AdafruitAPDS9960 Library Documentation

Release 1.0

Michael McWethy

Aug 25, 2018

Contents

The APDS9960 is a specialize chip that detects hand gestures, proximity detection and ambient light color over I2C. Its available on Adafruit as a breakout.

CHAPTER 1

Dependencies

This driver depends on:

- Adafruit CircuitPython

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

2.1 Hardware Set-up

Connect Vin to 3.3 V or 5 V power source, GND to ground, SCL and SDA to the appropriate pins.

2.2 Basics

Of course, you must import i2c bus device, board pins, and the library:

```
from board import SCL, SDA, A1
from adafruit_apds9960.apds9960 import APDS9960
import busio
import digitalio
```

To set-up the device to gather data, initialize the I2CDevice using SCL and SDA pins. Then initialize the library. Optionally provide an interrupt pin for proximity detection.

```
int_pin = digitalio.DigitalInOut(A1)
i2c = busio.I2C(SCL, SDA)
apds = APDS9960(i2c, interrupt_pin=int_pin)
```

2.3 Gestures

To get a gesture, see if a gesture is available first, then get the gesture Code

```
gesture = apds.gesture()
if gesture == 1:
  print("up")
if gesture == 2:
```

(continues on next page)

(continued from previous page)

```
  print("down")
if gesture == 3:
  print("left")
if gesture == 4:
  print("right")
```

2.4 Color Measurement

To get a color measure, enable color measures, wait for color data, then get the color data.

```
apds.enable_color = True

while not apds.color_data_ready:
    time.sleep(0.005)

r, g, b, c = apds.color_data
print("r: {}, g: {}, b: {}, c: {}".format(r, g, b, c))
```

2.5 Proximity Detection

To check for a object in proximity, see if a gesture is available first, then get the gesture Code

```
apds.enable_proximity = True

# set the interrupt threshold to fire when proximity reading goes above 175
apds.proximity_interrupt_threshold = (0, 175)

# enable the proximity interrupt
apds.enable_proximity_interrupt = True

while True:
  if not interrupt_pin.value:
    print(apds.proximity())

    # clear the interrupt
    apds.clear_interrupt()
```

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-travis-build-tools package.

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

Then run the build:

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/apds9960_color_simpletest.py

```
import time
import board
import busio
import digitalio
from adafruit_apds9960.apds9960 import APDS9960
from adafruit_apds9960 import colorutility

i2c = busio.I2C(board.SCL, board.SDA)
int_pin = digitalio.DigitalInOut(board.D5)
apds = APDS9960(i2c)
apds.enable_color = True


while True:
    #create some variables to store the color data in

    #wait for color data to be ready
    while not apds.color_data_ready:
        time.sleep(0.005)


    #get the data and print the different channels
    r, g, b, c = apds.color_data
    print("red: ", r)
    print("green: ", g)
    print("blue: ", b)
    print("clear: ", c)
```

(continues on next page)

(continued from previous page)

```

    print("color temp {}".format(colorutility.calculate_color_temperature(r, g, b)))
    print("light lux {}".format(colorutility.calculate_lux(r, g, b)))
    time.sleep(0.5)
```

Listing 2: examples/apds9960_gesture_simpletest.py

```
from board import SCL, SDA
import busio
from adafruit_apds9960.apds9960 import APDS9960

i2c = busio.I2C(SCL, SDA)

apds = APDS9960(i2c)
apds.enable_proximity = True
apds.enable_gesture = True

while True:
    gesture = apds.gesture()

    if gesture == 0x01:
        print("up")
    elif gesture == 0x02:
        print("down")
    elif gesture == 0x03:
        print("left")
    elif gesture == 0x04:
        print("right")
```

Listing 3: examples/apds9960_proximity_simpletest.py

```
import board
import busio
import digitalio
from adafruit_apds9960.apds9960 import APDS9960

i2c = busio.I2C(board.SCL, board.SDA)
int_pin = digitalio.DigitalInOut(board.D5)
apds = APDS9960(i2c, interrupt_pin=int_pin)

apds.enable_proximity = True
apds.proximity_interrupt_threshold = (0, 175)
apds.enable_proximity_interrupt = True

while True:
    # print the proximity reading when the interrupt pin goes low
    if not int_pin.value:
        print(apds.proximity())

        # clear the interrupt
        apds.clear_interrupt()
```

5.2 APDS9960

Driver class for the APDS9960 board. Supports gesture, proximity, and color detection.

- Author(s): Michael McWethy

class adafruit_apds9960.apds9960.**APDS9960**(*i2c*, ***, *interrupt_pin=None*, *address=57*, *integration_time=1*, *gain=1*)

APDS9900 provide basic driver services for the ASDS9960 breakout board

clear_interrupt()
Clear all interrupts

color_data
Tuple containing r, g, b, c values

color_data_ready
Color data ready flag. zero if not ready, 1 is ready

color_gain
Color gain value

enable
Board enable. True to enable, False to disable

enable_color
Color detection enable flag. True when color detection is enabled, else False

enable_gesture
Gesture detection enable flag. True to enable, False to disable. Note that when disabled, gesture mode is turned off

enable_proximity
Enable of proximity mode

enable_proximity_interrupt
Proximity interrupt enable flag. True if enabled, False to disable

gesture()
Returns gesture code if detected. =0 if no gesture detected =1 if an UP, =2 if a DOWN, =3 if an LEFT, =4 if a RIGHT

gesture_dimensions
Gesture dimension value: range 0-3

gesture_fifo_threshold
Gesture fifo threshold value: range 0-3

gesture_gain
Gesture gain value: range 0-3

gesture_proximity_threshold
Proximity threshold value: range 0-255

integration_time
Proximity integration time: range 0-255

proximity()
proximity value: range 0-255

proximity_interrupt_threshold
Tuple containing low and high threshold followed by the proximity interrupt persistance. When setting the

proximity interrupt threshold values using a tuple of zero to three values: low threshold, high threshold, persistance. persistance defaults to 4 if not provided

5.3 colorutility

Helper functions for color calculations

- Author(s): Michael McWethy

adafruit_apds9960.colorutility.**calculate_color_temperature**(*r*, *g*, *b*)
Converts the raw R/G/B values to color temperature in degrees Kelvin

adafruit_apds9960.colorutility.**calculate_lux**(*r*, *g*, *b*)
Calculate ambient light values

CHAPTER 6

Indices and tables

- genindex
- modindex
- search

Python Module Index

a

Index