Overview

Introduction

The BME68X Environmental Sensor is a four-in-one environmental sensor that can measure temperature, humidity, barometric pressure, and air quality. It is compact, low power, and suitable for smart homes, mobile application environment monitoring, wearable devices, etc.

BME680 Environmental Sensor

BME688 Environmental Sensor

I2C, SPI

Feature

- Onboard BME68X sensor to measure temperature, humidity, barometric pressure, and gas.
- Supports I2C communication, I2C address configurable, with I2C bus cascading support.
- Supports SPI communication, enabled via CS pin (I2C bus by default).
- Onboard voltage translator, compatible with 3.3V/5V level.
- Comes with online development resources and manual (examples for Raspberry Pi / Raspberry Pi Pico / Arduino / ESP32).

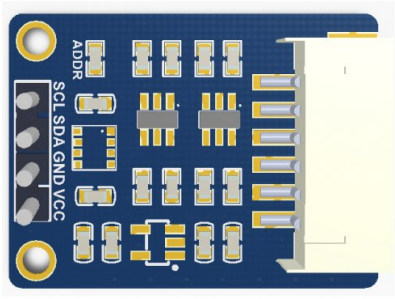
Specifications

Model	BME280	BME680	BME688
Function	Barometric pressure, Environmental temperature, Relative humidity	Barometric pressure, Environmental temperature, Relative humidity, VOC gas change detection (supports IAQ calculation in combination with the software package)	Similar to BME680, Suitable for detecting various additional gases (such as VSC, carbon monoxide, hydrogen, etc.) Multiple gas discrimination Artificial intelligence (requires secondary development by the user)
Communication Interface	I2C and SPI		
Temperature Measuring Range	-40~85°C		
Temperature Measuring Accuracy	±1.0℃ (0~65℃)		±0.5℃ (0~65℃)
Humidity Measuring Range	0~100% r.H.		
Humidity Measuring Accuracy	±3% r.H.		
Barometric Pressure Measurement Range	300~1100 hPa		
Barometric			

Pressure Measurement Accuracy	±1.0hPa (0~65℃)	±0.6hPa (0~65℃)
IAQ Measuring Range	Not support	0~500 IAQ (The sensor outputs changes in resistance due to VOC gas, and the Bosch BSEC library is required to output IAQ.)
Dimensions	27mm × 20mm	

The BME680 and BME688 sensors contain a mini MOX sensor. The heated metal oxide changes its resistance according to the concentration of volatile organic compounds (VOC) in the air, making it capable of detecting gases and alcohols such as ethanol, alcohol, and carbon monoxide, and measuring air quality. It provides a resistance value (Gas resistance in the figure), which represents the total VOC content, but cannot differentiate between different gases or alcohols. To convert this value to an IAQ air quality index, it is necessary to use the official BSEC software library (which is not open-source). Bosch imposes certain restrictions and licensing requirements on the use of this software library, and users are advised to study the details of its use and integration according to their specific needs.

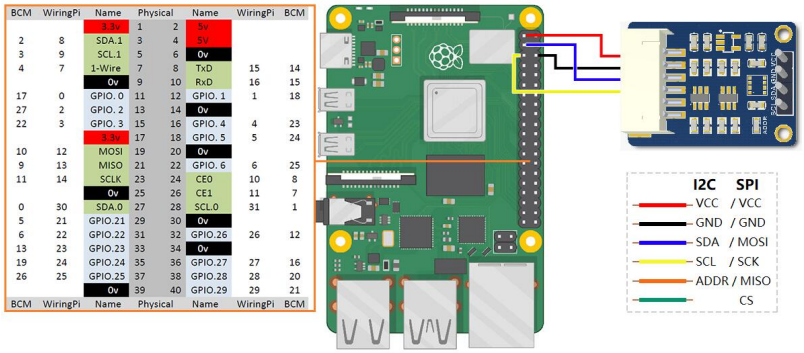
Interface Definition



I2C		SPI	
Pins	Description	Pins	Description
VCC	Power Input	VCC	Power Input
GND	Ground	GND	Ground
SDA	Data Pin	MOSI	SPI Data Input
SCL	I2C Clock Pin	SCK	SPI Clock Input
ADDR	Address chip selection (high level by default): high level, the address is 0x77 low level, the address is 0x76	MISO	SPI data output
CS	NC	CS	SPI chip selection, low active

Working with Raspberry Pi

Hardware Connection



The above figure is connected to the I2C interface as an example as a demonstration, where the ADDR pin can be used to set the I2C address of the sensor, the default non-connected I2C address is 0x77, if the ADDR is connected to GND, the I2C address is 0x76. If you want to connect Raspberry Pi through the SPI interface for communication, please refer to the following table for connection.

I2C		SPI	
Pins	Raspberry Pin	Pins	Raspberry Pin
VCC	3.3V /5V	VCC	3.3V /5V
GND	GND	GND	GND
SDA	SDA.1	MOSI	MOSI
SCL	SCL.1	SCK	SCLK
ADDR	NC/GND	MISO	MISO
CS	NC	CS	27(wiringPi)

Software Config

Enable I2C/SPI Interface

- Execute the following commands to configure the Raspberry Pi:

```
sudo raspi-config
```

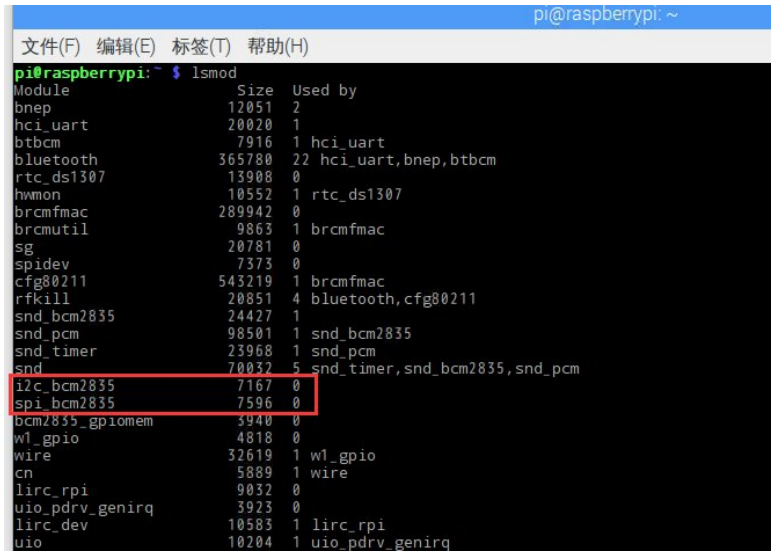
- Choose Interfacing Options -> I2C -> yes to enable I2C kernel driver.
- Choose Interfacing Options -> SPI -> yes to enable SPI kernel driver.
- Save, exit, and then reboot the Raspberry Pi:

```
sudo reboot
```

- After rebooting, run the commands to view. Check whether the I2C and SPI modules are enabled.

```
lsmod
```

- The following print message will be available.



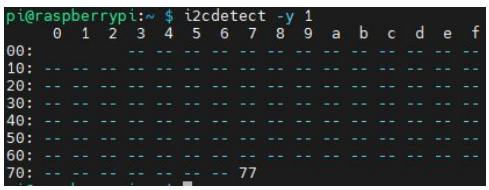
- If i2c_bcm2835 and spi_bcm2835 are displayed then the I2C, SPI module is booted.
- Connect the BME68x module to the Raspberry Pi as described in the previous I2C bus interface instructions.
- The default I2C device address of the BME68x module is 0x77, if ADDR is grounded, the device address will be changed to 0x76.
- Install the i2c-tools tool to confirm.

```
sudo apt-get install i2c-tools
```

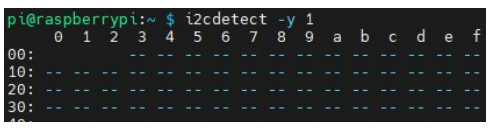
- Query connected I2C devices

```
i2cdetect -y 1
```

- The following message will be printed.



- If 77 is displayed then the BME68x module is successfully connected to the Raspberry Pi successfully.
- If the ADDR is connected to GND then 76 is printed.



```
40: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
50: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
60: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
70: -- -- -- -- -- -- 76 --
```

Note: The above test ensures that there are no devices on the I2C bus that have the same address as the device. If the above test is successful, the I2C module is loaded successfully, and the BME68x module is successfully connected to the Raspberry Pi. In addition, the BME68x module supports the SPI driver, and you can refer to the SPI interface description section to connect the BME68x to the Raspberry Pi.

Download Example Demo

- Download the example demo, decompress, and modify the file permissions.

```
cd ~
wget https://files.waveshare.com/upload/4/49/BME68X_Environmental_Sensor_code.zip
unzip BME68X_Environmental_Sensor_code.zip
sudo chmod -R 777 BME68X_Environmental_Sensor_code
```

C

Demo

- After connecting the hardware as shown above and configuring the software properly.
- If I2C driver is used: first determine the I2C device address, BME68x module default I2C device address is 0x77, if the ADDR pin is grounded (or short the pad marked `ADDR` silkscreen on the PCB), then its I2C device address changes to 0x76.
- Enter `BME68X_Environmental_Sensor_code/RaspberryPi/C`:

```
cd BME68X_Environmental_Sensor_code/RaspberryPi/C
```

- Open main.c file:

```
nano main.c
```

- Make sure the USEIIC macro in main.c is defined as 1 to adopt the I2C driver.

```
19  //Default write it to the register in one time
20  #define USESPISINGLEREADWRITE 0
21
22  //This definition you use I2C or SPI to drive the bme68x
23  //When it is 1 means use I2C interface, When it is 0,use SPI interface
24  #define USEIIC 1
25
26  #define BME68X_VALID_DATA UINT8_C(0xB0)
```

- Also check the I2C device address in main.c to make sure it is the same as the current BME68x module device address (default I2C device address is 0x77 (BME68X_I2C_ADDR_HIGH). If ADDR is grounded then its device address is 0x76 (BME68X_I2C_ADDR_HIGH)).

```
235 #if(USEIIC)
236 int main(int argc, char* argv[])
237 {
238     struct bme68x_dev dev;
239     static uint8_t dev_addr=BME68X_I2C_ADDR_HIGH;
240     int8_t rslt = BME68X_OK;
241
242     if ((fd = open(IIC_Dev, O_RDWR)) < 0) {
243         printf("Failed to open the i2c bus %s", argv[1]);
244         exit(1);
245     }
246     if (ioctl(fd, I2C_SLAVE, dev_addr) < 0) {
247         printf("Failed to acquire bus access and/or talk to slave.\n");
248         exit(1);
249     }
250     //dev.dev_id = BME68X_I2C_ADDR_PRIM;//0x76
251     dev.intf_ptr = &dev_addr; //0x77
252     dev.intf = BME68X_I2C_INTF;
253     dev.read = user_i2c_read;
254     dev.write = user_i2c_write;
255     dev.delay_us = user_delay_us;
```

- If SPI driver is used: wire the BME68x module according to the SPI bus wiring in the interface description and change the USEIIC macro definition in the main.c file to 0.

```
19  //Default write it to the register in one time
20  #define USESPISINGLEREADWRITE 0
```

```
21
22  //This definition you use I2C or SPI to drive the bme68x
23  //When it is 1 means use I2C interface, When it is 0,use SPI interface
24  #define USEIIC 0
25
26  #define BME68X_VALID_DATA  UINT8_C(0xB0)
```

- Save and exit the editor, then recompile.

```
sudo make clean
sudo make
```

- Run:

```
sudo ./bme68x
```

- The following data will be displayed.

```
pi@raspberrypi:~/Raspberry $ sudo ./bme68x

 BME68X Init Result is:0
Temperature          Pressure             Humidity             Gas resistance
temperature:25.06*C   pressure:1022.93hPa   humidity:35.91%   Gas resistance:143085.09 ohmm
```

- From left to right, the temperature (°C), barometric pressure (hPa), relative humidity (%RH), and gas resistance (ohms) measured by the BME68x are displayed. If the data is not displayed successfully, or if the data is not displayed properly, please check the connection, communication method, and device address for errors.

Python

- Python demo only has I2C mode.

Install Function Library

```
sudo pip3 install bme680
```

Demo

- Enter the example demo file:

```
cd BME68X_Environmental_Sensor_code/RaspberryPi/Python/examples
```

- Run the demo:

```
sudo python3 read-all.py
```

- The demo will print a series of module information, from left to right, the temperature (°C), barometric pressure (hPa), relative humidity (%RH), and gas resistance (ohms) measured by the BME68x are displayed. If the data is not displayed successfully, or if the data is not displayed properly, please check the connection, the communication method, and the device address for errors.

```
pi@uuuuuu:~/bme680-python-master/bme680-python-master/examples $ sudo python read-all.py
read-all.py - Displays temperature, pressure, humidity, and gas.

Press Ctrl+C to exit!


Calibration data:
par_gh1: -45
par_gh2: -18303
par_gh3: 18
par_h1: 709
par_h2: 1023
par_h3: 0
par_h4: 45
par_h5: 20
par_h6: 120
par_h7: -100
par_p1: 36169
par_p10: 30
par_p2: -10398
par_p3: 88
par_p4: 7877
par_p5: -172
par_p6: 30
par_p7: 42
par_p8: -2607
par_p9: -2613
par_t1: 26058
par_t2: 26363
par_t3: 3
range_sw_err: 13
res_heat_range: 1
res_heat_val: 33
t_fine: 129387


Initial reading:
gas_index: 0
gas_resistance: 55858512.678457275
```

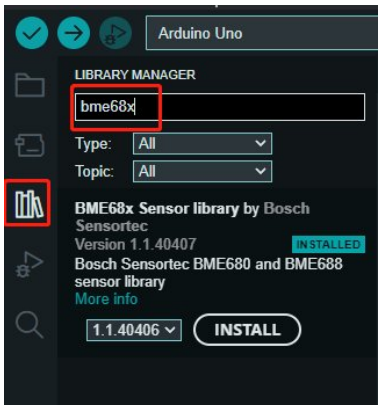
```
heat_stable: False
humidity: 64.188
meas_index: 0
pressure: 1010.66
status: 32
temperature: 25.27


Polling:
25.28 C,1010.66 hPa,64.19 %RH
25.30 C,1010.69 hPa,64.06 %RH,32837.35248845562 Ohms
25.34 C,1010.68 hPa,63.84 %RH,51990.25182778229 Ohms
25.39 C,1010.70 hPa,63.62 %RH,68577.55156710421 Ohms
25.42 C,1010.69 hPa,63.39 %RH,82500.80567193039 Ohms
25.46 C,1010.70 hPa,63.18 %RH,94955.48961424333 Ohms
25.49 C,1010.69 hPa,63.04 %RH,103790.79667545104 Ohms
25.51 C,1010.67 hPa,62.91 %RH,110870.50671286273 Ohms
25.53 C,1010.67 hPa,62.82 %RH,116948.37825491093 Ohms
25.55 C,1010.67 hPa,62.75 %RH,122224.87467175937 Ohms
25.56 C,1010.71 hPa,62.66 %RH,127141.79289793893 Ohms
```

Working with Arduino

Install Library

The library for the BME68x sensor can be downloaded from the library manager of the Arduino IDE:

- Open Arduino IDE 2.0.
- Open the "Library Manager" option in the left toolbar and search for BME68x.



Hardware Connection

I2C Interface		SPI Interface	
Pins	Arduino Pin	Pins	Arduino Pin
VCC	3.3V /5V	VCC	3.3V /5V
GND	GND	GND	GND
SDA	SDA	MOSI	D11
SCL	SCL	SCK	D13
ADDR	NC/GND	MISO	D12
CS	NC	CS	D10

Demo

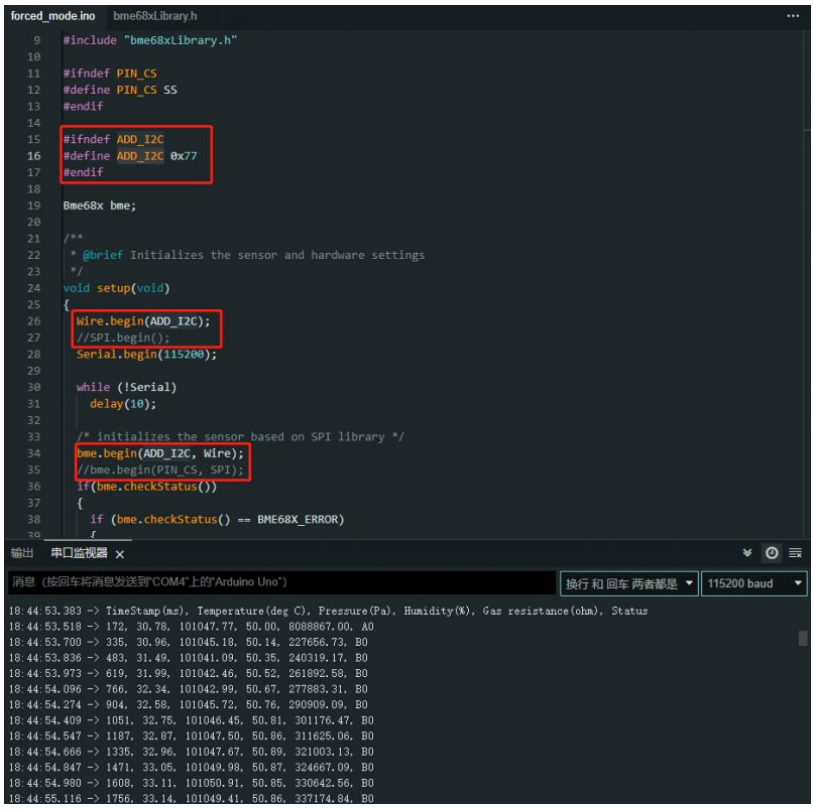
SPI

- The default communication method of this demo is SPI, refer to the table above to connect the module to the development board (this demo uses Arduino Uno).
- Click File -> examples -> BME68x Sensor library -> forced_mode to open the sample demo.
- Connect the development board to the computer (this demo uses Arduino uno), click Tools->Development Board, select the corresponding development board, click: Tools->Port select the corresponding port.
- Click on the upload button to compile and upload the demo to see the development board and wait for a successful upload.
- Click on Tools -> Serial Monitor, which shows from left to right the temperature (°C), barometric pressure (hPa), relative humidity (%RH), altitude (m), and gas resistance (ohms) measured by the BME68x sensor.

- If the data is not displayed successfully, or if the data is not displayed normally, please check the connection, communication method, and device address for errors.

I2C

- If you want to change the communication way to I2C, you should modify the hardware connection according to the I2C.
- Modify the main demo according to the following figure.
- Compile and upload the demo, and open SSCOM. From left to right, the temperature (°C), barometric pressure (hPa), relative humidity (%RH), altitude (m), and gas resistance (ohms) measured by the BME68x sensor are shown.

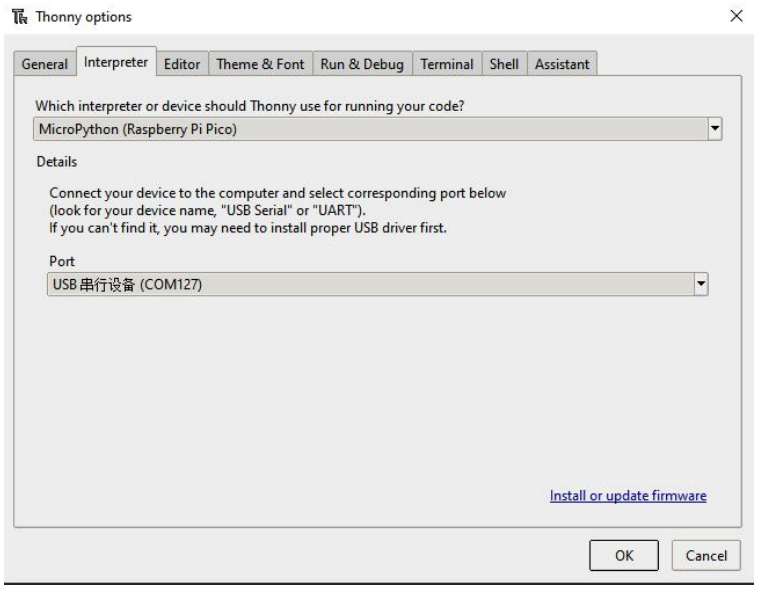


Working with Raspberry Pi Pico

Set up Environment

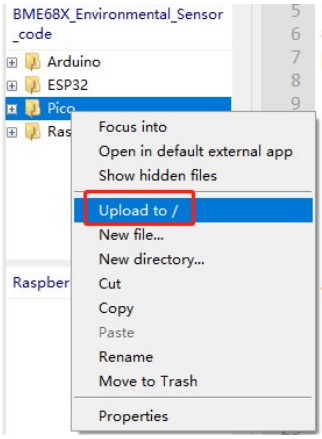
This tutorial uses Thonny for code testing, click to download the relevant IDE and install it, then open Thonny.

- Please refer to the official documentation to set up the python environment, in Thonny: Tools -> Options -> Interprete select the Raspberry Pi Pico device, as shown in the following figure:



Download the Demo

1. Download the demo.
2. Unzip the sample demo.
3. Open Thonny, and check whether it is connected to the pico. Then, open the unpacked demo path in the upper left corner, right-click on the pico folder, and select Upload, as shown in the picture.

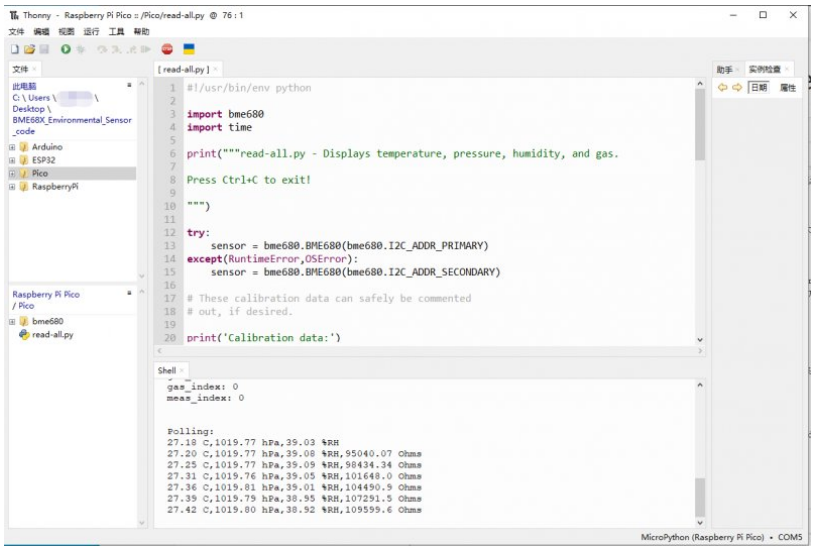


Hardware Connection

I2C Interface	
Pins	Pico Pin
VCC	3.3V /5V
GND	GND
SDA	GP6
SCL	GP7
ADDR	NC/GND
CS	NC

Demo

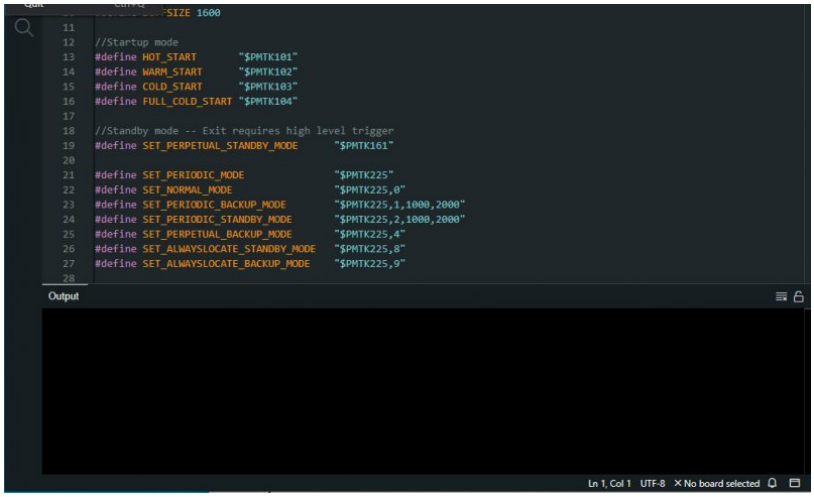
1. Open Thonny IDE, choose the pico directory, and double-click to open the read-all.py file. The demo is shown below:



Working with ESP32

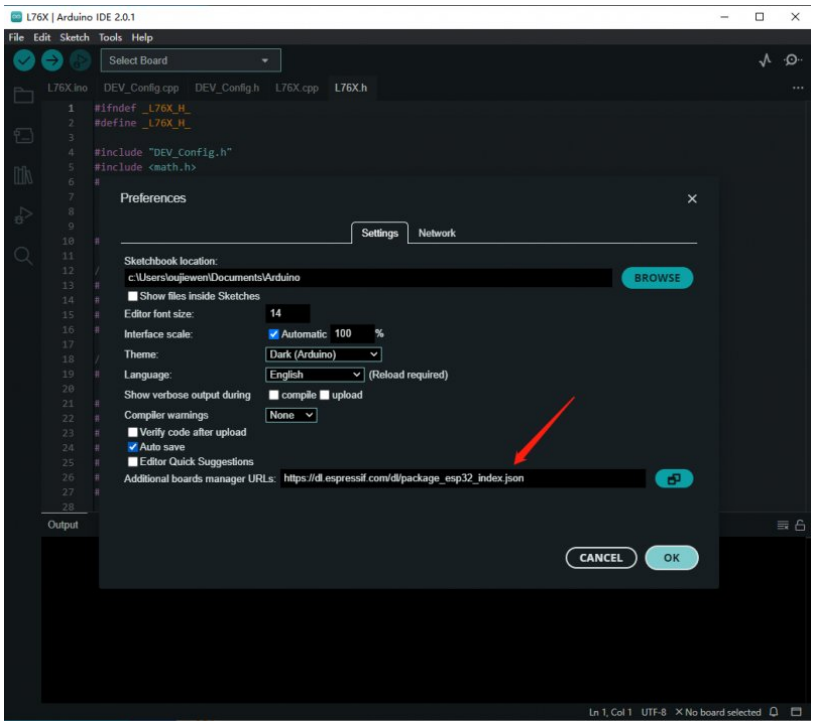
Install ESP32 Plug-in in Arduino IDE

1. Open Arduino IDE, click "File" at the upper left corner, and choose "Preferences".



2. Add the following link to the Additional Development Board Manager URL and click OK.

```
https://dl.espressif.com/dl/package_esp32_index.json
```



Note: If you already have the ESP8266 board URL, you can separate the URLs with commas like this:

```
https://dl.espressif.com/dl/package_esp32_index.json, http://arduino.esp8266.com/stable/package_esp8266com_index.json
```

3. Download the package and copy the packages file to the following path:

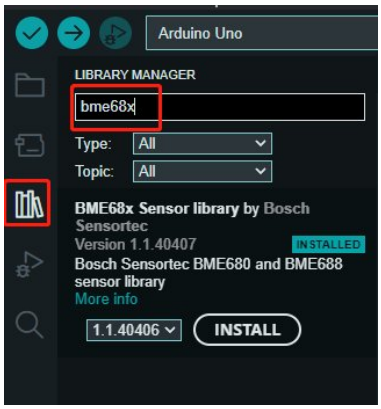
```
C:\Users\xutong\AppData\Local\Arduino15
```

Note: Replace the username: xutong with your own username.

Install Library

The library for the BME68x sensor can be downloaded from the library manager of the Arduino IDE:

- Open Arduino IDE 2.0.
- Open the "Library Manager" option in the left toolbar and search for BME68x.



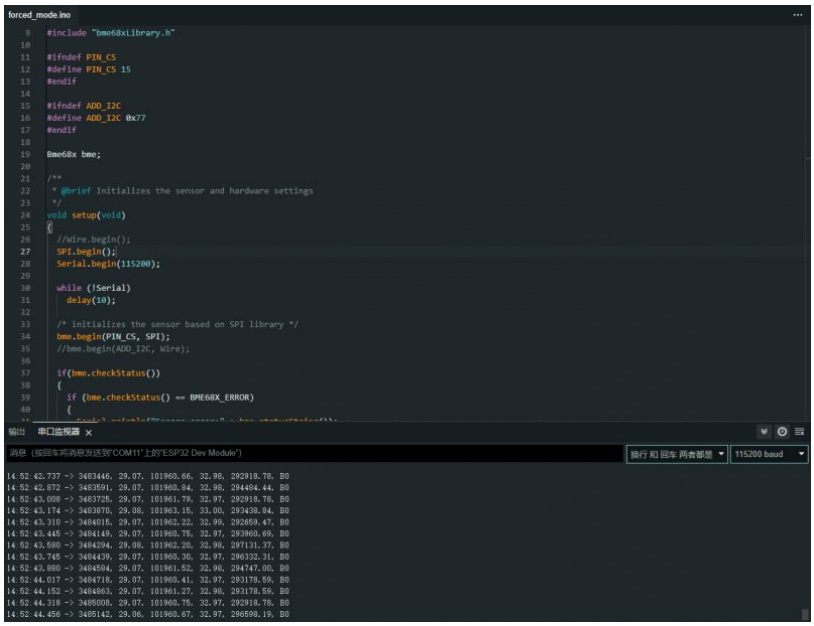
Hardware Connection

I2C Interface		SPI Interface	
Pins	ESP32 Pin	Pins	ESP32 Pin
VCC	3.3V /5V	VCC	3.3V /5V
GND	GND	GND	GND
SDA	P21	MOSI	P23
SCL	P22	SCK	P18
ADDR	NC/GND	MISO	P19
CS	NC	CS	P15

Demo

SPI

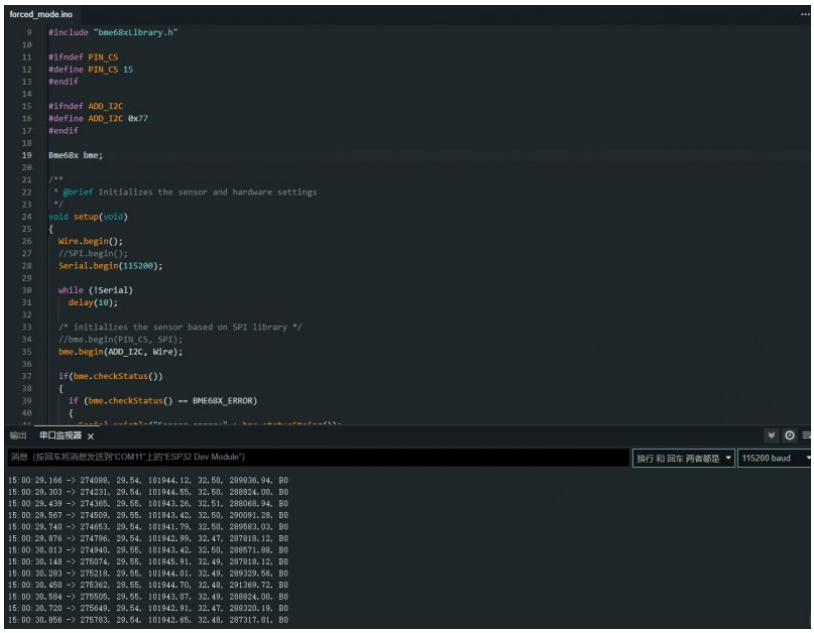
- The default communication method of this demo is SPI, refer to the table above to connect the module to the development board.
- Click on: File -> Examples -> BME68x Sensor library -> forced_mode to open the sample demo.
- Connect the development board to the computer, click Tools->Development Board, select the corresponding development board, and click: Tools -> Port to select the corresponding port.
- Click the upload button to compile and upload the demo to the watch development board and wait for a successful upload.
- Click on Tools -> Serial Monitor, which shows from left to right the temperature (°C), barometric pressure (hPa), relative humidity (%RH), altitude (m), and gas resistance (ohms) measured by the BME68x sensor.



- If the data is not displayed successfully, or if the data is not displayed properly, please check the connection, communication method, and device address for errors.

I2C

- If you need to modify the communication mode to I2C, first modify the hardware connection according to the I2C mode.
- Refer to the following diagram, and modify the original main demo;
- Compile and upload the demo, open the serial monitor, which from left to right shows the temperature (°C), barometric pressure (hPa), relative humidity (%RH), altitude (m), and gas resistance (ohms) measured by the BME68x sensor.



Resource

Document

- Schematic

Demo

- Example demo

Software

- Arduino IDE
- SSCOM Serial Assistant

Related Resource

- BME680 Datasheet
- BME688 Datasheet

Support

Technical Support

If you need technical support or have any feedback/review, please click the **Submit Now** button to submit a ticket, Our support team will check and reply to you within 1 to 2 working days. Please be patient as we make every effort to help you to resolve the issue.
Working Time: 9 AM - 6 AM GMT+8 (Monday to Friday)

Submit Now