Adafruit BMP280 Library Documentation

Release 1.0

ladyada

May 22, 2019

Contents

CircuitPython driver from BMP280 Temperature and Barometic Pressure sensor

CHAPTER 1

Installation and 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.

1.1 Installing from PyPI

On supported GNU/Linux systems like the Raspberry Pi, you can install the driver from PyPI. To install for the current user:

```
pip3 install adafruit-circuitpython-bmp280
```

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

```
sudo pip3 install adafruit-circuitpython-bmp280
```

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-bmp280
```

CHAPTER 2

Usage Example

```
import board
import digitalio
import busio
import time
from adafruit_bmp280 import adafruit_bmp280

# Create library object using our Bus I2C port
i2c = busio.I2C(board.SCL, board.SDA)
bmp280 = adafruit_bmp280.Adafruit_BMP280_I2C(i2c)

# OR create library object using our Bus SPI port
#spi = busio.SPI(board.SCK, board.MOSI, board.MISO)
#bmp_cs = digitalio.DigitalInOut(board.D10)
#bmp280 = adafruit_bmp280.Adafruit_BMP280_SPI(spi, bmp_cs)

# change this to match the location's pressure (hPa) at sea level
bmp280.seaLevelhPa = 1013.25

while True:
    print("\nTemperature: %0.1f C" % bmp280.temperature)
    print("Pressure: %0.1f hPa" % bmp280.pressure)
    print("Altitude = %0.2f meters" % bmp280.altitude)
    time.sleep(2)
```

CHAPTER 3

Contributing

Contributions are welcome! Please read our Code of Conduct before contributing to help this project stay welcoming.

CHAPTER 4

Building locally

To build this library locally you'll need to install the circuitpython-build-tools package.

```
python3 -m venv .env
source .env/bin/activate
pip install circuitpython-build-tools
```

Once installed, make sure you are in the virtual environment:

```
source .env/bin/activate
```

Then run the build:

```
circuitpython-build-bundles --filename_prefix adafruit-circuitpython-bmp280 --library_
↪location .
```

4.1 Sphinx documentation

Sphinx is used to build the documentation based on rST files and comments in the code. First, install dependencies (feel free to reuse the virtual environment from above):

```
python3 -m venv .env
source .env/bin/activate
pip install Sphinx sphinx-rtd-theme
```

Now, once you have the virtual environment activated:

```
cd docs
sphinx-build -E -W -b html . _build/html
```

This will output the documentation to `docs/_build/html`. Open the index.html in your browser to view them. It will also (due to -W) error out on any warning like Travis will. This is a good way to locally verify it will pass.

CHAPTER 5

Table of Contents

5.1 Simple test

Ensure your device works with this simple test.

Listing 1: examples/bmp280_simpletest.py

```
import time

import board
# import digitalio # For use with SPI
import busio

import adafruit_bmp280

# Create library object using our Bus I2C port
i2c = busio.I2C(board.SCL, board.SDA)
bmp280 = adafruit_bmp280.Adafruit_BMP280_I2C(i2c)

# OR create library object using our Bus SPI port
#spi = busio.SPI(board.SCK, board.MOSI, board.MISO)
#bmp_cs = digitalio.DigitalInOut(board.D10)
#bmp280 = adafruit_bmp280.Adafruit_BMP280_SPI(spi, bmp_cs)

# change this to match the location's pressure (hPa) at sea level
bmp280.sea_level_pressure = 1013.25

while True:
    print("\nTemperature: %0.1f C" % bmp280.temperature)
    print("Pressure: %0.1f hPa" % bmp280.pressure)
    print("Altitude = %0.2f meters" % bmp280.altitude)
    time.sleep(2)
```

5.2 adafruit_bmp280 - Adafruit BMP280 - Temperature & Barometic Pressure Sensor

CircuitPython driver from BMP280 Temperature and Barometic Pressure sensor

- Author(s): ladyada

class adafruit_bmp280.**Adafruit_BMP280**
Base BMP280 object. Use *Adafruit_BMP280_I2C* or *Adafruit_BMP280_SPI* instead of this. This checks the BMP280 was found, reads the coefficients and enables the sensor for continuous reads

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

iir_filter
Controls the time constant of the IIR filter Allowed values are set in the IIR_FILTER enum class

measurement_time_max
Maximum time in milliseconds required to complete a measurement in normal mode

measurement_time_typical
Typical time in milliseconds required to complete a measurement in normal mode

mode
Operation mode Allowed values are set in the MODE enum class

overscan_pressure
Pressure Oversampling Allowed values are set in the OVERSCAN enum class

overscan_temperature
Temperature Oversampling Allowed values are set in the OVERSCAN enum class

pressure
The compensated pressure in hectoPascals. returns None if pressure measurement is disabled

sea_level_pressure = None
Pressure in hectoPascals at sea level. Used to calibrate *altitude*.

standby_period
Control the inactive period when in Normal mode Allowed standby periods are set the STANDBY enum class

temperature
The compensated temperature in degrees celsius.

class adafruit_bmp280.**Adafruit_BMP280_I2C**(*i2c*, *address=119*)
Driver for I2C connected BMP280. Default address is 0x77 but another address can be passed in as an argument

class adafruit_bmp280.**Adafruit_BMP280_SPI**(*spi*, *cs*, *baudrate=100000*)
Driver for SPI connected BMP280. Default clock rate is 100000 but can be changed with 'baudrate'

CHAPTER 6

Indices and tables

- genindex
- modindex
- search

Python Module Index

a

Index