Bitify

Tinkering with the Raspberry Pi and other geeky stuff

Thursday, 7 November 2013

Reading data from the MPU-6050 on the Raspberry Pi

In a previous post I showed how to connect an Accelerometer & Gyro sensor to the Raspberry Pi, in this post I'll show some simple Python code to read the data it offers.

To be able to read from the I^2C using Python bus we need to install the smbus module

```
sudo apt-get install python-smbus
```

Now to some code, this is just simple test code to make sure the sensor is working

```
#!/usr/bin/python

import smbus
import math

# Power management registers
power_mgmt_1 = 0x6b
power_mgmt_2 = 0x6c

def read_byte(adr):
    return bus.read_byte_data(address, adr)

def read_word(adr):
    high = bus.read_byte_data(address, adr)
    low = bus.read_byte_data(address, adr+1)
    val = (high << 8) + low
    return val

def read_word_2c(adr):
    val = read_word(adr)
    if (val >= 0x8000):
        return -((65535 - val) + 1)
    else:
        return val

def dist(a,b):
    return math.sqrt((a*a)+(b*b))

def get_y_rotation(x,y,z):
    radians = math.atan2(x, dist(y,z))
    return -math.degrees(radians)

def get_x_rotation(x,y,z):
    radians = math.atan2(y, dist(x,z))
    return math.degrees(radians)

bus = smbus.SMBus(0) # or bus = smbus.SMBus(1) for Revision 2 boards
address = 0x68       # This is the address value read via the i2cdetect command

# Now wake the 6050 up as it starts in sleep mode
bus.write_byte_data(address, power_mgmt_1, 0)

print "gyro data"
print "---------"

gyro_xout = read_word_2c(0x43)
gyro_yout = read_word_2c(0x45)
gyro_zout = read_word_2c(0x47)

print "gyro_xout: ", gyro_xout, " scaled: ", (gyro_xout / 131)
print "gyro_yout: ", gyro_yout, " scaled: ", (gyro_yout / 131)
print "gyro_zout: ", gyro_zout, " scaled: ", (gyro_zout / 131)

print
print "accelerometer data"
print "------------------"

accel_xout = read_word_2c(0x3b)
accel_yout = read_word_2c(0x3d)
accel_zout = read_word_2c(0x3f)

accel_xout_scaled = accel_xout / 16384.0
accel_yout_scaled = accel_yout / 16384.0
accel_zout_scaled = accel_zout / 16384.0

print "accel_xout: ", accel_xout, " scaled: ", accel_xout_scaled
print "accel_yout: ", accel_yout, " scaled: ", accel_yout_scaled
print "accel_zout: ", accel_zout, " scaled: ", accel_zout_scaled

print "x rotation: " , get_x_rotation(accel_xout_scaled, accel_yout_scaled,
accel_zout_scaled)
print "y rotation: " , get_y_rotation(accel_xout_scaled, accel_yout_scaled,
accel_zout_scaled)
```

When you run the code you will see output similar to this

```
gyro data
---------
gyro_xout:  -92  scaled:  -1
gyro_yout:  294  scaled:  2
gyro_zout:  -104 scaled:  -1

accelerometer data
------------------
accel_xout:  -3772  scaled:  -0.230224609375
accel_yout:  -52    scaled:  -0.003173828125
accel_zout:  15408  scaled:  0.9404296875
x rotation:  -13.7558411667
y rotation:  -0.187818934829
```



```
accel_yout = read_word_2c(0x3d)
accel_zout = read_word_2c(0x3f)
```

These three lines read the raw X,Y & Z accelerometer values, the parameter in each call is the register within the sensor that holds the data. The sensor has a number of registers which have different functionality as documented in this datasheet. The registers we are interested in for the acceleromter data are 0x3b, 0x3d, 0x3f and these hold the raw data in 16 bit two's complement format.

The following code reads a word (16 bits) from a given register and converts it from two's complement

```
def read_word_2c(adr):
    val = read_word(adr)
    if (val >= 0x8000):
        return -((65535 - val) + 1)
    else:
        return val
```

Once we have the raw data we need to scale it and then convert it into something useful like a rotation angle. Again from the data sheet we can see the default scaling we need to apply to the raw accelerometer values is 16384, so we divide the raw data by this value.

```
accel_xout_scaled = accel_xout / 16384.0
accel_yout_scaled = accel_yout / 16384.0
accel_zout_scaled = accel_zout / 16384.0
```

Now we have the values that gravity is exerting on the sensor in each of the three dimensions, from this we can calculate the rotations in the X & Y axes.

```
def dist(a,b):
    return math.sqrt((a*a)+(b*b))

def get_x_rotation(x,y,z):
    radians = math.atan(x / dist(y,z))
    return math.degrees(radians)

def get_y_rotation(x,y,z):
    radians = math.atan(y / dist(x,z))
    return math.degrees(radians)
```

Here is an excellent article showing the details behind the maths for this. What this gives us is the rotation angle in degrees for both the X & Y axes and is shown in the output.

```
x rotation: -13.755841166
y rotation: -0.187818934829
```

So in this instance the sensor is rotated by -13.7^{o} around X and -0.1^{o} around Y.

Gyroscope data

In a similar manner we can read the data from the Gyroscope part of the sensor. This is done in the following code

```
gyro_xout = read_word_2c(0x43)
gyro_yout = read_word_2c(0x45)
gyro_zout = read_word_2c(0x47)

print "gyro_xout: ", gyro_xout, " scaled: ", (gyro_xout / 131)
print "gyro_yout: ", gyro_yout, " scaled: ", (gyro_yout / 131)
print "gyro_zout: ", gyro_zout, " scaled: ", (gyro_zout / 131)
```

So we read the values from the registers 0x43, 0x45 & 0x47, again we can see from the datasheet that these hold the raw gyro data. To scale these we divide by 131 to give the degrees per second rotation value.

```
gyro_xout:  -92  scaled:  -1
gyro_yout:  294  scaled:  2
gyro_zout:  -104 scaled:  -1
```

thought I'd write a post on how to read the data from the R...

Temperature logging with a DS18B20 and a Raspberry Pi

I wanted to do some temperature logging so I hooked up a DS18B20 temperature sensor to a Raspberry Pi. About the DS18B20 Dallas DS18B...

Labels

1-wire (1)

ADXL345 (1)

BMP085 (2)

GY80 (1)

HMC5883L (3)

L3G4200D (1)

MPU-6050 (6)

OpenGL (2)

Python (8)

Raspberry Pi (8)

Raspbian (4)

temperature (1)

Blog Archive

► 2014 (2)

▼ 2013 (7)

► December (1)

▼ November (6)

Interfacing a BMP085 Digital Pressure sensor to th...

Connecting and calibrating a HMC5883L Compass on t...

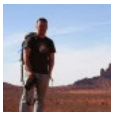
Using a complementary filter to combine Accelerome...

3D OpenGL visualisation of the data from an MPU-60...

Reading data from the MPU-6050 on the Raspberry Pi...

Interfacing Raspberry Pi and MPU-6050

About Me



Andrew Birkett

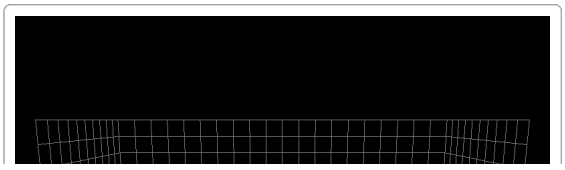
View my complete profile

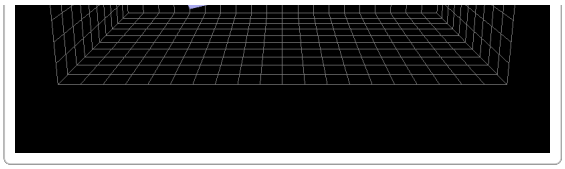
The output in my case show the gyro wasn't moving when I took reading.

Final thoughts

The code I present here is very basic and should be extended to handle errors and allow the sensor to be configured with different sensitivity levels. I've done this in my application and embedded it into a web server. This allows me to make a simple http request to the Raspberry Pi and get a reading from the sensor.

To help me test and visualise the data better I've written some simple OpenGL code to graphically represent the sensor's orientation in 3D space.





This OpenGL code runs on my Linux desktop machine and queries the Pi periodically to get the data and renders the above image. See this post for details how

In the next article I'll show how to combine the accelerometer and gyroscope data together to get a more accurate reading and help reduce noise.

Posted by Andrew Birkett at 20:39

Labels: MPU-6050, OpenGL, Python, Raspberry Pi, Raspbian

Walker Snow 2 years ago (edited) - Shared publicly
My MPU-6050 always says "Device or resource busy" when i use the command i2cget, as a result of these your python code doesn't work. What could be the problem ?
I also want to add that i2cdetect only detect a device at 0x3b but the number at that address isn't something like 68, it is UU.

1 · Reply

View all 6 replies

Walker Snow 1 year ago
+Trevor Allen
You need to pull-down the ADO pin to ground for giving the mpu6050 correct addressing(0x68).
Source:http://forum.arduino.cc/index.php?topic=103408.30

Trevor Allen 1 year ago
Thanks, Karda!

Wim

1 · Reply

Andrew Birkett 1 month ago
It's linked in the above article :) http://blog.bitify.co.uk/2013/11/3d-opengl-visualisation-of-data-from.html

Keith Ellis 1 year ago - Shared publicly
Thanks for the tutorial, I was thinking of using this gyro so I could determine when on object had rotated through a set angle, say 45 degrees. Could you point me in the right direction please.

1 · Reply

Andrew Birkett 1 year ago
Try this post http://blog.bitify.co.uk/2013/11/using-complementary-filter-to-combine.html it shows how to get more accurate angles from the sensor.

Gustavo Humeres Garcés 1 year ago - Shared publicly

when i try to execute the code i have problem with bus.write_byte_data(address, power_mgmt_1, 0).
IOerror=[errno5] input/output error.
can you help me please

+2 1 · Reply

Andrew Birkett 1 year ago
Make sure you run the program with sudo, if that doesn't help you might have a damages sensor, also check Simon Nobes comments below.

Jacob M 1 year ago - Shared publicly
Hey excellent post - best I have found yet. Everything is working for me, but I am wondering how I would go about adding rotation on the the Z axis?

1 · Reply

Andrew Birkett 1 year ago
For that you need a compass http://blog.bitify.co.uk/2013/11/connecting-and-calibrating-hmc5883l.html

Simon Nobes 1 year ago - Shared publicly
Hi Andrew, hope you're still monitoring this (very helpful) blog . . .

I have followed the previous tutorial to this one and the response to your test (sudo i2cdetect -y) confirms that the address of my MPU-6050 board is 0x68. I assume therefore that I have correctly installed all the necessary I2C libraries etc on my Raspberry Pi.

+1 1 · Reply

View all 3 replies

Simon Nobes 1 year ago (edited)
Thanks Andrew. I did a bit more reading around, followed your link and started again from scratch, including pulling the IMU off the RPi. In short, I re-assembled the connections and I now have reliable output, so either I missed something in the setup or I had my wires crossed! (I told you I was new at this . . .)

Andrew Birkett 1 year ago
Glad to hear you resolved it :)

Shres L 2 years ago - Shared publicly
Hi Andrew,
Thanks a lot for the tutorial.Im new to RPi, so to run the Python code, should I have to copy and paste the code in idle on my RPi and run it. Is that all. Please advise.Thank you.

1 · Reply

Andrew Birkett 2 years ago
Hi, Yes that should be fine as long as you make sure it runs as root. Usually you would save it to a file and then run the file rather than run it from something liek Idle.

Poul Dürr Pedersen 2 years ago - Shared publicly

Andrew Birkett 2 years ago
Thanks for spotting that, I've corrected the typo.

Ankit Bhadoria 1 year ago - Shared publicly
i am reading all zeros.... even though device is detected. what could be the issue ?

1 · Reply

Andrew Birkett 1 year ago
It sounds like the device isn't coming out of sleep mode.

Florian Pieper 1 year ago - Shared publicly
Great tutorial! But i have some problems...Maybe you can help me.
The I2C connection is working fine, i2cdetect finds the device, and i can read out and set the Power management bit. But if i try to read out the gyro and accelerometer values, i only get zeros, even with the sleep mode disabled.
Sry for my bad english...it's my second language. Thanks!

1 · Reply

Andrew Birkett 1 year ago
I've not seen that problem before, are you sure the device is powering up OK ? have you tried reading from the device with the i2cget command from the PI to see if that works?

Marco Massaro 1 year ago - Shared publicly
Thanks a lot for the tutorial. Only a question: In this way what is the default value for the gyro's sample rate when you call gyro_xout = read_word_2c(0x43) ? Thanks

1 · Reply

Andrew Birkett 1 year ago
I'm not sure what the default value actually is. I couldn't see it in the datasheet and I have never bothered to read from the device to see what value it is.

Colin Addison 2 years ago - Shared publicly
Hi Andy,
Great piece of work for getting us Noobs up & running.
Quite a learning curve!
I have got as far as “Reading Data from the MPU-6050”.

1 · Reply

View all 3 replies

Shmulik E 1 year ago (edited)
I got this error too, i am using b+ model and i used:
bus = smbus.SMBus(1)
and still getting this error.

Shmulik E 1 year ago
problem solved !
need to add "sudo" to geany - set Build commands

https://learn.sparkfun.com/tutorials/raspberry-gpio/using-an-ide

Scott Walker 1 year ago - Shared publicly

Hi thanks for the tutorial! One problem: the scaled gyro data for y and z when still shows around 0, as expected. But the x value always shows around 26 when perfectly still. Is this normal? Would this throw the complimentary filter off when I start to move on to balancing?

1 · Reply

Carlos Andres Gutierrez Valdes 1 year ago - Shared publicly

it an excellent tutorial, I have one question, I tried to measure the distance with the accelerometer on the Y axis that is expressed in g (9.81 m/s^2) but the data sheet says the sensor have a margin of +-2g, there is a way to measure cm? I want to do an sequence every 3 cm

Thanks!!

1 · Reply

a solid object comes in contact with an upright beam. The MPU6050 will be attached to the top of the beam. When contact is detected (I though I would + or - acceleration values on x and y axes) I then need to light up a cluster of LEDs (switched with GPIO pin26). To do this can I embed the switching code within your code or do I need to use a call-out using something like the Subprocess module (after

1 · Reply

Omar Aflak 5 months ago - Shared publicly

Hi! Just to say I made a library for the GY-521 in C++.

If some of you are still interested, this is the tutorial I made:

https://en.causeyourestuck.io/2015/12/28/gyroscope-gy-521-on-raspberry-pi/

1

Show more

Newer Post Home Older Post

Subscribe to: Post Comments (Atom)