adafruit learning system

LSM303 Accelerometer + Compass Breakout

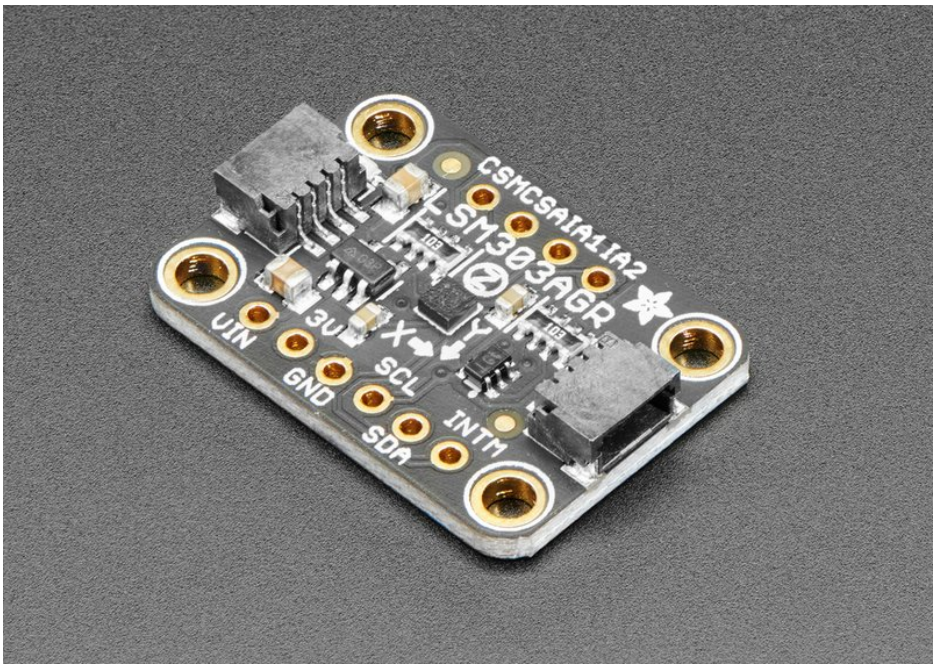
Created by Bill Earl

Last updated on 2020-01-08 10:19:58 PM UTC

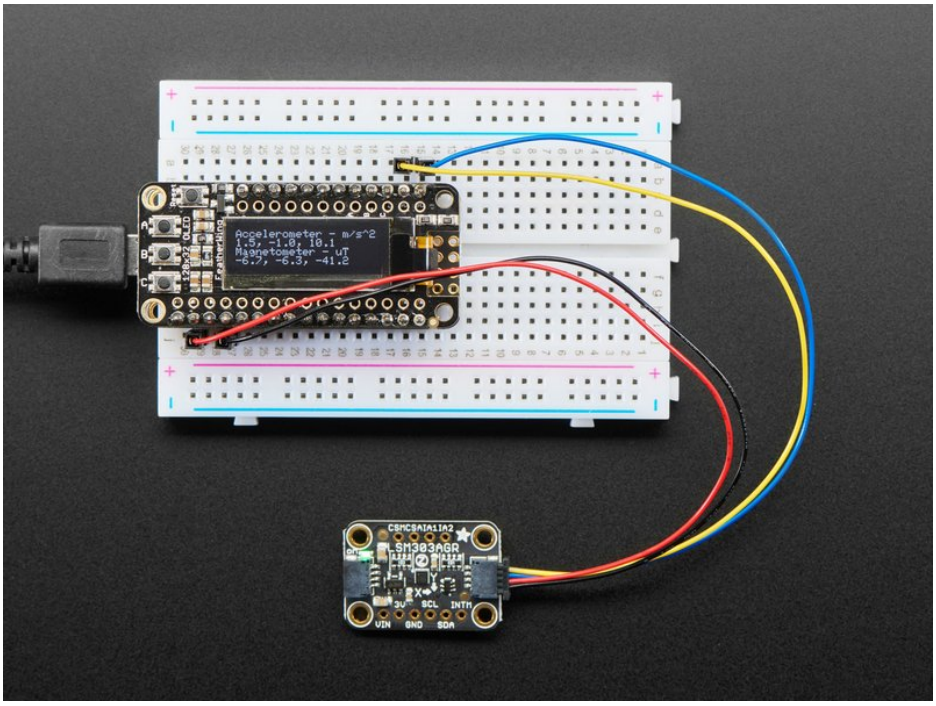
Overview

The LSM303 (http://adafru.it/4413) breakout board combines a magnetometer/compass module with a triple-axis accelerometer to make a compact navigation subsystem. The I2C interface is compatible with both 3.3v and 5v processors and the two pins can be shared by other I2C devices. Combined with a 3-axis gyro such as the L3GD20 (http://adafru.it/1032), you have all the sensors you need for a complete IMU (Inertial Measurement Unit) for use in aerial, terrestrial or marine navigation.

In this tutorial we will show you how to connect the LSM303 to an Arduino, CircuitPython board, or Blinka supported device, and how to use it to measure orientation relative to the earth's magnetic field, and acceleration in three axis.



One great feature of the newer LSM303AGR is that it includes a **STEMMA QT (https://adafru.it/Ft4)** connectors that allow you to use it *without soldering.* Just plug in a STEMMA QT to male header cable (https://adafru.it/FA-) to connect it to a breadboard or development board with female headers and you're off to the races! You can even chain it with other STEMMA QT breakouts using a STEMMA QT cable (https://adafru.it/FNS)



How it Works:

MEMS - Micro Electro-Mechanical Systems

The sensor consists of micro-machined structures on a silicon wafer. There are structures designed to measure acceleration and magnetic fields in the X, Y and Z axis

Acceleration Measurement

These structures are suspended by polysilicon springs which allow them to deflect when subject to acceleration in the X, Y and/or Z axis. Deflection causes a change in capacitance between fixed plates and plates attached to the suspended structure. This change in capacitance on each axis is converted to an output voltage proportional to the acceleration on that axis.

Magnetic Field Measurement

These structures are similar to the accelerometer structures, but are etched with microscopic coils. An excitation current is passed through the coils, and the Lorentz Force (https://adafru.it/c25) due to the magnetic field causes the structure to deflect. Once again the deflection is converted to an output voltage proportional to the strength of the magnetic field in that axis.

Which LSM303 Do I Have?

There are now two Adafruit made LSM303 breakouts! Schnikes! They are nearly identical in function but different enough that they require different drivers for the magnetometer. Why only the magnetometer? Well, I'm gonna let you in on a little secret: the LSM303 is two sensors stuck in the same box!

Because both packages have the same accelerometer sensor with the same register layout, we can use the same accelerometer driver for both. The magnetometers *despite both having the same I2C address* (0x1E), have completely different register layouts which means they each need their own driver library.

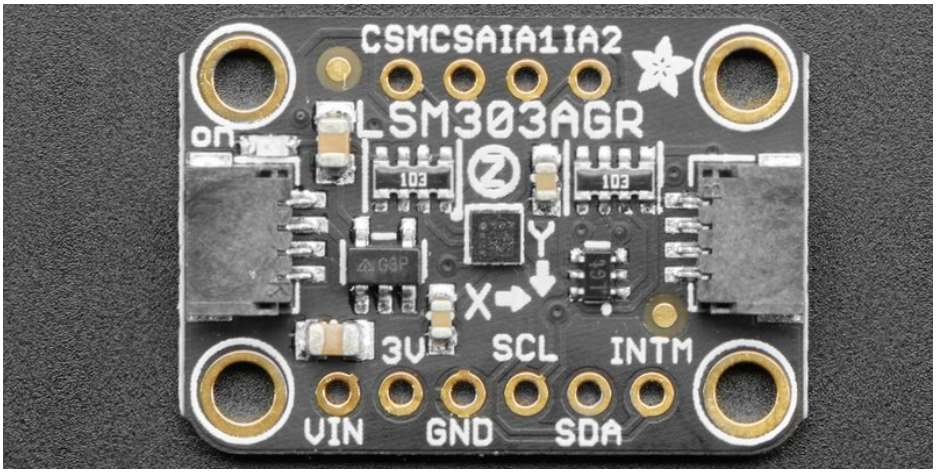
So, you're going to need to know which version you have. Here is a highly scientific way of telling the two ***Adafruit*** breakouts apart:

The LSM303AGR IS BLACK AND SAYS "LSM303AGR" unlike the LSM303DLHC WHICH IS BLUE AND SAYS "LSM303DLHC"

Complex, I know, but I believe in you. There are more subtle differences outlined below

LSM303AGR

The new kid on the adafruit block is the LSM303AGR. It's an update to the LSM303 series that is *much* smaller and has a different magnetometer chip inside. Our breakout looks like this:



In addition to being a physically smaller and easier to lose chip, there are a few other difference as well. The chip itself exposes an interrupt pin for the magnetometer **(INTM)** (perhaps a feature of the new magnetometer sensor inside?) as well as chip select pins for both the accelerometer and magnetometer (**CSA** and **CSM** respectively).

Another exciting difference between the two is the addition of two **STEMMA QT (https://adafru.it/Ft4)** connectors, one on each side of the board. This allows you to use the sensor without needing to solder as well as being able to chain it with other STEMMA QT sensors (https://adafru.it/GfR).

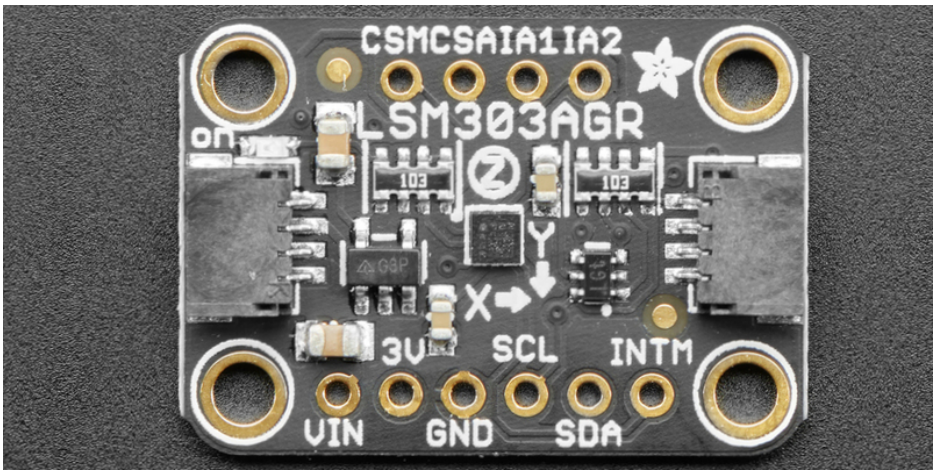
Please note that while the LSM303AGR breakout has CS pins, it only supports 3-wire SPI which we were not able to get working. The adventurous of spirit may consult the datasheet and give it a try! PRs are welcome ;)

LSM303/LSM303DLHC

The older, or perhaps *more seasoned* AdafruitLSM303 breakout uses the LSM303**DLHC** and looks like this:

Compared to the LSM303AGR, you can see that the sensor itself (the black rectangle in the center) is physically larger. Additionally the pins are in a different order and the LSM303DLH exposes a **DRDY** (data ready) pin.

Pinouts



Power Pins

- **Vin** - this is the power pin. Since the sensor chip uses 3.3 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 microcontroller like Arduino, use 5V
- **3Vo** - 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 Logic Pins

- **SCL** - I2C clock pin, connect to your microcontroller's 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 microcontroller's I2C data line. This pin is level shifted so you can use 3-5V logic, and there's a **10K pullup** on this pin.
- **STEMMA QT (https://adafru.it/Ft4) -** These connectors allow you to connectors to dev boards with **STEMMA QT** connectors or to other things with various associated accessories (https://adafru.it/Ft6)

Other Common Pins

- **IA1** - Tthe accelerometer's first interrupt pin. 3V logic only

- **IA2 -** The accelerometer's second interrupt pin. 3V logic only

LSM303AGR Only

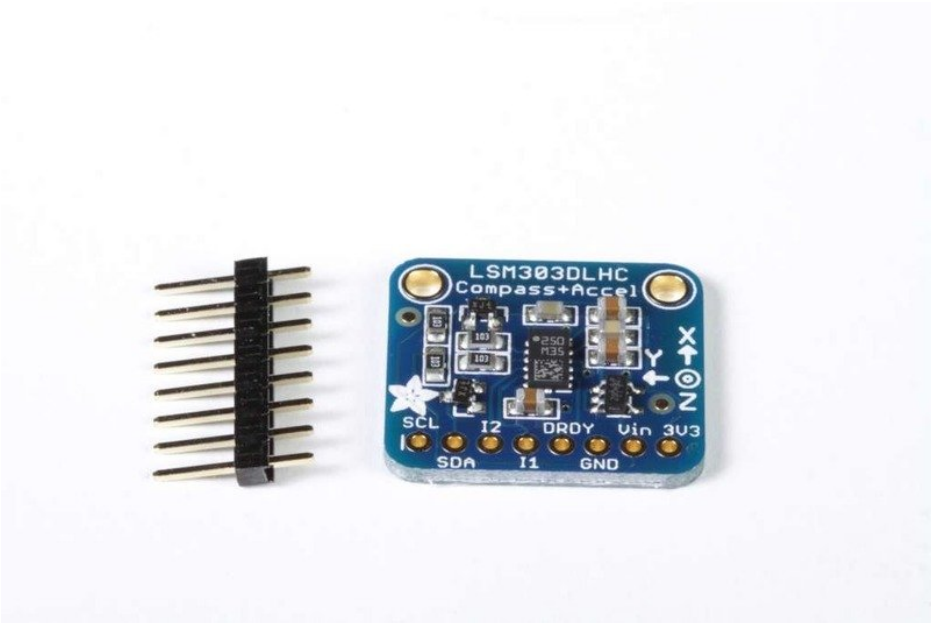
- **INTM** - The interrupt pin for the magnetometer. 3V logic only
- **CSA -** The accelerometer's CS pin. 3V logic only
- **CSM -** The magnetometer's CS pin. 3V logic only

Please note that while the LSM303AGR breakout has CS pins, it only supports 3-wire SPI which we were not able to get working. The adventurous of spirit may consult the datasheet and give it a try! PRs are welcome ;)

LSM303DLH Only

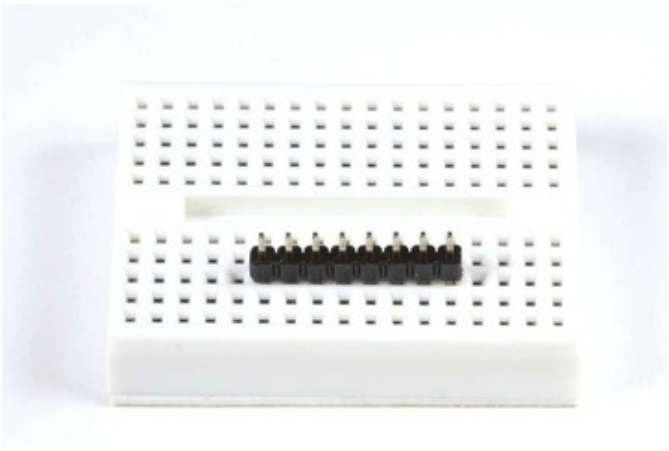
- **DRDY** - This is the data ready pin. 3V logic only

Assembly



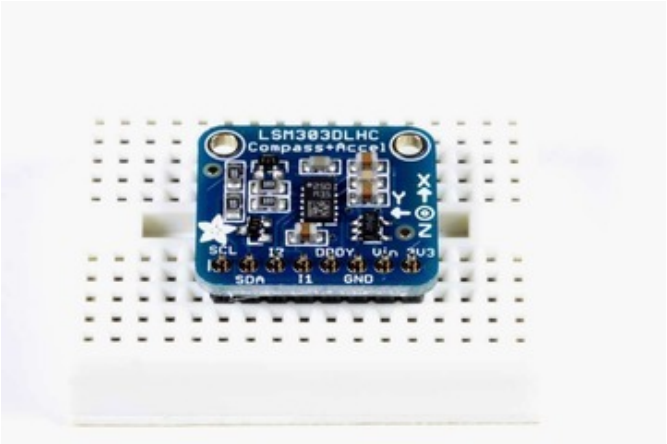
Board Assembly:

All surface mount components are pre-soldered to the board. You can solder connections directly to the board, or you can install the header strip (provided) to simplify use in a breadboard.



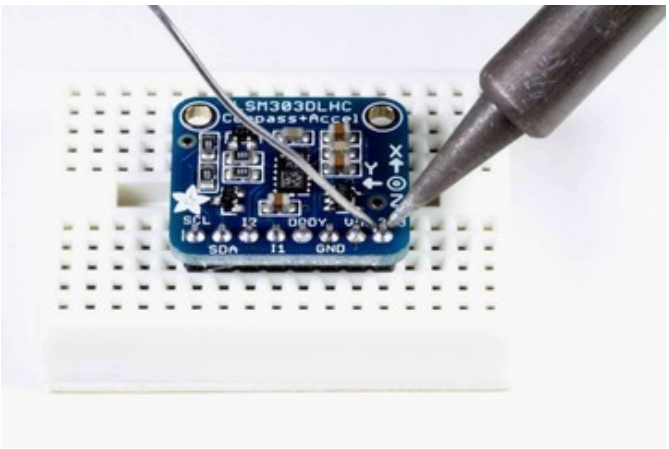
Position the header

Cut the header to length if necessary and insert - long pins down - into a breadboard.



Position the board

Place the board on top of the header pins. Prop the back-side up of necessary to level the board before soldering.

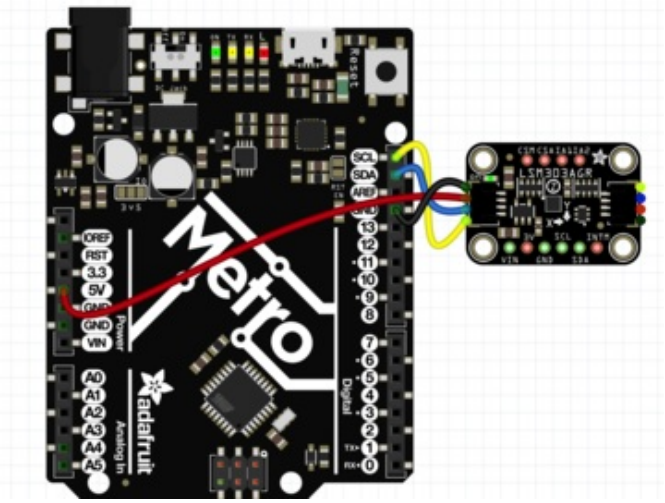


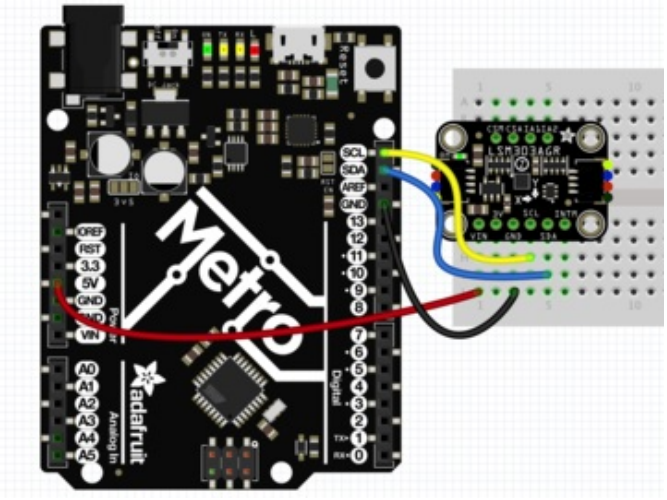
And Solder!

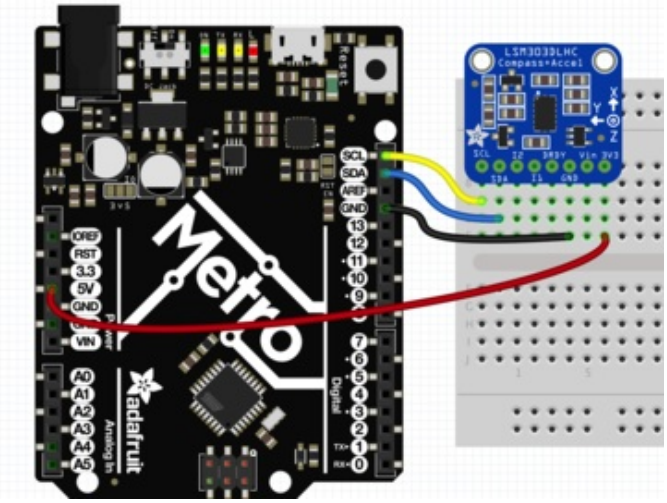
Solder each pin to assure a good electrical connection.

If you are new to soldering, check out our Guide to Excellent Soldering (https://adafru.it/aTk).

Arduino Wiring





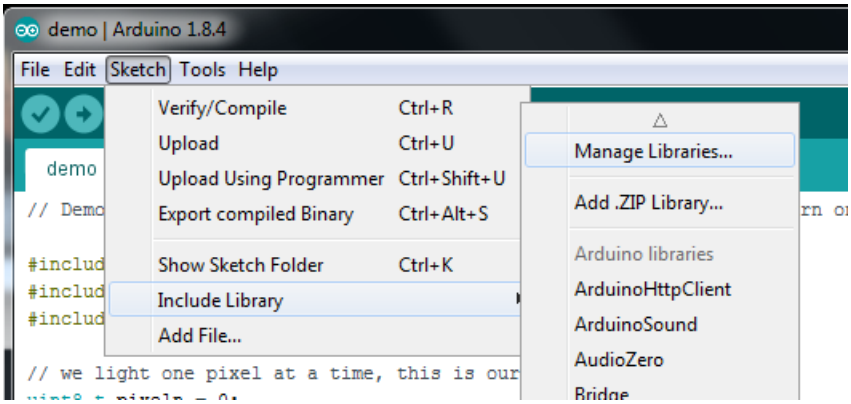


- Connect **board VCC (red wire)** to **Arduino 5V** if you are running a **5V** board Arduino (Uno, etc.). If your board is **3V,** connect to that instead.
- Connect **board GND (black wire)** to **Arduino GND**
- Connect **board SCL (yellow wire)** to **Arduino SCL**
- Connect **board SDA (blue wire)** to **Arduino SDA**

Install the Libraries

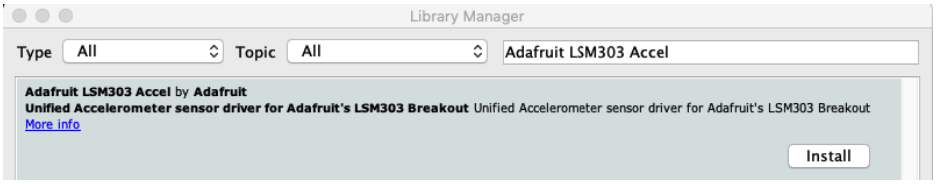
To get started with the LSM303, you'll need to install the accelerometer library and the magnetometer library for your board. Additionally you will need the Adafruit_Sensor (https://adafru.it/aZm) library that allows it to return data in a consistent way with other similar sensors, as well as the Adafruit_BusIO (https://adafru.it/GxD) library. All of the libraries can be installed using the Library Manager in the Arduino IDE:

Make sure you know which version of the sensor (https://adafru.it/GyB) you have before continuing



Install the Accelerometer Library

Click the **Manage Libraries ...** menu item, search for **Adafruit LSM303 Accel**, and select the **Adafruit_LSM303_Accel** library:

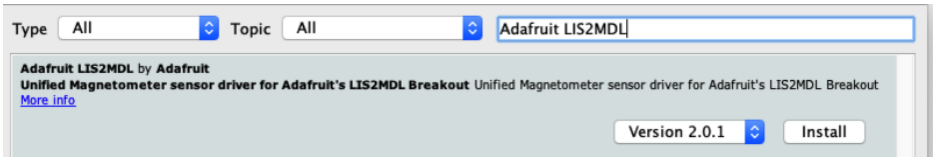


Install the Magnetometer Library

Next you'll need to install the library for the magnetometer in the LSM303. Make sure to **download the correct driver for your breakout board**.

LSM303AGR

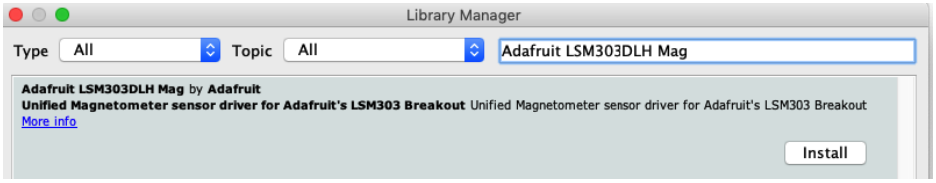
Search the library manager for the **Adafruit_LIS2MDL** library:



Why Adafruit_LIS2MDL? That's the model number of the magnetometer included with the LSM303AGR

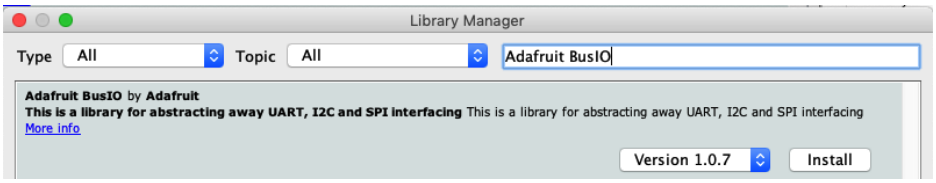
LSM303/LSM303DLHC

Search the library manager for the **Adafruit_LSM303DLH_Mag** library:

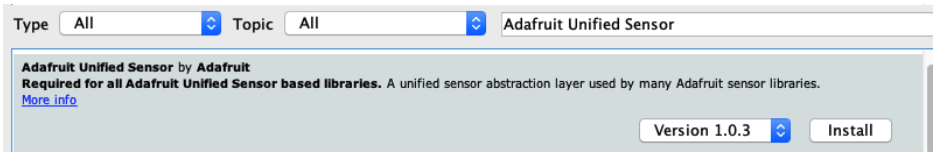


Install the Helper Libraries

Follow the same process for the **Adafruit BusIO** library:



Finally do the same for the **Adafruit Unified Sensor** library:



Accelerometer Demo

This first demo will show you how to get readings of what an accelerometer does best: measure acceleration!

Open up **File -> Examples -> Adafruit LSM303 Accel-> accelsensor** and upload to your Arduino wired up to the sensor.

Upload the sketch to your board and open up the Serial Monitor (**Tools -> Serial Monitor**) at **115200 baud**. You should see the the values for the current configuration settings printed on startup, followed by acceleration readings for the X, Y, and Z axes similar to this:

```
Accelerometer Test

------------------------------------
Sensor:       LSM303
Driver Ver:   1
Unique ID:    54321
Max Value:    0.00 m/s^2
Min Value:    0.00 m/s^2
Resolution:   0.00 m/s^2
------------------------------------

Range set to: +- 4G
Mode set to: Normal
X: 0.31  Y: 0.00  Z: 9.89  m/s^2
X: 0.38  Y: -0.08  Z: 9.97  m/s^2
X: 0.38  Y: -0.08  Z: 9.82  m/s^2
```

The Adafruit_LSM303_Accel_Unified sensor class in the Adafruit_LSM303_Accel library reports X, Y and Z axis accelerometer readings directly in meters per second squared (https://adafru.it/c26). The accelsensor example code (https://adafru.it/GdR) in the library reads from the sensor and prints the acceleration readings to the Serial Monitor.

At rest, the sensor should report no acceleration except that due to gravity (about 9.8 meters/second squared). By

calculating the angle of the gravity vector with respect to the X, Y and Z axis, the device can be used as an inclinometer (https://adafru.it/c28).

Basic Magnetometer Readings

This first demo will show you how to get readings of what an accelerometer does best: measure acceleration!

LSM303AGR

Open up **File -> Examples -> Adafruit LIS2MDL-> magsensor** and upload to your Arduino wired up to the sensor.

LSM303/LSM303DLH

Open up **File -> Examples -> Adafruit LSM303DLH Mag-> magsensor** and upload to your Arduino wired up to the sensor.

Upload the sketch to your board and open up the Serial Monitor (**Tools -> Serial Monitor**) at **115200 baud**. You should see the the values for the current configuration settings printed on startup, followed by magnetic field readings for the X, Y, and Z axes similar to this:

```
Magnetometer Test

------------------------------------
Sensor:       LIS2MDL
Driver Ver:   1
Unique ID:    12345
Max Value:    0.00 uT
Min Value:    0.00 uT
Resolution:   0.00 uT
------------------------------------

X: 17.40  Y: 65.55  Z: -29.40  uT
X: 16.35  Y: 65.25  Z: -30.00  uT
X: 16.65  Y: 65.85  Z: -29.40  uT
```

The sensor class in the magnetometer library reports X, Y and Z axis magnetometer readings directly inmicro-Teslas (https://adafru.it/c29). The **magsensor** example code reads from the sensor and prints the micro-Tesla readings to the Serial Monitor.

In the absence of any strong local magnetic fields, the sensor readings should reflect the magnetic field of the earth (between 20 and 60 micro-Teslas). When the sensor is held level, by calculating the angle of the magnetic filed with respect to the X and Y axis, the device can be used as a compass.

Computing a Compass Heading

To convert the microTesla readings into a 0-360 degree compass heading, we can use the atan2() function (https://adafru.it/c2b) to compute the angle of the vector defined by the Y and X axis readings. The result will be in radians, so we multiply by 180 degrees and divide by Pi to convert that to degrees.

LSM303AGR

Open up **File -> Examples -> Adafruit LIS2MDL-> compass** and upload to your Arduino wired up to the sensor.

LSM303/LSM303DLH

Open up **File -> Examples -> Adafruit LSM303DLH Mag-> compass** and upload to your Arduino wired up to the sensor.

Upload the sketch to your board and open up the Serial Monitor (**Tools -> Serial Monitor**) at **115200 baud**. You will see heading calculations printed out to the serial monitor. If you rotate the sensor as it runs you can see the heading change:

```
Magnetometer Test

Compass Heading: 91.14
Compass Heading: 87.49
Compass Heading: 81.48
```

Accelerometer Demo Code

```
#include <Adafruit_LSM303_Accel.h>
#include <Adafruit_Sensor.h>
#include <Wire.h>

/* Assign a unique ID to this sensor at the same time */
Adafruit_LSM303_Accel_Unified accel = Adafruit_LSM303_Accel_Unified(54321);

void displaySensorDetails(void) {
  sensor_t sensor;
  accel.getSensor(&sensor);
  Serial.println("------------------------------------");
  Serial.print("Sensor:       ");
  Serial.println(sensor.name);
  Serial.print("Driver Ver:   ");
  Serial.println(sensor.version);
  Serial.print("Unique ID:    ");
  Serial.println(sensor.sensor_id);
  Serial.print("Max Value:    ");
  Serial.print(sensor.max_value);
  Serial.println(" m/s^2");
  Serial.print("Min Value:    ");
  Serial.print(sensor.min_value);
  Serial.println(" m/s^2");
  Serial.print("Resolution:   ");
  Serial.print(sensor.resolution);
  Serial.println(" m/s^2");
  Serial.println("------------------------------------");
  Serial.println("");
  delay(500);
}

void setup(void) {
#ifndef ESP8266
  while (!Serial)
    ; // will pause Zero, Leonardo, etc until serial console opens
#endif
  Serial.begin(9600);
  Serial.println("Accelerometer Test");
  Serial.println("");

  /* Initialise the sensor */
  if (!accel.begin()) {
    /* There was a problem detecting the ADXL345 ... check your connections */
    Serial.println("Ooops, no LSM303 detected ... Check your wiring!");
    while (1)
      ;
  }
```

```
  /* Display some basic information on this sensor */
  displaySensorDetails();

  accel.setRange(LSM303_RANGE_4G);
  Serial.print("Range set to: ");
  lsm303_accel_range_t new_range = accel.getRange();
  switch (new_range) {
  case LSM303_RANGE_2G:
    Serial.println("+- 2G");
    break;
  case LSM303_RANGE_4G:
    Serial.println("+- 4G");
    break;
  case LSM303_RANGE_8G:
    Serial.println("+- 8G");
    break;
  case LSM303_RANGE_16G:
    Serial.println("+- 16G");
    break;
  }

  accel.setMode(LSM303_MODE_NORMAL);
  Serial.print("Mode set to: ");
  lsm303_accel_mode_t new_mode = accel.getMode();
  switch (new_mode) {
  case LSM303_MODE_NORMAL:
    Serial.println("Normal");
    break;
  case LSM303_MODE_LOW_POWER:
    Serial.println("Low Power");
    break;
  case LSM303_MODE_HIGH_RESOLUTION:
    Serial.println("High Resolution");
    break;
  }
}

void loop(void) {
  /* Get a new sensor event */
  sensors_event_t event;
  accel.getEvent(&event);

  /* Display the results (acceleration is measured in m/s^2) */
  Serial.print("X: ");
  Serial.print(event.acceleration.x);
  Serial.print("  ");
  Serial.print("Y: ");
  Serial.print(event.acceleration.y);
  Serial.print("  ");
  Serial.print("Z: ");
  Serial.print(event.acceleration.z);
  Serial.print("  ");
  Serial.println("m/s^2");

  /* Delay before the next sample */
  delay(500);
}
```

LSM303AGR Magnetometer and Compass Code

```
#include <Adafruit_LIS2MDL.h>
#include <Adafruit_Sensor.h>
#include <Wire.h>

/* Assign a unique ID to this sensor at the same time */
Adafruit_LIS2MDL mag = Adafruit_LIS2MDL(12345);

void displaySensorDetails(void) {
  sensor_t sensor;
  mag.getSensor(&sensor);
  Serial.println("------------------------------------");
  Serial.print("Sensor:       ");
  Serial.println(sensor.name);
  Serial.print("Driver Ver:   ");
  Serial.println(sensor.version);
  Serial.print("Unique ID:    ");
  Serial.println(sensor.sensor_id);
  Serial.print("Max Value:    ");
  Serial.print(sensor.max_value);
  Serial.println(" uT");
  Serial.print("Min Value:    ");
  Serial.print(sensor.min_value);
  Serial.println(" uT");
  Serial.print("Resolution:   ");
  Serial.print(sensor.resolution);
  Serial.println(" uT");
  Serial.println("------------------------------------");
  Serial.println("");
  delay(500);
}

void setup(void) {
#ifndef ESP8266
  while (!Serial)
    ; // will pause Zero, Leonardo, etc until serial console opens
#endif
  Serial.begin(115200);
  Serial.println("Magnetometer Test");
  Serial.println("");

  /* Enable auto-gain */
  mag.enableAutoRange(true);

  /* Initialise the sensor */
  if (!mag.begin()) {
    /* There was a problem detecting the LIS2MDL ... check your connections */
    Serial.println("Ooops, no LIS2MDL detected ... Check your wiring!");
    while (1)
      ;
  }

  /* Display some basic information on this sensor */
  displaySensorDetails();
}

void loop(void) {
```

```
  /* Get a new sensor event */
  sensors_event_t event;
  mag.getEvent(&event);

  /* Display the results (magnetic vector values are in micro-Tesla (uT)) */
  Serial.print("X: ");
  Serial.print(event.magnetic.x);
  Serial.print("  ");
  Serial.print("Y: ");
  Serial.print(event.magnetic.y);
  Serial.print("  ");
  Serial.print("Z: ");
  Serial.print(event.magnetic.z);
  Serial.print("  ");
  Serial.println("uT");

  /* Note: You can also get the raw (non unified values) for */
  /* the last data sample as follows. The .getEvent call populates */
  /* the raw values used below. */
  // Serial.print("X Raw: "); Serial.print(mag.raw.x); Serial.print("  ");
  // Serial.print("Y Raw: "); Serial.print(mag.raw.y); Serial.print("  ");
  // Serial.print("Z Raw: "); Serial.print(mag.raw.z); Serial.println("");

  /* Delay before the next sample */
  delay(500);
}
```

```
#include <Wire.h>
#include <Adafruit_Sensor.h>
#include <Adafruit_LIS2MDL.h>


Adafruit_LIS2MDL mag = Adafruit_LIS2MDL(12345);

void setup(void)
{
  Serial.begin(115200);
  Serial.println("Magnetometer Test"); Serial.println("");

  /* Initialise the sensor */
  if(!mag.begin())
  {
    /* There was a problem detecting the LIS2MDL ... check your connections */
    Serial.println("Ooops, no LIS2MDL detected ... Check your wiring!");
    while(1);
  }
}

void loop(void)
{
  /* Get a new sensor event */
  sensors_event_t event;
  mag.getEvent(&event);

  float Pi = 3.14159;

  // Calculate the angle of the vector y,x
  float heading = (atan2(event.magnetic.y,event.magnetic.x) * 180) / Pi;

  // Normalize to 0-360
  if (heading < 0)
  {
    heading = 360 + heading;
  }
  Serial.print("Compass Heading: ");
  Serial.println(heading);
  delay(500);
}
```

LSM303/LSM303DLH Magnetometer and Compass Code

```
#include <Adafruit_LSM303DLH_Mag.h>
#include <Adafruit_Sensor.h>
#include <Wire.h>

/* Assign a unique ID to this sensor at the same time */
Adafruit_LSM303DLH_Mag_Unified mag = Adafruit_LSM303DLH_Mag_Unified(12345);

void displaySensorDetails(void) {
  sensor_t sensor;
  mag.getSensor(&sensor);
  Serial.println("------------------------------------");
  Serial.print("Sensor:       ");
  Serial.println(sensor.name);
```

```
  Serial.print("Driver Ver:   ");
  Serial.println(sensor.version);
  Serial.print("Unique ID:    ");
  Serial.println(sensor.sensor_id);
  Serial.print("Max Value:    ");
  Serial.print(sensor.max_value);
  Serial.println(" uT");
  Serial.print("Min Value:    ");
  Serial.print(sensor.min_value);
  Serial.println(" uT");
  Serial.print("Resolution:   ");
  Serial.print(sensor.resolution);
  Serial.println(" uT");
  Serial.println("------------------------------------");
  Serial.println("");
  delay(500);
}

void setup(void) {
#ifndef ESP8266
  while (!Serial)
    ; // will pause Zero, Leonardo, etc until serial console opens
#endif
  Serial.begin(115200);
  Serial.println("Magnetometer Test");
  Serial.println("");

  /* Enable auto-gain */
  mag.enableAutoRange(true);

  /* Initialise the sensor */
  if (!mag.begin()) {
    /* There was a problem detecting the LSM303 ... check your connections */
    Serial.println("Ooops, no LSM303 detected ... Check your wiring!");
    while (1)
      ;
  }

  /* Display some basic information on this sensor */
  displaySensorDetails();
}

void loop(void) {
  /* Get a new sensor event */
  sensors_event_t event;
  mag.getEvent(&event);

  /* Display the results (magnetic vector values are in micro-Tesla (uT)) */
  Serial.print("X: ");
  Serial.print(event.magnetic.x);
  Serial.print("  ");
  Serial.print("Y: ");
  Serial.print(event.magnetic.y);
  Serial.print("  ");
  Serial.print("Z: ");
  Serial.print(event.magnetic.z);
  Serial.print("  ");
  Serial.println("uT");

  /* Delay before the next sample */
```

```
  /* Delay before the next sample */
  delay(500);
}
```

```
#include <Adafruit_LSM303DLH_Mag.h>
#include <Adafruit_Sensor.h>
#include <Wire.h>

Adafruit_LSM303DLH_Mag_Unified mag = Adafruit_LSM303DLH_Mag_Unified(12345);

void setup(void) {
  Serial.begin(115200);
  Serial.println("Magnetometer Test");
  Serial.println("");

  /* Initialise the sensor */
  if (!mag.begin()) {
    /* There was a problem detecting the LSM303 ... check your connections */
    Serial.println("Ooops, no LSM303 detected ... Check your wiring!");
    while (1)
      ;
  }
}

void loop(void) {
  /* Get a new sensor event */
  sensors_event_t event;
  mag.getEvent(&event);

  float Pi = 3.14159;

  // Calculate the angle of the vector y,x
  float heading = (atan2(event.magnetic.y, event.magnetic.x) * 180) / Pi;

  // Normalize to 0-360
  if (heading < 0) {
    heading = 360 + heading;
  }
  Serial.print("Compass Heading: ");
  Serial.println(heading);
  delay(500);
}
```

The Adafruit_LSM303DLHC Library has been archived

The Adafruit_LSM303DLHC Library for the older LSM303DLH breakout (https://adafru.it/Gya) has been archived and replaced by Adafruit_LSM303_Accel (https://adafru.it/GdQ) and Adafruit_LSM303DLH_Mag (https://adafru.it/Gyb) libraries which are used above. We recommend switching to the newer libraries as the old one will not be supported or have features added.

If you still wish to use the old library, you can click the buttons below to go to the repository or download it as a ZIP

https://adafru.it/aYU

https://adafru.it/aYU

https://adafru.it/cMP

https://adafru.it/cMP

Arduino Accel Docs

Arduino Accel Docs (https://adafru.it/Gyd)

Arduino AGR Mag Docs

Arduino AGR Mag Docs (https://adafru.it/lbm)

Arduino DLH Mag Docs

Arduino DLH Mag Docs (https://adafru.it/Gyf)

Python & CircuitPython

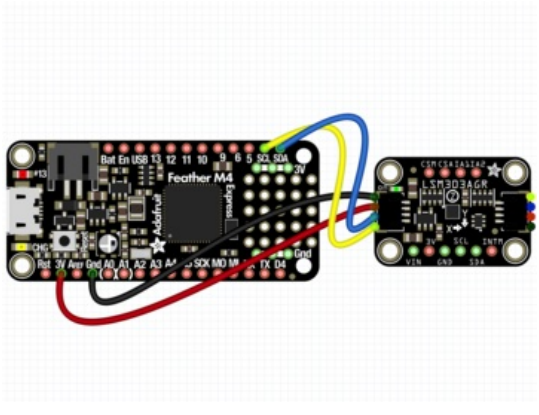
It's easy to use the LSM303 sensor with CircuitPython and the Adafruit CircuitPython LSM303 Accelerometer (https://adafru.it/Gob)r (https://adafru.it/Gob) and Adafruit CircuitPython LIS2MDL (https://adafru.it/lbr)or Adafruit CircuitPython LSM303DLH Magnetometer (https://adafru.it/Goc) libraries. These libraries allow you to easily write Python code that reads the accelerometer and magnetometer values from the sensor.

You can use this sensor with any CircuitPython microcontroller board or with a computer that has GPIO and Python thanks to Adafruit_Blinka, our CircuitPython-for-Python compatibility library (https://adafru.it/BSN).

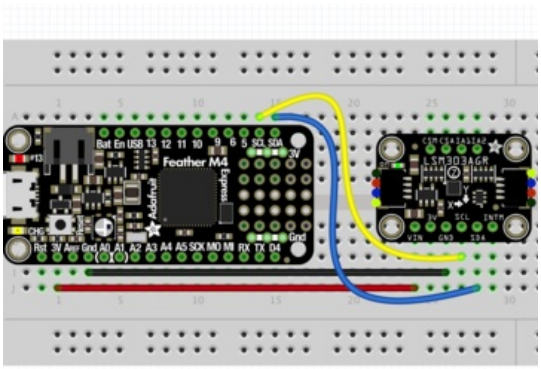
Make sure you know which version of the sensor (https://adafru.it/GyB) you have before continuing

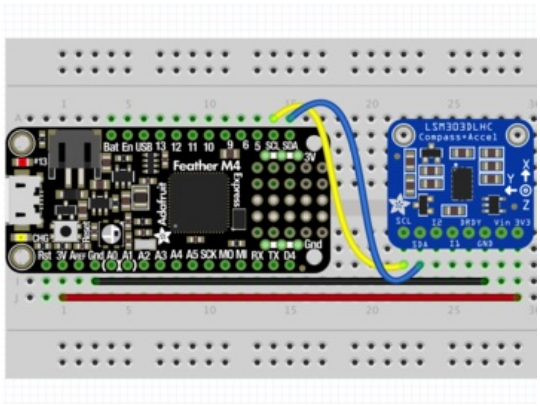
CircuitPython Microcontroller Wiring

First wire up a LSM303 to your board exactly as shown on the previous pages for Arduino using an I2C connection. Here's an example of wiring a Feather M0 to the sensor with I2C:



- **Board 3V** to **sensor VIN (red wire)**
- **Board GND** to **sensor GND (black wire)**
- **Board SCL** to **sensor SCL (yellow wire)**
- **Board SDA** to **sensor SDA (blue wire)**
-

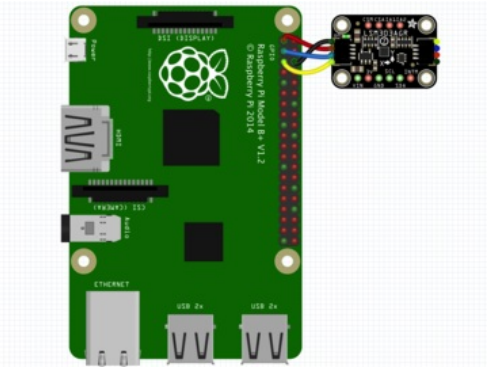


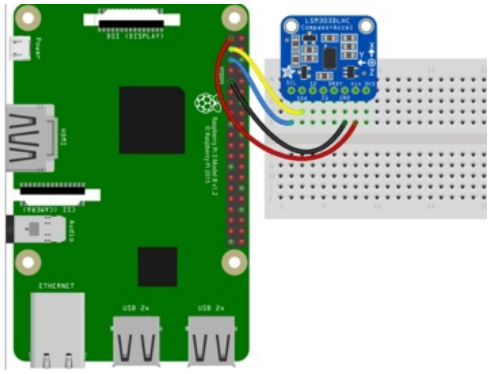


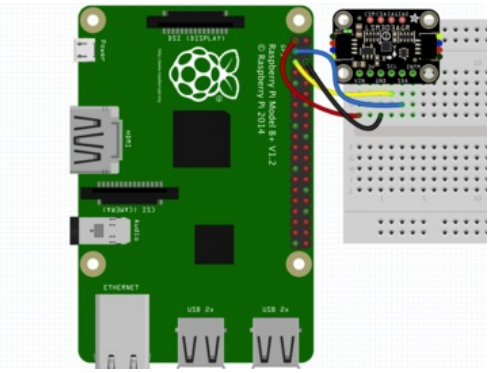
Python Computer Wiring

Since there's *dozens* of Linux computers/boards you can use we will show wiring for Raspberry Pi. For other platforms, please visit the guide for CircuitPython on Linux to see whether your platform is supported (https://adafru.it/BSN).

Here's the Raspberry Pi wired with I2C:







- **Pi 3V** to **sensor VIN (red wire)**
- **Pi GND** to **sensor GND (black wire)**
- **Pi SCL** to **sensor SCL (yellow wire)**
- **Pi SDA** to **sensor SDA (blue wire)**

CircuitPython Installation of LSM303 Library

Next you'll need to instal the Adafruit CircuitPython LSM303 (https://adafru.it/Gob) accelerometer and Adafruit CircuitPython LIS2MDL (https://adafru.it/lbr) or Adafruit CircuitPython LSM303DLH (https://adafru.it/Goc) magnetometer (depending on which LSM303 you have (https://adafru.it/GyB)) libraries on your CircuitPython board.

First make sure you are running the latest version of Adafruit CircuitPython (https://adafru.it/Amd) for your board.

Next you'll need to install the necessary libraries to use the hardware--carefully follow the steps to find and install these libraries from Adafruit's CircuitPython library bundle (https://adafru.it/ENC). Our introduction guide has a great page on how to install the library bundle (https://adafru.it/ABU) for both express and non-express boards.

Remember for non-express boards like the, you'll need to manually install the necessary libraries from the bundle:

- **adafruit_lis2mdl.mpy or adafruit_lsm303dlh_mag.mpy**

- adafruit_lsm303_accel.mpy
- adafruit_bus_device
- adafruit_register

Before continuing make sure your board's lib folder or root filesystem has the **adafruit_lis2mdl.mpy or adafruit_lsm303dlh_mag.mpy, adafruit_lsm303_accel.mpy, adafruit_register,** and **adafruit_bus_device** files and folders copied over.

Next connect to the board's serial REPL (https://adafru.it/Awz) so you are at the CircuitPython >>> prompt.

Why LIS2MDL? That's the model number of the magnetometer included in the LSM303AGR

Python Installation of LSM303 Libraries

You'll need to install the **Adafruit_Blinka** library that provides the CircuitPython support in Python. This may also require enabling I2C on your platform and verifying you are running Python 3. Since each platform is a little different, and Linux changes often, please visit the CircuitPython on Linux guide to get your computer ready (https://adafru.it/BSN)!

Once that's done, from your command line run the following command:

- `sudo pip3 install adafruit-circuitpython-lsm303-accel`

Then install the magnetometer library for your version of the LSM303:

LSM303AGR:

- `sudo pip3 install adafruit-circuitpython-lis2mdl`

LSM303DLH:

- `sudo pip3 install adafruit-circuitpython-lsm303dlh-mag`

If your default Python is version 3 you may need to run 'pip' instead. Just make sure you aren't trying to use CircuitPython on Python 2.x, it isn't supported!

CircuitPython & Python Usage

These examples use the adafruit_lis2mdl library. If you have a lsm303dlh you'll have to change the code to use the adafruit_lsm303dlh_mag library

To demonstrate the usage of the sensor we'll initialize it and read the accelerometer and compass/magnetometer from the board's Python REPL.

Run the following code to import the necessary modules and initialize the I2C connection with the sensor:

LSM303AGR

```
import board
import busio
import adafruit_lsm303_accel
import adafruit_lis2mdl

i2c = busio.I2C(board.SCL, board.SDA)
mag = adafruit_lis2mdl.LIS2MDL(i2c)
accel = adafruit_lsm303_accel.LSM303_Accel(i2c)
```

```
>>> import board
>>> import busio
>>> import adafruit_lsm303_accel
>>> import adafruit_lis2mdl
>>> i2c = busio.I2C(board.SCL, board.SDA)
>>> mag = adafruit_lis2mdl.LIS2MDL(i2c)
>>> accel = adafruit_lsm303_accel.LSM303_Accel(i2c)
```

LSM303DLH

```
import board
import busio
import adafruit_lsm303_accel
import adafruit_lsm303dlh_mag

i2c = busio.I2C(board.SCL, board.SDA)
mag = adafruit_lsm303dlh_mag.LSM303DLH_Mag(i2c)
accel = adafruit_lsm303_accel.LSM303_Accel(i2c)
```

```
>>> import board
>>> import busio
>>> import adafruit_lsm303_accel
>>> import adafruit_lsm303dlh_mag
>>> i2c = busio.I2C(board.SCL, board.SDA)
>>> mag = adafruit_lsm303dlh_mag.LSM303DLH_Mag(i2c)
>>> accel = adafruit_lsm303_accel.LSM303_Accel(i2c)
```

Now you're ready to read values from the sensor using one of these properties:

Accelerometer

- **acceleration** - A 3-tuple of X, Y, Z acceleration values in meters per second per second (m/s^2).

Magnetometer

- **magnetic** - A 3-tuple of the X, Y, Z magnetometer readings in micro-Teslas.

```
print("Acceleration (m/s^2): X=%0.3f Y=%0.3f Z=%0.3f"%accel.acceleration)
print("Magnetometer (micro-Teslas)): X=%0.3f Y=%0.3f Z=%0.3f"%mag.magnetic)
```

```
>>> print("Acceleration (m/s^2): X=%0.3f Y=%0.3f Z=%0.3f"%accel.acceleration)
Acceleration (m/s^2): X=-1.147 Y=-0.038 Z=10.173
>>> print("Magnetometer (micro-Teslas)): X=%0.3f Y=%0.3f Z=%0.3f"%mag.magnetic)
Magnetometer (micro-Teslas)): X=-22.273 Y=14.455 Z=-40.510
>>>
```

Tap Detection

The LSM303's accelerometer can also detect single or double taps!

The **set_tap** method can be used to set the number of taps (1 or 2) to detect, as well as setting a tap threshold. The lower the threshold, the more sensitive the accelerometer will be to detecting taps. You will likely have to play around with this number to find a value that suits your needs

Here we're also going to use the **range** property to change the measurement range of the accelerometer to **+/-8G**

```
accel.range = adafruit_lsm303_accel.Range.RANGE_8G
accel.set_tap(1, 30)
while True:
    if accel.tapped:
        print("Tapped!\n")
```

When you type code above into the REPL and give the sensor a nice solid tap, you should see something similar to the output below. If you don't see the **"Tapped!"** message, you may need to adjust the threshold.

```
>>> accel.range = adafruit_lsm303_accel.Range.RANGE_8G
>>> accel.set_tap(1, 30)
>>> while True:
...         if accel.tapped:
...                     print("Tapped!\n")
...
Tapped!

Tapped!

Tapped!

Tapped!
```

As you can see from the above, the tapped property will return True over a number of cycles for a single tap event. You will have to account for this in your code. Unfortunately this just seems to be a quirk of the sensor

That's all there is to using the LSM303 with CircuitPython!

Below is a complete example of reading the sensor and printing its values every second. Save this as **code.py** on your board and open the REPL to see the output.

Tap Detection Code

```
Temporarily unable to load content:
```

LSM303AGR Example Code

```
from time import sleep
import board
import busio
import adafruit_lsm303_accel
import adafruit_lsm303agr_mag

i2c = busio.I2C(board.SCL, board.SDA)
mag = adafruit_lsm303agr_mag.LSM303AGR_Mag(i2c)
accel = adafruit_lsm303_accel.LSM303_Accel(i2c)

while True:
    print("Acceleration (m/s^2): X=%0.3f Y=%0.3f Z=%0.3f"%accel.acceleration)
    print("Magnetometer (micro-Teslas)): X=%0.3f Y=%0.3f Z=%0.3f"%mag.magnetic)
    print("")
    sleep(0.5)
```

LSM303DLH Example Code

```
Temporarily unable to load content:
```

The Adafruit_CircuitPython_LSM303 Library has been archived

The Adafruit_CircuitPython_LSM303 Library for the older LSM303DLH breakout (https://adafru.it/Gya) has been archived and replaced by Adafruit_CircuitPython_LSM303_Accel (https://adafru.it/Gob) and Adafruit_CircuitPython_LSM303DLH_Mag (https://adafru.it/Goc) libraries which are used above. We recommend switching to the newer libraries as the old one will not be supported or have features added.

If you still wish to use the old library, you can click the buttons below to go to the repository or download it as a ZIP

https://adafru.it/Axe

https://adafru.it/Axe

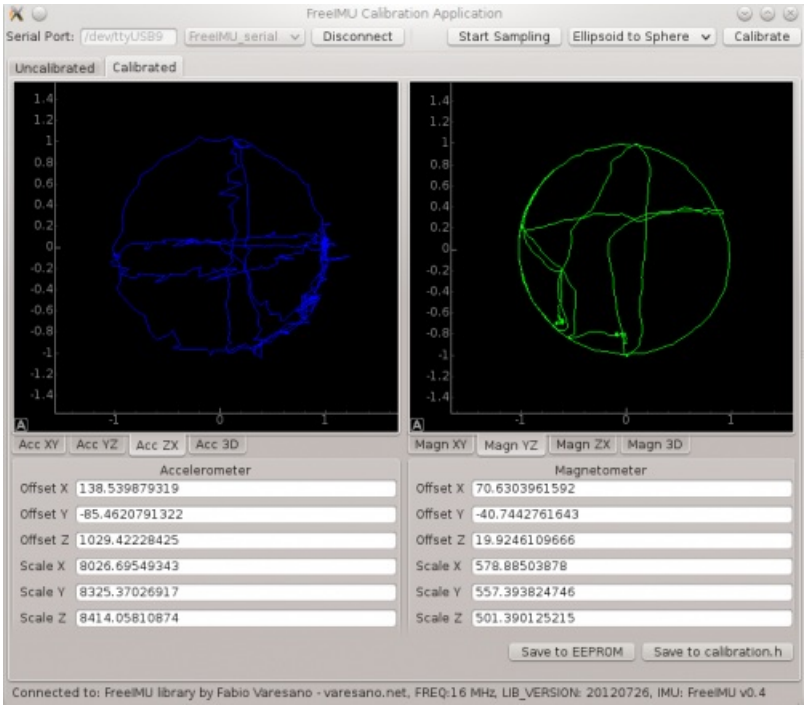
https://adafru.it/GEn

https://adafru.it/GEn

Python Docs

Python Docs (https://adafru.it/C56)

Calibration



(Calibration GUI image by Fabio Varesano)

The LSM303 chips are factory calibrated to a level of accuracy sufficient for many purposes. But for ultra-critical applications such as an IMU, you may want to further calibrate the device.

Ultimate Calibration:

For super-precise accelerometer calibration, you will want to check out the FreeIMU Magnetometer and Accelerometer GUI (https://adafru.it/c2d)by the late Fabio Varesano. The image above (from Fabio's site) shows a graphical representation of the sensor readings and the resulting calibration offsets calculated from the raw data.

This comprehensive calibration suite is designed to run on a PC. It is much too large to run on a microcontroller, like the Arduino, but the resulting calibration offsets it generates can be incorporated into your Arduino sketch for better accuracy.

Simplified Calibration:

A simpler method that still generates good results can be accomplished on the Arduino. This method uses a simple sketch to record the minimum and maximum readings on all 3 axis. While running the sketch, slowly rotate the LSM303 module multiple times in all three axis. The object is to record the absolute minimums and maximums for each axis, so the more you rotate it, the more likely you are to capture the absolute peak.

Be sure to rotate the sensor slowly about its center so that the accelerometer readings will represent just acceleration due to gravity and not linear acceleration of the sensor due to movement. After a while, the sketch output will stabilize. The values displayed will be the the min and max ranges for each axis and can be used to re-scale the output of the sensor.

The values obtained via the calibration sketch can be used to perform a 2-point calibration on each of the 3 axis: Two Point Calibration (https://adafru.it/Dva)

Calibration Sketch:
LSM303AGR

```
#include <Wire.h>
#include <Adafruit_Sensor.h>
#include <Adafruit_LIS2MDL.h>


Adafruit_LIS2MDL mag = Adafruit_LIS2MDL(12345);

float MagMinX, MagMaxX;
float MagMinY, MagMaxY;
float MagMinZ, MagMaxZ;

long lastDisplayTime;

void setup(void)
{
  Serial.begin(115200);
  Serial.println("LIS2MDL Calibration"); Serial.println("");

  /* Initialise the magnetometer */
  if(!mag.begin())
  {
    /* There was a problem detecting the LIS2MDL ... check your connections */
    Serial.println("Ooops, no LIS2MDL detected ... Check your wiring!");
    while(1);
  }
  lastDisplayTime = millis();
}

void loop(void)
{
  /* Get a new sensor event */
  sensors_event_t magEvent;

  mag.getEvent(&magEvent);

  if (magEvent.magnetic.x < MagMinX) MagMinX = magEvent.magnetic.x;
  if (magEvent.magnetic.x > MagMaxX) MagMaxX = magEvent.magnetic.x;

  if (magEvent.magnetic.y < MagMinY) MagMinY = magEvent.magnetic.y;
  if (magEvent.magnetic.y > MagMaxY) MagMaxY = magEvent.magnetic.y;

  if (magEvent.magnetic.z < MagMinZ) MagMinZ = magEvent.magnetic.z;
  if (magEvent.magnetic.z > MagMaxZ) MagMaxZ = magEvent.magnetic.z;

  if ((millis() - lastDisplayTime) > 1000)  // display once/second
  {
    Serial.print("Mag Minimums: "); Serial.print(MagMinX); Serial.print("  ");Serial.print(MagMinY);
Serial.print("  "); Serial.print(MagMinZ); Serial.println();
    Serial.print("Mag Maximums: "); Serial.print(MagMaxX); Serial.print("  ");Serial.print(MagMaxY);
Serial.print("  "); Serial.print(MagMaxZ); Serial.println(); Serial.println();
    lastDisplayTime = millis();
  }
}
```

LSM303/LSM303DLH

```
#include <Adafruit_Sensor.h>
```

```
#include <Wire.h>
//#include <Adafruit_LSM303_U.h>
#include <Adafruit_LSM303DLH_Mag.h>

Adafruit_LSM303DLH_Mag_Unified mag = Adafruit_LSM303DLH_Mag_Unified(12345);

float MagMinX, MagMaxX;
float MagMinY, MagMaxY;
float MagMinZ, MagMaxZ;

long lastDisplayTime;

void setup(void) {
  Serial.begin(115200);
  Serial.println("LSM303 Calibration");
  Serial.println("");

  /* Initialise the magnetometer */
  if (!mag.begin()) {
    /* There was a problem detecting the LSM303 ... check your connections */
    Serial.println("Ooops, no LSM303 detected ... Check your wiring!");
    while (1)
      ;
  }
  lastDisplayTime = millis();
}

void loop(void) {
  /* Get a new sensor event */
  sensors_event_t magEvent;

  mag.getEvent(&magEvent);

  if (magEvent.magnetic.x < MagMinX)
    MagMinX = magEvent.magnetic.x;
  if (magEvent.magnetic.x > MagMaxX)
    MagMaxX = magEvent.magnetic.x;

  if (magEvent.magnetic.y < MagMinY)
    MagMinY = magEvent.magnetic.y;
  if (magEvent.magnetic.y > MagMaxY)
    MagMaxY = magEvent.magnetic.y;

  if (magEvent.magnetic.z < MagMinZ)
    MagMinZ = magEvent.magnetic.z;
  if (magEvent.magnetic.z > MagMaxZ)
    MagMaxZ = magEvent.magnetic.z;

  if ((millis() - lastDisplayTime) > 1000) // display once/second
  {
    Serial.print("Mag Minimums: ");
    Serial.print(MagMinX);
    Serial.print("  ");
    Serial.print(MagMinY);
    Serial.print("  ");
    Serial.print(MagMinZ);
    Serial.println();
    Serial.print("Mag Maximums: ");
    Serial.print(MagMaxX);
    Serial.print("  ");
```

```
    Serial.print(MagMaxY);
    Serial.print("  ");
    Serial.print(MagMaxZ);
    Serial.println();
    Serial.println();
    lastDisplayTime = millis();
  }
}
```

Making Tracks!

Now let's use the LSM303 module to do some simple navigation!

The images on this page show the older LSM303DLH. You can use the newer LSM303AGR as well but you'll need to refer to the Pinouts page to determine which pins to use

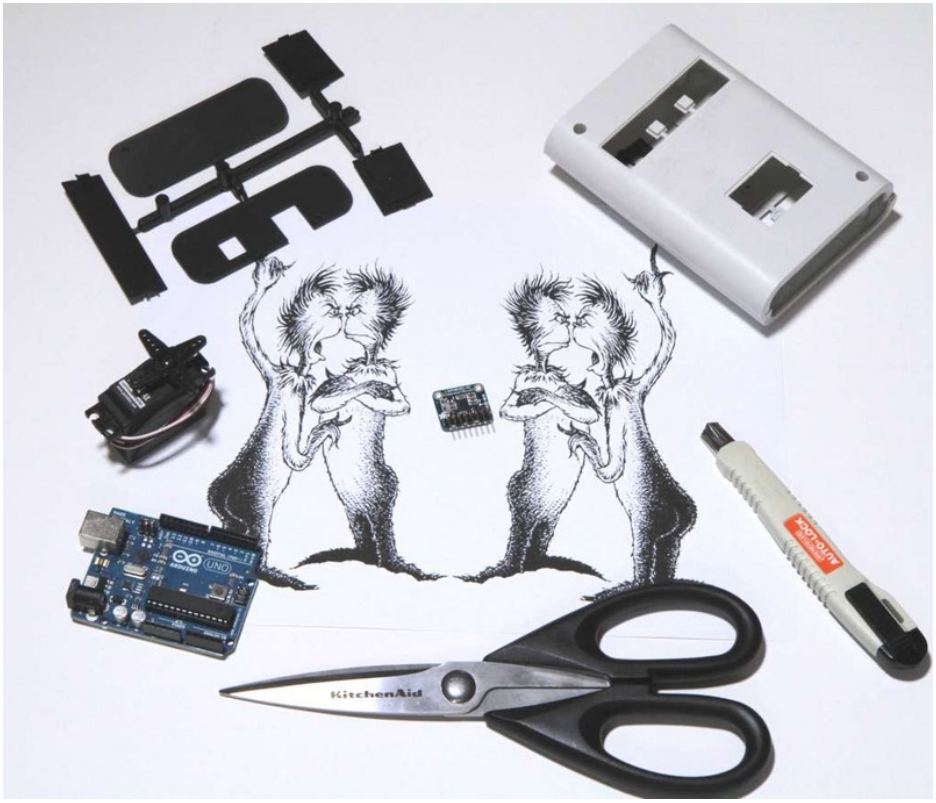
> *One day, making tracks*
> *In the prairie of Prax,*
> *Came a North-Going Zax*
> *And a South-Going Zax.*

This Zax-O-Meter is the perfect navigation instrument for either a North or a South-going Zax (https://adafru.it/c2e).

This project demonstrates how to use the LSM303 magnetometer output to implement a simple navigation system. No matter which way you turn, the pointer will always rotate to the desired direction of travel.

> *Never budge! That's my rule.*
> *Never budge in the least!*
> *Not an inch to the west!*
> *Not an inch to the east!*

The Zax-O-Meter uses the computed compass heading as feedback to a continuous rotation servo. When the compass heading is zero degrees (due north), the servo stops rotating. Any deviation from that value causes the servo to rotate in the opposite direction to compensate. This basic principle of negative feedback can be used to build a navigation system for an autonomous robot.



Materials:

To build the Zax-O-Meter, you will need:

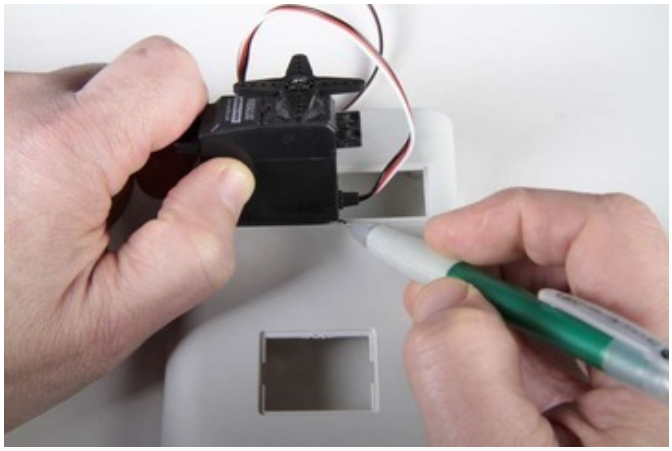
- Adafruit LSM303 Breakout Board (http://adafru.it/1120)
- Arduino Uno (http://adafru.it/50)
- Arduino Enclosure (http://adafru.it/271)
- Continuous Rotation Servo (http://adafru.it/154)
- Jumper Wires (https://adafru.it/aM5)
- Card-stock
- Cardboard or Foam Core

Calibrate your servo:

For the Zax-O-Meter to function accurately, you first need to find the 'neutral' point of your continuous rotation servo.

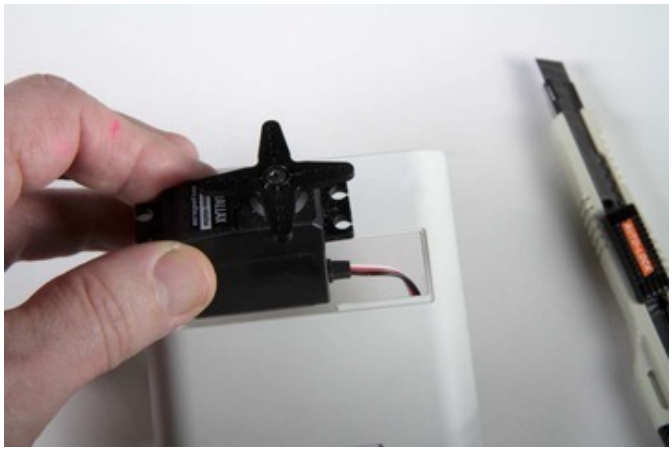
That is the servo output value that results in minimum rotation. This value is typically about 90, but varies somewhat between servos. Run this sketch and modify the value until you get minimum rotation. That is the value you should use for ServoNeutral in the Zax-O-Meter sketch.

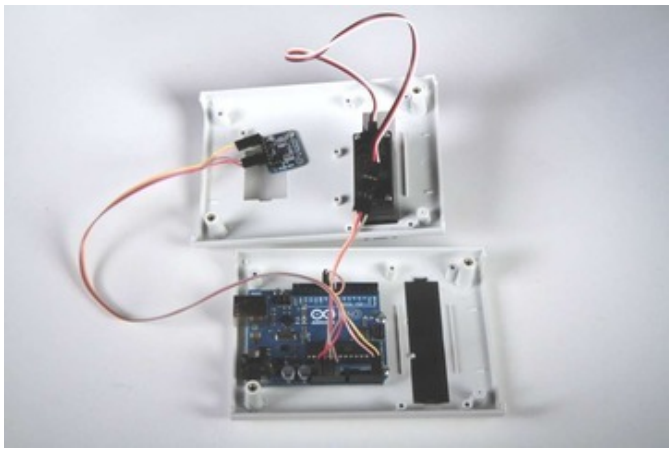
```
#include <Servo.h>
Servo servo;

void setup()
{
  servo.attach(9);  // attaches the servo on pin 9 to the servo object
  servo.write(90);  // change this value to achieve minimum rotation!
}

void loop()
{
}
```

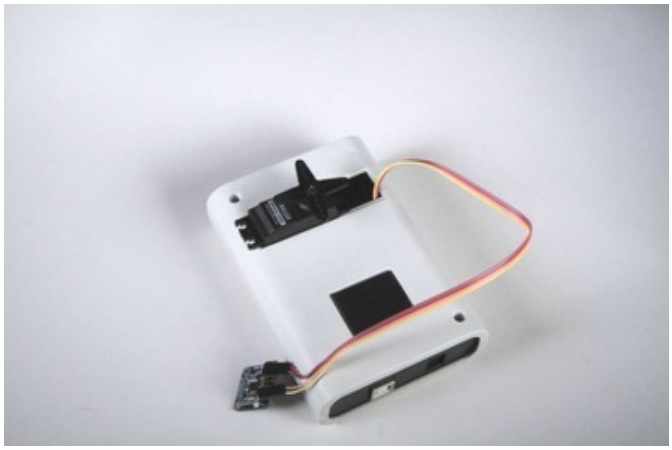


Mount the servo

Mark and widen the opening in the enclosure to fit the servo. Insert the servo with the rotor toward the center of the enclosure. Press down until the flanges are flush with the surface, the servo should snap into place and be held securely.







Mount the Uno and wire it up

Mount the Uno in the enclosure with the supplied screws. Wire the servo and sensor as follows:

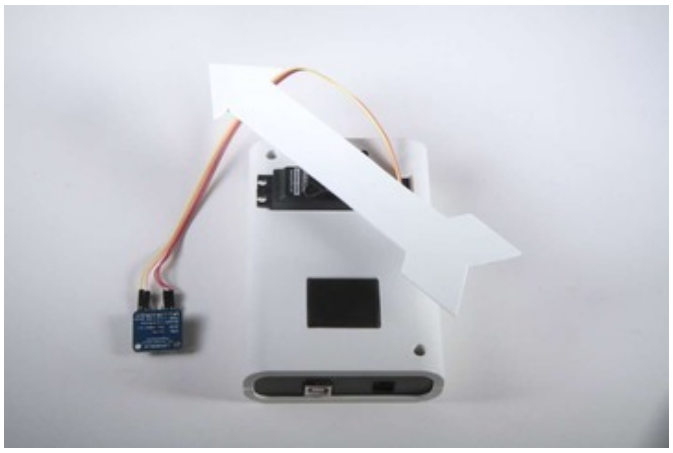
Servo:

- Black -> Gnd
- Red -> 5v
- White -> Digital Pin 9

LSM303:

- Gnd -> Gnd
- Vin -> 3.3v
- SDA -> Analog 4
- SCL -> Analog 5

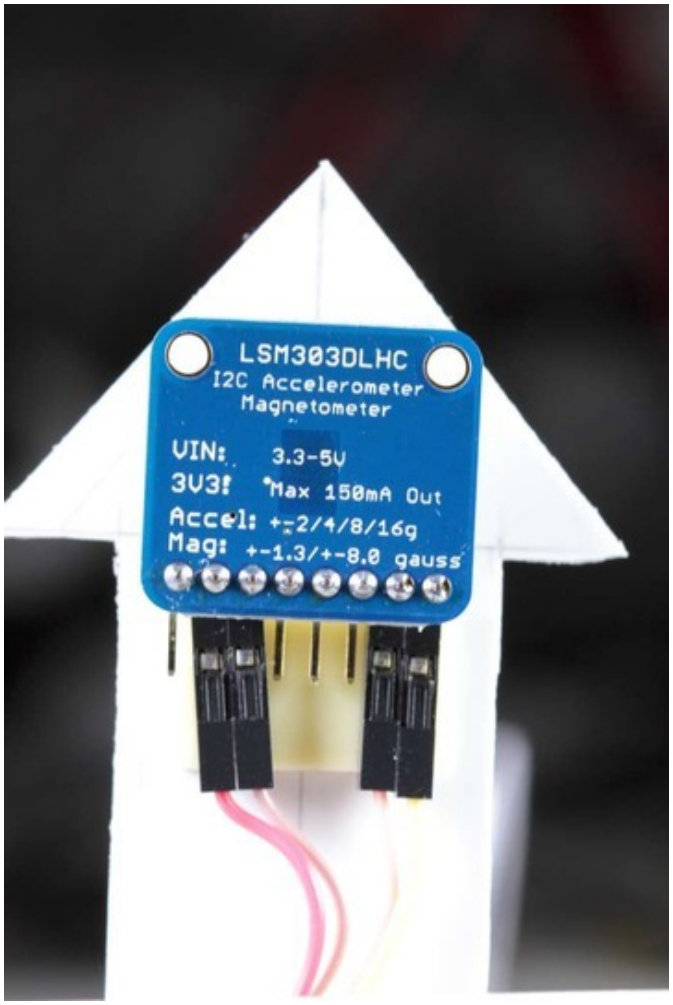
Then route the sensor wire through the opening next to the servo and close up the enclosure.



Add a pointer

Cut a pointer from some stiff cardboard or foam-core and attach it to the servo horn with some double-sided foam tape.

Attach the sensor to the underside of the arrow with some more double-sided foam tape. Locate the sensor as far away from the servo body as possible to avoid magnetic interference from the motor.

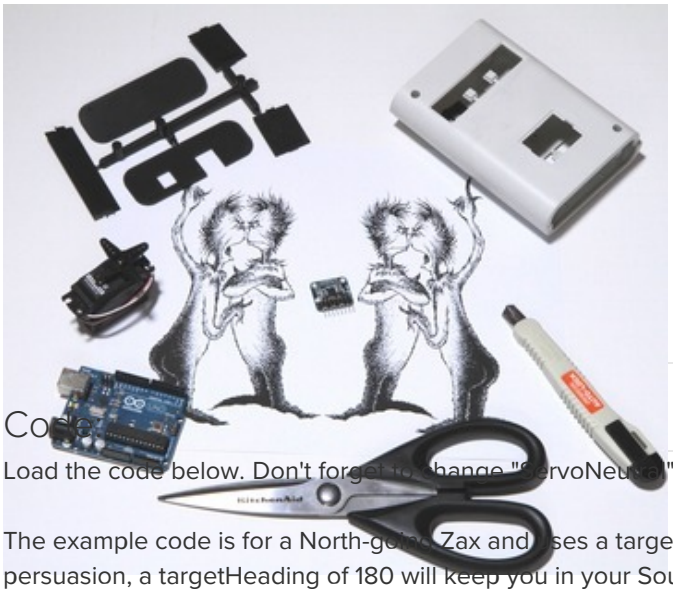


Add Zaxen!

Find an image of your favorite Zax (https://adafru.it/AI5). Use Paint or other image editing tool to make a mirror image pair.

Print the image on some heavy card-stock and fold it over to make a double-sided image.

Cut it out and mount to the top of your indicator arrow with some double-sided tape.



Code

Load the code below. Don't forget to change "ServoNeutral" to the neutral value from the Servo Calibration step.

The example code is for a North-going Zax and uses a targetHeading of 0. If you are a Zax of the South-going persuasion, a targetHeading of 180 will keep you in your South-going groove. For those with a rebellious streak, choose any targetHeading setting from 0 - 360 degrees and start making tracks in the heading of your choice!

```
// ******************************************
// Zax-O-Meter Sketch
// for the Adafruit LSM303 Magnetometer Breakout
//
// Written by Bill Earl for Adafruit Industries
//
// ******************************************

#include <Wire.h>
#include <Adafruit_Sensor.h>
#include <Adafruit_LSM303_U.h>
#include <Servo.h>

/* Assign a unique ID to this sensor at the same time */
Adafruit_LSM303_Mag_Unified mag = Adafruit_LSM303_Mag_Unified(12345);

// This is our continuous rotation servo
Servo servo;

// Pi for calculations - not the raspberry type
const float Pi = 3.14159;

// This is the value that gives you minimal rotation on
// a continuous rotation servo.  It is usually about 90.
// adjust this value to give minimal rotation for your servo
const float ServoNeutral = 97;

// This is the desired direction of travel
// expressed as a 0-360 degree compass heading
// 0.0 = North
// 90.0 = East
// 180.0 = South
// 270 = West
const float targetHeading = 0.0;


void setup(void)
{
```

```
{
  Serial.begin(9600);
  Serial.println("Magnetometer Test"); Serial.println("");

  /* Initialise the sensor */
  if(!mag.begin())
  {
    /* There was a problem detecting the LSM303 ... check your connections */
    Serial.println("Ooops, no LSM303 detected ... Check your wiring!");
    while(1);
  }

  servo.attach(9);  // Attach servo to pin 9
}

void loop(void)
{
  /* Get a new sensor event */
  sensors_event_t event;
  mag.getEvent(&event);

  // Calculate the angle of the vector y,x
  float heading = (atan2(event.magnetic.y,event.magnetic.x) * 180) / Pi;
  // Normalize to 0-360
  if (heading < 0)
  {
    heading = 360 + heading;
  }

  // Calculate the error between tha measured heading and the target heading.
  float error = heading - targetHeading;
  if (error > 180)
  {
    error = error - 360;  // for angles > 180, correct in the opposite direction.
  }
  // A non-zero difference between the heading and the
  // targetHeading will bias the servoNeutral value and
  // cause the servo to rotate back toward the targetHeading.
  // The divisor is to reduce the reaction speed and avoid oscillations
  servo.write(ServoNeutral + error / 4 );

  delay(40);
}
```

Downloads and Links

Files

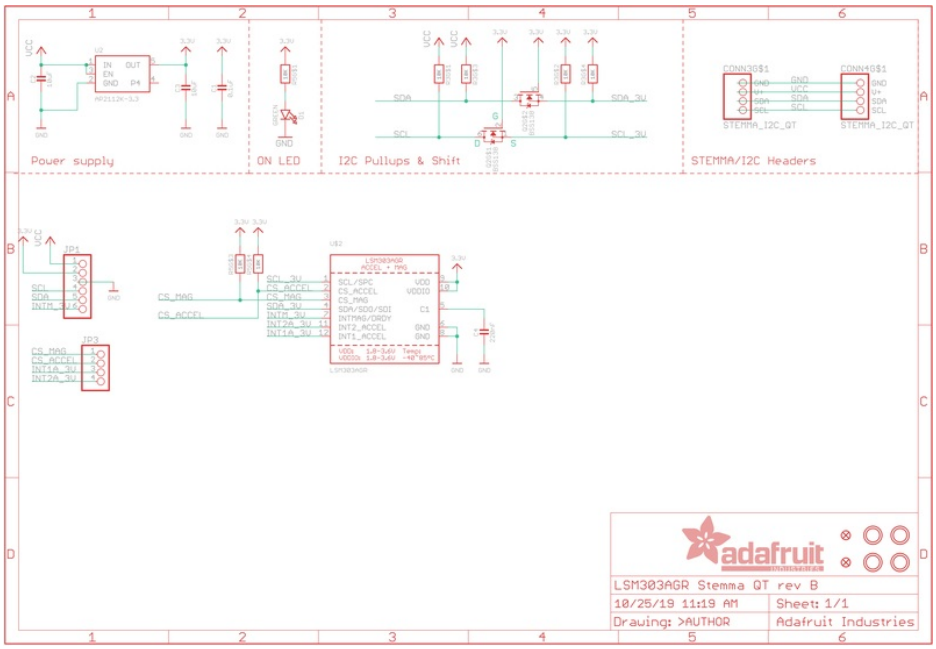
LSM303AGR

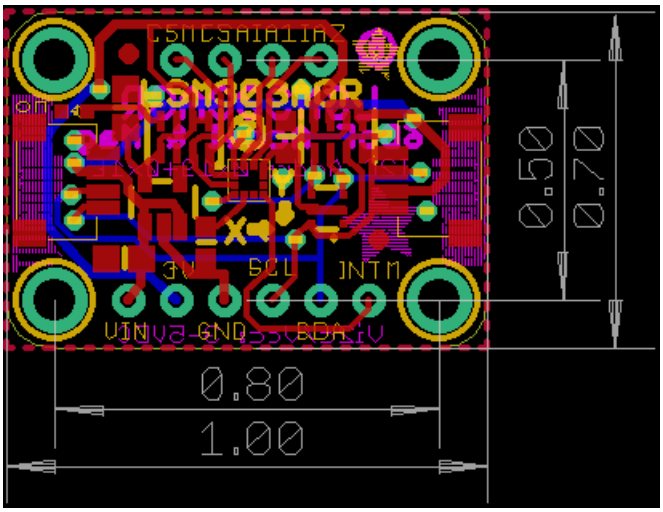
- LSM303AGR datasheet (https://adafru.it/Gub)
- PCB Files (Eagle Format) (https://adafru.it/Guc)
- Fritzing object in the Adafruit Fritzing library (https://adafru.it/Gud)

LSM303DLHC

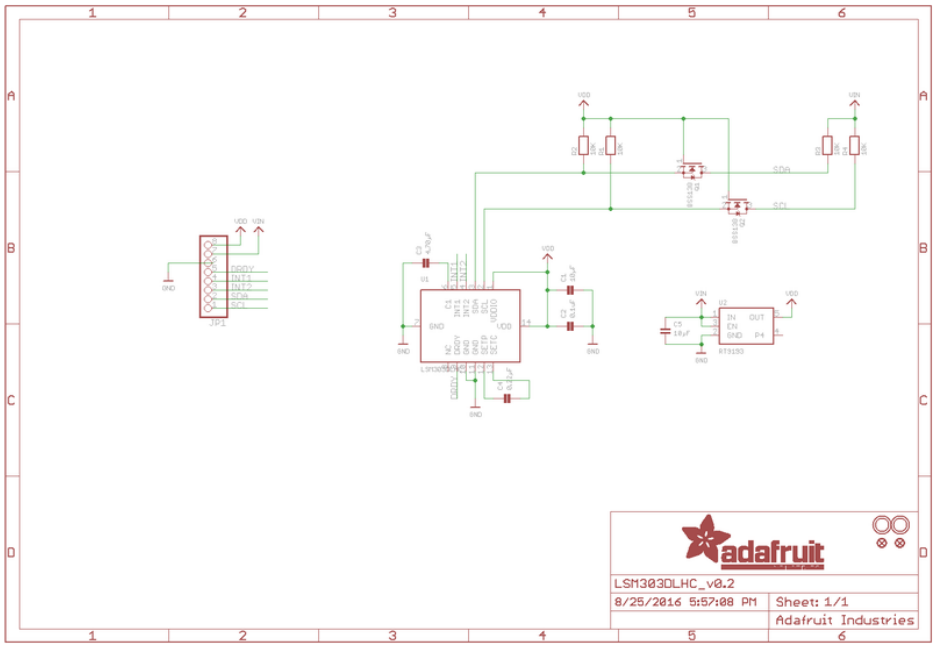
- LSM303DLHC datasheet (https://adafru.it/rgd)
- PCB Files (Eagle Format) (https://adafru.it/cCn)
- Fritzing object in the Adafruit Fritzing library (https://adafru.it/aP3)

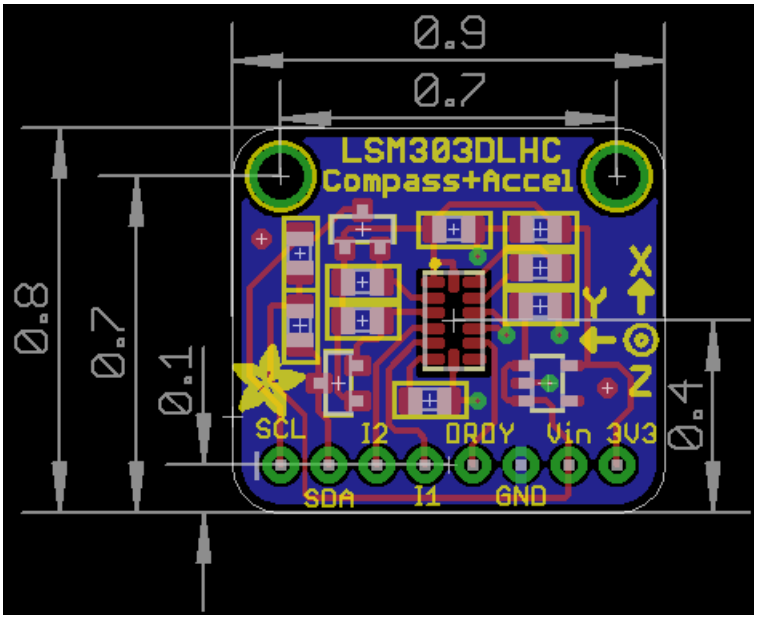
LSM0303AGR Schematic & Fab Print





LSM303DLHC Schematic & Fabrication Print




0.9
0.7
0.8
0.7
0.1
0.4
LSM303DLHC
Compass+Accel
X
Y
Z
SCL
I2
DRDY
Vin
3V3
SDA
I1
GND