
Adafruit BME680 Library Documentation

Release 1.0

ladyada

Apr 26, 2021

Contents

1	Dependencies	3
2	Installing from PyPI	5
3	Usage Example	7
4	Contributing	9
5	Documentation	11
6	Table of Contents	13
6.1	Simple test	13
6.2	SPI Example	14
6.3	adafruit_bme680	14
6.3.1	Implementation Notes	14
7	Indices and tables	17
	Python Module Index	19
	Index	21

CircuitPython driver for BME680 sensor over I2C

CHAPTER 1

Dependencies

This driver depends on:

- [Adafruit CircuitPython](#)
- [Bus Device](#)

Please ensure all dependencies are available on the CircuitPython filesystem. This is easily achieved by downloading the [Adafruit library and driver bundle](#).

CHAPTER 2

Installing from PyPI

On supported GNU/Linux systems like the Raspberry Pi, you can install the driver locally [from PyPI](#). To install for current user:

```
pip3 install adafruit-circuitpython-bme680
```

To install system-wide (this may be required in some cases):

```
sudo pip3 install adafruit-circuitpython-bme680
```

To install in a virtual environment in your current project:

```
mkdir project-name && cd project-name
python3 -m venv .env
source .env/bin/activate
pip3 install adafruit-circuitpython-bme680
```


CHAPTER 3

Usage Example

```
import adafruit_bme680
import time
import board

# Create sensor object, communicating over the board's default I2C bus
i2c = board.I2C() # uses board.SCL and board.SDA
bme680 = adafruit_bme680.Adafruit_BME680_I2C(i2c)

# change this to match the location's pressure (hPa) at sea level
bme680.sea_level_pressure = 1013.25

while True:
    print("\nTemperature: %0.1f C" % bme680.temperature)
    print("Gas: %d ohm" % bme680.gas)
    print("Humidity: %0.1f %" % bme680.relative_humidity)
    print("Pressure: %0.3f hPa" % bme680.pressure)
    print("Altitude = %0.2f meters" % bme680.altitude)

    time.sleep(2)
```


CHAPTER 4

Contributing

Contributions are welcome! Please read our [Code of Conduct](#) before contributing to help this project stay welcoming.

CHAPTER 5

Documentation

For information on building library documentation, please check out [this guide](#).

6.1 Simple test

Ensure your device works with this simple test.

Listing 1: examples/bme680_simpletest.py

```
1  # SPDX-FileCopyrightText: 2021 ladyada for Adafruit Industries
2  # SPDX-License-Identifier: MIT
3
4  import time
5  import board
6  import adafruit_bme680
7
8  # Create sensor object, communicating over the board's default I2C bus
9  i2c = board.I2C() # uses board.SCL and board.SDA
10 bme680 = adafruit_bme680.Adafruit_BME680_I2C(i2c, debug=False)
11
12 # change this to match the location's pressure (hPa) at sea level
13 bme680.sea_level_pressure = 1013.25
14
15 # You will usually have to add an offset to account for the temperature of
16 # the sensor. This is usually around 5 degrees but varies by use. Use a
17 # separate temperature sensor to calibrate this one.
18 temperature_offset = -5
19
20 while True:
21     print("\nTemperature: %0.1f C" % (bme680.temperature + temperature_offset))
22     print("Gas: %d ohm" % bme680.gas)
23     print("Humidity: %0.1f %" % bme680.relative_humidity)
24     print("Pressure: %0.3f hPa" % bme680.pressure)
25     print("Altitude = %0.2f meters" % bme680.altitude)
26
27     time.sleep(1)
```

6.2 SPI Example

Showcase the use of the SPI bus to read the sensor data.

Listing 2: examples/bme680_spi.py

```
1  # SPDX-FileCopyrightText: 2021 Carter Nelson for Adafruit Industries
2  # SPDX-License-Identifier: MIT
3
4  import time
5  import board
6  import digitalio
7  import adafruit_bme680
8
9  # Create sensor object, communicating over the board's default SPI bus
10 cs = digitalio.DigitalInOut(board.D10)
11 spi = board.SPI()
12 bme680 = adafruit_bme680.Adafruit_BME680_SPI(spi, cs)
13
14 # change this to match the location's pressure (hPa) at sea level
15 bme680.sea_level_pressure = 1013.25
16
17 # You will usually have to add an offset to account for the temperature of
18 # the sensor. This is usually around 5 degrees but varies by use. Use a
19 # separate temperature sensor to calibrate this one.
20 temperature_offset = -5
21
22 while True:
23     print("\nTemperature: %0.1f C" % (bme680.temperature + temperature_offset))
24     print("Gas: %d ohm" % bme680.gas)
25     print("Humidity: %0.1f %" % bme680.relative_humidity)
26     print("Pressure: %0.3f hPa" % bme680.pressure)
27     print("Altitude = %0.2f meters" % bme680.altitude)
28
29     time.sleep(1)
```

6.3 adafruit_bme680

CircuitPython library for BME680 temperature, pressure and humidity sensor.

- Author(s): Limor Fried

6.3.1 Implementation Notes

Hardware:

- Adafruit BME680 Temp, Humidity, Pressure and Gas Sensor

Software and Dependencies:

- Adafruit CircuitPython firmware for the supported boards: <https://github.com/adafruit/circuitpython/releases>
- Adafruit's Bus Device library: https://github.com/adafruit/Adafruit_CircuitPython_BusDevice

```
class adafruit_bme680.Adafruit_BME680(*, refresh_rate=10)
    Driver from BME680 air quality sensor
```

Parameters `refresh_rate` (*int*) – Maximum number of readings per second. Faster property reads will be from the previous reading.

altitude

The altitude based on current `pressure` vs the sea level pressure (`sea_level_pressure`) - which you must enter ahead of time)

filter_size

The filter size for the built in IIR filter

gas

The gas resistance in ohms

humidity

The relative humidity in RH %

humidity_oversample

The oversampling for humidity sensor

pressure

The barometric pressure in hectoPascals

pressure_oversample

The oversampling for pressure sensor

relative_humidity

The relative humidity in RH %

sea_level_pressure = None

Pressure in hectoPascals at sea level. Used to calibrate `altitude`.

temperature

The compensated temperature in degrees Celsius.

temperature_oversample

The oversampling for temperature sensor

```
class adafruit_bme680.Adafruit_BME680_I2C(i2c, address=119, debug=False, *, refresh_rate=10)
```

Driver for I2C connected BME680.

Parameters

- `i2c` (*I2C*) – The I2C bus the BME680 is connected to.
- `address` (*int*) – I2C device address. Defaults to 0x77
- `debug` (*bool*) – Print debug statements when `True`. Defaults to `False`
- `refresh_rate` (*int*) – Maximum number of readings per second. Faster property reads will be from the previous reading.

Quickstart: Importing and using the BME680

Here is an example of using the `BMP680_I2C` class. First you will need to import the libraries to use the sensor

```
import board
import adafruit_bme680
```

Once this is done you can define your `board.I2C` object and define your sensor object

```
i2c = board.I2C()    # uses board.SCL and board.SDA
bme680 = adafruit_bme680.Adafruit_BME680_I2C(i2c)
```

You need to setup the pressure at sea level

```
bme680.sea_level_pressure = 1013.25
```

Now you have access to the temperature, gas, relative_humidity, pressure and altitude attributes

```
temperature = bme680.temperature
gas = bme680.gas
relative_humidity = bme680.relative_humidity
pressure = bme680.pressure
altitude = bme680.altitude
```

class adafruit_bme680.**Adafruit_BME680_SPI** (*spi, cs, baudrate=100000, debug=False, *, refresh_rate=10*)

Driver for SPI connected BME680.

Parameters

- **spi** (*SPI*) – SPI device
- **cs** (*DigitalInOut*) – Chip Select
- **debug** (*bool*) – Print debug statements when *True*. Defaults to *False*
- **baudrate** (*int*) – Clock rate, default is 100000
- **refresh_rate** (*int*) – Maximum number of readings per second. Faster property reads will be from the previous reading.

Quickstart: Importing and using the BME680

Here is an example of using the `BMP680_SPI` class. First you will need to import the libraries to use the sensor

```
import board
from digitalio import DigitalInOut, Direction
import adafruit_bme680
```

Once this is done you can define your `board.SPI` object and define your sensor object

```
cs = digitalio.DigitalInOut(board.D10)
spi = board.SPI()
bme680 = adafruit_bme680.Adafruit_BME680_SPI(spi, cs)
```

You need to setup the pressure at sea level

```
bme680.sea_level_pressure = 1013.25
```

Now you have access to the temperature, gas, relative_humidity, pressure and altitude attributes

```
temperature = bme680.temperature
gas = bme680.gas
relative_humidity = bme680.relative_humidity
pressure = bme680.pressure
altitude = bme680.altitude
```

CHAPTER 7

Indices and tables

- `genindex`
- `modindex`
- `search`

a

adafruit_bme680, [14](#)

A

Adafruit_BME680 (*class in adafruit_bme680*), [14](#)
adafruit_bme680 (*module*), [14](#)
Adafruit_BME680_I2C (*class in adafruit_bme680*),
[15](#)
Adafruit_BME680_SPI (*class in adafruit_bme680*),
[16](#)
altitude (*adafruit_bme680.Adafruit_BME680 at-*
tribute), [15](#)

F

filter_size (*adafruit_bme680.Adafruit_BME680 at-*
tribute), [15](#)

G

gas (*adafruit_bme680.Adafruit_BME680 attribute*), [15](#)

H

humidity (*adafruit_bme680.Adafruit_BME680 at-*
tribute), [15](#)
humidity_oversample
(*adafruit_bme680.Adafruit_BME680 at-*
tribute), [15](#)

P

pressure (*adafruit_bme680.Adafruit_BME680 at-*
tribute), [15](#)
pressure_oversample
(*adafruit_bme680.Adafruit_BME680 at-*
tribute), [15](#)

R

relative_humidity
(*adafruit_bme680.Adafruit_BME680 at-*
tribute), [15](#)

S

sea_level_pressure
(*adafruit_bme680.Adafruit_BME680 at-*
tribute), [15](#)

T

temperature (*adafruit_bme680.Adafruit_BME680 at-*
tribute), [15](#)
temperature_oversample
(*adafruit_bme680.Adafruit_BME680 at-*
tribute), [15](#)