**BMX160 9-axis Absolute Orientation Sensor Module Wiki - DFRobot**



**Introduction**

This compact BMX160 9-axis sensor from Bosch Sensortec is an ideal solution for applications that face strict constraints of board space, power consumption and appearance like wearable fitness trackers, smart remote controls, quadcopters and so on.
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.
Due to the compact design, BMX160 can be ideally integrated into wearables like smart watches or glasses for augmented reality. Combined with BSX sensor data fusion software library of Bosch Sensortec, the sensor performance can be further improved.

**Features**

* BMX160 9-Axis Sensor
	+ Integrated Three Sensors:
		- 16-bit Gyroscope
		- 16-bit Accelerometer
		- Geomagnetic Sensor
	+ Intelligent Power Management System: normal, low power, sleep

**Specification**



* Operating Voltage: 3.3V-5.5V
* BMX160 Accelerometer:
	+ Accelerometer: ±2g/±4g/±8g/±16
* BMX160 Gyroscope:
	+ Range: ±125°/s ~ 2000°/s
* BMX160 Geomagnetic:
	+ Range: ±1150uT(x-,y-axis);±2500uT(z-axis)
	+ Resolution: ~0.3
* Outline Dimension: 20 x 12.5mm/0.79 x 0.49”
* Mounting Hole Position: 15mm
* Mounting Hole Size: inner diameter 2mm/outer diameter 3.7mm

**Board Overview**





| **Silkscreen** | **Function Description** |
| --- | --- |
| VCC | + |
| GND | - |
| SCL | I2C Clock |
| SDA | I2C Data |
| CSB | BMX160 Protocol Select Pin |
| INIT1 | BMX160 External Interrupt 1 |
| INIT2 | BMX160 External Interrupt 2 |
| ADDR | I2C Address Select |

**API Functions**

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**

Visit the I2C address of BMX160 via I2C interface to get the related position data.

**Requirements**

* **Hardware**
	+ [DFRduino UNO R3](https://www.dfrobot.com/product-838.html) (or similar) x 1
	+ BMX160 9-axis Sensor Module x1
	+ Jumper wires
* **Software**
	+ [Arduino IDE](https://www.arduino.cc/en/Main/Software)
	+ Download and install the [**BMX160 Library**](https://github.com/DFRobot/DFRobot_BMX160) ([About how to install the library?](https://www.arduino.cc/en/Guide/Libraries#.UxU8mdzF9H0))

**Connection Diagram**



**BMX160 Tutorial**

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

/\*!

 \* @file readAllData.ino

 \* @brief Through the example, you can get the sensor data by using getSensorData:

 \* @n get all data of magnetometer, gyroscope, accelerometer.

 \* @n With the rotation of the sensor, data changes are visible.

 \* @copyright Copyright (c) 2010 DFRobot Co.Ltd (http://www.dfrobot.com)

 \* @license The MIT License (MIT)

 \* @author [luoyufeng] (yufeng.luo@dfrobot.com)

 \* @maintainer [Fary](feng.yang@dfrobot.com)

 \* @version V1.0

 \* @date 2021-10-20

 \* @url https://github.com/DFRobot/DFRobot\_BMX160

 \*/

#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

 /\*\*

 \* enum{eGyroRange\_2000DPS,

 \* eGyroRange\_1000DPS,

 \* eGyroRange\_500DPS,

 \* eGyroRange\_250DPS,

 \* eGyroRange\_125DPS

 \* }eGyroRange\_t;

 \*\*/

 //bmx160.setGyroRange(eGyroRange\_500DPS);

 /\*\*

 \* enum{eAccelRange\_2G,

 \* eAccelRange\_4G,

 \* eAccelRange\_8G,

 \* eAccelRange\_16G

 \* }eAccelRange\_t;

 \*/

 //bmx160.setAccelRange(eAccelRange\_4G);

 delay(100);

}

void loop(){

 sBmx160SensorData\_t 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);

}

Copy

**Expected Results**

