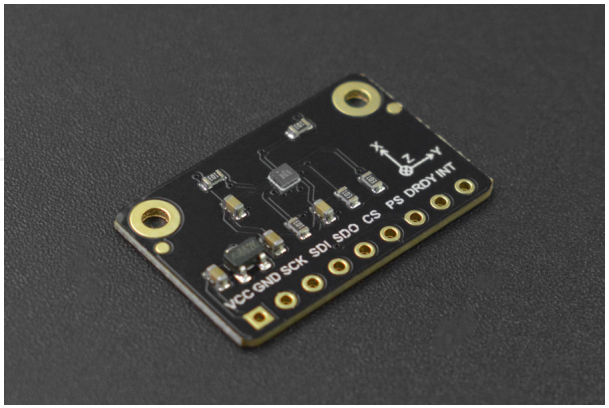
SKU:SEN0419 (https://www.dfrobot.com/product-2507.html)

(https://www.dfrobot.com/product-2507.html)



Introduction

The BMM150 is a low-power and low noise 3-axis digital geomagnetic sensor that perfectly matches the requirements of compass applications. Based on Bosch's proprietary FlipCore technology, the BMM150 provides absolute spatial orientation and motion vectors with high accuracy and dynamics. Featuring small size and lightweight, it is also especially suited for supporting drones in accurate heading. The BMM150 can also be used together with an inertial measurement unit consisting of a 3-axis accelerometer and a 3-axis gyroscope.

Features

- Low power
- Low noise
- High accuracy
- Small in size and lightweight

Application

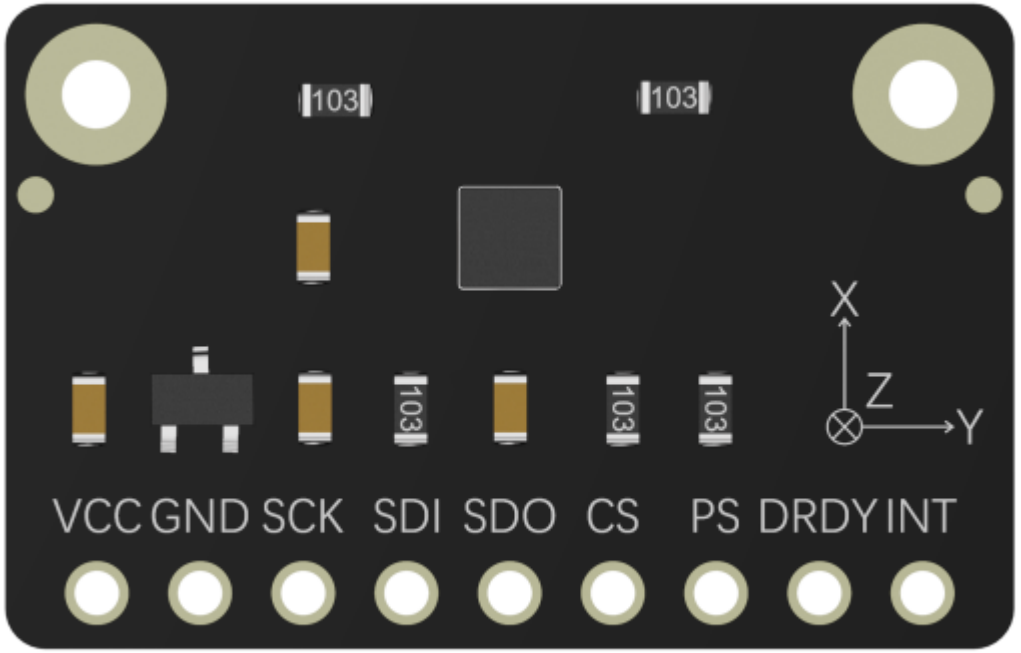
- Drones
- Gaming
- Indoor/Outdoor navigation

- Magnetic heading information
- Augmented reality
- Tilt-compensated electronic compass for map rotation

Specification

- Power Supply: 3.3V
- Temperature Range: -40~85°C
- Digital Interface: I2C/SPI
- Programmable Interrupt
- Resolution: 0.3uT
- Zero-B Offset: ±40uT/±2uT(Software optimized)
- Non-linearity: <1%FS
- Magnetic Range: ±1300μT (x,y-axis)，±2047μT (z-axis)
- Average Current Consumption: 170 μA (low power preset); 500 μA (normal mode)
- Low Noise: 0.3-1.4uT
- Magnetometer Heading Accuracy: 30μT horizontal geomagnetic field component, ±2.5°
- Start-up Time: 3ms

Board Overview



No.	Silkscreen	Function
1	VCC	3.3V Power Input
2	SCK	Clock line

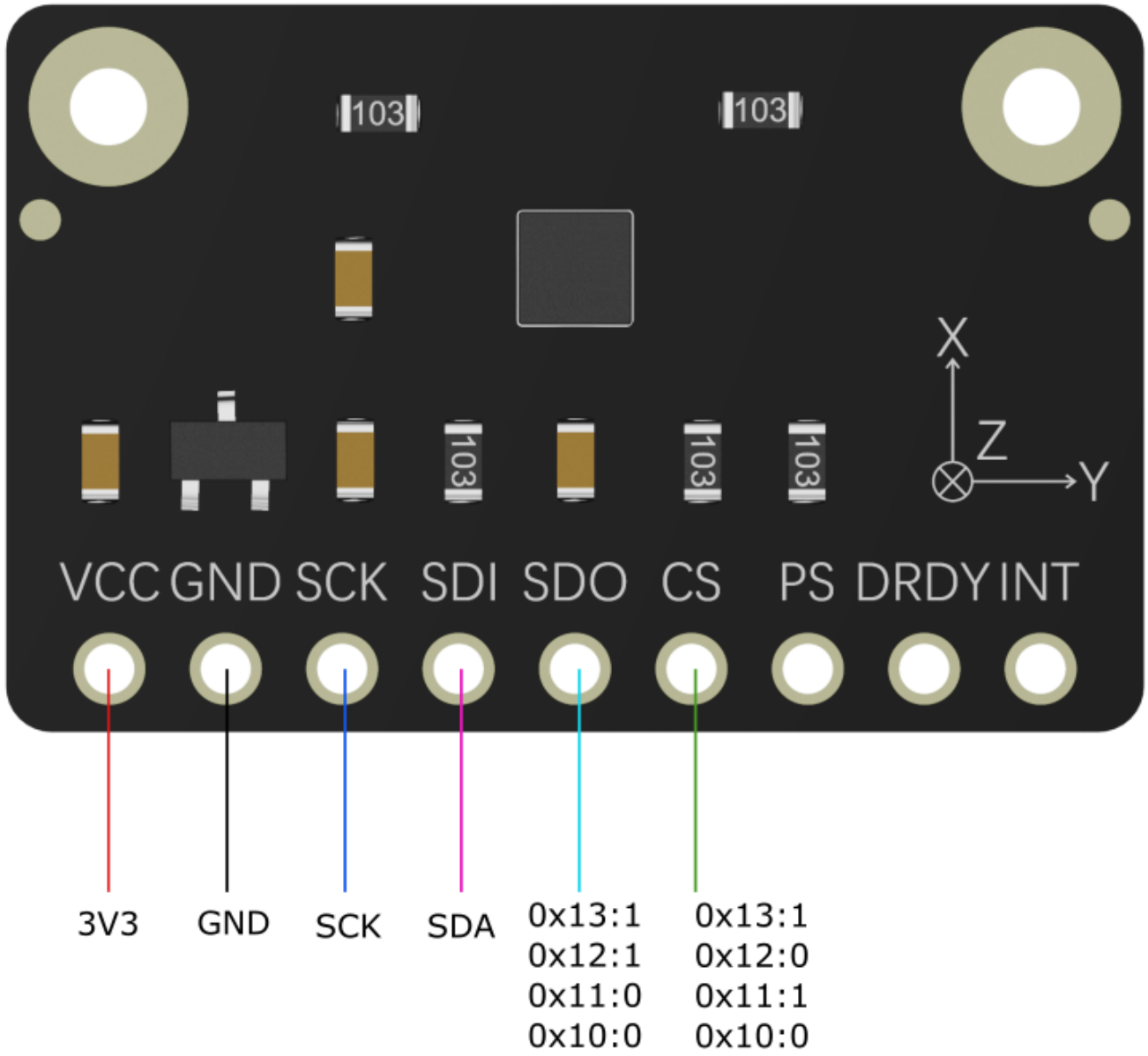
No.	Silkscreen	Function
3	SDI	SPI/I2C data line(input)
4	SDO	SPI/I2C data line(output)
5	CS	SPI chip-select, I2C Ground
6	PS	Select communication protocol
7	DRDY	Data Receiving/Transmitting ready status
8	INT	Interrupt Pin

Note:

- All voltages of data outputs are 3.3V.
- Only 3.3V input is allowed.

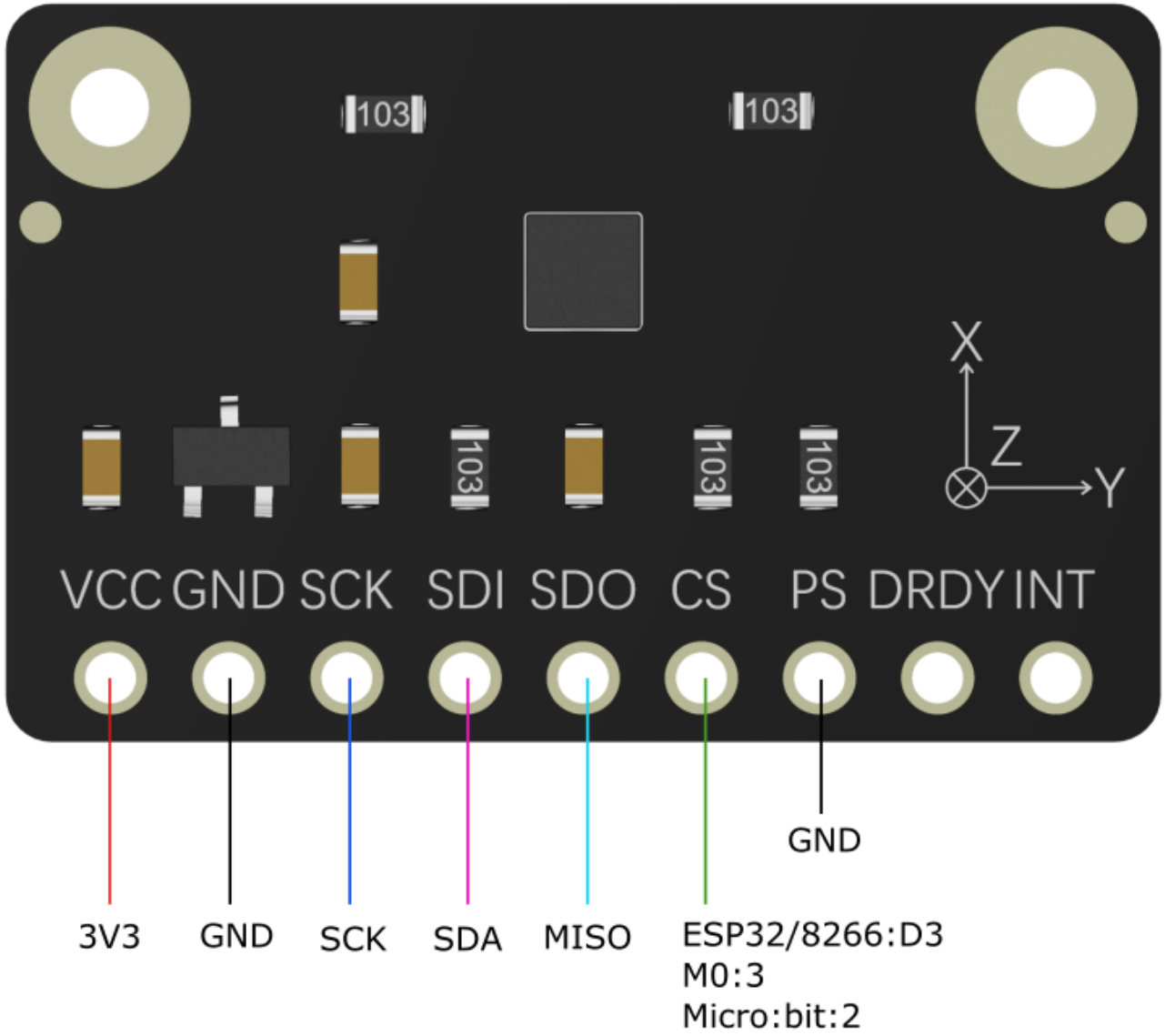
Connections for different communications

- I2C: the default address is 0x13



Note: When using magDrdyInterrupt, connect pin DRDY to the related interrupt pin of mainboard instead of the pin INT.

- Connection for SPI



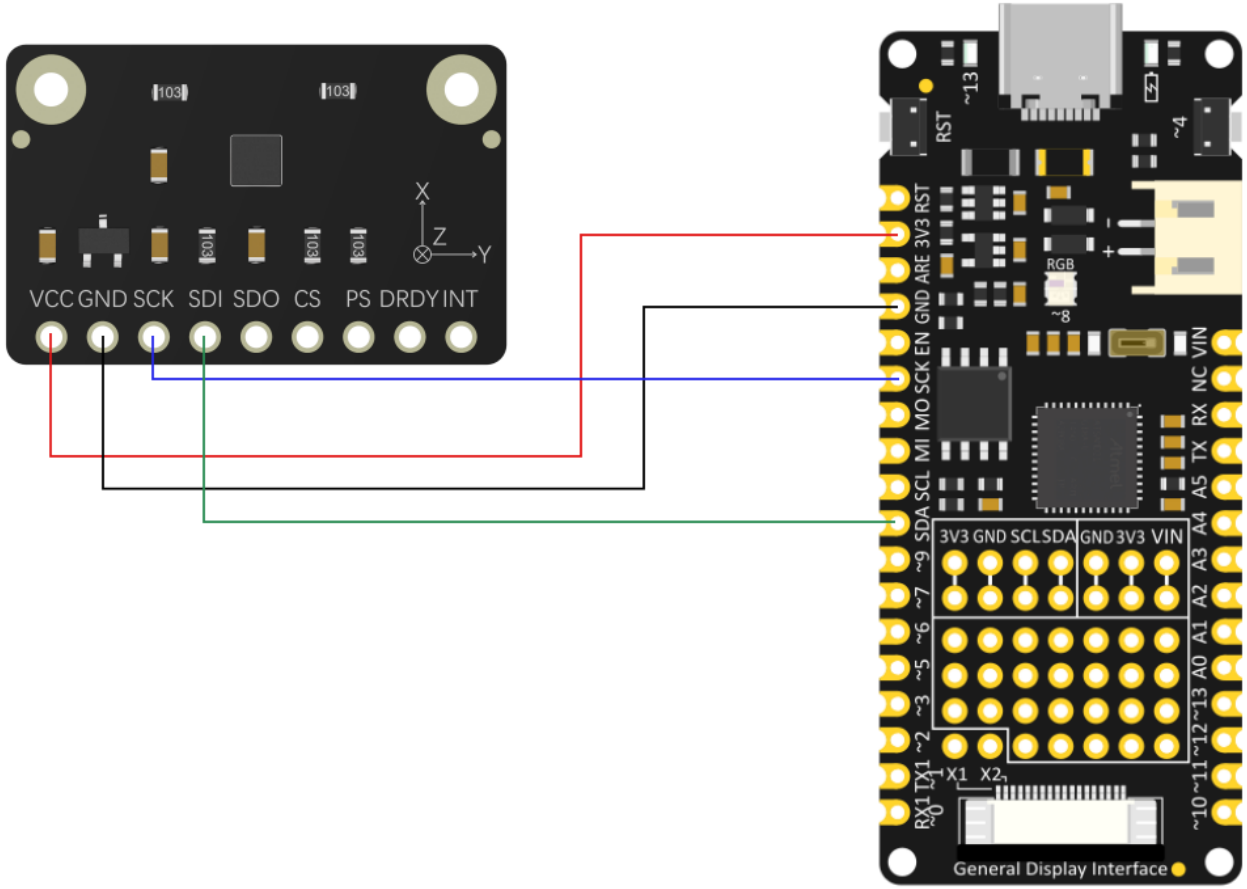
Note: Pin PS should be connected to GND for SPI connection.

- Interrupt Pin Connection

Mainboard	Pin
Leonardo	D3
micro:bit	P0
ESP32/ESP8266/ARDUINO_SAM_ZERO（M0）	D6
Raspberry Pi	GPIO25
Only needs to be connected when interrupt is necessary	

Tutorial for Using on M0

Connect the sensor with M0(or other mainboard) as the diagram shown below.



The default address is 0x13.

Requirements

- **Hardware**
  - Firebeetle Board-M0 (https://www.dfrobot.com/product-2095.html) (or similar) x 1
  - BMM150 Triple Axis Magnetometer Sensor x1
  - Dupont Wires

- **Software**
  - Arduino IDE (https://www.arduino.cc/en/Main/Software)
  - Download and install the **DFRobot_BMM150_library** (https://github.com/DFRobot/DFRobot_BMM150) (About how to install the library? (https://www.arduino.cc/en/Guide/Libraries#.UxU8mdzF9H0))

Sample Code

- Sample Code 1 - get configuration status(getAllState.ino)
- Sample Code 2 - get geomagnetic data, compass angle(getGeomagneticData.ino)
- Sample Code 3 - prepare interrupt function(magDrdyInterrupt.ino)
- Sample Code 4 - threshold interrupt function(thresholdInterrupt.ino)

Main API Function List


```
/**
 * @fn softReset
 * @brief Soft reset, restore to suspended mode after soft reset and then enter sleep mode,
 */
void softReset(void);

/**
 * @fn setOperationMode
 * @brief Set sensor operation mode
 * @param opMode mode
 * @n BMM150_POWERMODE_NORMAL      normal mode  Get geomagnetic data normally
 * @n BMM150_POWERMODE_FORCED      forced mode  Single measurement, the sensor restores to
 * @n BMM150_POWERMODE_SLEEP       sleep mode   Users can visit all the registers, but can’
 * @n BMM150_POWERMODE_SUSPEND     suspend mode At the time the sensor cpu doesn’t work and
 */
void setOperationMode(uint8_t opMode);

/**
 * @fn getOperationMode
 * @brief Get sensor operation mode
 * @return result Return sensor operation mode as a character string
 */
String getOperationMode(void);

/**
 * @fn setPresetMode
 * @brief Set preset mode, make it easier for users to configure sensor to get geomagnetic
 * @param presetMode
 * @n BMM150_PRESETMODE_LOWPOWER       Low power mode, get a fraction of data and take the
 * @n BMM150_PRESETMODE_REGULAR        Regular mode, get a number of data and take the mean
 * @n BMM150_PRESETMODE_ENHANCED       Enhanced mode, get a plenty of data and take the mea
 * @n BMM150_PRESETMODE_HIGHACCURACY   High accuracy mode, get a huge number of data and ta
 */
```

```
void setPresetMode(uint8_t presetMode);

/**
 * @fn setRate
 * @brief Set the rate of obtaining geomagnetic data, the higher, the faster (without delay
 * @param rate
 * @n BMM150_DATA_RATE_02HZ
 * @n BMM150_DATA_RATE_06HZ
 * @n BMM150_DATA_RATE_08HZ
 * @n BMM150_DATA_RATE_10HZ  (default rate)
 * @n BMM150_DATA_RATE_15HZ
 * @n BMM150_DATA_RATE_20HZ
 * @n BMM150_DATA_RATE_25HZ
 * @n BMM150_DATA_RATE_30HZ
 */
void setRate(uint8_t rate);

/**
 * @fn getRate
 * @brief Get the config data rate, unit: HZ
 * @return rate
 */
uint8_t getRate(void);

/**
 * @fn getGeomagneticData
 * @brief Get the geomagnetic data of 3 axis (x, y, z)
 * @return Geomagnetic data structure, unit: (uT)
 */
sBmm150MagData_t getGeomagneticData(void);

/**
 * @fn getCompassDegree
 * @brief Get compass degree
 * @return Compass degree (0° - 360°)
 * @n      0° = North, 90° = East, 180° = South, 270° = West.
```

```
 */
float getCompassDegree(void);

/**
 * @fn setDataReadyPin
 * @brief Enable or disable data ready interrupt pin
 * @n After enabling, the DRDY pin jump when there's data coming.
 * @n After disabling, the DRDY pin will not jump when there's data coming.
 * @n High polarity: active on high, the default is low level, which turns to high level wh
 * @n Low polarity: active on low, default is high level, which turns to low level when the
 * @param modes
 * @n     DRDY_ENABLE        Enable DRDY
 * @n     DRDY_DISABLE       Disable DRDY
 * @param polarity
 * @n     POLARITY_HIGH      High polarity
 * @n     POLARITY_LOW       Low polarity
 */
void setDataReadyPin(uint8_t modes, uint8_t polarity=POLARITY_HIGH);

/**
 * @fn getDataReadyState
 * @brief Get the data ready status, determine whether the data is ready
 * @return status
 * @n true  Data ready
 * @n false Data is not ready
 */
bool getDataReadyState(void);

/**
 * @fn setMeasurementXYZ
 * @brief Enable the measurement at x-axis, y-axis and z-axis, default to be enabled. After
 * @param channelX
 * @n   MEASUREMENT_X_ENABLE        Enable the measurement at x-axis
 * @n   MEASUREMENT_X_DISABLE       Disable the measurement at x-axis
 * @param channelY
 * @n   MEASUREMENT_Y_ENABLE        Enable the measurement at y-axis
```

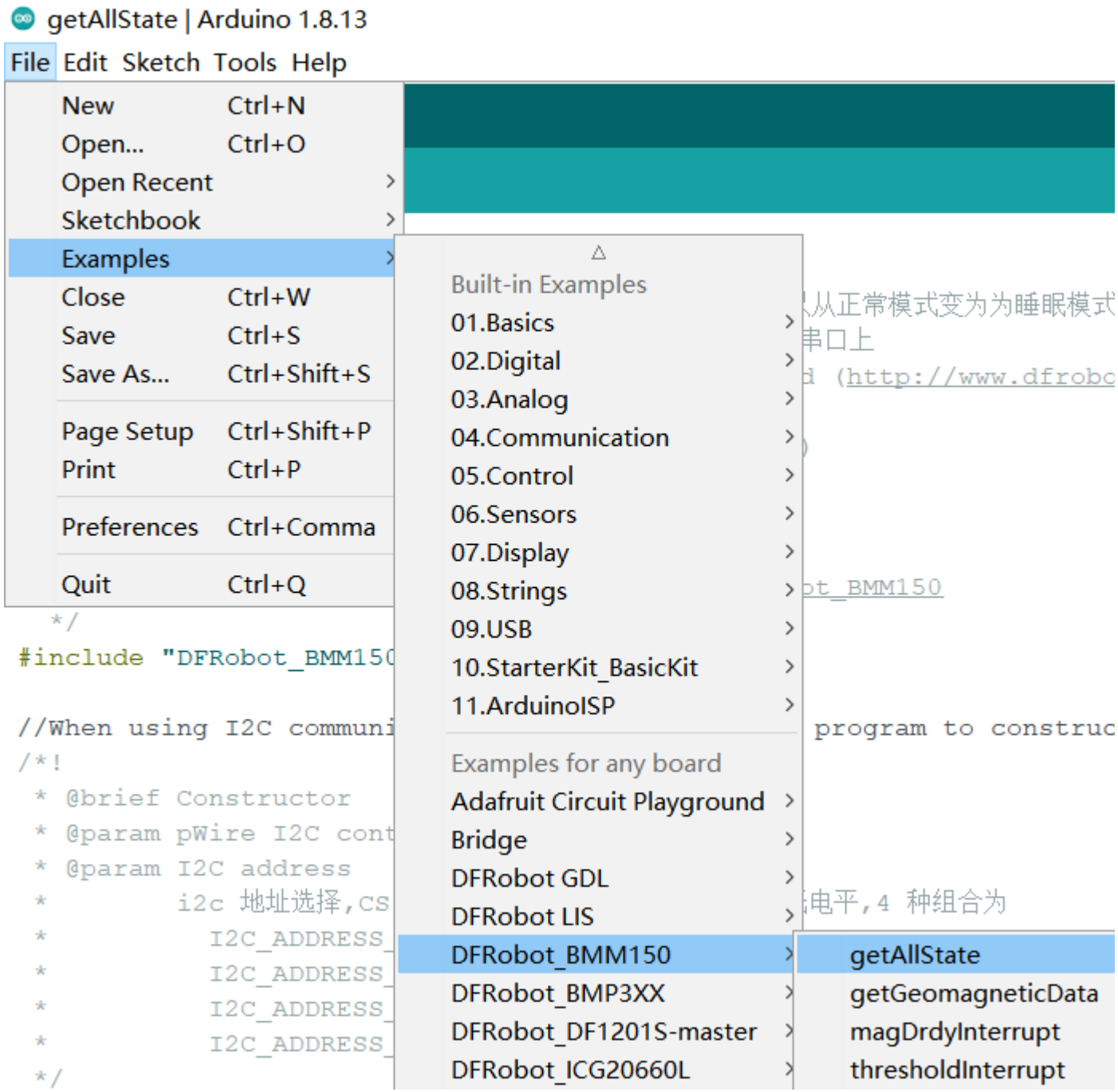
```
 * @n   MEASUREMENT_Y_DISABLE       Disable the measurement at y-axis
 * @param channelZ
 * @n   MEASUREMENT_Z_ENABLE        Enable the measurement at z-axis
 * @n   MEASUREMENT_Z_DISABLE       Disable the measurement at z-axis
 */
void setMeasurementXYZ(uint8_t channelX = MEASUREMENT_X_ENABLE, uint8_t channelY = MEASUREM

/**
 * @fn getMeasurementStateXYZ
 * @brief Get the enabling status at x-axis, y-axis and z-axis
 * @return result Return enabling status as a character string
 */
String getMeasurementStateXYZ(void);

/**
 * @fn setThresholdInterrupt(uint8_t modes, int8_t threshold, uint8_t polarity)
 * @brief Set threshold interrupt, an interrupt is triggered when the geomagnetic value of
 * @n      High polarity: active on high level, the default is low level, which turns to hi
 * @n      Low polarity: active on low level, the default is high level, which turns to low
 * @param modes
 * @n     LOW_THRESHOLD_INTERRUPT       Low threshold interrupt mode
 * @n     HIGH_THRESHOLD_INTERRUPT      High threshold interrupt mode
 * @param  threshold
 * @n     Threshold, default to expand 16 times, for example: under low threshold mode, if
 * @param polarity
 * @n     POLARITY_HIGH      High polarity
 * @n     POLARITY_LOW       Low polarity
 */
void setThresholdInterrupt(uint8_t modes, int8_t threshold, uint8_t polarity);

/**
 * @fn setThresholdInterrupt(uint8_t modes, uint8_t channelX, uint8_t channelY, uint8_t cha
 * @brief Set threshold interrupt, an interrupt is triggered when the geomagnetic value of
 * @n   When an interrupt occurs, INT pin level will jump
 * @n   High polarity: active on high level, the default is low level, which turns to high
 * @n   Low polarity: active on low level, the default is high level, which turns to low le
```

```
 * @param modes
 * @n   LOW_THRESHOLD_INTERRUPT   Low threshold interrupt mode
 * @n   HIGH_THRESHOLD_INTERRUPT  High threshold interrupt mode
 * @param channelX
 * @n   INTERRUPT_X_ENABLE        Enable high threshold interrupt at x-axis
 * @n   INTERRUPT_X_DISABLE       Disable high threshold interrupt at x-axis
 * @param channelY
 * @n   INTERRUPT_Y_ENABLE         Enable high threshold interrupt at y-axis
 * @n   INTERRUPT_Y_DISABLE        Disable high threshold interrupt at y-axis
 * @param channelZ
 * @n   INTERRUPT_Z_ENABLE         Enable high threshold interrupt at z-axis
 * @n   INTERRUPT_Z_DISABLE        Disable high threshold interrupt at z-axis
 * @param  threshold
 * @n   Threshold, default to expand 16 times, for example: if the threshold is set to be 1
 * @param polarity
 * @n     POLARITY_HIGH      High polarity
 * @n     POLARITY_LOW       Low polarity
 */
void setThresholdInterrupt(uint8_t modes, uint8_t channelX, uint8_t channelY, uint8_t chann

/**
 * @fn getThresholdData
 * @brief Get the data with threshold interrupt occurred
 * @return Returns the structure for storing geomagnetic data, the structure stores the dat
 * @n The interrupt is not triggered when the data at x-axis, y-axis and z-axis are NO_DATA
 * @n String state The storage state is binary data string
 * @n uint8_t value The storage state is binary raw value, the data format are as follows:
 * @n bit0 is 1 Indicate the interrupt occur at x-axis
 * @n bit1 is 1 Indicate the interrupt occur at y-axis
 * @n bit2 is 1 Indicate the interrupt occur at z-axis
 * @n ------------------------------------
 * @n | bit7 ~ bit3 | bit2 | bit1 | bit0 |
 * @n ------------------------------------
 * @n |  reserved   |  0   |  0   |  0   |
 * @n ------------------------------------
 */
```

```
sBmm150ThresholdData_t getThresholdData(void);

/**
 * @fn selfTest
 * @brief The sensor self test, the returned value indicate the self test result.
 * @param testMode:
 * @n    BMM150_SELF_TEST_NORMAL              Normal self test, test whether x-axis, y-axi
 * @n    BMM150_SELF_TEST_ADVANCED            Advanced self test, test the data accuracy a
 * @return result The returned character string is the self test result
 */
String selfTest(uint8_t testMode);
```

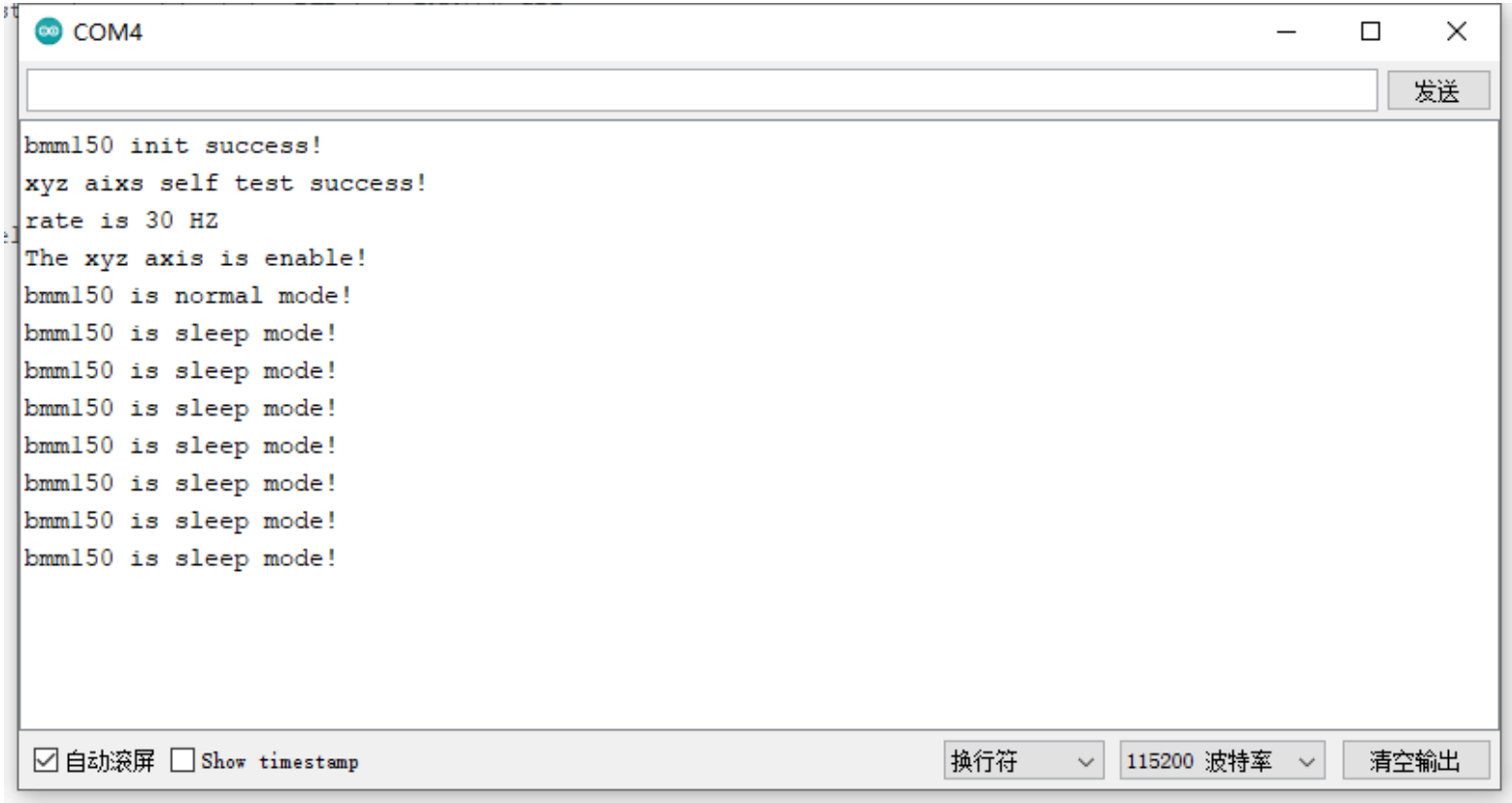
Sample Code 1 - get configuration status(getAllState.ino)

- Select getAllState.ino



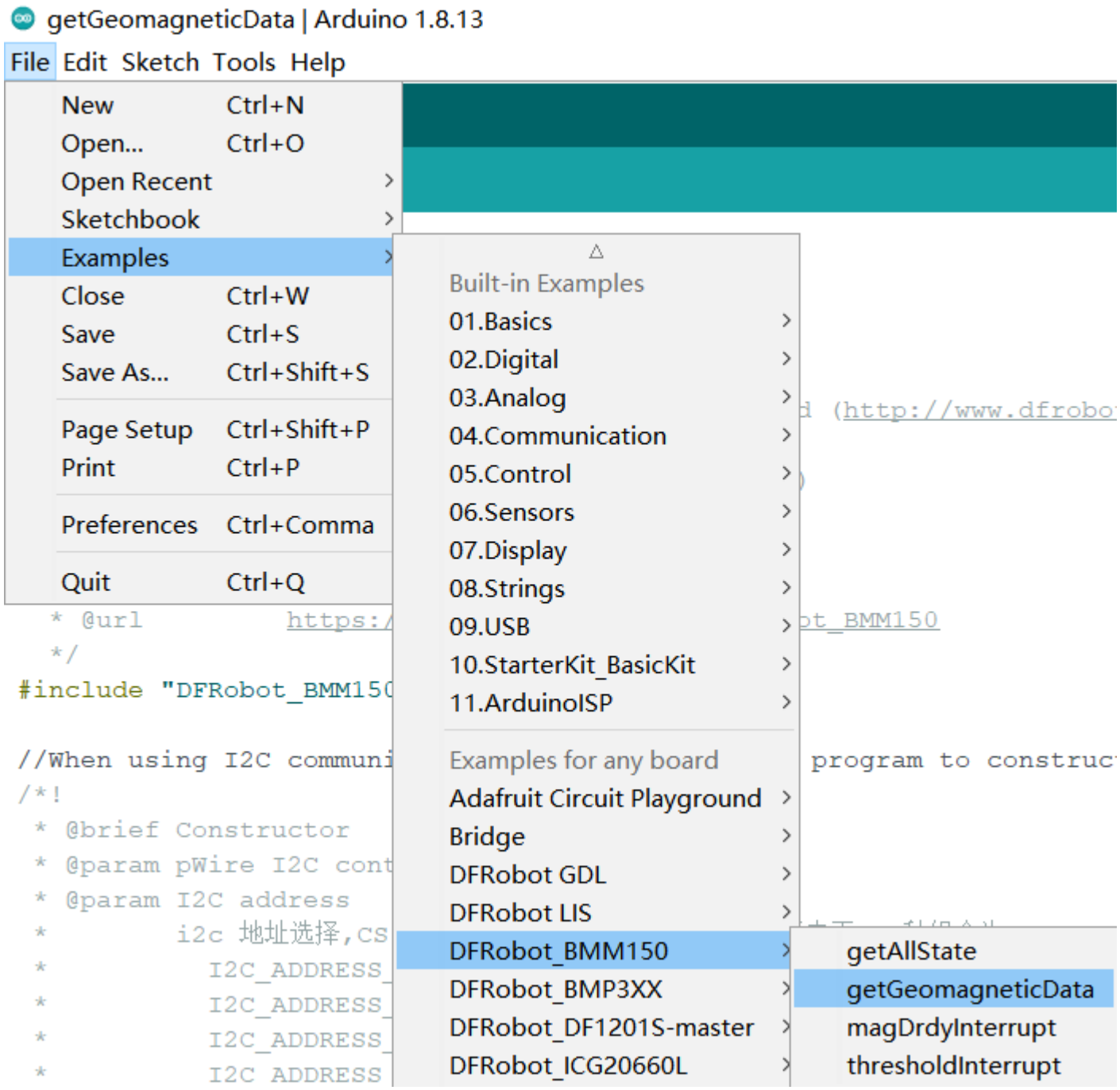
- Buring codes into your board

Result



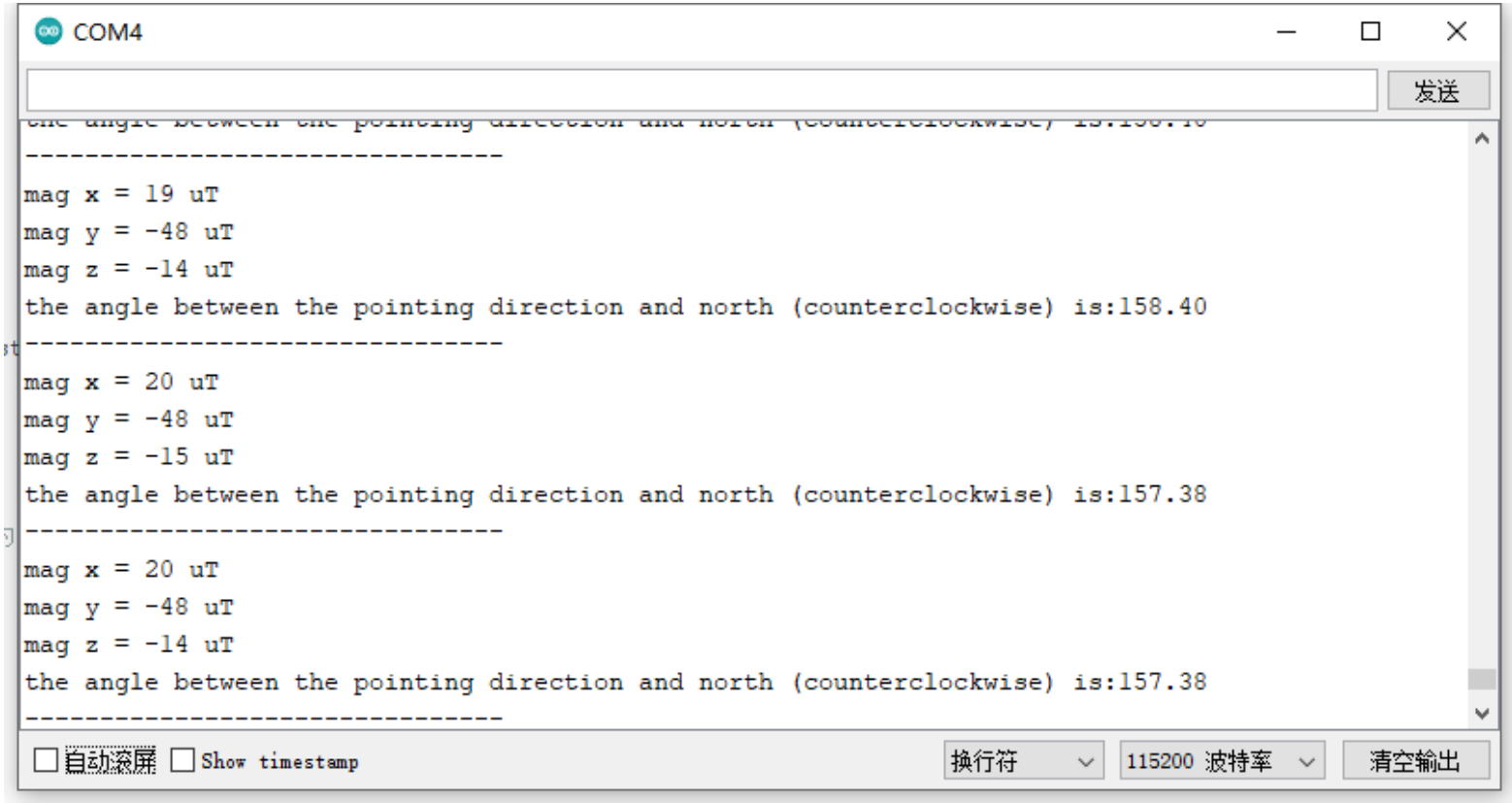
Sample Code 2 - get geomagnetic data, compass angle(getGeomagneticData.ino)

- Select getGeomagneticData.ino



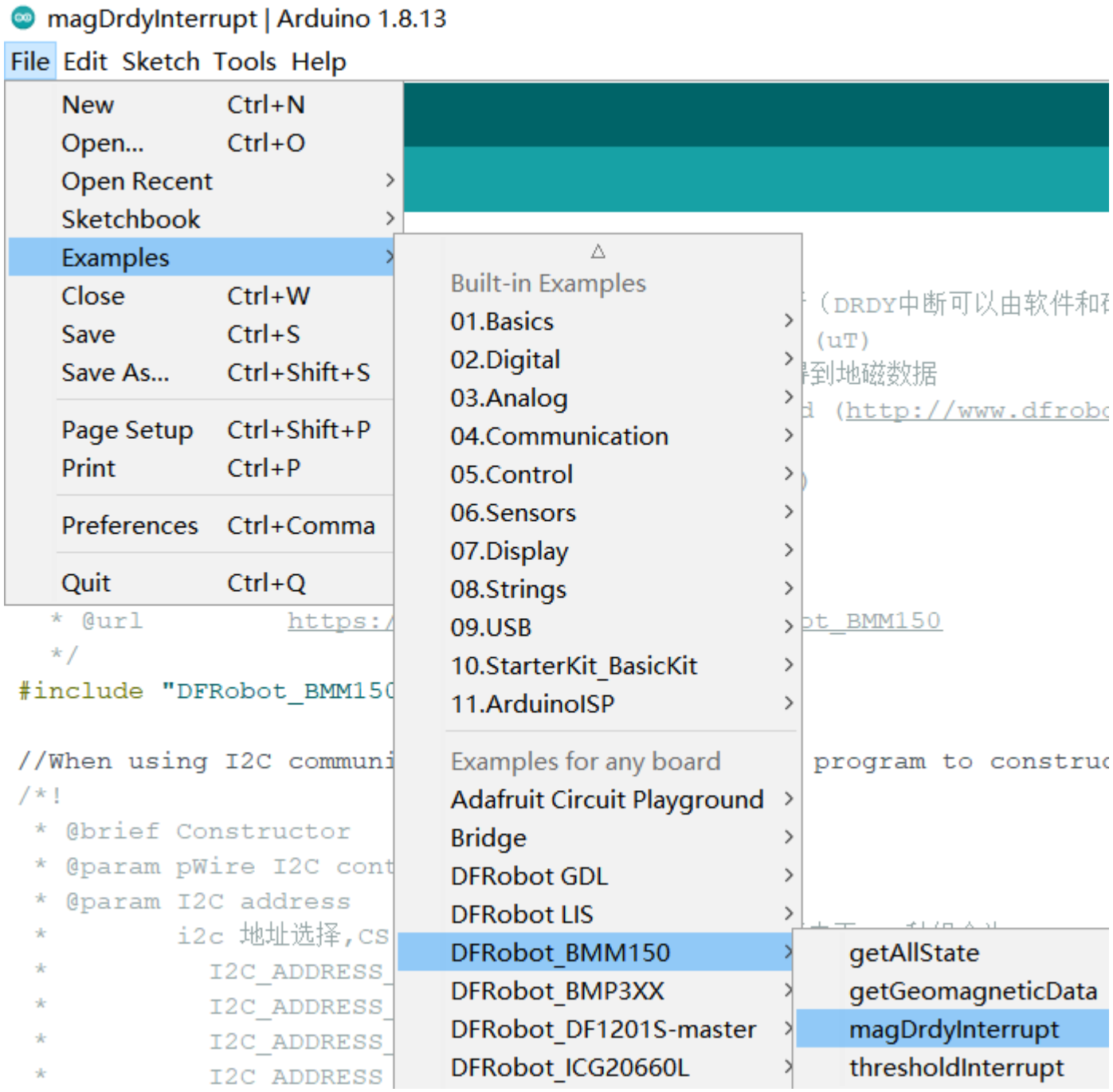
- Burning codes into your board

Result



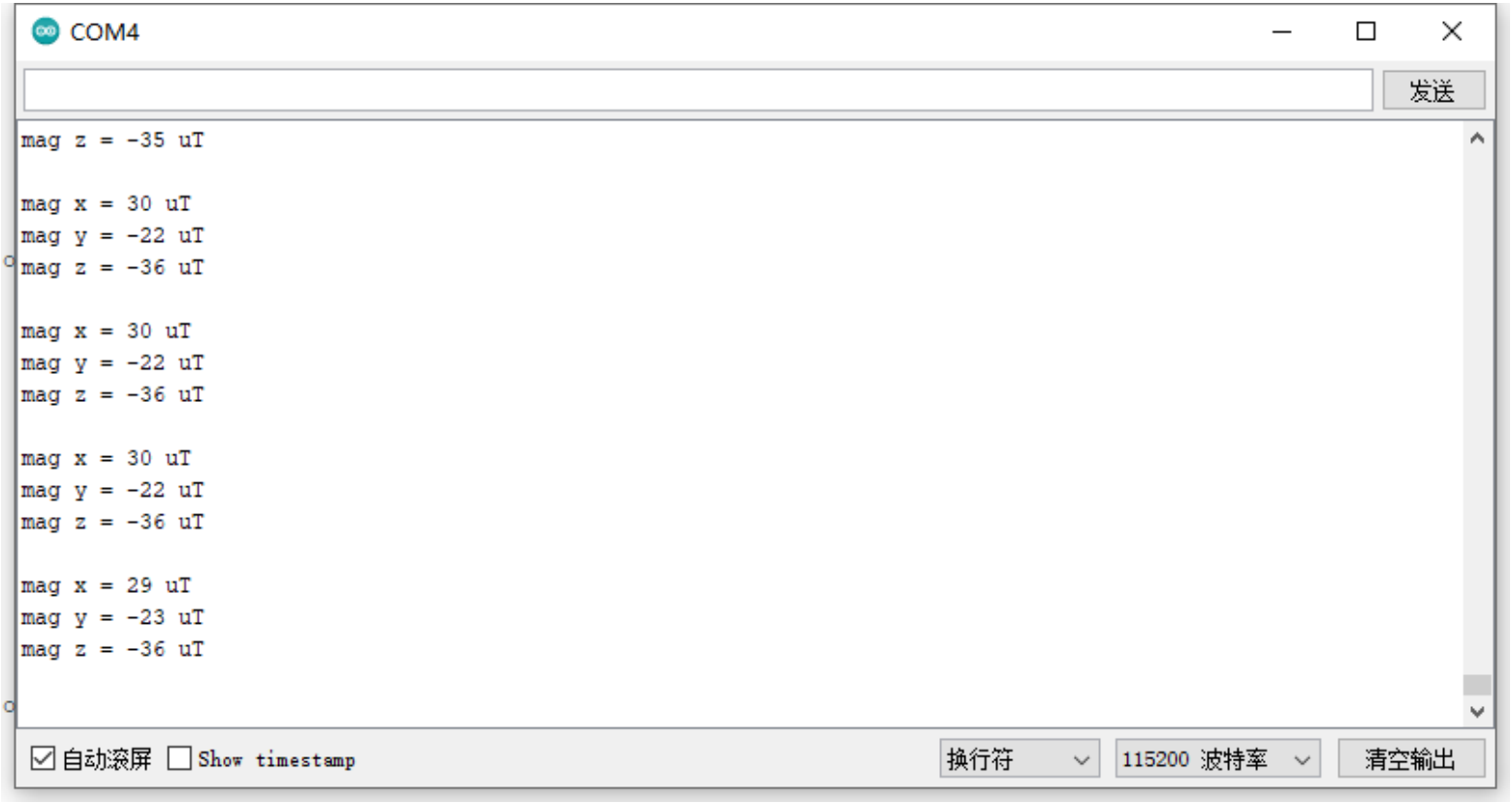
Sample Code 3 - prepare interrupt function(magDrdyInterrupt.ino)

- Select magDrdyInterrupt.ino



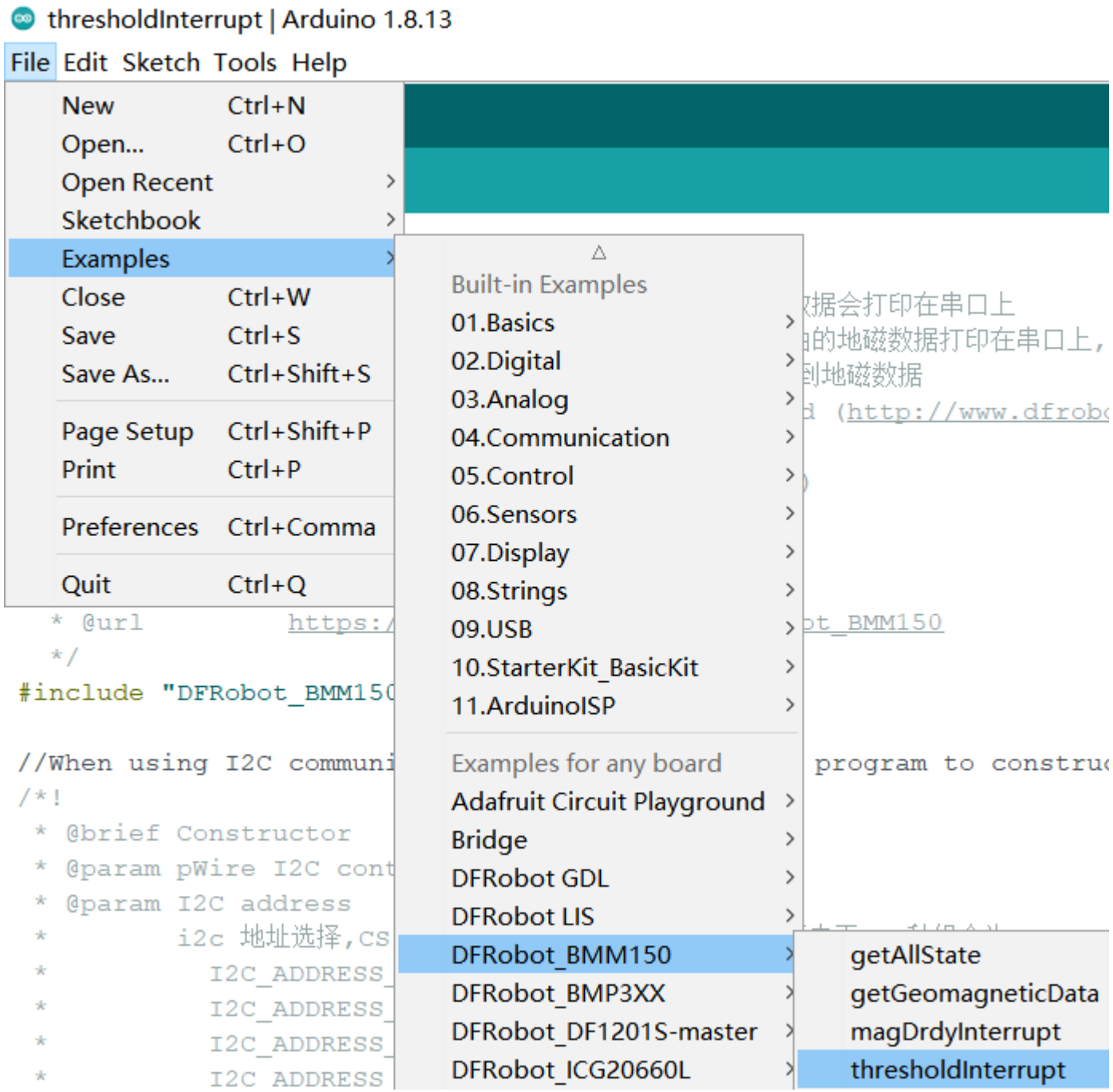
- Burning codes into your board

Result



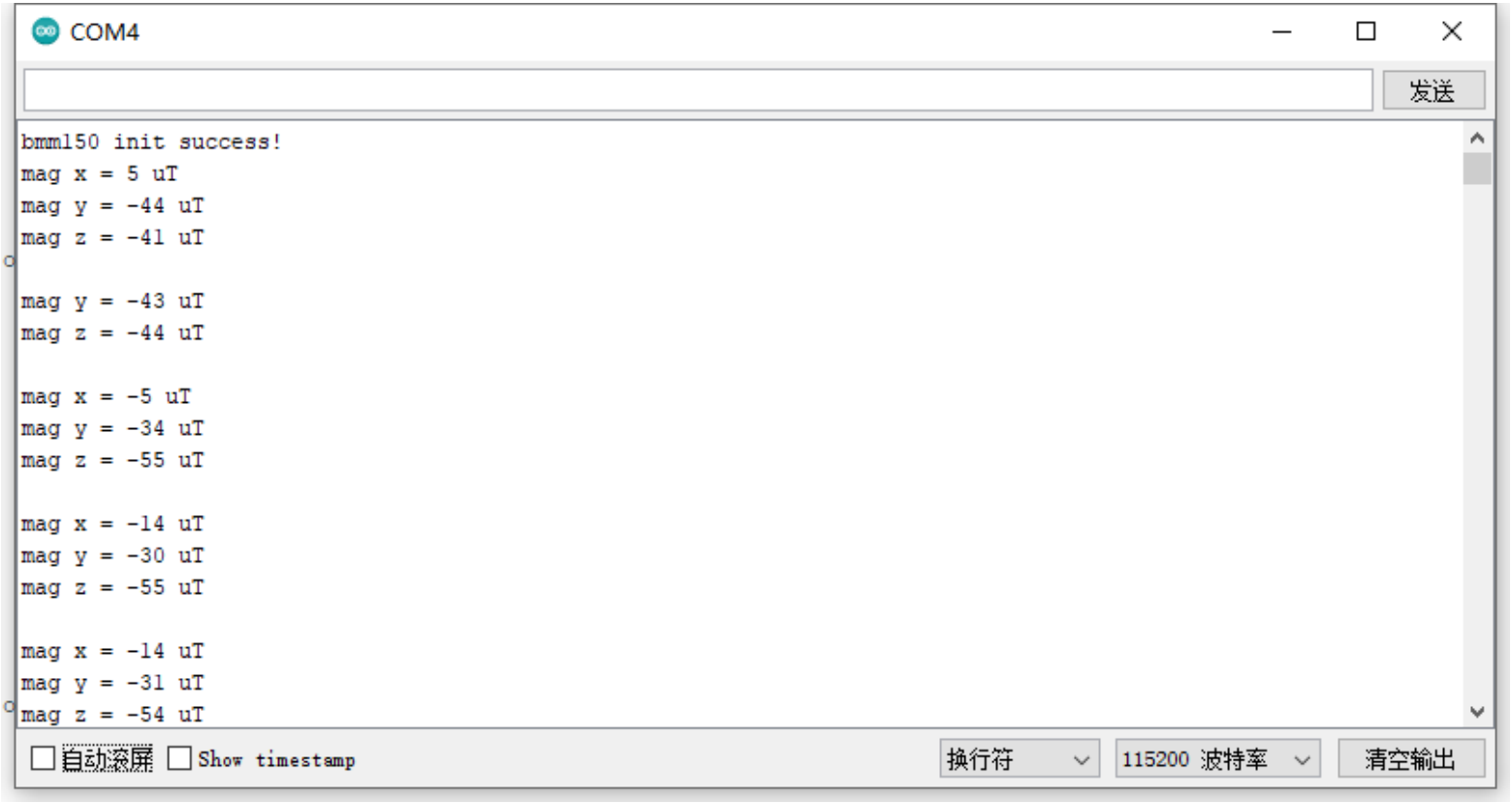
Sample Code 4 - threshold interrupt function(thresholdInterrupt.ino)

- Select thresholdInterrupt.ino



- Burning codes into your board

Result



Tutorial for Using on Raspberry Pi

Requirements

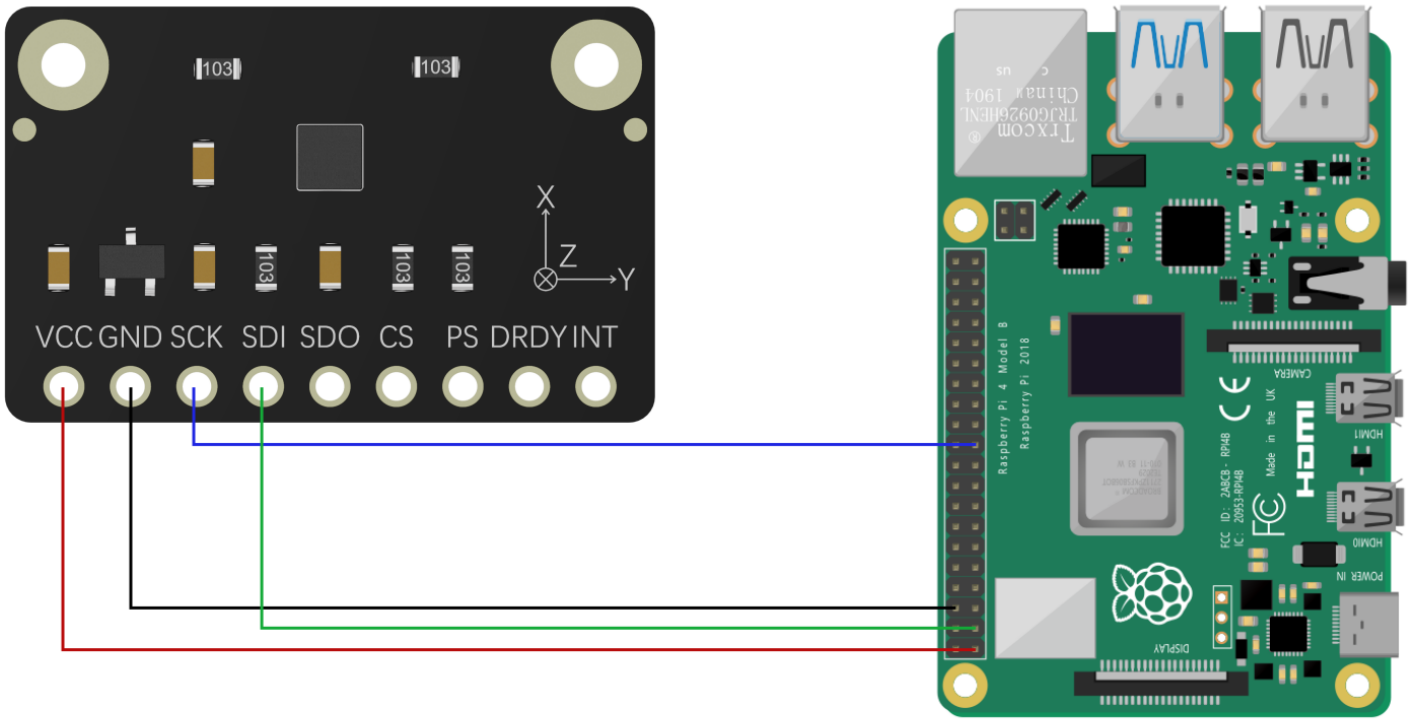
- **Hardware**
  - Raspberry Pi 4 Model B-2GB (https://www.dfrobot.com/product-1876.html) (or similar) x 1

- BMM150 Triple Axis Magnetometer Sensor x1
- Dupont Wires

- **Software**
  - BMM150 Sensor Python Library
  - RASPBIAN Official OS (https://www.raspberrypi.org/downloads/raspbian)

Connection Diagram

- Connect the module with Raspberry Pi as shown below. Default address: 0x13



Installing Driver

1. Enable Raspberry Pi I2C(skip this step if it is already enabled). Open the terminal and enter the following command.

```
sudo raspi-config
```

Use the up/down keys to select "5 Interface Options". Press **enter**, select "P5 I2C". Press **enter** to comfirm "YES". Restart your Raspberry Pi.

2. Install Python dependency lirary and git(skip this step if already installed). Raspberry Pi needs to be connected with internet. Open the terminal and enter the following commands in order.

```
sudo apt-get update
```

```
sudo apt-get install build-essential python-dev python-smbus git
```

3. Download BMM150 library. Open the terminal and enter the following commands in order.

```
cd Desktop
```

```
git clone https://github.com/DFRobot/DFRobot_BMM150
```

Sample Code

- Sample Code 1 - get configuration status(get_all_state.py)

- Sample Code 2 - get geomagnetic data, compass angle(get_geomagnetic_data.py)
- Sample Code 3 - prepare interrupt function(data_ready_interrupt.py)
- Sample Code 4 - threshold interrupt function(threshold_interrupt.py)

Sample Code 1 - get configuration status(get_all_state.py)

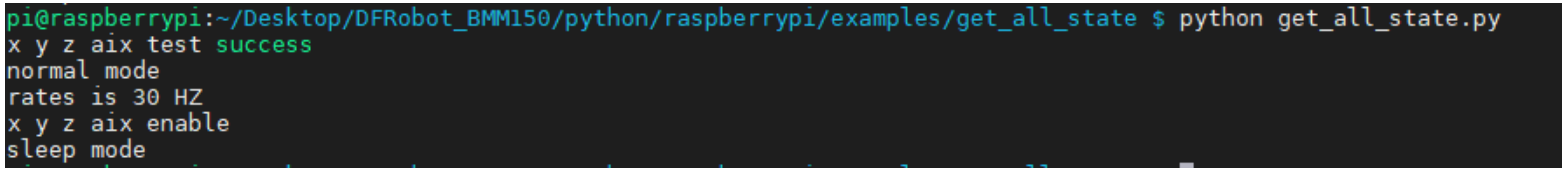
- Open the terminal and enter the following commands, and run the sample code.

```
cd DFRobot_BMM150/python/raspberrypi/examples
```

```
cd get_all_state
```

```
python get_all_state.py
```

Result



Sample Code 2 - get geomagnetic data, compass angle(get_geomagnetic_data.py)

- Open the terminal and enter the following commands, and run the sample code.

```
cd DFRobot_BMM150/python/raspberrypi/examples
```

```
cd get_geomagnetic_data
```

```
python get_geomagnetic_data.py
```

Result

```
pi@raspberrypi:~/Desktop/DFRobot_BMM150/python/raspberrypi/examples/get_geomagnetic_data $ python get_geomagnetic_data.py
mag x = 22 ut
mag y = 5 ut
mag z = -69 ut
the angle between the pointing direction and north (counterclockwise) is: 77.20

mag x = 23 ut
mag y = 5 ut
mag z = -69 ut
the angle between the pointing direction and north (counterclockwise) is: 77.74

mag x = 22 ut
mag y = 6 ut
mag z = -70 ut
the angle between the pointing direction and north (counterclockwise) is: 74.74

mag x = 22 ut
mag y = 5 ut
mag z = -69 ut
the angle between the pointing direction and north (counterclockwise) is: 77.20

mag x = 22 ut
mag y = 5 ut
mag z = -69 ut
the angle between the pointing direction and north (counterclockwise) is: 77.20

mag x = 22 ut
mag y = 5 ut
mag z = -69 ut
the angle between the pointing direction and north (counterclockwise) is: 77.20

mag x = 23 ut
mag y = 6 ut
mag z = -69 ut
the angle between the pointing direction and north (counterclockwise) is: 75.38
```

Sample Code 3 - prepare interrupt function(data_ready_interrupt.py)

- Open the terminal and enter the following commands, and run the sample code.

```
cd DFRobot_BMM150/python/raspberrypi/examples
```

```
cd data_ready_interrupt
```

```
python data_ready_interrupt.py.py
```

Result

```
pi@raspberrypi:~/Desktop/DFRobot_BMM150/python/raspberrypi/examples/data_ready_interrupt $ python data_ready_interrupt.py
please check connect!
mag x = 36 ut
mag y = -17 ut
mag z = 15 ut

mag x = 36 ut
mag y = -17 ut
mag z = 15 ut

mag x = 36 ut
mag y = -17 ut
mag z = 15 ut

mag x = 36 ut
mag y = -16 ut
mag z = 15 ut

mag x = 36 ut
mag y = -17 ut
mag z = 15 ut

mag x = 36 ut
mag y = -17 ut
mag z = 15 ut

mag x = 36 ut
mag y = -17 ut
mag z = 15 ut

mag x = 36 ut
mag y = -17 ut
mag z = 15 ut
```

Sample Code 4 - threshold interrupt function(threshold_interrupt.py)

- Open the terminal and enter the following commands, and run the sample code.

```
cd DFRobot_BMM150/python/raspberrypi/examples
```

```
cd threshold_interrupt
```

```
python threshold_interrupt.py
```

Result

```
pi@raspberrypi:~/Desktop/DFRobot_BMM150/python/raspberrypi/examples/threshold_interrupt $ python threshold_interrupt.py
mag x = -8 ut
mag y = -24 ut
mag z = -65 ut

mag x = -48 ut
mag z = -28 ut

mag x = -54 ut
mag z = -20 ut

mag y = -5 ut
mag z = -63 ut

mag x = 31 ut
mag z = -64 ut

mag x = -47 ut
mag z = -11 ut

mag x = -30 ut
mag z = -10 ut

mag x = -23 ut

mag z = -68 ut

mag x = -22 ut
mag y = -1 ut
mag z = -67 ut

mag x = -47 ut

mag x = -32 ut
mag z = -44 ut
```

FAQ

For any questions, advice or cool ideas to share, please visit the **DFRobot Forum** (https://www.dfrobot.com/forum/).

More Documents

- Schematics.pdf (https://dfimg.dfrobot.com/nobody/wiki/fb1720f4b072ed489252ad9d2ce65da8.pdf)
- 3D Model File (https://dfimg.dfrobot.com/nobody/wiki/1d39fb0cd510f588a153552959a337e0.rar)
- CAD File (https://dfimg.dfrobot.com/nobody/wiki/ed9831f1c414bdbfeadfc6be55c3be77.rar)

- BMM150 Datasheet.pdf (https://dfimg.dfrobot.com/nobody/wiki/3c1a1469e7052a671c45bc7adc22cc60.pdf)
- Download all file in one click (https://dfimg.dfrobot.com/nobody/wiki/f79ed810465e854d6c7fdba82a93e1ea.rar)

Get **Fermion: BMM150 Triple Axis Magnetometer Sensor(Breakout)** (https://www.dfrobot.com/product-2507.html) from DFRobot Store or **DFRobot Distributor**. (https://www.dfrobot.com/distributor)

Turn to the Top