

Overview

- Introduction
- Feature
- Specifications
- Interface
- Definition
- Working with Raspberry Pi
 - Hardware
 - Connection
- Software Config
 - Enable I2C/SPI
 - Interface
- Download
- Example Demo
 - C
 - Demo
- Python
 - Install
 - Function
 - Library
 - Demo
- Working with Arduino
 - Install Library
 - Hardware
 - Connection
 - Demo
 - SPI
 - I2C
- Working with Raspberry Pi Pico
 - Set up
 - Environment
 - Download the Demo
 - Hardware
 - Connection
 - Demo
- Working with ESP32
 - Install ESP32
 - Plug-in in Arduino IDE
 - Install Library
 - Hardware
 - Connection
 - Demo
 - SPI
 - I2C
- Resource
 - Document
 - Demo
 - Software
 - Related Resource

Support

[To Top](#)

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.

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°C (0~65°C) | | ±0.5°C (0~65°C) |
| Humidity Measuring Range | 0~100% r.H. | | |
| Humidity Measuring Accuracy | ±3% r.H. | | |
| Barometric Pressure Measurement Range | 300~1100 hPa | | |
| Barometric | | | |

BME680 Environmental Sensor



BME688 Environmental Sensor

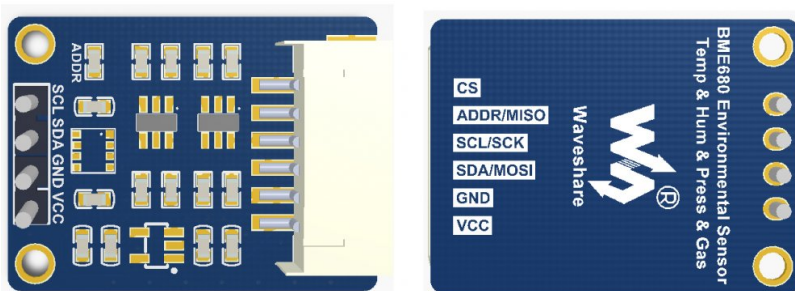


I2C, SPI

| | | |
|-------------------------------|------------------|---|
| Pressure Measurement Accuracy | ±1.0hPa (0~65°C) | ±0.6hPa (0~65°C) |
| 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.

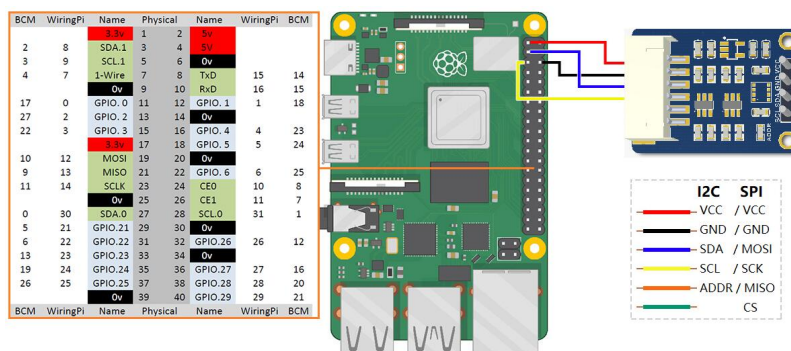
Interface Definition



| 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
```

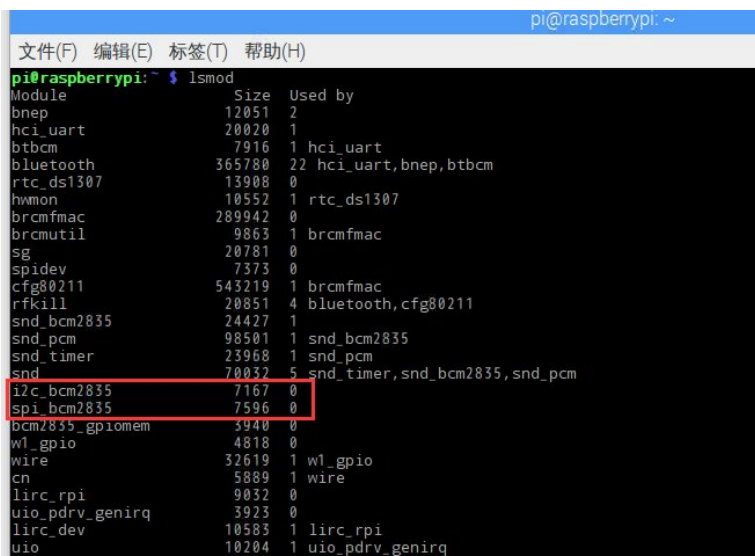
- Choose Interfacing Options -> I2C -> yes to enable I2C kernel driver.
- Choose Interfacing Options -> SPI -> yes to enable SPI kernel driver.
- Save, exit, and then reboot the Raspberry Pi:

```
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.



```
pi@raspberrypi: ~  
文件(F) 编辑(E) 标签(T) 帮助(H)  
pi@raspberrypi:~$ lsmod  
Module                Size  Used by  
bnep                   12051  2  
hci_uart               20020  1  
btbcm                   7916  1 hci_uart  
bluetooth              365780  22 hci_uart, bnep, btbcm  
rtc_ds1307              13908  0  
hwmmon                 10552  1 rtc_ds1307  
brcmfmac               289942  0  
brcmutil                9863  1 brcmfmac  
sg                      20781  0  
spidev                  7373  0  
cfg80211               543219  1 brcmfmac  
rfkill                 20851  4 bluetooth, cfg80211  
snd_bcm2835            24427  1  
snd_pcm                 98501  1 snd_bcm2835  
snd_timer              23968  1 snd_pcm  
snd                     70032  5 snd_timer, snd_bcm2835, snd_pcm  
i2c_bcm2835            7167  0  
spi_bcm2835            7596  0  
bcm2835_gpiomem        3940  0  
w1_gpio                 4818  0  
wire                    32619  1 w1_gpio  
cn                      5889  1 wire  
lirc_rpi                9032  0  
uio_pdrv_genirq        3923  0  
lirc_dev                10583  1 lirc_rpi  
uio                     10204  1 uio_pdrv_genirq
```

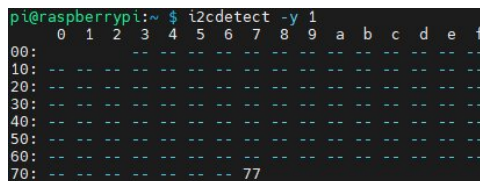
- If `i2c_bcm2835` and `spi_bcm2835` are displayed then the I2C, SPI module is booted.
- Connect the BME68x module to the Raspberry Pi as described in the previous I2C bus interface instructions.
- The default I2C device address of the BME68x module is 0x77, if ADDR is grounded, the device address will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

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

- Query connected I2C devices

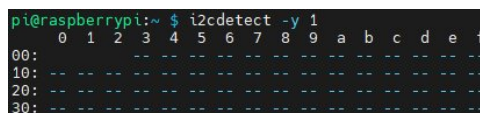
```
i2cdetect -y 1
```

- The following message will be printed.



```
pi@raspberrypi:~$ i2cdetect -y 1  
00:  0  1  2  3  4  5  6  7  8  9  a  b  c  d  e  f  
10:  -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --  
20:  -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --  
30:  -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --  
40:  -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --  
50:  -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --  
60:  -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --  
70:  -- -- -- -- -- -- -- 77
```

- 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.



```
pi@raspberrypi:~$ i2cdetect -y 1  
00:  0  1  2  3  4  5  6  7  8  9  a  b  c  d  e  f  
10:  -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --  
20:  -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --  
30:  -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
```

```
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 USE_IIC 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 ohm

```

- 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: -10303
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,116948.37825401093 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

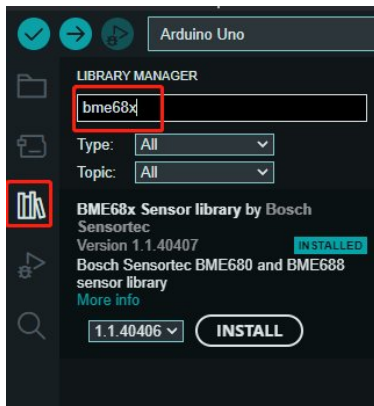
```

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.

The screenshot shows the Serial Monitor window in the Arduino IDE. The output text is as follows:

```

18:41:02.180 -> TimeSetup (ms), Temperature(deg C), Pressure(Pa), Humidity(%), Gas resistance(ohm), Status
18:41:02.281 -> 178, 25.62, 101050.14, 63.99, 365116.00, 00
18:41:02.416 -> 322, 23.91, 101054.02, 54.17, 256770.31, 00
18:41:02.551 -> 489, 34.34, 101050.65, 54.38, 289892.47, 00
18:41:02.674 -> 612, 26.94, 101049.99, 54.60, 291550.72, 00
18:41:02.838 -> 745, 35.19, 101051.05, 54.72, 299327.69, 00
18:41:02.974 -> 889, 35.42, 101051.35, 54.79, 314236.81, 00
18:41:03.097 -> 1026, 35.59, 101050.43, 54.96, 328115.66, 00
18:41:03.260 -> 1168, 35.72, 101050.23, 54.97, 339185.16, 00
18:41:03.395 -> 1312, 36.81, 101050.47, 54.80, 347826.09, 00
18:41:03.561 -> 1457, 35.36, 101050.40, 54.78, 356432.14, 00
18:41:03.681 -> 1602, 35.93, 101057.58, 54.76, 361454.23, 00
18:41:03.818 -> 1736, 35.90, 101050.66, 54.67, 368106.53, 00

```


- 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.

```

forced_mode_ino  bme68xLibrary.h
9   #include "bme68xLibrary.h"
10
11  #ifndef PIN_CS
12  #define PIN_CS SS
13  #endif
14
15  #ifndef ADD_I2C
16  #define ADD_I2C 0x77
17  #endif
18
19  Bme68x bme;
20
21  /**
22   * @brief Initializes the sensor and hardware settings
23   */
24  void setup(void)
25  {
26    Wire.begin(ADD_I2C);
27    //SPI.begin();
28    Serial.begin(115200);
29
30    while (!Serial)
31      delay(10);
32
33    /* Initializes the sensor based on SPI library */
34    bme.begin(ADD_I2C, Wire);
35    //bme.begin(PIN_CS, SPI);
36    if (bme.checkStatus())
37    {
38      if (bme.checkStatus() == BME68X_ERROR)
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000

```

串口监视器 x

消息 (按回车将消息发送到"COM4"上的"Arduino Uno") 换行和回车两者都是 115200 baud

```

18:44:53.383 -> TimeStamp(ms), Temperature(deg C), Pressure(Pa), Humidity(%), Gas resistance(ohm), Status
18:44:53.518 -> 172, 30.78, 101047.77, 50.00, 8088867.00, AO
18:44:53.700 -> 335, 30.96, 101046.18, 50.14, 227656.73, BO
18:44:53.836 -> 483, 31.49, 101041.09, 50.35, 240319.17, BO
18:44:53.973 -> 619, 31.99, 101042.46, 50.52, 261892.58, BO
18:44:54.096 -> 766, 32.34, 101042.99, 50.67, 277882.31, BO
18:44:54.274 -> 904, 32.58, 101045.72, 50.76, 290909.09, BO
18:44:54.409 -> 1051, 32.75, 101046.45, 50.81, 301176.47, BO
18:44:54.547 -> 1187, 32.87, 101047.50, 50.86, 311625.06, BO
18:44:54.666 -> 1335, 32.96, 101047.67, 50.89, 321003.13, BO
18:44:54.847 -> 1471, 33.05, 101049.98, 50.87, 324667.09, BO
18:44:54.980 -> 1608, 33.11, 101050.91, 50.85, 330642.56, BO
18:44:55.116 -> 1756, 33.14, 101049.41, 50.86, 337174.84, BO

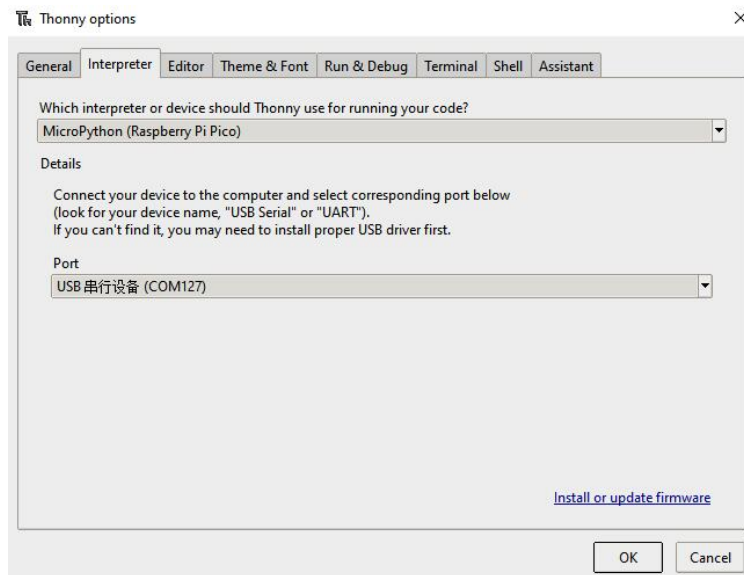
```

Working with Raspberry Pi Pico

Set up Environment

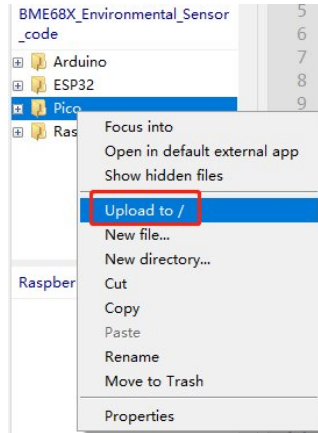
This tutorial uses [Thonny](#) for code testing, click to download the relevant IDE and install it, then open Thonny.

- Please refer to the [official documentation](#) to set up the python environment, in Thonny: Tools -> Options -> Interpreter 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.

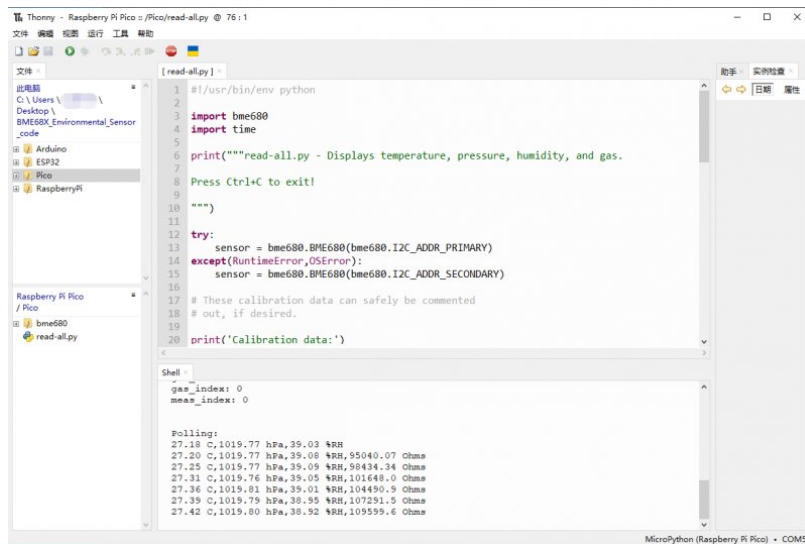


Hardware Connection

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

Demo

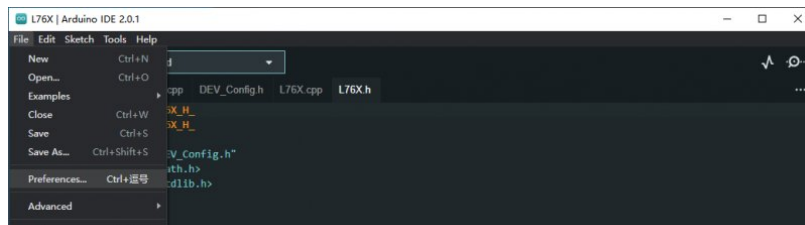
1. Open Thonny IDE, choose the pico directory, and double-click to open the read-all.py file. The demo is shown below:



Working with ESP32

Install ESP32 Plug-in in Arduino IDE

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



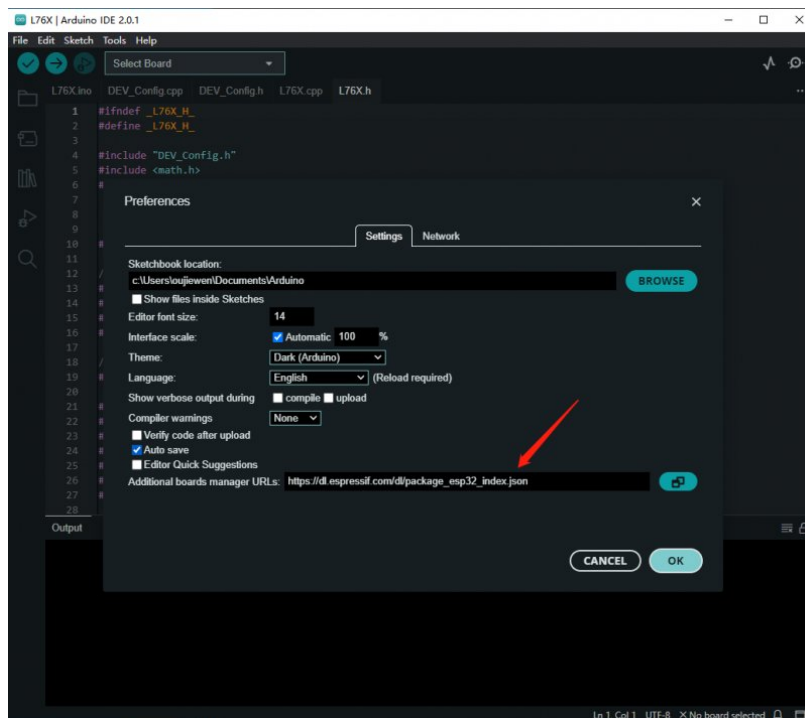

```

11 //Startup mode
12 //Startup mode
13 #define HOT_START "$PMTK101"
14 #define WARM_START "$PMTK102"
15 #define COLD_START "$PMTK103"
16 #define FULL_COLD_START "$PMTK104"
17
18 //Standby mode -- Exit requires high level trigger
19 #define SET_PERPETUAL_STANDBY_MODE "$PMTK161"
20
21 #define SET_PERIODIC_MODE "$PMTK225"
22 #define SET_NORMAL_MODE "$PMTK225,0"
23 #define SET_PERIODIC_BACKUP_MODE "$PMTK225,1,1000,2000"
24 #define SET_PERIODIC_STANDBY_MODE "$PMTK225,2,1000,2000"
25 #define SET_PERPETUAL_BACKUP_MODE "$PMTK225,4"
26 #define SET_ALWAYSLOCATE_STANDBY_MODE "$PMTK225,8"
27 #define SET_ALWAYSLOCATE_BACKUP_MODE "$PMTK225,9"
28

```

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

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

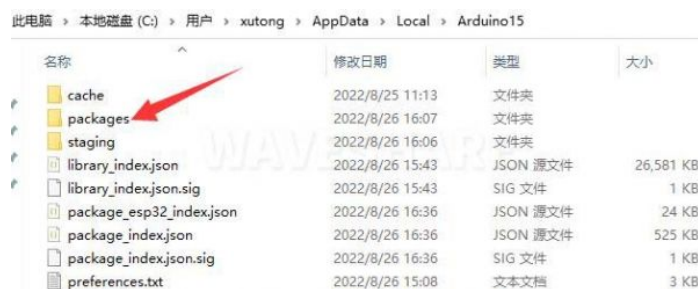


Note: If you already have the ESP8266 board URL, you can separate the URLs with commas like this:

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

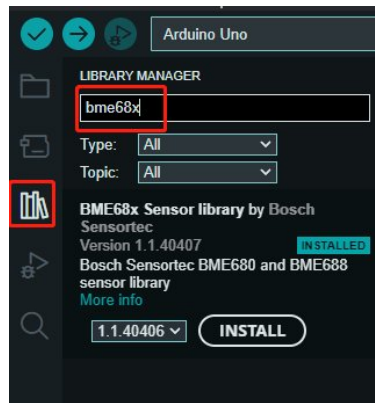


Note: Replace the username: xutong with your own username.

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.

```

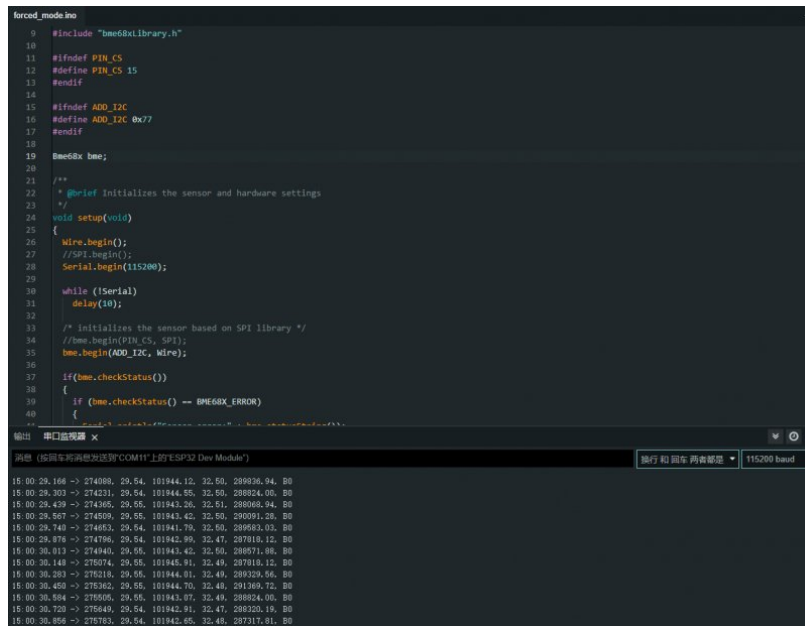
forced_mode.ino
9 #include "bme68xlibrary.h"
10
11 #ifndef PIN_CS
12 #define PIN_CS 15
13 #endif
14
15 #ifndef ADDR_I2C
16 #define ADDR_I2C 0x77
17 #endif
18
19 Bme68x bme;
20
21 /**
22  * @brief Initializes the sensor and hardware settings
23  */
24 void setup(void)
25 {
26     //Wire.begin();
27     SPI.begin();
28     Serial.begin(115200);
29
30     while (!Serial)
31         delay(10);
32
33     /* Initializes the sensor based on SPI library */
34     bme.begin(PIN_CS, SPI);
35     //bme.begin(ADDR_I2C, Wire);
36
37     if(bme.checkStatus())
38     {
39         if (bme.checkStatus() == BME68X_ERROR)
40         {
41             //Serial.println("BME68X Error");
42         }
43     }
44 }
45
46 void loop()
47 {
48     //Serial.println("BME68X Data");
49     //Serial.println(bme.getTemperature());
50     //Serial.println(bme.getHumidity());
51     //Serial.println(bme.getPressure());
52     //Serial.println(bme.getAltitude());
53     //Serial.println(bme.getGasResistance());
54 }
55
56 #endif

```

- 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.



```
forced_mode.ino
9 #include "bme68xLibrary.h"
10
11 #ifndef PIN_CS
12 #define PIN_CS 15
13 #endif
14
15 #ifndef ADD_I2C
16 #define ADD_I2C 0x77
17 #endif
18
19 BME68x bme;
20
21 /**
22  * @brief Initializes the sensor and hardware settings
23  */
24 void setup(void)
25 {
26   Wire.begin();
27   //SPI.begin();
28   Serial.begin(115200);
29
30   while (!Serial)
31     delay(10);
32
33   /* Initializes the sensor based on SPI library */
34   //bme.begin(PIN_CS, SPI);
35   bme.begin(ADD_I2C, Wire);
36
37   if(bme.checkStatus())
38   {
39     if (bme.checkStatus() == BME68X_ERROR)
40     {
41       //Serial.println("BME68X Error");
42     }
43   }
44 }
45
46 void loop()
47 {
48   //Serial.println("Temperature:");
49   float temp = bme.getTemp();
50   //Serial.println(temp);
51   //Serial.println("Barometric Pressure:");
52   float bar = bme.getBar();
53   //Serial.println(bar);
54   //Serial.println("Relative Humidity:");
55   float hum = bme.getHum();
56   //Serial.println(hum);
57   //Serial.println("Altitude:");
58   float alt = bme.getAlt();
59   //Serial.println(alt);
60   //Serial.println("Gas Resistance:");
61   float gas = bme.getGas();
62   //Serial.println(gas);
63
64   delay(1000);
65 }
66
67 #endif
```

串口监视器 x

浏览 (按照车辆制造商选择"COM11") 25FESP32 Dev Module 执行和回车 再高顿范 115200 baud

```
15:00:29.168 -> 274089, 29.64, 101944.12, 32.60, 299926.84, 89
15:00:29.303 -> 274231, 29.64, 101944.55, 32.60, 299924.00, 89
15:00:29.439 -> 274365, 29.65, 101943.25, 32.61, 299968.04, 89
15:00:29.567 -> 274509, 29.65, 101943.42, 32.60, 299991.28, 89
15:00:29.700 -> 274652, 29.64, 101941.79, 32.60, 299963.02, 89
15:00:29.838 -> 274796, 29.64, 101942.89, 32.47, 297918.12, 89
15:00:30.013 -> 274940, 29.65, 101943.42, 32.60, 298571.88, 89
15:00:30.148 -> 275074, 29.65, 101945.91, 32.49, 297918.12, 89
15:00:30.283 -> 275218, 29.65, 101944.81, 32.49, 298329.56, 89
15:00:30.400 -> 275362, 29.65, 101944.70, 32.49, 291369.72, 89
15:00:30.504 -> 275505, 29.65, 101943.07, 32.49, 298924.00, 89
15:00:30.729 -> 275649, 29.64, 101942.91, 32.47, 298220.19, 89
15:00:30.856 -> 275793, 29.64, 101942.65, 32.49, 297917.81, 89
```

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](#)