Adafruit's CCS811 Library Documentation

Release 1.0

Dean Miller, Scott Shawcroft

Jun 07, 2021

Contents

CircuitPython driver for the CCS811 air quality sensor.

CHAPTER 1

Dependencies

This driver depends on:

- Adafruit CircuitPython
- Bus Device
- Register

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-ccs811
```

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

```
sudo pip3 install adafruit-circuitpython-ccs811
```

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-ccs811
```

CHAPTER 3

Usage Notes

See the guide for wiring and installation instructions.

Of course, you must import the library to use it:

```
import busio
import adafruit_ccs811
```

Next, initialize the I2C bus object.

```
from board import *
i2c = board.I2C()   # uses board.SCL and board.SDA
```

Once you have created the I2C interface object, you can use it to instantiate the CCS811 object

```
ccs =  adafruit_ccs811.CCS811(i2c)
```

3.1 Reading Sensor

To read the gas sensor simply read the attributes:

```
print("CO2: ", ccs.eco2, " TVOC:", ccs.tvoc)
```

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.

CHAPTER 6

Table of Contents

6.1 Simple test

Ensure your device works with this simple test.

Listing 1: examples/ccs811_simpletest.py

```
# SPDX-FileCopyrightText: 2021 ladyada for Adafruit Industries
# SPDX-License-Identifier: MIT

import time
import board
import adafruit_ccs811

i2c = board.I2C()  # uses board.SCL and board.SDA
ccs811 = adafruit_ccs811.CCS811(i2c)

# Wait for the sensor to be ready
while not ccs811.data_ready:
    pass

while True:
    print("CO2: {} PPM, TVOC: {} PPB".format(ccs811.eco2, ccs811.tvoc))
    time.sleep(0.5)
```

6.2 adafruit_ccs811

This library supports the use of the CCS811 air quality sensor in CircuitPython.

Author(s): Dean Miller for Adafruit Industries

Hardware:

- Adafruit CCS811 Air Quality Sensor Breakout - VOC and eCO2

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
- Adafruit's Register library: https://github.com/adafruit/Adafruit_CircuitPython_Register

Notes:

#. Datasheet

class adafruit_ccs811.**CCS811**(*i2c_bus*, *address=90*)

CCS811 gas sensor driver.

Parameters

- **i2c_bus** (*I2C*) – The I2C bus the BME280 is connected to
- **address** (*int*) – The I2C address of the CCS811. Defaults to `0x5A`

Quickstart: Importing and using the CCS811

Here is an example of using the *CCS811* class. First you will need to import the libraries to use the sensor

```
import board
import adafruit_ccs811
```

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
ccs811 = adafruit_ccs811.CCS811(i2c)
```

Now you have access to the *eco2* and *tvoc* attributes.

```
eco2 = ccs811.eco2
tvoc = ccs811.tvoc
```

baseline

The property reads and returns the current baseline value. The returned value is packed into an integer. Later the same integer can be used in order to set a new baseline.

data_ready

True when new data has been read.

eco2

Equivalent Carbon Dioxide in parts per million. Clipped to 400 to 8192ppm.

error

True when an error has occured.

error_code

Error code

reset()

Initiate a software reset.

set_environmental_data(*humidity*, *temperature*)

Set the temperature and humidity used when computing eCO2 and TVOC values.

Parameters

- **humidity** (*int*) – The current relative humidity in percent.
- **temperature** (*float*) – The current temperature in Celsius.

set_interrupt_thresholds(*low_med*, *med_high*, *hysteresis*)
Set the thresholds used for triggering the interrupt based on eCO2. The interrupt is triggered when the value crossed a boundary value by the minimum hysteresis value.

Parameters

- **low_med** (*int*) – Boundary between low and medium ranges
- **med_high** (*int*) – Boundary between medium and high ranges
- **hysteresis** (*int*) – Minimum difference between reads

temp_offset = 0.0
Temperature offset.

temperature
Deprecated since version 1.1.5: Hardware support removed by vendor

Temperature based on optional thermistor in Celsius.

tvoc
Total Volatile Organic Compound in parts per billion.

CHAPTER 7

Indices and tables

- genindex
- modindex
- search

Python Module Index

a

Index