25.MPU6050 Module

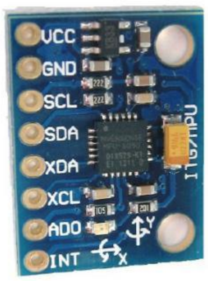
Introduction

The MPU-6050 is the world ' s first and only 6-axis motion tracking devices (3-axis Gyroscope and 3-axis Accelerometer) designed for smartphones, tablets and wearable sensors that have these features, including the low power, low cost, and high performance requirements.

In this lesson, use I2C to obtain the values of the three-axis acceleration sensor and three-axis gyroscope for MPU6050 and display them on the screen.

Hardware Required

- ✓ 1 * Raspberry Pi
- ✓ 1 * T-Extension Board
- ✓ 1 * MPU6050 Module
- ✓ 1 * 40-pin Cable
- ✓ Several Jumper Wires
- ✓ 1 * Breadboard



Principle

MPU6050

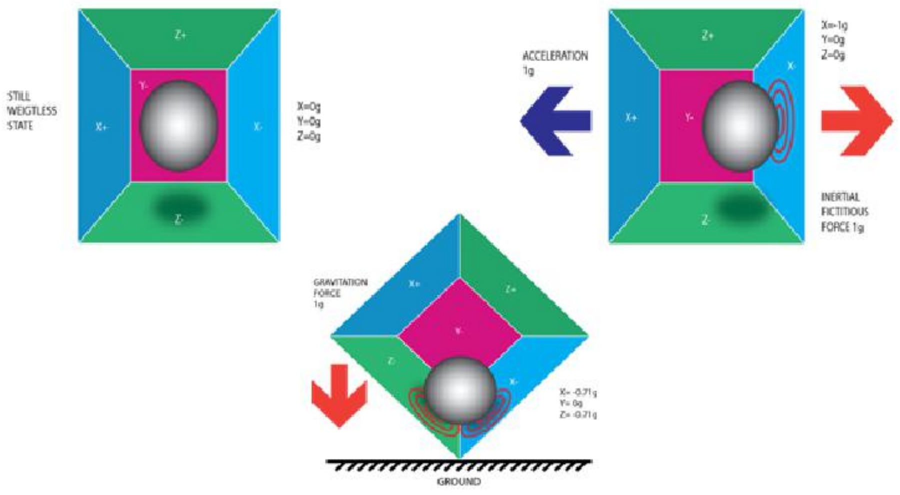
The InvenSense MPU-6050 sensor contains a MEMS accelerometer and a MEMS gyro in a single chip. It is very accurate, as it contains 16-bits analog to digital conversion hardware for each channel. Therefore it captures the x, y, and z channel at the same time. The sensor uses the I2C-bus to interface with the Raspberry Pi.

The MPU-6050 is not expensive, especially given the fact that it combines both an accelerometer and a gyro.

IMU sensors are one of the most inevitable type of sensors used today in all kinds of electronic gadgets. They are seen in smart phones, wearables, game controllers, etc. IMU sensors help us in getting the attitude of an object, attached to the sensor in three dimensional space. These values usually in angles, thus help us to determine its attitude. Thus, they are used in smart phones to detect its orientation. And also in wearable gadgets like the nike fuel band or fit bit, which use IMU sensors to track movement.

How does it work?

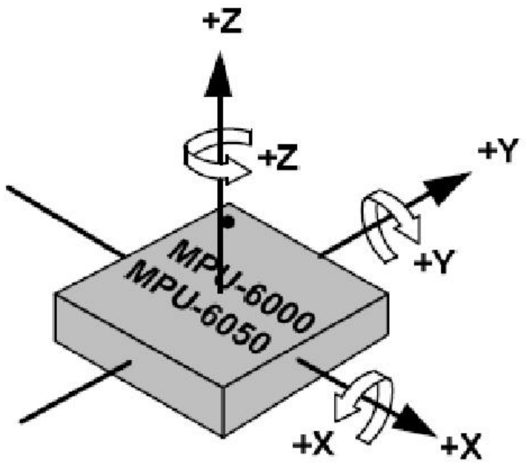
IMU sensors usually consists of two or more parts. Listing them by priority, they are : accelerometer, gyroscope, magnetometer and altimeter. The MPU 6050 is a 6 DOF (Degrees of Freedom) or a six axis IMU sensor, which means that it gives six values as output. Three values from the accelerometer and three from the gyroscope. The MPU 6050 is a sensor based on MEMS (Micro Electro Mechanical Systems) technology. Both the accelerometer and the gyroscope is embedded inside a single chip. This chip uses I2C (Inter Integrated Circuit) protocol for communication



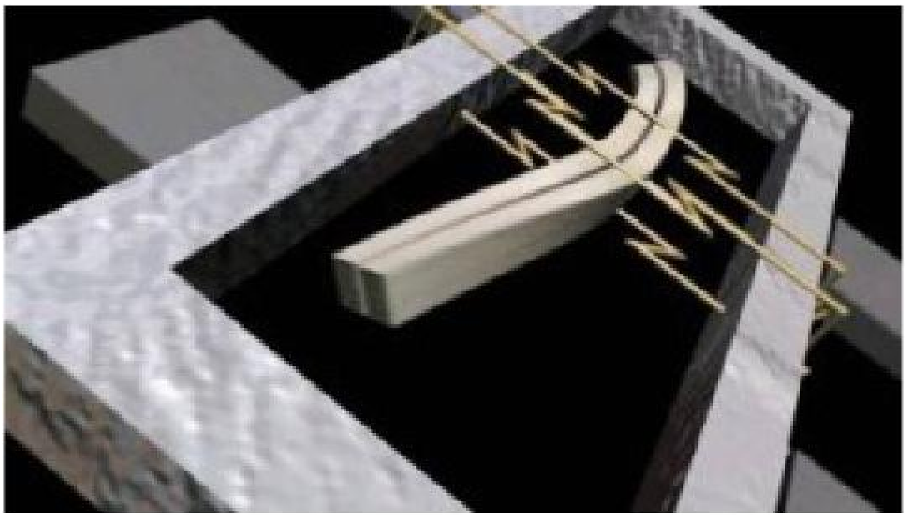
How does an accelerometer work?

An accelerometer works on the principle of piezo electric effect. Here, imagine a cuboidal box, having a small ball inside it, like in the picture above. The walls of this box are made with piezo electric crystals. Whenever you tilt the box, the ball is forced to move in the direction of the inclination, due to gravity. The wall with which the ball collides, creates tiny piezo electric currents. There are totally, three pairs of opposite walls in a cuboid. Each pair corresponds to an axis in 3D space: X, Y and Z axes. Depending on the current produced from the piezo electric walls, we can determine the direction of inclination and its magnitude. For more information check this.

25.MPU6050 Module



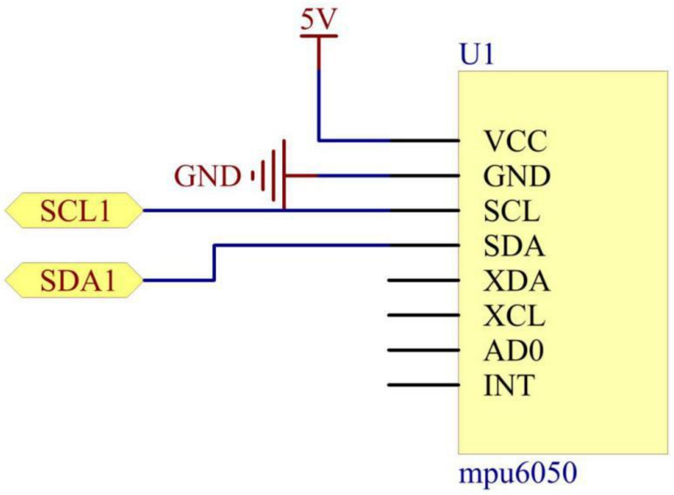
How does a gyroscope work?



Gyroscopes work on the principle of Coriolis acceleration. Imagine that there is a fork like structure, which is in constant back and forth motion. It is held in place using piezo electric crystals. Whenever, you try to tilt this arrangement, the crystals experience a force in the direction of inclination. This is caused as a result of the inertia of the moving fork. The crystals thus produce a current in consensus with the piezo electric effect, and this current is amplified. The values are then refined by the host microcontroller.

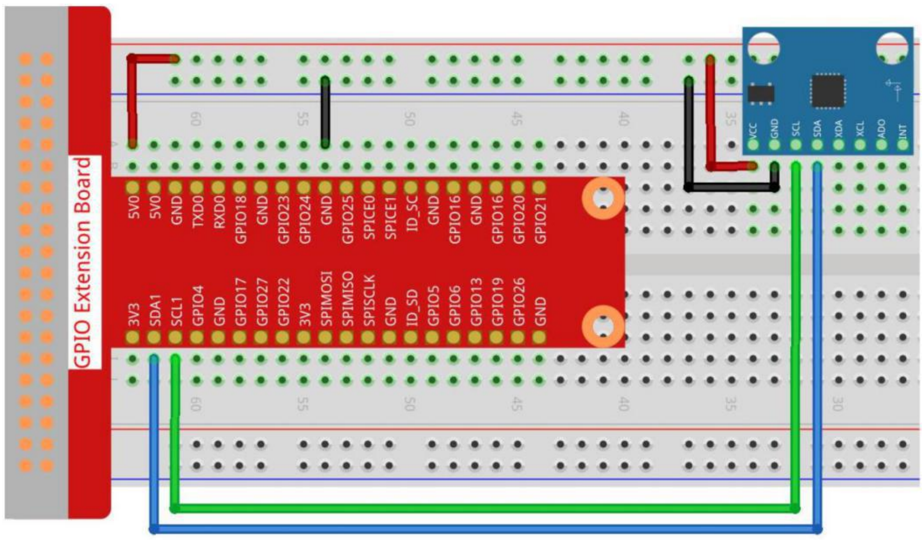
Schematic Diagram

T-Board Name	physical
SDA1	Pin 3
SCL1	Pin 5



Experimental Procedures

Step 1: Build the circuit.



Step 2: Setup I2C (see Appendix. If you have set I2C, skip this step.)

For C Language Users

Step 3: Go to the folder of the code.

```
cd /home/pi/REXQualis_Raspberry_Pi_Complete_Starter_Kit/C/25.MPU6050_Module
```

Step 4: Compile the code.

```
gcc 25.MPU6050_Module.cpp I2Cdev.cpp MPU6050.cpp -o MPU6050_Module.out -lwiringPi
```

Note: The program contains custom headers that are compiled when CPP files are compiled.

You can use this method to simplify the instruction.

```
gcc *.cpp -lwiringPi
```

Note: Here we use the wildcard (*), which causes all the files that conform to the format to be processed together.

Step 5: Run the executable file .

```
sudo ./MPU6050_Module.out
```

With the code run, the acceleration/angular velocity on each axis read by MPU6050 will be printed on the screen after being calculating.

Code

```
#include <stdio.h>
#include <stdint.h>
#include <unistd.h>
#include "I2Cdev.h"
#include "MPU6050.h"

MPU6050 mpu;          //instantiate a MPU6050 class object

int16_t ax, ay, az;         //store acceleration data
int16_t gx, gy, gz;         //store gyroscope data

void loop() {
        // read raw accel/gyro measurements from device
        mpu.getMotion6(&ax, &ay, &az, &gx, &gy, &gz);
        // display accel/gyro x/y/z values
        printf("Acceleration:\n");
        printf("X:%.2f g    Y:%.2f g    Z:%.2f
g\n",(float)ax/16384,(float)ay/16384,(float)az/16384);
        printf("Angular velocity:\n");
        printf("X:%.2f d/s Y:%.2f d/s Z:%.2f
d/s\n",(float)gx/131,(float)gy/131,(float)gz/131);
}

int main()
{
        mpu.initialize();          //initialize MPU6050
        // verify connection
        printf(mpu.testConnection() ? "" : "initialize failed\n");
        while(1){
                loop();
        }
        return 0;
}
```

Code Explanation

```
#include "I2Cdev.h"
#include "MPU6050.h"
```

These two files are open source files used to drive MPU6050 which makes it easy to use MPU6050. They read the values of the 3-axis Accelerometer and the 3-axis Gyroscope so that we can calculate the acceleration and angular velocity.

```
MPU6050 mpu;
```

Instantiate a MPU6050 class object.

```
        mpu.getMotion6(&ax, &ay, &az, &gx, &gy, &gz);
```

This function is used to obtain the accelerometer and gyro readings of the MPU6050.

```
        printf("X:%.2f g    Y:%.2f g    Z:%.2f
g\n",(float)ax/16384,(float)ay/16384,(float)az/16384);
```

ax /16384 is a simplified calculation that maps the reading from the value range to the acceleration range. The complete calculation is ax/65536*4. Refer to the principle section of this lesson for the relationship between ranges and value range.

```
    printf("X:%.2f d/s Y:%.2f d/s Z:%.2f
d/s\n",(float)gx/131,(float)gy/131,(float)gz/131);
```

gx /131 is a simplified version of the angular velocity calculation. The complete calculation is gx/65536 * 500.

```
    mpu.initialize();
```

This function is used to initialize the mpu.

```
mpu.testConnection()
```

This function is used to test whether MPU6050 works

For Python Language Users

Step 3: Go to the folder of the code.

```
cd /home/pi/REXQualis_Raspberry_Pi_Complete_Starter_Kit/Python/25.MPU6050_Module
```

Step 4: Run the executable file.

```
sudo python3 25.MPU6050_Module.py
```

With the code run, the acceleration/angular velocity on each axis read by MPU6050 will be printed on the screen after being calculating.

Code

The code here is for Python3, if you need for Python2, please open the code with the suffix py2 in the attachment.

```
#!/usr/bin/env python3


import MPU6050
import time


mpu = MPU6050.MPU6050()       #instantiate a MPU6050 class object
ac = [0]*3                    #store accelerometer data
gy = [0]*3                    #store gyroscope data
```

```
def setup():
    mpu.dmp_initialize()        #initialize MPU6050

def loop():
    while(True):
        ac = mpu.get_acceleration()          #get accelerometer data
        gy = mpu.get_rotation()                  #get gyroscope data
        print("Acceleration:\n")
        print("X:%.2f g    Y:%.2f g    Z:%.2f
g\n"%(ac[0]/16384.0,ac[1]/16384.0,ac[2]/16384.0))
        print("Angular velocity:\n")
        print("X:%.2f d/s Y:%.2f d/s Z:%.2f
d/s\n"%(gy[0]/131.0,gy[1]/131.0,gy[2]/131.0))
        time.sleep(0.1)

if __name__ == '__main__':          # Program start from here
    setup()
    try:
        loop()
    except KeyboardInterrupt:    # When 'Ctrl+C' is pressed,the program will exit.
        pass
```

Code Explanation

```
import MPU6050
```

Import the MPU6050 library, the library is used to drive MPU6050 that makes it easy for us to use MPU6050. They read the values of the 3-axis Accelerometer and the 3-axis Gyroscope so that we can calculate the acceleration and angular velocity.

```
mpu = MPU6050.MPU6050()
```

Instantiate a MPU6050 class object.

```
mpu.dmp_initialize()
```

This function is used to initialize the mpu6050.

```
ac = mpu.get_acceleration()        #get accelerometer data
gy = mpu.get_rotation()            #get gyroscope data
```

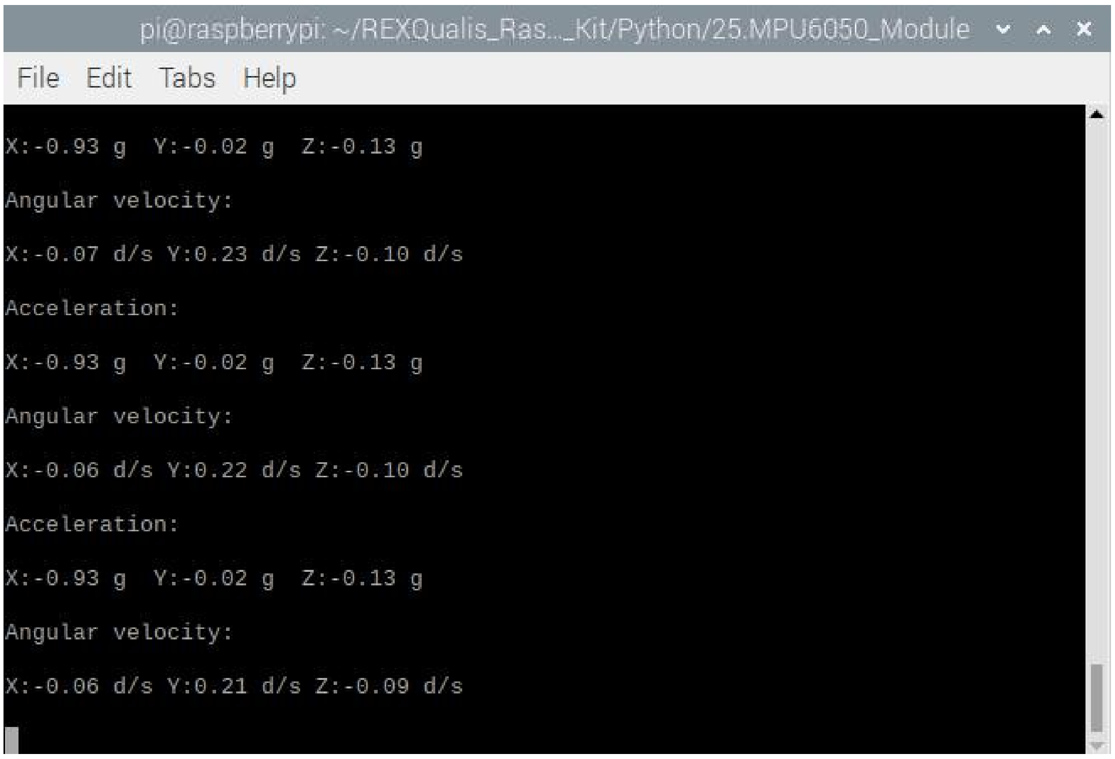
These two functions are used to obtain the accelerometer and gyro readings of the MPU6050.

```
print("X:%.2f g    Y:%.2f g    Z:%.2f g\n"%(ac[0]/16384.0,ac[1]/16384.0,ac[2]/16384.0))
```

ac [0]/16384.0 is a simplified calculation that maps the reading from the value range to the acceleration range. The complete calculation is ax/65536*4. Refer to the principle section of this lesson for the relationship between ranges and values.

```
print("X:%.2f d/s Y:%.2f d/s Z:%.2f d/s\n"%(gy[0]/131.0,gy[1]/131.0,gy[2]/131.0))
```

gy [0]/131.0 is a simplified version of calculating angular velocity, and the complete calculation is gx/65536 * 500.


pi@raspberrypi: ~/REXQualis_Ras..._Kit/Python/25.MPU6050_Module
File Edit Tabs Help

```
X:-0.93 g  Y:-0.02 g  Z:-0.13 g
Angular velocity:
X:-0.07 d/s Y:0.23 d/s Z:-0.10 d/s
Acceleration:
X:-0.93 g  Y:-0.02 g  Z:-0.13 g
Angular velocity:
X:-0.06 d/s Y:0.22 d/s Z:-0.10 d/s
Acceleration:
X:-0.93 g  Y:-0.02 g  Z:-0.13 g
Angular velocity:
X:-0.06 d/s Y:0.21 d/s Z:-0.09 d/s
```

Phenomenon Picture