AdafruitMPU6050 Library Documentation

Release 1.0

Bryan Siepert

Jun 07, 2021

Contents

CircuitPython helper library for the MPU6050 6-DoF Accelerometer and Gyroscope

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-mpu6050
```

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

```
sudo pip3 install adafruit-circuitpython-mpu6050
```

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-mpu6050
```

CHAPTER 3

Usage Example

```
import time
import board
import adafruit_mpu6050

i2c = board.I2C()  # uses board.SCL and board.SDA
mpu = adafruit_mpu6050.MPU6050(i2c)

while True:
    print("Acceleration: X:%.2f, Y: %.2f, Z: %.2f m/s^2"%(mpu.acceleration))
    print("Gyro X:%.2f, Y: %.2f, Z: %.2f degrees/s"%(mpu.gyro))
    print("Temperature: %.2f C"%mpu.temperature)
    print("")
    time.sleep(1)
```

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/mpu6050_simpletest.py

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

import time
import board
import adafruit_mpu6050

i2c = board.I2C()  # uses board.SCL and board.SDA
mpu = adafruit_mpu6050.MPU6050(i2c)

while True:
    print("Acceleration: X:%.2f, Y: %.2f, Z: %.2f m/s^2" % (mpu.acceleration))
    print("Gyro X:%.2f, Y: %.2f, Z: %.2f rad/s" % (mpu.gyro))
    print("Temperature: %.2f C" % mpu.temperature)
    print("")
    time.sleep(1)
```

6.2 Plotter Example

See the effects of changing the gyroscope and accelerometer range by viewing the data in a serial plotter

Listing 2: examples/mpu6050_plotter_example.py

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

(continues on next page)

(continued from previous page)

```

import time
import board
import adafruit_mpu6050

i2c = board.I2C()  # uses board.SCL and board.SDA
mpu = adafruit_mpu6050.MPU6050(i2c)
mpu.accelerometer_range = adafruit_mpu6050.Range.RANGE_2_G
mpu.gyro_range = adafruit_mpu6050.GyroRange.RANGE_250_DPS

while True:
    # this prints out all the values like a tuple which Mu's plotter prefer
    print("(%.2f, %.2f, %.2f " % (mpu.acceleration), end=", ")
    print("%.2f, %.2f, %.2f)" % (mpu.gyro))
    time.sleep(0.010)
```

6.3 Sleep Example

Observe how the cycle and sleep modes effect measurements by viewing the data in a serial plotter

Listing 3: examples/mpu6050_sleep_example.py

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

import time
import board
import adafruit_mpu6050

i2c = board.I2C()  # uses board.SCL and board.SDA
mpu = adafruit_mpu6050.MPU6050(i2c)

# This example is meant to be used with the serial plotter which makes
# it easier to see how the readings change with different settings.
# Make sure to poke and prod the sensor while the demo is running to
# generate some interesting data!

while True:
    # first show some 'normal' readings

    mpu.sleep = False
    mpu.cycle = False

    for count in range(0, 100):
        print(mpu.acceleration)
        time.sleep(0.010)

    # Next, set a slow cycle rate so the effect can be seen clearly.
    mpu.cycle_Rate = adafruit_mpu6050.Rate.CYCLE_5_HZ
    # ensure that we're not sleeping or cycle won't work
    mpu.sleep = False
    # Finally, enable cycle mode
    mpu.cycle = True

```

(continues on next page)

(continued from previous page)

```
    for count in range(0, 100):
        print(mpu.acceleration)
        time.sleep(0.010)

    # Finally enable sleep mode. Note that while we can still fetch
    #  data from the measurement registers, the measurements are not
    #  updated. In sleep mode the accelerometer and gyroscope are
    #  deactivated to save power, so measurements are halted.

    mpu.cycle = False
    mpu.sleep = True

    for count in range(0, 100):
        print(mpu.acceleration)
        time.sleep(0.010)
```

6.4 Inclinometer Example

Provides an example on how to use the sensor as an inclinometer

Listing 4: examples/mpu6050_inclinometer.py

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

# Display inclination data five times per second

# See this page to learn the math and physics principals behind this example:
# https://learn.adafruit.com/how-tall-is-it/gravity-and-acceleration

import time
from math import atan2, degrees
import board
import adafruit_mpu6050

i2c = board.I2C()  # uses board.SCL and board.SDA
sensor = adafruit_mpu6050.MPU6050(i2c)


# Given a point (x, y) return the angle of that point relative to x axis.
# Returns: angle in degrees


def vector_2_degrees(x, y):
    angle = degrees(atan2(y, x))
    if angle < 0:
        angle += 360
    return angle


# Given an accelerometer sensor object return the inclination angles of X/Z and Y/Z
# Returns: tuple containing the two angles in degrees


```

(continues on next page)

(continued from previous page)

```
def get_inclination(_sensor):
    x, y, z = _sensor.acceleration
    return vector_2_degrees(x, z), vector_2_degrees(y, z)


while True:
    angle_xz, angle_yz = get_inclination(sensor)
    print("XZ angle = {:6.2f}deg   YZ angle = {:6.2f}deg".format(angle_xz, angle_yz))
    time.sleep(0.2)
```

6.5 adafruit_mpu6050

CircuitPython helper library for the MPU6050 6-DoF Accelerometer and Gyroscope

- Author(s): Bryan Siepert

6.5.1 Implementation Notes

Hardware:

- Adafruit MPU-6050 6-DoF Accel and Gyro Sensor (Product ID: 3886)

Software and Dependencies:

- Adafruit CircuitPython firmware for the supported boards: https://circuitpython.org/downloads
- Adafruit's Bus Device library: https://github.com/adafruit/Adafruit_CircuitPython_BusDevice
- Adafruit's Register library: https://github.com/adafruit/Adafruit_CircuitPython_Register

class adafruit_mpu6050.**Bandwidth**
Allowed values for *filter_bandwidth*.

- Bandwidth.BAND_260_HZ
- Bandwidth.BAND_184_HZ
- Bandwidth.BAND_94_HZ
- Bandwidth.BAND_44_HZ
- Bandwidth.BAND_21_HZ
- Bandwidth.BAND_10_HZ
- Bandwidth.BAND_5_HZ

class adafruit_mpu6050.**GyroRange**
Allowed values for *gyro_range*.

- GyroRange.RANGE_250_DPS
- GyroRange.RANGE_500_DPS
- GyroRange.RANGE_1000_DPS
- GyroRange.RANGE_2000_DPS

class adafruit_mpu6050.**MPU6050**(*i2c_bus*, *address=104*)
Driver for the MPU6050 6-DoF accelerometer and gyroscope.

Parameters

- **i2c_bus** (*I2C*) – The I2C bus the device is connected to
- **address** (*int*) – The I2C device address. Defaults to `0x68`

Quickstart: Importing and using the device

Here is an example of using the *MPU6050* class. First you will need to import the libraries to use the sensor

```
import board
import adafruit_mpu6050
```

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
mpu = adafruit_mpu6050.MPU6050(i2c)
```

Now you have access to the *acceleration*, *gyro* and *temperature* attributes

```
acc_x, acc_y, acc_z = sensor.acceleration
gyro_x, gyro_y, gyro_z = sensor.gyro
temperature = sensor.temperature
```

acceleration
Acceleration X, Y, and Z axis data in m/s^2

accelerometer_range
The measurement range of all accelerometer axes. Must be a *Range*

cycle
Enable or disable periodic measurement at a rate set by *cycle_rate()*. If the sensor was in sleep mode, it will be waken up to cycle

cycle_rate
The rate that measurements are taken while in *cycle* mode. Must be a *Rate*

filter_bandwidth
The bandwidth of the gyroscope Digital Low Pass Filter. Must be a *GyroRange*

gyro
Gyroscope X, Y, and Z axis data in $/s$

gyro_range
The measurement range of all gyroscope axes. Must be a *GyroRange*

reset()
Reinitialize the sensor

sample_rate_divisor
The sample rate divisor. See the datasheet for additional detail

sleep
Shuts down the accelerometers and gyroscopes, saving power. No new data will be recorded until the sensor is taken out of sleep by setting to `False`

temperature
The current temperature in º Celsius

class adafruit_mpu6050.**Range**
Allowed values for *accelerometer_range*.

- Range.RANGE_2_G
- Range.RANGE_4_G
- Range.RANGE_8_G
- Range.RANGE_16_G

class adafruit_mpu6050.**Rate**

Allowed values for *cycle_rate*.

- Rate.CYCLE_1_25_HZ
- Rate.CYCLE_5_HZ
- Rate.CYCLE_20_HZ
- Rate.CYCLE_40_HZ

CHAPTER 7

Indices and tables

- genindex
- modindex
- search

Python Module Index

a

Index