear the statistical counters at a periodic interval as determined by the timer (count) value in EWIS_PMTICK_CTRL.PMTICK_DUR. In this mode, the EWIS_PMTICK_CTRL.PMTICK_SRC must be configured for internal mode and the EWIS_PMTICK_CTRL.PMTICK_ENA bit must be asserted. The receive path clock is used to drive the PMTICK counter, thus the periodicity of the timer can vary during times of loss of lock and loss of frame.
- The VSC8489-17 device may be configured to latch and clear the statistical counters at the occurrence of a rising edge detected at a GPIO pin configured as a PMTICK input pin. In this mode, the EWIS_PMTICK_CTRL.PMTICK_SRC bit must be deasserted, and the EWIS_PMTICK_CTRL.PMTICK_ENA must be asserted. Corresponding GPIO must be configured as the PMTICK input pin.

Regardless of EWIS_PMTICK_CTRL.PMTICK_SRC, when the PMTICK event occurs, the EWIS_INTR_PEND2.PMTICK_PEND is asserted until read. This event may propagate an interrupt to either WIS_INTA or WIS_INTB, based on the mask enable bits EWIS_INTR_MASKA_2.PMTICK_MASKA and EWIS_INTR_MASKB_2.PMTICK_MASKB, respectively.

Given the size of the error counters and the maximum allowable error counts per frame, care must be taken in the frequency of polling the registers to ensure accurate values. All PMTICK counters saturate at their maximum values.

Table 14 • **PMTICK Counters**

Counter Name	Description	Registers	Maximum Increase Count Per Frame	Maximum Increase Count Per Second	Time Until Overflow
B1_ERR_CNT	B1 section error count	EWIS_B1_ERR_CNT1 EWIS_B1_ERR_CNT0	8	64,000	67,109
B2_ERR_CNT	B2 line error count	EWIS_B2_ERR_CNT1 EWIS_B2_ERR_CNT0	1536	12,288,000	350
B3_ERR_CNT	B3 path error count	EWIS_B3_ERR_CNT1 EWIS_B3_ERR_CNT0	8	64,000	67,109
EWIS_REIP_CNT	Far-end B3 path error count	EWIS_REIP_CNT1 EWIS_REIP_CNT0	8	64,000	67,109
EWIS_REIL_CNT	Far-end B2 line error count	EWIS_REIL_CNT1 EWIS_REIL_CNT0	1536	12,288,000	350

Both individual and block mode accumulation of B1, B2, and B3 error indications are supported and selectable using the control bits EWIS_CNT_CFG.B1_BLK_MODE, EWIS_CNT_CFG.B2_BLK_MODE, and EWIS_CNT_CFG.B3_BLK_MODE. In individual accumulation mode 0, the counter is incremented for each bit mismatch between the calculated B1, B2, and/or B3 error and the extracted B1, B2, and/or B3. In block accumulation mode 1, the counter is incremented only once for any nonzero number of bit mismatches between the calculated B1, B2, and/or B3 and the extracted B1, B2, and/or B3 (maximum of one error per frame).

3.3.6 Defects and Anomalies

All defects and anomalies listed in the following table can be forced and masked by the user. The VSC8489-17 device does not automatically generate TIM-P, but does support forcing defects using MDIO.

Table 15 • **Defects and Anomalies**

Defect or Anomaly	Description	Type	Force Bit	Status Bit
Far-end PLM-P or LCD-P	These two errors are indistinguishable when reported by the far-end through the G1 octet (ERDI-P), because the far-end reports both PLM-P and LCD-P with the same error code.	Far-end defect	EWIS_RX_ERR_FRC2.FRC_RX_FE_PLMP	WIS_STAT3.FEPLMP_LCDP
Far-end AIS-P or LOP-P	These two errors are indistinguishable when reported by the far-end through the G1 octet (ERDI-P), because the far-end reports both AIS-P and LOP-P with the same error code.	Far-end defect	EWIS_RX_ERR_FRC2.FRC_RX_FE_AISP	WIS_STAT3.FEAISP_LOPP
PLM-P	Path label mismatch. The detection and reporting of the PLM-P defect follows section 7.5 of ANSI T1.416-1999.	Near-end defect; propagated to PCS	EWIS_RX_ERR_FRC2.FRC_RX_PLMP	WIS_STAT3.PLMP
AIS-L	Generated on LOPC, LOS, LOF, if enabled by EWIS_RXTX_CTRL.RXAISL_ON_LOPC, EWIS_RXTX_CTRL.RXAISL_ON_LOS, EWIS_RXTX_CTRL.RXAISL_ON_LOF, or when forced by user.	Near-end defect	The AIS-L defect is only processed and reported by the WIS Receive process; it is never transmitted by the WIS Transmit process, according to IEEE 802.3ae.	EWIS_RX_ERR_FRC1.FRC_RX_AISL/WIS_STAT3.AISL
AIS-P	Path alarm indication signal.	Near-end defect; propagated to PCS	EWIS_RX_ERR_FRC1.FRC_RX_AISP	WIS_STAT3.AISP
LOP-P	Path loss of pointer. Nine consecutive invalid pointers result in loss of pointer detection. See Figure 15, page 25 for the pointer interpreter state machine.	Near-end defect; propagated to PCS	EWIS_RX_ERR_FRC1.FRC_RX_LOP	WIS_STAT3.LOPP
LCD-P	Path loss of code group delineation. See Table 13, page 29. This is also reported to the far-end if it persists for at least 3 ms.	Near-end defect	EWIS_RX_ERR_FRC2.FRC_LCDP	WIS_STAT3.LCDP
LOPC	Loss of optical carrier alarm. This is an input from the XFP module’s loss of signal output. The polarity can be inverted for use with other module types. This defect can be used independently or in place of LOS.	Near-end defect	EWIS_RX_ERR_FRC1.FRC_LOPC	EWIS_INTR_STAT2.LOPC_STAT

Table 15 • Defects and Anomalies *(continued)*

Defect or Anomaly	Description	Type	Force Bit	Status Bit
LOS	The PMA circuitry detects a loss of signal (LOS) defect if the input signal falls below the assert threshold. When a PMA LOS is declared the framer is held in reset to prevent it from looking for a frame boundary.	Near-end defect	EWIS_RX_ERR_FRC1.FRC_LOS	WIS_STAT3.LOS
SEF	Severely errored frame. Generated when device cannot frame to A1 A2 pattern. SEF indicates synchronization process is not in the SYNC state, as defined by the state diagram of IEEE 802.3ae clause 50.4.2.	Near-end defect; propagated to PCS	EWIS_RX_ERR_FRC2.FRC_RX_SEF	WIS_STAT3.SEF
LOF	Generated when SEF persists for 3 ms. Terminated when no SEF occurs for 1 ms to 3 ms.	Near-end defect	EWIS_RX_ERR_FRC2.FRC_RX_LOF	WIS_STAT3.LOF
B1 PMTICK error count is nonzero	BIP-N(S) - 32-bit near-end section BIP error counter is nonzero.	Near-end anomaly	EWIS_RX_ERR_FRC2.FRC_RX_B1	EWIS_INTR_STAT2.B1_NZ_STAT
B2 PMTICK error count is nonzero	BIP-N(L) - 32-bit near-end line BIP error counter is nonzero.	Near-end anomaly	EWIS_RX_ERR_FRC2.FRC_RX_B2	EWIS_INTR_STAT2.B2_NZ_STAT
B3 PMTICK error count is nonzero	BIP-N(P) - 32-bit near-end path BIP error counter is nonzero.	Near-end anomaly	EWIS_RX_ERR_FRC2.FRC_RX_B3	EWIS_INTR_STAT2.B3_NZ_STAT
REI-L	Line remote error indicator octet is nonzero. Far-end BIP-N(L).	Far-end anomaly	EWIS_RX_ERR_FRC2.FRC_RX_REIL	EWIS_INTR_STAT2.REIL_STAT
REI-L PMTICK error count is nonzero	Line remote error indicator is nonzero. Far-end BIP-N(L).	Far-end anomaly	EWIS_RX_ERR_FRC2.FRC_REIL	EWIS_INTR_STAT2.REIL_NZ_STAT
RDI-L	Line remote defect indicator.	Far-end defect	EWIS_RX_ERR_FRC1.FRC_RX_RDIL	WIS_STAT3.RDIL
REI-P	Path remote error indicator octet is nonzero. Far-end BIP-N(P).	Far-end anomaly	EWIS_RX_ERR_FRC2.FRC_RX_REIP	EWIS_INTR_STAT2.REIP_STAT
REI-P PMTICK error count is nonzero	Path remote error indicator. Far-end BIP-N(P).	Far-end anomaly	EWIS_RX_ERR_FRC2.FRC_REIP	EWIS_INTR_STAT2.REIP_NZ_STAT
UNEQ-P	Unequipped path.	Near-end defect	EWIS_RX_ERR_FRC2.FRC_RX_UNEQP	EWIS_INTR_STAT2.UNEQP_STAT
Far-end UNEQ-P	Far-end unequipped path.	Far-end defect	EWIS_RX_ERR_FRC2.FRC_RX_FE_UNEQP	EWIS_INTR_STAT2.FEUNEQP_STAT

3.3.7 Interrupt Pins and Interrupt Masking

The VSC8489-17 device generates interrupts for each defect and anomaly. The interrupts for the BIP error counts (B1, B2, and B3 counters) and the interrupts for the far-end error counts (REI-L and REI-P) are generated when the PMTICK counters become nonzero. Mask enable bits propagate the interrupt pending event to GPIO pins configured as WIS_INTA and WIS_INTB. Each event may be optionally masked for each WIS_INTA/B pin.

The WIS_INTA and WIS_INTB signals are routed off-chip through GPIO pins. Many specialized functions share the GPIO pins. The WIS_INTA and WIS_INTB signals do not go to dedicated pins.

For each defect or anomaly defined in IEEE 802.3ae, the VSC8489-17 device supports the standard WIS register. In addition, the VSC8489-17 device supports another set of registers in the WIS Vendor Specific area. These registers provide a STATUS bit to indicate the current real-time status of the event, a PENDING bit to indicate if the STATUS bit has changed state, and two mask enable bits for each interrupt pin (WIS_INTA and WIS_INTB). The STATUS bit is set if and only if the interrupt currently exists. This STATUS bit does not latch.

The defects and anomalies are constructed in a hierarchy such that lower order alarms are squelched when higher order events are detected. For more information about the dependencies between squelches and events, see the WIS interrupt registers.

3.3.8 Overhead Serial Interfaces

The VSC8489-17 device includes provisions for off-chip processing of the critical SONET/SDH transport overhead 9-bit words through two independent serial interfaces. The transmit overhead serial interface (TOSI) is used to insert 9-bit words into the transmit frames, and the receive overhead serial interface (ROSI) is used to recover the 9-bit words from the received frames. Each interface consists of three pins: a clock output, a frame pulse output, and a data input (Tx) /output (Rx). These I/O are LVTTL compatible for easy connection to an external device such as an FPGA.

Note: Extended WIS TOSI and ROSI do not support path overhead bytes.

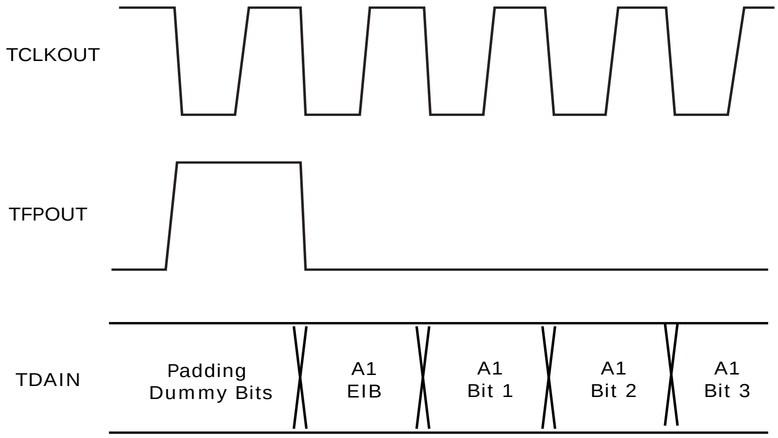
The TOSI / ROSI signals are routed off-chip through GPIO pins. If the ROSI/TOSI interfaces are to be used, there are no GPIO pins left in the design for any other function—loss-of-lock for Sync-E applications, activity LED drivers, WIS_interrupts, or two-wire serial (slave) interface.

All references to TCLKOUT, TFPOUT, TDAIN, RCLKOUT, RFPOUT and RDAOUT are the TOSI/ROSI signals routed through GPIO pins.

3.3.8.1 Transmit Overhead Serial Interface (TOSI)

The TOSI port enables the user to individually program 222 separate 9-bit words in the SONET/SDH overhead. The SONET/SDH frame rate is 8 kHz as signaled by the frame pulse (TFPOUT) signal. The TOSI port is clocked from a divided-down version of the WIS transmit clock made available on TCLKOUT. To provide a more standard clock rate, 9-bit dummy words are added per frame, resulting in a clock running at one five-hundred-twelfth of the line rate (or 19.44 MHz). For each 9-bit word, the external device indicates the desire to transmit that byte by using an enable indicator bit (EIB) that is appended to the beginning of the 9-bit word. If EIB = 0, the data on the serial interface is ignored for that overhead 9-bit word. If EIB = 1, the serial interface data takes precedence over the value generated within the VSC8489-17 device. The EIB is present before the 9-bit dummy words too, however its value has no effect as the 9-bit dummy words are ignored within the device. The first EIB bit should be transmitted by the external device on the first rising edge of TCLKOUT after TFPOUT, as shown in the following illustration. The data should be provided with the most significant bit (MSB) first. After reception of the TOSI data for a complete frame, the values are placed in the overhead for the next transmitted frame.

Figure 16 • **TOSI Timing Diagram**



Some 9-bit words are error masks, such that the transmitted 9-bit word is the XOR of the TOSI 9-bit word and the pre-defined value within the chip if the EIB is enabled. This feature is best used for test purposes only.

The order of the 9-bit word required by the TOSI port is summarized in the following table, where the number of registers is the number of bytes on the serial interface and the number of bytes is the number of STS channels on which the byte is transmitted. For H1 and H2 pointers, bytes 2 to 192 are concatenation indication bytes consistent with STS-192c frames. There are not 192 different point locations as in STS-192 frames.

Table 16 • **TOSI/ROSI Addresses**

Byte Name	9-Bit Word	TOSI/ ROSI Byte Order	Number of Registers	Number of Bytes	Type
Frame boundary	A1	0	1	192	Programmable byte that is identical for all locations
Frame boundary	A2	1	1	192	Programmable byte that is identical for all locations
Section trace	J0	2	1	1	Programmable byte
Section growth	Z0	3	1	191	Programmable byte that is identical for all locations
Dummy byte		4	1	1	Programmable byte
Section BIP-8	B1	5	1	1	TOSI inserts error mask; ROSI extracts XOR of B1 value and received data
Orderwire	E1	6	1	1	Programmable byte
Section user channel	F1	7	1	1	Programmable byte
Dummy byte		8	1	1	Programmable bytes
Section DCC 1	D1	9	1	1	Programmable byte
Section DCC 2	D2	10	1	1	Programmable byte
Section DCC 3	D3	11	1	1	Programmable byte
Dummy byte		12	1	1	Programmable byte
Pointer 1	H1	13	1	1	Programmable byte affecting the first H1 byte

Table 16 • **TOSI/ROSI Addresses *(continued)***

Byte Name	9-Bit Word	TOSI/ ROSI Byte Order	Number of Registers	Number of Bytes	Type
Pointer 2	H2	14	1	1	Programmable byte affecting the first H2 byte
Pointer action	H3	15	1	192	Programmable byte that is identical for all locations
Dummy byte		16	1	1	Programmable byte
Line BIP-8	B2	17 to 208	192	192	TOSI inserts error mask for each byte; ROSI extracts XOR of B2 value and received data for each byte
Automatic protection switching (APS) channel and remote defect indicator (RDI)	K1	209	1	1	Programmable byte
Automatic protection switching (APS) channel and remote defect indicator (RDI)	K2	210	1	1	Programmable byte
Dummy byte		211	1	1	Programmable byte
Line DCC 4	D4	212	1	1	Programmable byte
Line DCC 5	D5	213	1	1	Programmable byte
Line DCC 6	D6	214	1	1	Programmable byte
Dummy byte		215	1	1	Programmable byte
Line DCC 7	D7	216	1	1	Programmable byte
Line DCC 8	D8	217	1	1	Programmable byte
Line DCC 9	D9	218	1	1	Programmable byte
Dummy byte		219	1	1	Programmable byte
Line DCC 10	D10	220	1	1	Programmable byte
Line DCC 11	D11	221	1	1	Programmable byte
Line DCC 12	D12	222	1	1	Programmable byte
Dummy byte		223	1	1	Programmable byte
Synchronization message	S1	224	1	1	Programmable byte
Growth 1	Z1	225	1	191	Programmable byte that is identical for all locations
Growth 2	Z2	226	1	190/191	Programmable byte that is identical for all locations; dependent upon 2xEC40.12
STS-1 REI-L	M0	227	1	1	Programmable byte
STS-N REI-L	M1	228	1	1	Programmable byte
Orderwire 2	E2	229	1	1	Programmable byte
Dummy byte		230	1	1	Programmable byte

Table 16 • **TOSI/ROSI Addresses *(continued)***

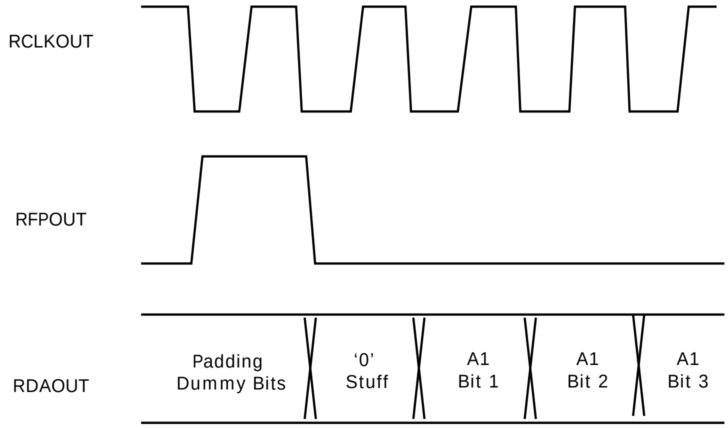
Byte Name	9-Bit Word	TOSI/ ROSI Byte Order	Number of Registers	Number of Bytes	Type
Padding dummy bytes		231 to 269	39		No function

3.3.8.2 Receive Overhead Serial Interface (ROSI)

The ROSI port extracts the same 222 overhead 9-bit words from the SONET/SDH frame, and consists of the clock output (RCLKOUT), frame pulse output (RFPOUT), and data output (RDAOUT). The ROSI port is clocked from a divided-down version of the WIS receive clock, and is valid during in-frame conditions only. As with the TOSI port, 9-bit dummy words are provided each frame period resulting in a 19.44 MHz RCLKOUT frequency. For each 9-bit word, including the 9-bit dummy words, an extra 0 bit is added at the beginning of each byte so that the TOSI and ROSI clock rates are identical. The first stuff bit for each frame is transmitted by RDAOUT on the first rising edge of RCLKOUT after the frame pulse (RFPOUT), as shown in the following illustration.

Because the receive path overhead can be split across two frames, the VSC8489-17 device buffers the overhead for an additional frame time so that a complete path overhead is presented. Table 16, page 35 outlines the order for each of the 9-bit words presented on the ROSI port. With the exception of the M0/M1 9-bit words, the extracted 9-bit words are from the first channel position. In place of parity and error 9-bit words, the VSC8489-17 device outputs the result of an XOR between the calculated BIP and the received value. Therefore, a count of ones within each of the BIP 9-bit words should correspond with the internal error accumulators. The following illustration shows the functional timing for the ROSI port.

Figure 17 • **ROSI Timing Diagram**



3.3.9 Pattern Generator and Checker

The VSC8489-17 device implements the square wave, PRBS31, and mixed-frequency test patterns as described in section 50.3.8 of IEEE 802.3ae as well as the test signal structure (TSS) and continuous identical digits (CID) pattern.

The square wave pattern is selected asserting WIS_CTRL2.TEST_PAT_SEL and the generator is enabled by asserting WIS_CTRL2.TEST_PAT_GEN. When WIS_CTRL2.TEST_PAT_SEL is deasserted, the mixed frequency test pattern is selected. The square wave frequency is configured according to EWIS_TXCTRL2.SQ_WV_PW. The WIS_CTRL2.TEST_PAT_ANA bit is used to enable the test pattern checker in the receive path. The checker does not operate on square wave receive traffic. Error counts from the mixed frequency pattern are presented in the SONET/SDH BIP-8 counters, B1_CNT WIS_B1_CNT, WIS_B2_CNT, and WIS_B3_CNT.

The VSC8489-17 device supports the PRBS31 test pattern as reflected in WIS_STAT2.PRBS31_ABILITY. The transmitter/generator is enabled by asserting

WIS_CTRL2.TEST_PRBS31_GEN, while the receiver/checker is enabled by asserting WIS_CTRL2.TEST_PRBS31_ANA. Because the mixed frequency/square wave test patterns have priority over the PRBS31 pattern, WIS_CTRL2.TEST_PAT_GEN must be disabled for the PRBS31 test pattern to be sent. Error counts from the PRBS31 checker are available in WIS_TSTPAT_CNT. This register does not roll over after reaching its maximum count and is cleared after every read operation. Two status bits are available from the PRBS checker. The EWIS_PRBS31_ANA_STAT.PRBS31_ERR bit indicates whether the error counter is nonzero. The EWIS_PRBS31_ANA_STAT.PRBS31_ANA_STATE bit if asserted indicates that checker is synchronized and actively checking received bits. For test purposes, the PRBS generator can inject single bit errors. By asserting EWIS_PMTICK_CTRL.PMTICK_SRC, a single bit error is injected, resulting in three bit errors being detected within the checker. The value of three comes from the specification, which indicates one error should be detected for each tap within the checker.

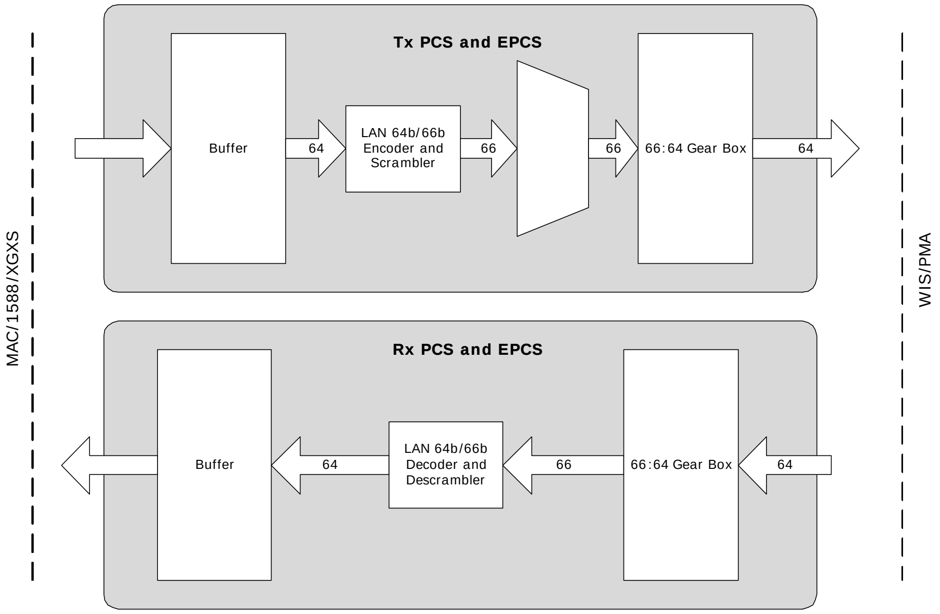
3.4 10G Physical Coding Sublayer (64B/66B PCS)

The 10G physical coding sublayer (PCS) is defined in IEEE 802.3ae Clause 49. It is composed of the PCS transmit, PCS receive, block synchronization, and BER monitor processes. The PCS functions can be further broken down into encode or decode, scramble or descramble, and gearbox functions, as well as various test and loopback modes.

The PCS is responsible for transferring data between the XAUI clock domain and the WIS/PMA clock domain. In addition, the PCS encodes and scrambles the data for efficient transport across the given medium.

The following illustration provides a block diagram of the 10G PCS block.

Figure 18 • **PCS Block Diagram**



3.4.1 Control Codes

The VSC8489-17 device supports the use of all control codes and ordered sets necessary for 10 GbE and 10 GFC operation. The following table lists the control characters, notation, and control codes.

Table 17 • **Control Codes**

Control Codes	Control Character	Notation[1]	XGMII Control Code	10-G BASE-R Control Code	10-G BASE-R O Code
8b/10b Code[2]	Idle	/I/	0x07	0x00	
K28.0 or	K28.3 or	K28.5	Start	/S/	0xfb

Table 17 • **Control Codes *(continued)***

Control Codes *(continued)*	Control Character	Notation[1]	XGMII Control Code	10-G BASE-R Control Code	10-G BASE-R O Code
Encoded by block type field		K27.7	Terminate	/T/	0xfd
Encoded by block type field		K29.7	Error	/E/	0xfe
0x1e		K30.7	Sequence ordered_set	/Q/	0x9c
Encoded by block type field plus O code	0x0	K28.4	Reserved 0	/R/	0x1c
0x2d		K28.0	Reserved 1		0x3c
0x33		K28.1	Reserved 2	/A/	0x7c
0x4b		K28.3	Reserved 3	/K/	0xbc
0x55		K28.5	Reserved 4		0xdc
0x66		K28.6	Reserved 5		0xf7
0x78		K23.7	Signal ordered_set[3]	/Fsig/	0x5c
Encoded by block type field plus O code	0xF	K28.2			

1. The codes for /A/, /K/ and /R/ are used on the XAUI interface to signal idle.
2. For information only. The 8b/10b code is specified in Clause 36. Usage of the 8b/10b code for 10 Gbps operation is specified in Clause 4.
3. Reserved for INCITS T11 - 10 GFC µs.

3.4.2 Transmit Path

In the transmit direction, the PCS accepts data from the XGXS interface, which runs off the XAUI recovered clock and transfers the data onto the PMA transmit clock domain, through the rate-disparity compensating FIFO. Based on the FIFO's fill level, idle characters are added or removed as needed.

Once in the PMA clock domain, the de-serialized XAUI input data (64-bit) is checked for validity. Transmitted data is handled according to IEEE 802.3ae Clause 49.

The characters are then processed in a two-step manner. First, the 64 bits are encoded and a 2-bit header is calculated to form a single 66-bit block. The two header bits are used for block delineation and classification. The only valid header codes are 01 to indicate a payload of all data octets and 10 to indicate the presence of one or more control characters within the payload. To maintain a DC balanced signal on the serial line, the 64-bit encoded payload is scrambled using a self-synchronizing scrambler that implements the polynomial $G(x) = 1 + x^{39} + x^{58}$. The header bits are not scrambled as they are already DC balanced. For debug purposes, the scrambler can be disabled by deasserting SCR_DIS (3x8005.9).

The 66-bit blocks are then passed to the PMA through a 66:64 gearbox. The gearbox merely feeds the 66-bit data into the WIS/PMA's 64-bit data path.

3.4.3 Receive Path

In the receive direction, the PCS accepts data from the WIS/PMA block and reformats it for transmission to the XGXS interface. Because of the data path width mismatches between the WIS/PMA and the PCS, a 64:66 gearbox is needed. The gearbox also performs block synchronization/alignment based upon the 2-bit synchronization header. When the receive logic receives 64 continuous valid sync headers, the BLOCK_LOCK (3x0021.15) bit is asserted. This bit is a latch-low bit; therefore, a second read of the bit returns the current status. If 16 invalid block sync headers are detected within a 125 µs period, the

PCS_HIGHBER (3x0021.14) bit is asserted. This bit is a latch-high bit, and therefore a second read of the bit returns the current status.

Once block synchronization is achieved, the occurrence of errored blocks are accumulated in the PCS_ERRORED_BLOCKS (3x0021.7:0) counter. An errored block is one that has one or more of the following defects:

- The sync field has a value of 00 or 11.
- The block type field contains a reserved value (for more information, see Table 17, page 38).
- Any control character contains an incorrect value.
- Any O code contains an incorrect value.
- The set of eight XGMII characters does not have a corresponding block format shown in the following illustration.

Figure 19 • **64B/66B Block Formats**

Input Data	**Sync**	**Block Payload**								
Bit Position : Data Block Format :	0 1	2								65
D0 D1 D2 D3/D4 D5 D6 D7	01	D_0	D_1	D_2	D_3	D_4	D_5	D_6	D_7	
Control Block Formats:		**Block Type Field**								
C0 C1 C2 C3/C4 C5 C6 C7	10	0x1e	C_0	C_1	C_2	C_3	C_4	C_5	C_6	C_7
C0 C1 C2 C3/O4 D5 D6 D7	10	0x2d	C_0	C_1	C_2	C_3	O_4	D_5	D_6	D_7
C0 C1 C2 C3/S4 D5 D6 D7	10	0x33	C_0	C_1	C_2	C_3		D_5	D_6	D_7
O0 D1 D2 D3/D4 D5 D6 D7	10	0x66	D_1	D_2	D_3	O_0		D_5	D_6	D_7
O0 D1 D2 D3/O4 D5 D6 D7	10	0x55	D_1	D_2	D_3	O_0	O_4	D_5	D_6	D_7
S0 D1 D2 D3/D4 D5 D6 D7	10	0x78	D_1	D_2	D_3	D_4	D_5	D_6	D_7	
O0 D1 D2 D3/C4 C5 C6 C7	10	0x4b	D_1	D_2	D_3	O_0	C_4	C_5	C_6	C_7
T0 C1 C2 C3/C4 C5 C6 C7	10	0x87		C_1	C_2	C_3	C_4	C_5	C_6	C_7
D0 T1 C2 C3/C4 C5 C6 C7	10	0x99	D_0		C_2	C_3	C_4	C_5	C_6	C_7
D0 D1 T2 C3/C4 C5 C6 C7	10	0xaa	D_0	D_1		C_3	C_4	C_5	C_6	C_7
D0 D1 D2 T3/C4 C5 C6 C7	10	0xb4	D_0	D_1	D_2		C_4	C_5	C_6	C_7
D0 D1 D2 D3/T4 C5 C6 C7	10	0xcc	D_0	D_1	D_2	D_3		C_5	C_6	C_7
D0 D1 D2 D3/D4 T5 C6 C7	10	0xd2	D_0	D_1	D_2	D_3	D_4		C_6	C_7
D0 D1 D2 D3/D4 D5 T6 C7	10	0xe1	D_0	D_1	D_2	D_3	D_4	D_5		C_7
D0 D1 D2 D3/D4 D5 D6 T7	10	0xff	D_0	D_1	D_2	D_3	D_4	D_5	D_6	

Valid blocks recover their original payload data by being descrambled. The descrambler is the same polynomial used by the transmitter. For test purposes, the descrambler may be disabled by asserting DSCR_DIS (3x8005.10). The data is checked for valid characters and sequencing.

The data is passed from the PMA/WIS clock domain to the XAUI clock domain through a FIFO. Based upon the FIFO's fill level, idle characters are added or removed as needed.

3.4.4 PCS Standard Test Modes

The PCS block offers all of the standard defined test pattern generators and analyzers. In addition, the VSC8489-17 device supports a 64-bit static user pattern and the optional PRBS31 pattern. Two error counters are available. Each is a saturating counter that is cleared upon a read operation. The first, PCS_ERR_CNT, is located in the IEEE Standard area while the 32-bit PCS_VSERR_CNT_0/PCS_VSERR_CNT_1 is located in the vendor specific area.

The IEEE specification defines two test pattern modes, a square wave generator and a pseudo-random test pattern. The square wave generator is enabled by first selecting the square wave pattern by asserting PCS_TSTPAT_SEL, and then enabling the test pattern generator, PCS_TSTPAT_GEN. The period of the square wave can be controlled in terms of bit times by writing to PCS_SQPW. There is no associated square wave checker within the VSC8489-17 device.

The pseudo-random test pattern is selected by deasserting PCS_TSTPAT_SEL. The pseudo-random test pattern contains two data modes. When PCS_TSTDAT_SEL is deasserted, the pseudo-random pattern is a revolving series of four blocks with each block 128-bits in length. The four blocks are the resultant bit sequence produced by the PCS scrambler when pre-loaded with the following seeds:

- PCS_SEEDA_0, PCS_SEEDA_1, PCS_SEEDA_2, PCS_SEEDA_3
- PCS_SEEDA invert
- PCS_SEEDB_0, PCS_SEEDB_1, PCS_SEEDB_2, PCS_SEEDB_3
- PCS_SEEDB invert

The pattern generator is enabled by asserting PCS_TSTPAT_GEN while the analyzer is enabled by asserting PCS_TSTPAT_ANA. Errors are accumulated in the clear-on-read saturating counter, PCS_ERR_CNT. In pseudo-random pattern mode, the error counter counts the number of errored blocks.

Support for the optional PRBS31 pattern is indicated by PRBS31_pattern_testing_ability register whose default is high. The PRBS31 test generator is selected by asserting PCS_PRBS31_GEN, while the checker is enabled by asserting PCS_PRBS31_ENA. IEEE standards specify that the error counter should increment for each linear feedback shift register (LFSR) tap that a bit is in error. Therefore, a single bit error increments the counter by three because there are three taps in the PRBS31 polynomial.

The user-defined 64-bit static pattern can be written to PCS_USRPAT registers and enabled by asserting PCS_USRPAT_ENA and PCS_TSTPAT_GEN. Enabling the user-defined pattern enables both the generator and analyzer.

3.5 1G Physical Coding Sublayer

The 1G physical coding sublayer (PCS) implements 1000BASE-X as specified by IEEE 802.3 Clause 36, and auto-negotiation as specified by IEEE 802.3 Clause 37. It provides for the encoding (and decoding) of GMII data octets to (and from) ten-bit code-groups (8B/10B) for communication with the underlying PMA. It also manages link control and the auto-negotiation process.

In addition to these standard 1000BASE-X functions, the 1G PCS also includes a conversion function that maps the standard GMII data to (and from) an internal XGMII-like interface. This allows the processing core to be largely agnostic to whether a channel is operating in 1G or 10G operation.

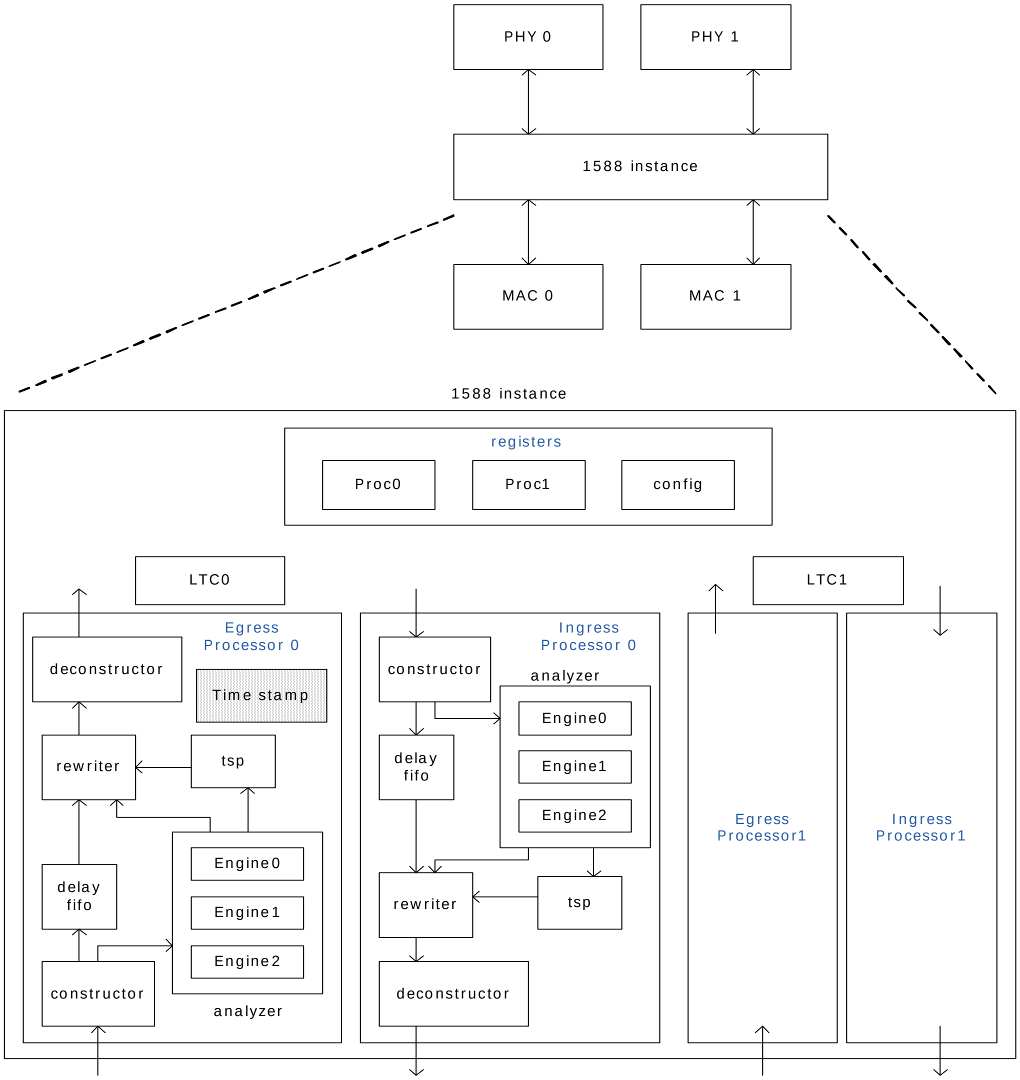
3.6 IEEE 1588 Block Operation

The VSC8489-17 device uses a second generation IEEE 1588 engine that is backward compatible with the earlier version of VeriTime™ (the Microsemi IEEE 1588 time stamping engine). It is also compatible with the IEEE 1588 operations supported in Microsemi CE switches. The following list shows the new features of the Microsemi second generation IEEE 1588.

- Higher time stamp accuracy and resolution
- Automatic clear enables after system time is read or written
- Ability to load or extract the current system time in serial format
- Full 48-bit math support for incoming correction field
- Ability to add or subtract fixed offset from system time to synchronize between slaves
- Independent control and bypass for each direction of IEEE 1588
- Support to extract frame signature in an IPv6 frame
- MPLS-TP OAM support in third analyzer engine
- Special mode where all frames traversing the system can be time stamped

The unique architecture of the second generation IEEE 1588 block provides for the lowest latency and maximum throughput on the channel. The following illustration shows a block diagram of the IEEE 1588 architecture in the VSC8489-17 device.

Figure 20 • IEEE 1588 Architecture



The following sections list some of the major IEEE 1588 applications.

3.6.1 IEEE 1588 Block

The IEEE 1588 engine may be configured to support one-step and two-step clocks as well as Ethernet and MPLS OAM delay measurement. It detects the IEEE 1588 frames in both the Rx and Tx paths, creates a time stamp, processes the frame, and updates them. It can add a 30/32-bit Rx time stamp to the 4-bytes reserved field of the PTP packet. It can also modify the IEEE 1588 correction field and update the CRC of changed frames. There are local time counters (reference for all time stamps) that can be preloaded and adjusted though the register interface.

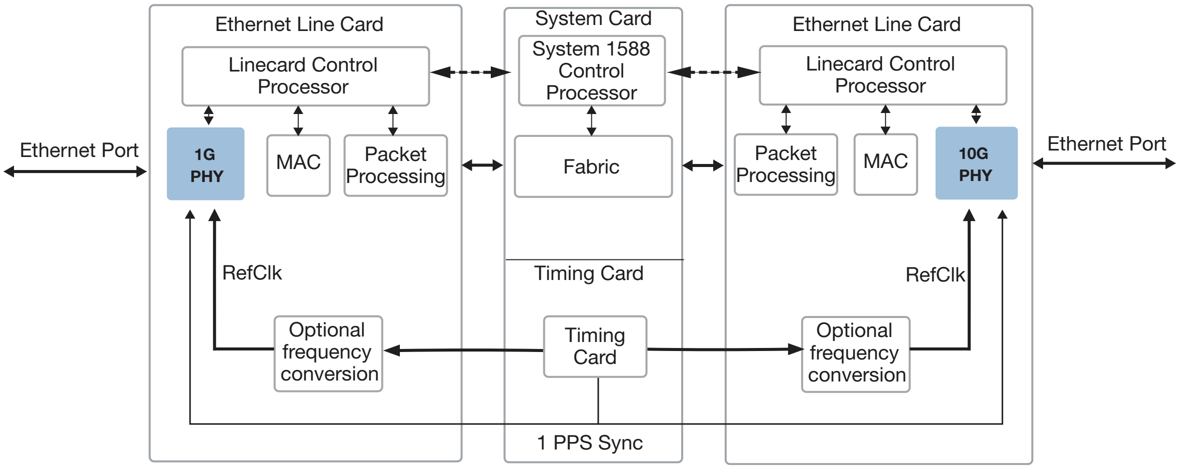
A local time counter is used to hold the local time for Rx and Tx paths. A small FIFO delays frames to allow time for processing and modification. An analyzer detects the time stamp frames (PTP and OAM) and a time stamp block calculates the new correction field. The rewriter block replaces the correction field with an updated one and checks/calculates the CRC. For the Tx path, a time stamp FIFO saves Tx event time stamp plus frame identifier for use in some modes.

The IEEE 1588 engine's registers and time stamps are accessible through the MDIO or 4-pin SPI. To overcome the MDIO or 4-pin SPI speed limitations, the dedicated "push-out" style SPI output bus can be used for faster or large amounts of time stamp reads. This SPI output is used to push out time stamp information to an external device only and does not provide read/write to the registers of the IEEE 1588 engine or registers of other blocks in the VSC8489-17 device. In addition, there is a LOAD/SAVE pin that is used to load the time in the PHYs and to ensure that all the PHYs are in sync. The local time counter may come from any one of the following sources:

- Data path clock (varies according to mode)
- 250 MHz from host-side PLL
- External clock (125 MHz or 250 MHz) from CLK1588P/N pins

The local time counters contain two counters: nanosecond_counter and second_counter. The 1 PPS (pulse per second signal) output pin can be used for skew monitoring and adjustment. The following illustration shows an overview of a typical system using IEEE 1588 PHYs. The LOAD/SAVE and 1 PPS pins are signals routed to the GPIO pins. The following illustration shows how the PHY is embedded in a system.

Figure 21 • **IEEE 1588 Block Diagram**



The system card has to drive the REFCLK (125 MHz or 250 MHz timetick clock) to all the PHYs, including the VSC8489-17 device. The system clock may need local frequency conversion to match the required reference clock frequency. The system clock may be locked to a PRC by SyncE or by IEEE 1588. If locked by IEEE 1588, the central CPU recovers the PTP timing and adjusts the frequency of the system clock to match the PTP frequency. If the system clock is free running, the central CPU must calculate the frequency offset between the system clock and the synchronized IEEE 1588 clock, and program the PHYs to make internal adjustments.

The system card also provides a sync pulse to all PHYs, including the VSC8489-17 device, to the LOAD/SAVE pin. This signal is used to load the time to the PHYs and to ensure that all the PHYs are in sync. This may just be a centrally divided down system clock that gives a pulse at fixed time intervals. The delay from the source of the signal to each PHY must be known and taken into account when writing in the load time in the PHYs.

The VSC8489-17 device supports a vast variety of IEEE 1588 applications. In simple one-step end-to-end transparent clock applications, the VSC8489-17 device can be used without any central CPU involvement (except for initial configuration). The IEEE 1588 block inside the VSC8489-17 device forwards Sync and Delay_req frames with automatic updates to the Correction field.

In other applications, the VSC8489-17 device enhances the performance by working with a central processor that runs the IEEE 1588 protocol. The VSC8489-17 device performs the accurate time stamp operations needed for all the different PTP operation modes. For example, at startup in a boundary clock application, the central CPU receives PTP sync frames that are time stamped by the ingress PHY and recovers the local time offset from the PTP master using the PTP protocol. It then sets the save bit in the VSC8489-17 device connected to the PTP master and later reads the saved time. The central CPU loads the expected time (time of the next LOAD/SAVE pulse, corrected by the offset to the recovered PTP time) into the PHY and sets the save bit. It checks that the time offset is 0. If not, it makes small adjustments to the time in the PHY by issuing add 1 ns or subtract 1 ns commands to the VSC8489-17 device through MDIO until the time matches the PTP master. A save command is issued to the PHY connected to the PTP master and reads the saved time. The central CPU then writes the saved time plus the sync pulse interval plus any sync pulse latency variation (trace length difference compared to the

PHY connected to the PTP master) to the other PHYs and sets the load bit in these VSC8489-17 devices.

The preceding sequence may be completed in several steps. Not all PHYs need to be loaded at once. The central CPU sets the save bit in all PHYs and reads back the values. They should all save the same value.

The central CPU continuously detects if the system time drifts off compared to the recovered PTP time. If needed, it can adjust each PHY for any known skew between PHYs without affecting the operation of the device. It can program the PHYs, including the VSC8489-17 device, to automatically add 1 ns or subtract 1 ns at specific time intervals.

3.6.2 IEEE 1588v2 One-Step End-to-End Transparent Clock

The timestamp block is located in PHYs and MACs with integrated PHYs that are placed on line cards. If Microsemi 1588 PHYs are used on all ports that support IEEE 1588 one-step end-to-end transparent clocks, the rest of the system does not need to be 1588-aware, and there is no CPU maintenance needed once the system is set up.

As all the PHYs in a system can be configured the same way, the system supports failover of 1588 masters without any CPU intervention.

This solution works for both blade systems and pizza boxes, where the devices placed on the system side of the PHYs don’t need to be 1588-aware. This allows an easy migration path for systems that do not support IEEE 1588, as this feature can be added by replacing existing PHYs with Microsemi 1588 PHYs on all ports.

Unique advantages for implementing IEEE 1588-2008 include:

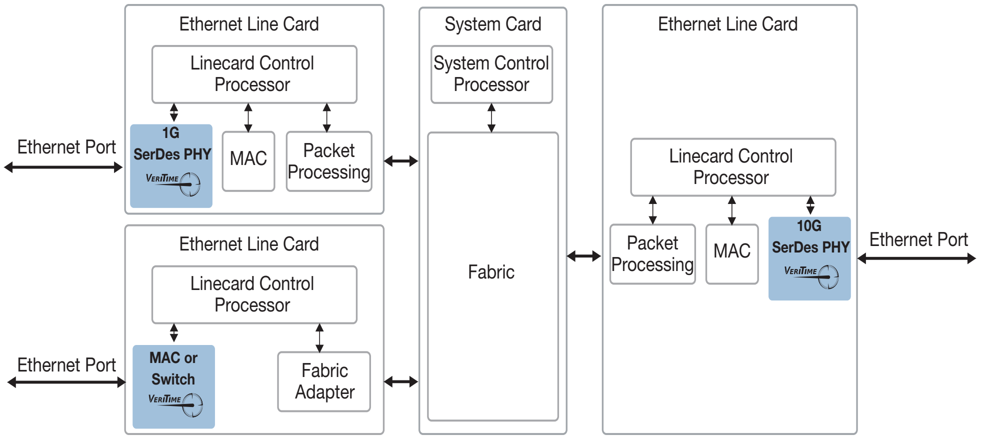
- When several VSC8489-17 devices or Microsemi PHYs with integrated IEEE 1588 time stamping blocks are used on all ports within the system that support IEEE 1588 one-step E2E TC, the rest of the system does not need to be IEEE 1588 aware, and there is no CPU maintenance needed once the system is set up.
- As all the PHYs in a system can be configured the same way, it supports fail-over of IEEE 1588 masters without any CPU intervention.
- VSC8489-17 and other Microsemi PHYs with integrated IEEE 1588 time stamping blocks also work for pizza box solutions, where the switch/router can be upgraded to support IEEE 1588 E2E TC.

Requirements for the rest of the system are:

- Delivery of a synchronous global timetick clock (or reference clock) to ensure that the “local time” for all PHYs in the system progresses at the same rate.
- Delivery of a global timetick load to synchronize the local time counters in each PHY.
- CPU access to each PHY to set up the required configuration. This can be through MDIO, two-wire slave, or 4-pin SPI.

The following illustration shows a diagram for the transparent clock line card application.

Figure 22 • **1588 Transparent Clock Line Card End-to-End PHY Application**

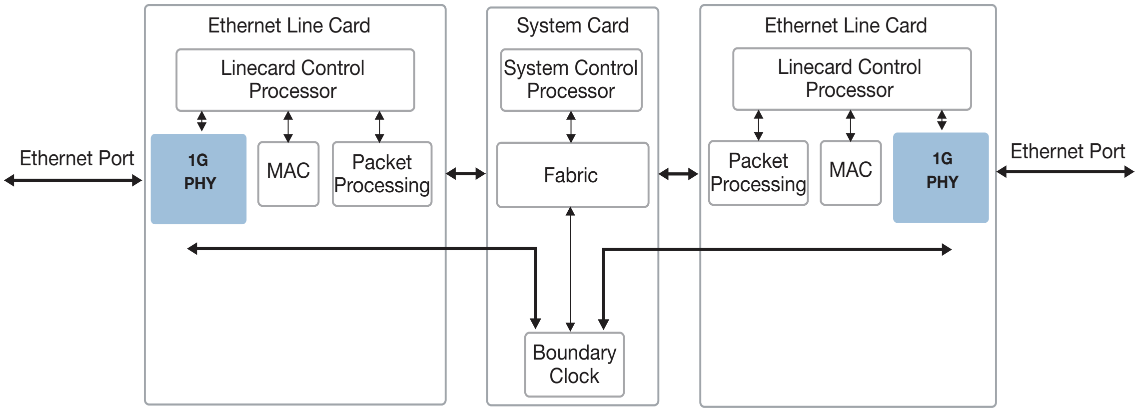


3.6.3 IEEE 1588v2 Transparent Clock and Boundary Clock

This is the same system as described previously, with the addition of a central IEEE 1588 engine (Boundary Clock). The IEEE 1588 engine is most likely a CPU system, possibly together with hardware support functions to generate Sync frames (for BC and ordinary clock masters). The switch/fabric needs to have the ability to redirect (and copy) PTP frames to the IEEE 1588 engine for processing.

This system uses a central 1588 engine, most likely a CPU system, together with hardware support functions to generate sync frames (for boundary clock and ordinary clock masters). The switch fabric needs to have the ability to redirect (and copy) PTP frames to the 1588 engine for processing. This system also works for pizza boxes.

Figure 23 • **Transparent Clock and Boundary Clock Line Card Application**



This solution also works for pizza boxes. To ensure that blade redundancy works, the PHYs for the redundant blades must have the same 1588-in-the-PHY configuration.

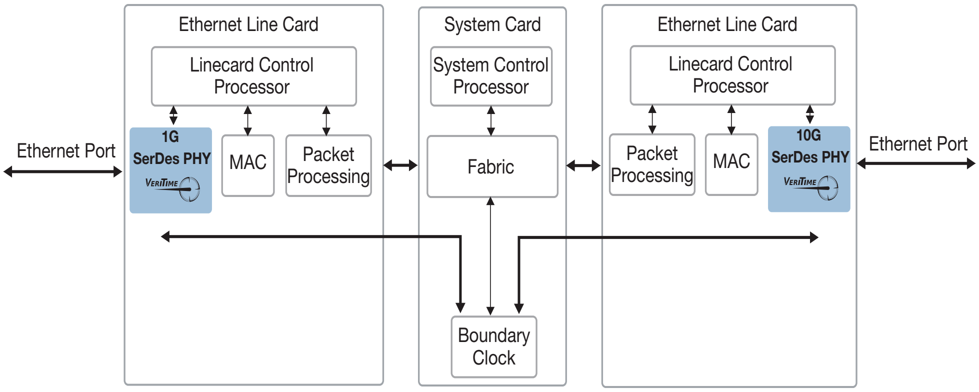
Requirements for the rest of the system are:

- Delivery of a synchronous global timetick clock (or reference clock) to the PHYs.
- Delivery of a global timetick load, that synchronizes the local time counters in each port.
- CPU access to each PHY to set up the required configuration. For one-step support, this can be MDC/MDIO. For two-step support, a higher speed CPU interface (such as the SPI) might be required (depending on the number of time stamps that are required to be read by the CPU). In blade systems, it might be required to have a local CPU on the blade that collects the information and sends it to the central IEEE 1588 engine by means of the control plane or the data plane. In advanced MAC/Switch devices, this might be an internal CPU.
- Fabric must be able to detect IEEE 1588 frames and redirect them to the central IEEE 1588 engine.

The same solution can also be used to add Y.1731 delay measurement support. This does not require a local CPU on the blade, but the fabric must be able to redirect OAM frames to a local/central OAM processor.

The following illustration shows a diagram for the boundary clock line card application.

Figure 24 • **1588 Boundary Clock Line Card Application**



3.6.4 Enhancing IEEE 1588 Accuracy for CE Switches and MACs

Connecting VSC8489-17 or other Microsemi PHYs that have integrated IEEE 1588 time stamping in front of the CE Switches and MACs improves the accuracy of the IEEE 1588 time stamp calculation. This is due to the clock boundary for the XAUI and SGMII/QSGMII interface. It will also add support for one-step TC and BC on the Jaguar-1 family of devices.

3.6.5 Supporting One-Step Boundary Clock/Ordinary Clock

In one-step boundary clock, the BC device acts as an ordinary clock slave on one port and as master on the other ports. On the master ports, Sync frames are transmitted from the IEEE 1588 engine that holds the Origin time stamp. These frames will have the correction field or the full Tx time stamp updated on the way out though the PHY.

Master ports also receive Delay_req from the slaves and respond with Delay_resp messages. The Delay_req messages are time stamped on ingress through the PHY and the IEEE 1588 engine receives the Delay_req frame and generates a Delay_resp message. The Delay_resp messages are not event messages, and are passed though the PHY as any other frame.

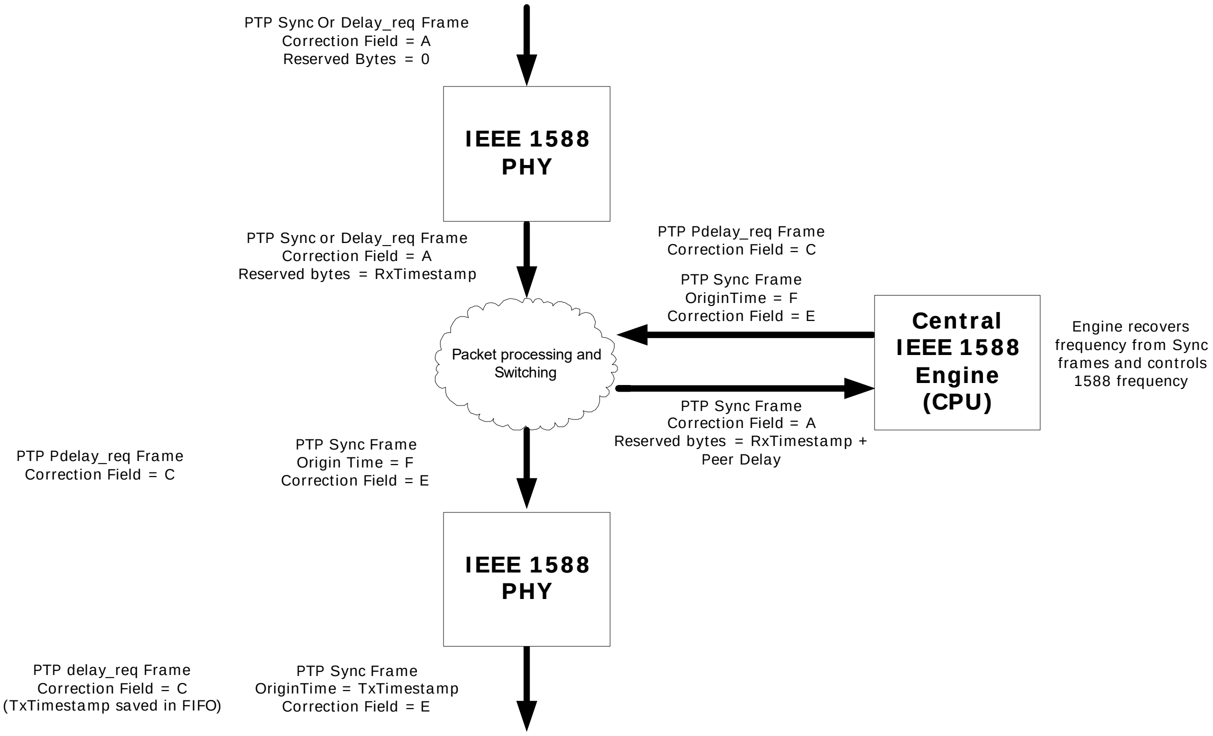
The port configured as slave receives Sync frames from its master. The Sync frames have a Rx time stamp added in the PHY and forwarded to the IEEE 1588 engine.

The IEEE 1588 engine also generates Delay_req frames that are sent on the port configured as slave port. Normally the transmit time for the Delay_req frames (t3) is saved in a time stamp FIFO in the PHYs, but when using Microsemi IEEE 1588 PHYs, a slight modification can be made to the algorithm to remove the CPU processing overhead of reading the t3 time stamp.

To modify the algorithm, the IEEE 1588 engine should send the Delay_req message with a software generated t3 value in the origin time stamp, the sub-second value of the t3 time stamp in the reserved bytes of the PTP header, and a correction field of 0. The software generated t3 time stamp should be within a second before the actual t3 time. The Egress PHY should then be configured to perform E2E TC egress operation, meaning calculate the “residence time” from the inserted t3 time stamp to the actual t3 time and insert this value in the correction field of the frame. When the local IEEE 1588 engine receives the corresponding Delay_resp frame back, it can use the software generated t3 value because the correction field of the Delay_resp frame contains a value that compensates for the actual t3 transmission time.

Boundary clocks and ordinary clocks must also reply to Pdelay_req messages just as P2P TC using the same procedure for the P2P TC. For more information, see Supporting One-Step Peer-to-Peer Transparent Clock, page 53.

Figure 25 • **One-Step E-nd-to-End Boundary Clock**



3.6.5.1 Ingress

Each time the PCS/PMA detects the start of a frame, it sends a pulse to the time stamp block, which saves the value of the Local_Time received from the Local Time counter. In the time stamp block, the programmed value in the local_correction register is subtracted from the saved time stamp. The local_correction register is programmed with the fixed latency from the measurement point to the place that the start of frame is detected in the PCS/PMA logic. The time stamp block also contains a register that can be programmed with the known link asymmetry. This value is added or subtracted from the correction field, depending on the frame type.

When the frame leaves the PCS/PMA block, it is loaded into a small FIFO block that delays and stores the frame data for a few clock cycles to allow for later modifications of the frame. The data is also copied to the analyzer block that parses the incoming frame to detect whether it is a IEEE 1588 Sync or Delay_req frame belonging to the PTP domain that the system is operating on. If so, it signals to the ingress time stamp block in the PHY which action to perform (Write). It also delivers the write offset and data size (location of the four reserved bytes in the PTP header, 4 bytes wide) to the rewriter block in the PHY.

If the analyzer detects that the frame is not matched, it signals to the time stamp block and the rewriter block to ignore the frame (NOP), which allows it to pass unmodified and flushes the saved time stamp in the time stamp block.

If the time stamp block gets the Write action, it delivers the value of the calculated time stamp for the frame to the rewriter block and the rewriter block adds this time stamp (ns part of it) to the four reserved bytes in the frame and recalculates FCS.

The rewriter block takes data out of the FIFO block continuously and feeds it to the system side PCS/PMA block using a counter to keep track of the byte positions of the frame. When the rewriter block receives a signal from the time stamp block to rewrite a specific position in the frame (that information comes from the analyzer block), it overwrites the position with the data from the time stamp block and replaces the FCS of the frame. The rewriter also checks the original FCS of the frame to ensure that a frame that is received with a bad FCS and then modified by the rewriter is also sent out with a bad FCS. This is achieved by inverting the new FCS. If the frame is an IPv4 frame the rewriter ensures that the IP checksum is 0. If the frame is IPv6 the rewriter keeps track of the modifications done to the frame and modifies a couple of bytes placed at the end of the PTP frame (for this specific purpose) so that the IP checksum stays correct.

The following full calculations are performed:

- Sync frames: Reserved_bytes = (Raw_Timestamp_ns – Local_correction) Correction field = Original Correction field + Asymmetry
- Delay_req frames: Reserved_bytes = (Raw_Timestamp_ns – Local_correction)

3.6.5.2 Egress

When a frame is received from the system side PCS/PMA block, it is loaded into a FIFO block that delays and stores the frame data for a few clock cycles to allow for later modifications of the frame. The data is also copied to the analyzer block that parses the incoming frame to detect whether it is a IEEE 1588 Sync or Delay_req frame belonging to the PTP domain that the system is operating on.

If the egress analyzer of the PHY detects that the frame is a IEEE 1588 Sync frame belonging to the PTP domain(s) of the system, it signals to the egress time stamp block which action to perform (Write). It also delivers the write offset and data size (location of the Tx time stamp inside the frame, 10 bytes wide) to the rewriter.

If the egress analyzer detects that the frame is a IEEE 1588 Delay_req frame belonging to the PTP domain(s) of the system, it signals to the time stamp block which action to perform (Write, Save). It also delivers the write offset and data size (location of the Tx time stamp inside the frame, 10 bytes wide) to the rewriter. It also outputs up to 16 bytes of frame identifier to the Tx time stamp FIFO, to be saved along with the Tx time stamp. The frame identifier bytes are selected information from the frame, configured in the analyzer.

If the time stamp block gets the (Write, Save) action it delivers the calculated time stamp and signals to the time stamp FIFO block that it must save the time stamp along with the frame identifier data it received from the analyzer block.

The Tx time stamp FIFO block contains a buffer memory. It simply stores the Tx time stamp values that it receives from the time stamp block together with the frame identifier data it receives from the analyzerblock, and has a CPU interface that allows the IEEE 1588 engine to read out the time stamp sets (Frame identifier + New Tx time stamp).

The following full calculations are performed:

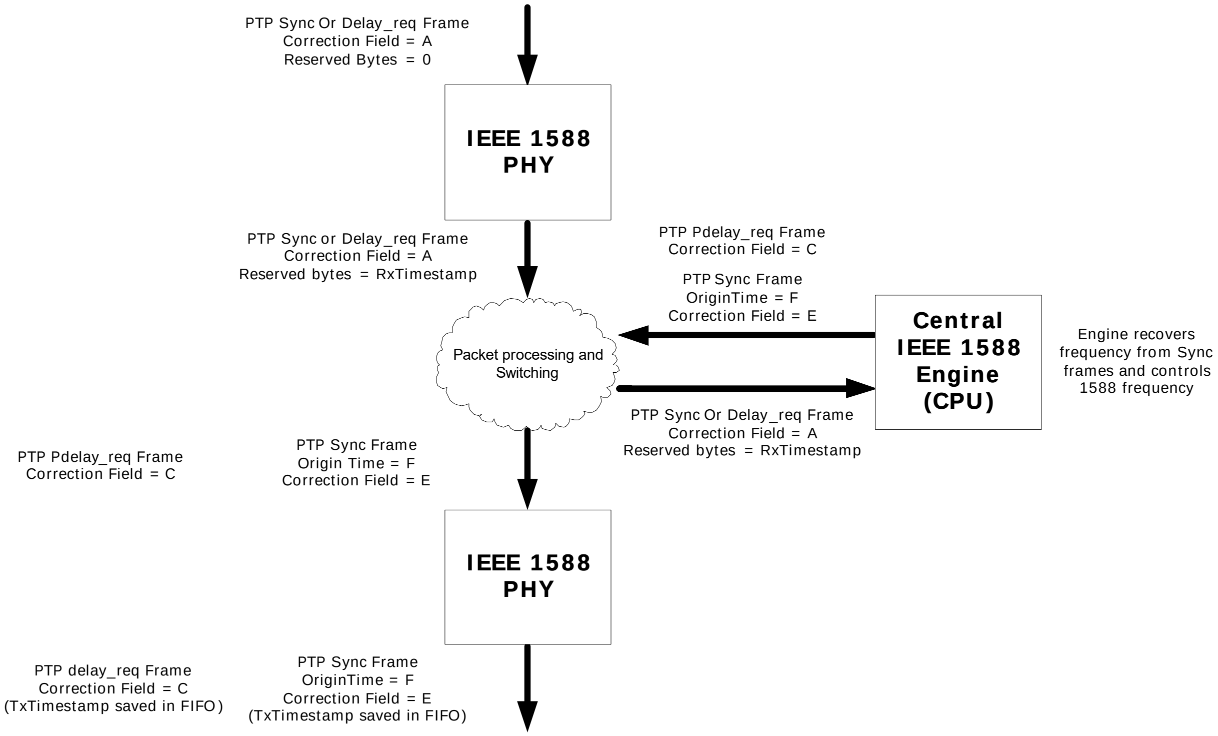
- Sync frames: OriginTimestamp = (Raw_Timestamp + Local_correction)
- Delay_req frames: OriginTimestamp = (Raw_Timestamp + Local_correction) Correction field = Original Correction field + Asymmetry

3.6.6 Supporting Two-Step Boundary Clock/Ordinary Clock

Two-step clocks are used in systems that cannot update the correction field on-the-fly, and this requires more CPU processing than one-step.

Each time a Tx time stamp is sent in a frame, the IEEE 1588 engine reads the actual Tx transmission time from the time stamp FIFO and issues a follow-up message containing this time stamp. Even though the VSC8489-17 device supports one-step operation, thereby eliminating the need to run in two-step mode, support for this mode is provided for networks that include two-step-only implementations.

Figure 26 • Two-Step End-to-End Boundary Clock



3.6.6.1 Ingress

If the ingress analyzer in the PHY detects that the frame is a IEEE 1588 Sync or Delay_req frame belonging to the PTP domain(s) of the system, it signals to the time stamp block which action to perform (Write). It also delivers the write offset and data size (location of the four reserved bytes in the PTP header, 4 bytes wide) to the rewriter.

If the time stamp block gets the Write action, it delivers the calculated time stamp to the rewriter block and the rewriter block adds this time stamp (ns part of it) to the four reserved bytes in the frame and recalculates FCS.

Note: When secure timing delivery is required (when using IPsec authentication, for instance), the four reserved bytes must be reverted back to 0 before performing integrity check.

The following full calculations are performed:

- Sync frames: Reserved_bytes = (Raw_Timestamp – Local_correction)
  Correction field = Original Correction field + Asymmetry
- Delay_req frames: Reserved_bytes = (Raw_Timestamp – Local_correction)

3.6.6.2 Egress

If the egress analyzer detects that the frame is a IEEE 1588 Sync or Delay_req frame belonging to the PTP domain(s) of the system, it signals to the time stamp block which action to perform (Write, Save). The analyzer outputs up to 15 bytes of frame identifier to the Tx time stamp FIFO to be saved along with the Tx time stamp. The frame identifier must include, at a minimum, the sequenceId field so the CPU can match the time stamp with the follow-up frame.

If the time stamp block gets the Write, Save action it delivers the calculated time stamp to the time stamp FIFO and signals to the time stamp FIFO block that it must save the time stamp along with the frame identified data it received from the analyzer block.

The following full calculations are performed:

- Sync frames: FIFO = (Raw_Timestamp + Local_correction)

- Delay_req frames: FIFO = (Raw_Timestamp + Local_correction)
  Correction field = Original Correction field – Asymmetry

3.6.7 Supporting One-Step End-to-End Transparent Clock

End-to-end transparent clocks add the residence time (the time it takes to traverse the system from the input to the output port(s)) to all Sync and Delay_req frames. It does not need to have any knowledge of the actual time, but if it is not locked to the frequency of the IEEE 1588 time, it will produce an error that is the ppm difference in frequency times the residence time.

When the TC is frequency-locked by means of IEEE 1588 or other methods (SyncE), the error is only caused by sampling inaccuracies.

The VSC8489-17 device supports a number of different transparent clock modes that can be divided into two main modes, as follows.

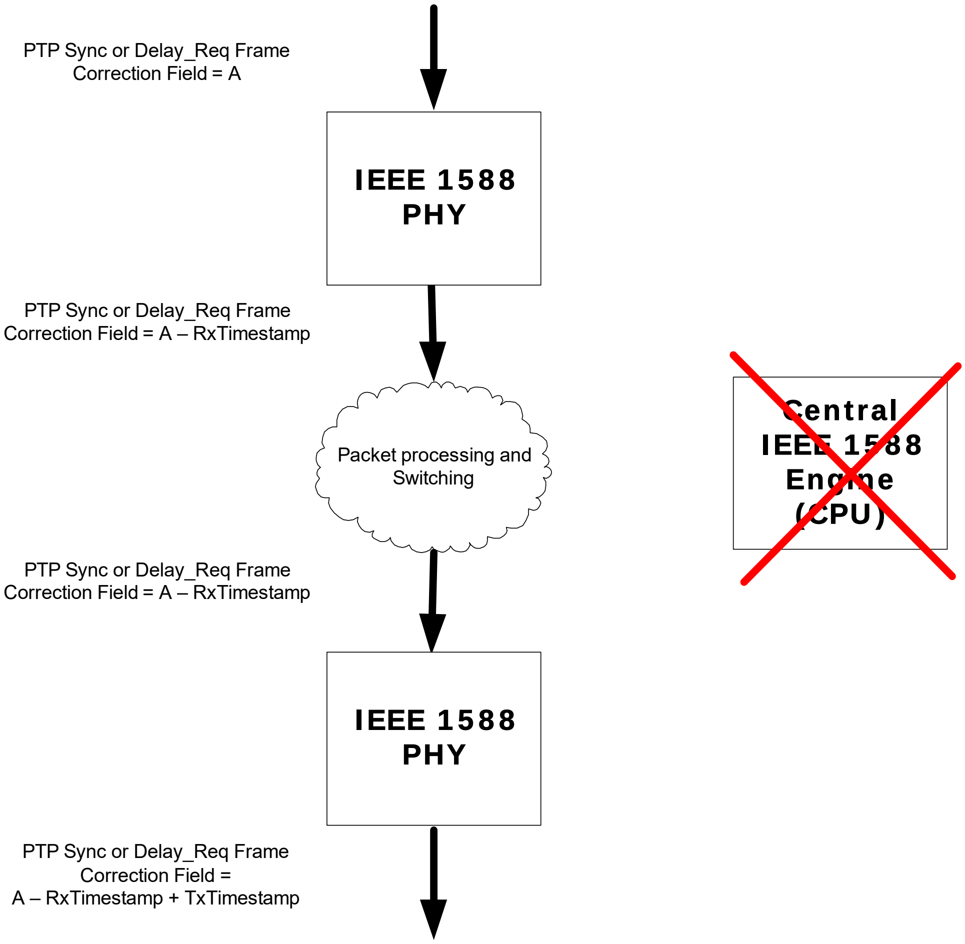
- **Mode A.** Subtracts the ingress time stamp at ingress and adds the egress time stamp at egress. This mode can run in a number of sub-modes, depending on the format of the time stamp that is subtracted or added.
- **Mode B.** Saves the ingress time stamp in the reserved bytes of the PTP header (just as is done in BC and ordinary clock modes) and performs the residence time calculation at the egress PHY where the calculated residence time is added to the correction field of the PTP frame.

Mode B is recommended because it has a number of advantages, including the option to support TC and BC operation in the same system and on the same traffic, and the ease of implementing syntonized TC operation.

When an E2E TC recovers frequency using IEEE 1588 and is using Mode A, it must either have a PHY with IEEE 1588 time stamping Mode A support or another way of adding the local time to the correction field placed in front of the IEEE 1588 engine. The IEEE 1588 engine is then able to receive Sync frames and adjust the local frequency to match the IEEE 1588 time.

If using Mode B, the IEEE 1588 engine can recover the frequency directly from the Sync frames because it can extract the ingress time stamp directly from the frames. The frequency adjustment can be done by adjusting the time counter in each PHY or by adjusting the global Timetick clock.

Figure 27 • **One-Step End-to-End Transparent Clock Mode A**



When the system works in one-step E2E TC mode Sync and Delay_req frames must be forwarded through the system and the residence time = (Egress time stamp – Ingress time stamp) must be added to the correction field in the frame before it leaves the system.

The following sections describe the operation in Modes A and B.

3.6.7.1 Ingress (Mode A)

If the analyzer detects that the frame is a IEEE 1588 Sync or Delay_req frame belonging to the PTP domain(s) of the system, it signals to the time stamp block which action to perform (Subtract), along with the correction field of the frame. It also delivers the write offset and data size (location of the correction field inside the frame, 8 bytes wide) to the rewriter.

If the time stamp block gets the Subtract action, it subtracts the time stamp converted to ns from the original correction field of the frame and outputs the value to the rewriter block.

As a result the frame is sent towards the system with a correction field containing the value: Original Correction field – Rx time stamp (converted to ns).

The following full calculations are performed:

- Sync frames: Internal Correction field = Original Correction field – (Raw_Timestamp_ns – Local_correction) + Asymmetry
- Delay_req frames: Internal Correction field = Original Correction field – (Raw_Timestamp_ns – Local_correction)

3.6.7.2 Egress (Mode A)

The egress side works that same way as ingress, but the analyzer is set up to add the active_timestamp to the correction field.

If the analyzer detects that the frame is a IEEE 1588 Sync or Delay_req frame belonging to the PTP domain(s) of the system, it signals to the time stamp block which action to perform (Add), along with the correction field of the frame. It also delivers the write offset and data size (location of the correction field inside the frame, 8 bytes wide) to the rewriter.

If the analyzer detects that the frame is not matched, it signals the time stamp block and the rewriter block to ignore the frame (let it pass unmodified and flush the saved time stamp in the time stamp block).

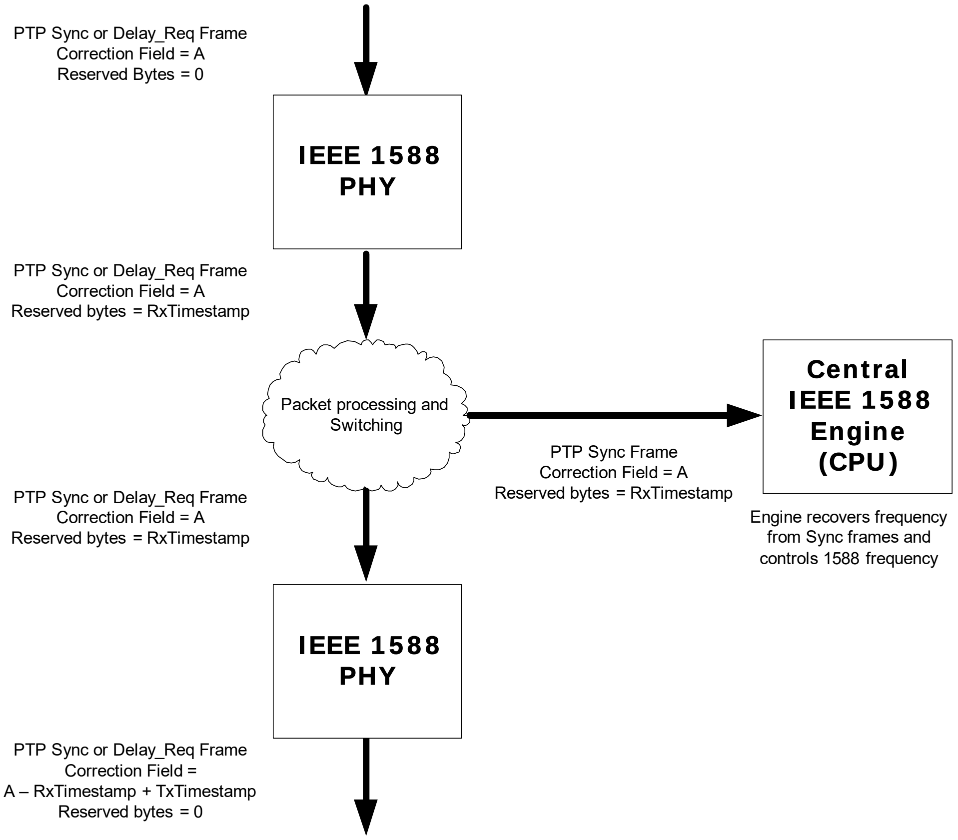
If the time stamp block gets the Add action, it adds the current value of the active_timestamp to the value of the correction field received from the analyzer and outputs the value to the rewriter block.

When the rewriter block receives a signal from the analyzer block to rewrite a specific position in the frame, it overwrites the position with the data received from the time stamp block and replaces the FCS of the frame. The rewriter also checks the original FCS of the frame and ensures that a frame that is received with a bad FCS and then modified by the rewriter is also sent out with a bad FCS. This is achieved by inverting the new FCS.

The following full calculations are performed:

- Sync frames: Correction field = Internal Correction field + (Raw_Timestamp_ns + Local_correction)
- Delay_req frames: Correction field = Internal Correction field + (Raw_Timestamp_ns + Local_correction) – Asymmetry

Figure 28 • **One-Step End-to-End Transparent Clock Mode B**



3.6.7.3 Ingress (Mode B)

In ingress mode B, all calculations are performed at the egress port.

On the ingress side, when the analyzer detects Sync or Delay_req frames, it adds the Rx time stamp to the four reserved bytes in the PTP frame.

The following full calculations are performed:

- Sync frames: Reserved_bytes = Raw_Timestamp_ns – Local_correction Correction field = Original Correction field + Asymmetry
- Delay_req frames: Reserved_bytes = Raw_Timestamp_ns – Local_correction

3.6.7.4 Egress (Mode B)

All calculations are done at the egress side. When the analyzer detects Sync or Delay_req frames, it performs the following calculation:

- Correction field = Original Correction field + Tx time stamp – Rx time stamp

The value of the Rx time stamp is extracted from four reserved bytes in the PTP header. The four reserved bytes are cleared back to 0 before transmission.

The result is that every Sync and Delay_req frame that belongs to the PTP domain(s) and is configured as one-step E2E TC in the system will exit the system with a correction field that contains the following:

- Correction field = Original correction field + Tx time stamp – Rx time stamp

All this is done without any interaction with a CPU system, other than the initial setup. There is no bandwidth expansion. Standard switching/routing tunneling can be done between the ingress and egress PHY, provided that the analyzers in the ingress PHY and egress PHY are set up to catch the Sync and Delay_req on both. If the PTP Sync and Delay_req frames are modified inside the system, the egress analyzer must be able to detect the egress Sync and Delay_req frames; otherwise, the egress Sync and Delay_req frames will have an incorrect correction field.

The following full calculations are performed:

- Sync frames: Correction field = Original Correction
  field + (Raw_Timestamp_ns + Local_correction) – Reserved_bytes
- Delay_req frames: Correction field = Original Correction
  field + (Raw_Timestamp_ns + Local_correction) – Reserved_bytes – Asymmetry

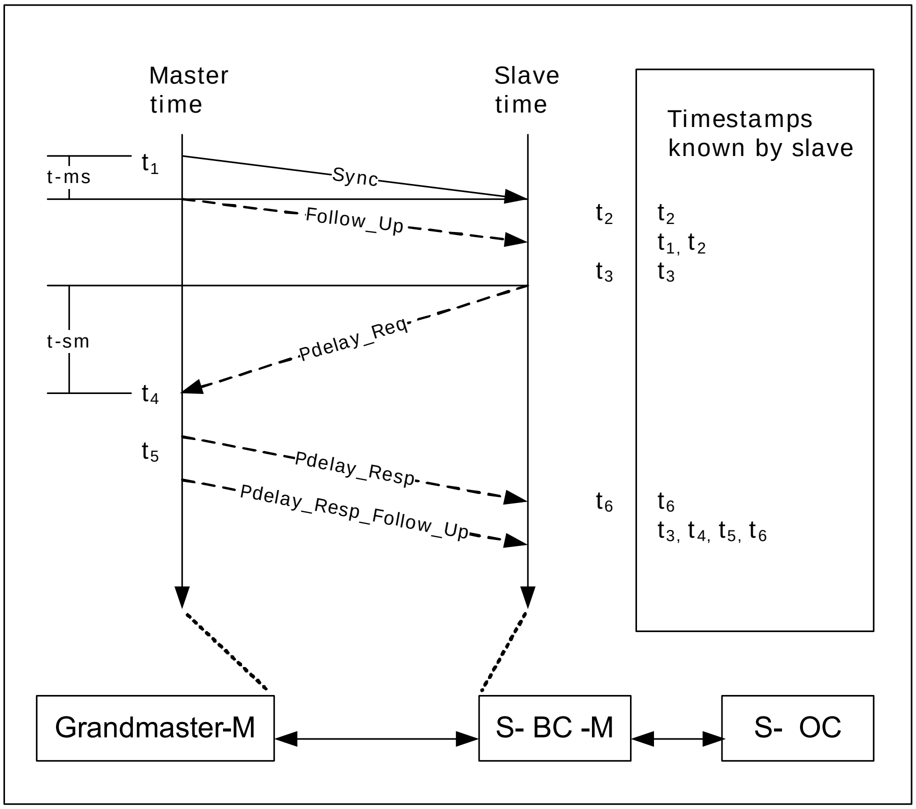
3.6.8 Supporting One-Step Peer-to-Peer Transparent Clock

When a Sync frame traverses a P2P TC, the correction field is updated with both the residence time and the calculated path delay on the port that the Sync frame came in on.

3.6.8.1 Peer Link Delay Measurement

In P2P TC, the P2P TC device actively sends and receives Pdelay_req and Pdelay_resp messages, and calculates the path delays to each neighbor node in the PTP network. The following illustration shows the delay measurements.

Figure 29 • **Delay Measurements**



To calculate the path delays on a link, the IEEE 1588 engine (located somewhere in the system) generates Pdelay_req messages on all ports. When transmitted, the actual Tx time stamp (t3) is saved for the CPU to read.

When a P2P TC, BC, or OC receives a Pdelay_req frame, it saves the Rx time stamp (t4) and generates a Pdelay_resp frame, which adds t5 – t4 to the correction field copied from the received Pdelay_req frame, where t5 is the time that the Pdelay_resp leaves the port (t5).

When a P2P TC receives the Pdelay_resp frame, it saves the Rx time stamp (t6) and then calculates the path delay as (t6 – t3 – the correction field of the frame)/2. The time stamp corrections are combined into a single formula as follows:

- Path delay = (t6 – (t3 + (t5 – t4))/2 = (t6 – t3 – t5 + t4)/2 = ((t4 – t3) + (t6 – t5))/2

The two path delays are divided by two, but in such a way as to cancel out any timing difference between the two devices.

A slight modification can be made to the algorithm to remove the CPU processing overhead of reading the t3 time stamp. To modify the algorithm, the IEEE 1588 engine should send the Pdelay_req message with a software generated t3 value in the origin time stamp, the sub-second value of the t3 time stamp in the reserved bytes of the PTP header, and a correction field of 0. The software generated t3 time stamp should just be within a second before the actual t3 time. The egress PHY should then be configured to perform E2E TC egress operation, meaning calculate the “residence time” from the inserted t3 time stamp to the actual t3 time and insert this value in the correction field of the frame. When the IEEE 1588 engine receives the corresponding Pdelay_resp frame back it can use the software generated t3 value as the correction field of the Pdelay_resp frame will contain a value that compensates for the actual t3 transmission time.

A P2P TC adds the calculated one-way path delay to the ingress correction field, and this ensures that the time stamp + correction field in the egress Sync frames is accurate and a slave connected to the P2P TC only needs to add the link delay from the TC to the slave.

The following sections describe both the standard and modified methods for taking P2P measurements. As with E2E TC operations, the VSC8489-17 device also supports the different TC modes: mode A (with different time stamp formats) and mode B. Mode B is also the preferred method to implement P2P TC.

3.6.8.2 Ingress, Mode A

If the analyzer detects that the frame is a IEEE 1588 Sync frame belonging to the PTP domain(s) of the system, it signals to the time stamp block which action to perform (subtract_p2p) along with the correction field of the frame. It also delivers the write offset and data size (location of the correction field inside the frame, 8 bytes wide) to the rewriter.

If the analyzer detects that the frame is a IEEE 1588 Pdelay_req or Pdelay_resp frame belonging to the PTP domain(s) of the system, it signals to the time stamp block which action to perform (Write). It also delivers the write offset and data size (location of the reserved 4 bytes in the PTP header that is used to save the ns part of the Rx time stamp, 4 bytes wide) to the rewriter.

If the time stamp block gets the subtract_p2p action, it subtracts the value in the ingress time stamp from the correction_field data, adds the configured path delay value, and delivers the result to the rewriter block.

If the time stamp block gets the Write action, it outputs the value of the ingress time stamp register to the rewrite block and the rewriter block writes the sub-second value to the reserved bytes in the PTP header.

The following full calculations are performed:

- Sync frames: Internal Correction field = Original Correction field – (Raw_Timestamp_ns – Local_correction) + Path_delay + Asymmetry
- Pdelay_req frames: Reserved_bytes = Raw_Timestamp_ns – Local_correction
- Pdelay_resp frames: Reserved_bytes = Raw_Timestamp_ns – Local_correction
  Correction Field = Original Correction field + Asymmetry

3.6.8.3 Egress, Mode A

If the analyzer detects that the frame is a IEEE 1588 Sync frame belonging to the PTP domain(s) of the system, it signals to the time stamp block which action to perform (Add), along with the correction field of the frame. It also delivers the write offset and data size (location of the correction field inside the frame, 8 bytes wide) to the rewriter.

If the analyzer detects that the frame is a IEEE 1588 Pdelay_req frame belonging to the PTP domain(s) of the system, it signals to the time stamp block which action to perform (Sub_add), along with the original correction field of the frame (will have the value of 0) and the time stamp extracted from the reserved bytes. It also delivers the write offset and data size (location of the correction field inside the frame, 8 bytes wide) to the rewriter.

If the user prefers to use to use the normal t3 handling where the t3 time stamp is saved in a time stamp FIFO, the following configuration should be used: If the analyzer detects that the frame is a IEEE 1588 Pdelay_req frame belonging to the PTP domain(s) of the system, it signals to the time stamp block which action to perform (Write, Save), along with the original correction field of the frame (will have the value of 0). It also delivers the write offset and data size (0- No data is actually written into the frame) to the rewriter. In addition, it outputs the field that holds the frame identifier (sequenceId from the PTP header) to the time stamp FIFO, to save along with the Tx time stamp.

If the analyzer detects that the frame is a IEEE 1588 Pdelay_resp frame belonging to the PTP domain(s) of the system, it signals to the time stamp block which action to perform (Sub_add), along with the original correction field of the frame (will have the value of the CF received from the Pdelay_req frame) and the time stamp extracted from the reserved bytes. It also delivers the write offset and data size (location of the correction field inside the frame, 8 bytes wide) to the rewriter.

If the analyzer detects that the frame is not matched, it signals to the time stamp block and the rewriter block to ignore the frame (let it pass unmodified and flush the saved time stamp in the time stamp block).

The following full calculations are performed:

- Sync frames: Correction field = Internal Correction field + (Raw_Timestamp_ns + Local_correction)

- Pdelay_req frames: Correction field = Internal Correction field + (Raw_Timestamp_ns + Local_correction) – Reserved_bytes – Asymmetry
- Pdelay_resp frames: Correction field = Original Correction field + (Raw_Timestamp_ns + Local_correction) – Reserved_bytes

3.6.8.4 Ingress, Mode B

If the analyzer detects that the frame is a IEEE 1588 Sync frame belonging to the PTP domain(s) of system, it signals to the time stamp block which action to perform (subtract_p2p), along with the correction field of the frame. It also delivers the write offset and data size (location of the correction field inside the frame, 8 bytes wide) to the rewriter.

If the analyzer detects that the frame is a IEEE 1588 Pdelay_req frame belonging to the PTP domain(s) of system, it signals to the time stamp block which action to perform (Write). It also delivers the write offset and data size (location of the reserved 4 bytes in the PTP header used to save the ns part of the Rx time stamp, 4 bytes wide) to the rewriter.

If the analyzer detects that the frame is a IEEE 1588 Pdelay_resp frame belonging to the PTP domain(s) of system, it signals to the time stamp block which action to perform (Write). It also delivers the write offset and data size (location of the reserved 4 bytes in the PTP header used to save the ns part of the Rx time stamp, 4 bytes wide) to the rewriter.

If the time stamp block gets the Subtract_p2p action, it subtracts the value in the active_timestamp_ns_p2p register from the correction_field data, and outputs the value on the New_Field bus to the Rewriter block.

If the time stamp block gets the Write action, it outputs the value of the active_timestamp_ns register on the New_field bus to the Rewriter block.

The following full calculations are performed:

- Sync frames: Internal Correction field = Original Correction field – (Raw_Timestamp_ns – Local_correction) + Path_delay + Asymmetry
- Pdelay_req frames: Reserved_bytes = Raw_Timestamp_ns – Local_correction
- Pdelay_resp frames: Reserved_bytes = Raw_Timestamp_ns – Local_correction + Asymmetry

3.6.8.5 Egress, Mode B

If the analyzer detects that the frame is a IEEE 1588 Sync frame belonging to the PTP domain(s) of the system, it signals to the time stamp block which action to perform (Add), along with the correction field of the frame. It also delivers the write offset and data size (location of the correction field inside the frame, 8 bytes wide) to the rewriter.

If the analyzer detects that the frame is a IEEE 1588 Pdelay_req frame belonging to the PTP domain(s) of system, it signals to the time stamp block which action to perform (Write, Save), along with the original correction field of the frame (will have the value of 0). It also delivers the write offset and data size (0- No data is actually written into the frame) to the rewriter. In addition, it outputs the field that holds the frame identifier (sequenceId from the PTP header) to the time stamp FIFO, to save along with the Tx time stamp.

If the analyzer detects that the frame is a IEEE 1588 Pdelay_resp frame belonging to the PTP domain(s) of system, it signals to the time stamp block which action to perform (Add - this requires that the IEEE 1588 engine has subtracted the Rx time stamp from the correction field), along with the original correction field of the frame. It also delivers the write offset and data size (location of the correction field inside the frame, 8 bytes wide) to the rewriter.

If the time stamp block gets the Write, Save action, it outputs the value of the active_timestamp_ns register on the New_field bus to the Rewriter block and sets the save_timestamp bit.

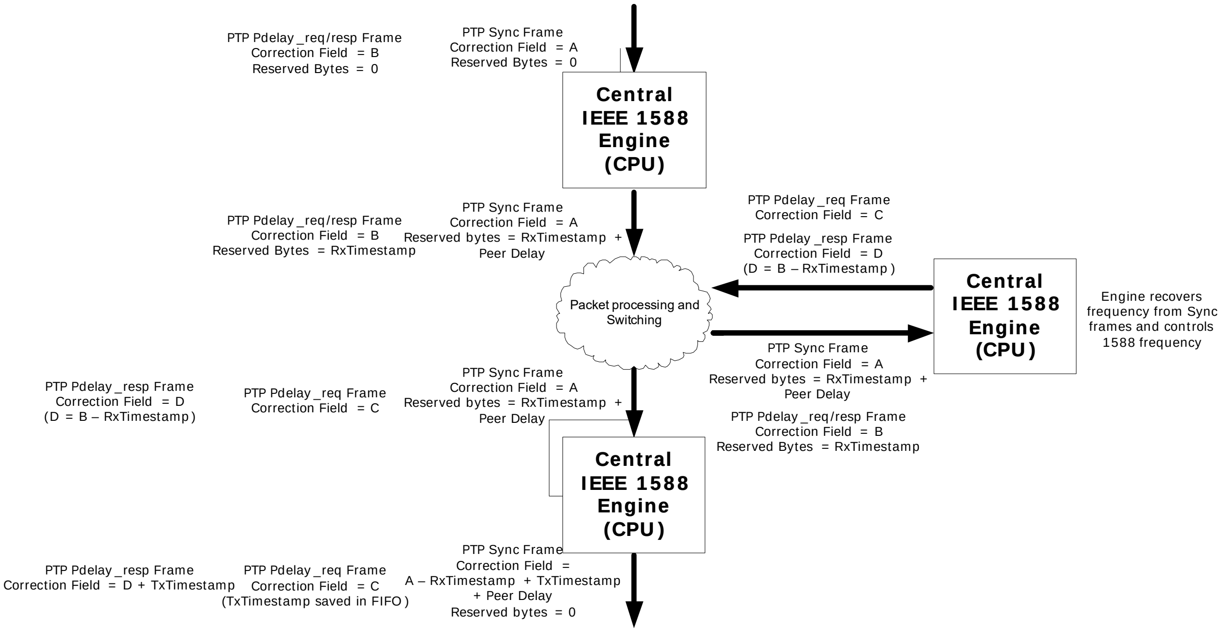
If the time stamp block gets the Add action, it adds the correction field value to the value in the active_timestamp_ns register and outputs the value on the New_Field bus to the Rewriter block.

The Tx time stamp FIFO block contains an (implementation specific) amount of buffer memory. It simply stores the Tx time stamp values that it receives from the time stamp block together with the frame identifier data it receives from the Analyzer block, and has a CPU interface that allows the IEEE 1588 engine to read out the time stamp sets (Frame identifier + New Tx time stamp).

The following full calculations are performed:

- Sync frames: Correction field = Internal Correction field + (Raw_Timestamp_ns + Local_correction)
- Pdelay_req frames: FIFO = Raw_Timestamp_ns + Local_correction – Asymmetry
- Pdelay_resp frames: Correction field = Internal Correction field + (Raw_Timestamp_ns + Local_correction)

Figure 30 • **One-Step Peer-to-Peer Transparent Clock Mode B**

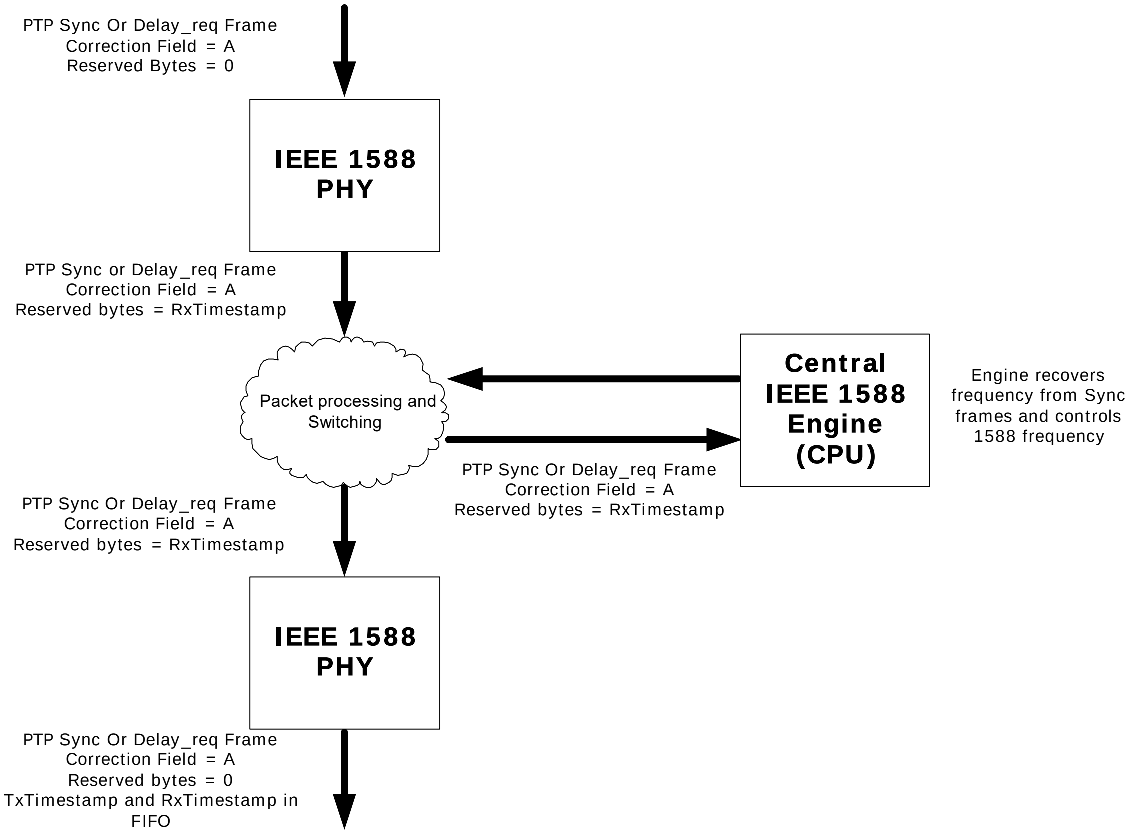


3.6.9 Supporting Two-Step Transparent Clock

In two-step transparent clocks, the Rx and Tx time stamps are saved for the IEEE 1588 engine to read, and the follow-up message is redirected to the IEEE 1588 engine so that it can update the correction field with the residence time.

Even though two-step transparent clocks can be used with this architecture, it is also possible to process the frames in the same manner as a one-step TC, because the slaves are required to take both the correction fields from the Sync frames and the follow-up frames into account. This significantly reduces the CPU load for the TC. The following illustration shows two-step transparent clock normal operation.

Figure 31 • **Two-Step End-to-End Transparent Clock**



3.6.9.1 Ingress

If the analyzer detects that the frame is a IEEE 1588 Sync or Delay_req frame belonging to the PTP domain(s) of the system, it signals to the time stamp block which action to perform (Write). The analyzer also delivers the write offset and data size to the rewriter (four reserved bytes in the PTP header, which will be passed out on the egress port of the system). A changed reserved value may be significant in security protection. This method allows the frames to be copied to the IEEE 1588 engine, so that it can extract the Rx time stamp and knows that it needs to read the Tx time stamps to be ready for the follow up message. It is also possible to save the Rx time stamp value along with the Tx time stamp in the Tx time stamp FIFO.

If the time stamp block gets the Write action, it outputs the current time stamp to the rewriter and the rewriter writes the ns part of the time stamp into the reserved bytes and recalculates FCS.

The following full calculations are performed:

- Sync frames: Reserved_bytes = (Raw_Timestamp_ns – Local_correction) Correction field = Original Correction field + Asymmetry
- Delay_req frames: Reserved_bytes = Raw_Timestamp_ns – Local_correction

3.6.9.2 Egress

If the analyzer detects that the frame is a IEEE 1588 Sync or Delay_req frame belonging to the PTP domain(s) of the system, it signals to the time stamp block which action to perform (Write, Save). The analyzer also delivers the write offset and data size (but as nothing is to be overwritten the values will be 0) to the rewriter. The analyzer outputs 10 bytes of frame identifier to the Tx time stamp FIFO to be saved along with the Tx time stamp. The frame identifier must include, at minimum, the sequenceId field so the CPU can match the time stamp with the follow-up frame. The analyzer also outputs the offset for the reserved fields in the PTP header to the rewriter, so that the rewriter field is reset to 0 and the temporary Rx time stamp value is cleared.

If the time stamp block gets the Write, Save action it outputs the current time stamp value to the rewriter (and time stamp FIFO) and sets the save_timestamp bit. The time stamp FIFO block saves the New_field data along with the frame identifier data it received from the analyzer block. The frame identifier data that is saved can contain the reserved field in the PTP header that was written with the Rx time stamp, so that the CPU now can read the set of Tx and Rx time stamp from the Tx time stamp FIFO.

The following full calculations are performed:

- Sync frames: FIFO = Raw_Timestamp_ns + Local_correction (reserved_bytes containing the Rx time stamp saved together with Tx time stamp)
- Delay_req frames: FIFO = Raw_Timestamp_ns + Local_correction – Asymmetry (reserved_bytes containing the Rx time stamp saved together with Tx time stamp)

3.6.10 Calculating OAM Delay Measurements

Frame delay measurements can be made as one-way and two-way delay measurements. Microsemi recommends that the delay measurement be measured before the packets enter the queues if the purpose is to measure the delay for different priority traffic, but it can be used with time stamping in the PHY to measure the delay through the network devices placed in the path between the measurement points.

The function is mainly an on-demand OAM function, but it can run continuously.

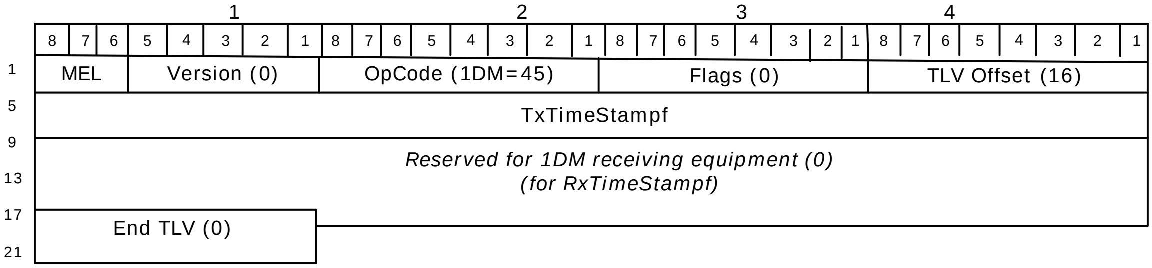
3.6.11 Supporting Y.1731 One-Way Delay Measurements

One-way delay measurements require that the two peers are synchronized in time. When they are not synchronized, only frame delay variations can be measured.

The MEP periodically sends out 1DM OAM frames containing a TxTimeStampf value in IEEE 1588 format.

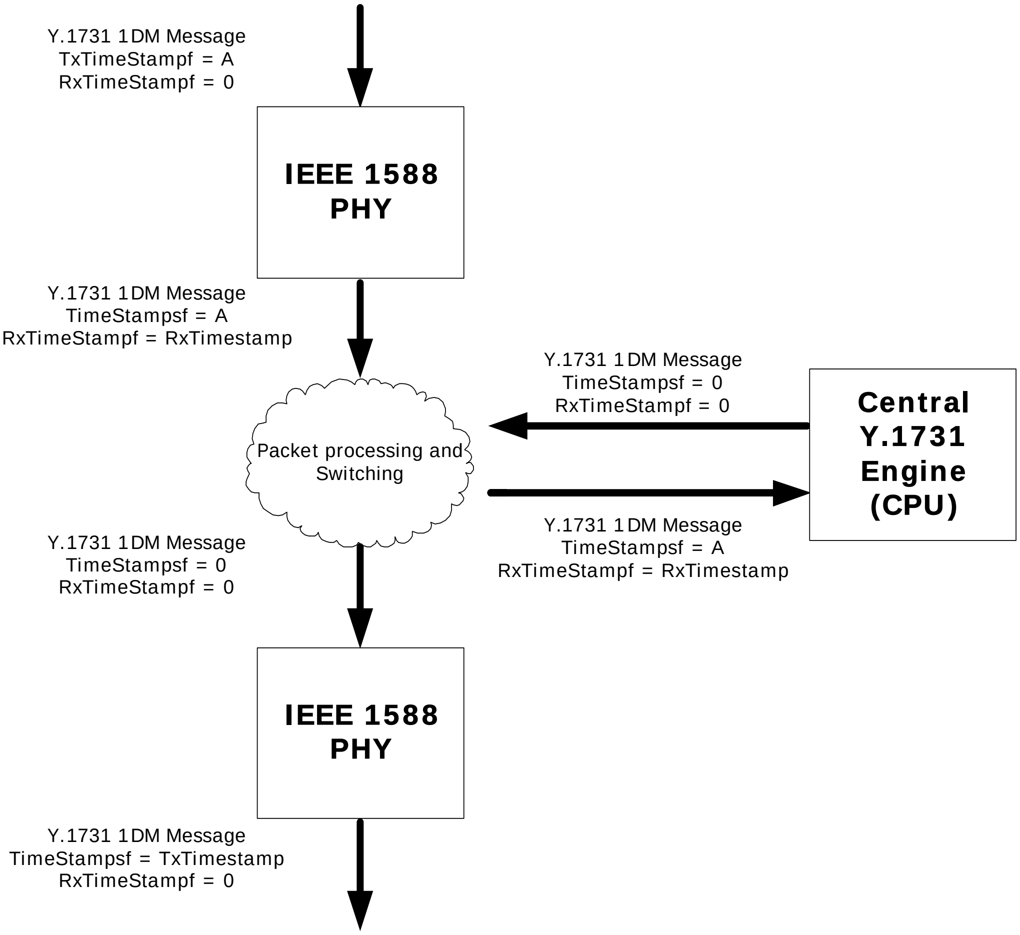
The receiver notes the time of reception of the 1DM frame and calculates the delay.

Figure 32 • **Y.1731 1DM PDU Format**



1. For one-way delay measurements, both MEPs must support IEEE 1588 and be in sync.
2. 1DM frame is generated by the CPU, but with an empty Tx time stamp.
3. The frame is transmitted by the initiating MEP.
4. The 1DM frame is classified as an outgoing 1DM frame by the egress PHY and the PHY rewrites the frame with the time as TxFCf.
5. The receiving PHY classifies the incoming 1DM frame and writes the receive time stamp in reserved place (RxTimeStampf).
6. The frame is received by the peer MEP.
7. The frame is forwarded to the CPU that can calculate the delay.

Figure 33 • **Y.1731 One-Way Delay**



3.6.11.1 Ingress

If the analyzer detects that the frame is a Y.1731 1DM PDU frame belonging to the MEP, it signals to the time stamp block which action to perform (Write). The analyzer also delivers the write offset and data size (location of the RxTimeStampf location in the frame, 8 bytes wide) to the rewriter.

If the time stamp block gets the Write action, it delivers the time stamp to the rewriter block and the rewriter block adds this time stamp to the reserved bytes in the frame and recalculates FCS.

The following calculation is performed for 1DM frames:

- RxTimeStampf = (Raw_Timestamp – Local_correction)

3.6.11.2 Egress

If the analyzer detects that the frame is a Y.1731 1DM PDU frame belonging to the MEP, it signals to the time stamp block which action to perform (Write). It also delivers the write offset and data size (location of the TxTimeStampf location in the frame, 8 bytes wide) to the rewriter.

If the time stamp block gets the Write action, it delivers the time stamp to the rewriter block and the rewriter block adds this time stamp to the reserved bytes in the frame and recalculates FCS.

The following calculation is performed for 1DM frames:

- TxTimeStampf = (Raw_Timestamp + Local_correction)

3.6.12 Supporting Y.1731 Two-Way Delay Measurements

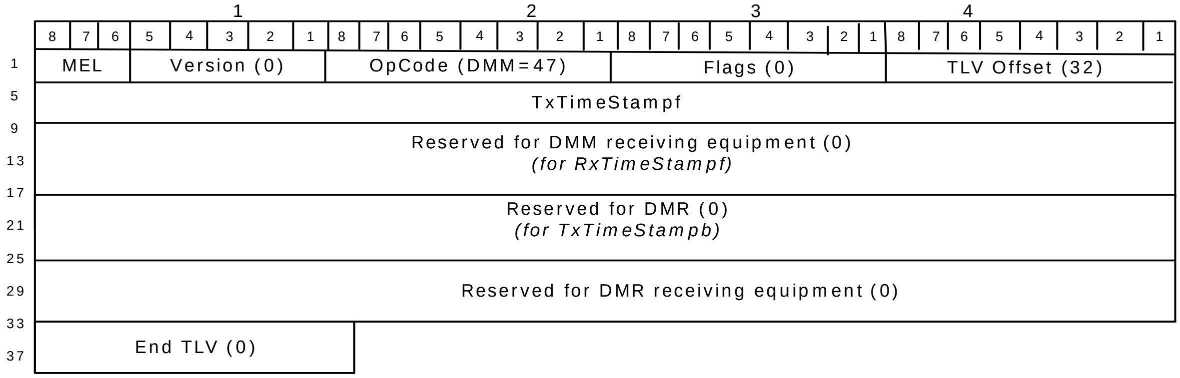
When performing two-way delay measurements, the initiating MEP transmits DMM frames containing a TxTimeStampf value. The receiving MEP replies with a DMR frame that is the same as the DMM frame, but with destination and source MAC address swapped and with a different OAMPDU opcode.

When the DMR frame is received back at the initiating MEP, the time of reception is noted and the total delay is calculated.

As an option, it is allowed to include two additional time stamps in the DMR frame: RxTimeStampf and TxTimeStampb. These contain the time that the DMM page is received for processing and the time the responding DMR reply is sent back, both in IEEE 1588 format.

Including these time stamps allows for the exclusion of the processing time in the peer MEP, but it does not require that the two MEPs are synchronized.

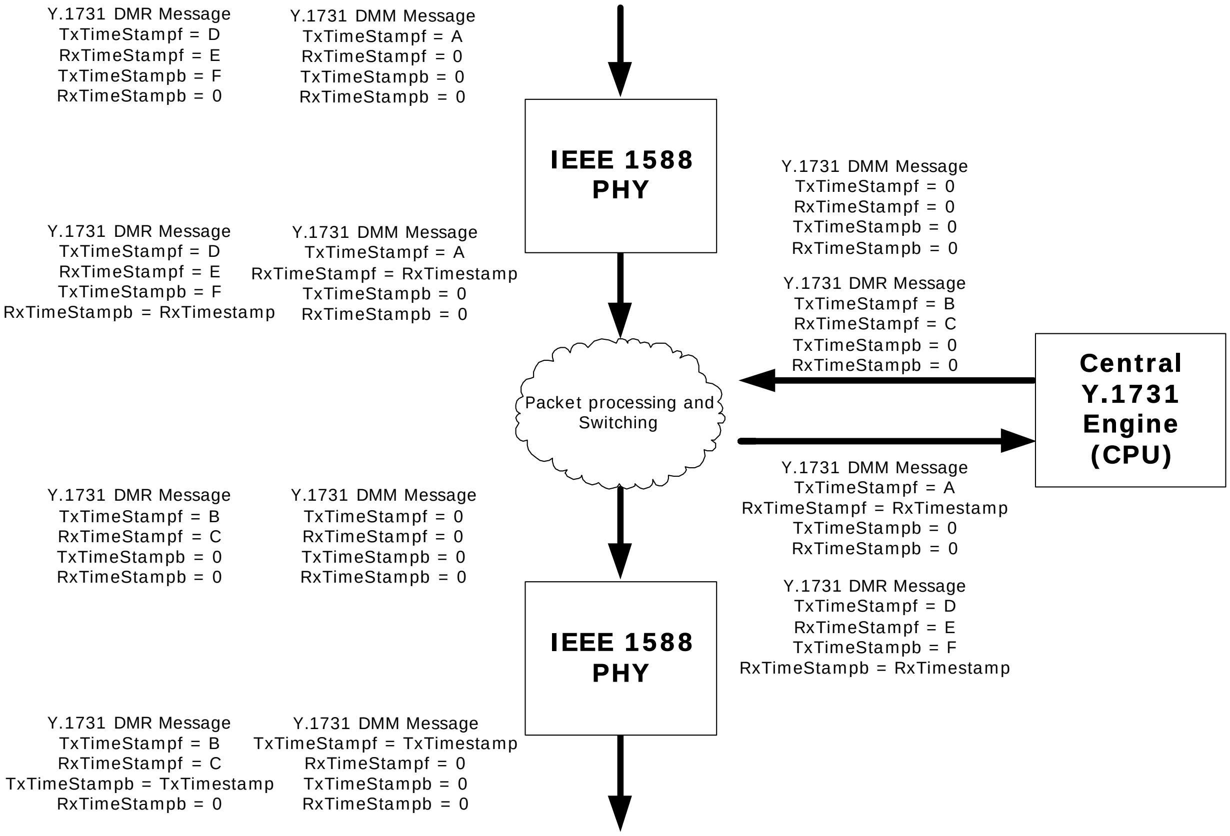
Figure 34 • Y.1731 DMM PDU Format



In that case, the following frame flow is needed (two-way delay measurement):

1. DMM frame is generated by the CPU (initiating MEP), but with an empty Tx time stamp.
2. In the egress PHY the DMM frame is classified as an outgoing DMM frame from the MEP and the PHY rewrites the frame with the time as TxTimeStampf.
3. In the ingress PHY the frame is classified as an incoming DMM belonging to the MEP and the RxTimeStampf in the frame is written (the frame has a reserved space for this).
4. The DMM frame is forwarded to the MEP (CPU).
5. The CPU processes the frame (swaps SA/DA MAC addresses, modifies the opcode to DMT) and sends out a DMT frame.
6. The outgoing DMT frame is detected in the egress PHY and the TxTimeStampb is written into the frame.
7. In the ingress PHY the frame is classified as an incoming DMT belonging to the MEP and the RxTimeStampb in the frame in written (the frame has a reserved space for this).
8. The frame is forwarded to the CPU that can calculate the delays.

Figure 35 • **Y.1731 Two-Way Delay**



3.6.12.1 Ingress

If the analyzer detects that the frame is a Y.1731 DMM PDU frame belonging to the MEP, it signals to the time stamp block which action to perform (Write). It also delivers the write offset and data size (location of the RxTimeStampf location in the frame, 8 bytes wide) to the rewriter.

If the analyzer detects that the frame is a Y.1731 DMT PDU frame belonging to the MEP, it signals to the time stamp block which action to perform (Write). It also delivers the write offset and data size (location of the RxTimeStampf location in the frame, 8 bytes wide) to the rewriter.

If the time stamp block gets the Write action, it delivers the time stamp to the rewriter block and the rewriter block adds this time stamp to the reserved bytes in the frame and recalculates FCS.

The following calculations are performed:

- DMM frames: RxTimeStampf = (Raw_Timestamp – Local_correction)
- DMR frames: RxTimeStampb = (Raw_Timestamp – Local_correction)

3.6.12.2 Egress

If the analyzer detects that the frame is a Y.1731 DMM PDU frame belonging to the MEP, it signals to the time stamp block which action to perform (Write). It also delivers the write offset and data size (location of the TxTimeStampf location in the frame, 8 bytes wide) to the rewriter.

If the analyzer detects that the frame is a Y.1731 DMT PDU frame belonging to the MEP, it signals to the time stamp block which action to perform (Write). It also delivers the write offset and data size (location of the TxTimeStampb location in the frame, 8 bytes wide) to the rewriter.

If the time stamp block gets the Write action, it delivers the time stamp to the rewriter block and the rewriter block adds the time stamp to the reserved bytes in the frame and recalculates FCS as follows:

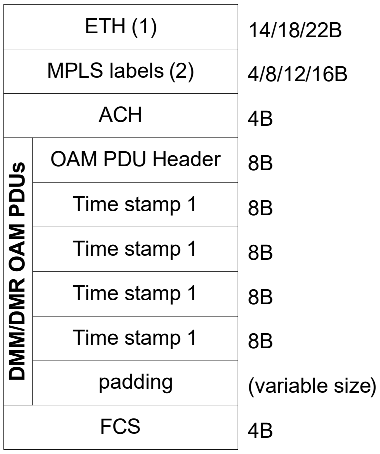
- DMM frames: TxTimeStampf = (Raw_Timestamp + Local_correction)

- DMR frames: TxTimeStampb = (Raw_Timestamp + Local_correction)

3.6.12.3 Supporting MPLS-TP One-Way and Two-Way Delay Measurements

MPLS-TP one- and two-way delay measurement are defined in RFC6374 (G.8113.2) and G.8113.1 (draft-bhh). These mechanisms are similar to the ones described for Y.1731 Ethernet OAM delay measurement, except for the encapsulations. The following illustrations show the PDU formats.

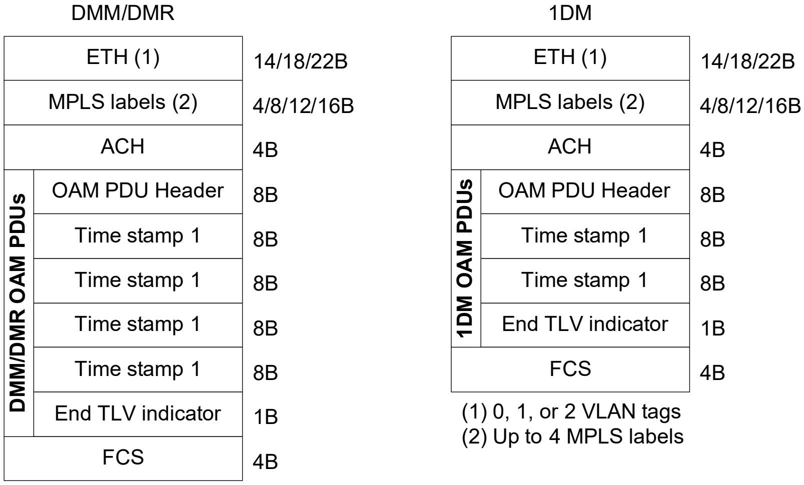
Figure 36 • **RFC6374 DMM/DMR OAM PDU Format**



	Field	Size
	ETH (1)	14/18/22B
	MPLS labels (2)	4/8/12/16B
	ACH	4B
DMM/DMR OAM PDUs	OAM PDU Header	8B
	Time stamp 1	8B
	Time stamp 1	8B
	Time stamp 1	8B
	Time stamp 1	8B
	padding	(variable size)
	FCS	4B

(1) 0, 1, or 2 VLAN tags
(2) Up to 4 MPLS labels

Figure 37 • **Draft-bhh DMM/DMR/1DM OAM PDU Formats**



DMM/DMR

	Field	Size
	ETH (1)	14/18/22B
	MPLS labels (2)	4/8/12/16B
	ACH	4B
DMM/DMR OAM PDUs	OAM PDU Header	8B
	Time stamp 1	8B
	Time stamp 1	8B
	Time stamp 1	8B
	Time stamp 1	8B
	End TLV indicator	1B
	FCS	4B

(1) 0, 1, or 2 VLAN tags
(2) Up to 4 MPLS labels

1DM

	Field	Size
	ETH (1)	14/18/22B
	MPLS labels (2)	4/8/12/16B
	ACH	4B
1DM OAM PDUs	OAM PDU Header	8B
	Time stamp 1	8B
	Time stamp 1	8B
	End TLV indicator	1B
	FCS	4B

(1) 0, 1, or 2 VLAN tags
(2) Up to 4 MPLS labels

3.6.13 Device Synchronization for IEEE 1588 Support

It is important to keep all the local clock blocks synchronized to the accurate time over a complete system. To maintain ns accuracy, the signal routing and internal signal delays must be taken into account when configuring a system.

The architecture described in this document assumes that there is a global synchronous clock available in the system. If the system is a telecom system where the system is locked to a PRC, the system clock can be adjusted to match the PRC, meaning that once locked, the frequency of the system clock ensures that the local clocks are progressing (counting) with the accurate frequency. This system clock can be locked to the PRC using IEEE 1588, SyncE, SDH, or by other means.

A global timing signal must also be distributed to all the devices. This could be a 1 pps pulse or another slow synchronization pulse, like a 4 kHz synchronization frequency. It can also just be a one-shot pulse. The system CPU can load each local counter with the time value that happens next time the synchronization pulse goes high (+ the known delay of the synchronization pulse traces). It can also just load the same approximate time value into all the local clock blocks (again + the known delay of the synchronization pulse traces) and load them in parallel. Then the local time can be adjusted to match the actual time by adjusting the local clock blocks using the ±1 ns function.

If the Save signal is triggered synchronously on all PHYs of the system, software can read the saved time stamp in each PHY and correct the time accordingly. On a blade with multiple PHYs, it is possible to connect the 1588_PPS_1 pin on one PHY to the 1588_LOAD_SAVE pin on the next PHY. If the routing delay (both internal chip delay and trace delay) is known, Microsemi recommends that the value saved in the next PHYs correspond to this delay.

If the global system clock is not synchronous, the PPM offset between system clock and the IEEE 1588 time progress can be calculated. This PPM offset can be used to calculate how many local-time-clocks is takes to reach a time offset of 1 ns and this value can be programmed into each local time block. The CPU still need to keep track of the smaller PPM offset and adjust the local time blocks with ± writes when necessary.

By measuring the skew between the 1 pps test output from each PHY, it is possible to measure the nominal correction values for the time counters in a system. These can be incorporated into the software of the system. Variations from system to system and temperature variations should be minimized by design.

3.6.14 Time Stamp Update Block

The IEEE 1588 block is also called the Time Stamp Update block (TSU) and supports the implementation of IEEE 1588v2 and ITU-T Y.1731 in PHY hardware by providing a mechanism for time stamp update (PTP) and time stamping (OAM).

The TSU block works with other blocks to identify PTP/OAM messages, process these messages, and insert accurate time stamp updates/time stamps where necessary. For IEEE 1588 timing distribution, the VSC8489-17 device supports ordinary clocks, boundary clocks, end-to-end transparent clocks, and peer-to-peer transparent clocks in a chassis based IEEE 1588 capable system. One-step and two-step processing is also supported. For details on the timing protocol, refer to IEEE 1588v2. For OAM details, refer to ITU-T Y.1731 and G.8113.1/G.8113.2. The TSU block implements part of the functionality required for full IEEE 1588 compliance.

The IEEE 1588 protocol has four different types of messages that require action by the TSU: Sync, Delay_req, Pdelay_req, and Pdelay_resp. These frames may be encapsulated in other protocols, several layers deep. The processor is able to detect PTP messages within these other protocols. The supported encapsulations are as follows:

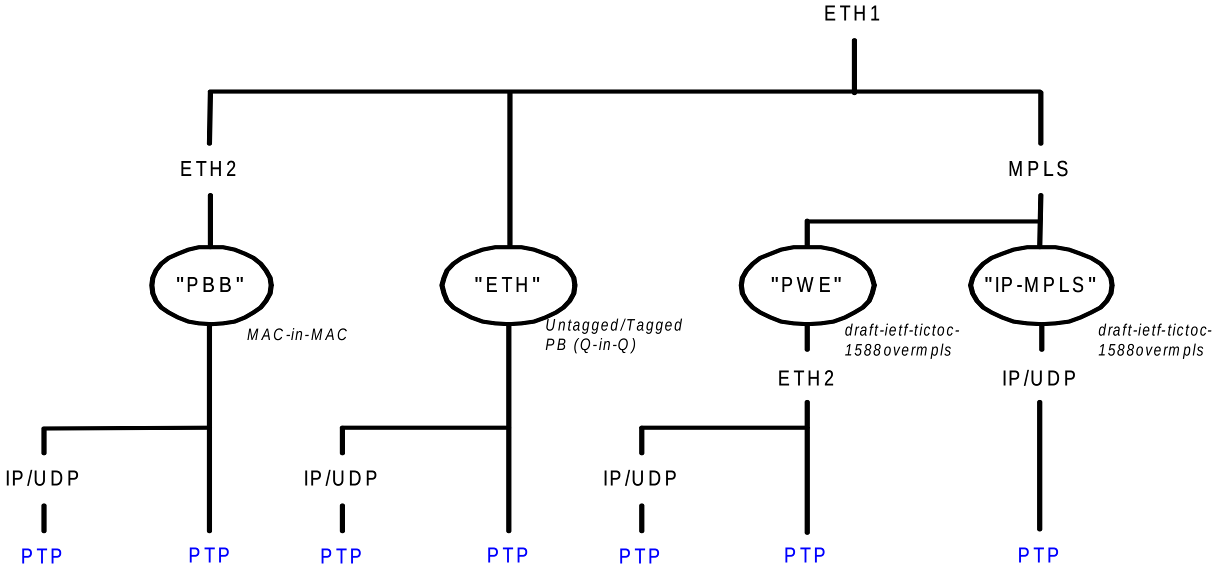
- Ethernet
- UDP over IPv4
- UDP over IPv6
- MPLS
- Pseudo-wires
- PBB and PBB-TE tunnels

OAM frames for delay measurement (1DM, DMM, and DMR) with the following supported encapsulations:

- Ethernet (Y.1731 Ethernet OAM)
- Ethernet in MPLS pseudo-wires (Y.1731 Ethernet OAM)
- MPLS-TP (G.8113.1 (~draft-bhh-mpls-tp-oam-y1731) and G.8113.2 (RFC6374))

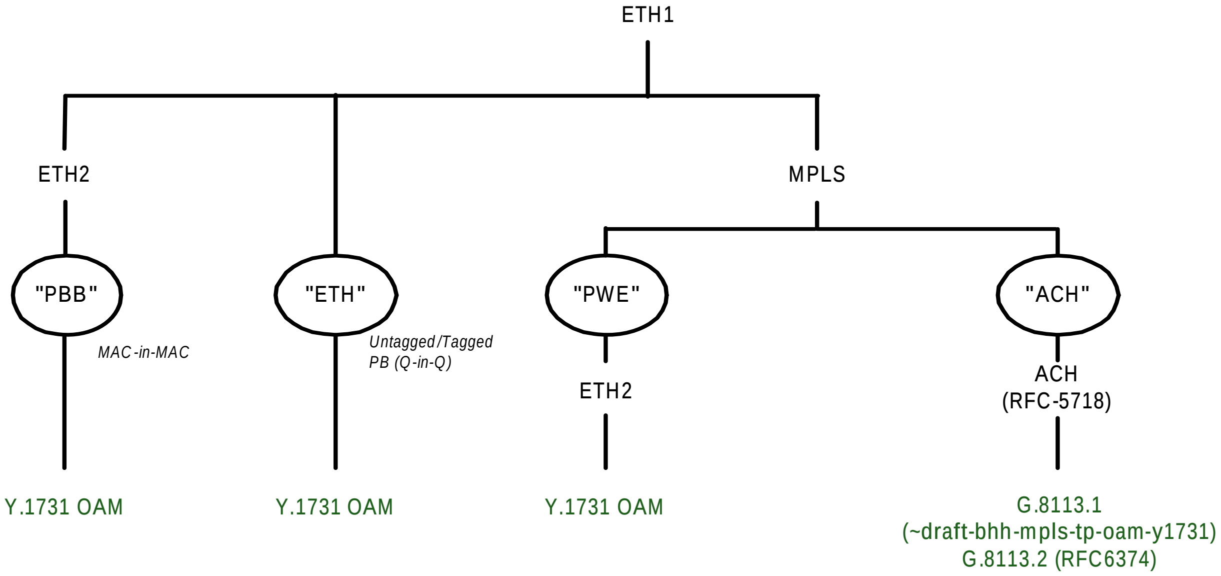
The following illustration shows an overview of the supported PTP encapsulations. Note that the implementation is flexible, so encapsulations not defined here may also be covered.

Figure 38 • **PTP Packet Encapsulations**



The following illustration shows the same overview of the supported encapsulations with the focus on OAM.

Figure 39 • **OAM Packet Encapsulations**



There is one TSU per channel in the VSC8489-17 device. The TSU detects and updates up to three different encapsulations of PTP/OAM. Non-matching frames are transferred transparently. This includes IFG, preamble, and SFD. For all frames, there is no bandwidth expansion/shrink.

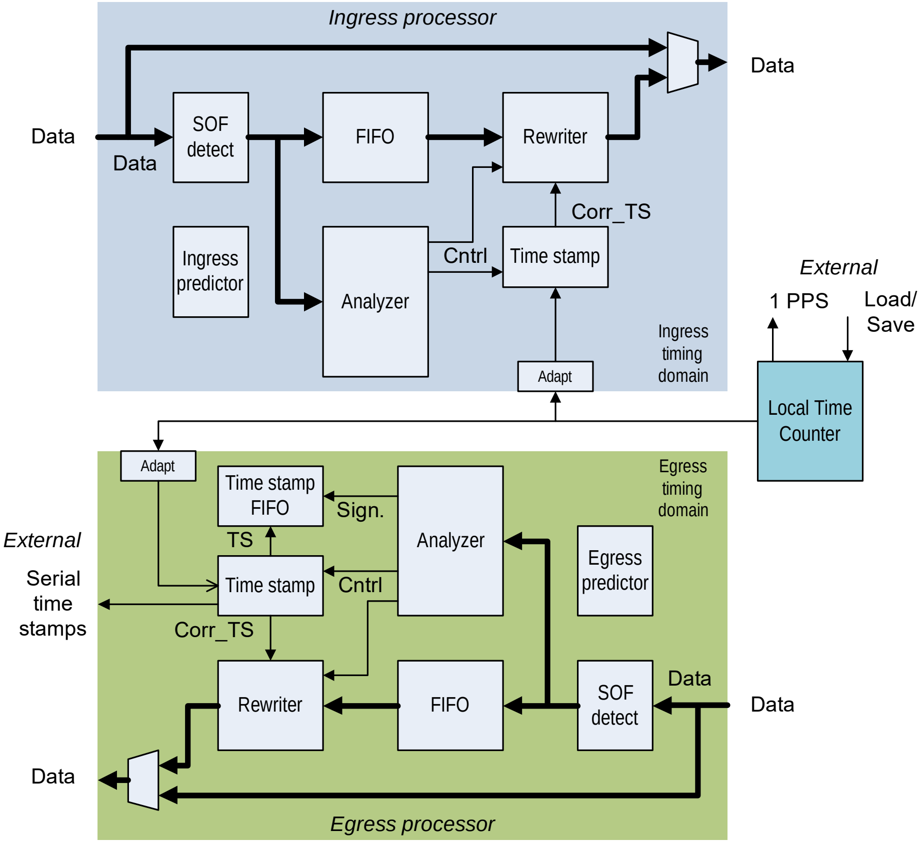
Once these frames are detected in the receive path, they are stamped with the ingress time and forwarded for further PTP/OAM processing. In the transmit path, the correction field of the appropriate PTP message (or the Rx and Tx fields of the OAM frame) is updated with the correct time stamp. A local time counter is maintained to provide the time stamps. Implementation of some of the IEEE 1588 protocol requires interaction with the TSU block over the CPU interface and external processing.

The system has an ingress processor, egress processor, and a local time counter. The ingress and egress processing logic blocks are identical, except that the time stamp FIFO is only required in the egress direction because the CPU needs to know the actual time stamps of some of the transmitted PTP frames. The CPU reads the time stamps and any associated frame information out of the time stamp FIFO. The FIFO saves the generated time stamps along with information that uniquely identifies the frame to be read out by the CPU.

The ingress and egress processing blocks run on the same clock as the data paths for the corresponding directions. The local time counter is the primary reference clock for the system and it maintains the local reference time used by the TSU logic. It should be synchronized by an external entity. The block provides a method to load and view its value when the 1588_LOAD_SAVE pin is asserted. The block also provides a one pulse-per-second output signal with a programmable duty cycle. The local time counter runs at several clock frequencies.

The following illustration shows the block diagram of the TSU.

Figure 40 • **TSU Block Diagram**



In both directions, the input data from the PHY layer is first fed to an SOF detect block. Data is then fed to both the programmable time-delay FIFO and the analyzer. The FIFO delays the data by the time needed to complete the operations necessary to update the PTP frame. That is, the data is delayed to the input of the rewriter so that the rewriter operations are known when the frame arrives. This includes the analyzer and time stamp processor block's functions.

The analyzer block checks the data stream and searches for PTP/OAM frames. When one is detected, it determines the appropriate operations to be performed based on the operating mode and the type of frame detected.

Note: The analyzer blocks of two channels share configuration registers and have identical setups.

The time stamp block waits for an SOF to be detected, captures a time stamp from the local time counter, and builds the new time stamp that is to be written into the PTP/OAM frame. Captured time stamps can be read by the CPU.

The rewriter block handles the actual writing of the new time stamp into the PTP/OAM frame. It is also able to clear parts of the frame such as the UDP checksum, if required, or it can update the frame to ensure that the UDP checksum is correct (for IPv6 PTP frames). The block also calculates the new FCS to be written to the PTP frame after updating the fields with the new time stamp.

The VSC8489-17 device has variable latency in the PCS block. These variations are predicted and used to compensate/maximize the accuracy of the IEEE 1588 time stamp logic.

If the time stamp update function is not used, the block can be bypassed. When the TSU is bypassed, the block can be configured and then enabled and taken out of bypass mode. The change in bypass mode takes effect only when an IDLE is in the bypass register. This allows the TSU block to be switched on without corrupting data.

Each direction of the IEEE 1588 can be bypassed individually by programming the INTERFACE_CTL.SPLIT_BYPASS bit. Bypass is then controlled by INTRERFACE_CTL.INGR_BYPASS and INTERFACE_CTL.EGR_BYPASS.

Pause frames pass unmodified through the TSU, but the latency may cause a violation of the allowed pause flow-control latency limits per IEEE 802.3.

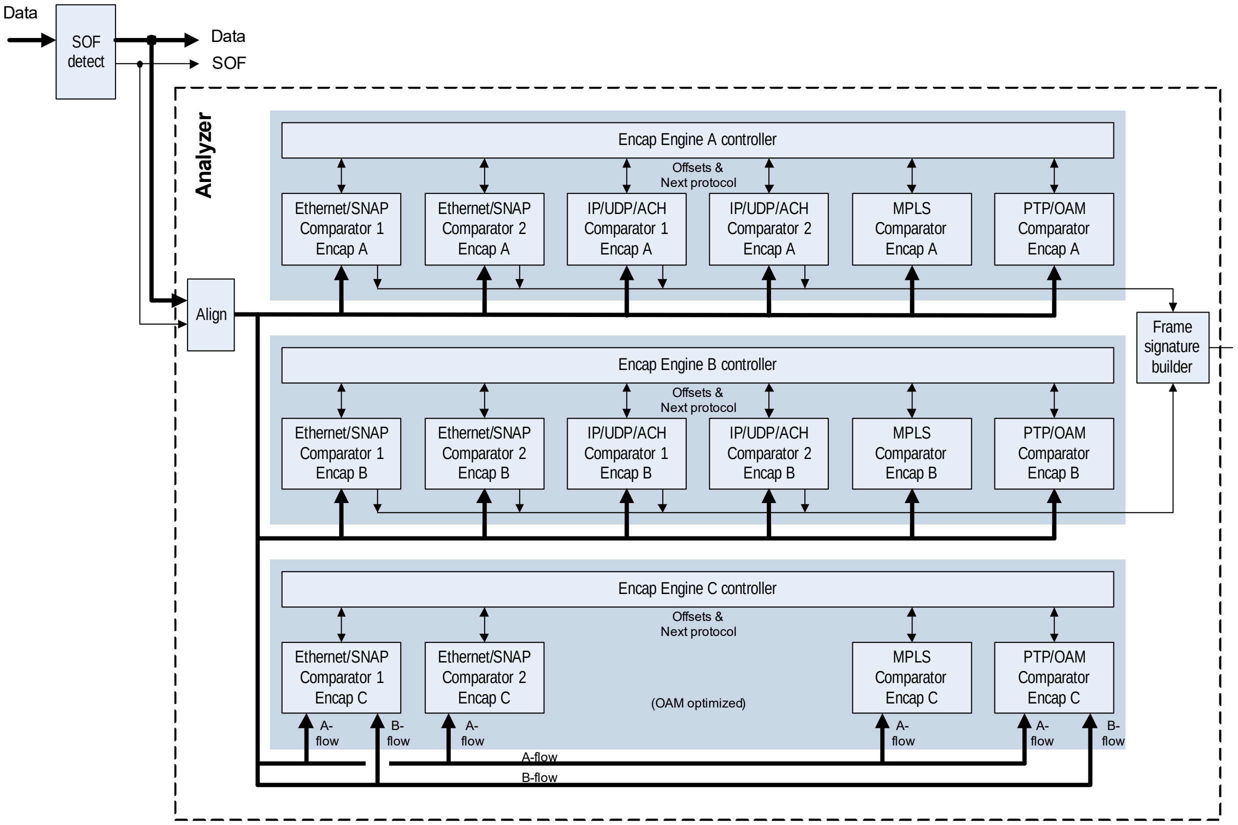
3.6.15 Analyzer

The packet analyzer parses incoming packets looking for PTP/OAM frames. It determines the offset of the correction field within the packet for all PTP frames/for the time stamp in Y.1731 OAM frames. The analyzer has the following characteristics:

- Can compare against two different filter sets plus one optimized for OAM
- Each filter targets PTP or OAM frames
- Flexible comparator sequence with fixed start (Ethernet/SNAP) and end (PTP/OAM) comparator. Configurable intermediate comparators (Ethernet/SNAP, 2x IP/UDP/ACH, and MPLS)

The following illustration shows a block diagram of the analyzer.

Figure 41 • Analyzer Block Diagram



The analyzer process is divided into engines and stages. Each engine represents a particular encapsulation stack that must be matched. There are up to six stages in each engine. Each stage uses a comparator block that looks for a particular protocol. The comparison is performed stage-by-stage until the entire frame header has been parsed.

Each engine has its own master enable so that it can be shut down for major reconfiguration, such as changes in encapsulation order, without stopping traffic. Other enabled engines are not affected.

The SOF detect block searches for the SFD in the preamble and uses that to indicate the SOF position. This information is carried along in the pipeline and also passed to the analyzer.

The first stage of the analyzer is a data path aligner that aligns the first byte of the packet (without the preamble & SFD) to byte 0 of the analyzer data path.

The encapsulation engine handles numerous types of encapsulation stacks. These can be broken down to their individual protocols, and a comparator is defined for each type. The order in which these are applied is configurable. Each comparator outputs a pattern/flow match bit and an offset to the start of the next protocol. The cumulative offset points to the time stamp field.

The sequence in which the protocol comparators are applied is determined by configuration registers associated with each comparator, and the transfer of parameters between comparators is controlled by the encapsulation engine controller.

It receives the pattern match and offset information from one comparator stage and feeds the start-of-protocol position to the next comparator. This continues until the entire encapsulation stack has been parsed and always ends with the PTP/OAM stage (or until a particular comparator stage cannot find a match in any of its flows). If at any point along the way no valid match is found in a particular stage, the analyzer sends the NOP communication to the time stamp block indicating that this frame does not need modification and that it should discard its time stamp.

There are two types of engines in the analyzer, one optimized for PTP frames and the other optimized for OAM frames. The two engine types are mostly identical, except that the IP comparators are removed from the OAM engines. The following table shows the comparator layout per engine type and the number of flows in each comparator. There are two PTP engines and one OAM engine in each analyzer. Additional differences in the Ethernet and MPLS blocks are defined in their respective sections. For more information, see Ethernet/SNAP/LLC Comparator, page 69 and MPLS Comparator, page 73.

Table 18 • **Flows Per Engine Type**

	Number of Flows	
Comparator	**PTP Engine**	**OAM Engine**
Ethernet 1	8	8
Ethernet 2	8	8
MPLS	8	8
IP/ACH 1	8	0
IP/ACH 2	8	0
PTP/OAM	6	6

Encapsulation matches can be set independently in each direction by setting the ANALYZER_MODE.SPLIT_ENCAP_FLOW register. However, strict and non-strict flow cannot be set independently for group A and group B of analyzer engine C.

Choice of strict flow or non-strict has to be made on each direction rather than on an engine by engine basis. Valid values for INGR_ENCAP_FLOW_ENA and EGR_ENCAP_FLOW_ENA are 3'b000 or 3'b111.

Each comparator stage has an offset register that points to the beginning of the next protocol relative to the start of the current one. The offset is in bytes, and the first byte of the current protocol counts as byte 0. As an example, the offset register for a stage would be programmed to 10 when the header to match is 10 bytes long. With the exception of the MPLS stage (offsets are automatically calculated in that stage), it is the responsibility of the programmer to determine the value to put in these registers. This value must be calculated based upon the expected length of the header, and is not expected to change from frame-to-frame when matching a given flow.

Table 19 • **Ethernet Comparator: Next Protocol**

Parameter	Width	Description
Encap_Engine_ENA	1 bit	For each encapsulation engine and enable bit that turns the engine on or off. The engine enables and disables either during IDLE (all 8 bytes must be IDLE) or at the end of a frame. If the enable bit is changed during the middle of a frame, the engine will wait until it sees either of those conditions before turning on or off.
Encap_Flow_Mode	1 bit	There is a separate bit for each engine. For each encapsulation engine: 1 = Strict flow matching, a valid frame must use the same flow IDs in all comparators in the engine except the PTP and MPLS comparators. 0 = A valid frame may match any enabled flow in all comparators If more than one encapsulation produces a match, the analyzer sends NOP to the rewriter and sets a sticky bit.

The following table shows the ID codes comparators use in the sequencing registers. The PTP packet target encapsulations require only up to five comparators.

Table 20 • **Comparator ID Codes**

ID	Name	Sequence
0	Ethernet Comparator 1	Must be the first
1	Ethernet Comparator 2	Intermediate
2	IP/UDP/ACH Comparator 1	Intermediate
3	IP/UDP/ACH Comparator 2	Intermediate
4	MPLS Comparator	Intermediate
5	PTP/OAM Comparator	Must be the last

The following sections describe the comparators. The frame format of each comparator type is described first, followed by match/mask parameter definition. All upper and lower bound ranges are inclusive and all match/mask registers work the same way. If the corresponding mask bit is 1, then the match bit is compared to the incoming frame. If a mask bit is 0, then the corresponding match bit is ignored (a wildcard).

3.6.15.1 Ethernet/SNAP/LLC Comparator

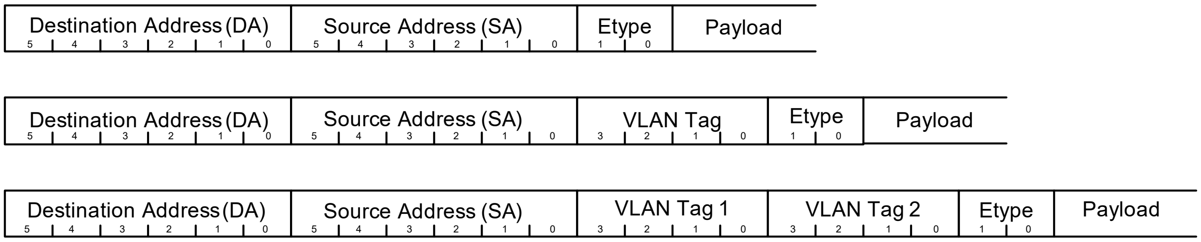
There are two such comparators in each engine. The first stage of each engine is always an Ethernet/SNAP/LLC comparator. The other comparator can be configured to be at any point in the chain.

Ethernet frames can have multiple formats. Frames that have an actual length value in the ether-type field (Ethernet type I) can have one of three formats: Ethernet with an EtherType (Ethernet type II), Ethernet with LLC, or Ethernet with LLC & SNAP. Each of these formats can be compounded by having one or two VLAN tags.

3.6.15.1.1 Type II Ethernet

Type II Ethernet is the most common and basic type of Ethernet frame. The Length/EtherType field contains an EtherType value and either 0, 1, or 2 VLAN tags. Both VLAN can be of type S/C (with EtherType 0x8a88/0x8100). The payload would be the start of the next protocol.

Figure 42 • **Type II Ethernet Basic Frame Format**

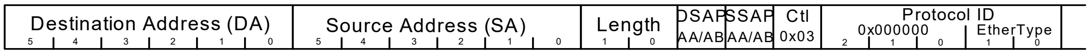


3.6.15.1.2 Ethernet with LLC and SNAP

If an Ethernet frame with LLC contains a SNAP header, it always follows a three-octet LLC header. The LLC values for DSAP & SSAP are either 0xAA or 0xAB and the control field contains 0x03. The SNAP header is five octets long and consists of two fields, the 3-octet OUI value and the 2-octet EtherType. As with the other types of Ethernet frames, this format can have 0, 1, or 2 VLAN tags. The OUI portion of the SNAP header is hard configured to be 0 or 0xf8.

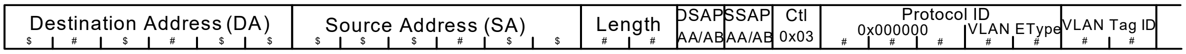
The following illustration shows an Ethernet frame with a length in the Length/EtherType field, an LLC header, and a SNAP header.

Figure 43 • **Ethernet Frame with SNAP**



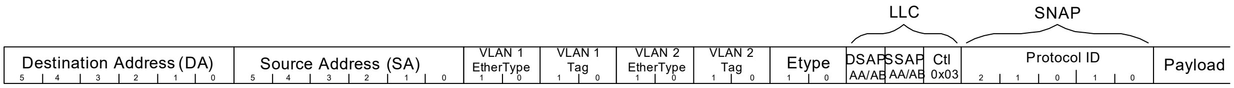
The following illustration shows an Ethernet frame with an LLC/SNAP header and a VLAN tag in the SNAP header. The Ethertype in the SNAP header is the VLAN identifier and tag immediately follows the SNAP header.

Figure 44 • **Ethernet Frame with VLAN Tag and SNAP**



The following illustration shows the longest form of the Ethernet frame header that needs to be supported: two VLAN tags, an LLC header, and a SNAP header.

Figure 45 • **Ethernet Frame with VLAN Tags and SNAP**



3.6.15.1.3 Provider Backbone Bridging (PBB) Support

The provider backbone bridging protocol is supported using two Ethernet comparator blocks back-to-back. The first portion of the frame has a type II Ethernet frame with either 0 or 1 VLAN tags followed by an I-tag. The following illustrations show two examples of the PBB Ethernet frame format.

Figure 46 • **PBB Ethernet Frame Format (No B-Tag)**

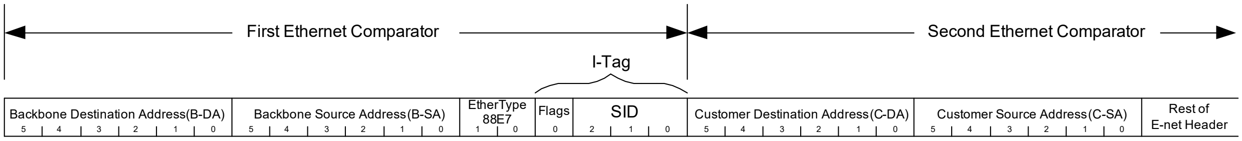
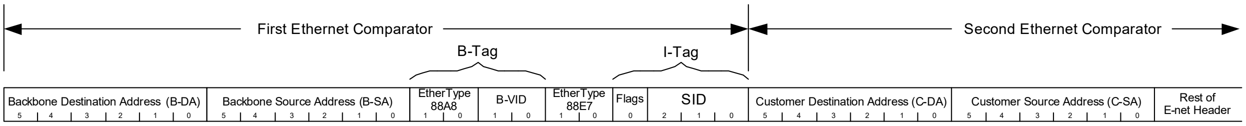


Figure 47 • **PBB Ethernet Frame Format (1 B-Tag)**



3.6.15.1.4 Ethernet Comparison

The Ethernet comparator block has two forms of comparison, as follows:

- Next protocol comparison is common for all flows in the comparator. It is the single set of registers and is used to verify what the next protocol in the encapsulated stack will be.
- Flow comparison is used to match any of the possible flows within the comparator.

3.6.15.1.5 Ethernet Next Protocol Comparison

The next protocol comparison field looks at the last EtherType field in the header (there can be multiple in the header) to verify the next protocol. It may also look at VLAN tags and the EtherType field when it is used as a length. Each has a pattern match/mask or range, and an offset.

The following table lists the next protocol parameters for the Ethernet comparator.

Table 21 • **Ethernet Comparator (Next Protocol)**

Parameter	Width	Description
Eth_Nxt_Comparator	3 bit	Pointer to the next comparator.
Eth_Frame_Sig_Offset	5 bit	Points to the start of the field used to build the frame signature.
Eth_VLAN-TPID_CFG	16 bit	Globally defines the value of the TPID for an S-tag, B-tag, or any other tag type other than a C-tag or I-tag.
Eth_PBB_ENA	1 bit	Configures if the packet carries PBB or not. This configuration bit is only present in the first Ethernet comparator block. PBB is disabled in Ethernet comparator block 2.
Eth_Etype_Match_Enable	1 bit	Configures if the Ethertype field match register is used or not. Only valid when the packet is a type II Ethernet packet.
Eth_Etype_Match	16 bit	If the packet is a type II Ethernet packet and Eth_Etype_Match_Enable is a 1, the Ethertype field in the packet is compared against this value.

3.6.15.1.6 Ethernet Flow Comparison

The Ethernet flow is determined by looking at VLAN tags and either the source address (SA) or the destination address (DA). There are a configurable number of these matched sets. The following table lists the flow parameters for the Ethernet comparator.

Table 22 • **Ethernet Comparator (Flow)**

Parameter	Width	Description
Eth_Flow_Enable	1 bit/flow	0 = Flow disabled 1 = Flow enabled
Eth_Channel_Mask	1 bit/channel/flow	0 = Do not use this flow match group for this channel 1 = Use this flow match group for this channel
Eth_VLAN_Tags	2 bit	Configures the number of VLAN tags in the frame (0, 1, or 2)
Eth_VLAN_Tag1_Type	1 bit	Configures the VLAN tag type for VLAN tag 1 If PBB is not enabled: 0 = C-tag, value of 0x8100 1 = S-tag, match to the value in CONF_VLAN_TPID (global for all ports/directions) If PBB enabled: 0 = S-tag (or B-tag), to the value in CONF_VLAN_TPID (global for all ports/directions) There must be 2 VLAN tags, 1 S-tag and one I-tag 1 = I-tag

Table 22 • **Ethernet Comparator (Flow) *(continued)***

Parameter	Width	Description
Eth_VLAN_Tag2_Type	1 bit	Configures the VLAN tag type for VLAN tag 2 If PBB is not enabled: 0 = C-tag, value of 0x8100 1 = S-tag, match to the value in CONF_VLAN_TPID (global for all ports/directions) If PBB enabled: The second tag is always an I-tag and this register control bit is not used. The second tag in PBB is always an I-tag.
Eth_Ethertype_Mode	1 bit	0 = Only type 2 Ethernet frames supported, no SNAP/LLC expected 1 = Type 1 & 2 Ethernet packets supported. Logic looks at the Ethertype/length field to determine the packet type. If the field is a length (less than 0x0600), then the packet is a type 1 packet and MUST include a SNAP & 3-byte LLC header. If the field is not a length, it is assumed to be an Ethertype and SNAP/LLC must not be present
Eth_VLAN_Verify_Ena	1 bit	0 = Parse for presence of VLAN tags but do not check the values. For PBB mode, the I-tag is still always checked. 1 = Verify the VLAN tag configuration including number and value of the tags.
Eth_VLAN_Tag_Mode	2 bit	0 = No range checking on either VLAN tag 1 = Range checking on VLAN tag 1 2 = Range checking on VLAN tag 2
Eth_Addr_Match	48 bit	Matches an address field selected by Eth_Addr_Match_Mode
Eth_Addr_Match_Select	2 bit	Selects the address to match 0 = Match the destination address 1 = Match the source address 2 = Match either the source or destination address 3 = Reserved, do not use
Eth_Addr_Match_Mode	3 bits per flow	Selects the address match mode. One or multiple bits can be set in this mode register allowing any combination of match types. For unicast or multicast modes, only the MSB of the address field is checked (0 = unicast; 1 = multicast). See section 3.2.3.1 of 802.3 for more details. 0 = Match the full 48-bit address 1 = Match any unicast address 2 = Match any multicast address
Eth_VLAN_Tag1_Match	12 bit	Match field for the first VLAN tag (if configured to be present).
Eth_VLAN_Tag1_Mask	12 bit	Mask for the first VLAN tag. If a match set is not used, set this register to all 0s.
Eth_VLAN_Tag2_Match	12 bit	Match field for the update VLAN tag (if configured to be present).
Eth_VLAN_Tag2_Mask	12 bit	Mask for the second VLAN tag. If a match set is not used, set this register to all 0s.

Table 22 • Ethernet Comparator (Flow) *(continued)*

Parameter	Width	Description
Eth_VLAN_Tag_Range_Upper	12 bit	Upper limit of the range for one of the VLAN fields selected by ETH_VLAN_TAG_MODE register. If PBB mode is enabled, this register is not used for range checking but rather is the upper 12 bit of the I-tag.
Eth_VLAN_Tag_Range_Lower	12 bit	Lower limit of the range for one of the VLAN fields selected by ETH_VLAN_TAG_MODE register. If PBB mode is enabled, this register is not used for range checking but rather is the lower 12 bit of the I-tag SID.
Eth_Nxt_Prot_Grp_Sel	1 bit	Per flow, maps a particular flow to a next-protocol group register set. This register only appears in the Ethernet block in the OAM-optimized engine.

If the Ethernet block is part of the OAM optimized engine, there are two sets of next-protocol configuration registers. Both sets are identical, except one has an _A suffix and the other has a _B suffix. In the per-flow registers, an additional register, ETH_NXT_PROT_SEL, is included to map a particular flow with a set of next protocol register set. This function allows the Ethernet block within the OAM-optimized engine to act like two separate engines with a configurable number of flows assignable to each (with a total maximum number of eight flows). It effectively allows two separate protocol encapsulation stacks to be handled within the engine.

3.6.15.2 MPLS Comparator

The MPLS comparator block counts MPLS labels to find the start of the next protocol. The MPLS header can have anywhere from 1 to 4 labels. Each label is 32 bit long and has the format shown in the following illustration.

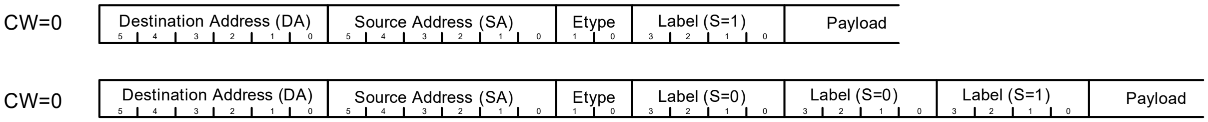
Figure 48 • MPLS Label Format

The S bit is used to indicate the last label in the stack, as follows: If S = 0, then there is another label. If S = 1, then this is the last label in the stack.

Also, the MPLS stack can optionally be followed by a control word (CW). This is configurable per flow.

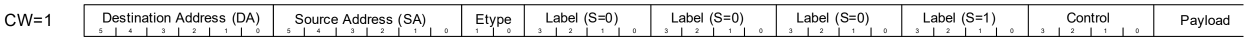
The following illustration shows a simple Ethernet packet with either one label or three labels and no control word.

Figure 49 • MPLS Label Stack within an Ethernet Frame



The following illustration shows an Ethernet frame with four labels and a control word. Keep in mind that this comparator is used to compare the MPLS labels and control words; the Ethernet portion is checked in the first stage.

Figure 50 • MPLS Labels and Control Word



There could be VLAN tags between the SA and the Etype fields and, potentially, a LLC and SNAP header before the MPLS stack, but these would be handled in the Ethernet/LLC/SNAP comparator.

The only configuration registers that apply to all flows within the comparator are the match_mode register and the nxt_comparator register. The match mode register determines how the match filters are used, and there is one per stage. Each flow has it own complete set of match registers.

Table 23 • **MPLS Comparator: Next Word**

Parameter	Width	Description
MPLS_Nxt_Comparator	3 bit	Pointer to the next comparator

Table 24 • **MPLS Comparator: Per-Flow**

<table>
<tr><th>Parameter</th><th>Width</th><th>Description</th></tr>
<tr><td>MPLS_Flow_Enable</td><td>1 bit per flow</td><td>0 = Flow disabled
1 = Flow enabled</td></tr>
<tr><td>MPLS_Channel_Mask</td><td>1 bit per channel per flow</td><td>0 = Do not use this flow match group for this channel
1 = Use this flow match group for this channel</td></tr>
<tr><td>MPLS_Ctl_Word</td><td>1 bit</td><td>Indicates if there is a 32-bit control word after the last label. This should only be set if the control word is not expected to be an ACH header. ACH headers are checked in the IP block. If the control word is a non-ACH control word, only the upper 4 bits of the control are checked and are expected to be 0.
0 = There is no control word after the last label
1 = There is expected to be a control word after the last label</td></tr>
<tr><td>MPLS_REF_PNT</td><td>1 bit</td><td>The MPLS comparator implements a searching algorithm to properly parse the MPLS header. The search can be performed from either the top of the stack or the end of the stack.
0 = All searching is performed starting from the top of the stack
1 = All searching if performed from the end of the stack</td></tr>
<tr><td>MPLS_STACK_DEPTH</td><td>4 bit</td><td>Each bit represents a possible stack depth, as shown in the following list.<table><tr><th>MPLS_STACK_DEPTH Bit</th><th>Allowed Stack Depth</th></tr><tr><td>0</td><td>1</td></tr><tr><td>1</td><td>2</td></tr><tr><td>2</td><td>3</td></tr><tr><td>3</td><td>4</td></tr></table></td></tr>
</table>

Table 25 • **MPLS Range_Upper/Lower Label Map**

Parameter	MPLS_REF_PNT = 0, top-of-stack referenced	MPLS_REF_PNT=1, end-of-stack referenced
MPLS_Range_Upper/Lower_0	Top label	Third label before the end label
MPLS_Range_Upper/Lower_1	First label after the top label	Second label before the end label
MPLS_Range_Upper/Lower_2	Second label after the top label	First label before the end label
MPLS_Range_Upper/Lower_3	Third label after the top label	End label

The offset to the next protocol is calculated automatically. It is based upon the number of labels found, and whether a control word is configured to be present. It points to the first octet after the last label or after the control word, if present.

Table 26 • **Next MPLS Comparator**

Parameter	Width	Description
MPLS_Range_Lower	20 bit × 4 labels	Lower value of the label range when range checking is enabled
MPLS_Range_Upper	20 bit × 4 labels	Upper value of the label range when range checking is enabled

If an exact label match is desired, set the upper and lower range values to the same value. If a label value is a don't care, then set the upper value to the maximum value and the lower value to 0.

The MPLS comparator block used in the OAM-optimized engine differs from the one used in the PTP-optimized engine.

Just like the Ethernet comparator block, there are two sets of next protocol blocks along with a next protocol association configuration field per-flow. This allows two different encapsulations to occur in a single engine.

Table 27 • **Next-Protocol Registers in OAM-Version of MPLS Block**

Parameter	Width	Description
MPLS_Nxt_Prot_Grp_Sel	1 bit per flow	Maps each flow to next-protocol-register set A or B

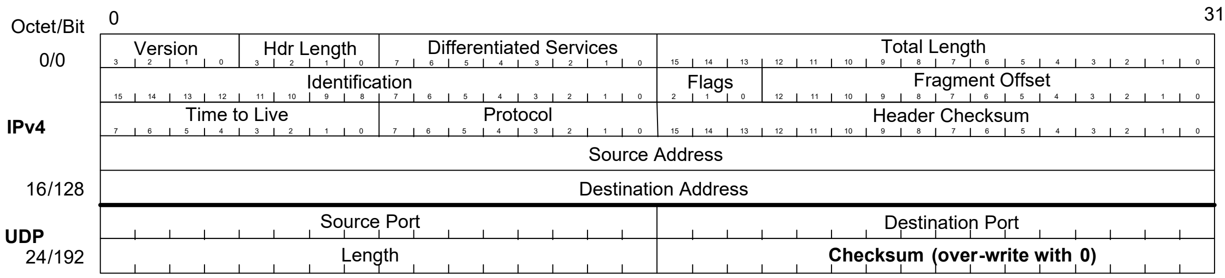
3.6.15.3 IP/UDP/ACH Comparator

The IP/UDP/ACH comparator is used to verify one of three possible formats: IPv4, IPV6, and ACH. Additionally, IPv4 and IPv6 can also have a UDP header after the IP header. There are two of these comparators and they can operate at stages 2, 3, or 4 of the analyzer pipeline. Note that if there is an IP-in-IP encapsulation, a UDP header will only exist with the inner encapsulation.

3.6.15.4 IPv4 Header Format

The following illustration shows an IPv4 frame header followed immediately by a UDP header. IPv4 does not always have the UDP header, but the comparator is designed to work with or without it. The Header Length field is used to verify the offset to the next protocol. It is a count of 32-bit words and does not include the UDP header. If the IPv4 frame contains a UDP header, the Source and Destination ports are also checked. These values are the same for all flows within the comparator. Note that IPv4 options, extended headers, and UDP fragments are not supported.

Figure 51 • **IPv4 with UDP**

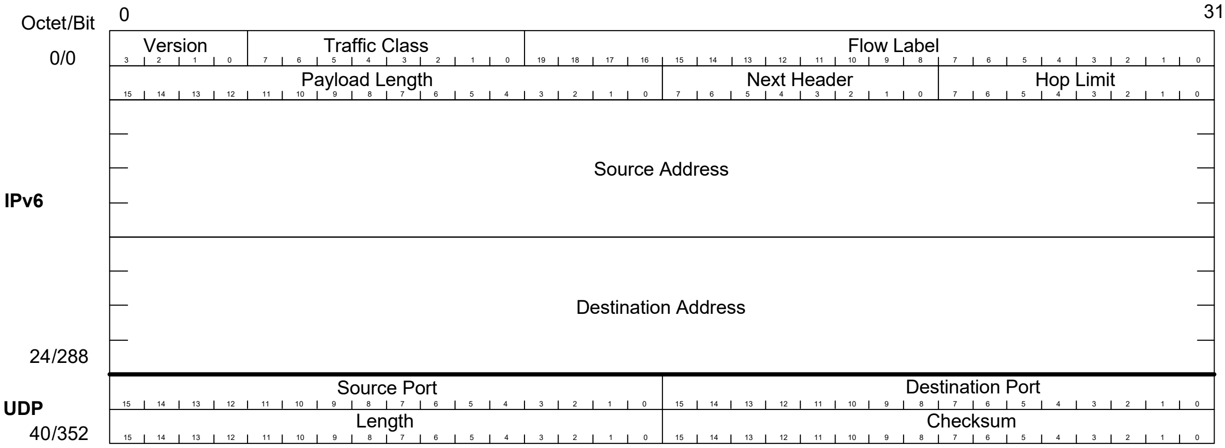


Per flow validation is performed on the Source or Destination Address in the IPv4 header. The comparator can be configured to indicate a match in the flow if the source, destination, or either the source or destination fields match.

3.6.15.5 IPv6 Header Format

The following illustration shows an IPv6 frame header followed immediately by a UDP header. IPv6 does not always have the UDP header, but the comparator is designed to work with or without it. The Next Header field is used to verify the offset to the next protocol. It is a count of 32-bit words and does not include the UDP header. If the IPv6 frame contains a UDP header, the Source and Destination ports are also checked. These values are the same for all flows within the comparator.

Figure 52 • **IPv6 with UDP**



Per flow validation is performed on the Source or Destination Address in the IPv6 header. The comparator can be configured to indicate a match in the flow if the source, destination, or either the source or destination fields match.

If the IPv6 frame is the inner most IP protocol, then the checksum field must be valid. This is accomplished using a pair of pad bytes after the PTP frame. The checksum is computed using one's compliment of the one's compliment sum of the IPv6 header, UDP header, and payload including the pad bytes. If any of the fields in the frame are updated, the pad byte field must be updated so that the checksum field does not have to be modified.

Note: IPv6 extension headers are not supported.

3.6.15.6 ACH Header Format

The following illustrations show ACH headers. They can appear after a MPLS label stack in place of the control word. ACH is verified as a protocol only. There are no flows within the protocol for ACH. The ACH header can optionally have a Protocol ID field. The protocol is verified using the Version, Channel Type, and optional Protocol ID field.

Figure 53 • **ACH Header Format**

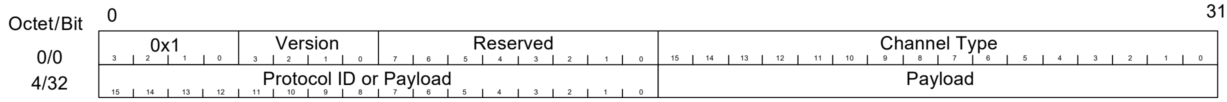
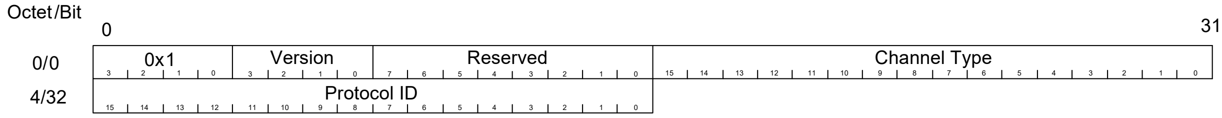


Figure 54 • **ACH Header with Protocol ID Field**



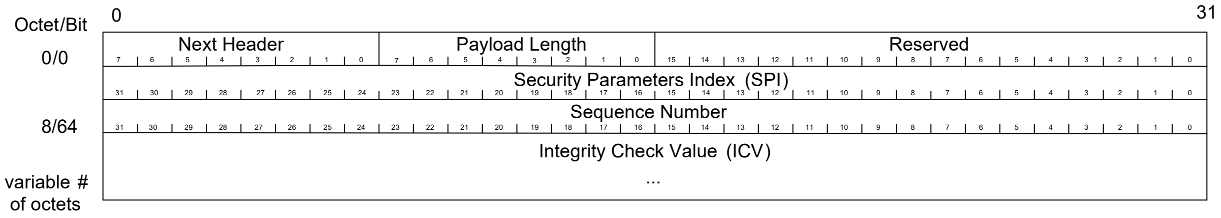
3.6.15.7 IPSec

IPSec adds security to the IP frame using an Integrity Check Value (ICV), a variable-length checksum that is encoded with a special key. The key value is known by the sender and the receiver, but not any of the devices in between. A frame must have a correct ICV to be valid. The sequence number field is a continuously incrementing value that is used to prevent replay attacks (resending a known good frame).

Little can be done with frames when IPSec is used, because the IEEE 1588 block cannot recalculate the ICV and the frame cannot be modified on egress. Therefore, one-step processing cannot be performed-only two-step processing can be done. The only task here is to verify the presence of the protocol header. Stored time stamps in the TS FIFO are used to create follow-up messages. On ingress, the time stamp can be added to the PTP frame by writing it into the reserved bytes or by overwriting the CRC with it and appending a new CRC. The CPU must know how to handle these cases correctly.

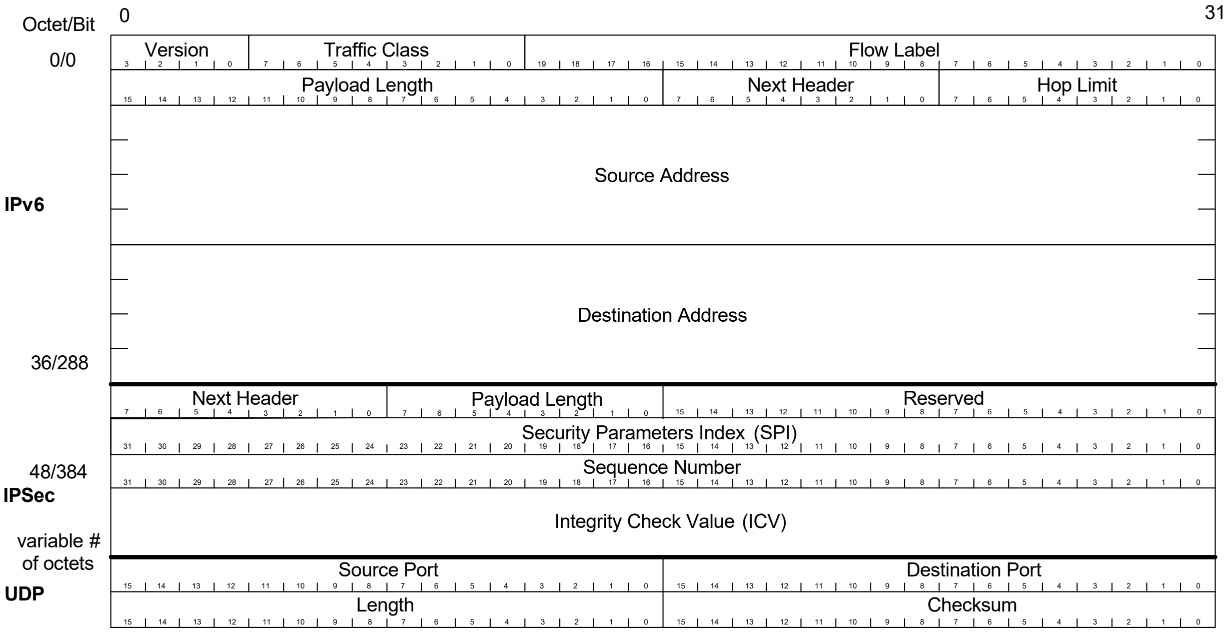
The following illustration shows the format of the IPSec frame. It normally appears between the IP header (IPv4 or IPv6) and the UDP header or at the start of the payload.

Figure 55 • IPSec Header Format



There is only one set of match/mask registers associated with IPSec, and they are used to verify the presence of the IPSec header. The following illustration shows the largest possible IP frame header with IPv6, IPSec, and UDP.

Figure 56 • IPv6 with UDP and IPSec



3.6.15.8 Comparator Field Summary

The following table shows a summary of the fields and widths to verify IPv4, IPv6, and ACH protocols.

Table 28 • **Comparator Field Summary**

Protocol	Next Protocol Fields	NPF Bit Widths	Flow Fields	Flow Bit Widths
IPv4	Header length	One 4-bit field	Source/ Destination Address	One 32-bit field
	UDP Source/Destination Port	One 32-bit field		
IPv6	Next header	One 8-bit field	Source/ Destination Address	One 128-bit field
	UDP Source/Destination Port	One 32-bit field		
ACH	Entire ACH header	One 64-bit field		
IPSec	Next Header/Payload Length/ SPI	One 64-bit field		

3.6.15.8.1 IP/ACH Comparator Next Protocol

The following table shows the registers used to verify the current header protocol and the next protocol. They are universal, and cover IPv4, IPv6, and ACH. They can also be used to verify other future protocols.

Table 29 • **IP/ACH Next-Protocol Comparison**

Parameter	Width	Description
IP_Mode	2 bit	Specifies the mode of the comparator. If IPv4 or IPv6 is selected, the version field is automatically checked to be either 4 or 6 respectively. If another protocol mode is selected, then the version field is not automatically checked. In IPv4, the fragment offset field must be 0, and the MF flag bit (LSB of the flag field) must be 0. 0 = IPv4 1 = IPv6 2 = Other protocol, 32-bit address match 3 = Other protocol, 128-bit address match
IP_Prot_Match_1	8 bit	Match bit for Protocol field in IPv4 or next header field in IPv6
IP_Prot_Mask_1	8 bit	Mask bits for IP_Prot_Match_1. For each bit, if it is a 1, the corresponding match bit is valid. If it is 0, the corresponding match bit is ignored. Disable this match/mask set by setting the mask register to all 0’s.
IP_Prot_Offset_1	5 bit	Indicates the starting position relative to the beginning of the IP frame header to start matching for the match/mask 1 register pair.
IP_Prot_Match_2	64 bit	Match bits for the IPSec header or any other desired field. For ACH, this register should be used to match the ACH header.
IP_Prot_Mask_2	64 bit	Mask bits for IP_Prot_Match_2. For each bit, if it is a 1, the corresponding match bit is valid. If it is 0, the corresponding match bit is ignored. Disable this match/mask set by setting the mask register to all 0’s.
IP_Prot_Offset_2	7 bit	Indicates the starting position relative to the beginning of the IP frame header to start matching for the match/mask two-register pair.

Table 29 • IP/ACH Next-Protocol Comparison *(continued)*

Parameter	Width	Description
IP_Nxt_Protocol	8 bit	Points to the start of the next protocol relative to the beginning of this header. It is the responsibility of the programmer to determine this offset, it is not calculated automatically. Each flow within an encapsulation engine must have the same encapsulation order and each header must be the same length. This field is current protocol header length in bytes.
IP_Nxt_Comparator	3 bit	Pointer to the next comparator. 0 = Reserved 1 = Ethernet comparator 2 2 = IP/UDP/ACH comparator 1 3 = IP/UDP/ACH comparator 2 4 = Reserved 5 = PTP/OAM comparator 6,7 = Reserved
IP_Flow_Offset	5 bit	Indicates the starting position relative to the beginning of the IP frame header to start matching for the flow match/mask register pair. When used with IPv4 or 6, this will point to the first byte of the source address. When used with a protocol other that IPv4 or 6, this register points to the beginning of the field that will be used for flow matching.
IP_UDP_Checksum_Clear_Ena	1 bit	If set, the 2-byte UDP checksum should be cleared (written with zeroes). This would only be used for UDP in IPv4.
IP_UDP_Checksum_Update_Ena	1 bit	If set, the last two bytes in the UDP frame must be updated to reflect changes in the PTP or OAM frame. This is necessary to preserve the validity of the IPv6 UDP checksum. Note that IP_UDP_Checksum_Clear_Ena & IP_UDP_Checksum_Update_Ena should never be set at the same time.
IP_UDP_Checksum_Offset	8 bit	This configuration field is only used if the protocol is IPv4. This register points to the location of the UDP checksum relative to the start of this header. This info is used later by the PTP/Y.1731 block to inform the rewriter of the location of the checksum in a UDP frame. This is normally right after the Log Message Interval field.
IP_UDP_Checksum_Width	2 bit	Specifies the length of the UDP checksum in bytes (normally 2 bytes)

The IP/ACH Comparator Flow Verification registers are used to verify the current frame against a particular flow within the engine. When this engine is used to verify IPv4 or IPv6 protocol, the flow is verified using either the source or destination address in the frame.

If the protocol is something other than IPv4 or IPv6, then the flow match can be used to match either a 32 or 128 bit field pointed to by the IP_Flow_Offset register. Mask bits can be used to shorten the length of the match, but there is no concept of source or destination address in this mode.

Table 30 • IP/ACH Comparator Flow Verification Registers

Parameter	Width	Description
IP_Flow_Ena	1 bit per flow	0 = Flow disabled 1 = Flow enabled

Table 30 • IP/ACH Comparator Flow Verification Registers *(continued)*

Parameter	Width	Description
IP_Flow_Match_Mode	2 bit per flow	This register is only valid when the comparator block is configured to match on IPv4 or IPv6. It allows the match to be performed on the source address, destination address, or either address. 0 = Match on the source address 1 = Match on the destination address 2 = Match on either the source or the destination address
IP_Flow_Match	128 bit	Match bits for source & destination address in IPv4 & 6. Also used as the flow match for protocols other than IPv4 or 6. When used with IPv4, only the upper 32 bits are used and the remaining bits are not used.
IP_Flow_Mask	128 bit	Mask bits for IP_Flow_Match. For each bit, if it is a 1, the corresponding match bit is valid. If it is 0, the corresponding match bit is ignored.
IP_Channel_Mask	1 bit per channel per flow	Enable for this match set for this channel
IP_Frame_Sig_Offset	5 bit	Points to the start of the field that will be used to build the frame signature. This register is only present in comparators where frame signature is supported. In other words, if there is no frame signature FIFO in a particular direction, this register will be removed.

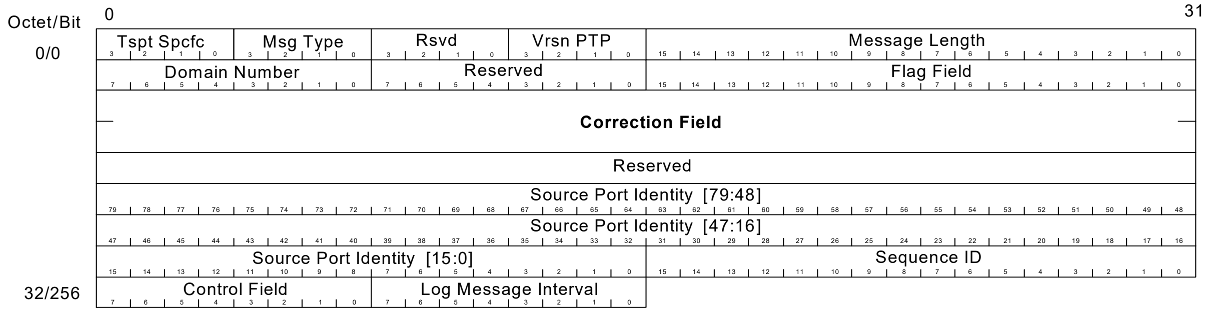
3.6.15.9 PTP/OAM Comparator

The PTP/OAM comparator is always the last stage in the analyzer for each encapsulation engine. It can validate IEEE 1588 PTP frames or OAM frames.

3.6.15.10 PTP Frame Header

The following illustration shows the header of a PTP frame.

Figure 57 • PTP Frame Layout

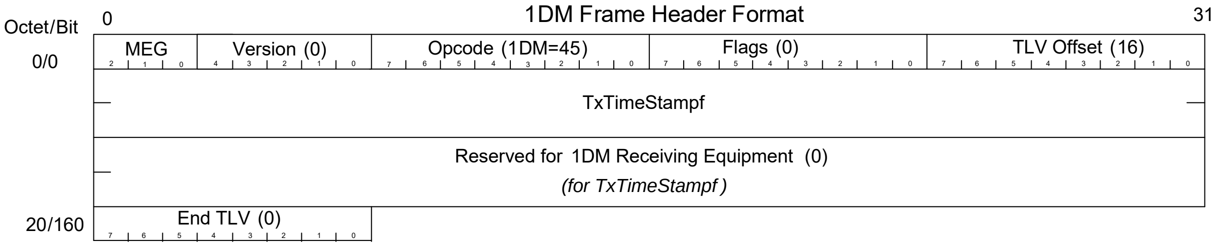


Unlike most of the other stages, there is no protocol validation for PTP frames; only interpretation of the header to determine what action to take. The first eight bytes of the header are used to determine the action to be taken. These match fields in the flow comparison registers with a corresponding set of command registers for each flow.

3.6.15.11 Y.1731 OAM Frame Header

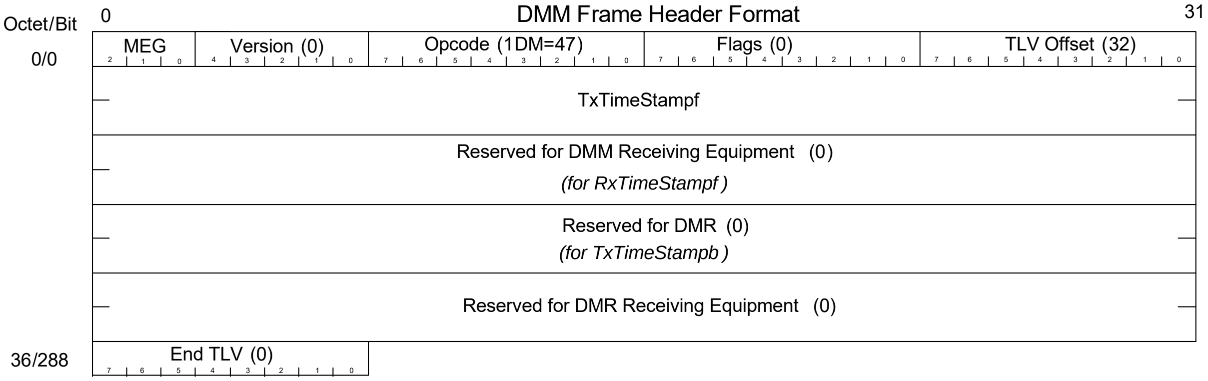
1DM, DMM, and DMR are the three supported Y.1731 frame headers. The following illustration shows the header part of a 1DM Y.1731 OAM frame.

Figure 58 • **OAM 1DM Frame Header Format**



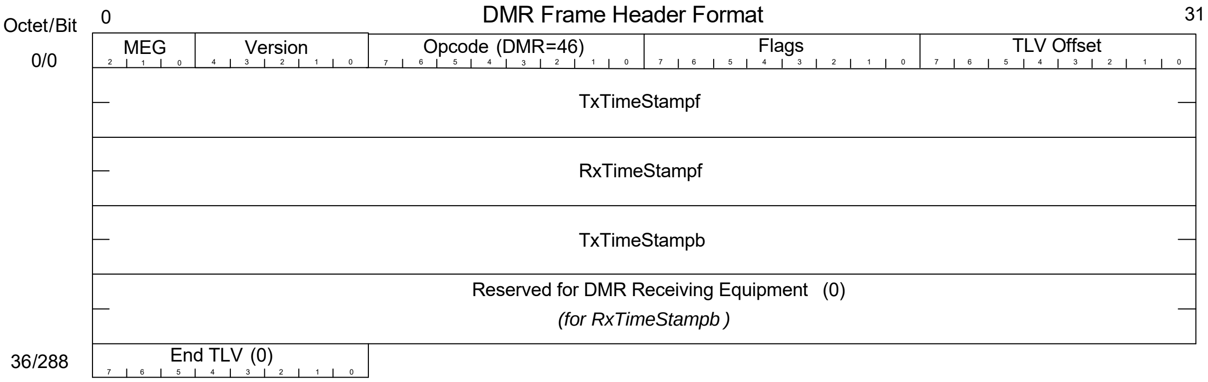
The following illustration shows a DMM frame header.

Figure 59 • **OAM DMM Frame Header Format**



The following illustration shows a DMR frame header.

Figure 60 • **OAM DMR Frame Header Format**



As with PTP, there is no protocol validation for Y.1731 frames; only interpretation of the header to determine what action to take. The first four bytes of the header are used to determine the action to be taken.

3.6.15.12 Y.1731 OAM PDU

1DM, DMM, and DMR are the three supported G.8113.1 PDUs and DMM/DMR are the two supported RFC6374 PDUs. The following illustrations show the PDU formats.

Figure 61 • RFC6374 DMM/DMR OAM PDU Format

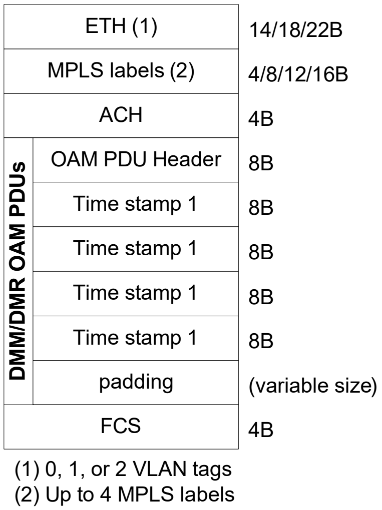
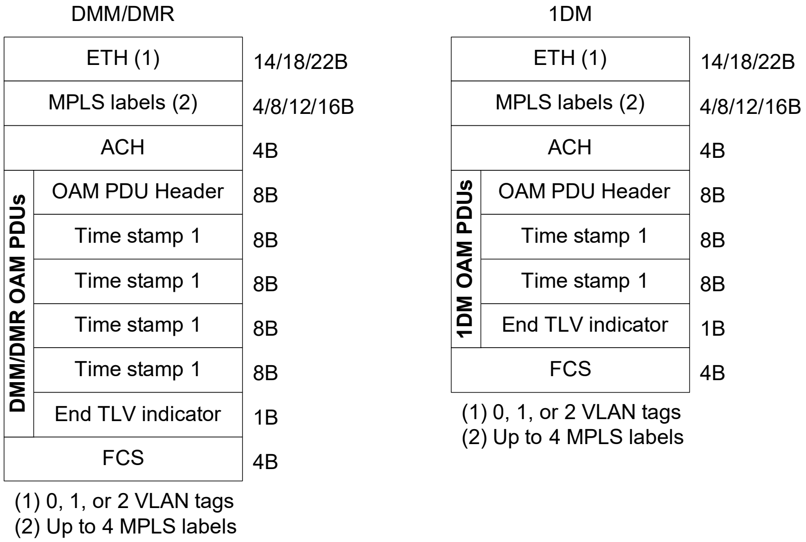


Figure 62 • G8113.1/draft-bhh DMM/DMR/1DM OAM PDU Format



As with PTP, there is no protocol validation for MPLS OAM; only interpretation of the header to determine what action to take. The first four bytes of the header are used to determine the action to be taken.

3.6.15.13 PTP Comparator Action Control Registers

The following registers perform matching on the frame header and define what action is to be taken based upon the match. There is one mask register for all flows, and the rest of the registers are unique for each flow.

Table 31 • PTP Comparison

Parameter	Width	Description
PTP_Flow_Match	64 bit	Matches bits in the PTP/Y.1731 frame starting at the beginning of the protocol header
PTP_Flow_Mask	64 bit	Mask bits for PTP_Flow_Match
PTP_Domain_Range_Lower	8 bit	Lower range of the domain field to match
PTP_Domain_Range_Upper	8 bit	Upper range of the domain field to match

Table 31 • PTP Comparison *(continued)*

Parameter	Width	Description		
PTP_Domain_Range_Enable	1 bit	Enable for range checking		
PTP_Domain_Offset	5 bit	Pointer to the domain field, or whatever field is to be used for range checking		
PTP_Action_Command	3 bit	**Command Value**	**Mnemonic**	**Action**
		0	NOP	Do nothing
		1	SUB	New correction field = Current correction field – Captured local time
		2	SUB_P2P	New correction field = Current correction field – Local latency + path_delay
		3	ADD	New correction field = Current correction field + Captured local time
		4	SUB_ADD	New correction field = Current correction field + (Captured local time + Local latency – Time storage field)
		5	WRITE_1588	Write captured local time to time storage field
		6	WRITE_P2P	Active_timestamp_ns = captured local time and path_delay written to time storage field and correction field (deprecated command)
		7	WRITE_NS	Write local time in nanoseconds to the new field
		8	WRITE_NS_P2P	Write local time in nanoseconds + p2p_delay to the new field and correction field
PTP_Save_Local_Time	1 bit	When set, saves the local time to the time stamp FIFO (only valid for egress ports).		
PTP_Correction_Field_Offset	5 bit	Points to the location of the correction field. Location is relative to the first byte of the PTP/OAM header.		
PTP_Time_Storage_Field_Offset	6 bit	Points to a location in a PTP frame where a time value can be stored or read.		
PTP_Add_Delay_Asymmetry_Enable	1 bit	When enabled, the value in the delay asymmetry register is added to the correction field of the frame.		
PTP_Subtract_Delay_Asymmetry_Enable	1 bit	When enabled, the value in the delay asymmetry register is subtracted from the correction field of the frame.		
PTP_Zero_Field_Offset	6 bit	Points to a location in the PTP/OAM frame to be zeroed if this function is enabled		
PTP_Zero_Field_Byte_Count	4 bit	The number of bytes to be zeroed. If this field is 0, then this function is not enabled.		

Table 31 • **PTP Comparison *(continued)***

Parameter	Width	Description
PTP_Modified_Frame_Byte_Offset	3 bit	Indicates the position relative to the start of the PTP frame in bytes where the Modified_Frame_Status bit resides. This value is also used to calculate the offset from the beginning of the Ethernet packet to this field for use by the Rewriter.
PTP_Modified_Frame_Status_Update	1 bit	If set, tells the rewriter to update the value of this bit. Configuration registers inside the rewriter indicate if the bit will be set to 0 or 1.
PTP_Rewrite_Bytes	4 bits	Number of bytes in the PTP or OAM frame that must be modified by the Rewriter for the time stamp
PTP_Rewrite_Offset	8 bits	Points to where in the frame relative to the SFD that the time stamp should be updated
PTP_New_CF_Loc	8 bits	Location where the updated correction field value is written relative to the PTP header start
PTP_Channel_Mask	1 bit per channel per flow	Enable for this match set for this channel
PTP_Flow_Enable	1 bit	When set, the fields associated with this flow are all valid

The following table shows controls that are common to all flows.

Table 32 • **PTP Comparison: Common Controls**

Parameter	Width	Description
PTP_IP_CHKSUM_Sel	1 bit	0 = Use IP checksum controls from comparator 1 1 = Use IP checksum controls from comparator 2
FSB_Adr_Sel	2 bits	Selects the source of the address for use in the frame signature builder

The following table shows the one addition, per-flow, register.

Table 33 • **PTP Comparison: Additions for OAM-Optimized Engine**

Parameter	Width	Description
PTP_NXT_Prot_Group_Mask	2 bits	There are two bits for each flow. Each bit indicates if the flow can be associated with next-protocol group A or B. One or both bits may be set. If a bit is 1 for a particular next-protocol group, then a flow match is valid if the prior comparator stages also produced matches with the same next-protocol group.

3.6.15.14 Future Protocol Compatibility

Except for MPLS, the comparators are not hardwired to their intended protocols. They can be used as generic field and range comparators because all of the offsets or pointers to the beginning of the fields are configurable. The IP comparator is the most generic and would probably be the first choice for validating a new protocol.

Additionally, if there are not enough comparison resources in a single comparator block to handle a new protocol, two comparators back-to-back can used by splitting up the comparison work. One portion can be validated in one comparator and then handed off to another. The only restriction is that there must be at least one 64-bit word of separation between the start of the protocol and where the second starts to operate.

3.6.15.15 Reconfiguration

There are three ways to perform reconfiguration:

1. Disable an entire encapsulation engine.
   Once an engine has been disabled, any of the configuration registers associated with it may be modified in any order. If other encapsulation engines are still active, they will still operate normally.
2. Disable a flow in an active engine.
   Each stage in the engine has an enable bit for each flow. If a flow is disabled in a stage, its registers may be modified. Once reconfiguration for a flow in a stage is complete, it can be enabled.
3. Disable a comparator.
   Each comparator within the active encapsulation engine can be disabled. If an Ethernet header according to the configuration Type I or Type II with SNAP/LLC is not found then subsequent flows will not be matched. The ETH1 comparator can also be disabled so that all frames flowing through the IEEE 1588 block are time stamped.

The disabling of engines and flows is always done in a clean manner so that partial matches do not occur. Flows and engines are always enabled or disabled during inter-packet gaps or at the end of a packet. This guarantees that when a new packet is received that it will be analyzed cleanly.

If strict flow matching is enabled and a flow is disabled in one of the stages, then the entire flow is automatically disabled.

If any register in a stage that applies to all flows needs to be modified, then the entire encapsulation engine must be disabled.

3.6.15.16 Frame Signature Builder

Along with time stamp and CRC updates, the analyzer outputs a frame signature that can be stored in the time stamp FIFO to help match frames with other info in the FIFO. This information is used by the CPU so that it can match time stamps in the time stamp FIFO with actual frames. The frame signature is up to 16 bytes long and contains information from the Ethernet header (SA or DA), IP header (SA or DA), and from the PTP or OAM frame. The frame signature is only used in the egress direction.

The PTP block contains a set of mapping registers to configure which bytes are mapped into the frame signature. The following tables show the mapping for each byte.

Table 34 • **Frame Signature Byte Mapping**

Select	Source Byte
0-23	PTP header byte number = (31-select)
24	PTP header byte number 6
25	PTP header byte number 4
26	PTP header byte number 0
27	Reserved
28-35	Selected address byte (select-28)

Table 35 • **Frame Signature Address Source**

Parameter	Width	Description
FSB_Map_Reg_0-15	6 bits	For each byte of the frame signature, use Table 34, page 85 to select which available byte is used. Frame signature byte 0 is the LSB. If not all 16 bytes are needed, the frame signature should be packed towards the LSB and the upper unused byte configuration values do not need to be programmed.

Table 35 • Frame Signature Address Source *(continued)*

Parameter	Width	Description	
FSB_Adr_Sel	2 bits	Selects the source of the address for use in the frame signature builder according to the following list	
		Select Value	**Address Source**
		0	Ethernet block 1
		1	Ethernet block 2
		2	IP block 1
		3	IP block 2

Configuration registers in each comparator block supply an address to select if it is the source address or the destination address.

A frame signature can be extracted from frames matching in all the three engines. The frame signature address selection is limited to Ethernet Block1 because only a limited number of encapsulations are supported in the third engine, Engine C.

Engine C has two parts: part A and part B. Part A supports ETH1, ETH2, MPLS protocols while part B supports only ETH1 protocol. Selection of Ethernet block 1 or 2 is dependent on whether part A flow matches or part B flow matches.

If a frame matches part A flow configuration, then the frame signature as configured in ETH1_NXT_PROTOCOL_A and ETH2_NXT_PROTOCOL_A using FSB_ADR_SEL will be considered in computing the frame signature.

If a frame matches part B flow configuration, then the frame signature as configured in ETH1_NXT_PROTOCOL_A and FSB_ADR_SEL will be considered in computing the frame signature. In this configuration if FSB_ADR_SEL is set to 1, to select ETH2 then all zeros are padded as frame signature because ETH2 is not supported by part B.

3.6.15.17 Configuration Sharing

The analyzer configuration services both channels. Each flow within each comparator has a channel-mask register that indicates which channels the flow is valid for. Each flow can be valid for channel A, channel B, or both channels.

A total of eight flows can be allocated the two channels if the analyzer configuration cannot be shared. They can each have four distinct flows (or three for the one, and five for the other, etc.).

3.6.15.18 OAM-Optimized Engine

The OAM optimized engine, Engine C, supports a fewer set of encapsulations such as ETH1, ETH2, MPLS, and ACH. Engine C is was enhanced with an ACH comparator to support the MPLS-TP OAM protocol. The MPLS-TP OAM protocol for Engine C is configured in the following registers.

- EGR2_ACH_PROT_MATCH_UPPER/LOWER_A
- EGR2_ACH_PROT_MASK_UPPER/LOWER_A
- EGR2_ACH_PROT_OFFSET_A

The ACH comparator will start the comparison operation right after the MPLS comparator.

In addition to the descriptions of the Ethernet and MPLS blocks in the OAM optimized engine, there is the notion of protocol-A/protocol-B. When a match occurs in the Ethernet 1 block the status of the protocol set that produced the match is indicated. There are two bits, one for protocol A and another for protocol B. If both sets produce a match, then both bits are set.

These bits are then carried to the next comparison block and only allow flow matches for the protocol sets that produced matches in the prior block. This block also produces a set of protocol match bits that are also carried forward.

This feature is provided to prevent a match with protocol set A in the first block and protocol set B in the second block.

3.6.16 Time Stamp Processor

The primary function of the time stamp processor block is to generate a new Timestamp_field or new Correction_field (Transparent clocks) for the rewriter block. The time stamp block generates an output that is either a snapshot of the corrected Local Time (struct time stamp) or a signed (two's complement) 64 bit Correction_field.

In the ingress direction, the time stamp block calculates a new time stamp for the rewriter that indicates the earlier time when the corresponding PTP event frame entered the chip (crossed the reference plane referred to in the IEEE 1588 standard).

In the egress direction, the time stamp block calculates a new time stamp for the rewriter in time for the PCS block to transmit the new time stamp field in the frame. In this case, the time stamp field indicates when the corresponding PTP event frame will exit the chip.

Transparent clocks correct PTP event messages for the time resided in the transparent clock. Peer-to-Peer transparent clocks additionally correct for the propagation time on the inbound link (Path_delay). The Path_delay [ns] input to the time stamp block is software programmed based upon IEEE 1588 path delay measurements.

In general, the IEEE 1588 standard allows for a transparent clock to update the Correction_Field for both PTP event messages as well as the associated follow up message (for two-step operation). However, the TSP only updates PTP event messages. Also, the 1588 standard allows that end-to-end transparent clocks correct and forward all PTP-timing messages while Peer-to-Peer transparent clocks only correct and forward Sync and Follow_Up messages. Again, the TSP only updates PTP event messages (not Follow_Up messages).

Internally, the time stamp block generates an Active_timestamp from the captured/time stamped Local time (Raw_timestamp). The Active_time stamp is the Raw_timestamp corrected for the both fixed (programmed) local chip, and variable chip latencies relative to where the Start_of_Frame_Indicator captures the local time. The time stamp block operates on the Active_timestamp based on the Command code.

The Active_timestamp is calculated differently in the Ingress and Egress directions and the equations are given below.

In the ingress direction:

```
Active_timestamp = Raw_timestamp - Local_latency - Variable_latency
```

In the egress direction:

```
Active_timestamp = Raw_timestamp + Local_latency + Variable_latency
```

In addition, the following values are also calculated for use by the commands:

```
Active_timestamp_ns = Active_timestamp converted to nanoseconds
Active_timestamp_p2p_ns = active_timestamp_ns + path delay
```

The Local_latency is a programmed fixed value while the Variable_latency is predicted from the PCS logic based upon the current state of the ingress or egress data pipeline.

For the option of Peer-to-Peer transparent clocks, the ingress Active_timestamp calculation includes an additional Path_delay component. The path delay is always added for a transparent clock per the standard. The path delay is always added to the correction field.

The signed 32-bit two's complement Delay Asymmetry register (bits 31–0) can be programmed by the user. Bit 31 is the sign bit. Bits 15–0 are scaled nanoseconds just like for the CorrectionField format. The DelayAsymmetry register (whether it be positive or negative) will be sign extended and added to the 64-bit correction field (signed add) if the Add_Delay_Asymmetry bit is set. The DelayAsymmetry register (whether it be positive or negative) will be sign extended and subtracted from the 64-bit correction field (signed Subtract) if the Subtract_Delay_Asymmetry bit is set.

The time stamp block keeps a shadow copy of the programmed latency values (Local_latency, Path_delay, and Delay_Asymmetry) to protect against CPU updates.

3.6.17 Time Stamp FIFO

The time stamp FIFO stores time stamps along with frame signature information. This information can be read out by a CPU or pushed out on a dedicated Serial Time Stamp Output Interface and used in 2-step processing mode to create follow-up messages. The time stamp FIFO is only present in the egress data path.

The time stamp FIFO takes a frame signature from the analyzer and the updated correction field, and the full data set for that time stamp is saved to the FIFO. This creates an interrupt to the CPU. If the FIFO ever overflows this is indicated with an interrupt.

The stored frame signature can be of varying sizes controlled by the EGR_TSFIFO_CSR.EGR_TS_SIGNAT_BYTES register. Only the indicated number of signature bytes is saved with each time stamp. The saved values are packed so that reducing the number of signature bytes allows more time stamps to be saved.

The packing of the time stamp data is done by logic before the write occurs to the FIFO. When no compression is used, each time stamp may contain 208 bits of information (consisting of 128 bits of frame signature and 80 bits of time stamp data). Therefore, a full-sized time stamp is 26 bytes long. Compressing the frame signature can reduce this to as little as 10 bytes (or 4 bytes if EGR_TSFIFO_CSR.EGR_TS_4BYTES = 1) if no signature information is saved (EGR_TSFIFO_CSR.EGR_TS_SIGNAT_BYTES = 0). The value to store is built up in an internal register. When the register contains 26 valid bytes, that data is written to the time stamp FIFO. Data in the FIFO is packed end-to-end. It is up to the reader of the data to unpack the data.

The time stamps in the FIFO are visible and accessible for the CPU as a set of 32-bit registers. Multiple register reads are required to read a full time stamp if all bits are used. Bit 31 in register EGR_TSFIFO_0 contains the current FIFO empty flag, which can be used by the CPU to determine if the current time stamps are available for reading. If the bit is set, the FIFO is empty and no time stamps are available. The value that was read can be discarded because it does not contain any valid time stamp data. If the bit is 0 (deasserted), the value contains 16 valid data bits of a time stamp. The remaining bits should be read from the other registers in the other locations and properly unpacked to recreate the time stamp. Care should be taken to read the time stamps one at a time, as each read of the last (7th) address will trigger a pop of the FIFO.

Time stamps are packed into seven registers named EGR_TSFIFO_0 to EGR_TSFIFO_6. If the time stamp FIFO registers are read to the point that the FIFO goes empty and there are remaining valid bytes in the internal packing register, then the packing register is written to the FIFO. In this case, the registers may not be fully packed with time stamps. Flag bits are used to indicate where the valid data ends within the set of seven registers. The flag bits are in register EGR_TSFIFO_0.EGR_TS_FLAGS (together with the empty flag) and are encoded as follows:

000 = Only a partial time stamp is valid in the seven register set

001 = One time stamp begins in the current seven register set

010 = Two time stamps begin in the current seven register set.

011 = Three time stamps begin in the current seven register set (4-byte mode)

100 = Four time stamps begin in the current seven register set (4-byte mode)

101 = Five time stamps begin in the current seven register set (4-byte mode)

110 = Six time stamps begin in the current seven register set (4-byte mode)

111 = The current seven register set is fully packed with valid time stamp data

The FIFO empty bit is visible in the EGR_TSFIFO_0.EGR_TS_EMPTY register so the CPU can poll this bit to know when time stamps are available. There is also a maskable interrupt which will assert whenever the time stamp FIFO level reaches the threshold given in EGR_TSFIFO_CSR.EGR_TS_THRESH register. The FIFO level is also visible in the EGR_TSFIFO_CSR.EGR_TS_LEVEL register. If the time stamp FIFO overflows, writes to the FIFO are inhibited. The data in the FIFO is still available for reading but new time stamps are dropped.

Note: Time stamp FIFO exists only in the Egress direction. There is no time stamp FIFO in the Ingress direction

3.6.18 Serial Time Stamp Output Interface

For each 1588 Processor 0 and 1, time stamp information stored in the Egress direction can be read through either the register interface or through the Serial Time Stamp interface. These two ways to read registers are mutually exclusive. While enabling/disabling the serial interface is done on a Processor level, only one serial interface exists. This means the serial interface can be enabled for Processor 0, while the time stamp FIFO can be read through registers for Processor 1. If the serial interface is enabled for both Processor 0 and 1, then the serial interface will arbitrate between two Egress time stamp FIFOs in Processor 0 and 1 and push the data out.

The time stamp FIFO serial interface block writes, or pushes, time stamp/frame signature pairs that have been enqueued and packed into time stamp FIFOs to the external chip interface consisting of three output pins: 1588_SPI_DO, 1588_SPI_CLK, and 1588_SPI_CS. There is one interface for all channels.

When the serial interface (SPI) is enabled, the time stamp/frame signature pairs are dequeued from time stamp FIFO(s) and unpacked. Unpacked time stamp/frame signature pairs are then serialized and sent one at a time to the external interface. Unpacking shifts the time stamp/frame signature into alignment considering the configured size of the time stamps and frame signatures (a single SI write may require multiple reads from a time stamp FIFO). The time stamp FIFO serial interface is an alternative to the MDIO register interface described in the time stamp FIFO section. When the serial time stamp interface is enabled in register TS_FIFO_SI_CFG.TS_FIFO_SI_ENA, data read from the time stamp FIFO registers described in Time Stamp FIFO, page 88 are invalid.

Time stamp/Frame signature pairs from two egress time stamp FIFOs are serialized one at a time and transmitted to the interface pins. The TS_FIFO_SI arbitrates in a round-robin fashion between the ports that have non-empty time stamp FIFOs. The port associated with each transmitted time stamp/frame signature pair is indicated in a serial address that precedes the data phase of the serial transmission. Because the time stamp FIFOs are instantiated in the per port clock domains, a small single entry asynchronous SI FIFO (per port) ensures that the time stamp/frame signature pairs are synchronized, staged, and ready for serial transmission. When an SI FIFO is empty, the SI FIFO control fetches and/or unpacks a single time stamp/frame signature performing any time stamp FIFO dequeues necessary. The SI FIFO goes empty following the completion of the last data bit of the serial transmission. Enabled ports (TS_FIFO_SI_CFG.TS_FIFO_SI_ENA) participate in the round-robin selection.

Register TS_FIFO_SI_TX_CNT accumulates the number of time stamp/frame signature pairs transmitted from the serial time stamp interface for each channel. Register EGR_TS_FIFO_DROP_CNT accumulates the number of time stamp/frame signature pairs that have been dropped per channel due to a time stamp FIFO overflow.

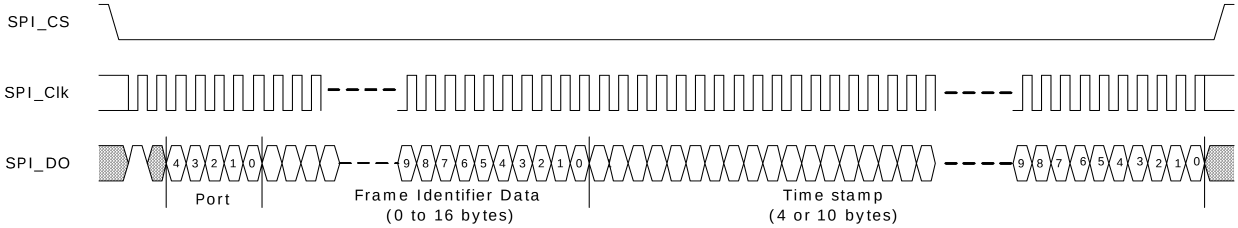
The SPI compatible interface asserts a chip select (SPI_CS) for each write followed by a write command data bit equal to 1, followed by a don't care bit (0), followed by an address phase, followed by a data phase, followed by a deselect where SPI_CS is negated. Each write command corresponds to a single time stamp/frame signature pair. The length of the data phase depends upon the sum of the configured lengths of the time stamp and signature, respectively. The address phase is fixed at five bits. The SPI_CLK is toggled to transfer each SPI_DO bit (as well as the command and address bits). The Time Stamp and Frame Identifier Data from the following illustration are sent MSB first down to LSB (bit 0) in the same format as stored in the seven registers of TS FIFO CSRs. For more information, see Time Stamp FIFO, page 88 and Figure 63, page 90.

The frequency of the generated output 1588_SPI_CLK can be flexibly programmed from 10 MHz up to 62.5 MHz using TS_FIFO_SI_CFG to set the number of CSR clocks that the 1588_SPI_CLK is both high and low. For example, to generate a 1588_SPI_CLK that is a divide-by-6 of the CSR clock, the CSR register would be set such that both SI_CLK_LO_CYCS and SI_CLK_HI_CYCS equal 3. Also, the number of CSR clocks after SPI_CS asserts before the first 1588_SPI_CLK is programmable (SI_EN_ON_CYCS), as is the number of clocks before SI_EN negates after the last 1588_SPI_CLK (SI_EN_OFF_CYCS). The number of clocks during which SI_EN is negated between writes is also programmable (SI_EN_DES_CYCS). The 1588_SPI_CLK may also be configured to be inverted (SI_CLK_POL).

Without considering de-selection between writes, if the PTP 16-byte SequenceID (frame signature) is used as frame identifier, each 10 byte time stamp write take 2 + 55 + 10 × 8 + 16 × 8 = 265 clocks (at

40 MHz) ~6625 ns. This corresponds to a time stamp bandwidth of > 0.15 M time stamp/second/port. The following illustration shows the serial time stamp/frame signature output.

Figure 63 • **Serial Time Stamp/Frame Signature Output**



3.6.19 Rewriter

When the rewriter block gets a valid indication it overwrites the input data starting at the offset specified in Rewrite_offset and replaces N bytes of the input data with updated N bytes. Frames are modified by the rewriter as indicated by the analyzer-only PTP/OAM frames are modified by the rewriter.

The output of the rewriter block is the frame data stream that includes both unmodified frames and modified PTP frames. The block also outputs a count of the number of modified PTP frames in INGR_RW_MODFRM_CNT/EGR_RW_MODFRM_CNT, depending upon the direction. This counter accumulates the number of PTP frames to which a write was performed and includes errored frames.

3.6.19.1 Rewriter Ethernet FCS Calculation

The rewriter block has to recalculate the Ethernet CRC for the PTP message to modify the contents by writing a new time stamp or clear bytes. Two versions of the Ethernet CRC are calculated in accordance with IEEE 802.3 Clause 3.2.9: one on the unmodified input data stream and one on the modified output data stream. The input frame FCS is checked against the input calculated FCS and if the values match, the frame is good. If they do not, then the frame is considered a bad or errored frame. The new calculated output FCS is used to update the FCS value in the output data frame. If the frame was good, then the FCS is used directly. If the frame was bad, the calculated output FCS is inverted before writing to the frame. Each version of the FCS is calculated in parallel by a separate FCS engine.

A count of the number of PTP/OAM frames that are in error is kept in the INGR_RW_FCS_ERR_CNT or EGR_RW_FCS_ERR_CNT register, depending upon the direction.

3.6.19.2 Rewriter UDP Checksum Calculation

For IPv6/UDP, the rewriter also calculates the value to write into the dummy blocks to correct the UDP checksum. The checksum correction is calculated by taking the original frame's checksum, the value in the dummy bytes, and the new data to be written; and using them to modify the existing value in the dummy byte location. The new dummy byte value is then written to the frame to ensure a valid checksum. The location of the dummy bytes is given by the analyzer. The UDP checksum correction is only performed when enabled using the following register bits:

- INGR_IP1_UDP_CHKSUM_UPDATE_ENA
- INGR_IP2_UDP_CHKSUM_UPDATE_ENA
- EGR_IP1_UDP_CHKSUM_UPDATE_ENA
- EGR_IP2_UDP_CHKSUM_UPDATE_ENA

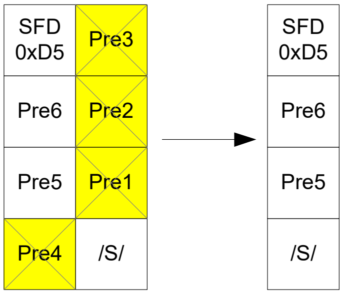
Based upon the analyzer command and the rewriter configuration, the rewriter writes the time stamp in one of the following ways:

- Using PTP_REWRITE_BYTES to choose four bytes write to PTP_REWRITE_OFFSET. This method is similar to other PTP frame modifications and the time stamp is typically written to the reserved field in the PTP header.
- Using PTP_REWRITE_BYTES and RW_REDUCE_PREAMBLE to select the mode of operation when writing Rx time stamps into the frame.
  In these modes, it cannot do both a time stamp write/append and a PTP operation in the same frame. If PTP_REWRITE_BYTES = 0xE and RW_REDUCE_PREAMBLE = 1, it does it by overwriting the existing FCS with the time stamp in the lowest four bytes of the calculated time stamp and generating a new FCS and appending it.

Because the rewriter cannot modify the IFG or change the size of the frame, if the original FCS is overwritten with time stamp data a new FCS needs to be appended and the frame shortened by reducing the preamble. The preamble length includes the /S/ character and all preamble characters up to but not including the SFD. In this mode, it is assumed that all incoming preambles are of sufficient (5 to 7-byte) length to delete four bytes and the preamble of every frame (not only PTP frames) will be reduced by four bytes by deleting four bytes of the preamble. Then, the new FCS is written at the end of the matched frame. For unmatched frames, or if the PTP_REWRITE_BYTES is anything but 0xE, the IFG is increased by adding four IDLE (/I/) characters after the /T/ which ends the packet.

To time stamp a frame in one of the modes, the actual length of the preamble is then checked and if the preamble is too short to allow a deletion of four bytes (if the preamble is not five bytes or more) then no operations are performed on the preamble, the FCS is not overwritten, and no time stamp is appended. For all such frames, a counter is maintained and every time an unsuccessful operation is encountered, the counter is incremented. This counter is read through register INGR_RW_PREAMBLE_ERR_CNT/EGR_RW_PREAMBLE_ERR_CNT. The following illustration shows the deleted preamble bytes.

Figure 64 • **Preamble Reduction in Rewriter**



If PTP_REWRITE_BYTES = 0xF and RW_REDUCE_PREAMBLE = 0, the rewriter replaces the FCS of the frame with the four lowest bytes of the calculated time stamp and does not write the FCS to the frame. In this mode, all the frames have corrupted FCSs and the MAC needs to be configured to handle this case. In the case of a CRC error in the original frame, the rewriter writes all ones (0xFFFFFFFF) to the FCS instead of the time stamp. This indicates an invalid CRC to the MAC because this is reserved to indicate an invalid time stamp. In the rare case that the actual time stamp has the value 0xFFFFFFFF and the CRC is valid, the rewriter increments the time stamp to 0x0 and writes that value instead. This causes an error of 1 ns, but is required to reserve the time stamp value of 0xFFFFFFFF for frames with an invalid CRC.

A flag bit may also be set in the PTP message header to indicate that the TSU has modified the frame (when set) or to clear the bit (on egress). The analyzer sends the byte offset of the flag byte to the rewriter in PTP_MOD_FRAME_BYTE_OFFSET and indicates whether the bit should be modified or not using PTP_MOD_FRAME_STATUS_UPDATE. The bit offset within the byte is programmed in the configuration register RW_FLAG_BIT. When the PTP frame is being modified, the selected bit is set to the value in the RW_FLAG_VAL. This only occurs when the frame is being modified by the rewriter; when the PTP frame matches and the command is not NOP.

3.6.20 Local Time Counter

The local time counter keeps the local time for the device and the time is monitored and synchronized to an external reference by the CPU. The source clock for the counter is selected externally to be a 250 MHz, 200 MHz, 125 MHz, or some other frequency. The clock may be a line clock or the dedicated CLK1588P/N pins. The clock source is selected in register LTC_CTRL.LTC_CLK_SEL.

To support other frequencies, a flexible counter system is used that can convert almost any frequency in the 125–250 MHz range into a usable source clock. Supported frequencies of local time counter are 125 MHz, 156.25 MHz, 200 MHz, and 250 MHz. The frequency is programmed in terms of the clock period. Set the LTC_SEQUENCE.LTC_SEQUENCE_A register to the clock period to the nearest whole number of nanoseconds to be added to the local time counter on each clock cycle. Set LTC_SEQ.LTC_SEQ_E to the amount of error between the actual clock period and the LTC_SEQUENCE.LTC_SEQUENCE_A setting in femtoseconds. Register LTC_SEQ.LTC_SEQ_ADD_SUB indicates the direction of the error.

An internal counter keeps track of the accumulated error. When the accumulated error exceeds 1 nanosecond, an extra nanosecond is either added or subtracted from the local time counter. Use the following as an example to program a 5.9 ns period:

```
LTC_SEQUENCE.LTC_SEQUENCE_A = 6 (6 ns)
LTC_SEQ.LTC_SEQ_E = 100000 (0.1 ns)
LTC_SEQ.LTC_SEQ_ADD_SUB = 0 (subtract an extra nanosecond, i.e add 5 ns)
```

To support automatic PPM adjustments, an internal counter runs on the same clock as the local time counter, and increments using the same sequence to count nanoseconds. The maximum (rollover) value of the internal counter in nanoseconds is given in register LTC_AUTO_ADJUST.LTC_AUTO_ADJUST_NS. At rollover, the next increment of the local time counter is increased by one additional or one less nanosecond as determined by the LTC_AUTO_ADJUST.LTC_AUTO_ADD_SUB_1NS register. When LTC_AUTO_ADJUST.LTC_AUTO_ADD_SUB_1NS is set to 0x1, an additional nanosecond is added to the local time counter. When it is set to 0x2, one less nanosecond is added to the local timer counter. No PPM adjustments are made when the register is set to 0x0 or 0x3.

PPM adjustments to the local time counter can be made on an as-needed basis by writing to the one-shot LTC_CTRL.LTC_ADD_SUB_1NS_REQ register. One nanosecond is added or subtracted from the local time counter each time LTC_CTRL.LTC_ADD_SUB_1NS_REQ is asserted. The LTC_CTRL.LTC_ADD_SUB_1NS register setting controls whether the local time counter adjustment is an addition or a subtraction.

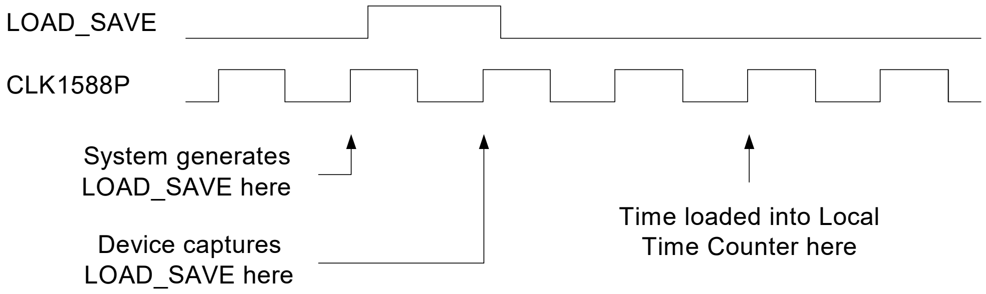
The current time is loaded into the local time counter with the following procedure.

1. Configure the 1588_LOAD_SAVE pin.
2. Write the time to be loaded into the local time counter in registers LTC_LOAD_SEC_H, LTC_LOAD_SEC_L and LTC_LOAD_NS.
3. Program LTC_CTRL.LTC_LOAD_ENA to a 1.
4. Drive the 1588_LOAD_SAVE pin from low to high.

The time in registers LTC_LOAD_SEC_H, LTC_LOAD_SEC_L and LTC_LOAD_NS is loaded into the local time counter when the rising edge of the 1588 LOAD_SAVE strobe is detected. The LOAD_SAVE strobe is synchronized to the local time counter clock domain.

When the 1588_DIFF_INPUT_CLK_P/N pins are the clock source for the local time counter, and the LOAD_SAVE strobe is synchronous to 1588_DIFF_INPUT_CLK_P/N, the LTC_LOAD* registers are loaded into the local time counter, as shown in the following illustration.

Figure 65 • **Local Time Counter Load/Save Timing**



When the LOAD_SAVE strobe is not synchronous to the 1588_DIFF_INPUT_CLK_P/N pins or an internal clock drives the local time counter, there is some uncertainty as to when the local time counter is loaded, when higher accuracy circuit is turned off. This reduces the accuracy of the time stamping function by the period of the local time counter clock. When higher accuracy circuit is ON, any difference between the 1588_DIFF_INPUT_CLK_P and the rising edge of 1588_LOAD_SAVE is compensated within an error of 1 ns. This applies to both load and save operations.

Note: There is a local time counter in each channel. The counter is initialized in both channels if the LTC_CTRL.LTC_LOAD_ENA register in each channel is asserted when the LOAD_SAVE strobe occurs.

When LTC_CTRL.LTC_SAVE_ENA register is asserted when the 1588 LOAD_SAVE input transitions from low to high, the state of the local time counter is stored in the LTC_SAVED_SEC_H, LTC_SAVED_SEC_L, and LTC_SAVED_NS registers.

The current local time can be stored in registers with the following procedure.

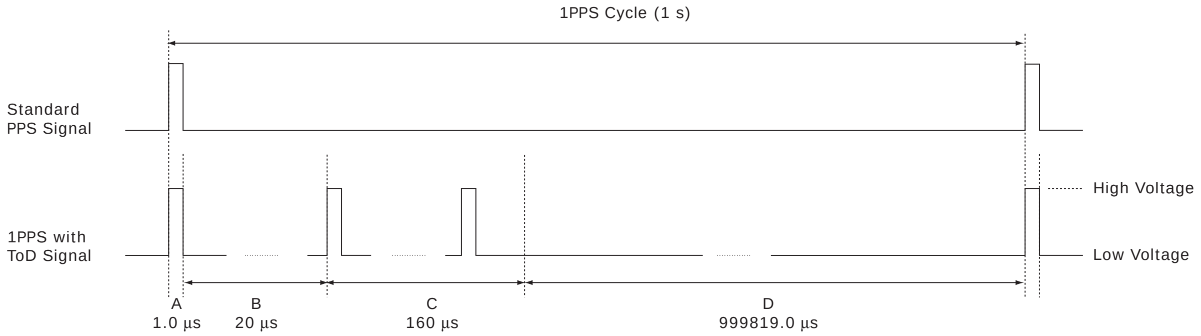
1. Configure the 1588_LOAD_SAVE pin.
2. Program LTC_CTRL.LTC_SAVE_ENA to a 1.
3. Set SER_TOD_INTF.LOAD_SAVE_AUTO_CLR to 1 if the operation is one-time save operation. This will clear LTC_CTRL.LTC_SAVE_ENA after the operation.
4. Drive the 1588_LOAD_SAVE pin from low to high.
5. Read the value from LTC_SAVED_SEC_H, LTC_SAVED_SEC_L, and LTC_SAVED_NS registers.

As with loading the local time counter, there is one clock cycle of uncertainty as to when the time is saved if the LOAD_SAVE strobe is not synchronous to the clock driving the counter.

3.6.21 Serial Time of Day

In addition to loading or saving as described in the preceding sections, it is possible to load or save LTC time in a serial fashion. For serial load, 1588_LOAD_SAVE has to send Time of Day (ToD) information in a specific format. For serial save, when the appropriate register bits is set, then PPS will drive out the ToD information. The following illustration shows the format for serial load and save.

Figure 66 • **Standard PPS and 1PPS with TOD Timing Relationship**



3.6.21.1 Pulse per Second Segment

In the preceding illustration, segment A is the pulse per second segment. The PPS signal is transmitted with high voltage. The rising edge of the PPS signal is aligned with the rising edge of the standard PPS signal. This segment lasts 1 µs. To obtain high accuracy, the response delay of the rising edge of the PPS signal should be considered.

3.6.21.2 Waiting Segment

In the preceding illustration, segment B is the Waiting segment. Due to the speed of operation, this segment is needed to make it easier for the receiver to obtain the following Time-of-Day information in current PPS cycle. The signal is in low voltage during this segment, which lasts 20 µs.

3.6.21.3 Time-of-Day Segment

In the preceding illustration, segment C is the Time-of-Day segment. The ToD information being carried in this segment indicates the time instant of the rising edge of the PPS signal transmitted in segment A of the current PPS cycle. The time instant is measured using the original network clock. In this segment, the ToD information is continuously transported and is represented in 16 octets. It consists of the following fields:

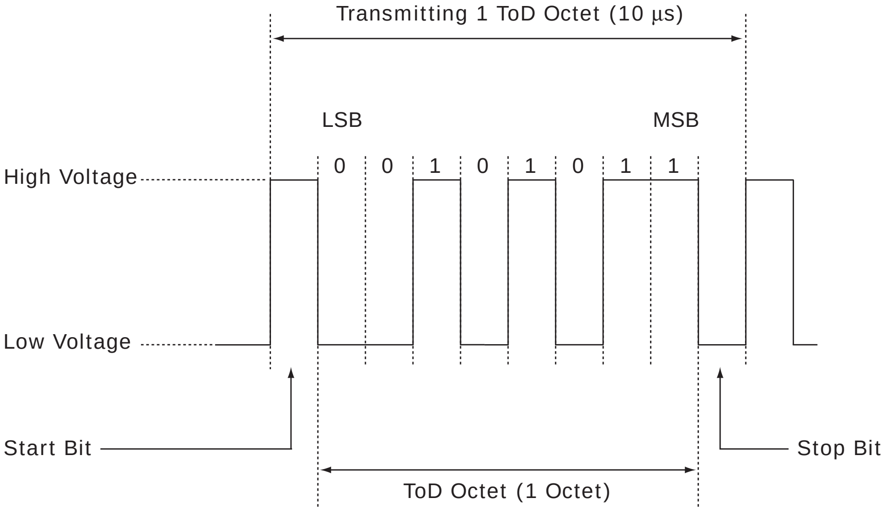
- Second field: 6 octets. It represents the time instant of the rising edge of the PPS signal in second. The value is defined as in IEEE 1588-2008.
- Date field: 6 octets. It represents the time instant of the rising edge of the PPS signal in year, month, day, hour, minute, and second. Each part is represented by one octet (the format of this field is 0xYYMMDDHHMMSS). In particular, only the lowest 2 decimal digits of the year are represented. The receiver can easily obtain the time instant of the rising edge of the PPS signal in this transparent format without additional circuitry to translate the value of the second field. It also has the significant

benefit of changing the value of this field when leap year or leap second occurs. (The Date field is ignored at the serial ToD input and is not generated at the serial ToD output.)

- Reserved field: 4 octets. Reserved for future use.

The ToD information is represented in units of octet, with each octet being transmitted with the low-order bit first. The following illustration shows an octet is transmitted between a start bit with high voltage and a stop bit with low voltage. The other octets are transmitted in the same manner. As a result, (1+8+1) × 1 µs = 10 µs are needed to transport one octet. This segment lasts 16 × 10 µs = 160 µs to convey the ToD information.

Figure 67 • **ToD Octet Waveform**



The entire Time-of-Day segment should be detected. If the second 6 octets representing the Date field are not used by the upper layer, the Date field should still be detected and its value can be ignored.

3.6.21.4 Idle Segment

Segment D is the Idle segment in Figure 66, page 93. It follows segment C with high voltage until the end of the PPS cycle. The duration of the Idle segment is given by the following calculation.

1 s – 0.5 µs – 20 µs – 160 µs = 999819.5 µs.

Use the following steps to enable Serial load.

1. Set SER_TOD_INTF.SER_TOD_INPUT_EN to 1
2. Set LTC_CTRL.LOAD_EN to 1.
3. Start the transmission of 1588_LOAD_SAVE conforming to the format.
4. To check the data transmission, enable serial save or save LTC time to check the registers.
5. To enable serial save, set SER_TOD_INTF.SER_TOD_OUTPUT_EN to 1.

The following table lists the different options to load or save LTC time.

Table 36 • **LTC Time Load/Save Options**

LTC_CTRL.LOAD_EN	SER_TOD_INTF.SER_TOD_INPUT_EN	LTC_CTR.SAVE_EN	Expected Operation
0	0	1	Parallel Save
0	1	1	Save
0	0	0	No operation
0	1	0	No operation
1	0	0	Parallel Load
1	1	0	Serial Load

Table 36 • **LTC Time Load/Save Options *(continued)***

LTC_CTRL.LOAD _EN	SER_TOD_INTF.SER_TOD_INPUT _EN	LTC_CTR.SAVE _EN	Expected Operation
1	0	1	Parallel Load and Save
1	1	1	Serial Load and Save

When SER_TOD_INTF.SERIAL_ToD_OUTPUT_EN is set, the PPS output is driven with a serial ToD output based on the LTC timer value.

3.6.22 Programmable Offset for LTC Load Register

When a new LTC value is loaded into the system, a fixed offset may need to be added to the loaded value. Program SER_TOD_INTF.LOAD_PULSE_DLY, and this value will be added to LTC counter whenever a new load occurs either through software, load_save pin, or through serial ToD.

3.6.23 Adjustment of LTC Counter

LTC counter value can be adjusted by about a second without reloading a new LTC value. LTC value can be programmed to tune the current value by adding or subtracting a specific value. The offset adjustment can be positive or negative, very similar to 1 ns adjustment being positive or negative. An adjustment every 232 ns can be performed using LTC_OFFSET_ADJ. Additionally, an adjustment every 220 ns can be performed using LTC_AUTO_M_x.

The purpose of this register is to add/subtract a programmable offset register of 30-bit width in ns, to the register block in order to cover the entire nanosecond portion of the 80-bit LTC. This offset control is independent of the LTC load control. The LTC timer is adjusted - added or subtracted as per the bit LTC_OFFSET_ADJ.LTC_ADD_SUB_OFFSET, by the value LTC_OFFSET_ADJ.LTC_OFFSET_VAL, when a load offset command is issued by the software (assertion of LTC_OFFSET_ADJ.LTC_OFFSET_ADJ). The hardware sets the status bit LTC_OFFSET_ADJ_STAT.LTC_OFFSET_DONE after completing the operation. However, in case the hardware cannot complete the operation because of the LTC value itself getting updated synchronously due to parallel or serial LTC load at the same time, it sets the bit LTC_OFFSET_ADJ_STAT.LTC_OFFSET_LOAD_ERR. The software, on seeing either of these status bits set (LTC_OFFSET_ADJ_STAT.LTC_OFFSET_DONE or LTC_OFFSET_ADJ_STAT.LTC_OFFSET_LOAD_ERR), de-asserts the control bit (LTC_OFFSET_ADJ.LTC_OFFSET_ADJ) and might potentially retry the operation.

The maximum value in nanoseconds for the offset LTC_OFFSET_ADJ.LTC_OFFSET_VAL can be up to 10^9 - 1. Thus, for addition operation, the maximum carry to the seconds counter is 2 because of the clock period addition to this maximum value present in the offset and LTC nanoseconds counter. For subtraction operation, if the resultant subtraction is negative or underflows the LTC timer would be set to wrong value. Therefore, such operations should never be allowed.

LTC_OFFSET_ADJ register (with LTC_OFFSET_VAL[29:0] and LTC_ADD_SUB_OFFSET) should be updated before asserting LTC_OFFSET_ADJ bit in LTC_OFFSET_ADJ register.

LTC_OFFSET_ADJ_STAT.LTC_OFFSET_DONE and LTC_OFFSET_ADJ_STAT.LTC_OFFSET_LOAD_ERR bits are set by the hardware and cleared by the software by writing a zero.

Should a conflict occur between LTC update due to parallel/serial load and LTC update due to offset adjustment, the load LTC takes precedence and the error condition is noted so that the polling software does not hang on the offset status bit assertion.

LTC counter could be adjusted for any known drift that occurs on every second. This feature will add or subtract one nanosecond every time LTC crosses over LTC_AUTO_ADJ_M_NS.

Example 1. If LTC_AUTO_ADJ_M_NS is 100 ns and LTC is started from reset with a value of 0 ns, then LTC counter will be added/subtracted 1 ns every time counter rolls over 100 ns.

Example 2. If LTC_AUTO_ADJ_M_NS is 100 ns and LTC is started from reset with a value of 0 ns, then LTC counter will be added/subtracted 1 ns every time counter rolls over. When counter is at 10 ns and

LTC counter is loaded with 2 sec, 80 ns. Now 1 ns is adjusted when counter increments from 10 ns and rolls over 100 ns. It does not add/subtract when LTC timer rolls over 100 ns.

Example 3. LTC_AUTO_ADJ_M_NS value is loaded with 400 ns and after some time LTC_AUTO_ADJ_M_NS value is loaded with 500 ns. The AUTO_ADJ_M_COUNTER value when the new value is loaded is 333 ns. Then the next adjustment happens after 177 ns after load because the AUTO_ADJ_M_COUNTER continues to count until it reaches the newly loaded value 500 ns.

Example 4. LTC_AUTO_ADJ_M_NS value is loaded with 400 ns and after some time LTC_AUTO_ADJ_M_NS value is loaded with 100 ns. The AUTO_ADJ_M_COUNTER value when the new value is loaded is 333 ns. Then adjustment happens immediately because 333 > 100 and the AUTO_ADJ_M_COUNTER is reset to zero after the adjustment

If LTC counter is loaded with a new value, set LTC_AUTO_ADJ_M_UPDATE bit to 1 and reload the LTC_AUTO_ADJ_M_NS value.

3.6.24 Pulse per Second Output

The local time counter generates a one pulse-per-second (1PPS) output signal with a programmable pulse width routed to GPIO pins. The pulse width of the 1PPS signal is determined by the LTC_1PPS_WIDTH_ADJ register.

When the LTC counter exceeds the value in PPS_GEN_CNT (both are in nanoseconds), the PPS signal is asserted. In default operation where PPS_GEN_CNT = 0 the LTC timer generates a PPS signal every time LTC crosses the 1 sec boundary. By writing a large value (such as 10^9-60 ns), the 1PPS pulse reaches its destination 60 ns away simultaneously with the LTC second wrap, thus providing time-of-day synchronism between two systems.

The 1PPS output has an alternate mode of operation that increases the frequency of the pulses. This mode may be used for applications such as locking an external DPLL to the IEEE 1588 frequency. In the alternate mode, the 1PPS signal is driven directly from a single bit of the nanosecond field counter of the local time counter. The pulse width cannot be controlled in this alternate operation mode. The alternate mode is enabled with register LTC_CTRL.LTC_ALT_MODE_PPS_BIT.

The output frequencies that result are 1 divided by powers of 2 nanoseconds (bit 4 = 1/32 ns, bit 5 = 1/64 ns, bit 6 = 1/128 ns, …). The output pulses may jitter by the amount of the programmed nanoseconds of the adder to the local nanoseconds counter, and any automatic or one-shot adjustments.

The following table shows the possible output pulse frequencies (including the range of 4 kHz to 10 MHz) usable for external applications.

Table 37 • **Output Pulse Frequencies**

Nanosecond Counter Bit	Output Pulse Frequency
4	31.25 MHz
5	15.625 MHz
6	7.8125 MHz
7	3.90625 MHz
8	1.953125 MHz
9	976.5625 kHz
10	488.28125 kHz
11	244.140625 kHz
12	122.0703125 kHz
13	61.03515625 kHz
14	30.51757813 kHz
15	15.25878906 kHz

Table 37 • **Output Pulse Frequencies *(continued)***

Nanosecond Counter Bit	Output Pulse Frequency
16	7.629394531 kHz
17	3.814697266 kHz

In addition to the preceding frequencies, a specific frequency can be chosen by enabling the synthesizer on the PPS pin using the following steps.

1. Set LTC_FREQ_SYNTH.LTC_FREQ_SYNTH_EN to 1.
2. A toggle signal with the frequency specified will be pushed out onto PPS. The number of nanoseconds the signal stays high can be specified by LTC_FREQ_SYNTH.FREQ_HI_DUTY_CYC_NS. The number of nanoseconds the signal stays low can be specified by LTC_FREQ_SYNTH.FREQ_LO_DUTY_CYC_NS.
3. The above nanoseconds should be exactly divisible by clock frequency, otherwise the signal may have a jitter as high as the high duration/clock period or low duration/clock period.
4. To disable the this feature and revert back to PPS functionality, reset LTC_FREQ_SYNTH.LTC_FREQ_SYNTH_EN to 0

For example, to output a 10 MHz signal, set the FREQ_HI_DUTY_CYC_NS to 50 ns and FREQ_LO_DUTY_CYC_NS to 50 ns. On a 250 MHz LTC clock, this will make high time and low time of signal shift between 48 ns and 52 ns.

3.6.25 Resolution

The IEEE 1588 processor achieves time stamp resolution in any mode of operation of 1 ns utilizing special high-resolution circuitry. The accuracy of a device using high-resolution circuitry is improved more than 100% over the first generation IEEE 1588 engine. High accuracy for these devices will be supported regardless of the local time counter clock frequency supplied to the reference clock input. The timestamp accuracy is a system-level property and may depend upon oscillator selection, port type, and speed, system configuration, and calibration decisions. Supported frequencies of the local time counter are 125 MHz, 156.25 MHz, 200 MHz, and 250 MHz.

There are a total of five high resolution blocks per port to improve resolution for the following events:

- One pulse-per-second (1PPS) output signal
- 1588_PPS_RI input signal
- Start-of-frame in the egress direction
- Start-of-frame in the ingress direction
- 1588_LOAD_SAVE input (strobe) signal direction

Each of these blocks can individually be enabled using ACC_CFG_STATUS. Contact Microsemi with any questions regarding PTP accuracy calculations.

3.6.26 Loopbacks

Loopback options provide a means to measure the delay at different points to evaluate delays between on chip wire delays and external delays down to a nanosecond.

3.6.26.1 Loopback from PPS to PPS_RI Pin

In this loopback, an external device will connect the PPS coming out of the IEEE 1588 to PPS_RI of the IEEE 1588 device. The external device could even process the PPS signal and then loopback at a far-end.

3.6.26.2 Loopback from LOAD_SAVE to PPS

When LOAD_SAVE_PPS_LPBK_EN is set, input load_save pin is connected to output PPS coming out of the IEEE 1588. In this mode, input load_save pin is taken as close to the pin as possible without going through any synchronization logic on the load_save pin.

3.6.26.3 Loopback of LOAD_SAVE Pin

When LOAD_SAVE_LPBK_EN is set, one clock cycle before the PPS is asserted, an output enable for load_save pin is generated and PPS signal is pushed out on the load_save pin acting as an output pin. After two cycles, output enable is brought down and load_save will behave as an input pin.

3.6.26.4 Loopback from PPS to LOAD_SAVE Pin

When PPS_LOAD_SAVE_LPBK_EN bit enabled, output pps signal is taken as close to the I/O as possible and looped back onto load_save input pin. This is to account for any delays from PPS generation block to the PPS output pin.

3.6.27 Accessing 1588 IP Registers

The following sections describe how the IEEE 1588 IP registers are accessed in the VSC8489-17 device.

Note: Contact Microsemi for an initialization script that supports the quick initialization of IEEE 1588 registers.

3.7 Flow Control Buffers

Flow control buffers are used in the data paths in 10G/1G operation when the MACs are enabled. Ethernet frames are stored in the buffers. When a buffer is close to being full, the MAC will issue a pause frame to the device sending data to the VSC8489-17 device. This is done to prevent the data path's flow control buffer from overflowing.

The flow control feedback is particularly common for the host interface when in 10G WAN mode due to the transmitted line WAN data rate being less than the received host LAN data rate.

3.8 Rate Compensating Buffers

Rate compensating buffers are used in the data paths when the MACs are disabled. The rate compensating buffers add and drop idle characters between ethernet packets when necessary to address clock rate differences between the line-side and host-side interfaces. Rate offsets from ideal frequencies measured in ppm (not MHz) can be tolerated.

The maximum data throughput on the line interface is less in 10G WAN mode than 10G LAN mode. The line's data rate is reduced to 9.953 Gbps from 10.3125 Gbps. Part of that bandwidth includes SONET/SDH frame overhead data.

Note: Care must be taken by the device sending data to the host interface in the egress data path to ensure the rate compensating buffer does not overflow.

3.9 Loopback

The VSC8489-17 device has several options available for routing traffic between the host-side and the line-side. The following table shows the name and location of the loopback modes. These modes may be extremely useful for both test and debug purposes.

Table 38 • **Host-Side Loopbacks**

Name	Location	Line-Side Tx	Notes
H2	XAUI-PHY interface (1G and 10G)	Mirror XAUI data	
H3	PCS after the gearbox (10G)	0xFF00 repeating	IEEE PCS system loopback
H4	WIS-PMA interface (10G)	0xFF00 repeating	IEEE WIS system loopback

Table 39 • **Line-Side Loopbacks**

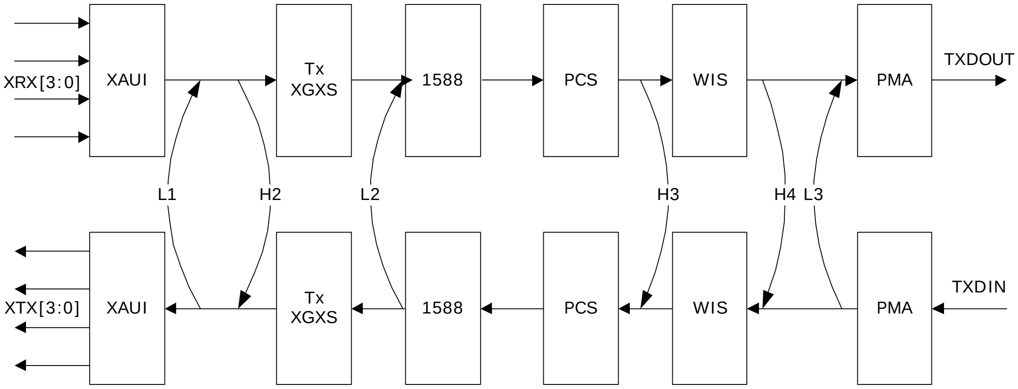
Name	Location	Host-Side Tx	Notes
L1	XAUI loopback (10G)	Mirror SFP+ data	IEEE PHY-XS network loopback ()
L2	XGMII interface (10G)	Mirror SFP+ data	

Table 39 • Line-Side Loopbacks (continued)

Name	Location	Host-Side Tx	Notes
L3	PMA interface (1G and 10G)	Mirror SFP+ data	

The following illustration shows the host and line-side loopbacks.

Figure 68 • Host-Side and Line-Side Loopbacks



3.10 Cross-Connect (Non-Hitless Operation)

The VSC8489-17 device features a cross-connect between the two adjacent ports to support protection-switching applications and failover capabilities. It supports cross-connect and line broadcast (bridge and select) to host-side between the two adjacent ports.

In the cross-connect configuration, each port's line-side, which is normally connected to its own port host-side, is connected to the other port's host-side. In a broadcast configuration, a line-side of one port is connected to both port's host-sides.

XAUI interfaces in the VSC8489-17 device support the following failover capabilities:

- Network traffic on Chan0 is switchable between XAUI channel 0 or 1.
- Network traffic on Chan1 is switchable between XAUI channel 1 or 0.

The VSC8489-17 device supports failover and SFI to XAUI broadcasting.

The XAUI data at channel_0 can be routed to either SFI of channel_0 or SFI of channel_1, but not both at the same time. Similarly, the XAUI data at channel_1 can be routed to ether SFI of channel_1 or SFI of channel_0, but not both at the same time. However, the SFI data of either channel_0 or channel_1 can be routed to XAUI of channel_0, or XAUI of channel_1, or broadcast to XAUI of both channel_0 and channel_1.

For example, in normal operation, the XAUI of channel_0 is routed to SFI of channel_0. When a problem occurs on the link connecting to SFI of channel_0, re-route the XAUI of channel_0 to SFI of channel_1. Or, if there is a problem at the MAC interfacing to the XAUI of channel_0, re-route the SFI data of channel_0 to XAUI of channel_1. Broadcasting from SFI to both XAUI ports enables the passing of traffic between XAUI of channel_0 and SFI of channel_0, and to use XAUI of channel_1 to snoop the incoming data from SFI of channel_0, if so desired.

The following table lists the available settings for failover/broadcasting modes.

Table 40 • Failover and Broadcasting Modes

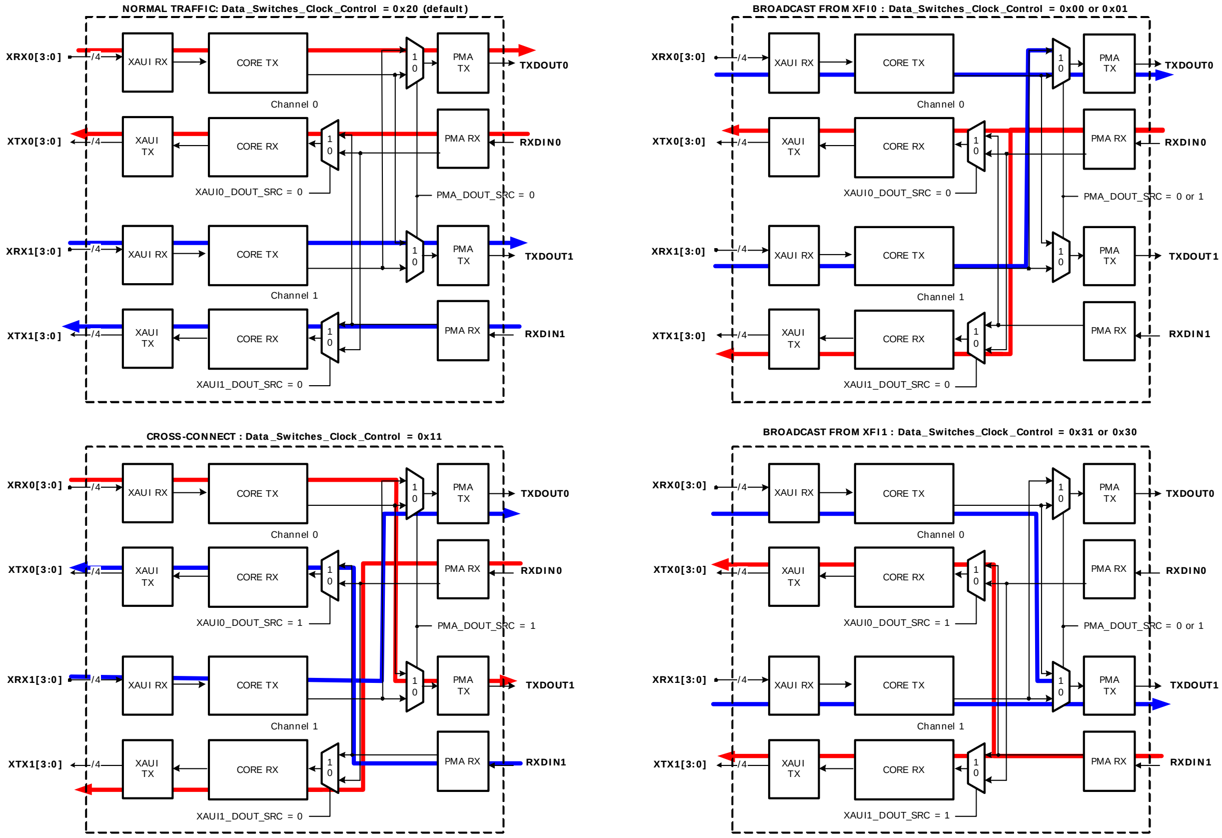
Setting	Mode
1Ex0003 =0x00	XAUI channel_0 to/from SFI channel_0 interface XAUI channel_1 from SFI channel_0 interface SFI channel_1 from XAUI channel_1

Table 40 • **Failover and Broadcasting Modes *(continued)***

Setting	Mode
1Ex0003 = 0x01	XAUI channel_0 from SFI channel_0 interface XAUI channel_1 to/from SFI channel_0 interface SFI channel_1 from XAUI channel_0
1Ex0003 =0x20	XAUI channel_0 to/from SFI channel_0 interface XAUI channel_1 to/from SFI channel_1
1Ex0003 =0x21	XAUI channel_0 from SFI channel_0 XAUI channel_1 from SFI channel_1 SFI channel_0 from XAUI channel_1 SFI channel_1 from XAUI channel_0
1Ex0003 =0x10	XAUI channel_0 from SFI channel_1 XAUI channel_1 from SFI channel_0 SFI channel_0 from XAUI channel_0 SFI channel_1 from XAUI channel_1
1Ex0003 =0x11	XAUI channel_0 to/from SFI channel_1 XAUI channel_1 to/from SFI channel_0
1Ex0003 =0x30	XAUI channel_0 from SFI channel_1 XAUI channel_1 to/from SFI channel_1 SFI channel_0 from XAUI channel_0
1Ex0003 =0x31	XAUI channel_0 to/from SFI channel_1 XAUI channel_1 from SFI channel_1 SFI channel_0 from XAUI channel_1

The following illustration shows the cross-connect configurations.

Figure 69 • **Cross-Connect Configuration**


NORMAL TRAFFIC: Data_Switches_Clock_Control = 0x20 (default)

BROADCAST FROM XFI0 : Data_Switches_Clock_Control = 0x00 or 0x01

CROSS-CONNECT : Data_Switches_Clock_Control = 0x11

BROADCAST FROM XFI1 : Data_Switches_Clock_Control = 0x31 or 0x30

3.11 Host-Side Interface

The XAUI, RXAUI, and 1 GbE host interfaces in the VSC8489-17 device support the following rates:

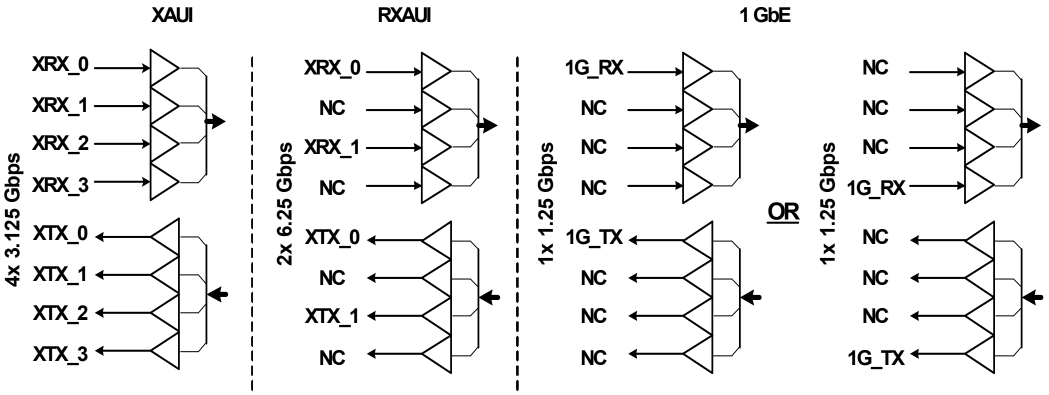
- XAUI: 4 × 3.125 Gbps
- RXAUI: 2 × 6.25 Gbps
- 1 GbE: 1 × 1.25 Gbps

In RXAUI mode, the two RXAUI lanes are XAUI lane 0 and lane 2. XAUI lane 0 is the RXAUI lane 0 and XAUI lane 2 is RXAUI lane 1. The LSB are sent on lane 0, while MSB are sent on late 1. In 1 GbE mode, the 1 GbE lanes may be either XAUI lane 0 or lane 3. The following illustration shows the host-side I/O interface.

The XAUI lane order could be swapped through register 4xF002, where bit 2 is to map lane 0 on 3, lane 1 on 2, lane 2 on 1, and lane 3 on 0 of the XAUI output. Bit 0 of same register is to map lane 0 on 3, lane 1 on 2, lane 2 on 1, and lane 3 on 0 of the XAUI input. Furthermore, bit 2 of 4xF002 could be used to invert the polarity of the differential pairs of all four XAUI outputs, and bit 1 of 4xF002 could be used to invert the polarity of the differential pairs of all four XAUI inputs. There is an API function call to assist with setting the lane swap and polarity inversion.

Note: Use AC-coupling for the receive and transmit sides of the host-side interface. For optional DC-coupling, contact your Microsemi representative.

Figure 70 • **Host-Side I/O Interface**



3.11.1 RXAUI Interoperability

The RXAUI implementation is fully interoperable with the Dune network mode 1 and mode 2 RXAUI specification, as summarized in the following table.

Table 41 • **RXAUI Interoperability**

Mode1	Mode2
De-interleaving uses XAUI ||A|| column (align column).	
Transmitter: No internal logic modification. Two 10b characters are presented to one physical lane at a time.	Transmitter: Replaces some |K| code groups with other code groups to allow receiver to perform de-interleaving (comma replacement).
Receiver: Separates two characters in double rate lane into two physical standard rate lanes. Lane byte ordering block ensures first |A| character mapped to first logical lane and |A| column aligned for the deskew block.	
Obeys 6.25 Gbps disparity rules.	Obeys 6.25 Gbps disparity rules.

A host-side SerDes macro (SD6G) is used by each XAUI lane. The SerDes macros are automatically configured to send/receive the proper data rate when the host interface changes between the XAUI/RXAUI/1 GbE modes. The appropriate lanes are also powered down when not in use. For example, XAUI lanes 1 and 3 are powered down when the RXAUI interface is in use.

3.12 Clocking

The following sections describe the clocking functionality of the VSC8489-17 device.

3.12.1 PLL

The VSC8489-17 device includes two PLLs, one on the line-side and another on the host-side. The line-side PLL uses either XREFCK or WREFCK as its reference clock. The host-side PLL uses XREFCK. The following table shows the supported reference clock frequencies.

Table 42 • **Supported Reference Clock Frequencies**

Clock	Applications	Frequencies
XREFCK	Mainly LAN mode	125 MHz and 156.25 MHz
WREFCK	LAN or WAN mode Synchronous Ethernet	155.52 MHz and 161.13 MHz

Table 42 • **Supported Reference Clock Frequencies *(continued)***

Clock	Applications	Frequencies
SREFCK	Synchronous Ethernet	125 MHz, 155.52 MHz, and 156.25 MHz

3.12.2 Reference Clock

The VSC8489-17 device uses three differential input CML level reference clocks: XREFCK, WREFCK, and SREFCK. The XREFCK is required all the time and may be either 156.25 MHz or 125 MHz.

The VSC8489-17 device features an internal frequency synthesizer that enables operation in 10G LAN/10G WAN/1G LAN modes using a single reference clock input (XREFCK). The host-side PLL is always driven by XREFCK. It is recommended to have the line-side PLL be driven by XREFCK.

For backward compatibility with previous generation PHY chips, WREFCK may be used to drive the line-side PLL. For Synchronous Ethernet applications with non-hitless XREFCK, SREFCK could be used to drive the line-side PLL.

The XREFCK frequency has to be decided before power up, and is selected using the MODE1 and MODE0 pins. The following table shows the MODE pin settings for the various XREFCK frequencies.

Table 43 • **XREFCK Frequency Selection**

MODE1 Pin	MODE0 Pin	XREFCK Frequency
0	0	156.25 MHz (default)
0	1	Reserved
1	0	125 MHz
1	1	Reserved

The following table shows the supported clock rates and modes.

Table 44 • **Supported Clock Rates and Modes**

Mode	XREFCK	WREFCK	SREFCK	Tx CMU REF	Rx CMU REF
10.3125G LAN single ref	156.25 MHz			XREFCK	XREFCK
9.95328G WAN single ref	156.25 MHz			XREFCK	XREFCK
1.25G LAN single ref	156.25 MHz			XREFCK	XREFCK
10.3125G LAN single ref	125 MHz			XREFCK	XREFCK
9.95328G WAN single ref	125 MHz			XREFCK	XREFCK
1.25G LAN single ref	125 MHz			XREFCK	XREFCK
10.3125G Sync-E LAN single ref	156.25 MHz (Hitless)			XREFCK	XREFCK
10.3125G Sync-E LAN dual ref	156.25 MHz		161.13 MHz	SREFCK	XREFCK
	156.25 MHz		156.25 MHz	SREFCK	XREFCK
	156.25 MHz		125 MHz	SREFCK	XREFCK
9.95328G Sync-E WAN single ref	156.25 MHz (Hitless)			XREFCK	XREFCK
9.95328G Sync-E WAN dual ref	156.25 MHz		155.52 MHz	SREFCK	XREFCK
9.95328G Sync-E WAN dual ref	156.25 MHz	155.52 MHz		WREFCK	WREFCK
1.25G Sync-E LAN single ref	156.25 MHz (Hitless)			XREFCK	XREFCK

Table 44 • **Supported Clock Rates and Modes *(continued)***

Mode	XREFCK	WREFCK	SREFCK	Tx CMU REF	Rx CMU REF
1.25G Sync-E LAN dual ref	156.25 MHz		125 MHz	SREFCK	XREFCK

3.12.3 Synchronous Ethernet Support

The VSC8489-17 device supports several synchronous Ethernet configurations for 1G and 10G modes of operation. In synchronous Ethernet applications, only one master at a time may be selected from one of the internal line-side Rx or the SREFCK.

- **Single device, internal master:** in this configuration, the line-side Rx captures the serial data input and generates a lane-synchronization signal that contains information about the incoming data rates. The signal is then distributed to all ports of the line-side Tx to form a source-synchronous operation.
- **Single clock LAN, external master:** in this configuration, the XREFCK is gradually changed to the externally generated synchronous Ethernet clock using an external clock distribution chip. The change has to be hitless to avoid data corruption. The XREFCK source may come from one of the port recovered clocks through the RXCKOUT pin.
- **Dual clock LAN, external master:** in this configuration, the XREFCK remains connected to the stable 156.25 MHz system clock or crystal. All the line-side transmits are then synchronized to SREFCK. The F to delta F block accepts the SREFCK clock from external synchronous Ethernet master and generates the lane Sync signal to effectively synchronize all the line-side Tx to this external clock. SREFCK must be 156.25 MHz.
- **Dual clock WAN, external master:** in this configuration, the XREFCK remains connected to the stable 156.25 MHz system clock or crystal. The XREFCK is configured to drive the host-side PLL, while the WREFCK is configured to drive the line-side PLL. The synchronous Ethernet clock is then routed through the SREFCK clock to synchronize all the line-side transmits. SREFCK must be 155.52 MHz.

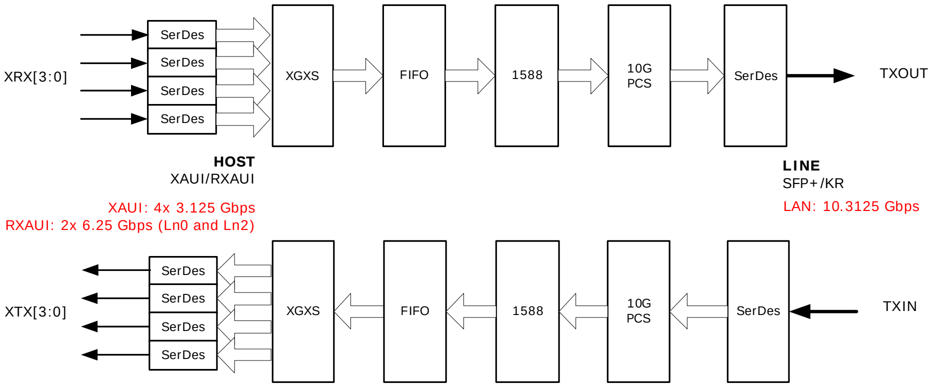
3.13 Operating Modes

The VSC8489-17 device has three main operation modes: LAN, WAN, and 1 GbE. Each mode may have the 1588 blocks on or off.

3.13.1 10G LAN with 1588

In 10G LAN mode with 1588, the host interface is XAUI (4 × 3.125 Gbps) or RXAUI (2 × 6.250 Gbps), and the line interface is LAN SFP+ (10.3125 Gbps). A single reference clock input pin, XREFCK, is used by both the line-and host-side interfaces to transmit appropriate rates. For more information about supported reference clock frequencies, see Table 44, page 103.

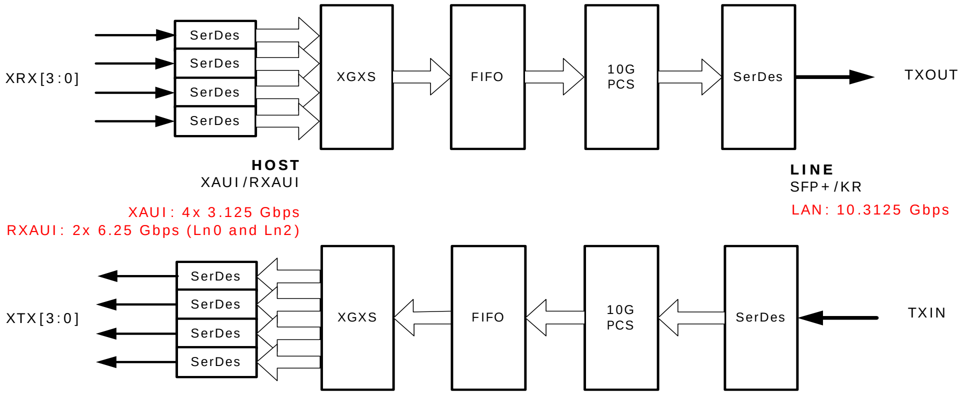
Figure 71 • **10G LAN with 1588**



3.13.2 10G LAN

In 10G LAN mode, the host interface is XAUI (4 × 3.125 Gbps) or RXAUI (2 × 6.250 Gbps), and the line interface is LAN SFP+ (10.3125 Gbps). A single reference clock input pin, XREFCK, is used by both the line-and host-side interfaces to transmit appropriate rates. For more information about supported reference clock frequencies, Table 44, page 103.

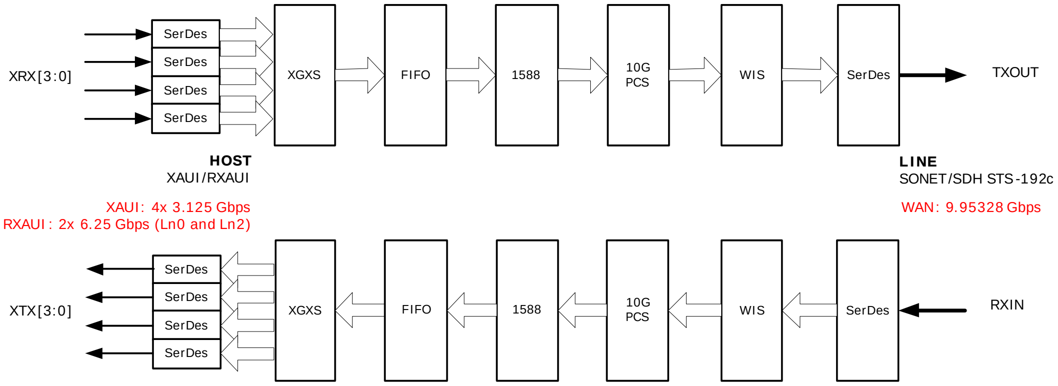
Figure 72 • **10G LAN**



3.13.3 10G WAN with 1588

In 10G WAN mode with 1588, the host interface is XAUI (4 × 3.125 Gbps) or RXAUI (2 × 6.250 Gbps), and the line interface is SONET/SDH STS-192c (9.95328 Gbps). A single reference clock input pin, XREFCK, is used by both the line-and host-side interfaces to transmit appropriate rates. For more information about supported reference clock frequencies, see Table 44, page 103. Optionally, WREFCK may be used as the 155.52 MHz reference clock for the line interface. A 622 MHz WREFCK reference frequency is not supported.

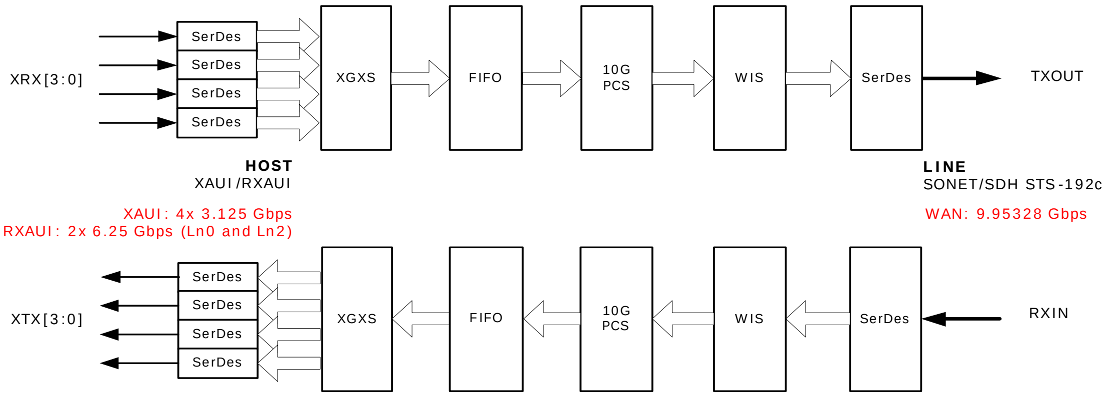
Figure 73 • **10G WAN with 1588**



3.13.4 10G WAN

In 10G WAN mode, the host interface is XAUI (4 × 3.125 Gbps) or RXAUI (2 × 6.250 Gbps), and the line interface is SONET/SDH STS-192c (9.95328 Gbps). A single reference clock input pin, XREFCK, is used by both the line-and host-side interfaces to transmit appropriate rates. For more information about supported reference clock frequencies, see Table 44, page 103. Optionally, WREFCK may be used as the 155.52 MHz reference clock for the line interface. A 622 MHz WREFCK reference frequency is not supported.

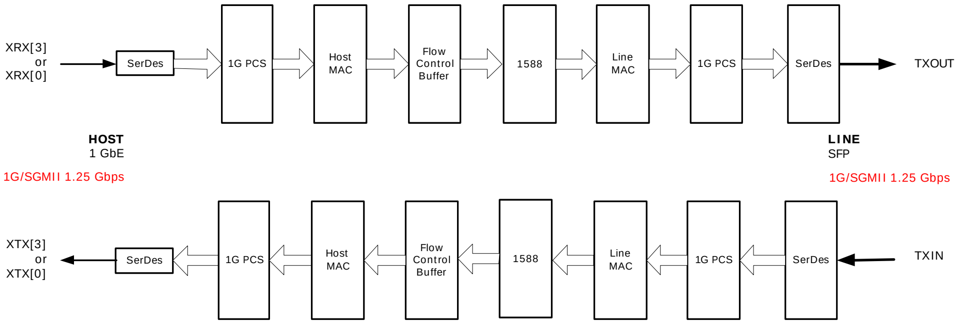
Figure 74 • **10G WAN**



3.13.5 1 GbE with 1588 and MACs

In 1 GbE mode with 1588 and MACs, one XAUI lane on the host interface services the 1 GbE signal (1.25 Gbps). A single reference clock input pin, XREFCK, is used by both the line-and host-side interfaces to transmit appropriate rates. For more information about supported reference clock frequencies, see Table 44, page 103. MACs should be enabled to meet the pause turn around time spec, as defined in the IEEE specifications.

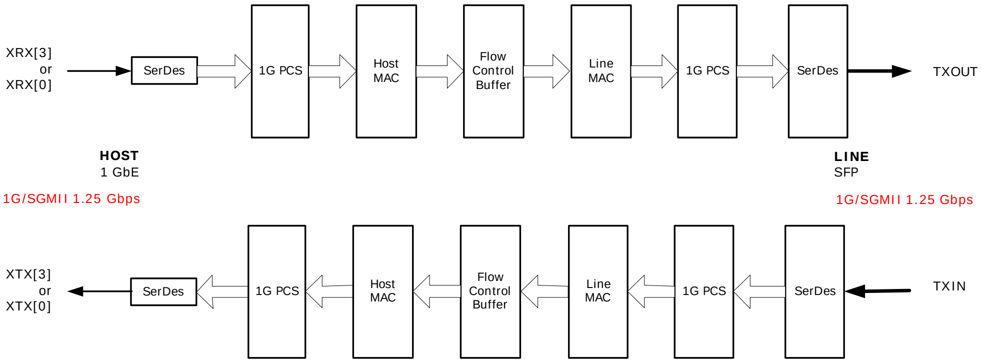
Figure 75 • **1 GbE with 1588 and MACs**



3.13.6 1 GbE

In 1 GbE mode, one XAUI lane on the host interface services the 1 GbE signal (1.25 Gbps). A single reference clock input pin, XREFCK, is used by both the line-and host-side interfaces to transmit appropriate rates. For more information about supported reference clock frequencies, see Table 44, page 103. MACs should be enabled to meet the pause turn around time spec, as defined in the IEEE specifications.

Figure 76 • **1 GbE**



3.14 Management Interfaces

This section contains information about the low-speed serial interfaces of the VSC8489-17 device. The primary control and monitor interfaces in the design are as follows:

- MDIO
- SPI slave
- Two-wire serial (slave)
- Two-wire serial (master)
- Push out SPI master for IEEE 1588 time stamp data
- GPIO
- JTAG

The VSC8489-17 device supports three different interfaces for accessing status and configuration registers: MDIO, SPI slave, and two-wire serial slave. Only one of the interfaces can be active at a time. The VSC8489-17 device doesn't arbitrate between these interfaces.

Note: Users must exercise caution, and ensure that multiple interfaces are not active at the same time. The SPI slave interface is the recommended interface for accessing the status and configuration registers of the 1588 block and the IEEE 1588 time stamp data updates.

3.14.1 MDIO Interface

The MDIO interface in the VSC8489-17 device complies with IEEE 802.3ae Clause 45. For more information, see the IEEE standard. The MDIO management interface consists of a bi-directional data path (MDIO) and a clock reference (MDC).

MDIO instructions can be used to read registers, write registers, and perform post-read-increment-address instructions. Due to its slow bandwidth and high latency, the MDIO interface is not recommended as the only interface to access the VSC8489-17 device.

Note: The maximum data rate of the MDIO interface is 2.5 Mbps.

The PADDR[4:1] pins select the MDIO port addresses to which the VSC8489-17 device will respond. A single VSC8489-17 device requires the use of two MDIO port addresses, one for each channel. The port address transmitted in MDIO read/write commands to access registers in a particular VSC8489-17 channel is shown in the following table. The port address is a function of the PADDR pins and a pre-programmed number indicating the channel number. Up to sixteen VSC8489-17 devices can be controlled by a single MDIO host.

Table 45 • MDIO Port Addresses Per Channel

Channel Number	Channel’s Port Address
1	{PADDR[4:1], 1}
0	{PADDR[4:1], 0}

3.14.1.1 Accessing 32-Bit Data Registers

Even though the MDIO interface is defined to access 16-bit data registers, 32-bit configuration and status registers are present in the line and host MACs, 1588, and line-side SerDes. Use the following steps when accessing registers in 32-bit blocks.

3.14.1.1.1 Write to 32-Bit Register

1. Issue address instruction specifying the MDIO address for bits [31:16].
2. Issue write instruction to write data to register bits [31:16].
3. Issue address instruction specifying the MDIO address for bits [15:0].
4. Issue write instruction to write data to register bits [15:0].

Note: Writing to the two halves of the 32-bit register in the opposite order is not permitted. Nor is it possible to write to only one-half of the register. All four MDIO instructions must be issued to write to a 32-bit register.

3.14.1.1.2 Read 32-Bit Register

1. Issue address instruction specifying the MDIO address for bits [15:0].
2. Issue read-increment instruction. The data read is the contents of register bits [15:0].
3. Issue read instruction. The data read is the contents of register bits [31:16].

Note: Perform all three steps to read a 32-bit register even when reading consecutive addresses. Issuing back-to-back read-increment instructions to read consecutive 32-bit register addresses is not supported.

Register addresses listed for the line and host MACs, 1588, and line-side SerDes apply to the SPI slave and two-wire serial slave interfaces, which support direct access to 32-bit data registers. There are two MDIO addresses for each of these 32-bit data registers: one address to access data bits [31:16] and one address to access data bits [15:0]. Contact Microsemi for support using the MDIO interface to access line and host MACs, 1588, and line-side SerDes registers.

3.14.1.2 MDIO Device and Register Addresses

The VSC8489-17 device registers are arranged according to the MDIO devices as defined in IEEE 802.3 clause 45, as shown in the following list:

- Device 1: PMA and line-side interface registers
- Device 2: WIS registers
- Device 3: 10G PCS, 1G PCS, FC buffers
- Device 4: XGXS PCS and host-side interface registers
- Device 1E: Global, SFP+, PLLs, and 1588 registers

3.14.2 SPI Slave Interface

The VSC8489-17 device supports the serial parallel interface (SPI) for reading and writing registers for high bandwidth tasks such as reading IEEE 1588 time stamp data. The SPI interface is also capable of accessing all status and configuration registers. The SPI slave port consists of a clock input (SCK), data input (MOSI), data output (MISO), and slave select input (SSN).

Note: The SPI slave interface is the recommended interface to access status and configuration registers for the rest of the device.

Drive the SSN pin low to enable the interface. The interface is disabled when SSN is high and MISO is placed into a high impedance state. The VSC8489-17 device captures the state of the MOSI pin on the rising edge of SCK. 56 data bits are captured on the MOSI pin and transmitted on the MISO pin for each SPI instruction. The serial data bits consist of 1 read/write command bit, 23 address bits and 32 register data bits.

The 23-bit addressing scheme consists of a 2-bit channel number, a 5-bit MDIO device number, and a 16-bit register number. For example, the 23-bit register address for accessing the GPIO_0_Config_Status register in channel 1 (device number is 0x1E and register number is 0x0100) is 0x3E0100. The notion of device number conforms to MDIO register groupings. For example, device 2 is assigned to WIS registers.

The following table shows the order in which the bits are transferred on the interface. Bit 55 is transferred first, and bit 0 is transferred last. This sequence applies to both the MOSI and MISO pins.

Table 46 • **SPI Slave Instruction Bit Sequence**

Bit	Name	Description
55	Read/Write	0: Read 1: Write
54:53	Port/Channel Number	00: Port/Channel 0 01: Port/Channel 1 10, 11: Reserved
52:48	Device Number	5 bit device number Bit 4 corresponds to SPI instruction bit 52 Bit 0 corresponds to SPI instruction bit 48

Table 46 • **SPI Slave Instruction Bit Sequence *(continued)***

Bit	Name	Description
47:32	Register Number	16 bit register number Bit 15 corresponds to SPI instruction bit 47 Bit 0 corresponds to SPI instruction bit 32
31:0	Data	32 bit data Bit 31 corresponds to SPI instruction bit 31 Bit 0 corresponds to SPI instruction bit 0

The register data received on the MOSI pin during a write operation is the data value to be written to a VSC8489-17 register. Register data received on the MOSI pin during a read operation is not used, but must still be delivered to the device.

The VSC8489-17 device SPI slave has a pipelined read process. Two read instructions must be sent to read a single register. The first read instruction identifies the register address to be read. The MISO data transmitted on the second read instruction contains the register contents from the address specified in the first instruction. While a pipelined read implementation is not the most efficient use of bandwidth to read a single register, it is very efficient when performing multiple back-to-back reads (as would be the case when reading 1588's TSFIFO_* registers). The second read instruction contains the address for the second register to be read, plus the data read from the first register. The third read instruction contains the address for the third register to be read, plus the data read from the second register. Register reads can continue in this fashion indefinitely. The following illustrations show the situations where back-to-back read instructions are issued.

Figure 77 • **SPI Single Register Read**

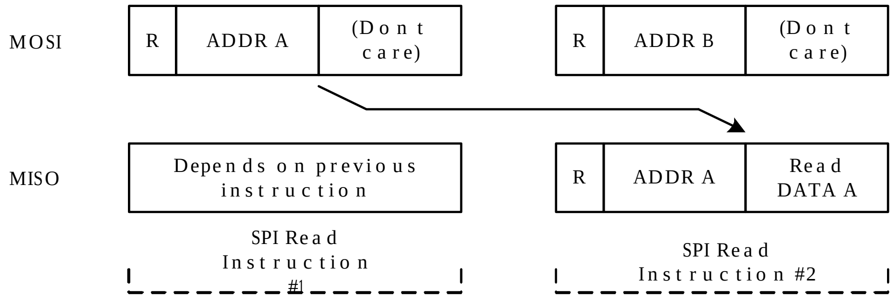
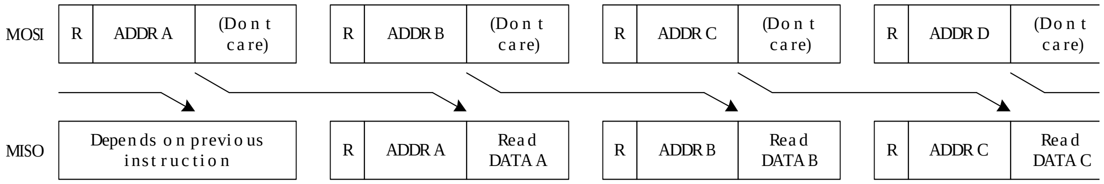
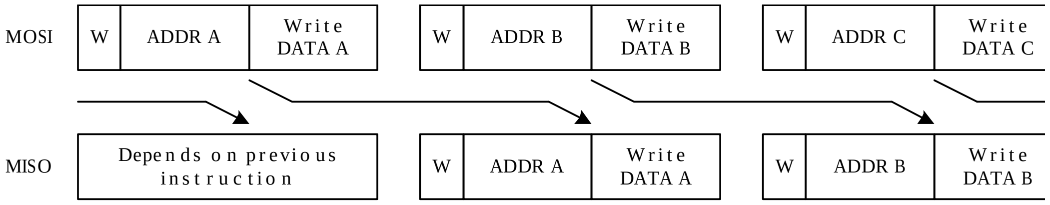


Figure 78 • **SPI Multiple Register Reads**



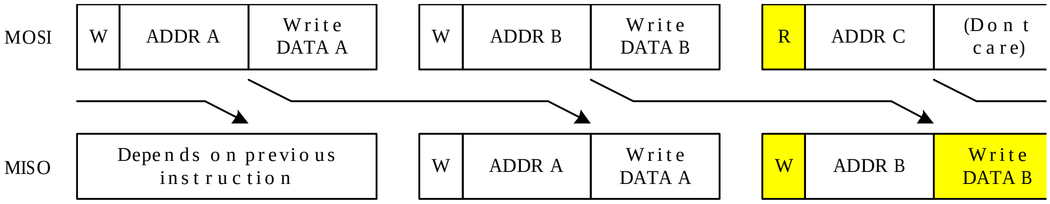
The SPI read instruction illustrations also point out the read/write state and address bits on the MISO output match the information received in the previous instruction. The SPI master could use this data to verify the device captured the previous instruction properly, or simply ignore the data. The following illustration shows the MISO output during write instructions reporting the previous instruction's read/write state, address, and register write data.

Figure 79 • **SPI Multiple Register Writes**



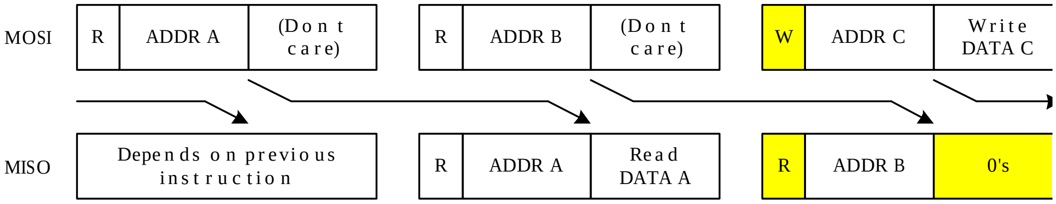
The following illustration shows that when a read instruction followings a write instruction, the MISO data during the read instruction is the data field from the previous write.

Figure 80 • SPI Read Following Write



The following illustration shows that when a write instruction followings a read instruction, the MISO data during the write instruction is not pipelined read data. MISO contains all 0's in the data field.

Figure 81 • SPI Write Following Read



Some VSC8489-17 registers are made up of less than 32 data bits. Any bits not defined for a register will return a 0 when the register is read. Reading an invalid register address will return 0x0.

There is one hazard condition to be aware of when issuing two read instructions to read a single clear-on-read register. Issuing two read instructions internally fetches data twice, even though valid read data is present only in the second instruction. Fetching data also resets a clear-on-read register. The address specified in the second read instruction should be something other than the clear-on-read register address. This prevents an event causing register re-assertion occurring between the two read instructions from being cleared and never detected. The address in the second instruction can be any register not having a clear-on-read function. Device_ID is one example. The same address can be used in each read instruction when continuously polling a clear-on-read register. This works because subsequently fetched data is transmitted from the interface, allowing assertion between reads to be detected. Only the last read instruction where fetched data is not transmitted, should some other address in the instruction be used.

3.14.2.1 MISO Output Timing Modes

MISO changes state when SCK transitions from high to low in the default SPI operating mode. This aids in meeting hold time at the SPI master, assuming the master captures the data on the rising SCK edge. The SPI port can run up to a maximum of 30 Mbps, depending upon the VSC8489-17 device SCK-to-MISO timing, SCK duty cycle, the board layout, and the external SPI master's interface timing requirements. For more information about SPI timing, see Table 72, page 132.

The SPI slave port has an alternate operating mode that allows the interface to run faster. Setting register bit SPI_CTRL.FAST_MODE=1 configures the SPI slave such that MISO changes state when SCK transitions from low to high. Thus, data is both transmitted from the SPI slave and captured by the SPI master on a rising SCK edge. The interface can run faster in this mode by using the entire SCK clock period instead of half the period to transfer data from the slave to the master. Care must be taken to ensure the SPI master's hold time requirement is met. The maximum data transfer rate for the SPI slave in this mode is 30 Mbps. The following illustrations show MISO timing in the default and slave modes.

Figure 82 • **SPI Slave Default Mode**

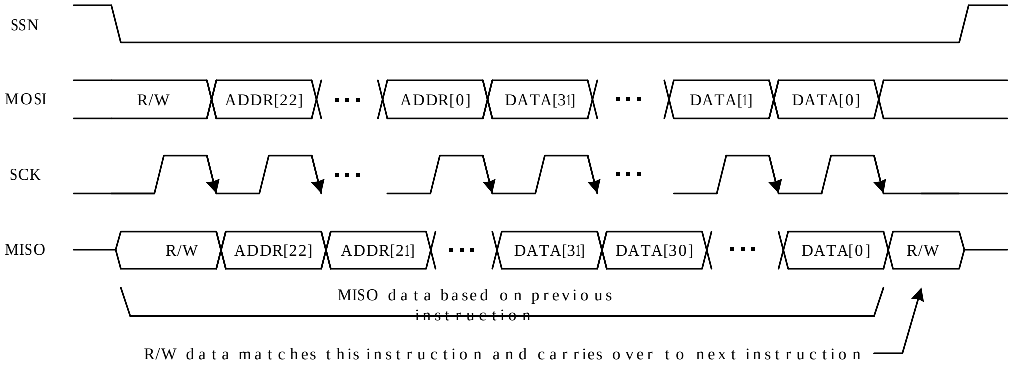
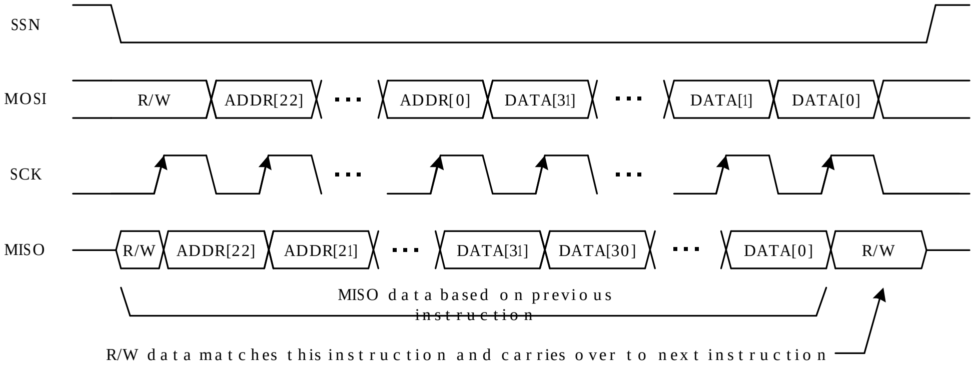


Figure 83 • **SPI Slave Fast Mode**



MISO output timing is the only difference between the two SPI modes. Sampling of MOSI on the rising SCK clock edge remains the same, so writing to the VSC8489-17 device registers is identical in both modes. Thus, the SPI_CTRL.FAST_MODE register setting may be modified using the SPI slave port to change the port's MISO output timing.

3.14.3 Two-Wire Serial (Slave) Interface

The VSC8489-17 device registers may be read and written using a two-wire serial slave interface. The two-wire serial slave SCL and SDA pins are multifunction general purpose I/O (GPIO) pins, GPIO_3 and GPIO_2, respectively. The GPIO pins are configured to serve SCL and SDA functions following device reset.

The slave address assigned to the VSC8489-17 device is a function of four fixed values and the MDIO port address pins. The 7-bit slave address is {1000, PADDR4, PADDR3, PADDR2}. The use of the port address pins allows multiple VSC8489-17 devices to be serviced by a single two-wire serial (master). The maximum data transfer rate for the interface is 400 kbps.

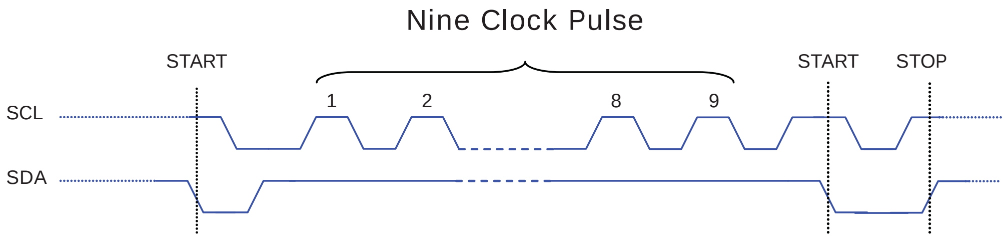
Note: The two-wire serial slave interface does not work with two-wire serial masters using 10-bit slave addresses.

A valid START condition is generated by a two-wire serial master device by transitioning the SDA line from high to low while the SCL line is high. Data is then transferred on the SDA line, most significant bit (MSB) first, with the SCL line clocking data. Data transitions during SCL low periods are valid (read) or latched (write) when SCL pulses high then low. Data transfers are acknowledged (ACK) by the receiving device for data writes and by the master for data reads. An acknowledge is signaled by holding the SDA signal low while pulsing SCL high then low. The master terminates data transfer by generating a STOP condition by transitioning SDA low to high while SCL is high.

Note: If the external two-wire serial master device gets out of sync with the two-wire serial slave interface, the master device must issue a bus reset sequence. This puts the two-wire serial slave interface back into a state that allows it to receive future two-wire serial instructions. The external two-wire serial master device and the two-wire serial slave interface can become out-of-sync and freeze the bus if either device is reset during an instruction.

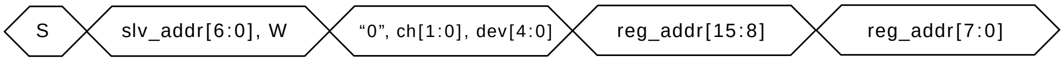
The following illustration shows a two-wire serial bus reset sequence. The reset sequence consists of a START symbol, nine SCK clock pulses while SDA is high, another START symbol, and a STOP symbol.

Figure 84 • **Two-Wire Serial Bus Reset Sequence**



Registers in the VSC8489-17 device are accessed using the 24-bit addressing scheme. The first 8 bits consist of one logic LOW, the channel number (00, 01, 10, 11), and the 5-bit MDIO device number of the register to be accessed. The next 16 bits are the register number. For example, the 24-bit register address for accessing the GPIO_0_Config_Status register in channel 1 (device number 0x1E and register number 0x0100) is 0x3E0100. The notion of device number conforms to MDIO register groupings. For example, device 2 is assigned to WIS registers. The following illustration shows the 24-bit addressing scheme used to access registers.

Figure 85 • **Two-Wire Serial Slave Register Address Format**



An illegal two-wire serial slave read instruction to an invalid channel number, device number, or register address will return a read value of 0x0000 when the slave address matches this device.

Four bytes of data are transferred on the two-wire serial bus after the address when a register is read or written. Data register bits [31:24] are transferred first, followed by bits [23:16], bits [15:8] and finally bits [7:0]. An ACK symbol is sent between each byte of data. Any bits not defined in a register will return a 0 when the register is read.

The following illustration shows the data transferred on the SDA pin during a register write operation. The R/W bit following the slave address is set to logic low to specify a write operation.

Figure 86 • **Two-Wire Serial Write Instruction**

The register address to be accessed is specified by initiating a write operation. After the slave address and three register address bytes are sent to the VSC8489-17 device, a START condition must be re-sent, followed by the slave address with the read/write bit set to logic high. The four-byte data register contents are then transmitted from the VSC8489-17 device. The two-wire serial (master) sends NO ACK after the fourth data byte to indicate it has finished reading data. The following illustration shows data transferred on the SDA pin during a register read operation.

Figure 87 • **Two-Wire Serial Read Instruction**

The two-wire serial slave interface supports sequential read and sequential write instructions.

3.14.4 Two-Wire Serial (Master) Interface

A two-wire serial master interface in the VSC8489-17 channel is available for SFP+/XFP module management. A two-wire serial master interface per channel is required, because the slave address in the optics modules are identical. Two-wire serial interface instructions used to access optics module registers are initiated by writing to VSC8489-17 registers. The two-wire serial interface busses are brought out through GPIO pins. Channel 0's two-wire serial interface is enabled by configuring GPIO_6 to function as SDA and GPIO_7 to function as SCL. Channel 1's two-wire serial interface is enabled by configuring GPIO_10 to function as SDA and GPIO_11 to function as SCL.

The two-wire serial master interface must be configured before initiating any instructions. The slave ID to be transmitted in the first byte of every instruction is selectable in the SLAVE_ID register. The default setting is 0x50. The interface's data rate is determined by the PRESCALE register. The default data rate is 400 kbps.

The two-wire serial master transmits instructions for slave devices with 8-bit data registers and 256 register addresses per slave ID. Always read register I2C_BUS_STATUS.I2C_BUS_BUSY or I2C_READ_STATUS_DATA.I2C_BUS_BUSY to verify the previous instruction has finished prior to initiating a new instruction. Instructions initiated when the interface is busy will be ignored. Both registers report the same interface busy status. The same busy status is reported in two registers for user convenience.

The two-wire serial master initiates a write instruction when the I2C_WRITE_CTRL register is written. The value written to I2C_WRITE_CTRL.WRITE_ADDR is the register address to be modified in the slave device. The value written to I2C_WRITE_CTRL.WRITE_DATA is the data to be written to the slave device's register. The I2C_BUS_STATUS register reports the status of the write instruction. I2C_BUS_STATUS.I2C_BUS_BUSY indicates when the instruction has finished. I2C_BUS_STATUS.I2C_WRITE_ACK=1 means the two-wire serial master received ACKs from the slave at appropriate times. I2C_BUS_STATUS.I2C_WRITE_ACK is cleared each time a new instruction is issued. If the two-wire serial master did not receive ACKs from the slave at appropriate times (I2C_BUS_STATUS.I2C_WRITE_ACK=0), the interface is likely stuck in a state waiting for the ACK. Writing a 1 to the BLOCK_LEVEL_RESET1.I2CM_RESET register will reset the two-wire serial master and release it from its stuck state. The slave device should then be put into a known state by writing any value to the I2C_RESET_SEQ register. The two-wire serial master issues a bus reset sequence when this register is written. For more information, see Two-Wire Serial (Slave) Interface, page 111.

The two-wire serial master initiates a read instruction when the I2C_READ_ADDR register is written. The value written to I2C_READ_ADDR.READ_ADDR is the register address to be accessed in the slave device. I2C_READ_STATUS_DATA.READ_DATA contains the data read from the slave device. READ_DATA is not valid until I2C_READ_STATUS_DATA.I2C_BUS_BUSY=0 to indicate the instruction completed. The two-wire serial master does not support read-increment instructions.

3.14.5 Push Out SPI Master Interface

To overcome MDIO speed limitations for faster or large amounts of time stamp reads, the VSC8489-17 device supports a push out SPI master interface. The SPI output is used to push out time stamp information to an external device only, and does not provide read/write to the rest of the status and configuration registers. For more information, see Serial Time Stamp Output Interface, page 89.

The push out SPI master interface consist of an SPI clock output (SPI_CLK), an SPI data output (SPI_DO), and an SPI chip select output, (SPI_CS).

3.14.6 GPIO

General purpose input/output (GPIO) pins in the VSC8489-17 device serve multiple functions. The GPIO pins are bidirectional where the driver portion is an open-drain buffer. The following table shows the functions that each pin supports and the registers used to configure the pin functions. Leave GPIO pins unconnected when not in use. When configured as output, they are open-drained and a pull-up is required.

Table 47 • GPIO Functions

Pin	Configuration Registers	Functions
GPIO_0	GPIO_0_Config_Status GPIO_0_Config2	Traditional I/O (default) Observed internal signals MOD_ABS_Channel_0 PMTICK ROSI frame pulse 0 Tx Activity LED WIS_INTB 1588-1PPS channel 0 Leave unconnected when not used
GPIO_1	GPIO_1_Config_Status GPIO_1_Config2	Traditional I/O (default) Observed internal signals ROSI_CLK_0 Rx Activity LED WIS_INTA 1588-Load/Save Leave unconnected when not used
GPIO_2	GPIO_2_Config_Status GPIO_2_Config2	Traditional I/O Observed internal signals Slave two-wire serial - SDA (default) ROSI_DATA_0 Tx Activity LED WIS_INTB
GPIO_3	GPIO_3_Config_Status GPIO_3_Config2	Traditional I/O Observed internal signals Slave two-wire serial - SCL (default) TOSI_FRAME_PULSE_0 Rx Activity LED WIS_INTB
GPIO_4	GPIO_4_Config_Status GPIO_4_Config2	Traditional I/O (default) Observed internal signals TOSI_CLK_0 Tx Activity LED WIS_INTB 1588-1PPS channel 1
GPIO_5	GPIO_5_Config_Status GPIO_5_Config2	Traditional I/O (default) Observed internal signals TOSI_INPUT_0 Rx Activity LED WIS_INTA 1588-PPS RI Leave unconnected when not used
GPIO_6	GPIO_6_Config_Status GPIO_6_Config2	Traditional I/O (default) Observed internal signals Ch0 SFP I2C SDA ROSI_FRAME_PULSE_1 Tx Activity LED WIS_INTB

Table 47 • **GPIO Functions *(continued)***

Pin	Configuration Registers	Functions
GPIO_7	GPIO_7_Config_Status GPIO_7_Config2	Traditional I/O (default) Observed internal signals Ch0 SFP I2C SCL ROSI_CLK_1 Rx Activity LED WIS_INTA
GPIO_8	GPIO_8_Config_Status GPIO_8_Config2	Traditional I/O (default) Observed internal signals MOD_ABS_Channel_0 PMTICK ROSI_DATA_1 Tx Activity LED WIS_INTA
GPIO_9	GPIO_9_Config_Status GPIO_9_Config2	Traditional I/O (default) Observed internal signals Mod_ABS channel 1 PMTICK TOSI_FRAME_PULSE_1 Rx Activity LED WIS_INTA 1588-1PPS channel 1
GPIO_10	GPIO_10_Config_Status GPIO_10_Config2	Traditional I/O (default) Observed internal signals Ch1 SFP I2C SDA TOSI_CLK_1 Tx Activity LED WIS_INSTB
GPIO_11	GPIO_11_Config_Status GPIO_11_Config2	Traditional I/O (default) Observed internal signals Ch1 SFP I2C SCL TOSI_INPUT_1 Rx Activity LED WIS_INTA 1588-1PPS channel 1
GPIO_12	GPIO_12_Config_Status GPIO_12_Config2	Traditional I/O (default) Observed internal signals Tx Activity LED WIS_INTA
GPIO_13	GPIO_13_Config_Status GPIO_13_Config2	Traditional I/O (default) Observed internal signals Rx Activity LED WIS_INTA
GPIO_14	GPIO_14_Config_Status GPIO_14_Config2	Traditional I/O (default) Observed internal signals Tx Activity LED WIS_INTA
GPIO_15	GPIO_15_Config_Status GPIO_15_Config2	Traditional I/O (default) Observed internal signals Rx Activity LED WIS_INTA

When a GPIO pin is programmed to be a traditional I/O, the pin may be driven high or low. It may also serve as an input and an LED driver capable of blinking at various rates. An interrupt pending register may optionally be asserted when the pin is in input mode and the pin changes state. All of these functions are configured using the pin configuration register settings shown in the preceding table.

The GPIO pin's output driver is automatically enabled when the pin function is set to observe internal signals. The second configuration register listed for each pin selects which internal signal is transmitted from the pin.

3.14.7 JTAG

The VSC8489-17 device has a IEEE 1149.1–2001 compliant JTAG interface. The following table shows the supported instructions and corresponding instruction register codes. The code's least significant bit is shifted into TDI first when loading an instruction (the 0 is shifted in first when loading the IDCODE instruction).

Table 48 • **JTAG Instructions and Register Codes**

Instruction	Register Code	Notes
IDCODE	111111111111111111111110	
BYPASS	111111111111111111111111	
EXTEST	111111111111111111101000	
EXTEST_PULSE	111111101111111111101000	
EXTEST_TRAIN	111111011111111111101000	
SAMPLE	111111111111111111111000	
PRELOAD	111111111111111111111000	
LV_HIGHZ	111111111111111111001111	Provides the ability to place outputs in a high impedance state to facilitate manufacturing test and PC board diagnostics. The XAUI and SFP+ serial data outputs are not put into the high impedance state when this instruction is loaded in the JTAG TAP controller.
CLAMP	111111111111111111101111	Provides the ability to place all outputs in a predefined state when the scan process is being used to test other devices on a PC board.

The RESETN pin must be driven to logic high before shifting data out of the DEVICE ID data register when the IDCODE instruction is loaded in the JTAG TAP controller.

4 Registers

Information about the registers for this product is available in the attached Adobe Acrobat file. To view or print the information, double-click the attachment icon.

5 Electrical Specifications

This section provides the DC characteristics, AC characteristics, recommended operating conditions, and stress ratings for the VSC8489-17 device.

5.1 DC Characteristics

This section contains the DC specifications for the VSC8489-17 device.

5.1.1 DC Inputs and Outputs

The following table lists the DC specifications for the LVTTL inputs and outputs for the VSC8489-17 device. The LVTTL inputs are 3.3 V tolerant.

Table 49 • **LVTTL Input and Push/Pull Output DC Characteristics**

Parameter	Symbol	Minimum	Maximum	Unit	Condition
Output high voltage, LVTTL	V_{OH_TTL}	1.8	V_{DDTTL}	V	V_{DDTTL} = 2.5 V and I_{OH} = –4 mA
Output low voltage, LVTTL	V_{OL}		0.5	V	V_{DDTTL}/V_{DDMDIO} = 2.5 V and I_{OL} = 4 mA
Input high voltage	V_{IH}	1.7	V_{DDTTL}	V	V_{DDTTL}/V_{DDMDIO} = 2.5 V
Input low voltage	V_{IL}		0.8	V	V_{DDTTL}/V_{DDMDIO} = 2.5 V
Input high current	I_{IH}		500	µA	$V_{IH} = V_{DDTTL}/V_{DDMDIO}$
Input low current	I_{IL}	–100		µA	V_{IL} = 0 V

Table 50 • **LVTTLOD Input and Open-Drain Output DC Characteristics**

Parameter	Symbol	Minimum	Maximum	Unit	Condition
Output high voltage, open drain	V_{OH_OD}	See note[1]	V_{DDTTL}	V	V_{DDTTL}/V_{DDMDIO} = 2.5 V and I_{OH} = –4 mA
Input high leakage current, open drain	I_{OZH}		100	µA	
Output low voltage, open drain	V_{OL}		0.5	V	V_{DDTTL}/V_{DDMDIO} = 2.5 V and I_{OL} = 4 mA
Input high voltage	V_{IH}	1.7	V_{DDTTL}	V	V_{DDTTL}/V_{DDMDIO} = 2.5 V
Input low voltage	V_{IL}		0.8	V	V_{DDTTL}/V_{DDMDIO} = 2.5 V
Input high current	I_{IH}		500	µA	$V_{IH} = V_{DDTTL}/V_{DDMDIO}$
Input low current	I_{IL}	–100		µA	V_{IL} = 0 V

1. Determined by the loading current of the other devices connecting to this pin, the I_{OZH} current of this pin, and the value of the pull-up resistor used.

5.1.2 Reference Clock

The following table lists the DC specifications for the reference clock for the VSC8489-17 device.

Table 51 • **Reference Clock DC Characteristics**

Parameter	Symbol	Minimum	Maximum	Unit	Condition
XREFCK/WREFCK differential input swing, low[1]	$\Delta V_{I\_DIFF\_LOW}$	200	1200	mV_{P-P}	CML reference clock input
XREFCK/WREFCK differential input swing, high[1]	$\Delta V_{I\_DIFF\_HIGH}$	1100	2400	mV_{P-P}	LVPECL reference clock input
SREFCK differential input swing	ΔV_{I_DIFF}	200	2400	mV_{P-P}	

1. An API call is used to set the input swing to be high or low.

5.2 AC Characteristics

This section contains the AC specifications for the VSC8489-17 device. The specifications apply to all channels. All the XAUI/RXAUI/SFI I/Os should be AC-coupled and work in differential.

5.2.1 Receiver Specifications

The specifications in the following table correspond to line-side 10G receiver input, SFI point D. Point D assumes that the input is from a compliant point C output and a compliant SFI or XFI channel according to the SFP+ standard (SFF-8431) or the XFP multisource agreement (INF-8077i). The measurement is done with a 9 dB channel loss unless stated otherwise. The jitter and amplitude measurements are calibrated at point C”, as specified in SFF-8431 revision 4.1.

Table 52 • **Line-Side 10G Receiver Input (SFI Point D 9.95328G) AC Characteristics**

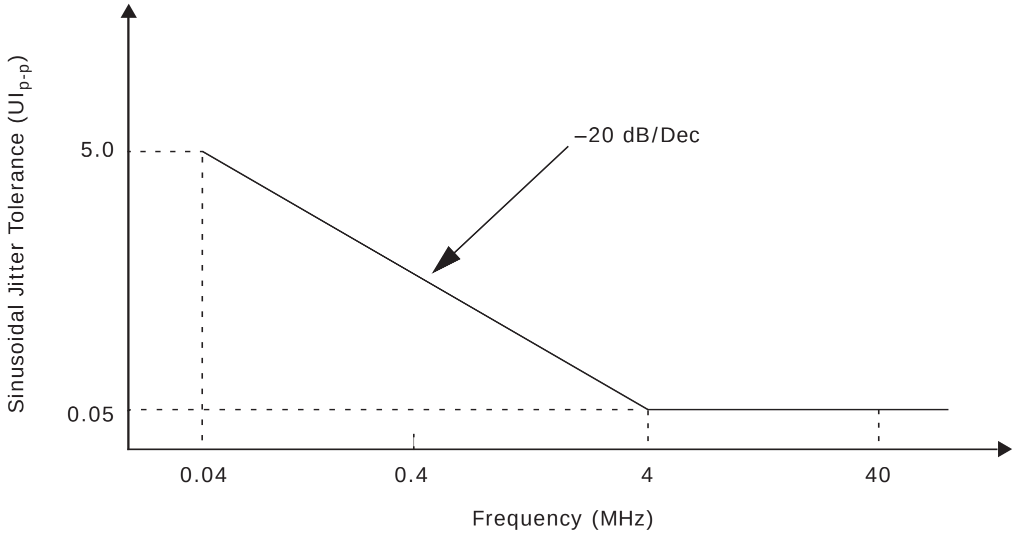
Parameter	Symbol	Minimum	Typical	Maximum	Unit	Condition
RXIN input data rate, 10 Gbps		9.95328 – 100 ppm	10.3125	10.3125 + 100 pp m	Gbps	10 Gbps LAN/WAN mode
RXIN linear mode differential input data swing	$\Delta VRXIN_{LINEAR}$	180		600	mV	Voltage modulation amplitude (VMA)
RXIN limiting mode differential input data swing	$\Delta VRXIN_{LIMITING}$	300		850	mV	Measured peak-to-peak
RXIN AC common-mode voltage	V_{CM}			15	mV_{RMS}	
Differential return loss	RL_{SDD11}			–12	dB	0.01 GHz to 2.0 GHz
Differential return loss	RL_{SDD11}			$-6.68 + 12.1 \times \log_{10}(f/5.5)$	dB	2.0 GHz to 11.1 GHz
Reflected differential to common-mode conversion	RL_{SCD11}			–10	dB	0.1 GHz to 11.1 GHz
99% jitter	$99\%_{JIT\_p-p}$			0.42	UI	
Pulse width shrinkage jitter	$DDPWS_{JIT\_p-p}$			0.3	UI	
Total jitter tolerance	TOL_{JIT_P-P}			0.70	UI	

Table 52 • **Line-Side 10G Receiver Input (SFI Point D 9.95328G) AC Characteristics *(continued)***

Parameter	Symbol	Minimum	Typical	Maximum	Unit	Condition
Eye mask X1	X1			0.35	UI	
Eye mask Y1	Y1	150			mV	
Eye mask Y2	Y2			425	mV	
Waveform distortion penalty	WDPc			9.3	dBe	BER 1E–12. This parameter of DAC is measured with 7 dB SFI channel loss.
Voltage modulation amplitude	VMA	180			mV	BER 1E–12. This parameter of DAC is measured with 7 dB SFI channel loss.
Optical sensitivity (ROP), back-to-back, 10.3 Gbps	S_{B2B}			–24	dBm	BER 1E–12, PRBS31 and 10 GbE frame, 5.76 dB SFI channel loss.
Optical sensitivity (ROP), with fiber plant, 10.3 Gbps	S_{FIBER}			–21	dBm	95 km single-mode fiber, BER 1E–12, PRBS31 and 10 GbE frame, 5.76 dB SFI channel loss.
Chromatic dispersion penalty	F_{CDP}		1.5	3	dB	1600 ps/nm, 5.76 dB SFI channel loss.
OSNR vs BER with fiber plant, 10.3 Gbps	$OSNR_{FEC}$	16			dB	95 km single-mode fiber, BER 7E–4, 5.76 dB SFI channel loss.

The following illustration shows the sinusoidal jitter tolerance for the SFI datacom.

Figure 88 • **SFI Datacom Sinusoidal Jitter Tolerance**



The following table lists the 10G input jitter specifications for the VSC8489-17 device.

Table 53 • **Line-Side SONET 10G Input Jitter AC Characteristics**

Parameter	Symbol	Minimum	Typical	Maximum	Unit	Condition
RXIN input data rate, 10 Gbps WAN		9.95328 – 100 ppm	9.95328	9.95328 + 100 ppm	Gbps	
Sinusoidal jitter tolerance, 9.95 Gbps	SJ_T	1.5x jitter mask				GR-253 according to SONET OC-192 standard

The host-side 6.25 Gbps receiver operating in RXAUI mode complies with the AC characteristics specified for CEI-6G-SR interfaces according to OIF-CEI-02.0.

Table 54 • **Host-Side RXAUI Receiver AC Characteristics**

Parameter	Symbol	Minimum	Maximum	Unit	Condition
Data rate		6.25 – 100 ppm	6.25 + 100 ppm	Gbps	
Differential peak-to-peak input voltage	V_{I_DIFF}	125	750	mV	AC-coupled, measured peak-to-peak each side (both sides driven)
Differential input return loss	RLI_{SDD11}		–8	dB	100 MHz to 4.6875 GHz
Differential input return loss	RLI_{SDD11}		–8 + 16.6 x log(f/4.6875)	dB	4.6875 GHz to 6.25 GHz
Common-mode return loss	RL_{SCC11}		–6	dB	100 MHz to 4.6875 GHz
Random jitter	RJ		0.15	UI_{P-P}	
Uncorrelated bounded high-probability jitter	UBHPJ		0.15	UI_{P-P}	
Correlated bounded high-probability jitter	CBHPJ		0.30	UI_{P-P}	
Total jitter	TJ		0.60	UI_{P-P}	
Eye mask X1	R_X1		0.30	UI_{P-P}	
Eye mask Y1	R_Y1	62.5		mV	
Eye mask Y2	R_Y2		375	mV	

The following table lists the host-side 3.125 Gbps receiver characteristics when operating in XAUI mode following IEEE 802.3 clauses 47, 54, and 71.

Table 55 • **Host-Side XAUI Receiver AC Characteristics**

Parameter	Symbol	Minimum	Maximum	Unit	Condition
Data rate		3.125 – 100 ppm	3.125 + 100 ppm	Gbps	
Differential peak-to-peak input voltage	V_{I_DIFF}	75	1600	mV	AC-coupled, measured peak-to-peak each side (both sides driven)
Differential input return loss	RLI_{SDD11}		–10	dB	100 MHz to 2.5 GHz
Common-mode return loss	RL_{SCC11}		–6	dB	100 MHz to 2.5 GHz
Random jitter	RJ		0.18	UI_{P-P}	

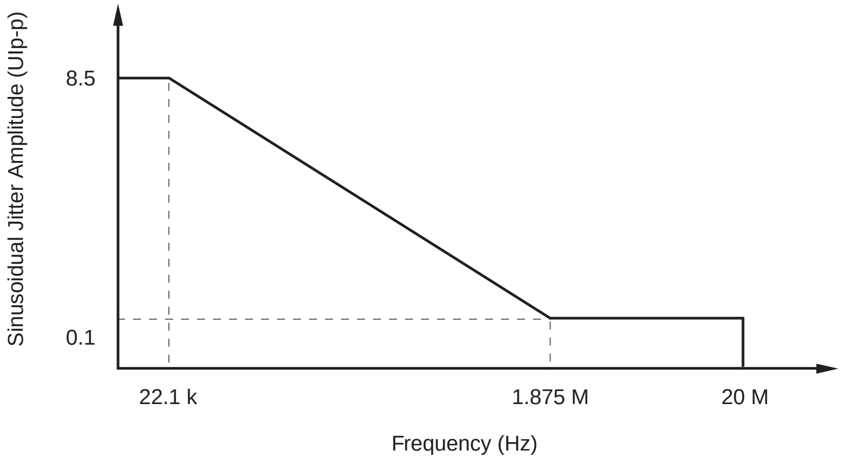
Table 55 • Host-Side XAUI Receiver AC Characteristics *(continued)*

Parameter	Symbol	Minimum	Maximum	Unit	Condition
Deterministic jitter	DJ		0.37	UI_{P-P}	
Total jitter tolerance[1]	TJ		0.65	UI_{P-P}	

1. Total jitter includes sinusoidal jitter according to IEEE 802.3 clause 47.3.4.6.

The following illustration shows the sinusoidal jitter tolerance for the XAUI receiver input.

Figure 89 • XAUI Receiver Input Sinusoidal Jitter Tolerance



The following table lists the line-side 1.25 Gbps SFI input specifications for the VSC8489-17 device.

Table 56 • Line-Side 1.25 Gbps SFI Input AC Characteristics

Parameter	Symbol	Minimum	Typical	Maximum	Unit	Condition
RXIN input data rate, 1.25 Gbps		1.25 – 100 ppm	1.25	1.25 + 100 ppm	Gbps	1.25 Gbps mode
Differential input return loss	RLI_{SDD11}			–10	dB	50 MHz to 625 MHz
Differential input return loss	RLI_{SDD11}			–10 + 10 x log(f/625 MHz)	dB	625 MHz to 1250 MHz
Total jitter tolerance	TJ_T			0.749	UI	Jitter above 637 kHz (IEEE 802.3 clause 38.5)
Deterministic jitter	DJ			0.462	UI_{P-P}	Jitter above 637 kHz (IEEE 802.3 clause 38.5)
Eye mask Y1	Y1	125			mV	
Eye mask Y2	Y2			600	mV	

The host-side 1.25 Gbps receiver operating in 1000BASE-KX mode complies with IEEE 802.3 clause 70.

Table 57 • **Host-Side 1.25 Gbps (1000BASE-KX) Receiver Input AC Characteristics**

Parameter	Symbol	Minimum	Maximum	Unit	Condition
Data rate		1.25 – 100 ppm	1.25 + 100 ppm	Gbps	
Differential input return loss	RLI_{SDD11}		–10	dB	50 MHz to 625 MHz
Differential input return loss	RLI_{SDD11}		–10 + 10 x log(f/625 MHz)	dB	625 MHz to 1250 MHz
Total jitter tolerance[1]	TOL_{TJ}		0.749	UI	Measured according to IEEE 802.3 clause 38.5
Deterministic jitter tolerance[1]	TOL_{DJ}		0.462	UI	Measured according to IEEE 802.3 clause 38.5

1. Jitter requirements represent high-frequency jitter (above 637 kHz) and not low-frequency jitter or wander.

5.2.2 Transmitter Specifications

This section includes the transmitter specifications.

The specifications in the following table correspond to line-side 10G transmitter output, SFI point B. Point B is after a standard-compliant SFI or XFI channel, as defined in the SFP+ standard (SFF-8431) or the XFP multisource agreement (INF-8077i). The measurement is done with a 9 dB channel loss unless stated otherwise.

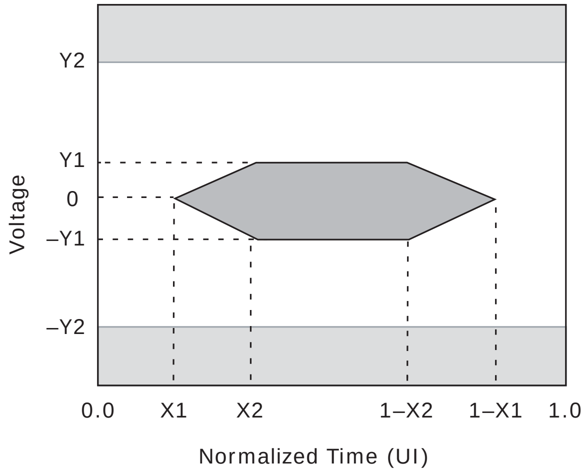
Table 58 • **Line-Side 10G Transmitter Output (SFI Point B) AC Characteristics**

Parameter	Symbol	Minimum	Maximum	Unit	Condition
Termination mismatch	ΔZ_M		5	%	
AC common-mode voltage	V_{OCM_AC}		15	mV_{RMS}	
Differential return loss	SDD22		–12	dB	0.01 GHz to 2.0 GHz
Differential return loss	SDD22		See note[1]	dB	2.0 GHz to 11.1 GHz
Common-mode return loss	SCC22		See note[2]	db	0.01 GHz to 2.5 GHz
Common-mode return loss	SCC22		–3	db	2.5 GHz to 11.1 GHz
Total jitter	TJ		0.28	UI	
Data-dependent jitter	DDJ		0.1	UI	
Pulse shrinkage jitter	DDPWS		0.055	UI	
Uncorrelated jitter	UJ		0.023	UI_{RMS}	
Eye mask X1	X1		0.12	UI	
Eye mask X2	X2		0.33	UI	
Eye mask Y1	Y1	95		mV	
Eye mask Y2	Y2		350	mV	

1. Reflection coefficient given by the equation SDD22(dB) = –6.68 + 12.1 Log_{10}(f/5.5), with f in GHz.
2. S-parameter equation SCC22(dB) = -7 + 1.6 × f, with f in GHz.

The following illustration shows the compliance mask associated with the Tx SFI transmit differential output.

Figure 90 • **SFI Transmit Differential Output Compliance Mask**



The following table shows the transmit path output specifications for SFI point B with 7 dB SFI channel loss.

Table 59 • **Transmitter SFP+ Direct Attach Copper Output AC Characteristics**

Parameter	Symbol	Minimum	Maximum	Unit	Condition
SFP+ direct attach copper voltage modulation amplitude, peak-to-peak	V_{MA}	300		mV	See SFF-8431 section D.7.
SFP+ direct attach copper transmitter Q_{SQ}	Q_{SQ}	63.1			See SFF-8431 section D.8.
SFP+ direct attach copper output AC common-mode voltage			12	mV (RMS)	See SFF-8431 section D.15.
SFP+ direct attach copper host output TWDPc	TWDPc		10.7	dB	Electrical output measured using SFF-8431 Appendix G, including copper direct attach stressor.

The following table shows that the 10 Gbps transmitter operating in 10GBASE-KR mode complies with IEEE 802.3 clause 72.7.

Table 60 • **10 Gbps Transmitter 10GBASE-KR AC Characteristics**

Parameter	Symbol	Minimum	Maximum	Unit	Condition
Signalling speed	T_{BAUD}	10.3125 – 100 ppm	10.3125 + 100 ppm	Gbps	
Differential output return loss	RLO_{SDD22}	9 9 – 12 x log(f/2.5)		dB	50 MHz to 2.5 GHz 2.5 GHz to 7.5 GHz R_L = 100 Ω ± 1%
Common mode return loss	RLO_{CM}	6 6 – 12 x log(f/2.5)		dB	50 MHz to 2.5 GHz 2.5 GHz to 7.5 GHz R_L = 100 Ω ± 1%
Transition time	T_R, T_F	24	47	ps	20% to 80%

Table 60 • **10 Gbps Transmitter 10GBASE-KR AC Characteristics *(continued)***

Parameter	Symbol	Minimum	Maximum	Unit	Condition
Random jitter	RJ		0.15	UI	BER 1E–12
Deterministic jitter	DJ		0.15	UI	
Duty cycle distortion (part of DJ)	DCD		0.035	UI	
Total jitter	TJ		0.28	UI	

The following table shows the transmit path SONET jitter specifications for point A, measured with register optimization and using a clock rate of 156.25 MHz or 155.52 MHz.

Table 61 • **Line-Side SONET 10G Output Jitter AC Characteristics**

Parameter	Symbol	Maximum	Unit
Total jitter, 20 kHz to 80 MHz	TJ	150	mUI
Total jitter, 4 MHz to 80 MHz	TJ	80	mUI

The near-end 6.25 Gbps transmitter output operating in RXAUI mode complies with the AC characteristics specified for CEI-6G-SR interfaces according to OIF-CEI-02.0.

Table 62 • **Near-end RXAUI Transmitter Output AC Characteristics**

Parameter	Symbol	Minimum	Maximum	Unit	Condition
Data rate		6.25 – 100 ppm	6.25 + 100 ppm	Gbps	
Differential output return loss	RLO_{SDD22}		–8	dB	100 MHz to 4.6875 GHz
Differential output return loss	RLO_{SDD22}		–8 + 16.6 x log(f/4.6875)	dB	4.6875 GHz to 6.25 GHz
Common-mode output return loss	RLO_{SCC22}		–6	dB	100 MHz to 4.6875 GHz
Rise time and fall time	t_R, t_F	30	130	ps	20% to 80%
Uncorrelated bounded high-probability jitter	UBHPJ		0.15	UI_{P-P}	
Duty cycle distortion	DCD		0.05	UI_{P-P}	
Total jitter	TJ		0.30	UI_{P-P}	
Eye mask X1	X1		0.15	UI_{P-P}	
Eye mask X2	X2		0.40	UI_{P-P}	
Eye mask Y1	Y1	200		mV	
Eye mask Y2	Y2		375	mV	

The far-end 6.25 Gbps transmitter output operating in RXAUI mode complies with the AC characteristics specified for CEI-6G-SR interfaces according to OIF-CEI-02.0.

Table 63 • **Far-end RXAUI Transmitter Output AC Characteristics**

Parameter	Symbol	Minimum	Maximum	Unit	Condition
Uncorrelated bounded high-probability jitter	UBHPJ		0.15	UI_{P-P}	

Table 63 • **Far-end RXAUI Transmitter Output AC Characteristics *(continued)***

Parameter	Symbol	Minimum	Maximum	Unit	Condition
Correlated bounded high-probability jitter	CBHPJ		0.30	UI_{P-P}	
Total jitter	TJ		0.60	UI_{P-P}	
Eye mask X1	R_X1		0.30	UI_{P-P}	
Eye mask Y1	R_Y1	62.5		mV	
Eye mask Y2	R_Y2		375	mV	

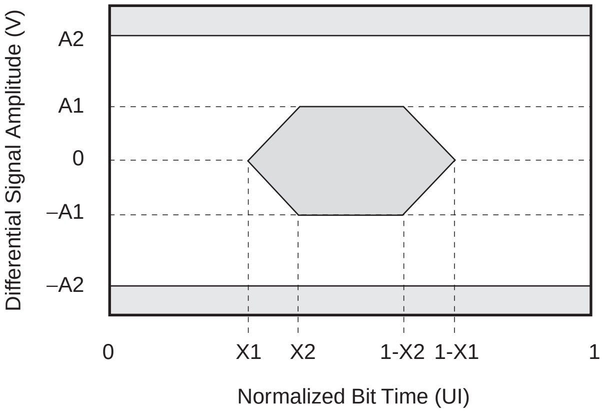
The following table lists the far-end XAUI output specifications for the VSC8489-17 device.

Table 64 • **Far-end XAUI Transmitter Output AC Characteristics**

Parameter	Symbol	Minimum	Maximum	Unit	Condition
Data rate		3.125 – 100 ppm	3.125 + 100 ppm	Gbps	
Differential output voltage	V_{OUT_DIFF}	600	1600	mV	Near-end
Differential output return loss	RLO_{SDD11}		–10	dB	312.5 MHz to 625 MHz
Differential output return loss	RLO_{SDD11}		–10 + 10 x log(f/625 MHz)	dB	625 MHz to 3.125 GHz
Rise time and fall time	t_R, t_F	60	130	ps	20% to 80%
Total jitter	TJ		0.55	UI	
Deterministic jitter	DJ		0.37	UI	
Eye mask X1	X1		0.275	UI	
Eye mask X2	X2		0.4	UI	
Eye mask A1	A1	100		mV	
Eye mask A2	A2		800	mV	

The following illustration shows the compliance mask for the XAUI output.

Figure 91 • **XAUI Output Compliance Mask**



The following table lists the line-side 1.25 Gbps SFI output specifications for the VSC8489-17 device.

Table 65 • **Line-Side 1.25 Gbps SFI Output AC Characteristics**

Parameter	Symbol	Minimum	Maximum	Unit	Condition
Differential output return loss	RLO_{SDD22}		–10	dB	50 MHz to 625 MHz
Differential output return loss	RLO_{SDD22}		–10 + 10 x log(f/625 MHz)	dB	625 MHz to 1250 MHz
Common mode return loss	RLO_{CM}		–6	dB	50 MHz to 625 MHz
Deterministic jitter	DJ		0.1	UI	Measured according to IEEE 802.3 clause 38.5
Total jitter	TJ		0.24	UI	Measured according to IEEE 802.3 clause 38.5
Eye mask Y1	Y1	150		mV	SFF-8431 1G specification
Eye mask Y2	Y2		500	mV	SFF-8431 1G specification

The host-side transmitter operating in 1000BASE-KX mode complies with IEEE 802.3 clause 70.

Table 66 • **Host-Side Transmitter 1000BASE-KX AC Characteristics**

Parameter	Symbol	Minimum	Maximum	Unit	Condition
Data rate		1.25 – 100 ppm	1.25 + 100 ppm	Gbps	
Differential output return loss	RLO_{SDD22}		–10	dB	50 MHz to 625 MHz
Differential output return loss	RLO_{SDD22}		–10 + 10 x log(f/625 MHz)	dB	625 MHz to 1250 MHz
Random jitter	RJ		0.15	UI_{P-P}	At BER 10^{-12}
Deterministic jitter	DJ		0.10	UI_{P-P}	
Total jitter	TJ		0.25	UI_{P-P}	

5.2.3 Timing and Reference Clock

The following table lists the reference clock specifications (XREFCK, SREFCK, WREFCK, and CLK1588) for the VSC8489-17 device.

Table 67 • **Reference Clock AC Characteristics**

Parameter	Symbol	Minimum	Typical	Maximum	Unit	Condition
XREFCK, SREFCK, and WREFCK frequency[1]	f_{REFCLK}	120		156.25	MHz	
XREFCK, SREFCK, and WREFCK frequency accuracy[1]	f_R	– 100 ppm		100 ppm	MHz	
Rise time and fall time	t_R, t_F			0.4	ns	Within ± 200 mV relative to VDD x 2/3
XREFCK and WREFCK Clock duty cycle	DC	40		60	%	At 50% threshold

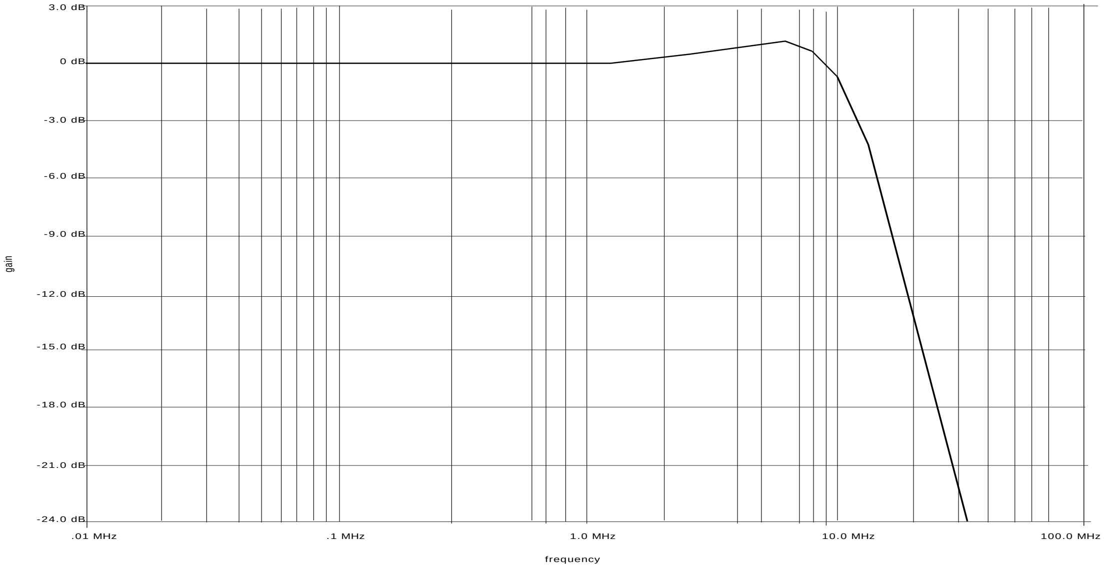
Table 67 • **Reference Clock AC Characteristics *(continued)***

Parameter	Symbol	Minimum	Typical	Maximum	Unit	Condition
SREFCK Clock duty cycle	DC_{SREFCK}	45		55	%	At 50% threshold
Jitter tolerance for XREFCLK, WREFCLK, and SREFCLK	JTL_{XREF}			0.7	ns	For frequency 2 KHz to 20 MHz
Jitter tolerance for CLK1588	JTL_{CLK_1588}			200	ps	
Frequency for CLK1588[2]	f_{CLK_1588}		125	250	MHz	
Duty cycle for CLK1588	$DC_{1588CLK}$	40	50	60	%	

1. XREFCK (LAN mode applications) frequency may be set to 125 MHz or 156.25 MHz. WREFCK (LAN or WAN mode Synchronous Ethernet applications) frequency may be set to 155.52 MHz. SREFCK (LAN mode Synchronous Ethernet applications) frequency is 156.25 MHz.
2. Contact your Microsemi representative for other frequencies.

The following illustration shows the worst-case clock jitter transfer characteristic for the XREFCK input.

Figure 92 • **XREFCK to Data Output Jitter Transfer**



5.2.4 Two-Wire Serial (Slave) Interface

This section contains information about the AC specifications for the two-wire serial slave interface for the VSC8489-17 device.

Table 68 • **Two-Wire Serial Interface AC Characteristics**

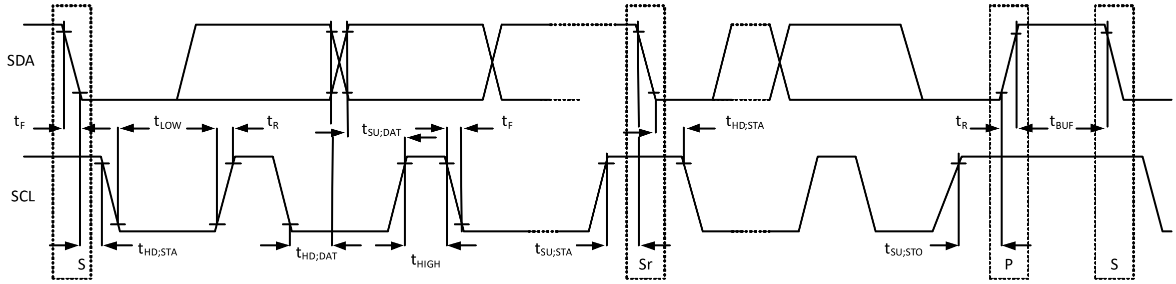
		Standard		Fast Mode		
Parameter	**Symbol**	**Minimum**	**Maximum**	**Minimum**	**Maximum**	**Unit**
Serial clock frequency	f_{SCL}		100		400	kHz
Hold time START condition after this period, the first clock pulse is generated	$t_{HD:STA}$	4.0		0.6		µs
Low period of SCL	t_{LOW}	4.7		1.3		µs
High period of SCL	t_{HIGH}	4.0		0.6		µs
Data hold time	$t_{HD:DAT}$	0	3.45	0	0.9	µs

Table 68 • **Two-Wire Serial Interface AC Characteristics *(continued)***

		Standard		Fast Mode		
Parameter	**Symbol**	**Minimum**	**Maximum**	**Minimum**	**Maximum**	**Unit**
Data setup time	$t_{SU:DAT}$	250		100		ns
Rise time for SDA and SCL	t_R		1000		300	ns
Fall time for SDA and SCL	t_F		300		300	ns
Setup time for STOP condition	$t_{SU:STO}$	4.0		0.6		µs
Bus free time between a STOP and START	t_{BUF}	4.7		1.3		µs
Capacitive load for SCL and SDA bus line	C_B		400		330	pF
External pull-up resistor[1]	R_P	900	$8 \times 10^{-7}/C_B$	900	$3 \times 10^{-7}/C_B$	Ω

1. Minimum value is determined from I_{OL} and internal reliability requirements. Maximum value is determined by load capacitance. Microsemi recommends 10 kΩ for typical applications in which capacitance loads are below the specified minimums.

Figure 93 • **Two-Wire Serial Interface Timing**



S = START, P = STOP, and Sr = repeated START.

5.2.5 MDIO Interface

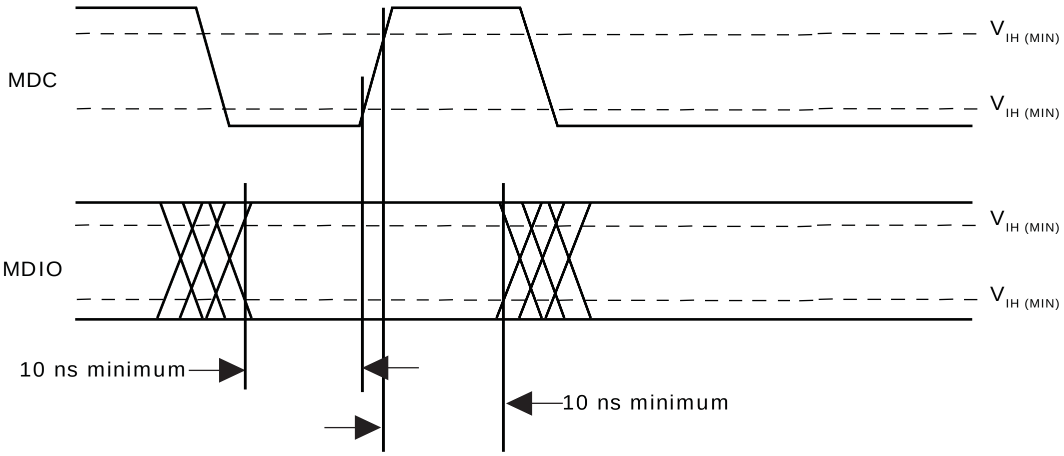
This section contains information about the AC specifications for the MDIO interface for the VSC8489-17 device.

Table 69 • **MDIO Interface AC Characteristics**

Parameter	**Symbol**	**Minimum**	**Maximum**	**Unit**
MDIO data hold time	t_{HOLD}	10		ns
MDIO data setup time	t_{SU}	10		ns
Delay from MDC rising edge to MDIO data change	t_{DELAY}		300	ns
MDC clock rate	f		2.5	MHz

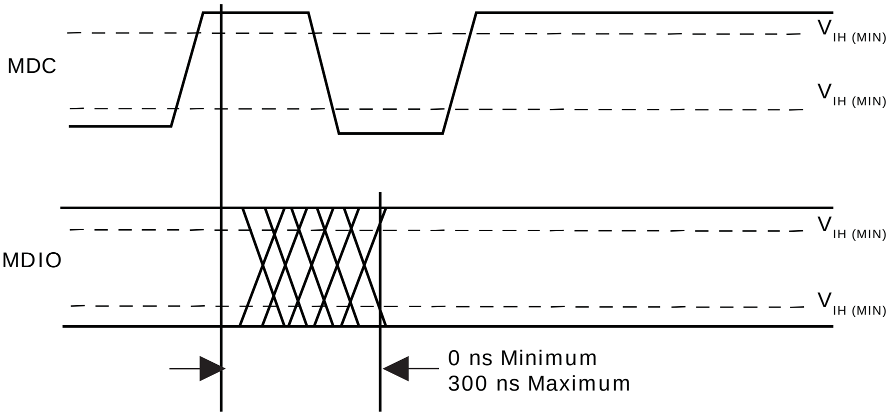
The following illustration shows the timing with the MDIO sourced by STA.

Figure 94 • **Timing with MDIO Sourced by STA**



The following illustration shows the timing with the MDIO sourced by MMD.

Figure 95 • **Timing with MDIO Sourced by MMD**



The following table lists the clock output specifications (RX0CKOUT, RX1CKOUT, TX0CKOUT, TX1CKOUT) for the VSC8489-17 device.

Table 70 • **Clock Output AC Characteristics**

Parameter	Symbol	Minimum	Maximum	Unit	Condition
RX0CKOUT, RX1CKOUT, TX0CKOUT, and TX1CKOUT jitter generation	JG_{C64}		10	ps_{RMS}	from 10 KHz to 10 MHz
RX0CKOUT, RX1CKOUT, TX0CKOUT, and TX1CKOUT differential output swing	ΔV	650	900	mV_{P-P}	

5.2.6 Synchronous Time-of-Day Load/Save Timing

When the 1588 Load/Save strobe (GPIO_1 pin) is applied to the device synchronous to CLK1588P/N, the setup and hold (minimum) times shown in the following table must be satisfied.

Table 71 • **Load/Save Setup and Hold Timing AC Characteristics**

Parameter	Symbol	Minimum	Unit
1588 LOAD/SAVE setup time	t_{SETUP}	1.1	ns
1588 LOAD/SAVE hold time	t_{HOLD}	0.1	ns

The following illustration shows the LOAD/SAVE AC timing.

Figure 96 • **Load/Save AC Timing**

5.2.7 SPI Slave Interface

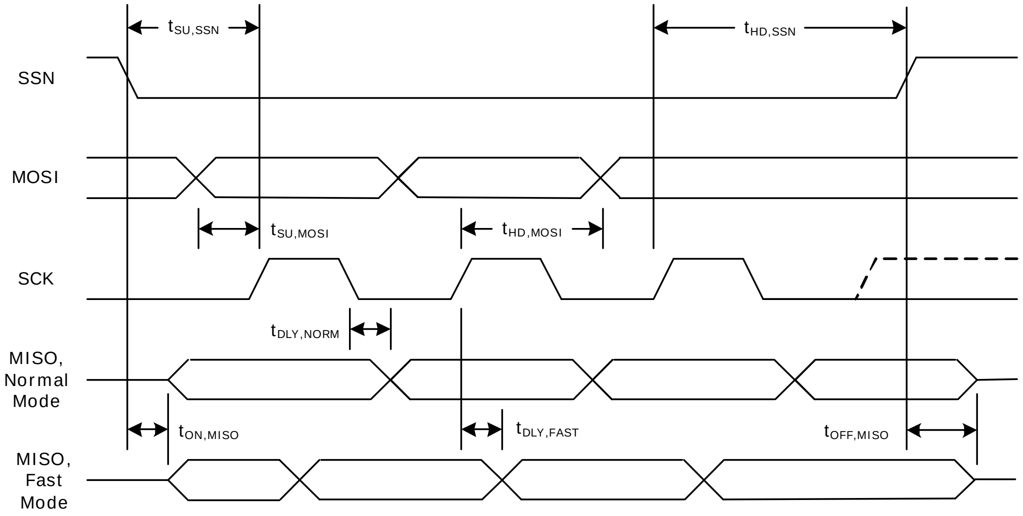
This section contains information about the AC specifications for the four-pin SPI slave interface used to read and write registers. The maximum clock rate is 30 MHz and it is configurable.

Table 72 • **SPI Slave Interface AC Characteristics**

Parameter	Symbol	Minimum	Maximum	Unit	Condition
MOSI data setup time	$t_{SU, MOSI}$	10		ns	
MOSI data hold time	$t_{HD, MOSI}$	10		ns	
SSN data setup time	$t_{SU, SSN}$	15		ns	SSN transition low to enable interface
SSN data hold time	$t_{HD, SSN}$	SCK clock period + 15.0		ns	SSN transition high to enable interface
SSN transition low to MISO valid	$t_{ON, MISO}$		17	ns	
SSN transition high to MISO high impedance	$t_{OFF, MISO}$	18		ns	
Falling SCK to valid MISO data, normal mode	$t_{DLY, NORM}$	14	30	ns	Maximum capacitance loading of 5 pF
Rising SCK to valid MISO data, fast mode	$t_{DLY, FAST}$	14	30	ns	Maximum capacitance loading of 5 pF

The following illustration shows the SPI interface timing.

Figure 97 • **SPI Interface Timing**



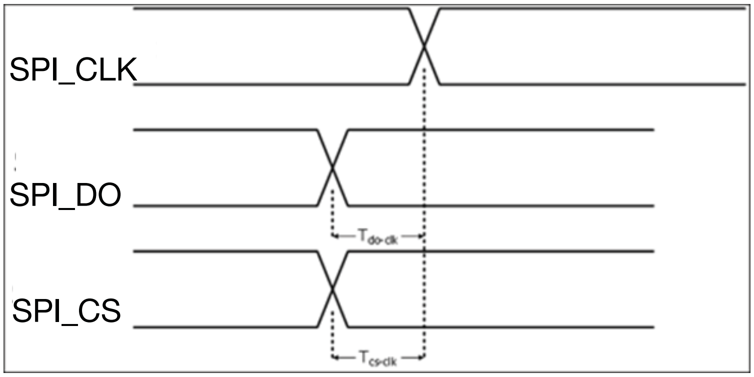
The following table lists the AC characteristics for the 3-pin push-out SPI.

Table 73 • **3-Pin Push-Out SPI AC Characteristics**

Parameter	Symbol	Minimum	Maximum	Unit
SPI_DO to SPI_CLK delay	$t_{DO, CLK}$	–1	6.5	ns
SPI_CS to SPI_CLK delay	$t_{CS, CLK}$	0.5	8	ns

The following illustration shows the 3-pin push-out SPI timing.

Figure 98 • **3-Pin Push-Out SPI Timing**



5.3 Operating Conditions

To ensure that the control pins remain set to the desired configured state when the VSC8489-17 device is powered up, perform a reset using the reset pin after power-up and after the control pins are steady for 1 ms.

Table 74 • **Recommended Operating Conditions**

Parameter	Symbol	Minimum	Typical	Maximum	Unit	Condition
1.0 V power supply voltage	V_{DDAH} V_{DDAL} V_{DDL}	0.95	1.0	1.05	V	
VSC8489-17 1.0 V power supply current	I_{DD}		2.0	2.7	A	XAUI to 10G in LAN mode.
1.2 V power supply voltage	V_{DDHSL}	1.14	1.2	1.26	V	
1.2 V power supply current	I_{DD12}		98	150	mA	
2.5 V TTL I/O power supply voltage	V_{DDTTL} V_{DDMDIO}	2.375	2.5	2.625	V	
TTL I/O power supply current	I_{DDTTL}		40		mA	
VSC8489-17 power consumption, 10G LAN	P_{DD_LAN}		2.3	3.1	W	XAUI to XFI in 10G LAN.
VSC8489-17 power consumption, 10G WAN	P_{DD_WAN}		2.3	3.15	W	XAUI to XFI in 10G WAN.
Operating temperature[1]	T	–40		110	°C	

1. Minimum specification is ambient temperature, and the maximum is junction temperature.

5.4 Stress Ratings

This section contains the stress ratings for the VSC8489-17 device.

Warning Stresses listed in the following table may be applied to devices one at a time without causing permanent damage. Functionality at or exceeding the values listed is not implied. Exposure to these values for extended periods may affect device reliability

Table 75 • **Stress Ratings**

Parameter	Symbol	Minimum	Maximum	Unit
1.0 V power supply voltage, potential to ground	V_{DDAH} V_{DDAL} V_{DDL}	–0.3	1.1	V
1.2 V power supply voltage, potential to ground	V_{DDHSL}	–0.3	1.32	V
2.5 V TTL I/O power supply voltage	V_{DDTTL} V_{DDMDIO}	–0.3	2.75	V
Storage temperature	T_S	–55	125	°C
Electrostatic discharge voltage, charged device model	V_{ESD_CDM}	–250	250	V
Electrostatic discharge voltage, human body model	V_{ESD_HBM}	See note[1]		V

1. This device has completed all required testing as specified in the JEDEC standard JESD22-A114, Electrostatic Discharge (ESD) Sensitivity Testing Human Body Model (HBM), and complies with a Class 2 rating. The definition of Class 2 is any part that passes an ESD pulse of 2000 V, but fails an ESD pulse of 4000 V.

Warning This device can be damaged by electrostatic discharge (ESD) voltage. Microsemi recommends that all integrated circuits be handled with appropriate precautions. Failure to observe proper handling and installation procedures may adversely affect reliability of the device.

6 Pin Descriptions

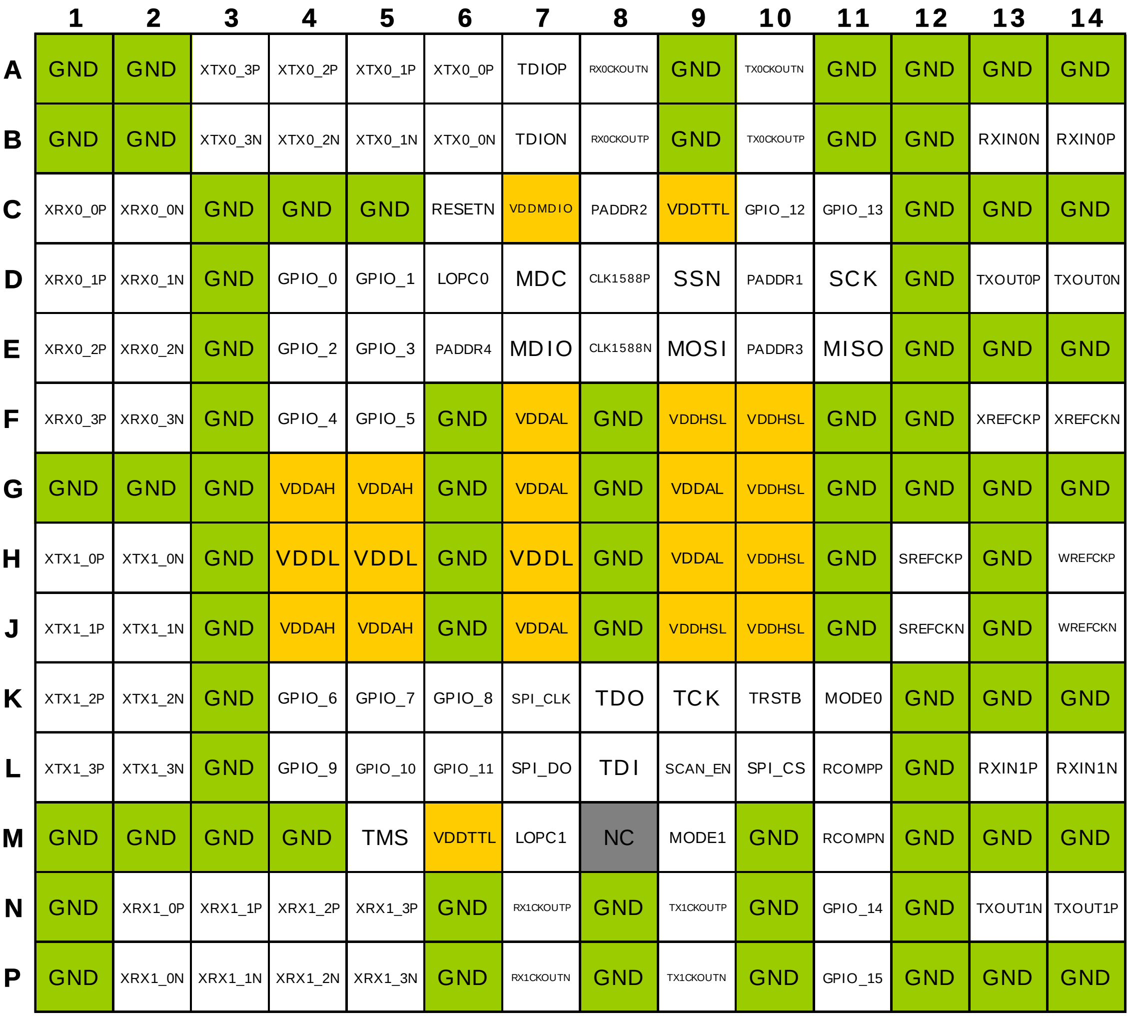
The VSC8489-17 device has 196 pins, which are described in this section.

The pin information is also provided as an attached Microsoft Excel file, so that you can copy it electronically. In Adobe Reader, double-click the attachment icon.

6.1 Pin Diagram

The following illustration is a representation of the VSC8489-17 device, as seen from the top view.

Figure 99 • **Pin Diagram**



	1	2	3	4	5	6	7	8	9	10	11	12	13	14
A	GND	GND	XTX0_3P	XTX0_2P	XTX0_1P	XTX0_0P	TDIOP	RX0CKOUTN	GND	TX0CKOUTN	GND	GND	GND	GND
B	GND	GND	XTX0_3N	XTX0_2N	XTX0_1N	XTX0_0N	TDION	RX0CKOUTP	GND	TX0CKOUTP	GND	GND	RXIN0N	RXIN0P
C	XRX0_0P	XRX0_0N	GND	GND	GND	RESETN	VDDMDIO	PADDR2	VDDTTL	GPIO_12	GPIO_13	GND	GND	GND
D	XRX0_1P	XRX0_1N	GND	GPIO_0	GPIO_1	LOPC0	MDC	CLK1588P	SSN	PADDR1	SCK	GND	TXOUT0P	TXOUT0N
E	XRX0_2P	XRX0_2N	GND	GPIO_2	GPIO_3	PADDR4	MDIO	CLK1588N	MOSI	PADDR3	MISO	GND	GND	GND
F	XRX0_3P	XRX0_3N	GND	GPIO_4	GPIO_5	GND	VDDAL	GND	VDDHSL	VDDHSL	GND	GND	XREFCKP	XREFCKN
G	GND	GND	GND	VDDAH	VDDAH	GND	VDDAL	GND	VDDAL	VDDHSL	GND	GND	GND	GND
H	XTX1_0P	XTX1_0N	GND	VDDL	VDDL	GND	VDDL	GND	VDDAL	VDDHSL	GND	SREFCKP	GND	WREFCKP
J	XTX1_1P	XTX1_1N	GND	VDDAH	VDDAH	GND	VDDAL	GND	VDDHSL	VDDHSL	GND	SREFCKN	GND	WREFCKN
K	XTX1_2P	XTX1_2N	GND	GPIO_6	GPIO_7	GPIO_8	SPI_CLK	TDO	TCK	TRSTB	MODE0	GND	GND	GND
L	XTX1_3P	XTX1_3N	GND	GPIO_9	GPIO_10	GPIO_11	SPI_DO	TDI	SCAN_EN	SPI_CS	RCOMPP	GND	RXIN1P	RXIN1N
M	GND	GND	GND	GND	TMS	VDDTTL	LOPC1	NC	MODE1	GND	RCOMPN	GND	GND	GND
N	GND	XRX1_0P	XRX1_1P	XRX1_2P	XRX1_3P	GND	RX1CKOUTP	GND	TX1CKOUTP	GND	GPIO_14	GND	TXOUT1N	TXOUT1P
P	GND	XRX1_0N	XRX1_1N	XRX1_2N	XRX1_3N	GND	RX1CKOUTN	GND	TX1CKOUTN	GND	GPIO_15	GND	GND	GND

6.2 Pin Identifications

This section contains the pin descriptions for the device, sorted according to their functional group.

The following table lists the definitions for the pin type symbols.

Table 76 • Pin Identifications

Symbol	Pin Type	Description
A	Analog I/O	Analog input for sensing variable voltage levels.
I	Input	Input signal.
O	Output	Output signal.
B	Bidirectional	Bidirectional input or output signal.
CML	Current mode logic	
NC	No connect	
LVTTL	Low voltage transistor-to-transistor logic	
LVTTLOD	Low-voltage transistor-to-transistor logic with open-drain output	

6.3 Pins by Function

This section contains the functional pin descriptions for the VSC8489-17 device.

Note: All the differential clock signals and differential data signals should be AC-coupled. A cap of 0.1 uF would be sufficient.

Functional Group	Name	Number	Type	Level	Description
1588	CLK1588N	E8	I	CML	1588 logic clock input, complement
1588	CLK1588P	D8	I	CML	1588 logic clock input, true
1588	SPI_CLK	K7	O	LVTTL	Pushout SPI clock output for 1588 timestamp
1588	SPI_CS	L10	O	LVTTL	Pushout SPI chip select output for 1588 timestamp
1588	SPI_DO	L7	O	LVTTL	Pushout SPI data output for 1588 timestamp
Clock Signal	RX0CKOUTN	A8	O	CML	Selectable clock output channel 0, complement. See register device 1, address A008.
Clock Signal	RX0CKOUTP	B8	O	CML	Selectable clock output channel 0, true. See register device 1, address A008.
Clock Signal	RX1CKOUTN	P7	O	CML	Selectable clock output channel 1, complement. See register device 1, address A008.
Clock Signal	RX1CKOUTP	N7	O	CML	Selectable clock output channel 1, true. See register device 1, address A008.
Clock Signal	SREFCKN	J12	I	CML	SyncE reference clock input, complement
Clock Signal	SREFCKP	H12	I	CML	SyncE reference clock input, true
Clock Signal	TX0CKOUTN	A10	O	CML	Selectable clock output channel 0, complement. See register device 1, address A009.

Clock Signal	TX0CKOUTP	B10	O	CML	Selectable clock output channel 0, true. See register device 1, address A009.
Clock Signal	TX1CKOUTN	P9	O	CML	Selectable clock output channel 1, complement. See register device 1, address A009.
Clock Signal	TX1CKOUTP	N9	O	CML	Selectable clock output channel 1, true. See register device 1, address A009.
Clock Signal	WREFCKN	J14	I	CML	WAN reference clock input, complement
Clock Signal	WREFCKP	H14	I	CML	WAN reference clock input, true
Clock Signal	XREFCKN	F14	I	CML	Reference clock input, complement
Clock Signal	XREFCKP	F13	I	CML	Reference clock input, true
JTAG	TCK	K9	I	LVTTL	Boundary scan, test clock input. Internally pulled high.
JTAG	TDI	L8	I	LVTTL	Boundary scan, test data input. Internally pulled high.
JTAG	TDO	K8	O	LVTTL	Boundary scan, test data output.
JTAG	TMS	M5	I	LVTTL	Boundary scan, test mode select. Internally pulled high.
JTAG	TRSTB	K10	I	LVTTL	Boundary scan, test reset input. Internally pulled high.
MDIO	MDC	D7	I	LVTTL	MDIO clock input
MDIO	MDIO	E7	B	LVTTLOD	MDIO data I/O
Miscellaneous	GPIO_0	D4	B	LVTTLOD	General purpose I/O 0
Miscellaneous	GPIO_1	D5	B	LVTTLOD	General purpose I/O 1
Miscellaneous	GPIO_2	E4	B	LVTTLOD	General purpose I/O 2
Miscellaneous	GPIO_3	E5	B	LVTTLOD	General purpose I/O 3
Miscellaneous	GPIO_4	F4	B	LVTTLOD	General purpose I/O 4
Miscellaneous	GPIO_5	F5	B	LVTTLOD	General purpose I/O 5
Miscellaneous	GPIO_6	K4	B	LVTTLOD	General purpose I/O 6
Miscellaneous	GPIO_7	K5	B	LVTTLOD	General purpose I/O 7
Miscellaneous	GPIO_8	K6	B	LVTTLOD	General purpose I/O 8
Miscellaneous	GPIO_9	L4	B	LVTTLOD	General purpose I/O 9
Miscellaneous	GPIO_10	L5	B	LVTTLOD	General purpose I/O 10
Miscellaneous	GPIO_11	L6	B	LVTTLOD	General purpose I/O 11
Miscellaneous	GPIO_12	C10	B	LVTTLOD	General purpose I/O 12
Miscellaneous	GPIO_13	C11	B	LVTTLOD	General purpose I/O 13
Miscellaneous	GPIO_14	N11	B	LVTTLOD	General purpose I/O 14
Miscellaneous	GPIO_15	P11	B	LVTTLOD	General purpose I/O 15
Miscellaneous	MODE0	K11	I	LVTTL	Mode select input bit 0
Miscellaneous	MODE1	M9	I	LVTTL	Mode select input bit 1
Miscellaneous	PADDR1	D10	I	LVTTL	MDIO port address bit 1. Internally pulled low.
Miscellaneous	PADDR2	C8	I	LVTTL	MDIO port address bit 2. Internally pulled low.

Miscellaneous	PADDR3	E10	I	LVTTL	MDIO port address bit 3. Internally pulled low.
Miscellaneous	PADDR4	E6	I	LVTTL	MDIO port address bit 4. Internally pulled low.
Miscellaneous	RCOMPN	M11		Analog	Resistor comparator, complement
Miscellaneous	RCOMPP	L11		Analog	Resistor comparator, truth
Miscellaneous	RESETN	C6	I	LVTTL	Reset. Low= reset. Internally pulled high.
Miscellaneous	SCAN_EN	L9	I	LVTTL	Scan enable input, factory test purposes only. Keep connected to Ground.
Miscellaneous	TDION	B7		Analog	Temperature diode, complement
Miscellaneous	TDIOP	A7		Analog	Temperature diode, truth
Power and Ground	GND	A1	P	GND	Ground
Power and Ground	GND	A2	P	GND	Ground
Power and Ground	GND	A9	P	GND	Ground
Power and Ground	GND	A11	P	GND	Ground
Power and Ground	GND	A12	P	GND	Ground
Power and Ground	GND	A13	P	GND	Ground
Power and Ground	GND	A14	P	GND	Ground
Power and Ground	GND	B1	P	GND	Ground
Power and Ground	GND	B2	P	GND	Ground
Power and Ground	GND	B9	P	GND	Ground
Power and Ground	GND	B11	P	GND	Ground
Power and Ground	GND	B12	P	GND	Ground
Power and Ground	GND	C3	P	GND	Ground
Power and Ground	GND	C4	P	GND	Ground
Power and Ground	GND	C5	P	GND	Ground
Power and Ground	GND	C12	P	GND	Ground
Power and Ground	GND	C13	P	GND	Ground
Power and Ground	GND	C14	P	GND	Ground
Power and Ground	GND	D3	P	GND	Ground
Power and Ground	GND	D12	P	GND	Ground
Power and Ground	GND	E3	P	GND	Ground
Power and Ground	GND	E12	P	GND	Ground
Power and Ground	GND	E13	P	GND	Ground
Power and Ground	GND	E14	P	GND	Ground
Power and Ground	GND	F3	P	GND	Ground
Power and Ground	GND	F6	P	GND	Ground
Power and Ground	GND	F8	P	GND	Ground
Power and Ground	GND	F11	P	GND	Ground
Power and Ground	GND	F12	P	GND	Ground
Power and Ground	GND	G1	P	GND	Ground
Power and Ground	GND	G2	P	GND	Ground

Power and Ground	GND	G3	P	GND	Ground
Power and Ground	GND	G6	P	GND	Ground
Power and Ground	GND	G8	P	GND	Ground
Power and Ground	GND	G11	P	GND	Ground
Power and Ground	GND	G12	P	GND	Ground
Power and Ground	GND	G13	P	GND	Ground
Power and Ground	GND	G14	P	GND	Ground
Power and Ground	GND	H3	P	GND	Ground
Power and Ground	GND	H6	P	GND	Ground
Power and Ground	GND	H8	P	GND	Ground
Power and Ground	GND	H11	P	GND	Ground
Power and Ground	GND	H13	P	GND	Ground
Power and Ground	GND	J3	P	GND	Ground
Power and Ground	GND	J6	P	GND	Ground
Power and Ground	GND	J8	P	GND	Ground
Power and Ground	GND	J11	P	GND	Ground
Power and Ground	GND	J13	P	GND	Ground
Power and Ground	GND	K3	P	GND	Ground
Power and Ground	GND	K12	P	GND	Ground
Power and Ground	GND	K13	P	GND	Ground
Power and Ground	GND	K14	P	GND	Ground
Power and Ground	GND	L3	P	GND	Ground
Power and Ground	GND	L12	P	GND	Ground
Power and Ground	GND	M1	P	GND	Ground
Power and Ground	GND	M2	P	GND	Ground
Power and Ground	GND	M3	P	GND	Ground
Power and Ground	GND	M4	P	GND	Ground
Power and Ground	GND	M10	P	GND	Ground
Power and Ground	GND	M12	P	GND	Ground
Power and Ground	GND	M13	P	GND	Ground
Power and Ground	GND	M14	P	GND	Ground
Power and Ground	GND	N1	P	GND	Ground
Power and Ground	GND	N6	P	GND	Ground
Power and Ground	GND	N8	P	GND	Ground
Power and Ground	GND	N10	P	GND	Ground
Power and Ground	GND	N12	P	GND	Ground
Power and Ground	GND	P1	P	GND	Ground
Power and Ground	GND	P6	P	GND	Ground
Power and Ground	GND	P8	P	GND	Ground
Power and Ground	GND	P10	P	GND	Ground
Power and Ground	GND	P12	P	GND	Ground
Power and Ground	GND	P13	P	GND	Ground

Power and Ground	GND	P14	P	GND	Ground
Power and Ground	VDDAH	G4	P	Supply	1.0 V power supply for host side analog
Power and Ground	VDDAH	G5	P	Supply	1.0 V power supply for host side analog
Power and Ground	VDDAH	J4	P	Supply	1.0 V power supply for host side analog
Power and Ground	VDDAH	J5	P	Supply	1.0 V power supply for host side analog
Power and Ground	VDDAL	F7	P	Supply	1.0 V power supply for line side analog
Power and Ground	VDDAL	G7	P	Supply	1.0 V power supply for line side analog
Power and Ground	VDDAL	G9	P	Supply	1.0 V power supply for line side analog
Power and Ground	VDDAL	H9	P	Supply	1.0 V power supply for line side analog
Power and Ground	VDDAL	J7	P	Supply	1.0 V power supply for line side analog
Power and Ground	VDDHSL	F9	P	Supply	1.2 V power supply for line side IOs
Power and Ground	VDDHSL	F10	P	Supply	1.2 V power supply for line side IOs
Power and Ground	VDDHSL	G10	P	Supply	1.2 V power supply for line side IOs
Power and Ground	VDDHSL	H10	P	Supply	1.2 V power supply for line side IOs
Power and Ground	VDDHSL	J9	P	Supply	1.2 V power supply for line side IOs
Power and Ground	VDDHSL	J10	P	Supply	1.2 V power supply for line side IOs
Power and Ground	VDDL	H4	P	Supply	1.0 V power supply for chip core
Power and Ground	VDDL	H5	P	Supply	1.0 V power supply for chip core
Power and Ground	VDDL	H7	P	Supply	1.0 V power supply for chip core
Power and Ground	VDDMDIO	C7	P	Supply	MDIO power supply
Power and Ground	VDDTTL	C9	P	Supply	LVTTL power supply
Power and Ground	VDDTTL	M6	P	Supply	LVTTL power supply
Receive and Transmit Path	LOPC0	D6	I	LVTTL	Loss of optical carrier, channel 0. Internally pulled high.
Receive and Transmit Path	LOPC1	M7	I	LVTTL	Loss of optical carrier, channel 1. Internally pulled high.
Receive and Transmit Path	RXIN0N	B13	I	CML	Receive channel 0 input data, complement
Receive and Transmit Path	RXIN0P	B14	I	CML	Receive channel 0 input data, true
Receive and Transmit Path	RXIN1N	L14	I	CML	Receive channel 1 input data, complement
Receive and Transmit Path	RXIN1P	L13	I	CML	Receive channel 1 input data, true
Receive and Transmit Path	TXOUT0N	D14	O	CML	Transmit channel 0 output data, complement
Receive and Transmit Path	TXOUT0P	D13	O	CML	Transmit channel 0 output data, true
Receive and Transmit Path	TXOUT1N	N13	O	CML	Transmit channel 1 output data, complement
Receive and Transmit Path	TXOUT1P	N14	O	CML	Transmit channel 1 output data, true
Reserved/No Connect	NC	M8			No connect (formerly labeled as ANATEST)

SPI	MISO	E11	O	LVTTL	SPI slave data output
SPI	MOSI	E9	I	LVTTL	SPI slave data input
SPI	SCK	D11	I	LVTTL	SPI slave clock input
SPI	SSN	D9	I	LVTTL	SPI slave chip select input
XAUI Channel	XRX0_0N	C2	I	CML	XAUI channel 0, Rx path lane 0, serial data input, complement
XAUI Channel	XRX0_0P	C1	I	CML	XAUI channel 0, Rx path lane 0, serial data input, true
XAUI Channel	XRX0_1N	D2	I	CML	XAUI channel 0, Rx path lane 1, serial data input, complement
XAUI Channel	XRX0_1P	D1	I	CML	XAUI channel 0, Rx path lane 1, serial data input, true
XAUI Channel	XRX0_2N	E2	I	CML	XAUI channel 0, Rx path lane 2, serial data input, complement
XAUI Channel	XRX0_2P	E1	I	CML	XAUI channel 0, Rx path lane 2, serial data input, true
XAUI Channel	XRX0_3N	F2	I	CML	XAUI channel 0, Rx path lane 3, serial data input, complement
XAUI Channel	XRX0_3P	F1	I	CML	XAUI channel 0, Rx path lane 3, serial data input, true
XAUI Channel	XRX1_0N	P2	I	CML	XAUI channel 1, Rx path lane 0, serial data input, complement
XAUI Channel	XRX1_0P	N2	I	CML	XAUI channel 1, Rx path lane 0, serial data input, true
XAUI Channel	XRX1_1N	P3	I	CML	XAUI channel 1, Rx path lane 1, serial data input, complement
XAUI Channel	XRX1_1P	N3	I	CML	XAUI channel 1, Rx path lane 1, serial data input, true
XAUI Channel	XRX1_2N	P4	I	CML	XAUI channel 1, Rx path lane 2, serial data input, complement
XAUI Channel	XRX1_2P	N4	I	CML	XAUI channel 1, Rx path lane 2, serial data input, true
XAUI Channel	XRX1_3N	P5	I	CML	XAUI channel 1, Rx path lane 3, serial data input, complement
XAUI Channel	XRX1_3P	N5	I	CML	XAUI channel 1, Rx path lane 3, serial data input, true
XAUI Channel	XTX0_0N	B6	O	CML	XAUI channel 0, Tx path lane 0, serial data output, complement
XAUI Channel	XTX0_0P	A6	O	CML	XAUI channel 0, Tx path lane 0, serial data output, true
XAUI Channel	XTX0_1N	B5	O	CML	XAUI channel 0, Tx path lane 1, serial data output, complement
XAUI Channel	XTX0_1P	A5	O	CML	XAUI channel 0, Tx path lane 1, serial data output, true
XAUI Channel	XTX0_2N	B4	O	CML	XAUI channel 0, Tx path lane 2, serial data output, complement

XAUI Channel	XTX0_2P	A4	O	CML	XAUI channel 0, Tx path lane 2, serial data output, true
XAUI Channel	XTX0_3N	B3	O	CML	XAUI channel 0, Tx path lane 3, serial data output, complement
XAUI Channel	XTX0_3P	A3	O	CML	XAUI channel 0, Tx path lane 3, serial data output, true
XAUI Channel	XTX1_0N	H2	O	CML	XAUI channel 1, Tx path lane 0, serial data output, complement
XAUI Channel	XTX1_0P	H1	O	CML	XAUI channel 1, Tx path lane 0, serial data output, true
XAUI Channel	XTX1_1N	J2	O	CML	XAUI channel 1, Tx path lane 1, serial data output, complement
XAUI Channel	XTX1_1P	J1	O	CML	XAUI channel 1, Tx path lane 1, serial data output, true
XAUI Channel	XTX1_2N	K2	O	CML	XAUI channel 1, Tx path lane 2, serial data output, complement
XAUI Channel	XTX1_2P	K1	O	CML	XAUI channel 1, Tx path lane 2, serial data output, true
XAUI Channel	XTX1_3N	L2	O	CML	XAUI channel 1, Tx path lane 3, serial data output, complement
XAUI Channel	XTX1_3P	L1	O	CML	XAUI channel 1, Tx path lane 3, serial data output, true

Functional Group	Name	Number	Type	Level	Description
1588	CLK1588N	E8	I	CML	1588 logic clock input, complement
1588	CLK1588P	D8	I	CML	1588 logic clock input, true
1588	SPI_CLK	K7	O	LVTTL	Pushout SPI clock output for 1588 timestamp
1588	SPI_CS	L10	O	LVTTL	Pushout SPI chip select output for 1588 timestamp
1588	SPI_DO	L7	O	LVTTL	Pushout SPI data output for 1588 timestamp
Clock Signal	RX0CKOUTN	A8	O	CML	Selectable clock output channel 0, complement. See register device 1, address A008.
Clock Signal	RX0CKOUTP	B8	O	CML	Selectable clock output channel 0, true. See register device 1, address A008.
Clock Signal	SREFCKN	J12	I	CML	SyncE reference clock input, complement
Clock Signal	SREFCKP	H12	I	CML	SyncE reference clock input, true
Clock Signal	TX0CKOUTN	A10	O	CML	Selectable clock output channel 0, complement. See register device 1, address A009.
Clock Signal	TX0CKOUTP	B10	O	CML	Selectable clock output channel 0, true. See register device 1, address A009.
Clock Signal	WREFCKN	J14	I	CML	WAN reference clock input, complement
Clock Signal	WREFCKP	H14	I	CML	WAN reference clock input, true
Clock Signal	XREFCKN	F14	I	CML	Reference clock input, complement
Clock Signal	XREFCKP	F13	I	CML	Reference clock input, true

JTAG	TCK	K9	I	LVTTL	Boundary scan, test clock input. Internally pulled high.
JTAG	TDI	L8	I	LVTTL	Boundary scan, test data input. Internally pulled high.
JTAG	TDO	K8	O	LVTTL	Boundary scan, test data output.
JTAG	TMS	M5	I	LVTTL	Boundary scan, test mode select. Internally pulled high.
JTAG	TRSTB	K10	I	LVTTL	Boundary scan, test reset input. Internally pulled high.
MDIO	MDC	D7	I	LVTTL	MDIO clock input
MDIO	MDIO	E7	B	LVTTLOD	MDIO data I/O
Miscellaneous	GPIO_0	D4	B	LVTTLOD	General purpose I/O 0
Miscellaneous	GPIO_1	D5	B	LVTTLOD	General purpose I/O 1
Miscellaneous	GPIO_2	E4	B	LVTTLOD	General purpose I/O 2
Miscellaneous	GPIO_3	E5	B	LVTTLOD	General purpose I/O 3
Miscellaneous	GPIO_4	F4	B	LVTTLOD	General purpose I/O 4
Miscellaneous	GPIO_5	F5	B	LVTTLOD	General purpose I/O 5
Miscellaneous	GPIO_6	K4	B	LVTTLOD	General purpose I/O 6
Miscellaneous	GPIO_7	K5	B	LVTTLOD	General purpose I/O 7
Miscellaneous	GPIO_8	K6	B	LVTTLOD	General purpose I/O 8
Miscellaneous	GPIO_9	L4	B	LVTTLOD	General purpose I/O 9
Miscellaneous	GPIO_10	L5	B	LVTTLOD	General purpose I/O 10
Miscellaneous	GPIO_11	L6	B	LVTTLOD	General purpose I/O 11
Miscellaneous	GPIO_12	C10	B	LVTTLOD	General purpose I/O 12
Miscellaneous	GPIO_13	C11	B	LVTTLOD	General purpose I/O 13
Miscellaneous	GPIO_14	N11	B	LVTTLOD	General purpose I/O 14
Miscellaneous	GPIO_15	P11	B	LVTTLOD	General purpose I/O 15
Miscellaneous	MODE0	K11	I	LVTTL	Mode select input bit 0
Miscellaneous	MODE1	M9	I	LVTTL	Mode select input bit 1
Miscellaneous	PADDR1	D10	I	LVTTL	MDIO port address bit 1. Internally pulled low.
Miscellaneous	PADDR2	C8	I	LVTTL	MDIO port address bit 2. Internally pulled low.
Miscellaneous	PADDR3	E10	I	LVTTL	MDIO port address bit 3. Internally pulled low.
Miscellaneous	PADDR4	E6	I	LVTTL	MDIO port address bit 4. Internally pulled low.
Miscellaneous	RCOMPN	M11		Analog	Resistor comparator, complement
Miscellaneous	RCOMPP	L11		Analog	Resistor comparator, truth
Miscellaneous	RESETN	C6	I	LVTTL	Reset. Low= reset. Internally pulled high.
Miscellaneous	SCAN_EN	L9	I	LVTTL	Scan enable input, factory test purposes only. Keep connected to Ground.
Miscellaneous	TDION	B7		Analog	Temperature diode, complement
Miscellaneous	TDIOP	A7		Analog	Temperature diode, truth
Power and Ground	GND	A1	P	GND	Ground
Power and Ground	GND	A2	P	GND	Ground
Power and Ground	GND	A9	P	GND	Ground

Power and Ground GND	A11	P	GND	Ground
Power and Ground GND	A12	P	GND	Ground
Power and Ground GND	A13	P	GND	Ground
Power and Ground GND	A14	P	GND	Ground
Power and Ground GND	B1	P	GND	Ground
Power and Ground GND	B2	P	GND	Ground
Power and Ground GND	B9	P	GND	Ground
Power and Ground GND	B11	P	GND	Ground
Power and Ground GND	B12	P	GND	Ground
Power and Ground GND	C4	P	GND	Ground
Power and Ground GND	C5	P	GND	Ground
Power and Ground GND	C12	P	GND	Ground
Power and Ground GND	C13	P	GND	Ground
Power and Ground GND	C14	P	GND	Ground
Power and Ground GND	D3	P	GND	Ground
Power and Ground GND	D12	P	GND	Ground
Power and Ground GND	E3	P	GND	Ground
Power and Ground GND	E12	P	GND	Ground
Power and Ground GND	E13	P	GND	Ground
Power and Ground GND	E14	P	GND	Ground
Power and Ground GND	F3	P	GND	Ground
Power and Ground GND	F6	P	GND	Ground
Power and Ground GND	F8	P	GND	Ground
Power and Ground GND	F11	P	GND	Ground
Power and Ground GND	F12	P	GND	Ground
Power and Ground GND	G1	P	GND	Ground
Power and Ground GND	G2	P	GND	Ground

Power and Ground GND	G3	P	GND	Ground
Power and Ground GND	G6	P	GND	Ground
Power and Ground GND	G8	P	GND	Ground
Power and Ground GND	G11	P	GND	Ground
Power and Ground GND	G12	P	GND	Ground
Power and Ground GND	G13	P	GND	Ground
Power and Ground GND	G14	P	GND	Ground
Power and Ground GND	H3	P	GND	Ground
Power and Ground GND	H6	P	GND	Ground
Power and Ground GND	H8	P	GND	Ground
Power and Ground GND	H11	P	GND	Ground
Power and Ground GND	H13	P	GND	Ground
Power and Ground GND	J3	P	GND	Ground
Power and Ground GND	J6	P	GND	Ground
Power and Ground GND	J8	P	GND	Ground
Power and Ground GND	J11	P	GND	Ground
Power and Ground GND	J13	P	GND	Ground
Power and Ground GND	K3	P	GND	Ground
Power and Ground GND	K12	P	GND	Ground
Power and Ground GND	K13	P	GND	Ground
Power and Ground GND	K14	P	GND	Ground
Power and Ground GND	L3	P	GND	Ground
Power and Ground GND	L12	P	GND	Ground
Power and Ground GND	M1	P	GND	Ground
Power and Ground GND	M2	P	GND	Ground
Power and Ground GND	M3	P	GND	Ground
Power and Ground GND	M4	P	GND	Ground

Power and Ground GND	M10	P	GND	Ground
Power and Ground GND	M12	P	GND	Ground
Power and Ground GND	M13	P	GND	Ground
Power and Ground GND	M14	P	GND	Ground
Power and Ground GND	N1	P	GND	Ground
Power and Ground GND	N6	P	GND	Ground
Power and Ground GND	N8	P	GND	Ground
Power and Ground GND	N10	P	GND	Ground
Power and Ground GND	N12	P	GND	Ground
Power and Ground GND	P1	P	GND	Ground
Power and Ground GND	P6	P	GND	Ground
Power and Ground GND	P8	P	GND	Ground
Power and Ground GND	P10	P	GND	Ground
Power and Ground GND	P12	P	GND	Ground
Power and Ground GND	P13	P	GND	Ground
Power and Ground GND	P14	P	GND	Ground
Power and Ground VDDAH	G4	P	Supply	1.0 V power supply for host side analog
Power and Ground VDDAH	G5	P	Supply	1.0 V power supply for host side analog
Power and Ground VDDAH	J4	P	Supply	1.0 V power supply for host side analog
Power and Ground VDDAH	J5	P	Supply	1.0 V power supply for host side analog
Power and Ground VDDAL	F7	P	Supply	1.0 V power supply for line side analog
Power and Ground VDDAL	G7	P	Supply	1.0 V power supply for line side analog
Power and Ground VDDAL	G9	P	Supply	1.0 V power supply for line side analog
Power and Ground VDDAL	H9	P	Supply	1.0 V power supply for line side analog
Power and Ground VDDAL	J7	P	Supply	1.0 V power supply for line side analog
Power and Ground VDDHSL	F9	P	Supply	1.2 V power supply for line side IOs
Power and Ground VDDHSL	F10	P	Supply	1.2 V power supply for line side IOs

Power and Ground	VDDHSL	G10	P	Supply	1.2 V power supply for line side IOs
Power and Ground	VDDHSL	H10	P	Supply	1.2 V power supply for line side IOs
Power and Ground	VDDHSL	J9	P	Supply	1.2 V power supply for line side IOs
Power and Ground	VDDHSL	J10	P	Supply	1.2 V power supply for line side IOs
Power and Ground	VDDL	H4	P	Supply	1.0 V power supply for chip core
Power and Ground	VDDL	H5	P	Supply	1.0 V power supply for chip core
Power and Ground	VDDL	H7	P	Supply	1.0 V power supply for chip core
Power and Ground	VDDMDIO	C7	P	Supply	MDIO power supply
Power and Ground	VDDTTL	C3	P	Supply	LVTTL power supply
Power and Ground	VDDTTL	C9	P	Supply	LVTTL power supply
Power and Ground	VDDTTL	M6	P	Supply	LVTTL power supply
Receive and Transmit Path	LOPC0	D6	I	LVTTL	Loss of optical carrier, channel 0. Internally pulled high.
Receive and Transmit Path	RXIN0N	B13	I	CML	Receive channel 0 input data, complement
Receive and Transmit Path	RXIN0P	B14	I	CML	Receive channel 0 input data, true
Receive and Transmit Path	TXOUT0N	D14	O	CML	Transmit channel 0 output data, complement
Receive and Transmit Path	TXOUT0P	D13	O	CML	Transmit channel 0 output data, true
Reserved/No Connect	NC	L13			No connect
Reserved/No Connect	NC	L14			No connect
Reserved/No Connect	NC	M7			No connect (could also be grounded)
Reserved/No Connect	NC	M8			No connect (formerly labeled as ANATEST)
Reserved/No Connect	NC	N7			No connect
Reserved/No Connect	NC	N9			No connect
Reserved/No Connect	NC	N13			No connect
Reserved/No Connect	NC	N14			No connect
Reserved/No Connect	NC	P7			No connect
Reserved/No Connect	NC	P9			No connect
SPI	MISO	E11	O	LVTTL	SPI slave data output

SPI	MOSI	E9	I	LVTTL	SPI slave data input
SPI	SCK	D11	I	LVTTL	SPI slave clock input
SPI	SSN	D9	I	LVTTL	SPI slave chip select input
XAUI Channel	XRX0_0N	C2	I	CML	XAUI channel 0, Rx path lane 0, serial data input, complement
XAUI Channel	XRX0_0P	C1	I	CML	XAUI channel 0, Rx path lane 0, serial data input, true
XAUI Channel	XRX0_1N	D2	I	CML	XAUI channel 0, Rx path lane 1, serial data input, complement
XAUI Channel	XRX0_1P	D1	I	CML	XAUI channel 0, Rx path lane 1, serial data input, true
XAUI Channel	XRX0_2N	E2	I	CML	XAUI channel 0, Rx path lane 2, serial data input, complement
XAUI Channel	XRX0_2P	E1	I	CML	XAUI channel 0, Rx path lane 2, serial data input, true
XAUI Channel	XRX0_3N	F2	I	CML	XAUI channel 0, Rx path lane 3, serial data input, complement
XAUI Channel	XRX0_3P	F1	I	CML	XAUI channel 0, Rx path lane 3, serial data input, true
XAUI Channel	XRX1_0N	P2	I	CML	XAUI channel 1, Rx path lane 0, serial data input, complement
XAUI Channel	XRX1_0P	N2	I	CML	XAUI channel 1, Rx path lane 0, serial data input, true
XAUI Channel	XRX1_1N	P3	I	CML	XAUI channel 1, Rx path lane 1, serial data input, complement
XAUI Channel	XRX1_1P	N3	I	CML	XAUI channel 1, Rx path lane 1, serial data input, true
XAUI Channel	XRX1_2N	P4	I	CML	XAUI channel 1, Rx path lane 2, serial data input, complement
XAUI Channel	XRX1_2P	N4	I	CML	XAUI channel 1, Rx path lane 2, serial data input, true
XAUI Channel	XRX1_3N	P5	I	CML	XAUI channel 1, Rx path lane 3, serial data input, complement
XAUI Channel	XRX1_3P	N5	I	CML	XAUI channel 1, Rx path lane 3, serial data input, true
XAUI Channel	XTX0_0N	B6	O	CML	XAUI channel 0, Tx path lane 0, serial data output, complement
XAUI Channel	XTX0_0P	A6	O	CML	XAUI channel 0, Tx path lane 0, serial data output, true
XAUI Channel	XTX0_1N	B5	O	CML	XAUI channel 0, Tx path lane 1, serial data output, complement
XAUI Channel	XTX0_1P	A5	O	CML	XAUI channel 0, Tx path lane 1, serial data output, true
XAUI Channel	XTX0_2N	B4	O	CML	XAUI channel 0, Tx path lane 2, serial data output, complement
XAUI Channel	XTX0_2P	A4	O	CML	XAUI channel 0, Tx path lane 2, serial data output, true
XAUI Channel	XTX0_3N	B3	O	CML	XAUI channel 0, Tx path lane 3, serial data output, complement
XAUI Channel	XTX0_3P	A3	O	CML	XAUI channel 0, Tx path lane 3, serial data output, true
XAUI Channel	XTX1_0N	H2	O	CML	XAUI channel 1, Tx path lane 0, serial data output, complement

XAUI Channel	XTX1_0P	H1	O	CML	XAUI channel 1, Tx path lane 0, serial data output, true
XAUI Channel	XTX1_1N	J2	O	CML	XAUI channel 1, Tx path lane 1, serial data output, complement
XAUI Channel	XTX1_1P	J1	O	CML	XAUI channel 1, Tx path lane 1, serial data output, true
XAUI Channel	XTX1_2N	K2	O	CML	XAUI channel 1, Tx path lane 2, serial data output, complement
XAUI Channel	XTX1_2P	K1	O	CML	XAUI channel 1, Tx path lane 2, serial data output, true
XAUI Channel	XTX1_3N	L2	O	CML	XAUI channel 1, Tx path lane 3, serial data output, complement
XAUI Channel	XTX1_3P	L1	O	CML	XAUI channel 1, Tx path lane 3, serial data output, true

Functional Group	Name	Number	Type	Level	Description
Clock Signal	RX0CKOUTN	A8	O	CML	Selectable clock output channel 0, complement. See register device 1, address A008.
Clock Signal	RX0CKOUTP	B8	O	CML	Selectable clock output channel 0, true. See register device 1, address A008.
Clock Signal	RX1CKOUTN	P7	O	CML	Selectable clock output channel 1, complement. See register device 1, address A008.
Clock Signal	RX1CKOUTP	N7	O	CML	Selectable clock output channel 1, true. See register device 1, address A008.
Clock Signal	SREFCKN	J12	I	CML	SyncE reference clock input, complement
Clock Signal	SREFCKP	H12	I	CML	SyncE reference clock input, true
Clock Signal	TX0CKOUTN	A10	O	CML	Selectable clock output channel 0, complement. See register device 1, address A009.
Clock Signal	TX0CKOUTP	B10	O	CML	Selectable clock output channel 0, true. See register device 1, address A009.
Clock Signal	TX1CKOUTN	P9	O	CML	Selectable clock output channel 1, complement. See register device 1, address A009.
Clock Signal	TX1CKOUTP	N9	O	CML	Selectable clock output channel 1, true. See register device 1, address A009.
Clock Signal	WREFCKN	J14	I	CML	WAN reference clock input, complement
Clock Signal	WREFCKP	H14	I	CML	WAN reference clock input, true
Clock Signal	XREFCKN	F14	I	CML	Reference clock input, complement
Clock Signal	XREFCKP	F13	I	CML	Reference clock input, true
JTAG	TCK	K9	I	LVTTL	Boundary scan, test clock input. Internally pulled high.
JTAG	TDI	L8	I	LVTTL	Boundary scan, test data input. Internally pulled high.
JTAG	TDO	K8	O	LVTTL	Boundary scan, test data output.
JTAG	TMS	M5	I	LVTTL	Boundary scan, test mode select. Internally pulled high.

JTAG	TRSTB	K10	I	LVTTL	Boundary scan, test reset input. Internally pulled high.
MDIO	MDC	D7	I	LVTTL	MDIO clock input
MDIO	MDIO	E7	B	LVTTLOD	MDIO data I/O
Miscellaneous	GPIO_0	D4	B	LVTTLOD	General purpose I/O 0
Miscellaneous	GPIO_1	D5	B	LVTTLOD	General purpose I/O 1
Miscellaneous	GPIO_2	E4	B	LVTTLOD	General purpose I/O 2
Miscellaneous	GPIO_3	E5	B	LVTTLOD	General purpose I/O 3
Miscellaneous	GPIO_4	F4	B	LVTTLOD	General purpose I/O 4
Miscellaneous	GPIO_5	F5	B	LVTTLOD	General purpose I/O 5
Miscellaneous	GPIO_6	K4	B	LVTTLOD	General purpose I/O 6
Miscellaneous	GPIO_7	K5	B	LVTTLOD	General purpose I/O 7
Miscellaneous	GPIO_8	K6	B	LVTTLOD	General purpose I/O 8
Miscellaneous	GPIO_9	L4	B	LVTTLOD	General purpose I/O 9
Miscellaneous	GPIO_10	L5	B	LVTTLOD	General purpose I/O 10
Miscellaneous	GPIO_11	L6	B	LVTTLOD	General purpose I/O 11
Miscellaneous	GPIO_12	C10	B	LVTTLOD	General purpose I/O 12
Miscellaneous	GPIO_13	C11	B	LVTTLOD	General purpose I/O 13
Miscellaneous	GPIO_14	N11	B	LVTTLOD	General purpose I/O 14
Miscellaneous	GPIO_15	P11	B	LVTTLOD	General purpose I/O 15
Miscellaneous	MODE0	K11	I	LVTTL	Mode select input bit 0
Miscellaneous	MODE1	M9	I	LVTTL	Mode select input bit 1
Miscellaneous	PADDR1	D10	I	LVTTL	MDIO port address bit 1. Internally pulled low.
Miscellaneous	PADDR2	C8	I	LVTTL	MDIO port address bit 2. Internally pulled low.
Miscellaneous	PADDR3	E10	I	LVTTL	MDIO port address bit 3. Internally pulled low.
Miscellaneous	PADDR4	E6	I	LVTTL	MDIO port address bit 4. Internally pulled low.
Miscellaneous	RCOMPN	M11		Analog	Resistor comparator, complement
Miscellaneous	RCOMPP	L11		Analog	Resistor comparator, truth
Miscellaneous	RESETN	C6	I	LVTTL	Reset. Low= reset. Internally pulled high.
Miscellaneous	SCAN_EN	L9	I	LVTTL	Scan enable input, factory test purposes only. Keep connected to Ground.
Miscellaneous	TDION	B7		Analog	Temperature diode, complement
Miscellaneous	TDIOP	A7		Analog	Temperature diode, truth
Power and Ground	GND	A1	P	GND	Ground
Power and Ground	GND	A2	P	GND	Ground
Power and Ground	GND	A9	P	GND	Ground
Power and Ground	GND	A11	P	GND	Ground
Power and Ground	GND	A12	P	GND	Ground

Power and Ground	GND	A13	P	GND	Ground
Power and Ground	GND	A14	P	GND	Ground
Power and Ground	GND	B1	P	GND	Ground
Power and Ground	GND	B2	P	GND	Ground
Power and Ground	GND	B9	P	GND	Ground
Power and Ground	GND	B11	P	GND	Ground
Power and Ground	GND	B12	P	GND	Ground
Power and Ground	GND	C3	P	GND	Ground
Power and Ground	GND	C4	P	GND	Ground
Power and Ground	GND	C5	P	GND	Ground
Power and Ground	GND	C12	P	GND	Ground
Power and Ground	GND	C13	P	GND	Ground
Power and Ground	GND	C14	P	GND	Ground
Power and Ground	GND	D3	P	GND	Ground
Power and Ground	GND	D12	P	GND	Ground
Power and Ground	GND	E3	P	GND	Ground
Power and Ground	GND	E12	P	GND	Ground
Power and Ground	GND	E13	P	GND	Ground
Power and Ground	GND	E14	P	GND	Ground
Power and Ground	GND	F3	P	GND	Ground
Power and Ground	GND	F6	P	GND	Ground
Power and Ground	GND	F8	P	GND	Ground
Power and Ground	GND	F11	P	GND	Ground
Power and Ground	GND	F12	P	GND	Ground

Power and Ground	GND	G1	P	GND	Ground
Power and Ground	GND	G2	P	GND	Ground
Power and Ground	GND	G3	P	GND	Ground
Power and Ground	GND	G6	P	GND	Ground
Power and Ground	GND	G8	P	GND	Ground
Power and Ground	GND	G11	P	GND	Ground
Power and Ground	GND	G12	P	GND	Ground
Power and Ground	GND	G13	P	GND	Ground
Power and Ground	GND	G14	P	GND	Ground
Power and Ground	GND	H3	P	GND	Ground
Power and Ground	GND	H6	P	GND	Ground
Power and Ground	GND	H8	P	GND	Ground
Power and Ground	GND	H11	P	GND	Ground
Power and Ground	GND	H13	P	GND	Ground
Power and Ground	GND	J3	P	GND	Ground
Power and Ground	GND	J6	P	GND	Ground
Power and Ground	GND	J8	P	GND	Ground
Power and Ground	GND	J11	P	GND	Ground
Power and Ground	GND	J13	P	GND	Ground
Power and Ground	GND	K3	P	GND	Ground
Power and Ground	GND	K12	P	GND	Ground
Power and Ground	GND	K13	P	GND	Ground
Power and Ground	GND	K14	P	GND	Ground
Power and Ground	GND	L3	P	GND	Ground

Power and Ground	GND	L12	P	GND	Ground
Power and Ground	GND	M1	P	GND	Ground
Power and Ground	GND	M2	P	GND	Ground
Power and Ground	GND	M3	P	GND	Ground
Power and Ground	GND	M4	P	GND	Ground
Power and Ground	GND	M10	P	GND	Ground
Power and Ground	GND	M12	P	GND	Ground
Power and Ground	GND	M13	P	GND	Ground
Power and Ground	GND	M14	P	GND	Ground
Power and Ground	GND	N1	P	GND	Ground
Power and Ground	GND	N6	P	GND	Ground
Power and Ground	GND	N8	P	GND	Ground
Power and Ground	GND	N10	P	GND	Ground
Power and Ground	GND	N12	P	GND	Ground
Power and Ground	GND	P1	P	GND	Ground
Power and Ground	GND	P6	P	GND	Ground
Power and Ground	GND	P8	P	GND	Ground
Power and Ground	GND	P10	P	GND	Ground
Power and Ground	GND	P12	P	GND	Ground
Power and Ground	GND	P13	P	GND	Ground
Power and Ground	GND	P14	P	GND	Ground
Power and Ground	VDDAH	G4	P	Supply	1.0 V power supply for host side analog
Power and Ground	VDDAH	G5	P	Supply	1.0 V power supply for host side analog
Power and Ground	VDDAH	J4	P	Supply	1.0 V power supply for host side analog

Power and Ground	VDDAH	J5	P	Supply	1.0 V power supply for host side analog
Power and Ground	VDDAL	F7	P	Supply	1.0 V power supply for line side analog
Power and Ground	VDDAL	G7	P	Supply	1.0 V power supply for line side analog
Power and Ground	VDDAL	G9	P	Supply	1.0 V power supply for line side analog
Power and Ground	VDDAL	H9	P	Supply	1.0 V power supply for line side analog
Power and Ground	VDDAL	J7	P	Supply	1.0 V power supply for line side analog
Power and Ground	VDDHSL	F9	P	Supply	1.2 V power supply for line side IOs
Power and Ground	VDDHSL	F10	P	Supply	1.2 V power supply for line side IOs
Power and Ground	VDDHSL	G10	P	Supply	1.2 V power supply for line side IOs
Power and Ground	VDDHSL	H10	P	Supply	1.2 V power supply for line side IOs
Power and Ground	VDDHSL	J9	P	Supply	1.2 V power supply for line side IOs
Power and Ground	VDDHSL	J10	P	Supply	1.2 V power supply for line side IOs
Power and Ground	VDDL	H4	P	Supply	1.0 V power supply for chip core
Power and Ground	VDDL	H5	P	Supply	1.0 V power supply for chip core
Power and Ground	VDDL	H7	P	Supply	1.0 V power supply for chip core
Power and Ground	VDDMDIO	C7	P	Supply	MDIO power supply
Power and Ground	VDDTTL	C9	P	Supply	LVTTL power supply
Power and Ground	VDDTTL	M6	P	Supply	LVTTL power supply
Receive and Transmit Path	LOPC0	D6	I	LVTTL	Loss of optical carrier, channel 0. Internally pulled high.
Receive and Transmit Path	LOPC1	M7	I	LVTTL	Loss of optical carrier, channel 1. Internally pulled high.
Receive and Transmit Path	RXIN0N	B13	I	CML	Receive channel 0 input data, complement
Receive and Transmit Path	RXIN0P	B14	I	CML	Receive channel 0 input data, true
Receive and Transmit Path	RXIN1N	L14	I	CML	Receive channel 1 input data, complement
Receive and Transmit Path	RXIN1P	L13	I	CML	Receive channel 1 input data, true

Receive and Transmit Path	TXOUT0N	D14	O	CML	Transmit channel 0 output data, complement
Receive and Transmit Path	TXOUT0P	D13	O	CML	Transmit channel 0 output data, true
Receive and Transmit Path	TXOUT1N	N13	O	CML	Transmit channel 1 output data, complement
Receive and Transmit Path	TXOUT1P	N14	O	CML	Transmit channel 1 output data, true
Reserved/No Connect	NC	D8			No connect
Reserved/No Connect	NC	E8			No connect
Reserved/No Connect	NC	K7			No connect
Reserved/No Connect	NC	L7			No connect
Reserved/No Connect	NC	L10			No connect
Reserved/No Connect	NC	M8			No connect (formerly labeled as ANATEST)
SPI	MISO	E11	O	LVTTL	SPI slave data output
SPI	MOSI	E9	I	LVTTL	SPI slave data input
SPI	SCK	D11	I	LVTTL	SPI slave clock input
SPI	SSN	D9	I	LVTTL	SPI slave chip select input
XAUI Channel	XRX0_0N	C2	I	CML	XAUI channel 0, Rx path lane 0, serial data input, complement
XAUI Channel	XRX0_0P	C1	I	CML	XAUI channel 0, Rx path lane 0, serial data input, true
XAUI Channel	XRX0_1N	D2	I	CML	XAUI channel 0, Rx path lane 1, serial data input, complement
XAUI Channel	XRX0_1P	D1	I	CML	XAUI channel 0, Rx path lane 1, serial data input, true
XAUI Channel	XRX0_2N	E2	I	CML	XAUI channel 0, Rx path lane 2, serial data input, complement
XAUI Channel	XRX0_2P	E1	I	CML	XAUI channel 0, Rx path lane 2, serial data input, true
XAUI Channel	XRX0_3N	F2	I	CML	XAUI channel 0, Rx path lane 3, serial data input, complement
XAUI Channel	XRX0_3P	F1	I	CML	XAUI channel 0, Rx path lane 3, serial data input, true
XAUI Channel	XRX1_0N	P2	I	CML	XAUI channel 1, Rx path lane 0, serial data input, complement
XAUI Channel	XRX1_0P	N2	I	CML	XAUI channel 1, Rx path lane 0, serial data input, true
XAUI Channel	XRX1_1N	P3	I	CML	XAUI channel 1, Rx path lane 1, serial data input, complement

XAUI Channel	XRX1_1P	N3	I	CML	XAUI channel 1, Rx path lane 1, serial data input, true
XAUI Channel	XRX1_2N	P4	I	CML	XAUI channel 1, Rx path lane 2, serial data input, complement
XAUI Channel	XRX1_2P	N4	I	CML	XAUI channel 1, Rx path lane 2, serial data input, true
XAUI Channel	XRX1_3N	P5	I	CML	XAUI channel 1, Rx path lane 3, serial data input, complement
XAUI Channel	XRX1_3P	N5	I	CML	XAUI channel 1, Rx path lane 3, serial data input, true
XAUI Channel	XTX0_0N	B6	O	CML	XAUI channel 0, Tx path lane 0, serial data output, complement
XAUI Channel	XTX0_0P	A6	O	CML	XAUI channel 0, Tx path lane 0, serial data output, true
XAUI Channel	XTX0_1N	B5	O	CML	XAUI channel 0, Tx path lane 1, serial data output, complement
XAUI Channel	XTX0_1P	A5	O	CML	XAUI channel 0, Tx path lane 1, serial data output, true
XAUI Channel	XTX0_2N	B4	O	CML	XAUI channel 0, Tx path lane 2, serial data output, complement
XAUI Channel	XTX0_2P	A4	O	CML	XAUI channel 0, Tx path lane 2, serial data output, true
XAUI Channel	XTX0_3N	B3	O	CML	XAUI channel 0, Tx path lane 3, serial data output, complement
XAUI Channel	XTX0_3P	A3	O	CML	XAUI channel 0, Tx path lane 3, serial data output, true
XAUI Channel	XTX1_0N	H2	O	CML	XAUI channel 1, Tx path lane 0, serial data output, complement
XAUI Channel	XTX1_0P	H1	O	CML	XAUI channel 1, Tx path lane 0, serial data output, true
XAUI Channel	XTX1_1N	J2	O	CML	XAUI channel 1, Tx path lane 1, serial data output, complement
XAUI Channel	XTX1_1P	J1	O	CML	XAUI channel 1, Tx path lane 1, serial data output, true
XAUI Channel	XTX1_2N	K2	O	CML	XAUI channel 1, Tx path lane 2, serial data output, complement
XAUI Channel	XTX1_2P	K1	O	CML	XAUI channel 1, Tx path lane 2, serial data output, true
XAUI Channel	XTX1_3N	L2	O	CML	XAUI channel 1, Tx path lane 3, serial data output, complement
XAUI Channel	XTX1_3P	L1	O	CML	XAUI channel 1, Tx path lane 3, serial data output, true

7 Package Information

The VSC8489YJU-17 package is a lead-free (Pb-free), 196-pin, flip chip ball grid array (FCBGA) with a 15 mm × 15 mm body size, 1 mm pin pitch, and 1.4 mm maximum height.

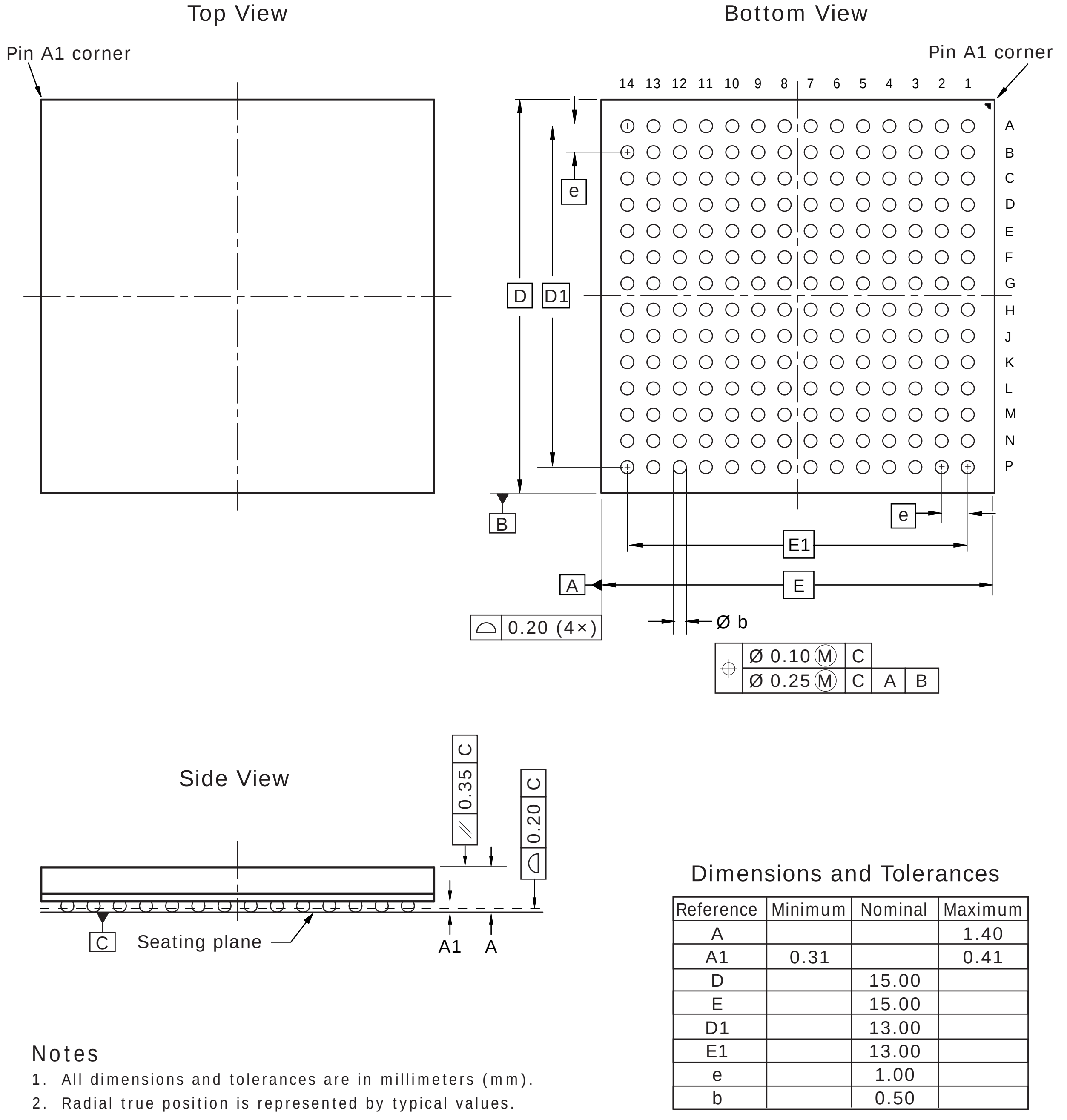
Lead-free products from Microsemi comply with the temperatures and profiles defined in the joint IPC and JEDEC standard IPC/JEDEC J-STD-020. For more information, see the IPC and JEDEC standard.

This section provides the package drawing, thermal specifications, and moisture sensitivity rating for the VSC8489-17 device.

7.1 Package Drawing

The following illustration shows the package drawing for the VSC8489-17 device. The drawing contains the top view, bottom view, side view, dimensions, tolerances, and notes.

Figure 100 • Package Drawing



Dimensions and Tolerances

Reference	Minimum	Nominal	Maximum
A			1.40
A1	0.31		0.41
D		15.00	
E		15.00	
D1		13.00	
E1		13.00	
e		1.00	
b		0.50	

Notes

1. All dimensions and tolerances are in millimeters (mm).
2. Radial true position is represented by typical values.

7.2 Thermal Specifications

Thermal specifications for this device are based on the JEDEC JESD51 family of documents. These documents are available on the JEDEC Web site at www.jedec.org. The thermal specifications are modeled using a four-layer test board with two signal layers, a power plane, and a ground plane (2s2p

PCB). For more information about the thermal measurement method used for this device, see the JESD51-1 standard.

Table 77 • **Thermal Resistances**

Symbol	°C/W	Parameter
θ_{JCtop}	3.35	Die junction to package case top
θ_{JB}	13.3	Die junction to printed circuit board
θ_{JA}	22.74	Die junction to ambient
θ_{JMA} at 1 m/s	18.6	Die junction to moving air measured at an air speed of 1 m/s
θ_{JMA} at 2 m/s	17.03	Die junction to moving air measured at an air speed of 2 m/s

To achieve results similar to the modeled thermal measurements, the guidelines for board design described in the JESD51 family of publications must be applied. For information about applications using FCBGA packages, see the following:

- JESD51-2A, *Integrated Circuits Thermal Test Method Environmental Conditions, Natural Convection (Still Air)*
- JESD51-6, *Integrated Circuit Thermal Test Method Environmental Conditions, Forced Convection (Moving Air)*
- JESD51-8, *Integrated Circuit Thermal Test Method Environmental Conditions, Junction-to-Board*
- JESD51-9, *Test Boards for Area Array Surface Mount Package Thermal Measurements*

7.3 Moisture Sensitivity

This device is rated moisture sensitivity level 4 as specified in the joint IPC and JEDEC standard IPC/JEDEC J-STD-020. For more information, see the IPC and JEDEC standard.

8 Design Considerations

This section provides information about design considerations for the VSC8489-17 device.

8.1 1588 bypass switch should not be activated on the fly

When the 1588 bypass switch is activated on the fly and traffic is flowing, one packet in the system is corrupted. Thereafter, the system recovers and the packets are processed normally. Disabling the bypass switch at any time has no impact on traffic.

Because the 1588 engine does not distinguish between PTP and any other traffic from the traffic flow standpoint, the system should prevent 1588 from being turned off when any traffic is flowing. However, if one packet corruption is acceptable to the system design, then the bypass switch can be enabled when traffic is flowing.

8.2 Low-power mode and SerDes calibration

SerDes re-initialization and re-calibration is required when the PHY comes out of the low power mode.

Use the API to enable the required low power and re-calibration functionality instead of the low power enabling bits at 1x0000.11, 2x0000.11, 3x0000.11, or 4x0000.11, which force a reset of the SerDes registers.

8.3 Low-power mode should not be enabled when failover switching is enabled

The device design was not intended to support the low power mode of operation when the failover switch is enabled. When low power mode is enabled in one channel, the data flow of the other channel could be adversely affected if the failover switch is enabled, leading to data errors.

Do not enable the low power mode when the failover switch is enabled.

8.4 Flow control with failover switching

Both Tx and Rx data paths of the channel have to be switched at the same time when flow control is enabled. The Tx data path of one channel in one direction and the Rx data path of another channel in the opposite direction cannot be mixed.

8.5 XAUI BIST Checker Compatibility

The XAUI BIST checker fails when checking the mixed frequency test pattern. This mixed frequency test pattern is optional in the IEEE802.3ae specifications.

8.6 SPI bus speeds

The maximum speed enabled on the 4-pin slave SPI bus is 15.4 MHz in normal mode and 30 MHz in fast mode. The maximum speed for the 3-pin push out only SPI is 40 MHz.

8.7 GPIO as TOSI

A small value pull-up is needed when a GPIO pin is used as TOSI. For more information, contact your Microsemi representative.

8.8 10GBASE-KR auto negotiation and training

10GBASE-KR negotiation and training (IEEE802.3 Clause 72 and Clause 73) is only available for 10G. It is not available for 1G.

8.9 Loopbacks in 10G WAN mode

Loopbacks L1, L2, and L2C are not available in 10G WAN mode if jumbo frames are used.

8.10 10/100M mode not supported

The PHY does not support modes of 10/100M in CuSFP. The autoneg feature is only supported in 1000BASE-X mode but not in SGMII mode. When interfacing with 1G SGMII mode (such as with CuSFP), the autoneg feature has to be turned off.

8.11 Limited access to registers during failover cross-connect mode

The following register bits should not be used if failover cross-connect is enabled (that is, if PMA0 is connected to channel_1 and PMA1 I connected to channel_0).

- 1x0001.2
- 1x0008.10
- 1x000A.0
- 1x9003.4
- 1x0008.11
- 1x9004.4

8.12 Limited auto negotiation support in 1G mode

In 1G mode, the device is specified to support basic auto negotiation for 1000BASE-X (optical interface) only. For an SGMII interface employed in interfacing CuSFP, auto negotiation is not supported. Otherwise, auto negotiation must be disabled on both the device and the CuSFP in order to have the data link be established.

8.13 Limited 1G status reporting

In 1G mode, the 1G status signal from the 1G PCS block is driven by a sticky bit rather than a latched bit, and so is useful only for link down (and not useful for link up conditions). Also, in 1000BASE-X mode, the link up indicator does not include AN done status.

8.14 Timestamp errors due to IEEE 1588 reference clock interruption

After 1588 clock interruption, a local time counter reload using the Unified API is required.

8.15 RXCKOUT squelching

RXCKOUT (positive and negative) can be squelched by varying link status (LOPC, PCS_Fault) in the device through the use of the API.

9 Ordering Information

The VSC8489YJU-17 package is a lead-free (Pb-free), 196-pin, flip chip ball grid array (FCBGA) with a 15 mm × 15 mm body size, 1 mm pin pitch, and 1.4 mm maximum height.

Lead-free products from Microsemi comply with the temperatures and profiles defined in the joint IPC and JEDEC standard IPC/JEDEC J-STD-020. For more information, see the IPC and JEDEC standard.

The following table lists the ordering information for the VSC8489-17 device.

Table 78 • **Ordering Information**

Part Order Number	Description
VSC8489YJU-17	Lead-free, 196-pin FCBGA with a 15 mm × 15 mm body size, 1 mm pin pitch, and 1.4 mm maximum height. The operating temperature is –40 °C ambient to 110 °C junction.