Gyroscope & Accelerometer Module

BMS56M605
Arduino Library Description

Revision: V1.00 Date: April 18, 2023

www.bestmodulescorp.com

Contents

Introduction

The BMS56M605 is a gyroscope and accelerometer module from Best Modules, which uses the I²C communication method. This document provides the description of the BMS56M605 Arduino Lib functions and how to install the Arduino Lib. The example demonstrates the functions of reading the 6-axis data and motion detection.

Arduino Lib Functions

Arduino Lib Name: BMS56M605		Lib Version: V1.0.1
Constructors & Initialisation		
1	BMS56M605(uint8_t intPin=8, TwoWire *theWire=&Wire)	
	Description	Constructor
	Parameter	intPin: INT pin *theWire: I²C communication interface selection
	Return Value	—
	Note	—
2	void begin(uint8_t addr=BMS56M605_IICADDR)	
	Description	Initialise module
	Parameter	addr: slave address, defaults to 0x68
	Return Value	void
	Note	—
Performance Functions		
3	void getEvent()	
	Description	Obtain data, including temperature value and 6-axis data
	Parameter	—
	Return Value	void
	Note	Stored respectively in class public variables float temperature; float accX, accY, accZ; float gyroX, gyroY, gyroZ
4	float readTemperature()	
	Description	Obtain the temperature value
	Parameter	—
	Return Value	Temperature value, unit: °C
	Note	—
5	float readAccelerationX()	
	Description	Obtain the X-axis acceleration value
	Parameter	—
	Return Value	X-axis acceleration value, unit: g
	Note	—
6	float readAccelerationY()	
	Description	Obtain the Y-axis acceleration value
	Parameter	—
	Return Value	Y-axis acceleration value, unit: g
	Note	—

7	float readAccelerationZ()	
	Description	Obtain the Z-axis acceleration value
	Parameter	—
	Return Value	Z-axis acceleration value, unit: g
	Note	—
8	float readGyroscopeX()	
	Description	Obtain the X-axis gyroscope value
	Parameter	—
	Return Value	X-axis gyroscope value, unit: °/s
	Note	—
9	float readGyroscopeY()	
	Description	Obtain the Y-axis gyroscope value
	Parameter	—
	Return Value	Y-axis gyroscope value, unit: °/s
	Note	—
10	float readGyroscopeZ()	
	Description	Obtain the Z-axis gyroscope value
	Parameter	—
	Return Value	Z-axis gyroscope value, unit: °/s
	Note	—
11	void setInterruptPinPolarity(uint8_t active_level)	
	Description	Setup interrupt pin polarity
	Parameter	active_level: 0(ACTIVE_HIGH): Active high 1(ACTIVE_LOW): Active low
	Return Value	void
	Note	—
12	void setINT(uint8_t mode, uint8_t isEnable=false)	
	Description	Setup interrupt source
	Parameter	mode: 5(ZERO_MOTION_MODE): ZERO MOTION MODE 6(MOTION_MODE): MOTION MODE 7(FREE_FALL_MODE): FREE FALL MODE isEnable: true: Enable false: Disable (default)
	Return Value	void
	Note	—
13	uint8_t getINT()	
	Description	Get INT pin level
	Parameter	—
	Return Value	INT pin level 1: High 0: Low
	Note	—
14	void writeReg(uint8_t addr, uint8_t data)	
	Description	Write register
	Parameter	addr: register address data: data to written
	Return Value	void
	Note	—

15	uint8_t readReg(uint8_t addr)	
	Description	Read register, 1 byte
	Parameter	addr: register address
	Return Value	Register value
	Note	—
16	void readReg(uint8_t addr, uint8_t rBuf[], uint8_t rLen)	
	Description	Continuously read registers, rLen bytes, stored in rBuf[] array
	Parameter	addr: register address rBuf[]: used to store read data rLen: number of continuously read bytes
	Return Value	void
	Note	—
17	void enableSleep(bool isEnable=false)	
	Description	Enable to enter Sleep mode
	Parameter	isEnable: true – Enable false – Disable (default)
	Return Value	void
	Note	—
18	void enableCycle(bool isEnable=false)	
	Description	Enable the Cycle Wake-up mode
	Parameter	isEnable: true – Enable false – Disable (default)
	Return Value	void
	Note	When the Sleep mode is disabled and the Cycle Wake-up mode is enabled, the module will be woken up at regular intervals, the wake-up rate is set by the setCycleRate function
19	void reset()	
	Description	Reset
	Parameter	—
	Return Value	void
	Note	Reset all registers to default values
Parameter Read & Configuration		
20	uint8_t getAccelerometerRange()	
	Description	Obtain the accelerometer range
	Parameter	—
	Return Value	Accelerometer range: 0: ±2g 1: ±4g 2: ±8g 3: ±16g
	Note	—
21	uint8_t getGyroRange()	
	Description	Obtain the gyroscope range
	Parameter	—
	Return Value	Gyroscope range: 0: ±250°/s 1: ±500°/s 2: ±1000°/s 3: ±2000°/s
	Note	—

22	uint8_t getFreefallThreshold()	
	Description	Obtain the free fall threshold
	Parameter	—
	Return Value	Free fall threshold, unit: mg
	Note	—
23	uint8_t getFreefallDuration()	
	Description	Obtain the free fall duration
	Parameter	—
	Return Value	Free fall duration, unit: ms
	Note	—
24	uint8_t getMotionThreshold()	
	Description	Obtain the motion detection threshold
	Parameter	—
	Return Value	Motion detection threshold, unit: mg
	Note	—
25	uint8_t getMotionDuration()	
	Description	Obtain the motion detection duration
	Parameter	—
	Return Value	Motion detection duration, unit: ms
	Note	—
26	uint8_t getZeroMotionThreshold()	
	Description	Obtain the zero motion detection threshold
	Parameter	—
	Return Value	Zero motion detection threshold, unit: mg
	Note	—
27	uint8_t getZeroMotionDuration()	
	Description	Obtain the zero motion detection duration
	Parameter	—
	Return Value	Zero motion detection duration, unit: ms
	Note	—
28	uint8_t getClock()	
	Description	Obtain the clock source
	Parameter	—
	Return Value	Clock source 0: internal 8MHz oscillator 1: PPL with X-axis gyroscope reference 2: PPL with Y-axis gyroscope reference 3: PPL with Z-axis gyroscope reference 4: PPL with external 32.768kHz reference 5: PPL with external 19.2MHz reference 6: reserved 7: stop the clock and keep the timer generator reset
	Note	—

29	uint8_t getFilterBandwidth()	
	Description	Obtain the filter bandwidth
	Parameter	—
	Return Value	Band pass filter range 0: ACC bandwidth 260Hz, GYRO bandwidth 256Hz 1: ACC bandwidth 184Hz, GYRO bandwidth 188Hz 2: ACC bandwidth 96Hz, GYRO bandwidth 98Hz 3: ACC bandwidth 44Hz, GYRO bandwidth 42Hz 4: ACC bandwidth 21Hz, GYRO bandwidth 20Hz 5: ACC bandwidth 10Hz, GYRO bandwidth 10Hz 6: ACC bandwidth 5Hz, GYRO bandwidth 5Hz
	Note	—
30	uint8_t getSampleRateDivisor()	
	Description	Obtain the sample rate divisor
	Parameter	—
	Return Value	Sample rate divisor
	Note	Sample rate=Gyroscope Output Rate/(1+divisor) The sampling rate can also be set by writing the Configuration CONFIG register
31	uint8_t getCycleRate()	
	Description	Obtain the cycle wake-up rate
	Parameter	—
	Return Value	Cycle wake-up rate 0: 1.25Hz 1: 2.5Hz 2: 5Hz 3: 10Hz
	Note	—
32	void setAccelerometerRange(uint8_t range)	
	Description	Obtain the accelerometer range
	Parameter	range: 0(ACC_RANGE_2G): ±2g 1(ACC_RANGE_4G): ±4g 2(ACC_RANGE_8G): ±8g 3(ACC_RANGE_16G): ±16g
	Return Value	void
	Note	—
33	void setGyroRange(uint8_t range)	
	Description	Obtain the gyroscope range
	Parameter	range: 0(GYRO_RANGE_250): ±250°/s 1(GYRO_RANGE_500): ±500°/s 2(GYRO_RANGE_1000): ±1000°/s 3(GYRO_RANGE_2000): ±2000°/s
	Return Value	void
	Note	—
34	void setFreefallThreshold(uint8_t threshold)	
	Description	Setup free fall threshold
	Parameter	threshold: free fall threshold, 1 LSB=1mg
	Return Value	void
	Note	—

35	void setFreefallDuration(uint8_t duration)	
	Description	Setup free fall duration
	Parameter	duration: free fall duration, 1 LSB=1ms
	Return Value	void
	Note	—
36	void setMotionThreshold(uint8_t threshold)	
	Description	Setup motion detection threshold
	Parameter	threshold: motion detection threshold, 1 LSB=1mg
	Return Value	void
	Note	—
37	void setMotionDuration(uint8_t duration)	
	Description	Setup motion detection duration
	Parameter	duration: motion detection duration, 1 LSB=1ms
	Return Value	void
	Note	—
38	void setZeroMotionThreshold(uint8_t threshold)	
	Description	Setup zero motion detection threshold
	Parameter	threshold: zero motion detection threshold, 1 LSB=1mg
	Return Value	void
	Note	—
39	void setZeroMotionDuration(uint8_t duration)	
	Description	Setup zero motion detection duration
	Parameter	duration: zero motion detection duration, 1 LSB=1ms
	Return Value	void
	Note	—
40	void setClock(uint8_t clock)	
	Description	Setup clock source
	Parameter	clock: 0(INTERAL_8MHZ): internal 8MHz oscillator 1(PLL_X_GYRO): PPL with X-axis gyroscope reference 2(PLL_Y_GYRO): PPL with Y-axis gyroscope reference 3(PLL_Z_GYRO): PPL with Z-axis gyroscope reference 4(PLL_EXTERNAL_32_768kHZ): PPL with external 32.768kHz reference 5(PLL_EXTERNAL_19_2MHZ): PPL with external 19.2MHz reference 6: reserved 7(STOP_CLOCK): stop the clock and keep the timer generator reset
	Return Value	void
	Note	—
41	void setFilterBandwidth(uint8_t band)	
	Description	Setup filter bandwidth
	Parameter	band: 0(ACC_260HZ_GYRO_256HZ): ACC bandwidth 260Hz, GYRO bandwidth 256Hz 1(ACC_184HZ_GYRO_188HZ): ACC bandwidth 184Hz, GYRO bandwidth 188Hz 2(ACC_96HZ_GYRO_98HZ): ACC bandwidth 96Hz, GYRO bandwidth 98Hz 3(ACC_44HZ_GYRO_42HZ): ACC bandwidth 44Hz, GYRO bandwidth 42Hz 4(ACC_21HZ_GYRO_20HZ): ACC bandwidth 21Hz, GYRO bandwidth 20Hz 5(ACC_10HZ_GYRO_10HZ): ACC bandwidth 10Hz, GYRO bandwidth 10Hz 6(ACC_5HZ_GYRO_5HZ): ACC bandwidth 5Hz, GYRO bandwidth 5Hz
	Return Value	void
	Note	—

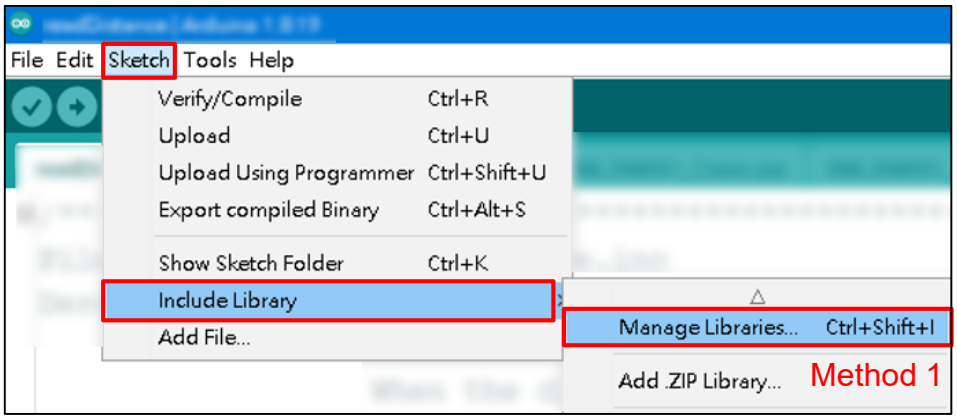
42	void setSampleRateDivisor(uint8_t divisor)	
	Description	Setup sample rate divisor
	Parameter	divisor: sample rate divisor
	Return Value	void
	Note	sample rate=Gyroscope Output Rate/(1+divisor)
43	void setCycleRate(uint8_t rate)	
	Description	Setup cycle wake-up rate
	Parameter	rate: 0(F_1_25HZ): 1.25Hz 1(F_2_5HZ): 2.5Hz 2(F_5HZ): 5Hz 3(F_10HZ): 10Hz
	Return Value	void
	Note	—

Arduino Lib Download and Installation

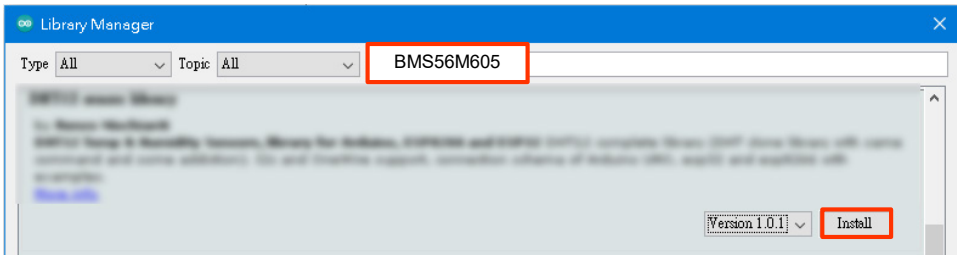
Refer to the following two methods to install the BMS56M605 Arduino Library.

Method 1: Search for installation

Search for installation: Arduino IDE→Sketch→Include Library→Manage Libraries→Search for "BMS56M605"→Install



Search for Installation Step 1

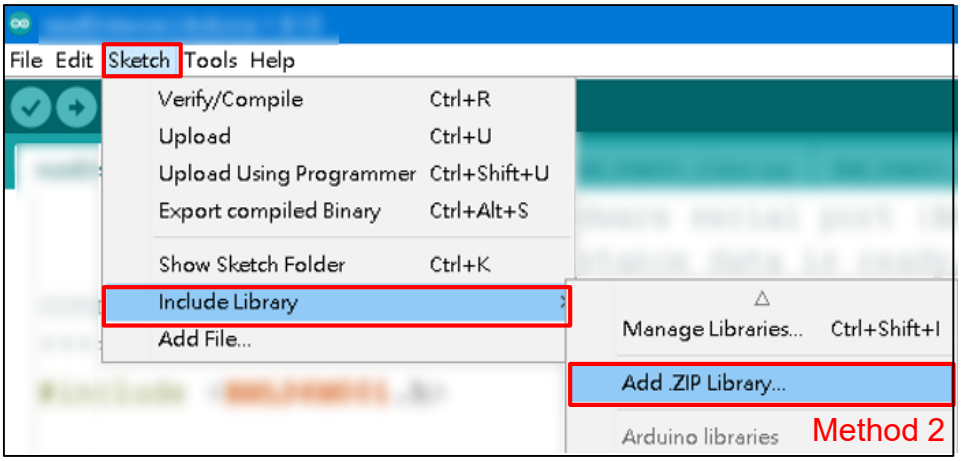


Search for Installation Step 2

Method 2: Download before adding a ZIP library

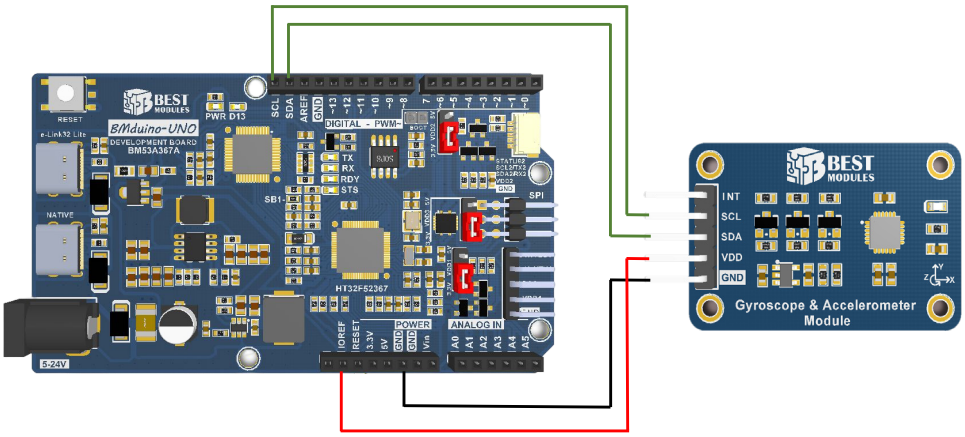
Download method: Open the Best Modules official website (https://www.bestmodulescorp.com/bms56M605.html#tab-product2) and download the BMS56M605 Library from "Arduino example program" under the "DOCUMENTS" menu.

Add .ZIP library: Arduino IDE→Sketch→Include Library→Add .ZIP Library....



Arduino Example

Example1: readAccelerationAndGyroscopeAndTemperature



Physical Connection Diagram

Example 1 function: It communicates with the module using the I²C interface to obtain the sensor 6-axis data and the temperature value and display it in "Serial Monitor".

1. Open an example program:

   File→Examples→Select Lib "BMS56M605"→Select the corresponding example program

2. Example program description:

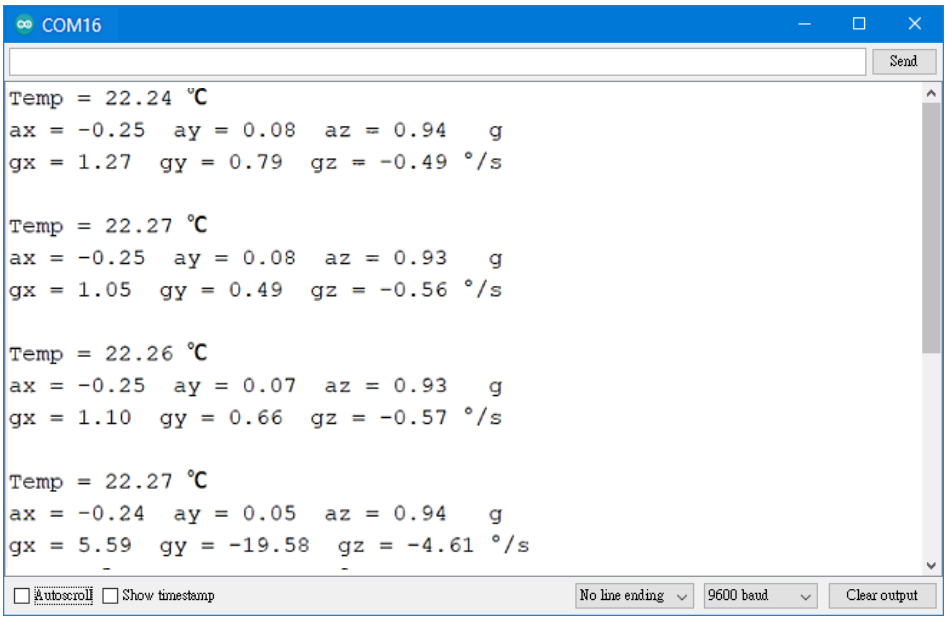
a. Create & Initialise objects

```
#include "BMS56M605.h"
BMS56M605 Mpu(8);       // Connect INT pin to D8
void setup()
{
  Mpu.begin();          // Initialise the module
  Serial.begin(9600);   // Initialise the serial monitor, select the
                        // baud rate to be 9600
}
```

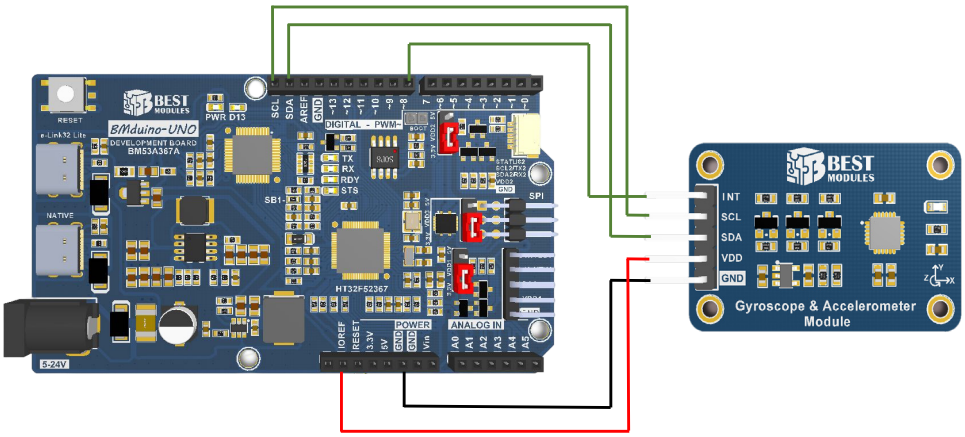
b. Obtain the sensor 6-axis data and the temperature value and display in Serial Monitor

```
void loop()
{
  Mpu.getEvent(); // Obtain a 3-axis acceleration, an angular velocity
                  // and an ambient temperature
/* Output the temperature in °C */
  Serial.print("Temp = ");
  Serial.print(Mpu.temperature);
  Serial.println(" °C");
/* Output the 3-axis acceleration in g */
  Serial.print("ax = ");
  Serial.print(Mpu.accX);
  Serial.print("ay = ");
  Serial.print(Mpu.accY);
  Serial.print("az = ");
  Serial.print(Mpu.accZ);
  Serial.println("   g");
/* Output the 3-axis angular velocity in °/s */
  Serial.print("gx = ");
  Serial.print(Mpu.gyroX);
  Serial.print("gy = ");
  Serial.print(Mpu.gyroY);
  Serial.print("gz = ");
  Serial.print(Mpu.gyroZ);
  Serial.println(" °/s");
  Serial.println();
  delay(1000);
}
```

3. Open the Serial Monitor and select the baud rate to be 9600. The serial monitor will display as follows.



Example2: getMotionStatus



Physical Connection Diagram

Example 2 function: It communicates with the module using the I²C mode to obtain the motion detection. Every time the module is shaken, the serial port will output a number once, and accumulate to 0xffff and then clear to zero, finally display it on serial monitor.

1. Open an example program:

File→Examples→Select Lib “BMS56M605”→Select the corresponding example program

2. Example program description:

a. Create & Initialise objects

```
#include "BMS56M605.h"
BMS56M605 Mpu(8);       // INT Pin connects to D8
uint16_t cnt = 0;
void setup()
{
  Mpu.begin();          // Initialise the module
  Serial.begin(9600);   // Initialise the serial monitor, select the
                        // baud rate to be 9600
  Mpu.setINT(MOTION_MODE,true);         // Select the motion detection
                                        // interrupt
  Mpu.setInterruptPinPolarity(active_LOW); // Select the interrupt pin
                                           // to be active low
  Mpu.setMotionThreshold(1);  // Setup the motion threshold to 1mg
  Mpu.setMotionDuration(30);  // Triggered when the motion duration
                              // time is set to 30ms
}
```

b. Obtain the sensor 6-axis data and the temperature value and display it in "Serial Monitor".

```
void loop()
{
  if( Mpu.getINT () == 0)      // Obtain the interrupt pin state
  {
    cnt++;
    if(cnt >= 0xffff) cnt = 0;
    Serial.print("Motion Detected!  x");
    Serial.println(cnt);
    Serial.println();
  }
}
```

3. Open the Serial Monitor and select the baud rate to be 9600. The serial monitor will display as follows.