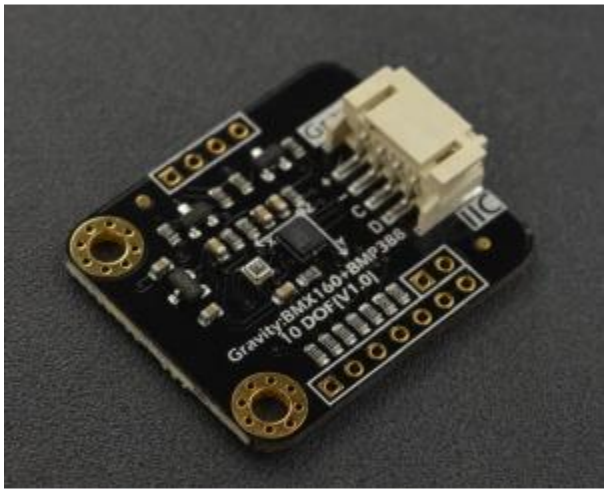
Gravity: BMX160+BMP388 10 DOF Sensor

SKU:SEN0252



Introduction

The BMX160 9-axis absolute orientation sensor from Bosch Sensortec is an ideal solution for applications that face strict constraints of board space, power consumption and appearance, especially for wearable device like smart watches or augmented reality glasses. This BMX160 sensor is the smallest 9-axis sensor in the industry. It comprises an accelerometer, gyroscope and geomagnetic sensor in a single package, and features less than 1.5mA power consumption. Combined with BSX sensor data fusion software library of Bosch Sensortec, the sensor performance can be further improved.

BMP388 is a high performance barometric pressure sensor with compact body, high resolution, and the smallest size in same series. The BMP388 delivers outstanding altitude stabilization in drones, where accurate measurement of barometric pressure provides the essential altitude data for improving flight stability and landing accuracy.

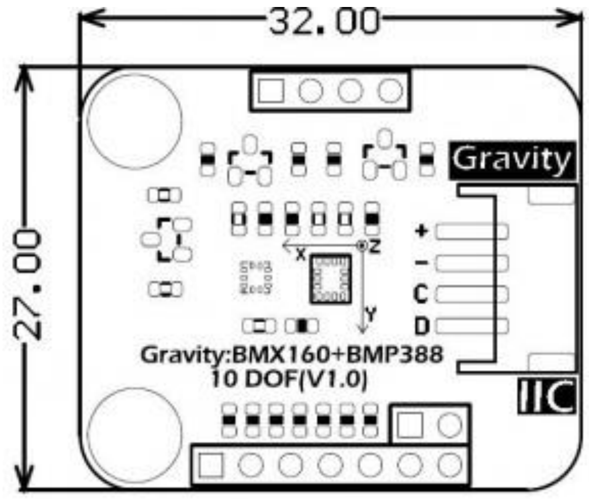
This module is a combination of BMX160 and BMP388 from DFRobot, which is perfectly suitable for using in drone applications to measure height, attitude and orientation.

Features

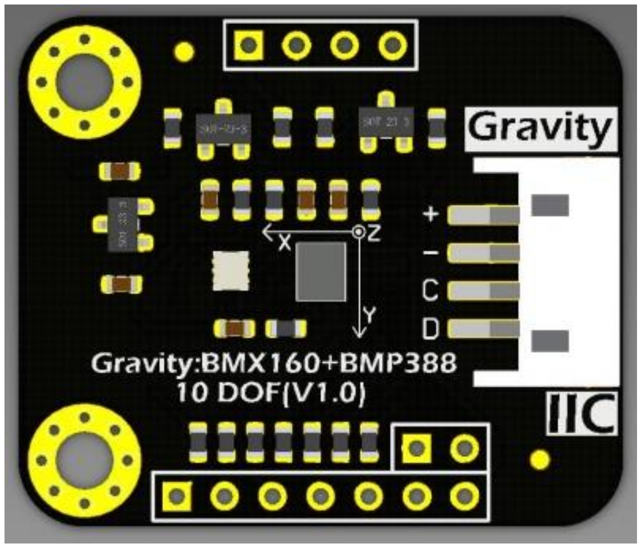
- BMX160 9-axis Sensor
    - Integrate 3 sensors: 16-bit accelerometer, 16-bit gyroscope and geomagnetic sensor
    - Smart Power Management: normal, low power, and sleep
- BMP388 Barometric Pressure & Temperature Sensor
    - Temperature Detection
    - Barometric Pressure Measurement
    - Altitude Measurement
    - Indoor Navigation (floor, elevator detection)
    - Outdoor Navigation, leisure and sports applications
    - Vertical Velocity Indication (eg. rise/sink speed)

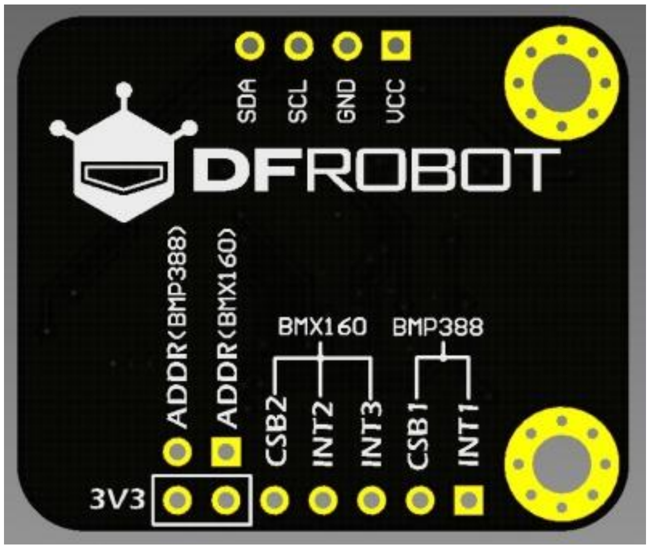
Specification

- BMX160 9-axis Sensor
    - Accelerometer: ±2g/±4g/±8g/±16
    - Gyroscope: ±125°/s~2000°/s
    - Geomagnetic Sensor: ±1150uT(x-,y-axis);±2500uT(z-axis)
    - Geomagnetic Sensor Resolution: 0.3μT
    - Default IIC Address: 0X68
- BMP388 Barometric Pressure & Temperature Sensor
    - Operation Range: 300…1250 hPa
    - Relative Accuracy Pressure: ±0.08 hPa (equivalent to ±0.66m @700-900hPa, 25℃-40℃)
    - Absolute Accuracy Pressure: ±0.5 hPa（0℃-65℃@300-1100hPa）
    - Temperature Coefficient Offset: ±0.75 Pa/K（-20℃-65℃@700-1100hPa）
    - Absolute Accuracy Temperature: ±0.5℃（@0℃-65℃）
    - Operating Temperature: -40℃~80℃ (more accurate in 0℃-65℃)
    - Default IIC Address: 0X76
- Dimension: 27mm x 32mm/1.06 x 1.26”
- Mount Hole Position: 20mm
- Mount Hole Size: inner 3mm/ outer 6mm
- Interface: Gravity-IIC PH2.0-4P



Board Overview





Silkscreen	Description
+/VCC	Positive
-/GND	Negative
C/SCL	IIC clock line
D/SDA	IIC data line
3V3	3.3V power
ADDR(BMP388)	BMP388 IIC address select
ADDR(BMX160)	BMX160 IIC address select
CSB2	BMX160 protocol select pin
INIT2	BMX160 external interrupt 2
INIT3	BMX160 external interrupt 1
CSB1	BMP388 Protocol select pin
INIT1	BMP388 external pin

API Function

```
class DFRobot_BMX160 {
/*
 * @function Gyroscope enum range, unit: G
 */
```

```
typedef enum{
 eGyroRange_2000DPS, /*Gyroscope sensitivity at 2000dps*/
 eGyroRange_1000DPS, /*Gyroscope sensitivity at 1000dps*/
 eGyroRange_500DPS,  /*Gyroscope sensitivity at 500dps*/
 eGyroRange_250DPS,  /*Gyroscope sensitivity at 250dps*/
 eGyroRange_125DPS   /*Gyroscope sensitivity at 125dps*/
}eGyroRange_t;

/*
 * @function Accelerometer enum range, unit, m/s^2
 */
typedef enum{
 eAccelRange_2G,  /* Macro for mg per LSB at +/- 2g sensitivity (1 LSB =
0.000061035mg) */
 eAccelRange_4G,  /* Macro for mg per LSB at +/- 4g sensitivity (1 LSB =
0.000122070mg) */
 eAccelRange_8G,  /* Macro for mg per LSB at +/- 8g sensitivity (1 LSB =
0.000244141mg) */
 eAccelRange_16G  /* Macro for mg per LSB at +/- 16g sensitivity (1 LSB =
0.000488281mg) */
}eAccelRange_t;

/*
 * @function reset sensor
 * @Return true if it succeeds
 */
bool softReset();

/*
 * @function init sensor
 * @Return true if it succeeds
 */
bool begin();

/*
 * @function set gyroscope range, unit: G
 * @Parameter One variable from eGyroRange_t
 */
void setGyroRange(eGyroRange_t bits);

/*
 * @function set accelerometer range, unit: m/s^2
 * @Parameter One variable from eAccelRange_t
 */
void setAccelRange(eAccelRange_t bits);

/*
 * @function Get data of accelerometer, gyroscope, geomagnetic sensor
 * @Parameter Store the address of all data
 */
void getAllData(struct bmx160SensorData *magn, struct bmx160SensorData *gyro, struct
bmx160SensorData *accel);

/*
```

```
 * @function Turn off geomagnetic sensor, gyroscope enters low power mode(there are
data output from accelerometer)
 */
void setLowPower();

/*
 * @function Turn on geomagnetic sensor, gyroscope enters normal mode
 */
void wakeUp();
```

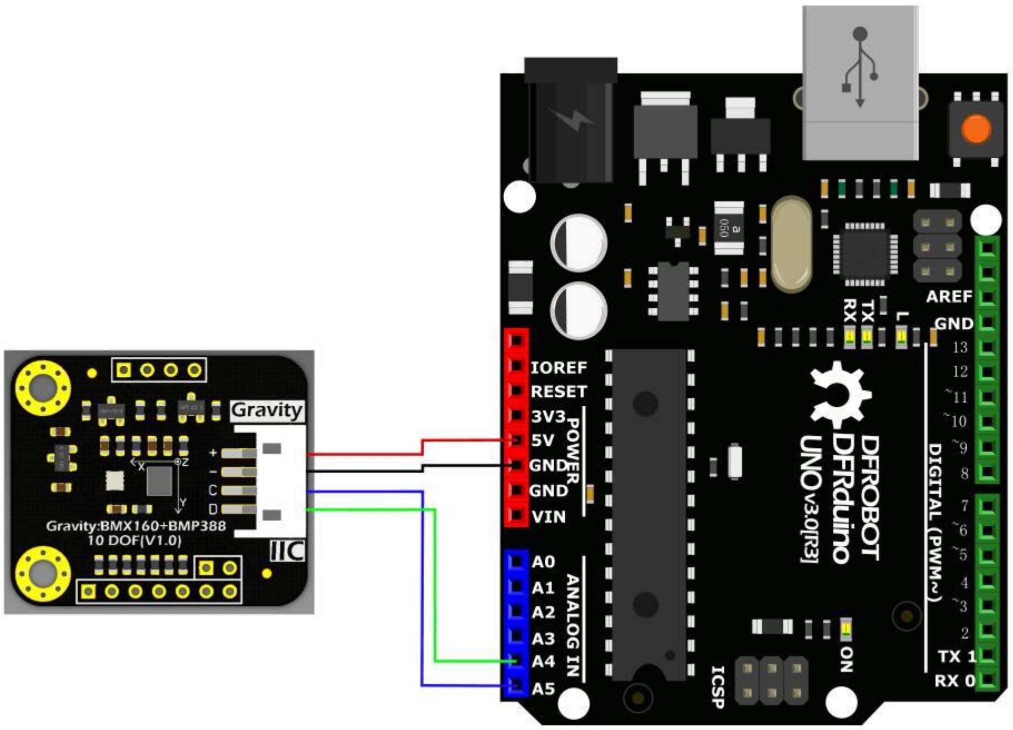
Tutorial

The 10 DOF sensor integrates BMX160 and BMP388. Visit the IIc address of BMX160 (default: 0x68) and BMP388 (0x76) via I2C interface to get the related position data and environment information.

Requirements

- **Hardware**
  - UNO Controller x 1
  - BMX160+BMP388 10 DOF Sensor x 1
  - Duponts
- **Software**
  - Arduino IDE
  - BMX160 Library
  - BMP388 Library
  - Download and install the **XXX Library** (About how to install the library?)

Connection Diagram

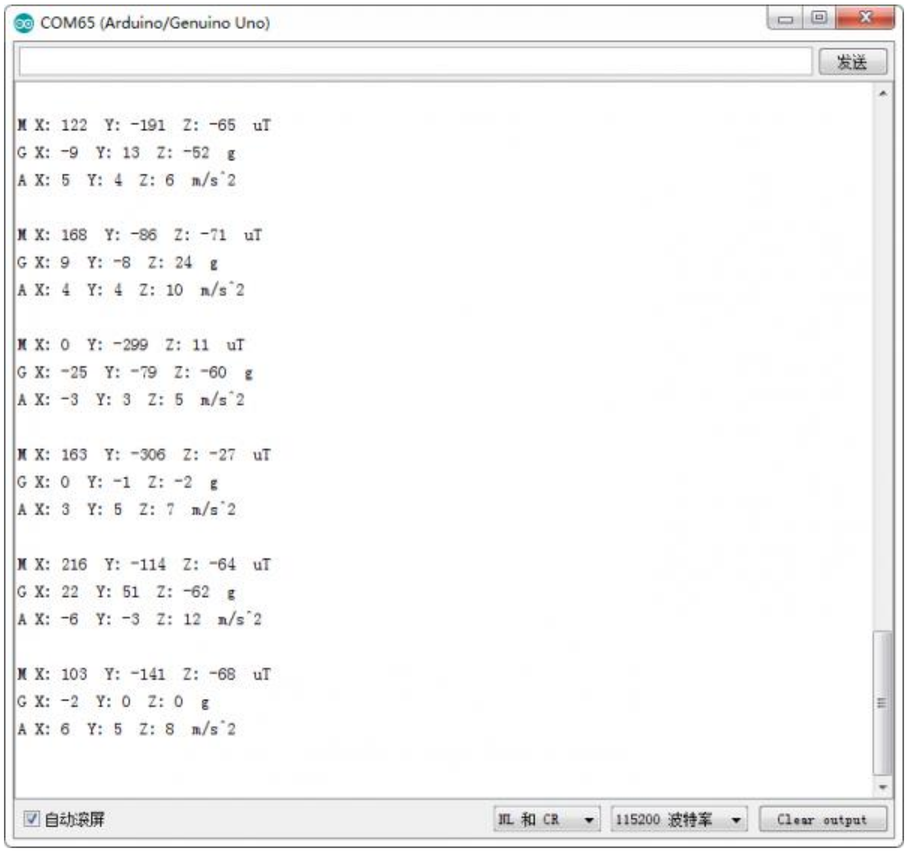


BMX160 Usage Tutorial

Program Function: read data of accelerometer, gyroscope and geomagnetic sensor of BMX160 via I2C interface, and print the readings through serial port.

```
/*!
  * file readAllData.ino
  *
  * Through the example, you can get the sensor data by using getSensorData:
  * get all data of magnetometer, gyroscope, accelerometer.
  *
  * With the rotation of the sensor, data changes are visible.
  *
  * Copyright   [DFRobot](http://www.dfrobot.com), 2016
  * Copyright   GNU Lesser General Public License
  *
  * version  V0.1
  * date  2019-6-25
  */

#include <DFRobot_BMX160.h>

DFRobot_BMX160 bmx160;
```

```
void setup(){
  Serial.begin(115200);
  delay(100);

  //init the hardware bmx160
  if (bmx160.begin() != true){
    Serial.println("init false");
    while(1);
  }
  //bmx160.setLowPower();   //disable the gyroscope and accelerometer sensor
  //bmx160.wakeUp();        //enable the gyroscope and accelerometer sensor
  //bmx160.softReset();     //reset the sensor

  /** @typedef enum{eGyroRange_2000DPS,
   *                eGyroRange_1000DPS,
   *                eGyroRange_500DPS,
   *                eGyroRange_250DPS,
   *                eGyroRange_125DPS
   *                }eGyroRange_t;
   **/
  //bmx160.setGyroRange(eGyroRange_500DPS);

  /** @typedef enum{eAccelRange_2G,
   *                eAccelRange_4G,
   *                eAccelRange_8G,
   *                eAccelRange_16G
   *                }eAccelRange_t;
   */
  //bmx160.setAccelRange(eAccelRange_4G);
  delay(100);
}

void loop(){
  bmx160SensorData Omagn, Ogyro, Oaccel;

  /* Get a new sensor event */
  bmx160.getAllData(&Omagn, &Ogyro, &Oaccel);

  /* Display the magnetometer results (magn is magnetometer in uTesla) */
  Serial.print("M ");
  Serial.print("X: "); Serial.print(Omagn.x); Serial.print("  ");
  Serial.print("Y: "); Serial.print(Omagn.y); Serial.print("  ");
  Serial.print("Z: "); Serial.print(Omagn.z); Serial.print("  ");
  Serial.println("uT");

  /* Display the gyroscope results (gyroscope data is in g) */
  Serial.print("G ");
  Serial.print("X: "); Serial.print(Ogyro.x); Serial.print("  ");
  Serial.print("Y: "); Serial.print(Ogyro.y); Serial.print("  ");
  Serial.print("Z: "); Serial.print(Ogyro.z); Serial.print("  ");
  Serial.println("g");

  /* Display the accelerometer results (accelerometer data is in m/s^2) */
  Serial.print("A ");
  Serial.print("X: "); Serial.print(Oaccel.x    ); Serial.print("  ");
  Serial.print("Y: "); Serial.print(Oaccel.y    ); Serial.print("  ");
  Serial.print("Z: "); Serial.print(Oaccel.z    ); Serial.print("  ");
  Serial.println("m/s^2");

  Serial.println("");

  delay(500);
}
```



```
M X: 122  Y: -191  Z: -65  uT
G X: -9  Y: 13  Z: -52  g
A X: 5  Y: 4  Z: 6  m/s^2

M X: 168  Y: -86  Z: -71  uT
G X: 9  Y: -8  Z: 24  g
A X: 4  Y: 4  Z: 10  m/s^2

M X: 0  Y: -299  Z: 11  uT
G X: -25  Y: -79  Z: -60  g
A X: -3  Y: 3  Z: 5  m/s^2

M X: 163  Y: -306  Z: -27  uT
G X: 0  Y: -1  Z: -2  g
A X: 3  Y: 5  Z: 7  m/s^2

M X: 216  Y: -114  Z: -64  uT
G X: 22  Y: 51  Z: -62  g
A X: -6  Y: -3  Z: 12  m/s^2

M X: 103  Y: -141  Z: -68  uT
G X: -2  Y: 0  Z: 0  g
A X: 6  Y: 5  Z: 8  m/s^2
```

BMP388 Usage Tutorial

For detailed usage of BMP388, refer to BMP388 wiki.

The default BMP388 SDO pin is Low, IIC address: BMP3_I2C_ADDR_PRIM

FAQ

For any questions, advice or cool ideas to share, please visit the **DFRobot Forum**

More Documents

- Schematic Diagram
- BNO055 Datasheet
- BMP280 Datasheet