AdafruitVEML6070 Library Documentation

Release 1.0

Limor Fried
Michael Schroeder

Mar 15, 2018

Contents

1 Dependencies **3**

2 Usage Example **5**

3 Contributing **7**

4 Building locally **9**
4.1 Sphinx documentation 9

5 Table of Contents **11**
5.1 Simple test 11
5.2 `adafruit_veml6070` - VEML6070 UV Sensor 11
5.2.1 Implementation Notes 12

6 Indices and tables **15**

Python Module Index **17**

CircuitPython driver for the VEML6070 UV Index Sensor Breakout

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

Usage Example

```
import time
import board
import busio
from adafruit_veml6070 import VEML6070

with busio.I2C(board.SCL, board.SDA) as i2c:
    uv = VEML6070(i2c)
    # Alternative constructors with parameters
    #uv = VEML6070(i2c, 'VEML6070_1_T')
    #uv = VEML6070(i2c, 'VEML6070_HALF_T', True)

    # take 10 readings
    for j in range(10):
        uv_raw = uv.read
        risk_level = uv.get_index(uv_raw)
        print('Reading: {0} | Risk Level: {1}'.format(uv_raw, risk_level))
        time.sleep(1)
```

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-veml6070 --
↪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 5.1: examples/veml6070_simpletest.py

```
# VEML6070 Driver Example Code

import time
import busio
import board
import adafruit_veml6070

with busio.I2C(board.SCL, board.SDA) as i2c:
    uv = adafruit_veml6070.VEML6070(i2c)
    # Alternative constructors with parameters
    #uv = adafruit_veml6070.VEML6070(i2c, 'VEML6070_1_T')
    #uv = adafruit_veml6070.VEML6070(i2c, 'VEML6070_HALF_T', True)

    # take 10 readings
    for j in range(10):
        uv_raw = uv.read
        risk_level = uv.get_index(uv_raw)
        print('Reading: {0} | Risk Level: {1}'.format(uv_raw, risk_level))
        time.sleep(1)
```

5.2 `adafruit_veml6070` - VEML6070 UV Sensor

CircuitPython library to support VEML6070 UV Index sensor.

- Author(s): Limor Fried & Michael Schroeder

5.2.1 Implementation Notes

Hardware:

- Adafruit VEML6070 UV Index Sensor Breakout (Product ID: 2899)

Software and Dependencies:

- Adafruit CircuitPython firmware (2.2.0+) for the ESP8622 and M0-based boards: https://github.com/adafruit/circuitpython/releases
- Adafruit's Bus Device library: https://github.com/adafruit/Adafruit_CircuitPython_BusDevice

Notes:

1. Datasheet: https://cdn-learn.adafruit.com/assets/assets/000/032/482/original/veml6070.pdf

class adafruit_veml6070.**VEML6070**(*i2c_bus*, *_veml6070_it='VEML6070_1_T'*, *ack=False*)

Driver base for the VEML6070 UV Light Sensor

Parameters

- **i2c_bus** – The `busio.I2C` object to use. This is the only required parameter.
- **_veml6070_it** (*str*) – The integration time you'd like to set initially. Availble options: `VEML6070_HALF_T`, `VEML6070_1_T`, `VEML6070_2_T`, and `VEML6070_4_T`. The higher the '_x_' value, the more accurate the reading is (at the cost of less samples per reading). Defaults to `VEML6070_1_T` if parameter not passed. To change setting after intialization, use `[veml6070].set_integration_time(new_it)`.
- **ack** (*bool*) – The inital setting of `ACKnowledge` on alert. Defaults to `False` if parameter not passed. To change setting after intialization, use `[veml6070].set_ack(new_ack)`.

Example:

```
from board import *
import busio, veml6070, time

with busio.I2C(SCL, SDA) as i2c:
    uv = veml6070.VEML6070(i2c, 'VEML6070_1_T', True)

    # take 10 readings
    for j in range(10):
        uv_raw = uv.read
        risk_level = uv.get_index(uv_raw)
        print('Reading: ', uv_raw, ' | Risk Level: ', risk_level)
        time.sleep(1)
```

get_index(*_raw*)

Calculates the UV Risk Level based on the captured UV reading. Requres the `_raw` argument (from `veml6070.read`). Risk level is available for Integration Times (IT) 1, 2, & 4. The result is automatically scaled to the current IT setting.

LEVEL* UV Index ===== ======== LOW 0-2 MODERATE 3-5 HIGH 6-7 VERY HIGH 8-10 EXTREME >=11

- Not to be considered as accurate condition reporting. Calculation is based on VEML6070 Application Notes: http://www.vishay.com/docs/84310/designingveml6070.pdf

read

Reads and returns the value of the UV intensity.

set_ack(*new_ack=False*)

Turns on or off the ACKnowledge function of the sensor. The ACK function will send a signal to the host when the value of the sensed UV light changes beyond the programmed threshold. Use `[veml6070].set_ack_threshold` to change between the two available threshold settings.

set_ack_threshold(*new_ack_thd=0*)

Sets the ACKnowledge Threshold, which alerts the host controller to value changes greater than the threshold. Available settings are: `0` = 102 steps; `1` = 145 steps. `0` is the default setting.

set_integration_time(*new_it*)

Sets the Integration Time of the sensor. This is the refresh interval of the sensor. The higher the refresh interval, the more accurate the reading is (at the cost of less sampling). The available settings are: `VEML6070_HALF_T`, `VEML6070_1_T`, `VEML6070_2_T`, `VEML6070_4_T`.

sleep()

Puts the VEML6070 into sleep ('shutdown') mode. Datasheet claims a current draw of 1uA while in shutdown.

wake()

Wakes the VEML6070 from sleep. `[veml6070].read` will also wake from sleep.

CHAPTER 6

Indices and tables

- genindex
- modindex
- search

Python Module Index

a

adafruit_veml6070, 11

Index

A

adafruit_veml6070 (module), 11

G

get_index() (adafruit_veml6070.VEML6070 method), 12

R

read (adafruit_veml6070.VEML6070 attribute), 12

S

set_ack() (adafruit_veml6070.VEML6070 method), 13
set_ack_threshold() (adafruit_veml6070.VEML6070 method), 13
set_integration_time() (adafruit_veml6070.VEML6070 method), 13
sleep() (adafruit_veml6070.VEML6070 method), 13

V

VEML6070 (class in adafruit_veml6070), 12

W

wake() (adafruit_veml6070.VEML6070 method), 13