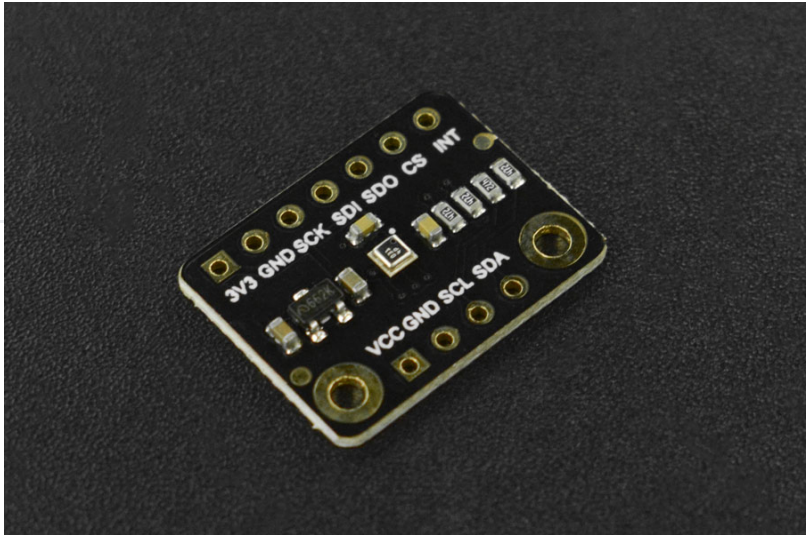
SKU:SEN0423 (https://www.dfrobot.com/product-2432.html)

(https://www.dfrobot.com/product-2432.html)



Introduction

BMP390L is an industrial-grade dedicated digital barometric pressure sensor newly developed by Bosch. It has extremely high temperature stability, low drift, low power consumption (sleep current is only 54μA, the highest power consumption in measurement mode is 650μA) and low noise impact

(0.02Pa). The maximum deviation of the absolute accuracy of this product is less than 50Pa, and the relative accuracy can reach ±3Pa (equivalent to 25cm), which is 166.7% higher than the previous generation. In addition, with a 12-month long-term stability deviation of only 16Pa, the new model's long-term stability performance far exceeds that of the previous generation.

Accurate, stable, low-power height detection makes this product very suitable for indoor positioning applications that cannot be covered by GPS signals. Since it offers extended availability up to 10 years, you can use this product with peace of mind.

Features

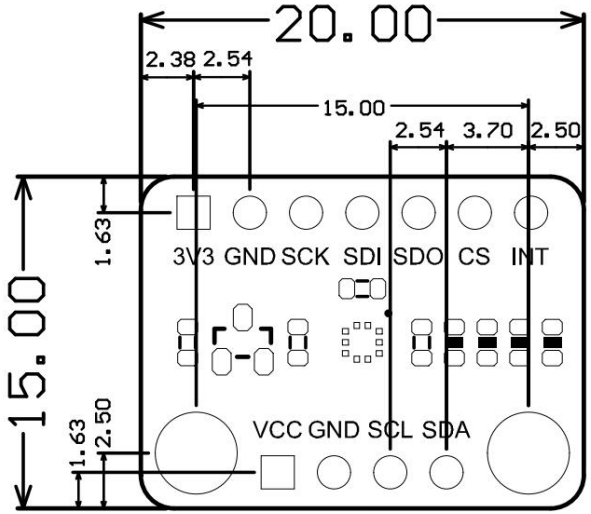
- High precision (absolute deviation <50Pa, relative accuracy <3Pa)
- High long-term stability (±16Pa/year)
- Extremely high temperature stability and low drift characteristics
- Low power consumption (sleep current: 54uA, working current <650uA)

Application

- Asset tracking & navigation(floor detection)
- Water-level measurement

- Water level measurement
- Vertical velocity indication(e.g. rise/sink speed)
- Enhancement of GPS navigation
- Outdoor navigation & applications
- Health care applications(e.g. spirometry)
- AR & VR applications
- Context awareness
- Flight stabilization
- Clogging detection

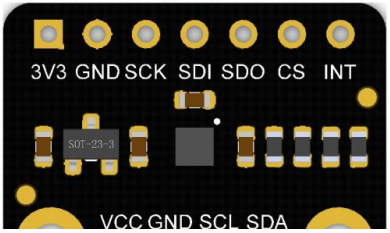
Specification



- Working Voltage: 3.3 V
- Working Current: sleep mode (54 μA) / normal mode (357~650 μA)
- Interface Mode: I2C/SPI
- I2C Address: 0x77 (default address)/0x76 (optional: SDO pin connects to GND)

- I2C Address: 0x77 (default address)/0x76 (optional: SDO pin connects to GND)
- Pressure Sensing Range: 300~1250 hPa
- Relative Accuracy: ±0.03hPa (equivalent to 25cm) (@700~1100hPa, 25 to 40 °C)
- Absolute Accuracy: ±0.50hPa
- Temperature Deviation Coefficient: ±0.6Pa/K (@900hPa, 25~40 °C)
- Long-term Stability: ±0.16hPa/year
- ODR Accuracy: ±2~±12%
- Start-up Time: 2 ms
- Operating Temperature Range: -40 °C ~ +85 °C (full precision measurement can be used under 0 °C ~ +65 °C)
- Module Size: 15×20 (mm) / 0.59×0.79(inch)

Board Overview



TOP

BOTTOM

Num	Label	Description
1	VCC	3.3v~5.5v power input
2	3V3	3.3v power output/input
3	GND	Power negative
4	SCL	I2C clock line
5	SDA	I2C data line
6	SCK	I2C data line
7	SDI	SPI data line (input)
8	SDO	SPI data line (output)
9	CS	SPI chip select line

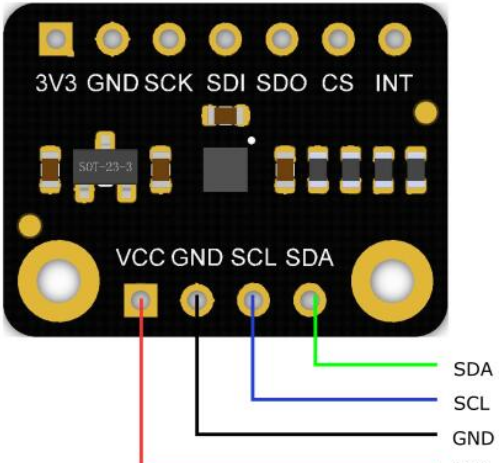
Num	Label	Description
10	INT	Interrupt pin

Notice:

- All data output voltage is 3.3V
- Pull the SDO pin low to switch the I2C address: 0x77

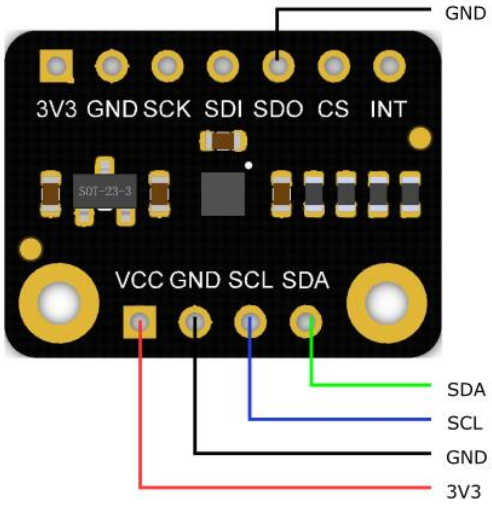
Connection tips for different communication methods:

- **I2C: 0x77 (default)**

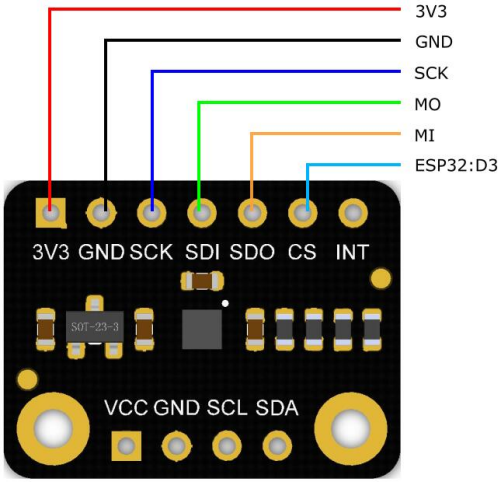



3V3

- I2C：0x76


GND
3V3 GND SCK SDI SDO CS INT
VCC GND SCL SDA
SDA
SCL
GND
3V3

- SPI



- **cs pin can choose the pin that does not conflict**
- **Interrupt pin connection**

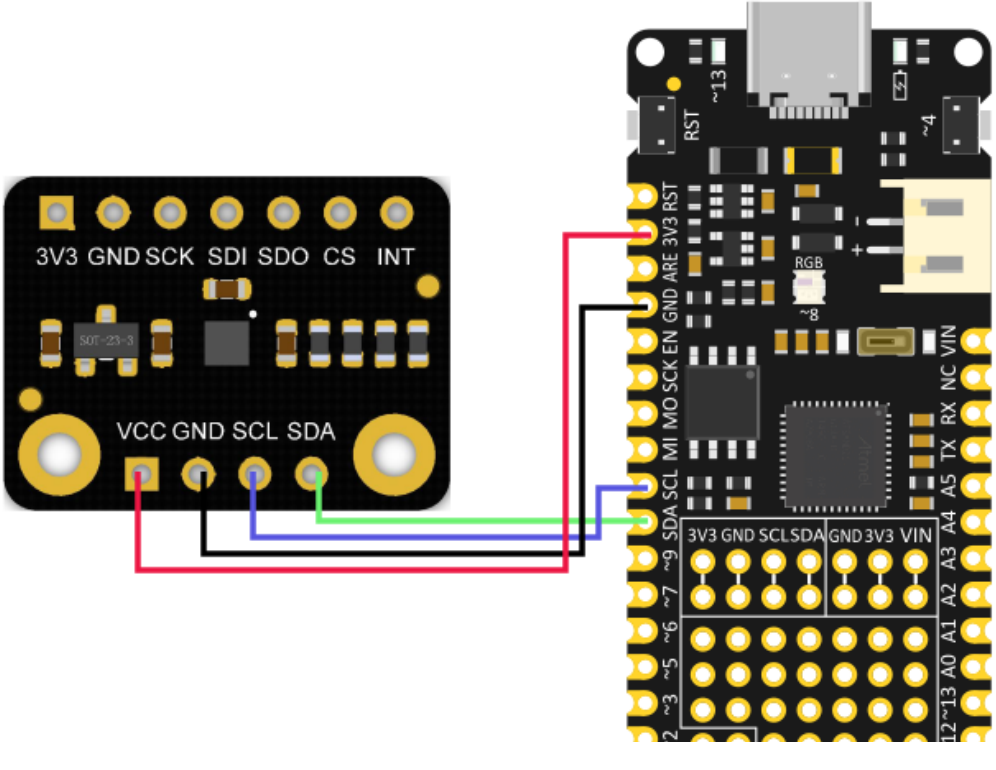
Motherboard	Default connection pin
UNO/MEGA2560	D2

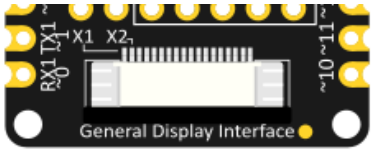
Motherboard	Default connection pin
UNO/MEGA2560	D2
Leonardo	D3
Micro:bit	P0

ESP32/ESP8266/ARDUINO_SAM_ZERO (M0)	ESP32/ESP8266/ARDUINO_SAM_ZERO (M0)
ESP32/ESP8266/ARDUINO_SAM_ZERO (M0)	GPIO25

Tutorial for M0

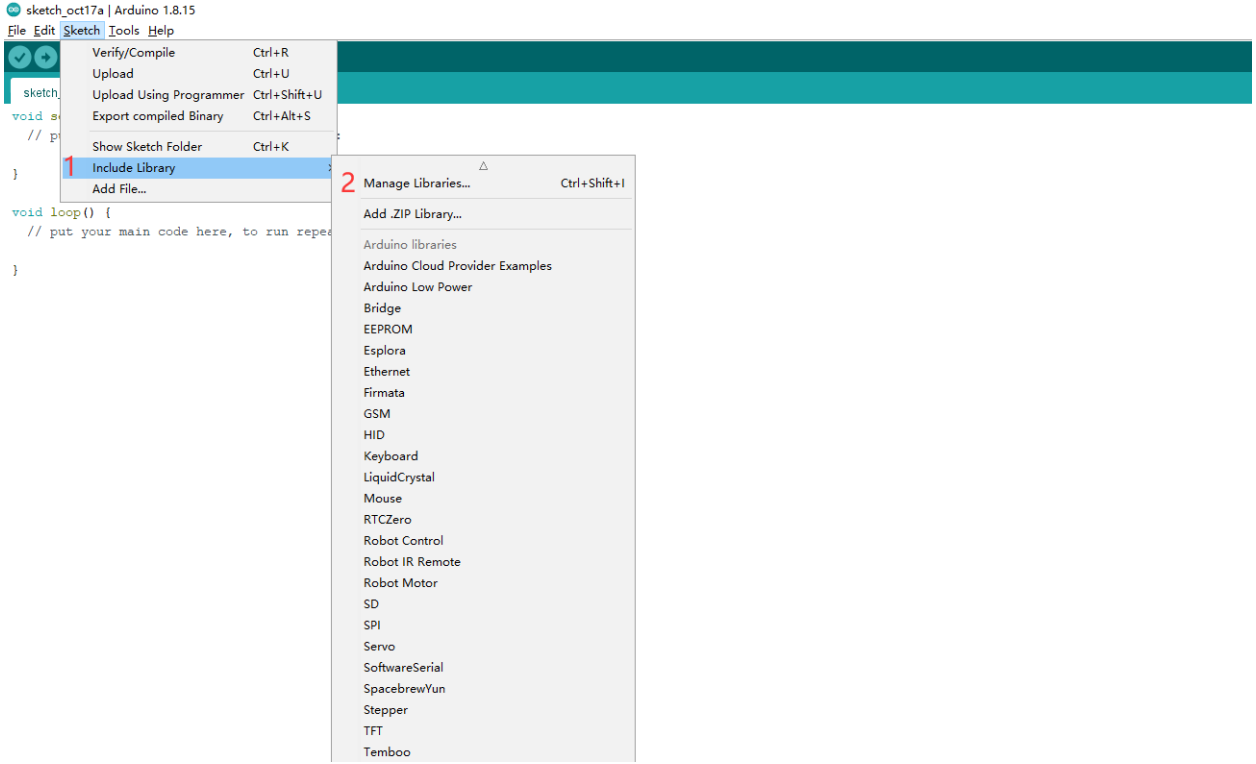
Please connect the sensor to M0 (or other motherboard) as shown in the wiring diagram.

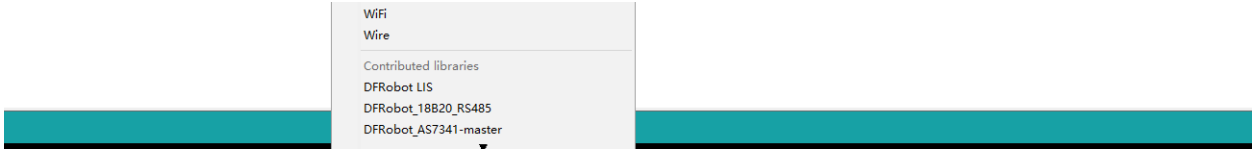

3V3 GND SCK SDI SDO CS INT
VCC GND SCL SDA
RST
~13
~4
RGB
~8
3V3 RST
ARE 3V3
GND EN
SCK NC
MI RX
MO TX
SCL A5
SDA A4
~9 A3
~7 A2
~6 A1
~5 A0
~3 ~13
2 12
3V3 GND SCL SDA
GND 3V3 VIN

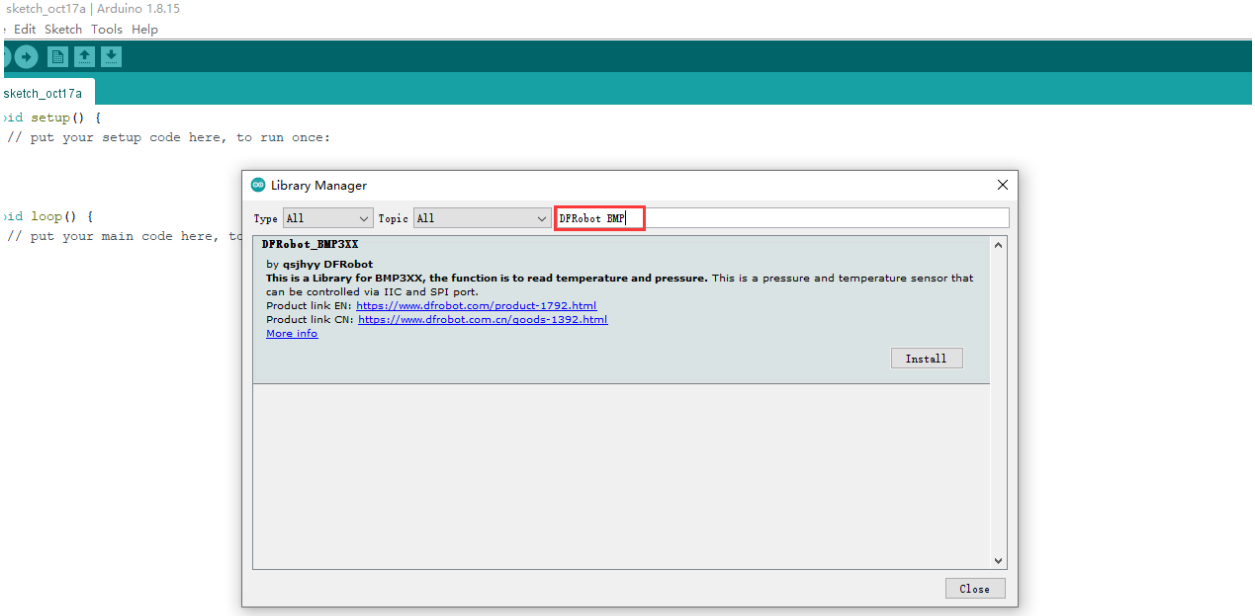


Requirements

- **Hardware**
  - Firebeetle Board-M0 x 1
  - BMP390L digital barometric pressure sensor × 1
  - Jumper wires
- **Software**
  - Arduino IDE (https://www.arduino.cc/en/Main/Software)
  - Download and install the **Library files and sample programs** (https://github.com/DFRobot/DFRobot_BMP3XX) (About how to install the library? (https://www.arduino.cc/en/Guide/Libraries#.UxU8mdzF9H0)) \
  - Or get Library files from Arduino :


sketch_oct17a | Arduino 1.8.15
File Edit Sketch Tools Help
Verify/Compile Ctrl+R
Upload Ctrl+U
Upload Using Programmer Ctrl+Shift+U
Export compiled Binary Ctrl+Alt+S
Show Sketch Folder Ctrl+K
1
Include Library
Add File...
2
Manage Libraries... Ctrl+Shift+I
Add .ZIP Library...
Arduino libraries
Arduino Cloud Provider Examples
Arduino Low Power
Bridge
EEPROM
Esplora
Ethernet
Firmata
GSM
HID
Keyboard
LiquidCrystal
Mouse
RTCZero
Robot Control
Robot IR Remote
Robot Motor
SD
SPI
Servo
SoftwareSerial
SpacebrewYun
Stepper
TFT
Temboo
void loop() {
// put your main code here, to run
}


WiFi
Wire
Contributed libraries
DFRobot LIS
DFRobot_18B20_RS485
DFRobot_AS7341-master


sketch_oct17a | Arduino 1.8.15
Edit Sketch Tools Help
sketch_oct17a
void setup() {
// put your setup code here, to run once:
void loop() {
// put your main code here, to
Library Manager
Type All
Topic All
DFRobot BMP
DFRobot_BMP3XX
by qsjhyy DFRobot
This is a Library for BMP3XX, the function is to read temperature and pressure. This is a pressure and temperature sensor that can be controlled via IIC and SPI port.
Product link EN: https://www.dfrobot.com/product-1792.html
Product link CN: https://www.dfrobot.com.cn/goods-1392.html
More info
Install
Close

- **Sample Code**

    - Sample code 1-Read the measured frequency and data (temperature, air pressure, altitude) (getTempPress.ino)
    - Sample code 2-data interrupt function (interruptDataDrdy.ino)
    - Sample code 3-FIFO water level or full interrupt function (interruptUsingFIFO.ino)
    - Sample code 4-data processing advanced setting function (set_ODR_OSR_IIR.ino)

⚠ **Notice: The tutorial example uses an altitude of 540 meters in Wenjiang District, Chengdu (China). Please change to the local altitude calibration when actually using it.**

- **List of main API interface functions**

```
/**
* @brief Initialize function
* @return Return 0 to indicate initialization succeed, return other values indicate fail
*/
virtual int begin(void);

/**
* @brief commonly used sampling modes that allows users to configure easily
* @param mode:
*        eUltraLowPrecision，Ultra-low precision, suitable for monitoring weather (lowes
*        eLowPrecision，Low precision, suitable for random detection, power is normal mo
*        eNormalPrecision1，Normal precision 1, suitable for dynamic detection on handhe
*        eNormalPrecision2，Normal precision 2, suitable for drones, power is normal mod
*        eHighPrecision，High precision, suitable for low-power handled devices （e.g mo
*        eUltraPrecision，Ultra-high precision, suitable for indoor navigation, its acqu
* @return Return True indicate configuration succeed, False indicate failed and remain i
*/
bool setSamplingMode(ePrecisionMode_t mode);
```

```
bool setSamplingMode(ePrecisionMode_t mode);

/**
* @brief Get the sampling period in the current mode
* @return Return sampling period, unit: us

*/
uint32_t getSamplingPeriodUS(void);

/**
* @brief Get temperature measurement value from register, working range (-40 – +85 °C)
* @return Return temperature measurements, unit: °C
*/
float readTempC(void);

/**
* @brief Get pressure measurement value from register, working range（300–1250 hPa）
* @brief If the reference value is provided before, the absolute value of the current po
* @n     atmospheric pressure
* @return Return pressure measurements, unit: Pa
*/
float readPressPa(void);

/**
* @brief Take the given current location altitude as the reference value to eliminate th
```

```
* @param altitude Altitude in current position
* @return If pass in the reference value successfully, return turn. If failed, return fa
*/
bool calibratedAbsoluteDifference(float altitude);


/**
* @brief Calculate the altitude based on the atmospheric pressure measured by the sensor
* @brief If the reference value is provided before, the absolute value of the current se
* @return Return altitude, unit: m
*/
float readAltitudeM(void);

/**
* @brief Get the cached data in the FIFO
* @brief Temperature unit: °C; Pressure unit: Pa
*/
void getFIFOData(float &FIFOTemperatureC, float &FIFOPressurePa);

/**
* @brief FIFO empty command and soft reset command of sensor
* @param mode Basic sensor commands, three types of commands:
*        BMP3XX_CMD_NOP, Null command
*        BMP3XX_CMD_FIFO_FLUSH, Clear all data in the FIFO without changing its settings
*        BMP3XX_CMD_SOFTRESET, Trigger a reset, all user configuration settings will be
*/
```

```
*/
void setCommand(uint8_t mode);

/**
* @brief FIFO water level settings configuration
* @param WTMSetting FIFO water level (0-511) needs to be set. That FIFO fills up to the
*/
void setFIFOWTM(uint16_t WTMSetting);

/**
* @brief FIFO configuration 1 (FIFO1)
* @param mode The FIFO mode needs to set, the following modes add up to mode:
*        eFIFODIS：  Disable FIFO , eFIFOEN：  Enable FIFO
*        eFIFOStopOnFullDIS：  Continue writing when full , eFIFOStopOnFullEN：Stop writi
*        eFIFOTimeDIS：  Disable , eFIFOTimeEN：  Enable return to the sensor time frame a
*        eFIFOPressDIS：  Disable pressure data storage , eFIFOPressEN：Enable pressure d
*        eFIFOTempDIS：  Disable temperature data storage, eFIFOTempEN：Enable temperatur
*/
void setFIFOMode1(uint8_t mode);

/**
* @brief FIFO Configuration 2(FIFO2)
* @param mode The FIFO mode needs to set, the following modes add up to mode:
*        8 FIFO sampling options for pressure and temperature data (1-128), the coeffici
```

```
*          eFIFOSubsampling0, eFIFOSubsampling1, eFIFOSubsampling2, eFIFOSubsampling3,
*          eFIFOSubsampling4, eFIFOSubsampling5, eFIFOSubsampling6, eFIFOSubsampling7,
*        eFIFODataSelectDIS:  Unfiltered data (compensated or uncompensated) , eFIFOData
*                             the same as "unfilt"
*/
void setFIFOMode2(uint8_t mode);

/**
* @brief Interrupt configuration(INT)
* @param mode The interrupt mode needs to set. The following modes add up to mode:
*        Interrupt pin output mode: eINTPinPP:  Push pull, eINTPinOD: Open drain
*        Interrupt pin active level: eINTPinActiveLevelLow:  Active low , eINTPinActiveL
*        Register interrupt latch: eINTLatchDIS:  Disable, eINTLatchEN: Enable
*        FIFO water level reached interrupt: eINTFWTMDIS:  Disable, eINTFWTMEN: Enable
*        FIFO full interrupt: eINTFFullDIS:  Disable, eINTFFullEN:  Enable
*        Interrupt pin initial (invalid, non-interrupt) level: eINTInitialLevelLOW: low
*        Interrupt pin initial (invalid, non-interrupt) level:  eINTDataDrdyDIS:  Disabl
*/
void setINTMode(uint8_t mode);

/**
* @brief Configure measurement mode and power mode
* @param mode The measurement mode and power mode that need to set. The following modes
*        ePressDIS: Disable pressure measurement , ePressEN:  Enable pressure measuremen
```

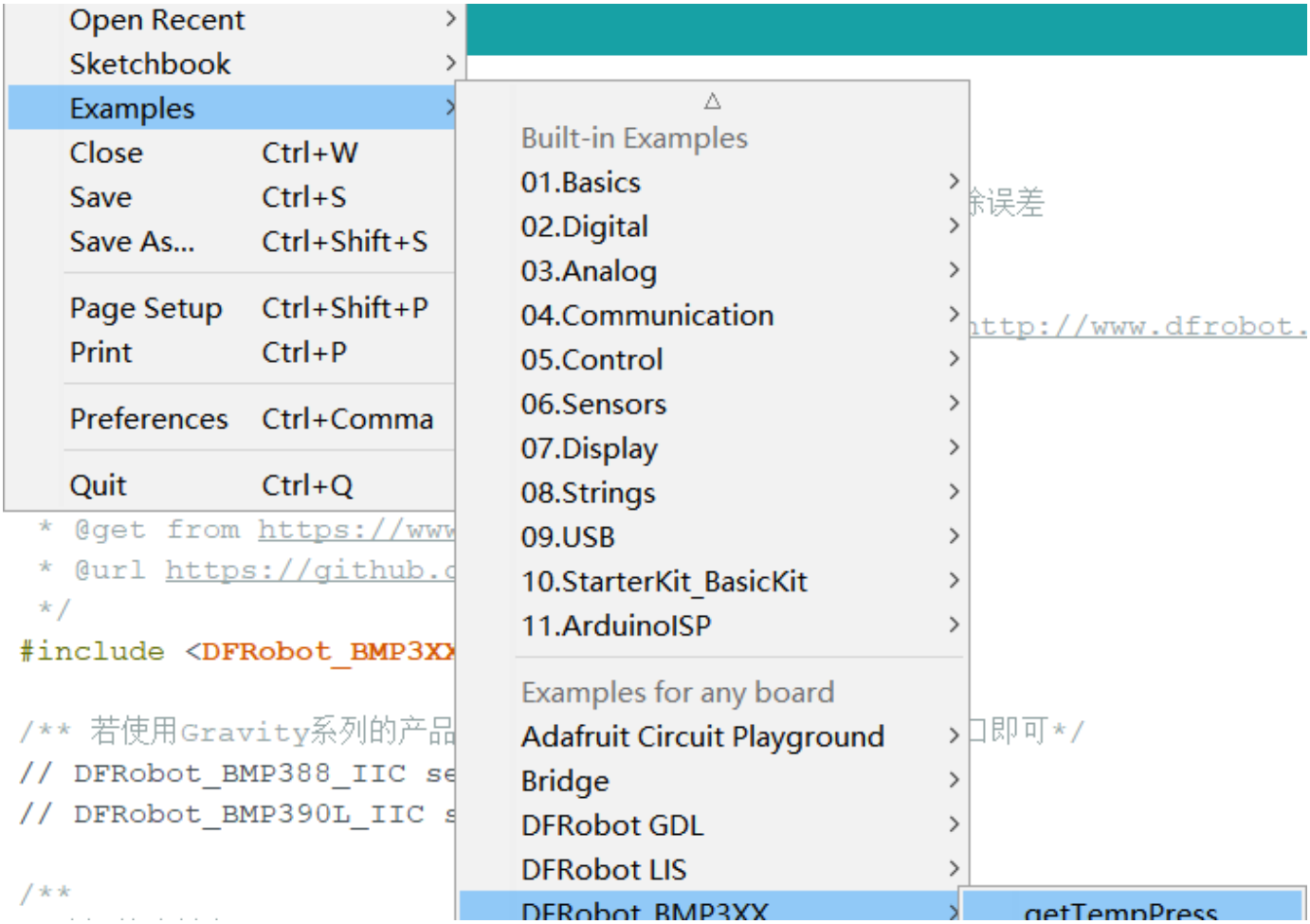
```
*        eTempDis:  Disable temperature measurement , eTempEn:  Enable temperature measu
*        eSleepMode, eForcedMode, eNormalMode Three modes:
*          Sleep mode: It will be in sleep mode by default after power-on reset. In this
*                      All registers are accessible for reading the chip ID and compensa
*          Forced mode: In this mode, the sensor will take a single measurement accordir
*                       completed, the sensor will return to sleep mode, and the measure
*          Normal mode: Continuously loop between the measurement period and the standby
*/
void setPWRMode(uint8_t mode);

/**
* @brief Configure the oversampling when measuring pressure and temperature (OSR:over-sa
* @param mode Oversampling mode of pressure and temperature measurement need to be set.
       6 pressure oversampling modes:
          ePressOSRMode1,  Pressure sampling×1，16 bit / 2.64 Pa（Recommend temperature
          ePressOSRMode2,  Pressure sampling×2，16 bit / 2.64 Pa（Recommend temperature
          ePressOSRMode4,  Pressure sampling×4，18 bit / 0.66 Pa（Recommend temperature
          ePressOSRMode8,  Pressure sampling×8，19 bit / 0.33 Pa（Recommend temperature
          ePressOSRMode16,  Pressure sampling×16，20 bit / 0.17 Pa（Recommend temperatur
          ePressOSRMode32,  Pressure sampling×32，21 bit / 0.085 Pa（Recommend temperatu
        6 temperature oversampling modes
          eTempOSRMode1,  Temperature sampling×1，16 bit / 0.0050 °C
          eTempOSRMode2,  Temperature sampling×2，16 bit / 0.0025 °C
          eTempOSRMode4,  Temperature sampling×4，18 bit / 0.0012 °C
```

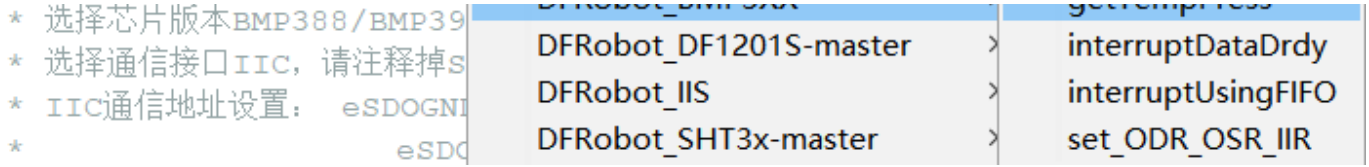
```
          eTempOSRMode8,  Temperature sampling×8, 19 bit / 0.0006 °C
          eTempOSRMode16,  Temperature sampling×16, 20 bit / 0.0003 °C
          eTempOSRMode32,  Temperature sampling×32, 21 bit / 0.00015 °C
*/
void setOSRMode(uint8_t mode);


/**
* @brief Set the output data rate setting in subdivision/sub-sampling mode(ODR:output da
* @param mode The output data rate needs to be set, configurable mode:
*        BMP3XX_ODR_200_HZ, BMP3XX_ODR_100_HZ, BMP3XX_ODR_50_HZ, BMP3XX_ODR_25_HZ, BMP3X
*        BMP3XX_ODR_6P25_HZ, BMP3XX_ODR_3P1_HZ, BMP3XX_ODR_1P5_HZ, BMP3XX_ODR_0P78_HZ, B
*        BMP3XX_ODR_0P2_HZ, BMP3XX_ODR_0P1_HZ, BMP3XX_ODR_0P05_HZ, BMP3XX_ODR_0P02_HZ, B
*        BMP3XX_ODR_0P006_HZ, BMP3XX_ODR_0P003_HZ, BMP3XX_ODR_0P0015_HZ
* @return Return True indicate configuration succeed, False indicate failed and remains
bool setODRMode(uint8_t mode);

/**
* @brief IIR filter coefficient configuration (IIR filtering)
* @param mode Set IIR filter coefficient, configurable mode:
*        BMP3XX_IIR_CONFIG_COEF_0, BMP3XX_IIR_CONFIG_COEF_1, BMP3XX_IIR_CONFIG_COEF_3,
*        BMP3XX_IIR_CONFIG_COEF_7, BMP3XX_IIR_CONFIG_COEF_15, BMP3XX_IIR_CONFIG_COEF_31,
*        BMP3XX_IIR_CONFIG_COEF_63, BMP3XX_IIR_CONFIG_COEF_127
*/
void setIIRMode(uint8_t mode);
```

```
/**
* @brief Get FIFO cached data size
* @return Range of returned value: 0-511
*/
uint16_t getFIFOLength(void);

/**
* @brief Get the water level set by FIFO
* @return Range of returned value: 0-511
*/
uint16_t getFIFOWTMValue(void);
```

Sample Code 1 - Read the measured frequency and data (temperature, air pressure, altitude) (getTempPress.ino)

- Choose getTempPress.ino


Open Recent
Sketchbook
Examples
Close Ctrl+W
Save Ctrl+S
Save As... Ctrl+Shift+S
Page Setup Ctrl+Shift+P
Print Ctrl+P
Preferences Ctrl+Comma
Quit Ctrl+Q
Built-in Examples
01.Basics
02.Digital
03.Analog
04.Communication
05.Control
06.Sensors
07.Display
08.Strings
09.USB
10.StarterKit_BasicKit
11.ArduinoISP
Examples for any board
Adafruit Circuit Playground
Bridge
DFRobot GDL
DFRobot LIS
DFRobot_BMP3XX
getTempPress
* @get from https://www
* @url https://github.c
*/
#include <DFRobot_BMP3XX
/** 若使用Gravity系列的产品
// DFRobot_BMP388_IIC se
// DFRobot_BMP390L_IIC s
/**
除误差
http://www.dfrobot.
口即可*/



- Burning program

```
/*!
 * @file getTempPress.ino
 * @brief Get measurement frequency and data of the sensor (temperature, pressure, altitud
 * @n You can write in the current altitude to calibrate the sensor, eliminating errors fo
 * @n the barometric altitude.
 * @n Get the current measurement frequency, temperature and pressure values.
 * @n Altitude is calculated from barometric pressure measurements and calibration values.
 * @copyright   Copyright (c) 2010 DFRobot Co.Ltd (http://www.dfrobot.com)
 * @licence     The MIT License (MIT)
 * @author [qsj](qsj.huang@dfrobot.com)
 * @version  V0.1
 * @date  2021-4-30
 * @get from https://www.dfrobot.com
 * @url https://github.com/DFRobot/DFRobot_BMP3XX
 */
#include <DFRobot_BMP3XX.h>

/** If using Gravity products, choose these two interfaces and comment subsequent interfac
// DFRobot_BMP388_I2C sensor();
```

```
// DFRobot_BMP388_IIC sensor();
// DFRobot_BMP390L_IIC sensor();

/**
* Select the chip version BMP388/BMP390L

* Select communication interface IIC, please comment out SPI interface.
* IIC communication address settings: eSDOGND: connect SDO pin to GND, I2C address is 0×76
*                   eSDOVDD: Connect SDO pin to VDDIO (3v3), I2C address is 0×77 now
*/
// DFRobot_BMP388_IIC sensor(&Wire, sensor.eSDOVDD);
 DFRobot_BMP390L_IIC sensor(&Wire, sensor.eSDOVDD);
/**
* Select chip version BMP388/BMP390L
* Select communication port SPI, please comment out IIC port
* Set up digital pin according to the on-board pin connected with SPI chip-select pin.
* Notice: csPin used here is D3 digital pin on ESP32, other non-conflicting pins can also
* as external interrupt pins.
*/
// uint8_t csPin = D3;
// DFRobot_BMP388_SPI sensor(&SPI, csPin);
// DFRobot_BMP390L_SPI sensor(&SPI, csPin);


/* If you do not need to eliminate the absolute difference of measurement, please comment
```

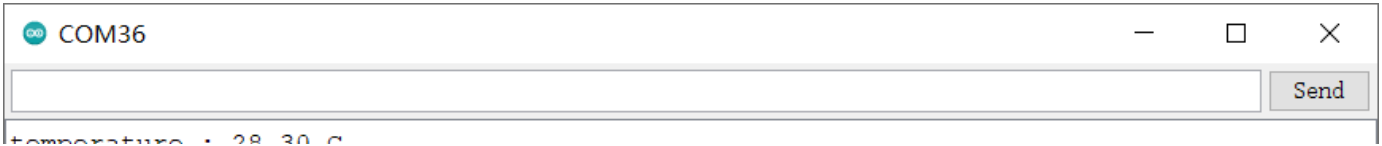
```
#define CALIBRATE_ABSOLUTE_DIFFERENCE

void setup(void)
{
  Serial.begin(115200);


  int rslt;
  while( ERR_OK != (rslt = sensor.begin()) ){
    if(ERR_DATA_BUS == rslt){
      Serial.println("Data bus error!!!");
    }else if(ERR_IC_VERSION == rslt){
      Serial.println("Chip versions do not match!!!");
    }
    delay(3000);
  }
  Serial.println("Begin ok!");

  /**
  * 6 commonly used sampling modes that allows users to configure easily, mode:
  *      eUltraLowPrecision, Ultra-low precision, suitable for monitoring weather (lowest
  *                          the power is mandatory mode.
  *      eLowPrecision, Low precision, suitable for random detection, power is normal mode
  *      eNormalPrecision1, Normal precision 1, suitable for dynamic detection on handheld
  *                         devices (e.g on mobile phones), power is normal mode.
```

```
*      eNormalPrecision2, Normal precision 2, suitable for drones, power is normal mode.
*      eHighPrecision, High precision, suitable for low-power handled devices (e.g mobil
*                      power is in normal mode.
*      eUltraPrecision, Ultra-high precision, suitable for indoor navigation, its acquis
*                       be extremely low, and the acquisition cycle is 1000 ms.
*/
while( !sensor.setSamplingMode(sensor.eUltraPrecision) ){
  Serial.println("Set samping mode fail, retrying....");
  delay(3000);
}

delay(100);
#ifdef CALIBRATE_ABSOLUTE_DIFFERENCE
/**
* Calibrate the sensor according to the current altitude
* In this example, we use an altitude of 540 meters in Wenjiang District of Chengdu (Chi
* Please change to the local altitude when using it.
* If this interface is not called, the measurement data will not eliminate the absolute
* Notice: This interface is only valid for the first call.
*/
if( sensor.calibratedAbsoluteDifference(540.0) ){
  Serial.println("Absolute difference base value set successfully!");
}
#endif
```

```
  /* Get the sampling period of the current measurement mode, unit: us */
  float sampingPeriodus = sensor.getSamplingPeriodUS();
  Serial.print("samping period : ");
  Serial.print(sampingPeriodus);
  Serial.println(" us");

  /* Get the sampling frequency of the current measurement mode, unit: Hz */
  float sampingFrequencyHz = 1000000 / sampingPeriodus;
  Serial.print("samping frequency : ");
  Serial.print(sampingFrequencyHz);
  Serial.println(" Hz");

  Serial.println();
  delay(1000);
}

void loop()
{
  /* Read currently measured temperature date directly, unit: °C */
  float temperature = sensor.readTempC();
  Serial.print("temperature : ");
  Serial.print(temperature);
  Serial.println(" C");
```

```
  /* Directly read the currently measured pressure data, unit: pa */
  float Pressure = sensor.readPressPa();
  Serial.print("Pressure : ");
  Serial.print(Pressure);
  Serial.println(" Pa");

  /* Read altitude, unit: m */
  float altitude = sensor.readAltitudeM();
  Serial.print("Altitude : ");
  Serial.print(altitude);
  Serial.println(" m");

  Serial.println();
  delay(1000);
}
```

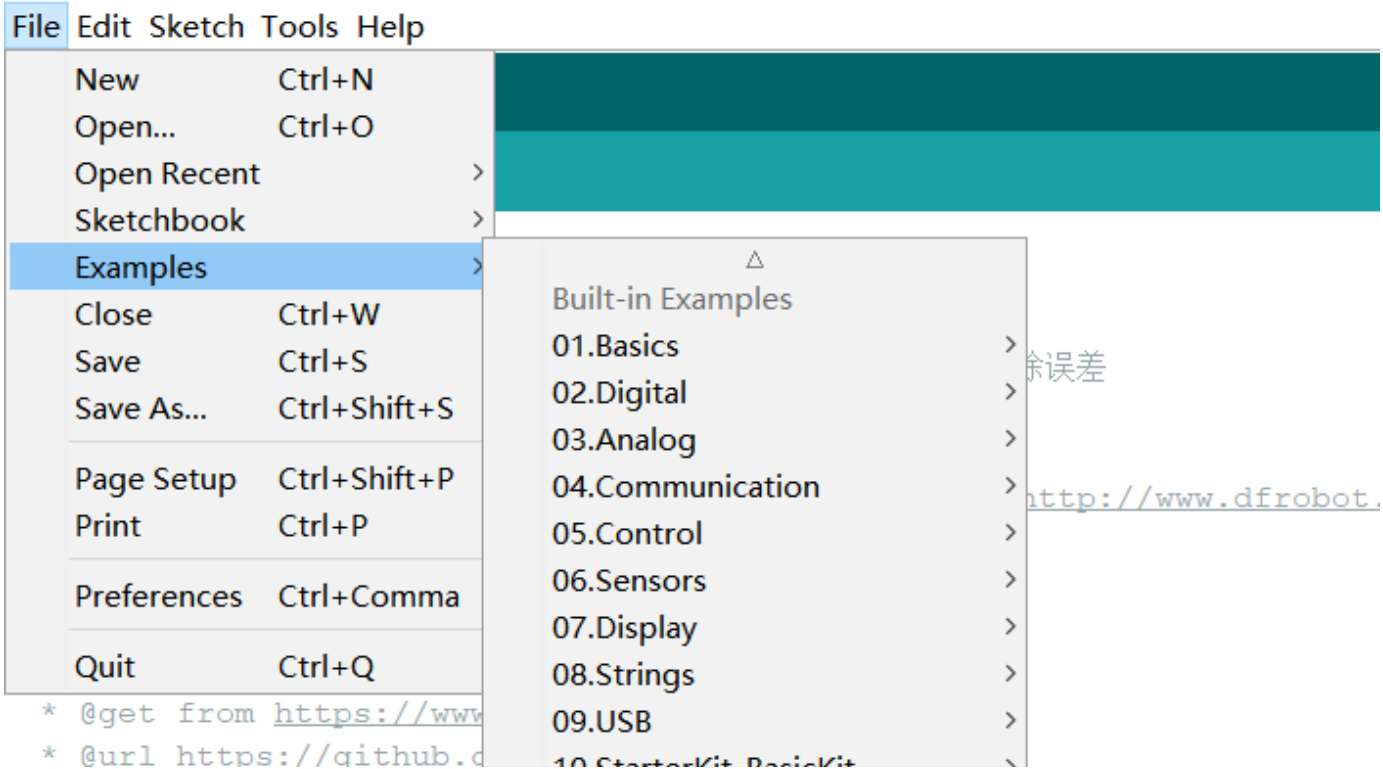
Result

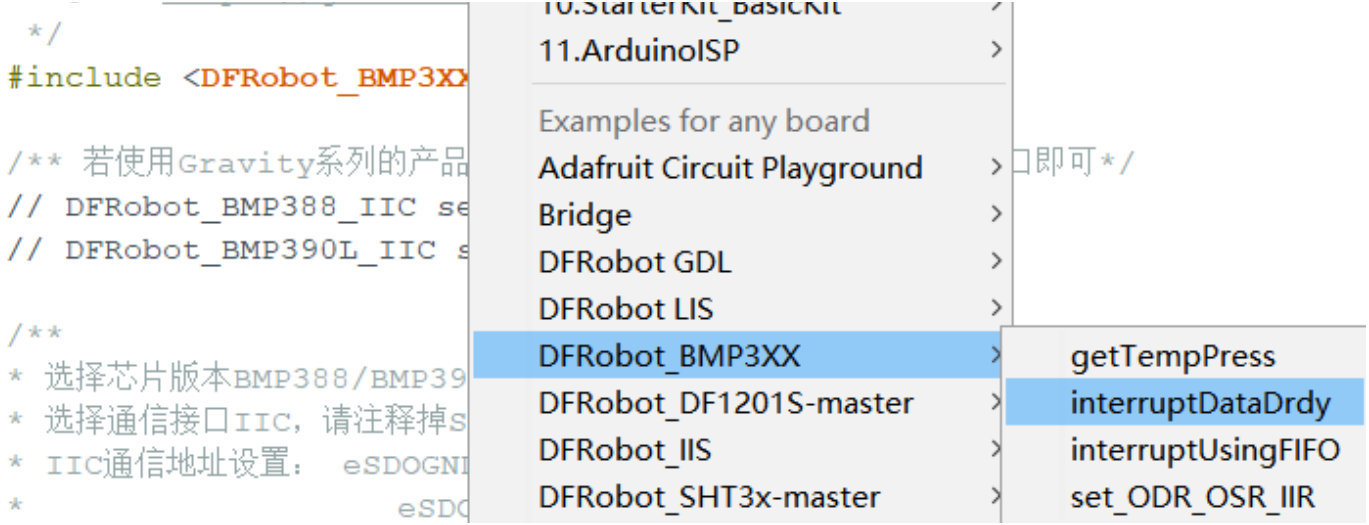


```
temperature : 28.30 C
Pressure : 94998.39 Pa
Altitude : 540.26 m

temperature : 28.28 C
Pressure : 94997.99 Pa
Altitude : 540.32 m

temperature : 28.26 C
Pressure : 94998.88 Pa
Altitude : 540.21 m

temperature : 28.23 C
Pressure : 94998.13 Pa
Altitude : 540.28 m

temperature : 28.22 C
Pressure : 94998.20 Pa
Altitude : 540.25 m

temperature : 28.20 C
Pressure : 94997.89 Pa
Altitude : 540.30 m
```

Sample Code 2-data interrupt function (interruptDataDrdy.ino)

- Choose interruptDataDrdy.ino





- Burning program

```
/*!
 * @file interruptDataDrdy.ino
 * @brief Demonstrate ready data (temperature/pressure) interrupt
 * @n When measured data, the sensor will generate a 2.5 ms pulse signal by INT in the nor
 * @n register locked state.
 * @copyright   Copyright (c) 2010 DFRobot Co.Ltd (http://www.dfrobot.com)
 * @licence     The MIT License (MIT)
 * @author [qsj](qsj.huang@dfrobot.com)
 * @version  V0.1
 * @date  2021-4-30
 * @get from https://www.dfrobot.com
 * @url https://github.com/DFRobot/DFRobot_BMP3XX
 */
#include <DFRobot_BMP3XX.h>

/**
* Select chip version BMP388/BMP390L
* Select IIC communication interface, please comment out SPI interface.
* IIC communication address settings: eSDOGND: connect SDO pin to GND, I2C address is 0x76
```

```
* IIC communication address settings: eSDOGND: Connect SDO pin to GND, I2C address is 0×7
*                   eSDOVDD: Connect SDO pin to VDDIO (3v3), I2C address is 0×77 now
* Notice: If using Gravity products, default IIC communication address is: 0×77（eSDOVDD）
*/
//DFRobot_BMP388_IIC sensor(&Wire, sensor.eSDOVDD);

 DFRobot_BMP390L_IIC sensor(&Wire, sensor.eSDOVDD);

/**
* Select the chip version BMP388/BMP390L
* Select IIC communication interface, please comment out SPI interface.
* Set up digital pin according to the on-board pin connected with SPI chip-select pin.
* Notice: csPin used here is D3 digital pin on ESP32, other non-conflicting pins can also
* as external interrupt pins.
*/
// uint8_t csPin = D3;
// DFRobot_BMP388_SPI sensor(&SPI, csPin);
// DFRobot_BMP390L_SPI sensor(&SPI, csPin);


/* If you do not need to eliminate the absolute difference of measurement, please comment
#define CALIBRATE_ABSOLUTE_DIFFERENCE

/* Interrupt flag */
volatile uint8_t flag = 0;
```

```
/* External interrupt flag */
void interrupt()
{
  if(flag ==0){
    flag = 1;
  }
}

void setup(void)
{
  Serial.begin(115200);

  int rslt;
  while( ERR_OK != (rslt = sensor.begin()) ){
    if(ERR_DATA_BUS == rslt){
      Serial.println("Data bus error!!!");
    }else if(ERR_IC_VERSION == rslt){
      Serial.println("Chip versions do not match!!!");
    }
    delay(3000);
  }
  Serial.println("Begin ok!");

  /**
```

```
* interrupt configuration
* mode The interrupt mode needs to set. The following modes add up to mode:
*      Interrupt pin output mode: eINTPinPP: Push pull, eINTPinOD: Open drain
*      Interrupt pin active level: eINTPinActiveLevelLow: Active low, eINTPinActiveLevel
*      Register interrupt latch: eINTLatchDIS: Disable, eINTLatchEN: Enable
*      FIFO water level reached interrupt: eINTFWTMDIS: Disable, eINTFWTMEN: Enable
*      FIFO full interrupt: eINTFFullDIS: Disable, eINTFFullEN: Enable
*      Interrupt pin initial (invalid, non-interrupt) level: eINTInitialLevelLOW: Low, e
*      Temperature/pressure data ready interrupt: eINTDataDrdyDIS: Disable, eINTDataDrdy
* Notice: In non-latching mode (eINTLatchDIS), interrupt signal is 2.5 ms pulse signal
* Note: When using eINTPinActiveLevelLow (Active low interrupt pin), you need to use eIN
*       level of interrupt pin is high). Please use “FALLING” to trigger the following i
*       When using eINTPinActiveLevelHigh (Active low interrupt pin), you need to use el
*       level of interrupt pin is high). Please use “RISING” to trigger the following in
*/
sensor.setINTMode(sensor.eINTPinPP +
                  sensor.eINTPinActiveLevelHigh +
                  sensor.eINTLatchDIS +
                  sensor.eINTFWTMDIS +
                  sensor.eINTFFullDIS +
                  sensor.eINTInitialLevelLOW +
                  sensor.eINTDataDrdyEN);

delay(100);
#if defined CALIBRATE_ABSOLUTE_DIFFERENCE
```

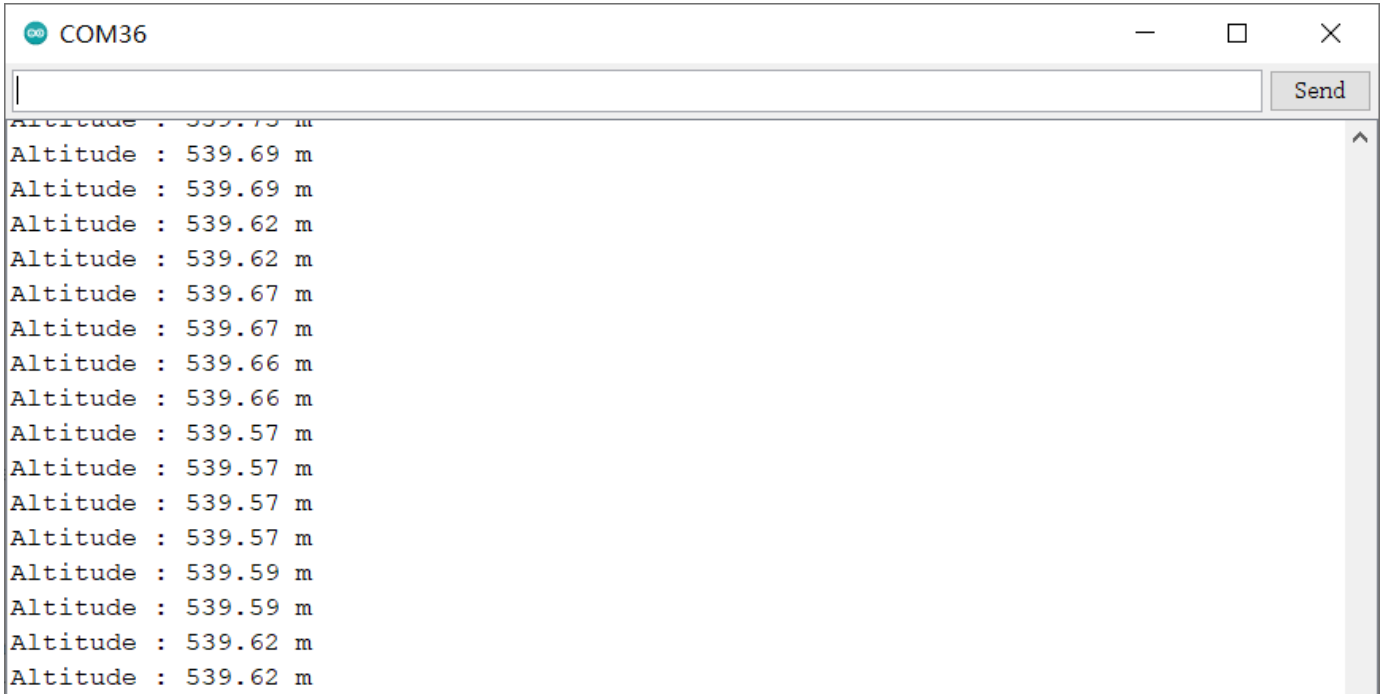
```
#ifdef CALIBRATE_ABSOLUTE_DIFFERENCE
/**
* Calibrate the sensor according to the current altitude
* In this example, we use an altitude of 540 meters in Wenjiang District of Chengdu (Chi
* Please change to the local altitude when using it.
* If this interface is not called, the measurement data will not eliminate the absolute
* Note: This interface is only valid for the first call
*/
if( sensor.calibratedAbsoluteDifference(540.0) ){
  Serial.println("Absolute difference base value set successfully!");
}
#endif

#if defined(ESP32) || defined(ESP8266)
//D6 pin is used as interrupt pin by default, other non-conflicting pins can also be sel
//interrupt pins.
attachInterrupt(digitalPinToInterrupt(D6)/*Query the interrupt number of the D6 pin*/,ir
#elif defined(ARDUINO_SAM_ZERO)
//Pin 5 is used as interrupt pin by default, other non-conflicting pins can also be sele
// interrupt pins
attachInterrupt(digitalPinToInterrupt(5)/*Query the interrupt number of the 5 pin*/,inte
#else
/*    The Correspondence Table of AVR Series Arduino Interrupt Pins And Terminal Numbers
 * ------------------------------------------------------------------------------------
```

```
 * |                                        |  DigitalPin  | 2  | 3  |
 * |    Uno, Nano, Mini, other 328-based    |--------------------------------------
 * |                                        | Interrupt No | 0  | 1  |
 * |-----------------------------------------------------------------------------
 * |                                        |    Pin       | 2  | 3  | 21 | 20 | 19 | 18
 * |               Mega2560                 |--------------------------------------
 * |                                        | Interrupt No | 0  | 1  | 2  | 3  | 4  | 5
 * |-----------------------------------------------------------------------------
 * |                                        |    Pin       | 3  | 2  | 0  | 1  | 7  |
 * |    Leonardo, other 32u4-based          |--------------------------------------
 * |                                        | Interrupt No | 0  | 1  | 2  | 3  | 4  |
 * |-----------------------------------------------------------------------------
 */
/*                    The Correspondence Table of micro:bit Interrupt Pins And Termina
 * ------------------------------------------------------------------------------
 * |             micro:bit                      | DigitalPin |P0-P20 can be used as an
 * |  (When using as an external interrupt,     |-----------------------------------
 * |no need to set it to input mode with pinMode)|Interrupt No|Interrupt number is a pir
 * |-----------------------------------------------------------------------------
 */
attachInterrupt(/*Interrupt No*/0,interrupt,CHANGE);//Open the external interrupt 0, cor
   //UNO(2), Mega2560(2), Leonardo(3), microbit(P0).
#endif
```

```
  /* Get the sampling period of the current measurement mode, unit: us */
  float sampingPeriodus = sensor.getSamplingPeriodUS();
  Serial.print("samping period : ");
  Serial.print(sampingPeriodus);
  Serial.println(" us");


  /* Get the sampling frequency of the current measurement mode, unit: Hz */
  float sampingFrequencyHz = 1000000 / sampingPeriodus;
  Serial.print("samping frequency : ");
  Serial.print(sampingFrequencyHz);
  Serial.println(" Hz");

  Serial.println();
  delay(1000);
}

void loop()
{
  if(flag == 1){
    flag = 0;
    /* When data is ready and the interrupt is triggered, read altitude, unit: m */
    float altitude = sensor.readAltitudeM();
    Serial.print("Altitude : ");
    Serial.print(altitude);
```

```
    Serial.println(" m");
  }
}
```

Result

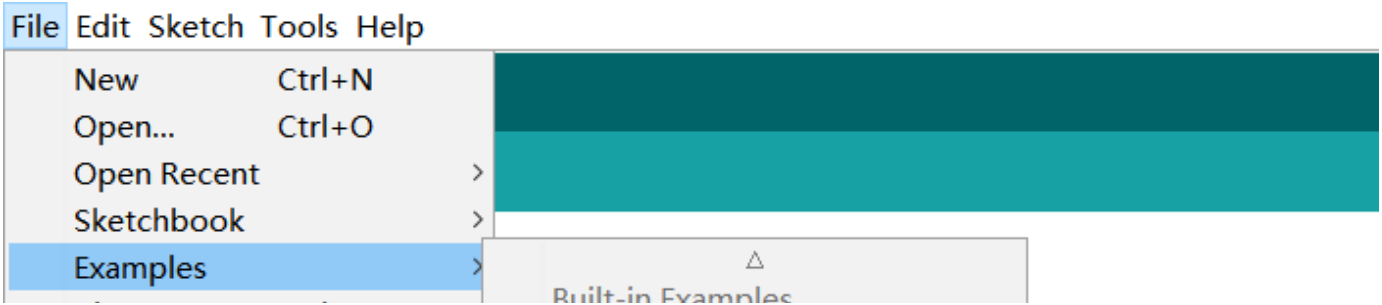


```
Altitude : 539.69 m
Altitude : 539.69 m
Altitude : 539.64 m
Altitude : 539.64 m
Altitude : 539.71 m
Altitude : 539.71 m
Altitude : 539.68 m
Altitude : 539.68 m
Altitude : 539.71 m
```

Autoscroll Show timestamp No line ending 115200 baud Clear output

Sample Code 3-FIFO water level or full interrupt function (interruptUsingFIFO.ino)

- Select interruptUsingFIFO.ino




Close Ctrl+W
Save Ctrl+S
Save As... Ctrl+Shift+S
Page Setup Ctrl+Shift+P
Print Ctrl+P
Preferences Ctrl+Comma
Quit Ctrl+Q
01.Basics
02.Digital
03.Analog
04.Communication
05.Control
06.Sensors
07.Display
08.Strings
09.USB
10.StarterKit_BasicKit
11.ArduinoISP
Examples for any board
Adafruit Circuit Playground
Bridge
DFRobot GDL
DFRobot LIS
DFRobot_BMP3XX
DFRobot_DF1201S-master
DFRobot_IIS
getTempPress
interruptDataDrdy
interruptUsingFIFO
* @get from https://www
* @url https://github.c
*/
#include <DFRobot_BMP3XX
/** 若使用Gravity系列的产品
// DFRobot_BMP388_IIC se
// DFRobot_BMP390L_IIC s
/**
* 选择芯片版本BMP388/BMP39
* 选择通信接口IIC，请注释掉S
* IIC通信地址设置： eSDOGN
除误差
http://www.dfrobot
口即可*/

- Burning program

```
/*!
 * @file interruptUsingFIFO.ino
 * @brief Demonstrate FIFO water level interrupt or FIFO full interrupt:
 * @n Empty the FIFO first, and then start to obtain the cached measurement data in it
 * @n When receiving FIFO water level interrupt signal generated by interrupt pin, read ou
 * @n and calculate the average value before printing it out.
 * @copyright   Copyright (c) 2010 DFRobot Co.Ltd (http://www.dfrobot.com)
 * @licence     The MIT License (MIT)
 * @author [qsj](qsj.huang@dfrobot.com)
 * @version  V0.1
 * @date  2021-4-30
 * @get from https://www.dfrobot.com
 * @url https://github.com/DFRobot/DFRobot_BMP3XX
 */
#include <DFRobot_BMP3XX.h>

/**
* Select chip version BMP388/BMP390L
* Select communication interface IIC, please comment out SPI interface
```

```
* Select communication interface IIC, please comment out SPI interface.
* IIC communication address settings: eSDOGND: connect SDO pin to GND, I2C address is 0x76
*                   eSDOVDD: Connect SDO pin to VDDIO (3v3), I2C address is 0x77 now
* Notice: If using Gravity products, default IIC communication address is: 0x77（eSDOVDD）
*/

// DFRobot_BMP388_IIC sensor(&Wire, sensor.eSDOVDD);
 DFRobot_BMP390L_IIC sensor(&Wire, sensor.eSDOVDD);

/**
* Select the chip version BMP388/BMP390L
* Select SPI communication interface, please comment out IIC interface.
* Set up digital pin according to the on-board pin connected with SPI chip-select pin.
* Notice: csPin used here is D3 digital pin on ESP32, other non-conflicting pins can also
*         selected as external interrupt pins.
*/
// uint8_t csPin = D3;
// DFRobot_BMP388_SPI sensor(&SPI, csPin);
// DFRobot_BMP390L_SPI sensor(&SPI, csPin);


/*If you do not need to eliminate the absolute difference of measurement, please comment t
#define CALIBRATE_ABSOLUTE_DIFFERENCE

/* Interrupt flag */
```

```
volatile uint8_t flag = 0;
/* External interrupt pin */
void interrupt()
{
  if(flag ==0){
    flag = 1;
  }
}

void setup(void)
{
  Serial.begin(115200);

  int rslt;
  while( ERR_OK != (rslt = sensor.begin()) ){
    if(ERR_DATA_BUS == rslt){
      Serial.println("Data bus error!!!");
    }else if(ERR_IC_VERSION == rslt){
      Serial.println("Chip versions do not match!!!");
    }
    delay(3000);
  }
  Serial.println("Begin ok!");

  /**
```

```
/**
* FIFO configuration 1
* mode The FIFO mode needs to set, the following modes add up to mode:
*      eFIFODIS: Disable FIFO, eFIFOEN: Enable FIFO
*      eFIFOStopOnFullDIS: Continue writing when full, eFIFOStopOnFullEN: Stop writing w
*      eFIFOTimeDIS: Disable, eFIFOTimeEN: Enable return to the sensor time frame after
*      eFIFOPressDIS: Disable pressure data storage, eFIFOPressEN: Enable pressure data
*      eFIFOTempDIS: Disable temperature data storage, eFIFOTempEN: Enable temperature d
*/
sensor.setFIFOMode1(sensor.eFIFOEN +
                    sensor.eFIFOStopOnFullDIS +
                    sensor.eFIFOTimeEN +
                    sensor.eFIFOPressEN +
                    sensor.eFIFOTempEN);

/**
* FIFO Configuration 2
* mode The FIFO mode needs to set, the following modes add up to mode:
*    8 FIFO sampling options for pressure and temperature data (1-128), the coefficient
*        eFIFOSubsampling0, eFIFOSubsampling1, eFIFOSubsampling2, eFIFOSubsampling3,
*        eFIFOSubsampling4, eFIFOSubsampling5, eFIFOSubsampling6, eFIFOSubsampling7,
*      eFIFODataSelectDIS: Unfiltered data (compensated or uncompensated), eFIFODataSele
*        data (compensated or uncompensated), plus two retention states: the same as "ur
*/
```

```
sensor.setFifoMode2(sensor.eFifoSubsampling2 +
                    sensor.eFIFODataSelectEN);

/**
* FIFO empty command and soft reset command of sensor
* mode Basic sensor commands, three types of commands:
*      BMP3XX_CMD_NOP, Null command
*      BMP3XX_CMD_FIFO_FLUSH, Clear all data in the FIFO without changing its settings
*      BMP3XX_CMD_SOFTRESET, Trigger a reset, all user configuration settings will be ov
*                            default state.
*/
sensor.setCommand(BMP3XX_CMD_FIFO_FLUSH);

/**
* FIFO water level settings configuration
* WTMSetting FIFO water level (0-511) needs to set. That FIFO fills up to the water leve
*/
uint16_t FIFOWTM = 500;
sensor.setFIFOWTM(FIFOWTM);

/**
* Interrupt configuration
* mode The interrupt mode needs to set. The following modes add up to mode:
*      Interrupt pin output mode: eINTPinPP: Push pull, eINTPinOD: Open drain
```

```
*      Interrupt pin active level: eINTPinActiveLevelLow: Active low, eINTPinActiveLevel
*      Register interrupt latch: eINTLatchDIS: Disable, eINTLatchEN: Enable
*      FIFO water level reached interrupt: eINTFWTMDIS: Disable, eINTFWTMEN: Enable
*      FIFO full interrupt: eINTFFullDIS: Disbale, eINTFFullEN: Enable
*      Initial(invalid, non-interrupt) interrupt pin level: eINTInitialLevelLOW: Low, el
*      Temperature/pressure data ready interrupt: eINTDataDrdyDIS: Disable, eINTDataDrdy
* Note: In non-latching mode (eINTLatchDIS), interrupt signal is 2.5 ms pulse signal
* Reminder: When using eINTPinActiveLevelLow (Active low interrupt pin), you need to use
*           (Initial level of interrupt pin is high). Please use “FALLING” to trigger th
*       When using eINTPinActiveLevelHigh (Active low interrupt pin),  you need to use e
*       level of interrupt pin is high). Please use “RISING” to trigger the following in
*/
sensor.setINTMode(sensor.eINTPinPP +
                  sensor.eINTPinActiveLevelHigh +
                  sensor.eINTLatchDIS +
                  sensor.eINTFWTMEN +
                  sensor.eINTFFullDIS +
                  sensor.eINTInitialLevelLOW +
                  sensor.eINTDataDrdyDIS);

delay(100);
#ifdef CALIBRATE_ABSOLUTE_DIFFERENCE
/**
* Calibrate the sensor according to the current altitude
```

```
* In this example, we use an altitude of 540 meters in Wenjiang District of Chengdu (Chi
* Please change to the local altitude when using it.
* If this interface is not called, the measurement data will not eliminate the absolute
* Note: This interface is only valid for the first call
*/

if( sensor.calibratedAbsoluteDifference(540.0) ){
  Serial.println("Absolute difference base value set successfully!");
}
#endif

#if defined(ESP32) || defined(ESP8266)
//D6 pin is used as interrupt pin by default, other non-conflicting pins can also be sel
//external interrupt pins
attachInterrupt(digitalPinToInterrupt(D6)/*Query the interrupt number of the D6 pin*/,ir
#elif defined(ARDUINO_SAM_ZERO)
//Pin 5 is used as interrupt pin by default, other non-conflicting pins can also be sele
//external interrupt pins
attachInterrupt(digitalPinToInterrupt(5)/*Query the interrupt number of the 5 pin*/,inte
#else
/*    The Correspondence Table of AVR Series Arduino Interrupt Pins And Terminal Numbers
 * ------------------------------------------------------------------------------------
 * |                                        |  DigitalPin  | 2  | 3  |
 * |    Uno, Nano, Mini, other 328-based    |---------------------------------------
 * |                                        | Interrupt No | 0  | 1  |
 * |
```

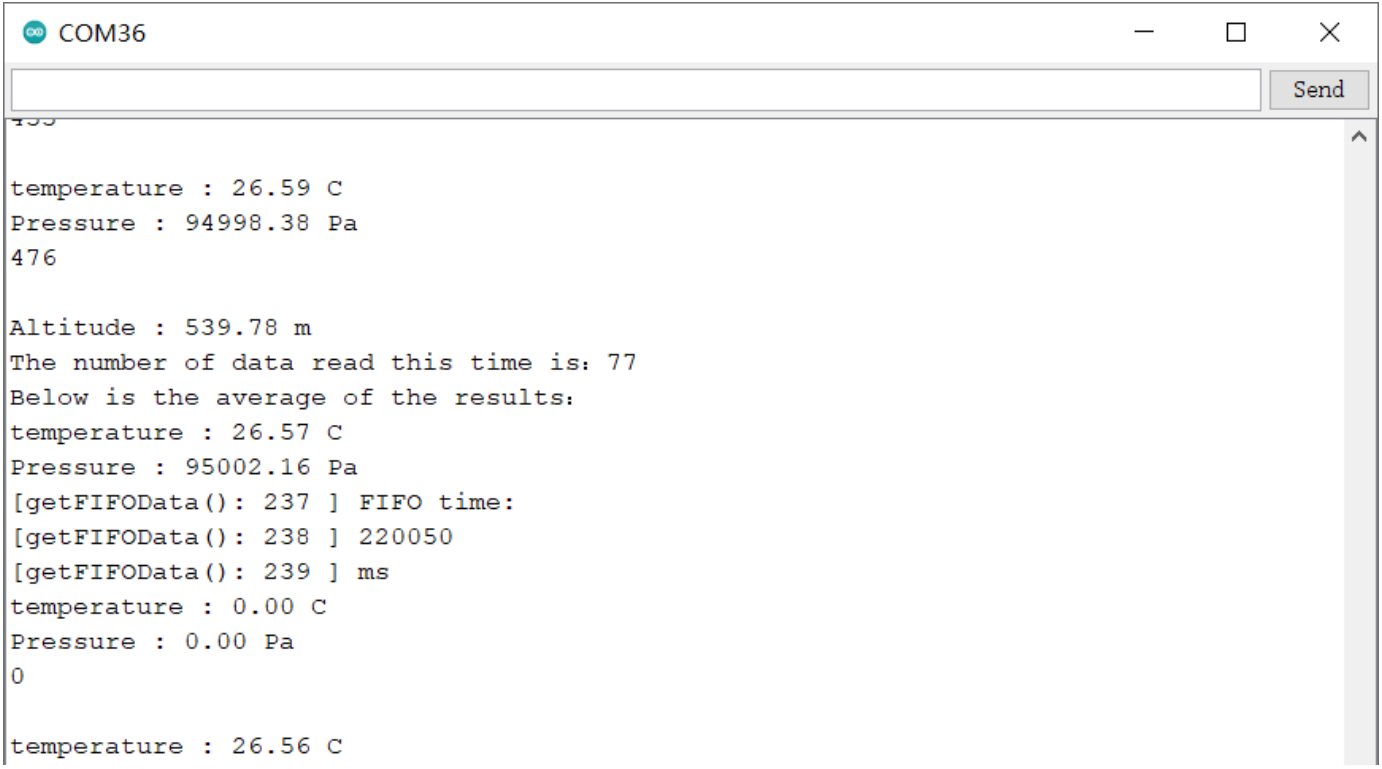
```
    * |-----------------------------------------------------------------------------
    * |                                        |    Pin       | 2  | 3  | 21 | 20 | 19 | 18
    * |               Mega2560                 |-------------------------------------------
    * |                                        | Interrupt No | 0  | 1  | 2  | 3  | 4  | 5
    * |-----------------------------------------------------------------------------
    * |                                        |    Pin       | 3  | 2  | 0  | 1  | 7  |
    * |    Leonardo, other 32u4-based          |-------------------------------------------
    * |                                        | Interrupt No | 0  | 1  | 2  | 3  | 4  |
    * |-----------------------------------------------------------------------------
    */
   /*                      The Correspondence Table of micro:bit Interrupt Pins And Termina
    * -----------------------------------------------------------------------------
    * |             micro:bit                       | DigitalPin |P0-P20 can be used as an
    * |  (When using as an external interrupt,      |-------------------------------------
    * |no need to set it to input mode with pinMode)|Interrupt No|Interrupt number is a pir
    * |-----------------------------------------------------------------------------
    */
   attachInterrupt(/*Interrupt No*/0,interrupt,CHANGE);//Open the external interrupt 0, cor
      //UNO(2), Mega2560(2), Leonardo(3), microbit(P0).
   #endif

   /* Empty data in FIFO, and its settings remains unchanged. */
   sensor.setCommand(BMP3XX_CMD_FIFO_FLUSH);
}
```

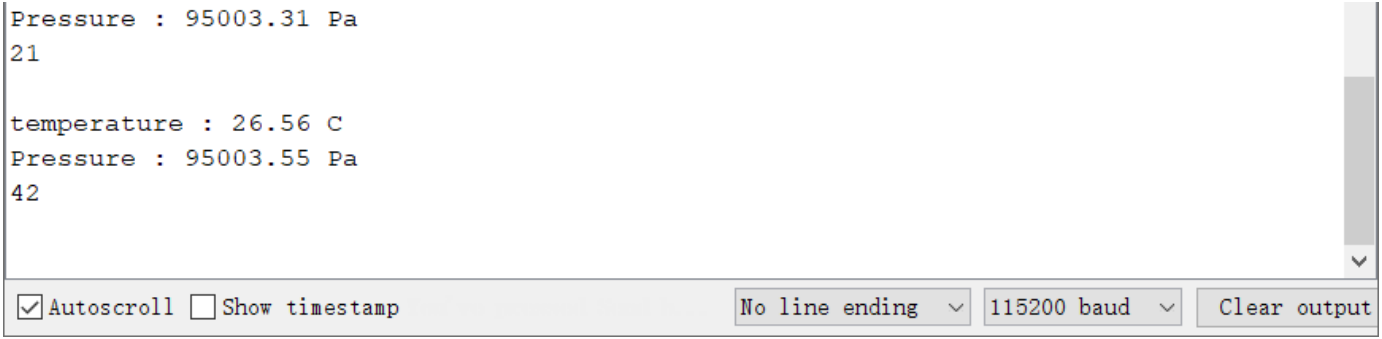
```
void loop()
{
  float fifoTemperatureC, fifoPressurePa;
  float fifoTemperatureSUM = 0, fifoPressureSUM = 0;
  uint8_t count = 0;

  if(flag == 1){
    /* When the water level interrupt is triggered, read the altitude, unit: m */
    float altitude = sensor.readAltitudeM();
    Serial.print("Altitude : ");
    Serial.print(altitude);
    Serial.println(" m");
    /* Read all the measurement data stored in the FIFO and sum them all  */
    while(sensor.getFIFOLength()){
      sensor.getFIFOData(fifoTemperatureC, fifoPressurePa);
      fifoTemperatureSUM += fifoTemperatureC;
      fifoPressureSUM += fifoPressurePa;
      count++;
    }
    Serial.print("The number of data read this time is: ");
    Serial.println(count);
    Serial.println("Below is the average of the results: ");
    Serial.print("temperature : ");
```

```
    /* At the same time, read and count the average temperature obtained in FIFO, unit: °C
    Serial.print(fifoTemperatureSUM/count);
    Serial.println(" C");
    Serial.print("Pressure : ");
    /* At the same time, read and count the average pressure value obtained in FIFO, unit:
    Serial.print(fifoPressureSUM/count);
    Serial.println(" Pa");
    count = 0;
    flag = 0;
  }
  sensor.getFIFOData(fifoTemperatureC, fifoPressurePa);
  Serial.print("temperature : ");
  /* Temperature data, unit: °C (When there is no temperature data in FIFO, its value will
  Serial.print(fifoTemperatureC);
  Serial.println(" C");
  Serial.print("Pressure : ");
  /* Pressure data, unit: pa (When there is no pressure data in FIFO, its value will be 0.
  Serial.print(fifoPressurePa);
  Serial.println(" Pa");
  /* The number of bytes of data stored in the FIFO */
  Serial.println(sensor.getFIFOLength());

  Serial.println();
  delay(300);
}
```

```
}
```

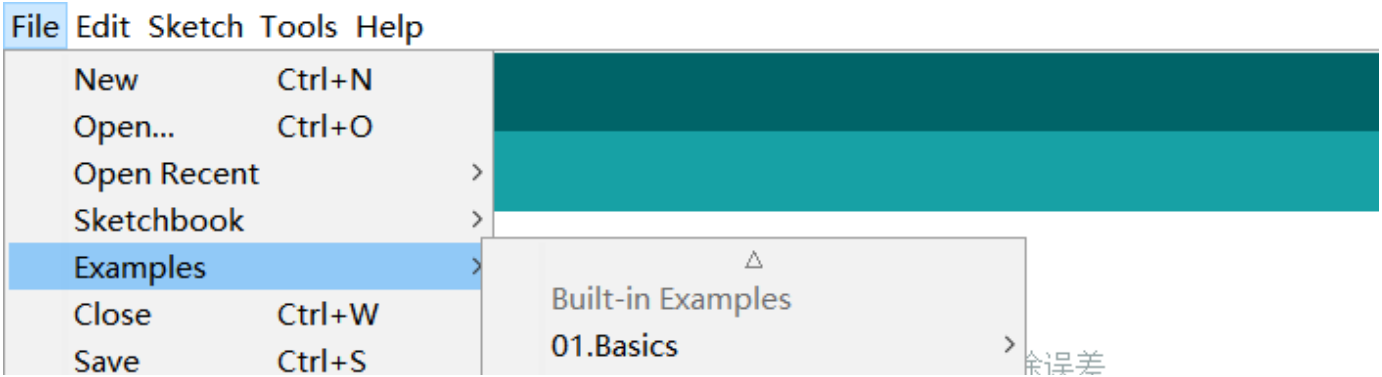
Result

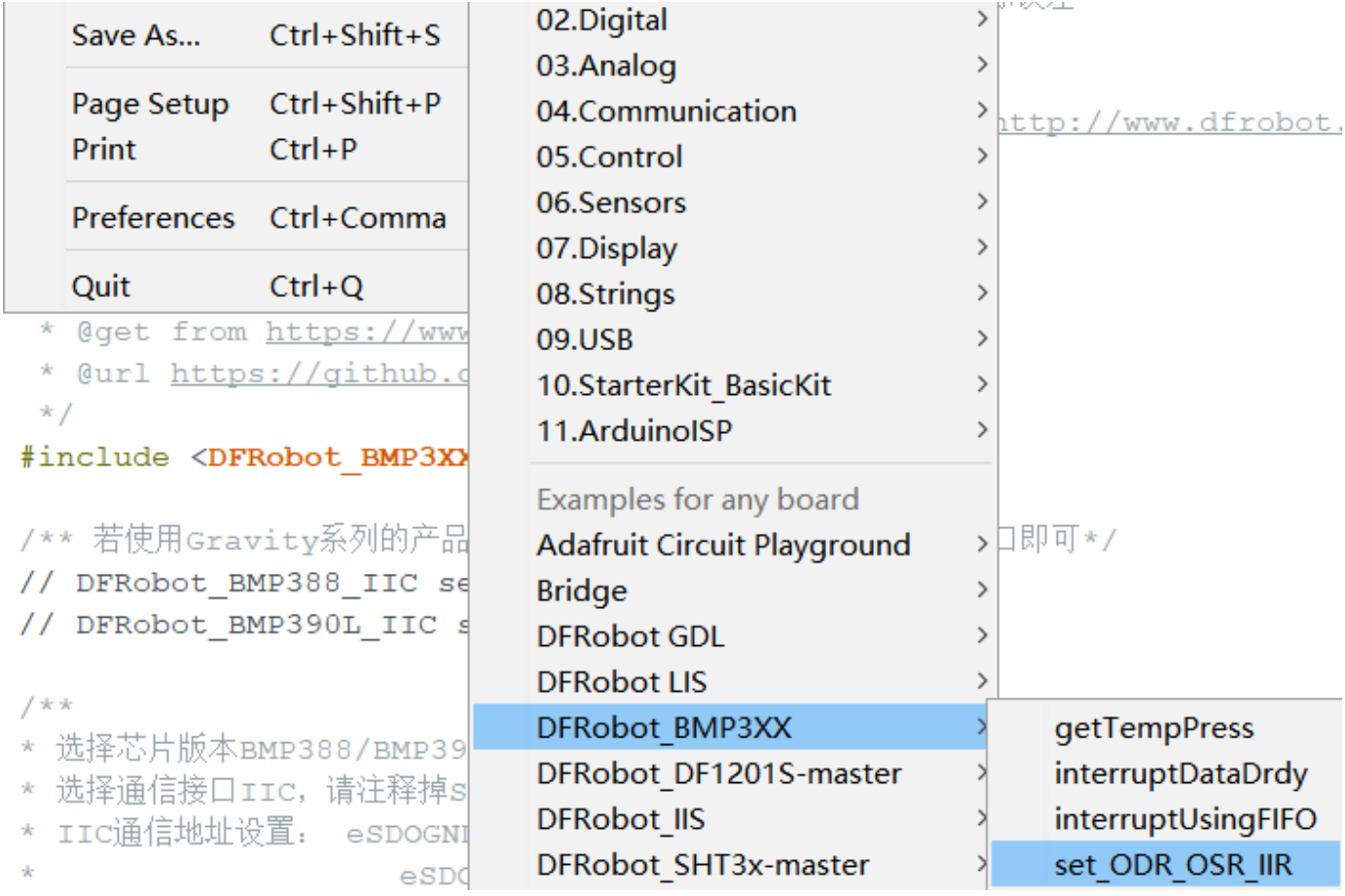

COM36
Send
temperature : 26.59 C
Pressure : 94998.38 Pa
476
Altitude : 539.78 m
The number of data read this time is: 77
Below is the average of the results:
temperature : 26.57 C
Pressure : 95002.16 Pa
[getFIFOData(): 237] FIFO time:
[getFIFOData(): 238] 220050
[getFIFOData(): 239] ms
temperature : 0.00 C
Pressure : 0.00 Pa
0
temperature : 26.56 C



Sample Code 4-data processing advanced setting function (set_ODR_OSR_IIR.ino)

- Select set_ODR_OSR_IIR.ino





- Burning program

- Burning program

```
/*!
 * @file set_ODR_OSR_IIR.ino
 * @brief Advanced data processing settings, configure more advanced data sampling and pro
 * @n Configure measurement mode: sleep mode, enforcement mode, normal mode
 * @n Configure pressure and temperature over-sampling mode (increase sampling times)
 * @n Set the output data rate setting in subdivision/sub-sampling mode (set the data outp
 * @n IIR filter coefficient setting (filter noise)
 * @copyright   Copyright (c) 2010 DFRobot Co.Ltd (http://www.dfrobot.com)
 * @licence     The MIT License (MIT)
 * @author [qsj](qsj.huang@dfrobot.com)
 * @version  V0.1
 * @date  2021-4-30
 * @get from https://www.dfrobot.com
 * @url https://github.com/DFRobot/DFRobot_BMP3XX
 */
#include <DFRobot_BMP3XX.h>

/**
* Select the chip version BMP388/BMP390L
```

```
* Select the chip version BMP388/BMP390L
* Select communication interface IIC, please comment out SPI interface.
* IIC communication address settings: eSDOGND: connect SDO pin to GND, I2C address is 0×76
*                    eSDOVDD: Connect SDO pin to VDDIO (3v3), I2C address is 0×77 now
* Notice: If using Gravity products, IIC communication address is 0×77 by default

*/
//DFRobot_BMP388_IIC sensor(&Wire, sensor.eSDOVDD);
 DFRobot_BMP390L_IIC sensor(&Wire, sensor.eSDOVDD);

/**
* Select the chip version BMP388/BMP390L
* Select communication interface SPI, please comment out IIC interface.
* Set up digital pin according to the on-board pin connected with SPI chip-select pin.
* Notice: csPin used here is D3 digital pin on ESP32, other non-conflicting pins can also
*/
// uint8_t csPin = D3;
// DFRobot_BMP388_SPI sensor(&SPI, csPin);
// DFRobot_BMP390L_SPI sensor(&SPI, csPin);


/* If you do not need to eliminate the absolute difference of measurement, please comment
#define CALIBRATE_ABSOLUTE_DIFFERENCE

void setup(void)
{
```

```
{
  Serial.begin(115200);

  int rslt;
  while( ERR_OK != (rslt = sensor.begin()) ){
    if(ERR_DATA_BUS == rslt){
      Serial.println("Data bus error!!!");
    }else if(ERR_IC_VERSION == rslt){
      Serial.println("Chip versions do not match!!!");
    }
    delay(3000);
  }
  Serial.println("Begin ok!");

  /**
  * Configure measurement mode and power mode
  * mode The measurement mode and power mode that need to set. The following modes add up
  *      ePressDIS: Disable pressure measurement, ePressEN: Enable pressure measurement
  *      eTempDIS: Disable temperature measurement, eTempEN: Enable temperature measuremer
  *      eSleepMode, eForcedMode/, eNormalMode Three modes:
  *        Sleep mode: It will be in sleep mode by default after power-on reset. In this r
  *                    are accessible for reading the chip ID and compensation coefficient
  *        Enforcement mode: In enforcement mode, the sensor will take a single measuremer
  *                     is completed, the sensor will return to sleep mode, and the measur
```

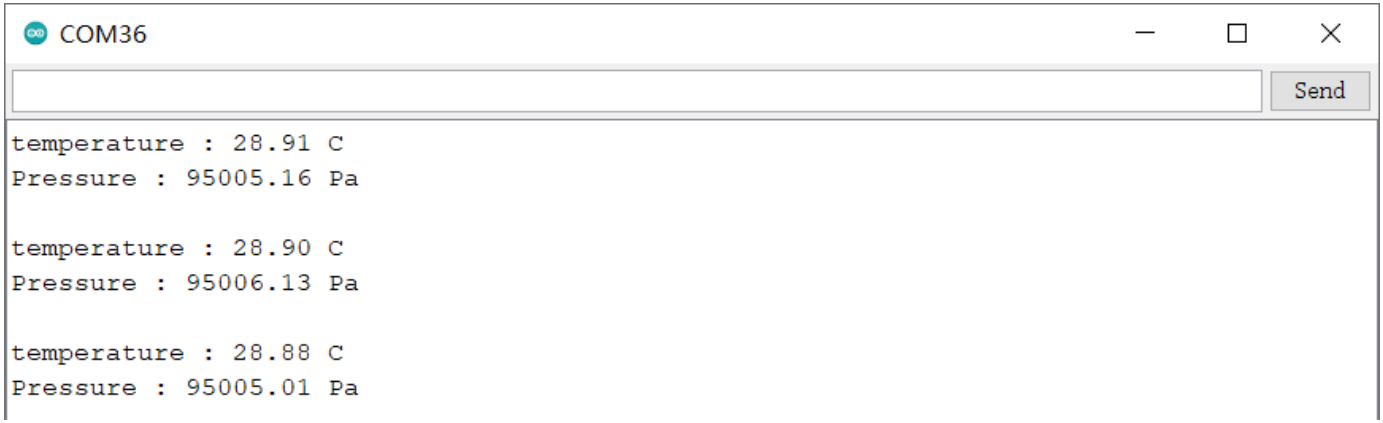
```
*        Normal mode: Continuously loop between the measurement period and the standby p
*                     the prescaler with different sampling frequency Fsampling=200Hz.
*/
sensor.setPWRMode(sensor.ePressEN +
                  sensor.eTempEN +
                  sensor.eNormalMode);

/**
*  Configure the oversampling when measuring pressure and temperature
*  mode Oversampling mode of pressure and temperature measurement need to set. The follo
*       6 pressure oversampling mode:
*         ePressOSRMode1, Pressure sampling × 1, 16 bit / 2.64 Pa (Recommend temperature
*         ePressOSRMode2, Pressure sampling × 2, 16 bit / 2.64 Pa (Recommend temperature
*         ePressOSRMode4, Pressure sampling × 4, 18 bit / 0.66 Pa (Recommend temperature
*         ePressOSRMode8, Pressure sampling × 8, 19 bit / 0.33 Pa (Recommend temperature
*         ePressOSRMode16, Pressure sampling × 16, 20 bit / 0.17 Pa (Recommend temperatu
*         ePressOSRMode32, Pressure sampling × 32, 21 bit / 0.085 Pa (Recommend temperat
*       6 temperature oversampling mode
*         eTempOSRMode1, Temperature sampling × 1, 16 bit / 0.0050 °C
*         eTempOSRMode2,  Temperature sampling × 2, 16 bit / 0.0025 °C
*         eTempOSRMode4,  Temperature sampling × 4, 18 bit / 0.0012 °C
*         eTempOSRMode8,  Temperature sampling × 8, 19 bit / 0.0006 °C
*         eTempOSRMode16,  Temperature sampling × 16, 20 bit / 0.0003 °C
*         eTempOSRMode32,  Temperature sampling × 32, 21 bit / 0.00015 °C
*/
```

```
*/
sensor.setOSRMode(sensor.ePressOSRMode4 +
                  sensor.eTempOSRMode1);

/**
* Configure output data rate in subdivision/sub-sampling mode
* mode The output data rate needs to set, configurable mode
*      BMP3XX_ODR_200_HZ, BMP3XX_ODR_100_HZ, BMP3XX_ODR_50_HZ, BMP3XX_ODR_25_HZ, BMP3XX_
*      BMP3XX_ODR_6P25_HZ, BMP3XX_ODR_3P1_HZ, BMP3XX_ODR_1P5_HZ, BMP3XX_ODR_0P78_HZ, BMP
*      BMP3XX_ODR_0P2_HZ, BMP3XX_ODR_0P1_HZ, BMP3XX_ODR_0P05_HZ, BMP3XX_ODR_0P02_HZ, BMP
*      BMP3XX_ODR_0P006_HZ, BMP3XX_ODR_0P003_HZ, BMP3XX_ODR_0P0015_HZ
*/
while( !sensor.setODRMode(BMP3XX_ODR_50_HZ) ){
  Serial.println("Set ODR mode fail! Please select lower frequency!");
  delay(3000);
}

/**
* IIR filter coefficient configuration
* mode Set IIR filter coefficient, configurable mode:
*      BMP3XX_IIR_CONFIG_COEF_0, BMP3XX_IIR_CONFIG_COEF_1, BMP3XX_IIR_CONFIG_COEF_3,
*      BMP3XX_IIR_CONFIG_COEF_7, BMP3XX_IIR_CONFIG_COEF_15, BMP3XX_IIR_CONFIG_COEF_31,
*      BMP3XX_IIR_CONFIG_COEF_63, BMP3XX_IIR_CONFIG_COEF_127
*/
```

```
  sensor.setIIRMode(BMP3XX_IIR_CONFIG_COEF_3);

  delay(100);
  #ifdef CALIBRATE_ABSOLUTE_DIFFERENCE
  /**
  * Calibrate the sensor according to the current altitude
  * In this example, we use an altitude of 540 meters in Wenjiang District of Chengdu (Chi
  * If this interface is not called, the measurement data will not eliminate the absolute
  * Note: This interface is only valid for the first call
  */
  if( sensor.calibratedAbsoluteDifference(540.0) ){
    Serial.println("Absolute difference base value set successfully!");
  }
  #endif
}

void loop()
{
  /* Read currently measured temperature date directly, unit: °C */
  float temperature = sensor.readTempC();
  Serial.print("temperature : ");
  Serial.print(temperature);
  Serial.println(" C");
```

```
  /* Directly read the currently measured pressure data, unit: pa */
  float Pressure = sensor.readPressPa();
  Serial.print("Pressure : ");
  Serial.print(Pressure);
  Serial.println(" Pa");


  Serial.println();
  delay(1000);
}
```

Result



```
temperature : 28.85 C
Pressure : 95003.78 Pa

temperature : 28.81 C
Pressure : 95002.47 Pa

temperature : 28.78 C
Pressure : 95001.75 Pa

temperature : 28.74 C
Pressure : 95002.94 Pa

temperature : 28.71 C
Pressure : 95002.49 Pa
```

Tutorial for Raspberry Pi

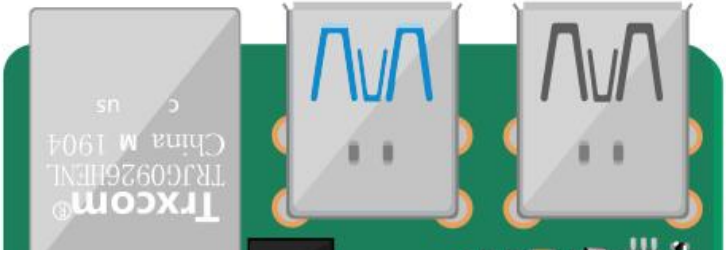
Requirements

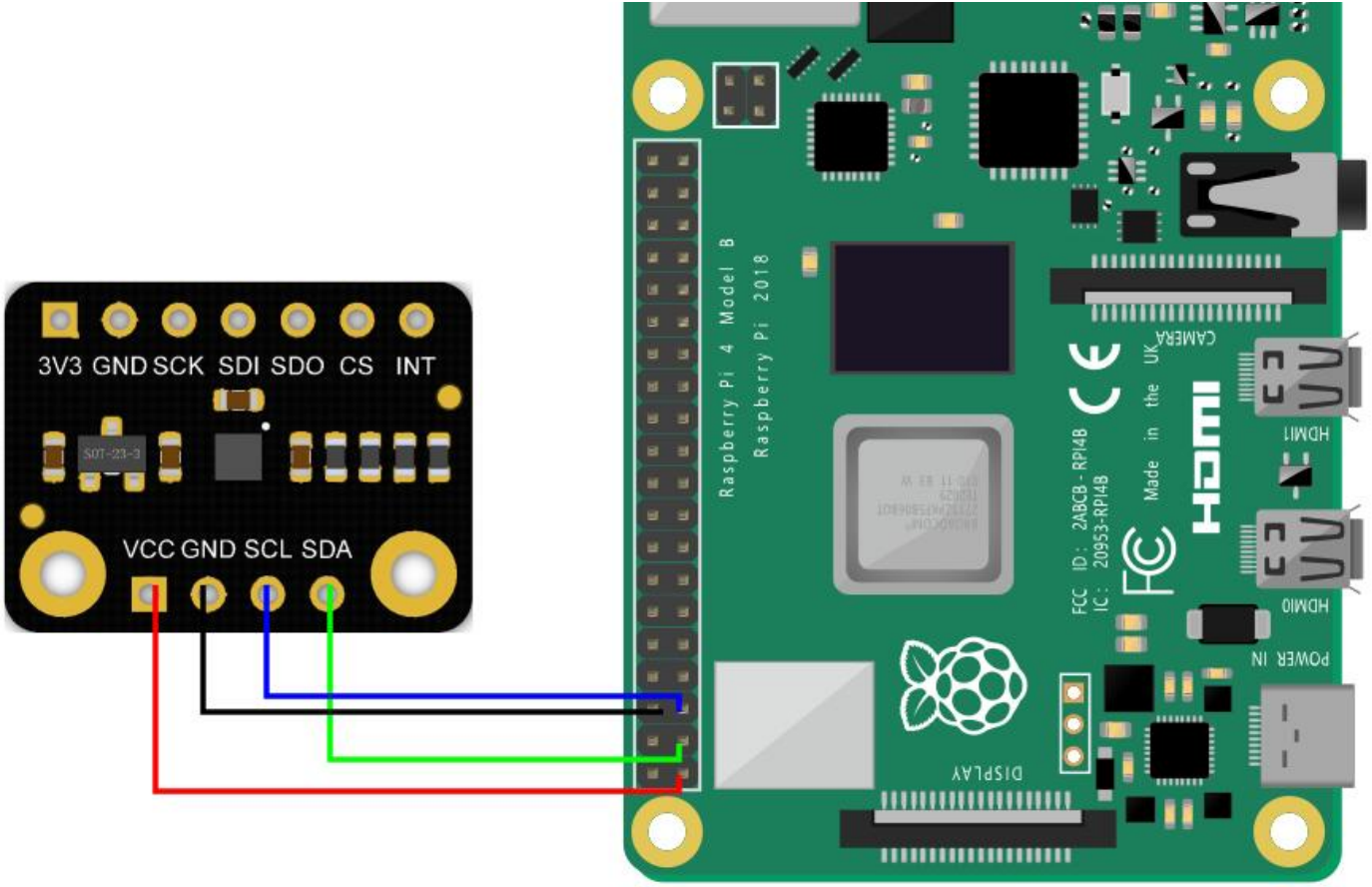
- Hardware

- Hardware
  - Raspberry Pi 4th Generation Type B (or similar) main control board x 1
  - BMP390L digital barometric pressure sensor × 1
  - Jumper wires
- **Software**
  - Download and install the **BMP3 series sensor python library** (https://github.com/DFRobot/DFRobot_BMP3XX). (About how to install the library? (https://www.arduino.cc/en/Guide/Libraries#.UxU8mdzF9H0))
  - RASPBIAN Raspberry Pi official operating system (https://www.raspberrypi.org/downloads/raspbian)

Connection Diagram

- Connect the module to the Raspberry Pi according to the connection diagram.





Install driver

Install driver

1. Start the I2C interface of the Raspberry Pi. If it is already enabled, you can skip this step. Open the terminal (Terminal), type the following command, and press Enter:

```
sudo raspi-config
```

Then use the up and down keys to select "5 Interfacing Options", press Enter, select "P5 I2C", and press Enter to confirm "YES". Restart the Raspberry Pi main control board.

2. To install Python dependent libraries and git, the Raspberry Pi needs to be connected to the Internet. If it is already installed, you can skip this step. In the terminal, type the following commands in sequence and press Enter:

```
sudo apt-get update
```

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

3. Download the BMP3 series driver library. In the terminal, type the following commands in

sequence and press Enter:

```
cd Desktop
```

```
git clone https://github.com/cdjq/DFRobot_BMP3XX
```

Notice: If you choose to use I2C or SPI communication, you need to modify the demo to the corresponding communication. You may encounter a situation where you do not have permission to modify the sample program. The following is the solution:

1. Query permissions under the file directory to be modified, the command is:

```
ls -al
```

2. Modify the file permissions, the command is:

```
sudo chmod a+w XXX.py
```

At this point, everyone has write access to the file.

Sample code

- Sample code 1-Read the measured frequency and data (temperature, air pressure, altitude) (get_temp_press.py)
- Sample code 2-data interrupt function (interrupt_data_drdy.py)
- Sample code 3-FIFO water level or full interrupt function (interrupt_using_FIFO.py)
- Sample code 4-data processing advanced setting function (set_ODR_OSR_IIR.py)

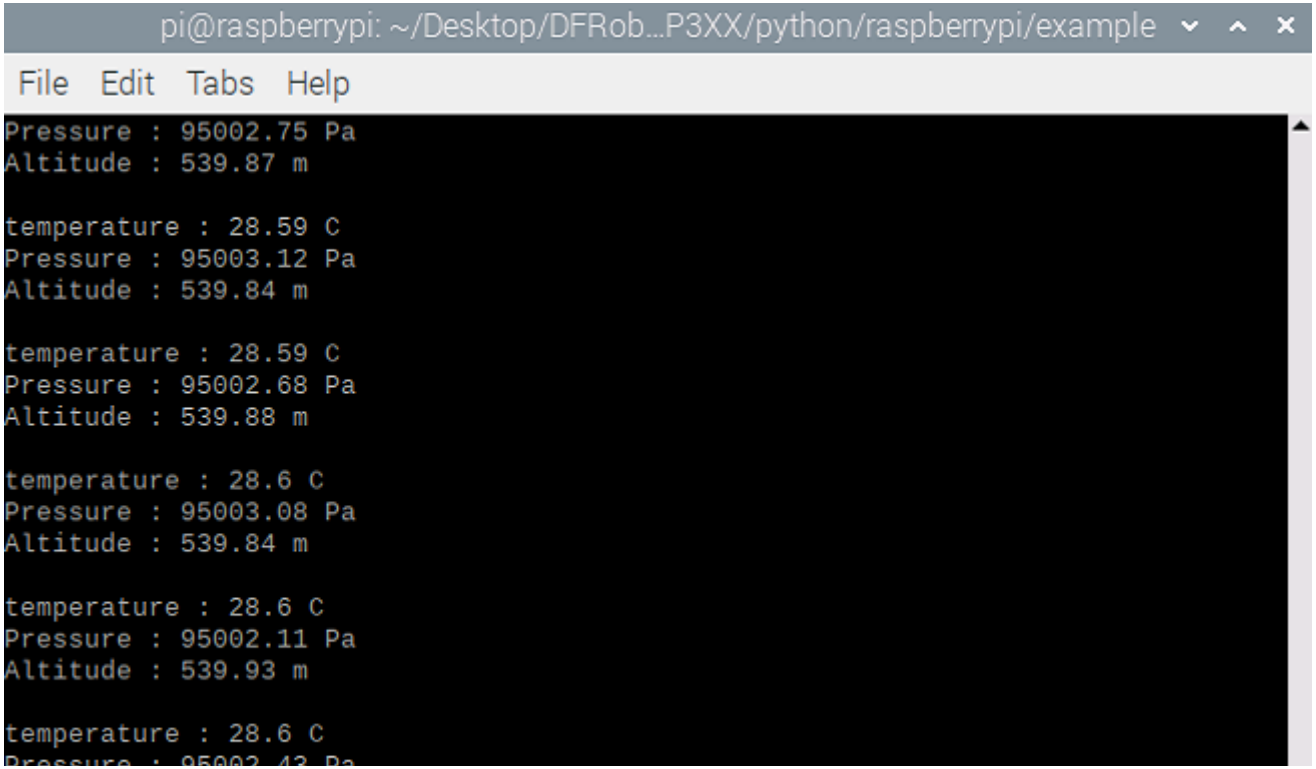
Sample code 1-Read the measured frequency and data (temperature, air pressure, altitude) (get_temp_press.py)

- In the terminal, type the following command and press Enter to run the sample code:

```
cd DFRobot_BMP3XX/python/raspberrypi/example
```

```
python3 get_temp_press.py
```

- Result


pi@raspberrypi: ~/Desktop/DFRob...P3XX/python/raspberrypi/example
File Edit Tabs Help
Pressure : 95002.75 Pa
Altitude : 539.87 m
temperature : 28.59 C
Pressure : 95003.12 Pa
Altitude : 539.84 m
temperature : 28.59 C
Pressure : 95002.68 Pa
Altitude : 539.88 m
temperature : 28.6 C
Pressure : 95003.08 Pa
Altitude : 539.84 m
temperature : 28.6 C
Pressure : 95002.11 Pa
Altitude : 539.93 m
temperature : 28.6 C

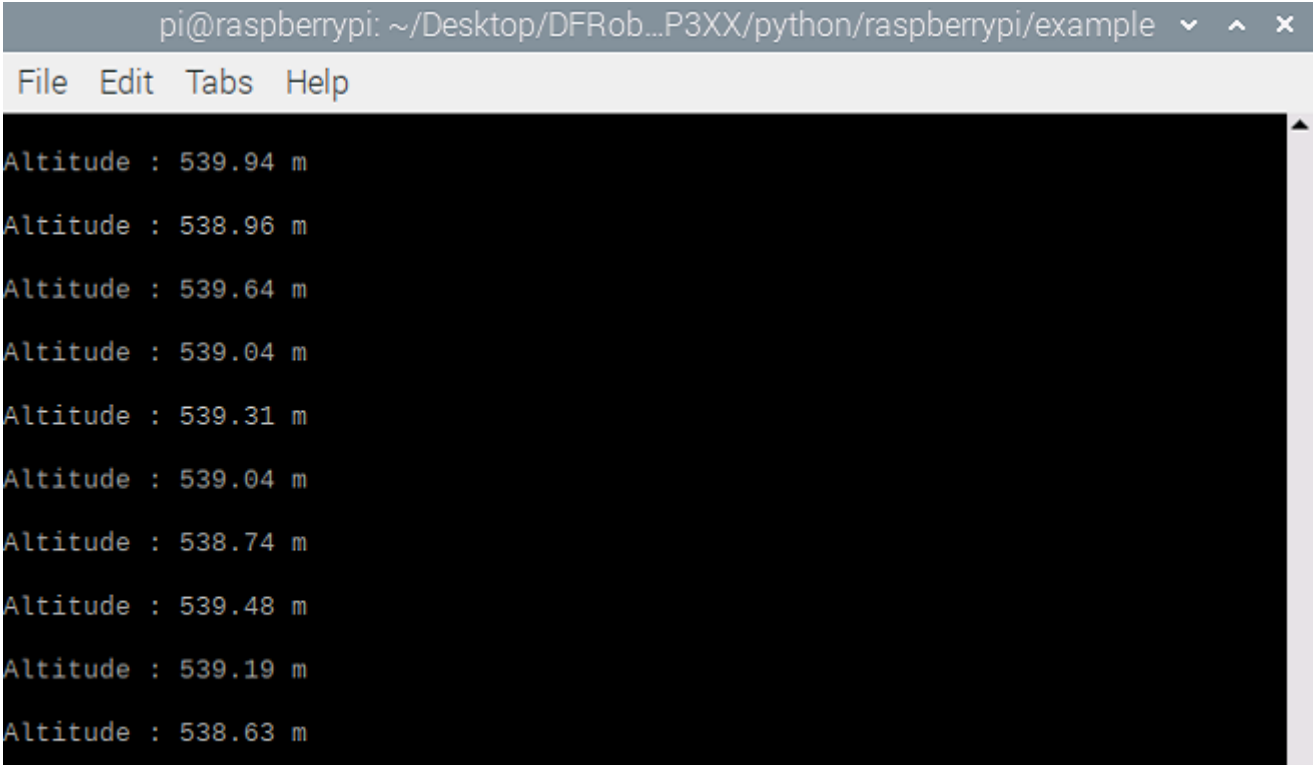
Sample code 2-data interrupt function (interrupt_data_drdy.py)

- In the terminal, type the following command and press Enter to run the sample code:

```
cd DFRobot_BMP3XX/python/raspberrypi/example
```

```
python3 interrupt_data_drdy.py
```

- Result


pi@raspberrypi: ~/Desktop/DFRob...P3XX/python/raspberrypi/example
File Edit Tabs Help
Altitude : 539.94 m
Altitude : 538.96 m
Altitude : 539.64 m
Altitude : 539.04 m
Altitude : 539.31 m
Altitude : 539.04 m
Altitude : 538.74 m
Altitude : 539.48 m
Altitude : 539.19 m
Altitude : 538.63 m

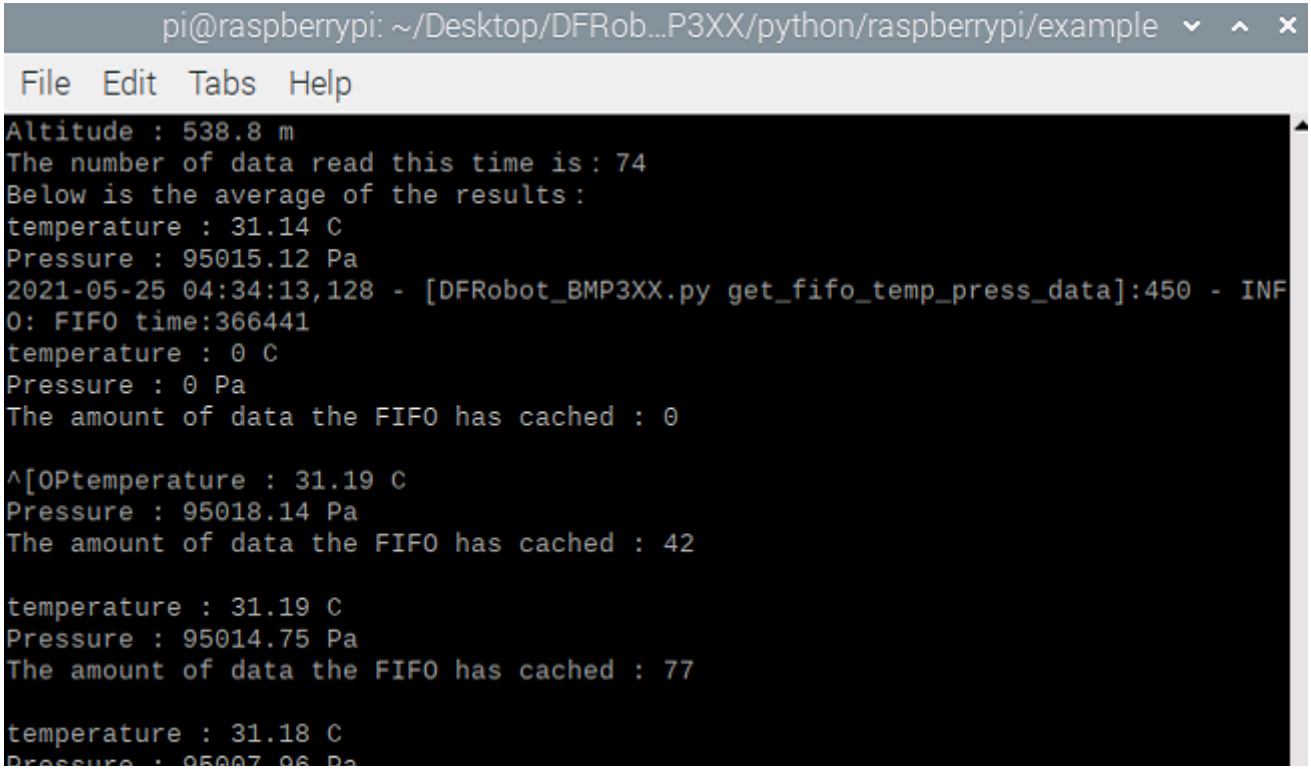
Sample code 3-FIFO water level or full interrupt function (interrupt_using_FIFO.py)

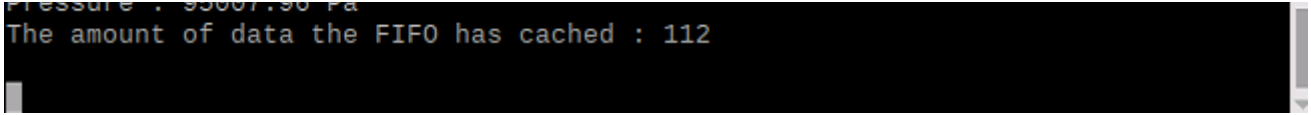
- In the terminal, type the following command and press Enter to run the sample code:

```
cd DFRobot_BMP3XX/python/raspberrypi/example
```

```
python3 interrupt_using_FIFO.py
```

- Result


pi@raspberrypi: ~/Desktop/DFRob...P3XX/python/raspberrypi/example
File Edit Tabs Help
Altitude : 538.8 m
The number of data read this time is : 74
Below is the average of the results :
temperature : 31.14 C
Pressure : 95015.12 Pa
2021-05-25 04:34:13,128 - [DFRobot_BMP3XX.py get_fifo_temp_press_data]:450 - INF
O: FIFO time:366441
temperature : 0 C
Pressure : 0 Pa
The amount of data the FIFO has cached : 0
^[OPtemperature : 31.19 C
Pressure : 95018.14 Pa
The amount of data the FIFO has cached : 42
temperature : 31.19 C
Pressure : 95014.75 Pa
The amount of data the FIFO has cached : 77
temperature : 31.18 C



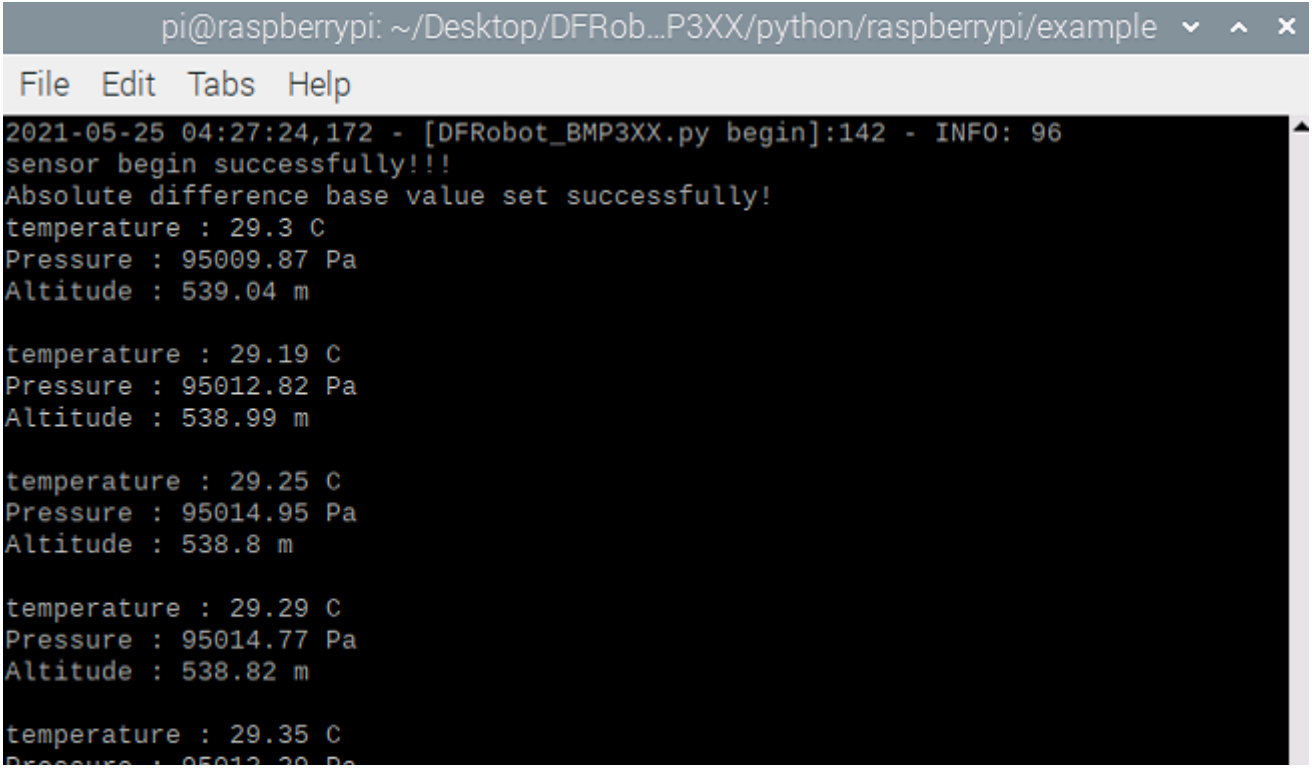
Sample code 4-data processing advanced setting function (set_ODR_OSR_IIR.py)

- In the terminal, type the following command and press Enter to run the sample code:

```
cd DFRobot_BMP3XX/python/raspberrypi/example
```

```
python3 set_ODR_OSR_IIR.py
```

- Result


pi@raspberrypi: ~/Desktop/DFRob...P3XX/python/raspberrypi/example
File Edit Tabs Help
2021-05-25 04:27:24,172 - [DFRobot_BMP3XX.py begin]:142 - INFO: 96
sensor begin successfully!!!
Absolute difference base value set successfully!
temperature : 29.3 C
Pressure : 95009.87 Pa
Altitude : 539.04 m
temperature : 29.19 C
Pressure : 95012.82 Pa
Altitude : 538.99 m
temperature : 29.25 C
Pressure : 95014.95 Pa
Altitude : 538.8 m
temperature : 29.29 C
Pressure : 95014.77 Pa
Altitude : 538.82 m
temperature : 29.35 C

FAQ

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

More Documents

- Schematics Diagram (https://img.dfrobot.com.cn/wiki/5fe933fdbddfc41c32938086/a3387ffa6e2c80f78706ce6c379e4e0d.pdf)
- SEN0423-Dimension.jpg (https://img.dfrobot.com.cn/wiki/5fe933fdbddfc41c32938086/e3a06cb9f5de7466debfbb48bbf10210.jpg)
- SEN0423-Datasheet.pdf (https://img.dfrobot.com.cn/wiki/5fe933fdbddfc41c32938086/b668f27c98f42171d43a7d755

dd66319.pdf)

 Get **Fermion BMP390L Digital Barometric Pressure Sensor** (https://www.dfrobot.com/product-2432.html) from DFRobot Store or **DFRobot Distributor**. (https://www.dfrobot.com/index.php?route=information/distributorslogo)

Turn to the Top