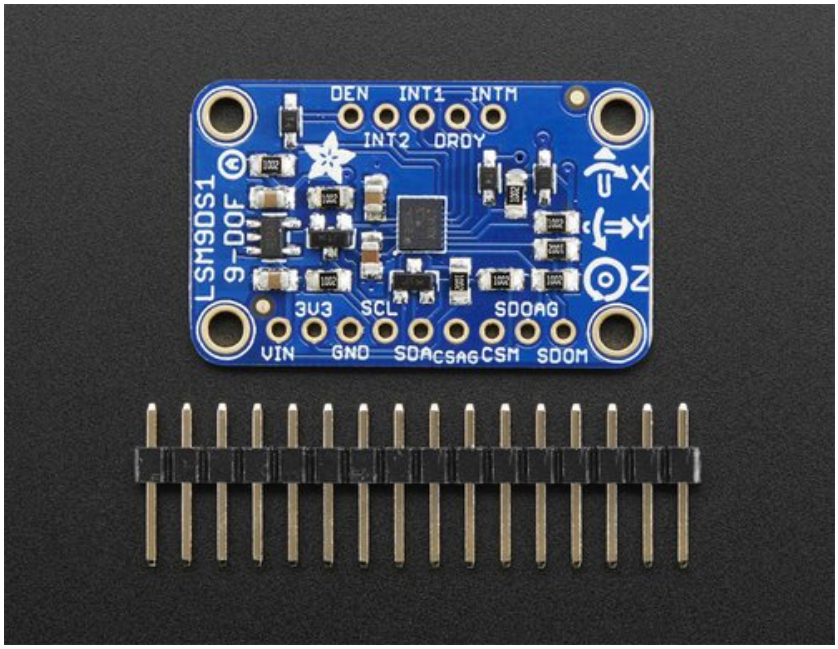
# Adafruit LSM9DS1 Accelerometer + Gyro + Magnetometer 9-DOF Breakout

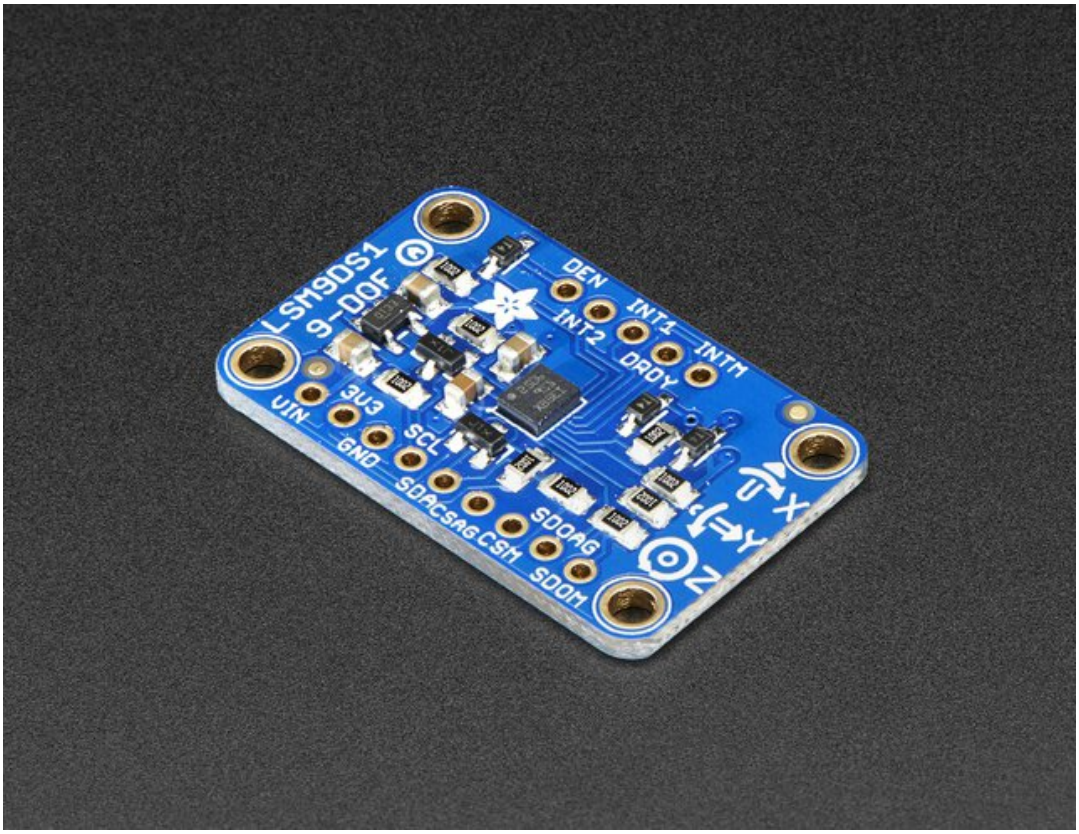
Created by lady ada



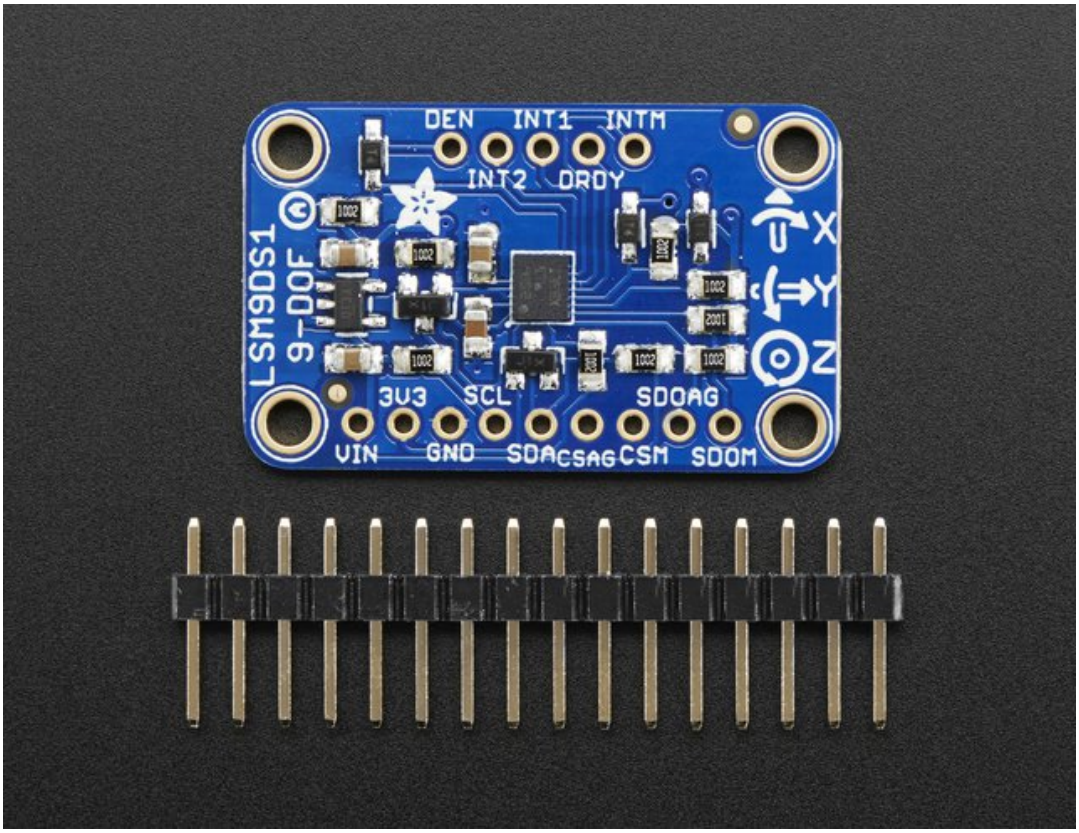

Last updated on 2017-02-02 01:50:59 AM UTC

# Guide Contents

# Overview




Add motion, direction and orientation sensing to your Arduino project with this all-in-one 9-DOF sensor. Inside the chip are three sensors, one is a classic 3-axis accelerometer, which can tell you which direction is down towards the Earth (by measuring gravity) or how fast the board is accelerating in 3D space. The other is a 3-axis magnetometer that can sense where the strongest magnetic force is coming from, generally used to detect magnetic north. The third is a 3-axis gyroscope that can measure spin and twist. By combining this data you can REALLY orient yourself.



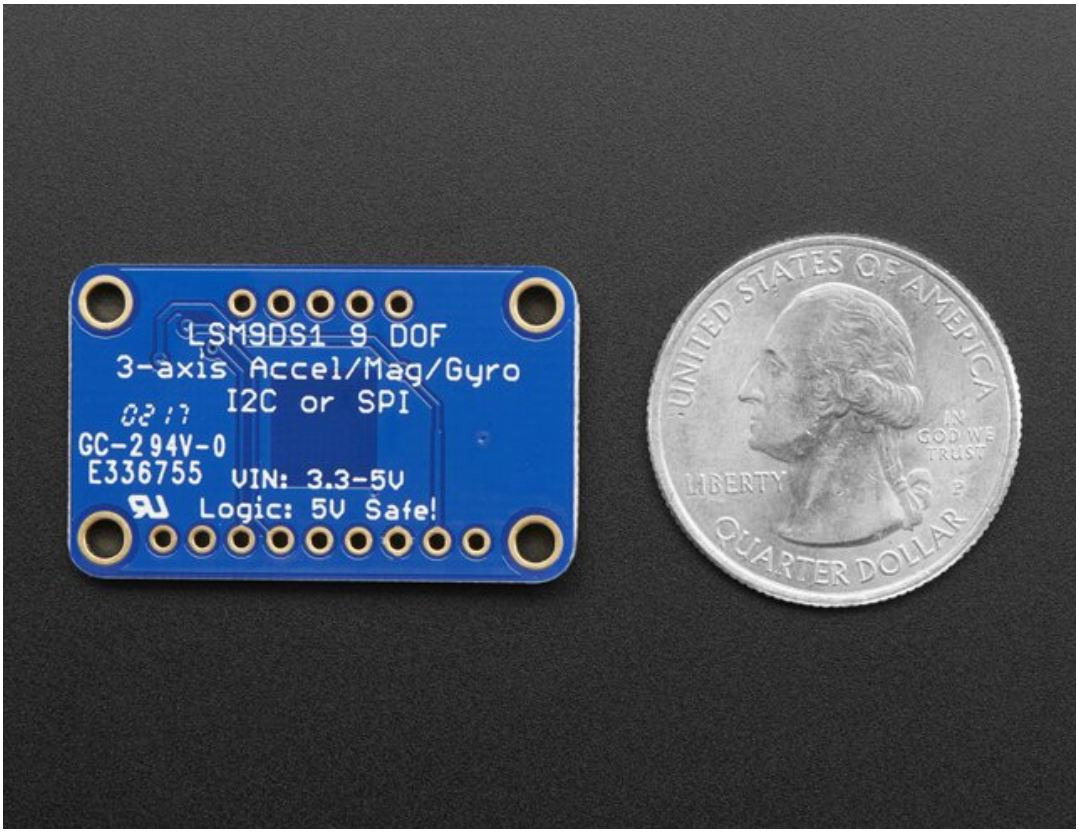

We've carried the LSM9DS0 from ST for a while, and the LSM9DS1 is their latest offering. We thought this could really make for a great breakout, at a very nice price! Design your own activity or motion tracker with all the data... We spun up a breakout board that has all the extra circuitry you'll want, for use with an Arduino (or other microcontroller)

The LSM9DS1 is not the same set of sensors as the LSM9DS0. Here are some of the differences

- LSM9DS0 accelerometer has ±2/±4/±6/±8/±16 *g* ranges. The LSM9DS1 has ±2/±4/±8/±16 *g* (no ±6 *g* range)
- LSM9DS0 magnetometer has ±2/±4/±8/±12 gauss ranges. The LSM9DS1 has ±4/±8/±12/±16 gauss ranges. So the LSM9DS0 has ±2 gauss low range where-as the LSM9DS1 has ±16 gauss high range
- LSM9DS0 and LSM9DS1 gyros *both* have the same ±245/±500/±2000 dps ranges.

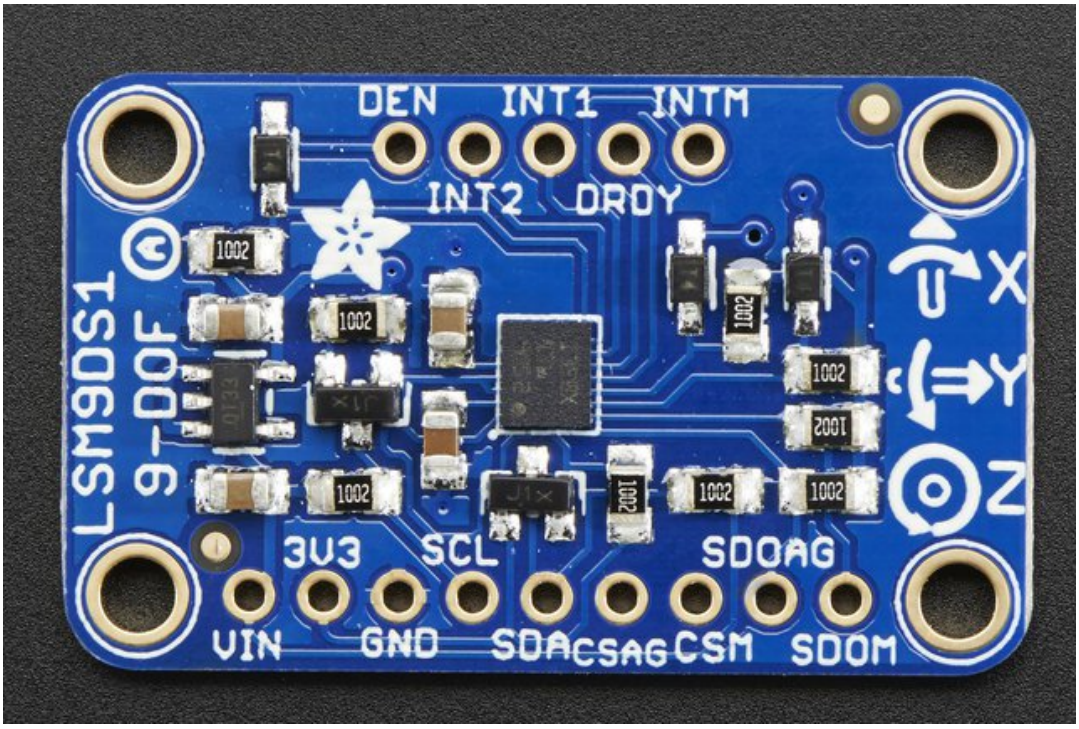
There are other differences, for example we noticed the LSM9DS1 has slightly worse accuracy. The gyro angular zero-rate (±25 for the LSM9DS0 and ±30 for the LSM9DS1 at the highest sensing range). The accelerometer offset accuracy is ±90 m*g* for the LSM9DS1 and ±60 m*g* for the LSM9DS0.

However, these offsets may not matter for most projects and the pricing of the LSM9DS1 is lower than the LSM9DS0




The breakout board version of this sensor has both I2C and SPI interfaces. Attaching it to the Arduino is simple, power Vin and GND with 3-5VDC, and wire up I2C data on SCL and SDA, and you're ready to go! More advanced users can use SPI, our library has support for both. The breakout comes fully assembled and tested, with some extra header so you can use it on a breadboard. Four mounting holes make for a secure connection, and we put the popular power+data pins on one side, and the interrupt pins on the other side for a nice & compact breakout.

# Pinouts




## Power Pins

The sensor on the breakout requires 3V power. Since many customers have 5V microcontrollers like Arduino, we tossed a 3.3V regulator on the board. Its ultra-low dropout so you can power it from 3.3V-5V just fine.

- **Vin** - this is the power pin. Since the chip uses 3 VDC, we have included a voltage regulator on board that will take 3-5VDC and safely convert it down. To power the board, give it the same power as the logic level of your microcontroller - e.g. for a 5V micro like Arduino, use 5V
- **3V3** - this is the 3.3V output from the voltage regulator, you can grab up to 100mA from this if you like
- **GND** - common ground for power and logic

## I2C Pins

- **SCL** - I2C clock pin, connect to your microcontrollers I2C clock line. This pin is level shifted so you can use 3-5V logic, and there's a 10K pullup on this pin.
- **SDA** - I2C data pin, connect to your microcontrollers I2C data line. This pin is level shifted so you can use 3-5V logic, and there's a 10K pullup on this pin.

## SPI Pins

If you're interested in using SPI to interface with the LSM9DS1, you can!

- **SCL** - this is also the SPI clock pin, it's level shifted so you can use 3-5V logic input
- **SDA** - this is also the SPI MOSI pin, it's level shifted so you can use 3-5V logic input

- **CSAG** - this is the Accelerometer+Gyro subchip Chip Select, it's level shifted so you can use 3-5V logic input
- **CSM** - this is the Magnetometer subchip Select, it's level shifted so you can use 3-5V logic input
- **SDOAG** - this is the Accelerometer+Gyro subchip MISO pin - it's 3V logic out, but can be read properly by 5V logic chips.
- **SDOM** - this is the Magnetometer subchip MISO pin - it's 3V logic out, but can be read properly by 5V logic chips.

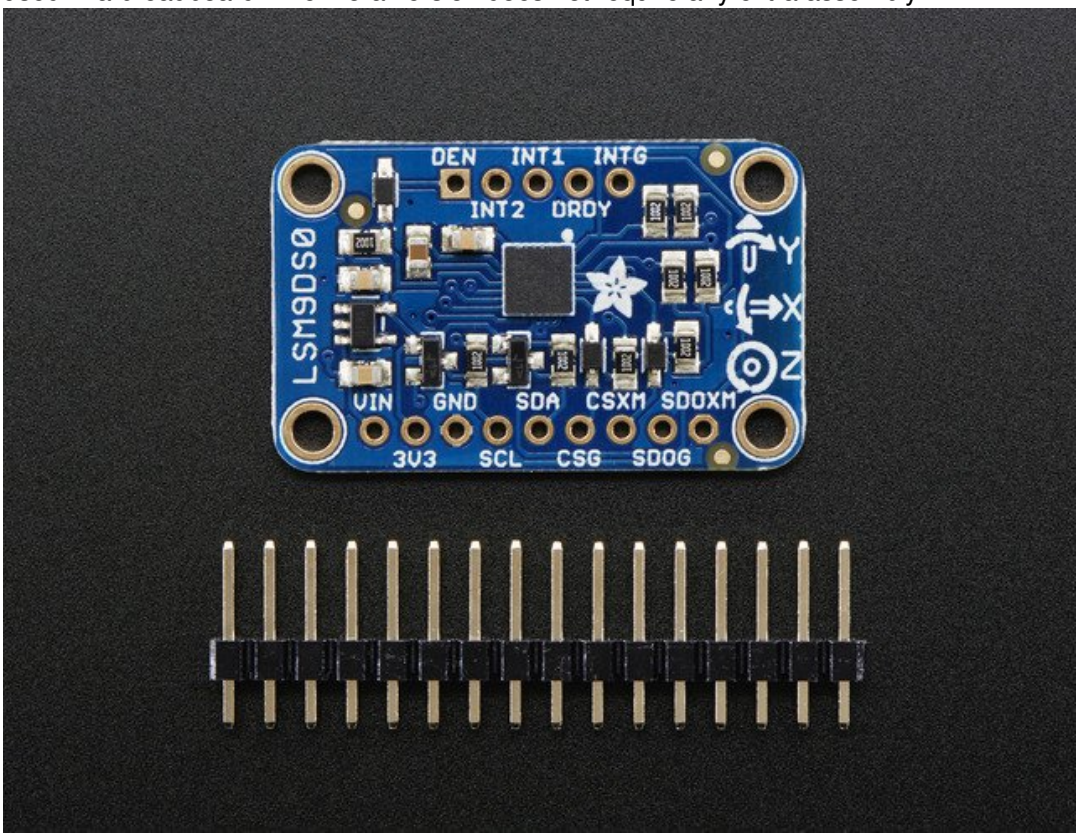
## Interrupt & Misc Pins

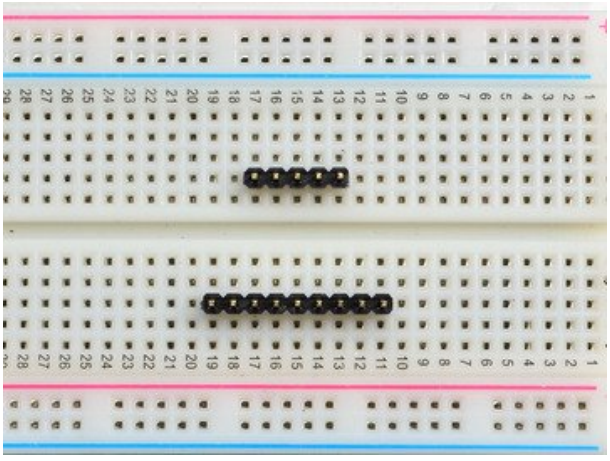
Since there's so many sensors in the LSM9DS1, there's quite a number of interrupt outputs.

- **DEN** - this is a pin that supposedly could be used to dynamically enable/disable the Gyro. There's actually no documentation on it but we break it out for you anyways.
- **INT1** & **INT2** - These are interrupts from the accelerometer/gyro subchip. We don't have specific library support for these so check the datasheet for what you can make these indicate. They are 3V-logic outputs
- **DRDY** - this is the accelerometer/gyro subchip data ready output. We don't have specific library support for these so check the datasheet for how you can set the registers to enable this pin. It is a 3V-logic output.
- **INTM** - This is the interrupt from the magnetometer subchip. We don't have specific library support for it so check the datasheet for what you can make it indicate. It is a 3V-logic output.

# Assembly

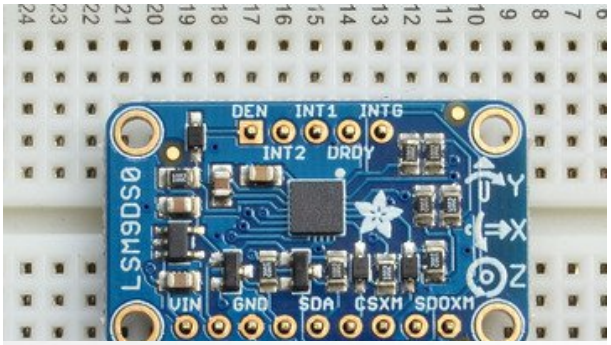
If you have the breadboard version of this sensor, you'll want to solder some header onto the sensor so it can be used in a breadboard. The Flora version does not require any extra assembly




- 

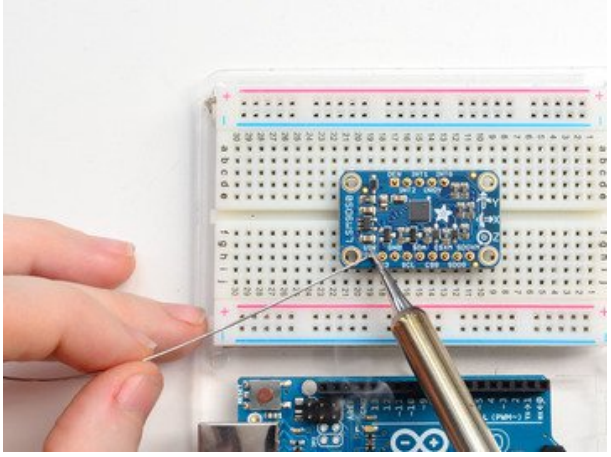
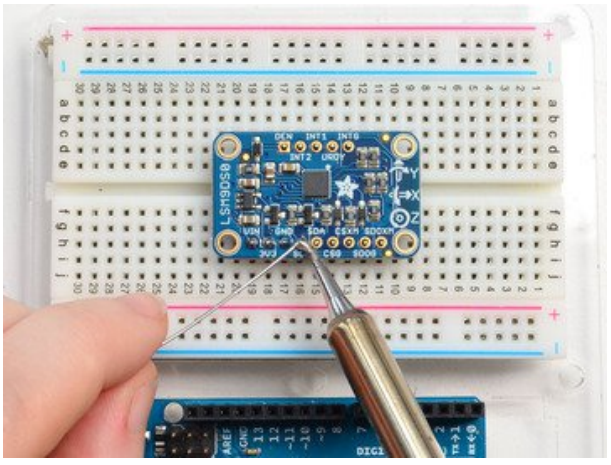
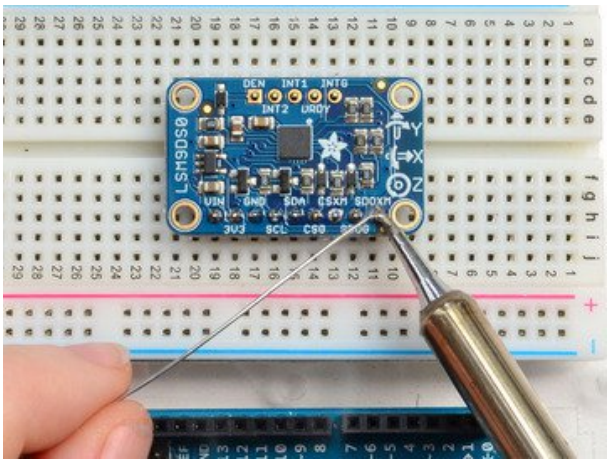
## Prepare the header strip:

Cut the strip to length if necessary. It will be easier to solder if you insert it into a breadboard - **long pins down**




## Add the breakout board:

Place the breakout board over the pins so that the short pins poke through the breakout pads

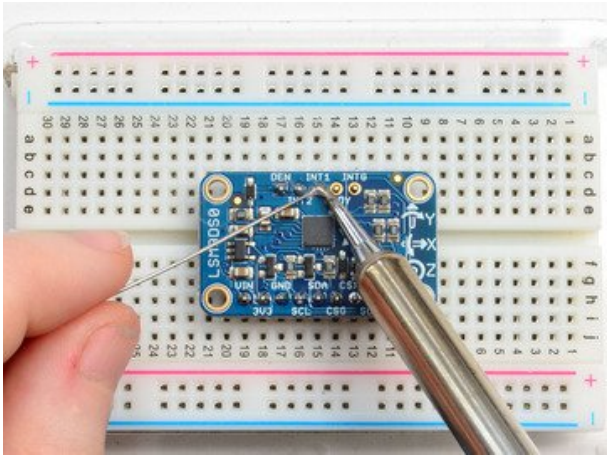
- 
- 
- 

## And Solder!

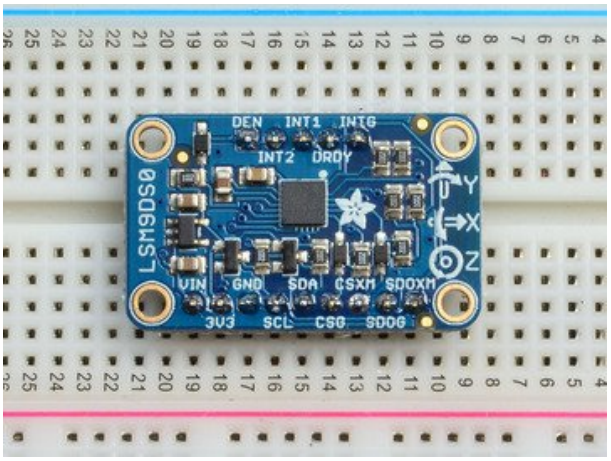
Be sure to solder all pins for reliable electrical contact.

Solder the longer power/data strip first

*(For tips on soldering, be sure to check out our[Guide to Excellent Soldering](http://adafru.it/aTk)).*

- 


If you plan to use the interrupts and/or you want the board to sit flatter in a breadboard, solder up the other strip!

- 

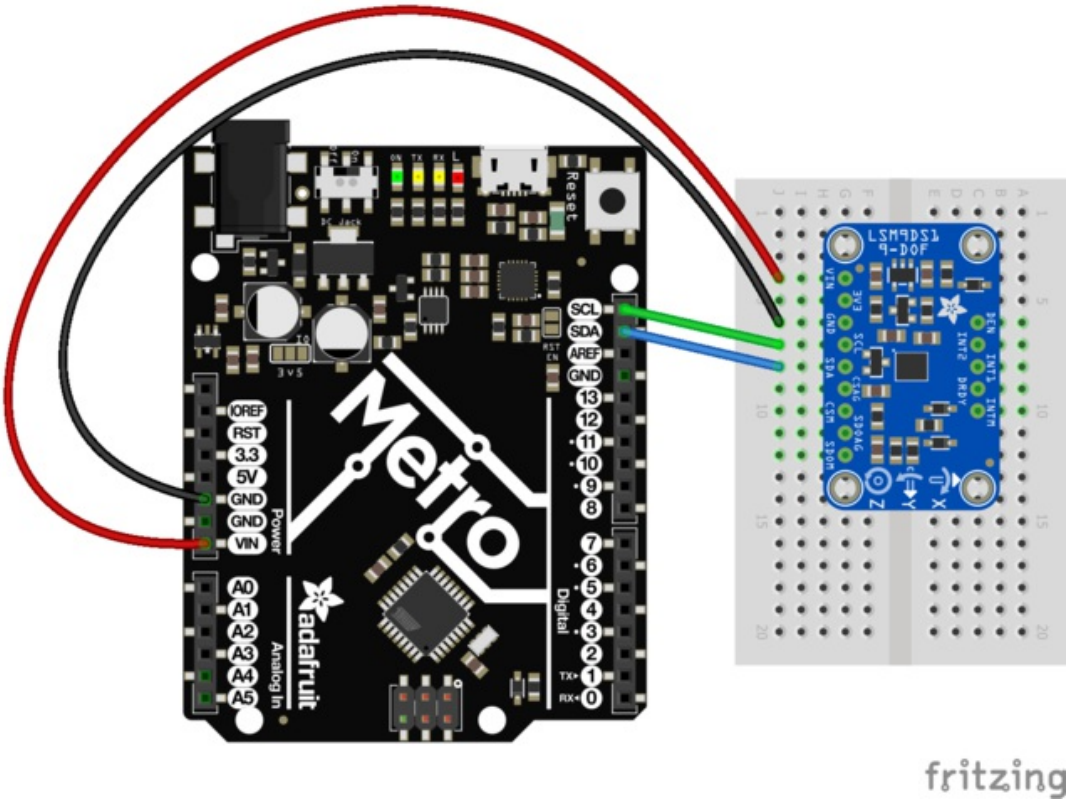

You're done! Check your solder joints visually and continue onto the next steps

# Wiring & Test

# Wiring for Arduino

You can easily wire this breakout to any microcontroller, we'll be using an Arduino. For another kind of microcontroller, just make sure it has I2C or SPI, then port the code - its pretty simple stuff!

Let's start with just I2C interfacing since it requires the fewest # of wires:




- Connect **Vin** to the power supply, 3-5V is fine. Use the same voltage that the microcontroller logic is based off of. For most Arduinos, that is 5V
- Connect **GND** to common power/data ground
- Connect the **SCL** pin to the I2C clock**SCL** pin on your Arduino. On an UNO & '328 based Arduino, this is also known as **A5**, on a Mega it is also known as **digital 21** and on a Leonardo/Micro, **digital 3**
- Connect the **SDA** pin to the I2C data**SDA** pin on your Arduino. On an UNO & '328 based Arduino, this is also known as **A4**, on a Mega it is also known as **digital 20** and on a Leonardo/Micro, **digital 2**

# Download Adafruit_LSM9DS1

To begin reading sensor data, you will need todownload the Adafruit_LSM9DS1 Library from our github repository (http://adafru.it/uaX). You can do that by visiting the github repo and manually downloading or, easier, just click this button to download the zip

Download Adafruit_LSM9DS1 Library

http://adafru.it/uaY

Rename the uncompressed folder **Adafruit_LSM9DS1** and check that the **Adafruit_LSM9DS1** folder contains **Adafruit_LSM9DS1.cpp** and **Adafruit_LSM9DS1.h**

Place the **Adafruit_LSM9DS1** library folder your **arduinosketchfolder/libraries/** folder.
You may need to create the**libraries** subfolder if its your first library. Restart the IDE.

We also have a great tutorial on Arduino library installation at:
http://learn.adafruit.com/adafruit-all-about-arduino-libraries-install-use (http://adafru.it/aYM)

# Download Adafruit_Sensor

The Adafruit_LSM9DS1 library uses the Adafruit_Sensor support backend so that readings can be normalized between sensors. You can grab Adafruit_Sensor from the github repo (http://adafru.it/aZm) or just click the button below.

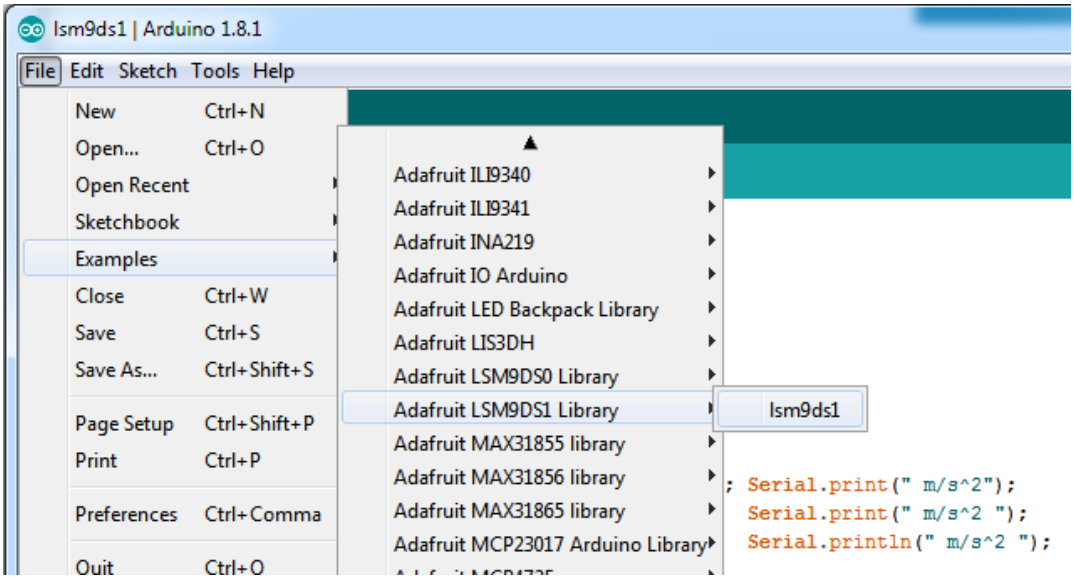
Download Adafruit_Sensor Library
http://adafru.it/cMO

Install like you did with Adafruit_LSM9DS1

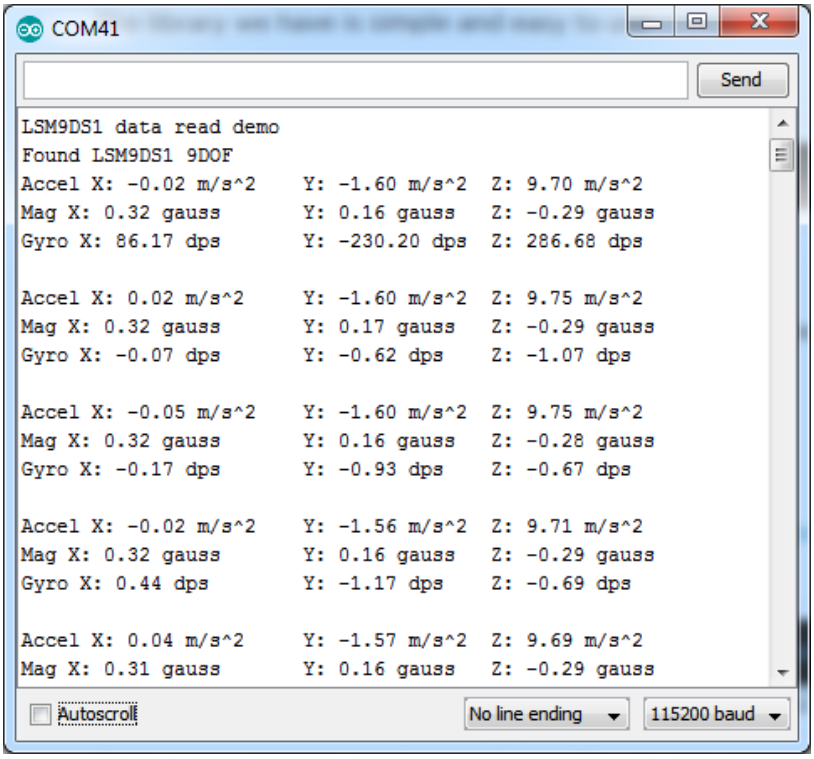
Restart the IDE!

# Load Demo Sketch

Now you can open up **File->Examples->Adafruit_LSM9DS1->lsm9ds1** and upload to your Arduino wired up to the sensor




Then open up the Serial console at 115200 baud to read the sensor output! You'll get 9 distinct data points, accelerometer x/y/z in meters/s$^2$, magetometer x/y/z in gauss and gyroscope in x/y/x degrees/second




We suggest using this **Adafruit_Sensor** interface as shown in this demo, since it will let you swap sensors without having to worry about units compatibility. Try twisting and moving the board around to see the sensors change value.

# Library Reference

The library we have is simple and easy to use

You can create the **Adafruit_LSM9DS1** object with:

```
Adafruit_LSM9DS1 lsm = Adafruit_LSM9DS1();  // i2c sensor
```

I2C does not have pins, as they are fixed in hardware.

If you're using "hardware" SPI, you will have to wire up the pins as follows:

- **SCL** -> **SPI CLK**
- **SDA** -> **SPI MOSI**
- **SDO_AG** & **SDO_M** -> **SPI MISO** (both together)

You can determine the hardware SPI pins for your Arduino here(http://adafru.it/d5h) Then pick two pins for the CS lines

```
Adafruit_LSM9DS1 lsm = Adafruit_LSM9DS1(LSM9DS1_XGCS, LSM9DS1_MCS);
```

If you don't want to use the hardware SPI, you can also try the soft SPI capability, which is bitbanged. You can basically use any pins you like!

```
Adafruit_LSM9DS1 lsm = Adafruit_LSM9DS1(LSM9DS1_SCK, LSM9DS1_MISO, LSM9DS1_MOSI, LSM9DS1_XGCS, LSM9DS1_MCS);
```

# Begin!

To initialize the sensor, call **lsm.begin()** which will check the sensor can be found. It returns true/false depending on these checks. We suggest you wrap begin() in a statement that will check if the sensor was located:

```
if(!lsm.begin())
  {
    /* There was a problem detecting the LSM9DS1 ... check your connections */
    Serial.print(F("Ooops, no LSM9DS1 detected ... Check your wiring!"));
    while(1);
  }
```

# Set Ranges

These chips have tons of registers, we basically provide interface code for the most useful stuff, such as setting the range. Each subsensor has it's own range. Higher ranges have less precision but can measure larger movements!
Set up the ranges with the **setup** functions:

```
// 1.) Set the accelerometer range
  lsm.setupAccel(lsm.LSM9DS1_ACCELRANGE_2G);
  //lsm.setupAccel(lsm.LSM9DS1_ACCELRANGE_4G);
  //lsm.setupAccel(lsm.LSM9DS1_ACCELRANGE_8G);
  //lsm.setupAccel(lsm.LSM9DS1_ACCELRANGE_16G);

  // 2.) Set the magnetometer sensitivity
  lsm.setupMag(lsm.LSM9DS1_MAGGAIN_4GAUSS);
  //lsm.setupMag(lsm.LSM9DS1_MAGGAIN_8GAUSS);
  //lsm.setupMag(lsm.LSM9DS1_MAGGAIN_12GAUSS);
  //lsm.setupMag(lsm.LSM9DS1_MAGGAIN_16GAUSS);

  // 3.) Setup the gyroscope
  lsm.setupGyro(lsm.LSM9DS1_GYROSCALE_245DPS);
  //lsm.setupGyro(lsm.LSM9DS1_GYROSCALE_500DPS);
  //lsm.setupGyro(lsm.LSM9DS1_GYROSCALE_2000DPS);
```

Choose whichever range you like, after you begin() the sensor!

# Read data

Read data using the Adafruit_Sensor API by first creating four events, one for each sub-sensor:

```
sensors_event_t accel, mag, gyro, temp;
```

Then pass these into the **getEvent** function

```
lsm.getEvent(&accel, &mag, &gyro, &temp);
```

The data is snapshotted at once, so you can read and manage the data later.

For the Accelerometer event you can read **accel.acceleration.x**, **accel.acceleration.y** or **accel.acceleration.z** which are in meters/second*second.

For the Magnetometer event you can read **mag.magnetic.x**, **mag.magnetic.y** or **mag.magnetic.z** which are in

gauss.

For the Gyro event you can read **gyro.gyro.x, gyro.gyro.y** or **gyro.gyro.z**, which are in degrees-per-second (dps)

The temperature event data is in **temp.temperature**, but we don't guarantee that the temperature data is in degrees C

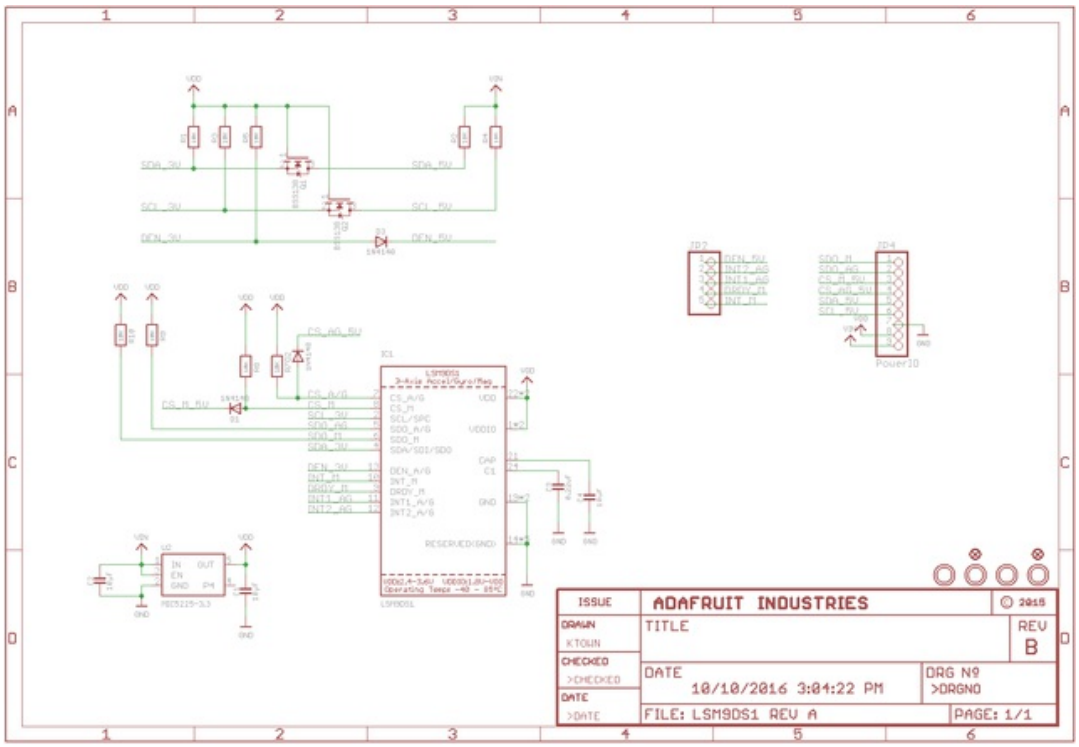
# Downloads

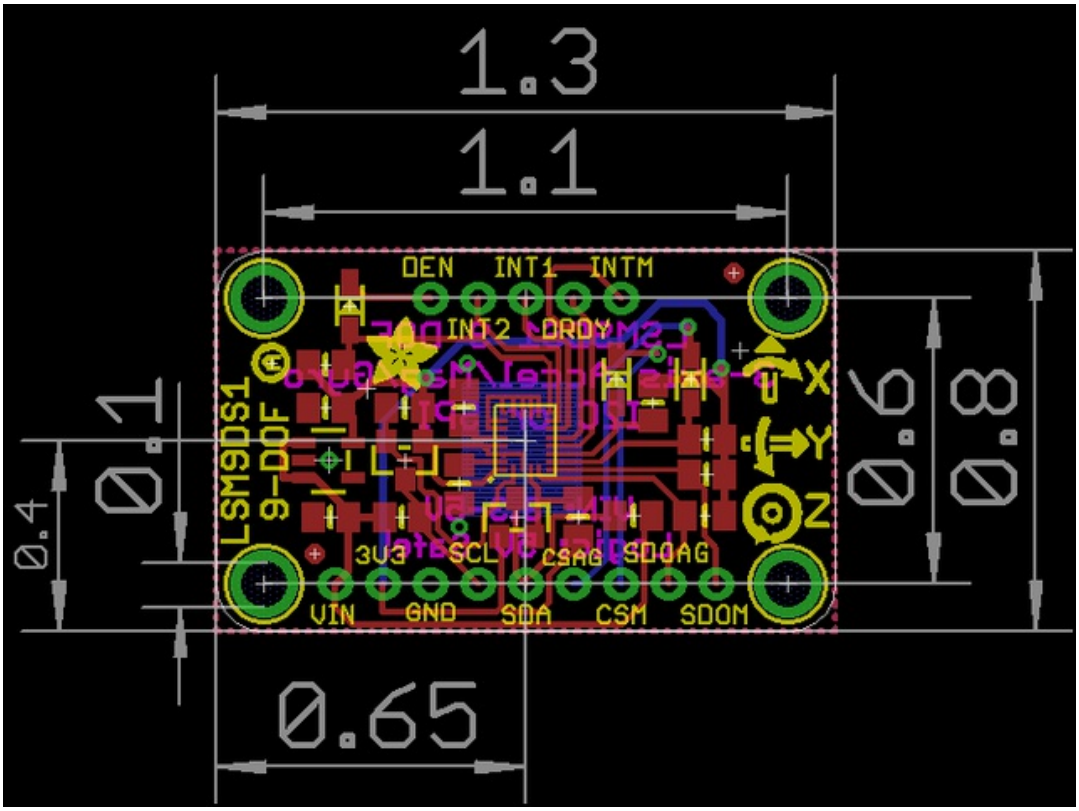
# Files

- EagleCAD PCB files on GitHub (http://adafru.it/ub2)
- Fritzing object available in Adafruit Fritzing library (http://adafru.it/aP3)

LSM9DS1.pdf Datasheet
http://adafru.it/ub3

# Schematic and Fabrication Print




1.3
1.1
DEN
INT1
INTM
INT2
DRDY
LSM9DS1
9-DOF
3V3
SCL
CSAG
SDOAG
VIN
GND
SDA
CSM
SDOM
X
Y
Z
0.1
0.4
0.6
0.8
0.65