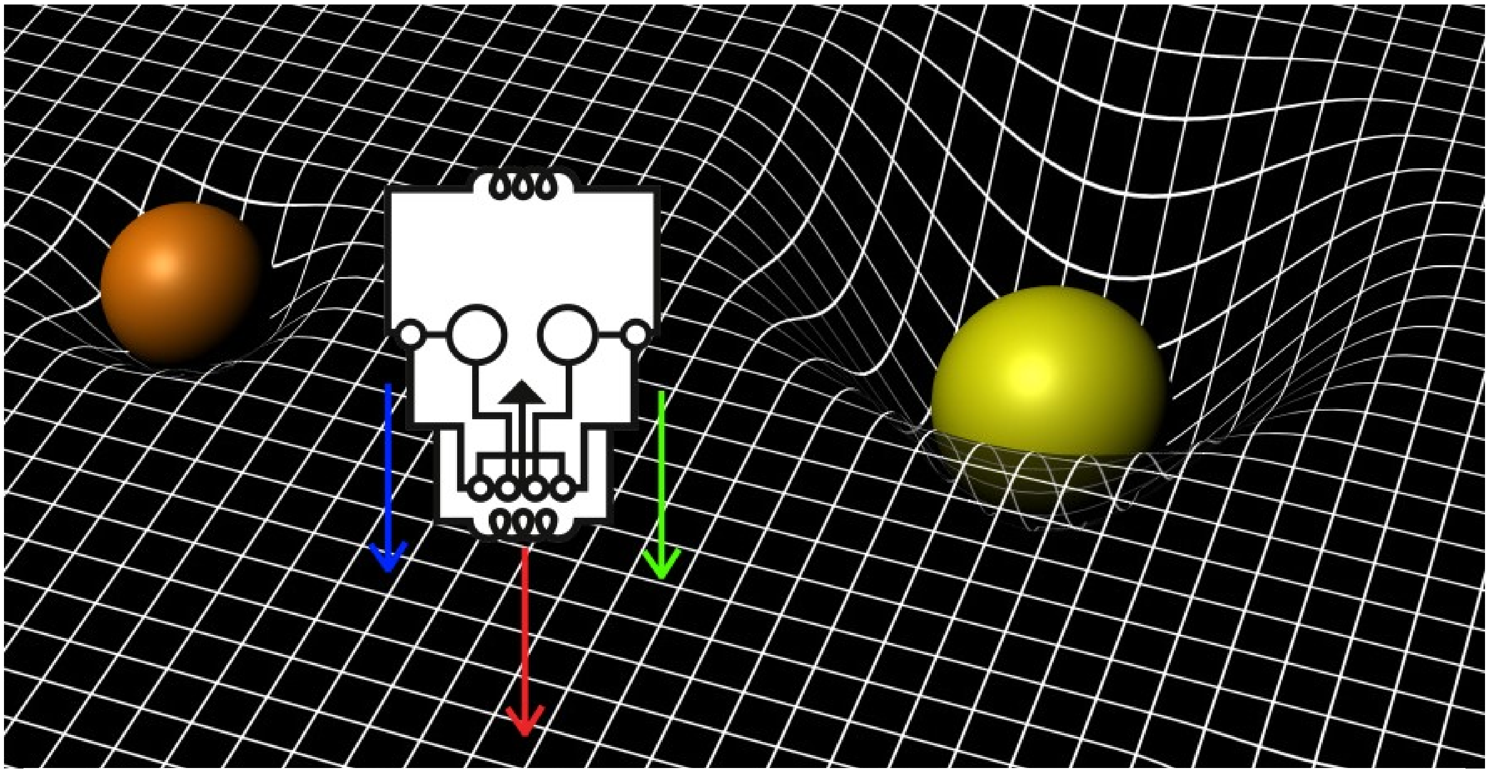
ozzmaker.com



BerryIMU

CONVERTING VALUES FROM AN ACCELEROMETER TO G

| Mark Williams | 34 Comments

In this post I will show how to convert the raw values read from an accelerometer to 'Gs'.

Git repository here
The code can be pulled down to your Raspberry Pi with;

```
pi@raspberrypi ~ $ git clone https://github.com/ozzmaker/BerryIMU.git
```

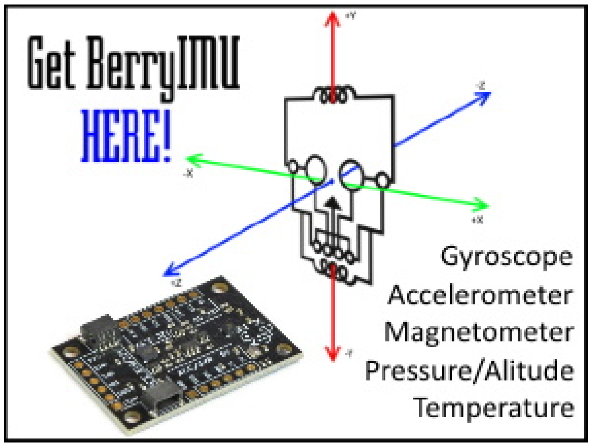
The code for this guide can be found under the python-BerryIMU-measure-G directory.

An accelerometer measures **proper acceleration**, which is the acceleration it experiences relative to freefall. This is most commonly called "G-Force" (G)

For example, an accelerometer at resting on a table would measure 1G (9.81 m/s2) straight upwards. By contrast, accelerometers in free fall and accelerating due to the gravity of Earth will measure zero.

The accelerometer used by the BerryIMU is a MEMS sensors(LSM9DS0), which outputs the raw readings as mg/LSB.
Most MEMS accelerometers use this output format.

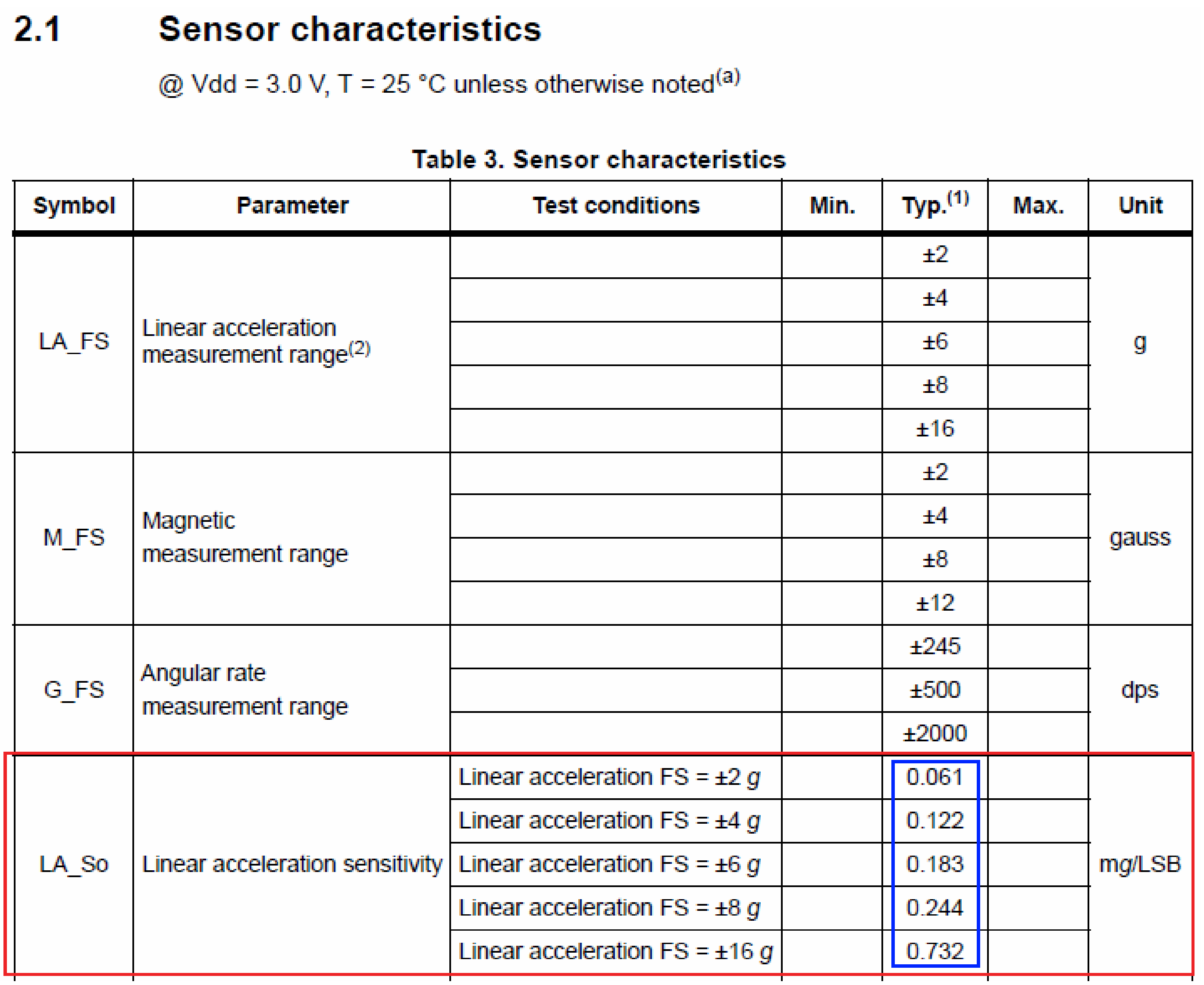
mg = milli-G's (just like milliliters)
1mG = 0.001 G's of acceleration, so 1000mG = 1G.
LSB = Least Significant bit, which is the last bit on the right.



The LSM9DS0 outputs a 16 bit value for the accelerometer readings.

If you look at the characteristics of the LSM9DS0 in the datasheet, you can see the sensitivity levels for the accelerometer highlighted in red below and the corresponding values for the **LSB**, which are highlighted in blue. *You can download the datasheet here;*

The raw values from the accelerometer are multiplied by the sensitive level to get the value in G.



2.1 Sensor characteristics

@ Vdd = 3.0 V, T = 25 °C unless otherwise noted[(a)]

Table 3. Sensor characteristics

Symbol	Parameter	Test conditions	Min.	Typ.[(1)]	Max.	Unit
LA_FS	Linear acceleration measurement range[(2)]			±2		g
				±4		
				±6		
				±8		
				±16		
M_FS	Magnetic measurement range			±2		gauss
				±4		
				±8		
				±12		
G_FS	Angular rate measurement range			±245		dps
				±500		
				±2000		
LA_So	Linear acceleration sensitivity	Linear acceleration FS = ±2 *g*		0.061		mg/LSB
		Linear acceleration FS = ±4 *g*		0.122		
		Linear acceleration FS = ±6 *g*		0.183		
		Linear acceleration FS = ±8 *g*		0.244		
		Linear acceleration FS = ±16 *g*		0.732		

Let's use FS ±2 g as an example sensitivity level. As the range is -2 to +2, this would be a total of 4g. Or 4,000 Milli-Gs.
The output is 16 bits. 16 bits equals 65,535. This means we can get 65,535 different readings for the range between -2 and +2. (or -2,000 MilliGs and +2,000 MilliGs)

4,000 MilliGs / 65,535 = 0.061

Each time the LSB changes by one, the value changes by 0.061, which is the value highlighted in blue in the table above.

For FS ±8 g, the range would be -8 to +8, which is a total of 16,000 MilliGs.
16,000 MilliGs / 65,535 = 0.244

Example when using ±2g sensitivity
In the table below, every time the raw values increments by one, the final calculated value(which is MilliG) increments by 0.061

RAW	BINARY	LSB value for +/-2G	Calc MilliG
16	10000	0.061	0.976
17	10001	0.061	1.037
18	10010	0.061	1.098

The above values of 16,17 and 18 above a very low and only used for illustration.
If your accelerometer is horizontal and resting and at rest when using a sensitive level of ±2g, the raw value for Z should hover around 16,500.
16,500 X **0.061** = **1006** MilliGs or **1**G

Example when using ±8g sensitivity
In the table below, every time the raw values increments by one, the final calculated value(which is MilliG) increments by 0.244

RAW	BINARY	LSB value for +/-2G	Calc MilliG
16	10000	0.244	3.904
17	10001	0.244	4.148
18	10010	0.244	4.392

If you accelerometer is horizontal and at rest, when using a sensitive level of ±8g, the raw value for Z should hover around 4,475.

4,175 X **0.244** = **1018.7** MilliGs or **1**G

The Code

The two above examples are easy to implement in python;
±8g Sensitivity

```python
writeACC(CTRL_REG2_XM, 0b00010000) #+/- 8G full scale
print("G Value for Z axis %f G" % ((ACCz * 0.244)/1000))
```

The first line above is used to initialise the accelerometer with a sensitivity level of ±2g.
The second line prints the calculated value as G using the raw values from the accelerometer.

±2g Sensitivity

```
writeACC(CTRL_REG2_XM, 0b00000000) #+/- 2G full scale
print("G Value for Z axis %f G" % ((ACCz * 0.061)/1000))
```

The first line above is used to initialise the accelerometer with a sensitivity level of ±2g.
The second line prints the calculated value as G uses using raw values from the accelerometer.
Below is a snippet from the main program;

```
import smbus
import time
import math
from LSM9DS0 import *
import datetime
bus = smbus.SMBus(1)


def writeACC(register,value):
        bus.write_byte_data(ACC_ADDRESS , register, value)
        return -1


def readACCx():
        acc_l = bus.read_byte_data(ACC_ADDRESS, OUT_X_L_A)
        acc_h = bus.read_byte_data(ACC_ADDRESS, OUT_X_H_A)
    acc_combined = (acc_l | acc_h <<8)
    return acc_combined  if acc_combined < 32768 else acc_combined - 655

def readACCy():
        acc_l = bus.read_byte_data(ACC_ADDRESS, OUT_Y_L_A)
        acc_h = bus.read_byte_data(ACC_ADDRESS, OUT_Y_H_A)
    acc_combined = (acc_l | acc_h <<8)
    return acc_combined  if acc_combined < 32768 else acc_combined - 655

def readACCz():
        acc_l = bus.read_byte_data(ACC_ADDRESS, OUT_Z_L_A)
        acc_h = bus.read_byte_data(ACC_ADDRESS, OUT_Z_H_A)
    acc_combined = (acc_l | acc_h <<8)
    return acc_combined  if acc_combined < 32768 else acc_combined - 655



#initialise the accelerometer
writeACC(CTRL_REG1_XM, 0b01100111) #z,y,x axis enabled, continuos update
writeACC(CTRL_REG2_XM, 0b00011000) #+/- 8G full scale

while True:


    #Read the accelerometer,gyroscope and magnetometer values
    ACCx = readACCx()
    ACCy = readACCy()
    ACCz = readACCz()
    print("##### X = %f G  #####" % ((ACCx * 0.244)/1000)),
    print(" Y =   %fG  #####" % ((ACCy * 0.244)/1000)),
    print(" Z =  %fG  #####" % ((ACCz * 0.244)/1000))


    #slow program down a bit, makes the output more readable
    time.sleep(0.03)
```

Guides and Tutorials

- Guide to interfacing a Gyro and Accelerometer with a Raspberry Pi
- Guide to interfacing a Gyro and Accelerometer with a Raspberry Pi - Kalman Filter
- Create a Digital Compass with the Raspberry Pi – Part 1 – “The Basics”
- Create a Digital Compass with the Raspberry Pi – Part 2 – “Tilt Compensation”
- Create a Digital Compass with the Raspberry Pi – Part 3 – “Calibration”
- Create a Digital Compass with the Raspberry Pi – Part 4- “Smartphone Replica”
- Using the BerryIMUv3 on a Raspberry Pi Pico
- Double tap detection with BerryIMUv3
- Connect BerryIMUv3 via SPI

BERRYIMU

34 THOUGHTS ON “CONVERTING VALUES FROM AN ACCELEROMETER TO G”

nuketard

JANUARY 8, 2017 AT 1:43 AM

What filter would be appropriate for the accelerometer data? It doesn't drift much, and at most is slightly offset(from mounting) which can easy be corrected with calibration. But the signal is quite noisy. I was thinking of using a low pass filter or a median filter. Are there more suitable alternatives or a worked example?

★ **Mark Williams**

JANUARY 8, 2017 AT 11:44 AM

Both filters would work and would do a good job.
Note that both would add a delay, which is in most cases negligible.
-A median filter would have a constant delay. It is relative to the table size
-A low pass filter would have a variable delay. It is relative to how much noise there is and you filter factor

I have code for two of them below. It is being used for compass, but you can just change it to the accelerometer values.

```
//Low Pass
float kLowPassFilterFactor = 0.1;
float kHighPassFilterFactor = 0.1;

*mag_raw = *mag_raw * kLowPassFilterFactor + oldXMagRawValue*(1 -
kLowPassFilterFactor);
*(mag_raw+1) = *(mag_raw+1) * kLowPassFilterFactor + oldYMagRawValue*(1 -
kLowPassFilterFactor);
*(mag_raw+2) = *(mag_raw+2) * kLowPassFilterFactor + oldZMagRawValue*(1 -
kLowPassFilterFactor);

oldXMagRawValue = *mag_raw;
oldYMagRawValue = *(mag_raw+1);
oldZMagRawValue = *(mag_raw+2);

//Median
#define MEDIANTABLESIZE 3

for (f = MEDIANTABLESIZE-1; f > 0 ; f--){
medianTable1.x[f] = medianTable1.x[f-1];
medianTable1.y[f] = medianTable1.y[f-1];
medianTable1.z[f] = medianTable1.z[f-1];
}
medianTable1.x[0] = *mag_raw;
medianTable1.y[0] = *(mag_raw+1);
medianTable1.z[0] = *(mag_raw+2);

medianTable2 = medianTable1;

qsort(medianTable2.x, MEDIANTABLESIZE, sizeof(int), cmpfunc);
qsort(medianTable2.y, MEDIANTABLESIZE, sizeof(int), cmpfunc);
qsort(medianTable2.z, MEDIANTABLESIZE, sizeof(int), cmpfunc);

*mag_raw = medianTable2.x[(int)MEDIANTABLESIZE/2];
*(mag_raw+1) = medianTable2.y[(int)MEDIANTABLESIZE/2];
*(mag_raw+2) = medianTable2.z[(int)MEDIANTABLESIZE/2];

int cmpfunc (const void * a, const void * b) {
return ( *(int*)a - *(int*)b );
```

```
    }
```

izaq09

FEBRUARY 21, 2017 AT 1:24 AM

Hi,

Thank you very much for your guide. It has provided me a lot of help.

For ±16 g, 32,000 MilliGs / 65,535 = 0.488, which is not same as the linear acceleration sensitivity (0.732) in the datasheet. In this case, which one should we use? The one obtained through calculation or the sensitivity in datasheet?

In fact, I had found something similar in your guide in interfacing gyroscope and accelerometer. In that case, you are using sensitivity.
https://ozzmaker.com/berryimu/

Thank you.

Regards,
Izaq

★ Mark Williams

FEBRUARY 21, 2017 AT 12:51 PM

you have ±16 g = 32,000 MilliG. I should be ±16 g = 16,000 MilliG.

Does that answer your question?

N

MARCH 12, 2017 AT 6:30 AM

Hey Mark,

How come for ±16 g, it's 16,000 MilliG, when for ±8 g, it's 16,000 MilliG...?

Thanks.
N

★ Mark Williams

MARCH 12, 2017 AT 9:52 PM

My comment above was incorrect. I got confused my my own math 🙂

I have to look into to why 0.732 is used for ±16 g.

Rory

MAY 5, 2018 AT 3:49 AM

Did you ever discover why 0.732 is used?

★ Mark Williams

MAY 5, 2018 AT 11:54 AM

Yes and No 🙂

I spoke to ST who make the LSM9DS1, they told me that not all 16 bits are used for ±16 G.
I asked how many are used and they told me they couldn't give me this information as it is proprietary.

Noel

MAY 16, 2018 AT 10:36 AM

Another way to explain the 0.732 is that the widest range is not ±16 g, but ±24 g !!!

Yifan Jia

MAY 24, 2017 AT 7:23 PM

May I ask the fastest frequency of getting the data, I found the sample code can provide data around 100Hz sampling rate, may I increase it to 1000Hz?

Thanks

★ Mark Williams

MAY 24, 2017 AT 8:53 PM

The output data rate is configured using the first 4 bits in the register below;

```
writeACC(CTRL_REG1_XM, 0b01100111) #z,y,x axis enabled, continuos update,
100Hz data rate
```

0110 = 100Hz

to set it to 1600Hz, sit it to 1010

```
writeACC(CTRL_REG1_XM, 0b10100111) #z,y,x axis enabled, continuos update,
100Hz data rate
```

This is documented on page 55 and 56 of the datasheet.

Yifan
MAY 26, 2017 AT 6:22 PM
Thank you! This is very helpful!

Eric Olson
JULY 4, 2017 AT 8:24 AM
In the example code (python-berryIMU-measure-G) the value .224 is used for the 8G scale. This value should be .244 per the table included in this article. Couldn't figure out why I wasn't reading 1G at level with no motion so double-checked code.

That said, great job. You've made the process of deploying one of these units as a vibration sensor in a large scale environment exceptionally easy. Saved me at least fifty hours with your efforts and this product. Thank you very much! 🙂

★ Mark Williams
JULY 4, 2017 AT 4:45 PM
thanks.. I'll get that fixed

RSC
JULY 7, 2017 AT 2:22 PM
excellent article.. brilliant..

Vishal
SEPTEMBER 27, 2017 AT 9:06 PM
hi, Will u plz help me how to convert X,Y,Z axis raw data of LIS3DH to acceleration and how to detect sudden acceleration and harsh breaking (deacceleration) in car.

Will
FEBRUARY 24, 2018 AT 9:02 PM
Why are you using "G" and "g" in the beginning? Are they not the same?

Payel
FEBRUARY 25, 2018 AT 7:10 AM

can you please explain why we are shifting value 8 bits left and then adding it.Below lines
def readACCx():
acc_l = bus.read_byte_data(ACC_ADDRESS, OUT_X_L_A)
acc_h = bus.read_byte_data(ACC_ADDRESS, OUT_X_H_A)
acc_combined = (acc_l | acc_h <<8)

return acc_combined if acc_combined < 32768 else acc_combined - 65536

PeterP

FEBRUARY 25, 2018 AT 10:19 AM

you need to shift to get the 16 bit value from two 8 bit values.
Each axis is represented as two bytes by the IMU. You read the high and low bytes from one axis. E.g X for the accelerometer. And then combine them. shifting and OR is needed to combine them
X_Low = 212 or 11010100 in binary
X_High = 25 or 00011001 in binary

`X_axis = (int16_t)(X_Low | X_High << 8); 6,612 axis = (int16_t)(212 | 25 << 8);` or in binary 00011001 11010100 = (11010100 | 00011001 <<8) You will see that 00011001 is now the first part of the 16 bit value and 11010100 is the second half of the 16 bit value.

JAMES

APRIL 11, 2018 AT 1:21 AM

You said that at resting the raw values should be as below:

+-2g - z = 16500
+-8g - z = 4475

Which is similar to what I have at resting. For x/y, I'm getting the below:

+-2g - x = 65100
+-8g - x = 65420

+-2g - y = 524
+-8g - y = 128

Are these values close to the average, or is my x-values completly wrong?

★ Mark Williams

APRIL 11, 2018 AT 2:28 PM

if the IMU is flat.. you should only be getting large values on Z. you may get some small values on X and Y, this would be because the IMU isn't absolutely horizontal.

Your X vales look incorrect. Are you using unmodified code from the git repo?

Aaron

NOVEMBER 13, 2018 AT 11:15 AM

Thank you sir, your work and time spent on this guide and the IMU/gps boards is appreciated. I have learned a lot. There is enough information here to get started, that is extremely valuable to using something! Seems like people always forget that... anyway, thanks!

Rocket Man

FEBRUARY 18, 2019 AT 12:03 PM

Is the Max G's of Acceleration a known quantity of what this sensor can detect? I'm trying to use it to collect data in my high powered rocket going supersonic, and I will probably need something that can sense an acceleration of up to 20 G's of force

---

★ Mark Williams

FEBRUARY 18, 2019 AT 5:17 PM

Yes, our accelerometer can measure up to 16G

Ravindra kumar

FEBRUARY 28, 2019 AT 6:37 PM

how i can detect the orientation and static tilt

Ahmed

APRIL 13, 2019 AT 12:30 AM

writeACC(CTRL_REG2_XM, 0b00000000) #+/- 2G full scale
print("G Value for Z axis %f G" % ((ACCz * 0.061)/1000))
****

Why divide ACCz * 0.061 by 1000 ??
****

Madhavi Gupta

MAY 13, 2019 AT 7:04 PM

Hi,
I am using ADXL335 accelerometer, I am getting the values but after some time, the sensor gives maximum value even if it is not moving. For eg,
1678 1987 4056
...(likewise 8-9 readings)
4058 290 1679
this happens after every 10-12 readings
Can you tell me why is this happening?

Markel Robregado

MAY 30, 2019 AT 3:27 AM

Hi, I am using LSM303AGR 2G normal mode 10 bit. So each bit will be 4000mg/ 2^10 = 3.90625 is this right?

Ben in Seattle

FEBRUARY 25, 2020 AT 4:55 PM

How common is it for accelerometer manufacturers to pull tricks like ST did for the ±16G range where they list the LSB as .732 when it should be .488?

The calculation would be simple to understand and wouldn't require a datasheet if we could just scale the range the normal way: multiply by the max possible output (e.g., 4G) and divide by max possible input (e.g., 32768, for n=16 bits).

$f(x) = x \times \text{rangemax} \div 2^{(n-1)}$

What is with the LSB malarkey? Is it to deal with the fact that two's complement binary numbers have one less negative number than positive? If so, the difference seems so minimal (<.0015%), that there is more error from the datasheet rounding to three places of precision.

Also, I agree with Noel that the range should have simply read ±24G since 24 ÷ 32768 = .000732421875. I suspect errata.

armin

JUNE 22, 2020 AT 11:55 PM

Hey there,

which unit does the raw data has? (mg/lsb?)
does that mean that I only have to multiply my raw data which i get from my accelerometer with the corresponding value (e.g: 0.183) to get the maximum acceleration?

greetings, armin

Bosley

AUGUST 18, 2021 AT 4:25 PM

Hi!

I'm looking to purchase this GPS/IMU for a project and I have a minor question. For the python examples listed here I see that they work for the BerryGPS v1-3. Will these also work for the v4? Or is there an updated version of these example scripts for the current iteration of the GPS/IMU?

Thanks!

★ Mark Williams

AUGUST 18, 2021 AT 8:34 PM

The python code works for **BerryIMUv3** and **BerryGPS-IMUv4**. These two devices use the same IMU.

Eoin

MARCH 16, 2022 AT 1:51 AM

Just wondering if I have a value of 4000 raw data from accelerometer how I convert to m/s^2 please.

Selim Sagir

JUNE 30, 2022 AT 6:23 PM

Hi,
Great explanation, thanks.
The point I wonder, is there any way to calculate resting values such as, 16500 for the ±2g. My sensor also gives the same value, but I don't understand the logic behind it.

Thanks!