
AdafruitMPU6050 Library Documentation

Release 1.0

Bryan Siepert

May 10, 2021

Contents

| | | |
|----------|--------------------------------|-----------|
| 1 | Dependencies | 3 |
| 2 | Installing from PyPI | 5 |
| 3 | Usage Example | 7 |
| 4 | Contributing | 9 |
| 5 | Documentation | 11 |
| 6 | Table of Contents | 13 |
| 6.1 | Simple test | 13 |
| 6.2 | Plotter Example | 13 |
| 6.3 | Sleep Example | 14 |
| 6.4 | Inclinometer Example | 15 |
| 6.5 | adafruit_mpu6050 | 16 |
| 6.5.1 | Implementation Notes | 16 |
| 7 | Indices and tables | 19 |
| | Python Module Index | 21 |
| | Index | 23 |

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](#).

6.1 Simple test

Ensure your device works with this simple test.

Listing 1: examples/mpu6050_simpletest.py

```
1 # SPDX-FileCopyrightText: 2021 ladyada for Adafruit Industries
2 # SPDX-License-Identifier: MIT
3
4 import time
5 import board
6 import adafruit_mpu6050
7
8 i2c = board.I2C() # uses board.SCL and board.SDA
9 mpu = adafruit_mpu6050.MPU6050(i2c)
10
11 while True:
12     print("Acceleration: X:%.2f, Y: %.2f, Z: %.2f m/s^2" % (mpu.acceleration))
13     print("Gyro X:%.2f, Y: %.2f, Z: %.2f rad/s" % (mpu.gyro))
14     print("Temperature: %.2f C" % mpu.temperature)
15     print("")
16     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

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

(continues on next page)

(continued from previous page)

```

3
4 import time
5 import board
6 import adafruit_mpu6050
7
8 i2c = board.I2C() # uses board.SCL and board.SDA
9 mpu = adafruit_mpu6050.MPU6050(i2c)
10 mpu.accelerometer_range = adafruit_mpu6050.Range.RANGE_2_G
11 mpu.gyro_range = adafruit_mpu6050.GyroRange.RANGE_250_DPS
12
13 while True:
14     # this prints out all the values like a tuple which Mu's plotter prefer
15     print("%.2f, %.2f, %.2f" % (mpu.acceleration), end=", ")
16     print("%.2f, %.2f, %.2f" % (mpu.gyro))
17     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

```

1 # SPDX-FileCopyrightText: 2021 ladyada for Adafruit Industries
2 # SPDX-License-Identifier: MIT
3
4 import time
5 import board
6 import adafruit_mpu6050
7
8 i2c = board.I2C() # uses board.SCL and board.SDA
9 mpu = adafruit_mpu6050.MPU6050(i2c)
10
11 # This example is meant to be used with the serial plotter which makes
12 # it easier to see how the readings change with different settings.
13 # Make sure to poke and prod the sensor while the demo is running to
14 # generate some interesting data!
15
16 while True:
17     # first show some 'normal' readings
18
19     mpu.sleep = False
20     mpu.cycle = False
21
22     for count in range(0, 100):
23         print(mpu.acceleration)
24         time.sleep(0.010)
25
26     # Next, set a slow cycle rate so the effect can be seen clearly.
27     mpu.cycle_rate = adafruit_mpu6050.Rate.CYCLE_5_HZ
28     # ensure that we're not sleeping or cycle won't work
29     mpu.sleep = False
30     # Finally, enable cycle mode
31     mpu.cycle = True
32

```

(continues on next page)

(continued from previous page)

```

33     for count in range(0, 100):
34         print(mpu.acceleration)
35         time.sleep(0.010)
36
37     # Finally enable sleep mode. Note that while we can still fetch
38     # data from the measurement registers, the measurements are not
39     # updated. In sleep mode the accelerometer and gyroscope are
40     # deactivated to save power, so measurements are halted.
41
42     mpu.cycle = False
43     mpu.sleep = True
44
45     for count in range(0, 100):
46         print(mpu.acceleration)
47         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

```

1  # SPDX-FileCopyrightText: 2021 ladyada for Adafruit Industries
2  # SPDX-License-Identifier: MIT
3
4  # Display inclination data five times per second
5
6  # See this page to learn the math and physics principals behind this example:
7  # https://learn.adafruit.com/how-tall-is-it/gravity-and-acceleration
8
9  import time
10 from math import atan2, degrees
11 import board
12 import adafruit_mpu6050
13
14 i2c = board.I2C() # uses board.SCL and board.SDA
15 sensor = adafruit_mpu6050.MPU6050(i2c)
16
17
18 # Given a point (x, y) return the angle of that point relative to x axis.
19 # Returns: angle in degrees
20
21
22 def vector_2_degrees(x, y):
23     angle = degrees(atan2(y, x))
24     if angle < 0:
25         angle += 360
26     return angle
27
28
29 # Given an accelerometer sensor object return the inclination angles of X/Z and Y/Z
30 # Returns: tuple containing the two angles in degrees
31
32

```

(continues on next page)

(continued from previous page)

```

33 def get_inclination(_sensor):
34     x, y, z = _sensor.acceleration
35     return vector_2_degrees(x, z), vector_2_degrees(y, z)
36
37
38 while True:
39     angle_xz, angle_yz = get_inclination(sensor)
40     print("XZ angle = {:.2f}deg   YZ angle = {:.2f}deg".format(angle_xz, angle_yz))
41     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 ° 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

[adafruit_mpu6050](#), 16

A

acceleration (*adafruit_mpu6050.MPU6050 attribute*), 17
accelerometer_range (*adafruit_mpu6050.MPU6050 attribute*), 17
adafruit_mpu6050 (*module*), 16

B

Bandwidth (*class in adafruit_mpu6050*), 16

C

cycle (*adafruit_mpu6050.MPU6050 attribute*), 17
cycle_rate (*adafruit_mpu6050.MPU6050 attribute*), 17

F

filter_bandwidth (*adafruit_mpu6050.MPU6050 attribute*), 17

G

gyro (*adafruit_mpu6050.MPU6050 attribute*), 17
gyro_range (*adafruit_mpu6050.MPU6050 attribute*), 17
GyroRange (*class in adafruit_mpu6050*), 16

M

MPU6050 (*class in adafruit_mpu6050*), 16

R

Range (*class in adafruit_mpu6050*), 17
Rate (*class in adafruit_mpu6050*), 18
reset () (*adafruit_mpu6050.MPU6050 method*), 17

S

sample_rate_divisor (*adafruit_mpu6050.MPU6050 attribute*), 17
sleep (*adafruit_mpu6050.MPU6050 attribute*), 17

T

temperature (*adafruit_mpu6050.MPU6050 attribute*), 17