adafruit learning system

LSM303 Accelerometer + Compass Breakout

Created by Bill Earl

Guide Contents

Overview

The LSM303 (http://adafru.it/1120) 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 and use it to measure orientation relative to the earth's magnetic field, and acceleration in three axis.

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.

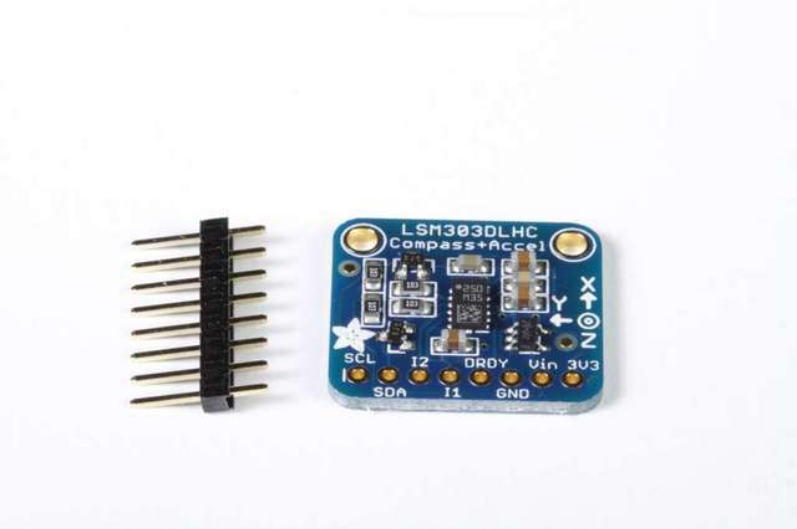
Acceleration Measurement

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

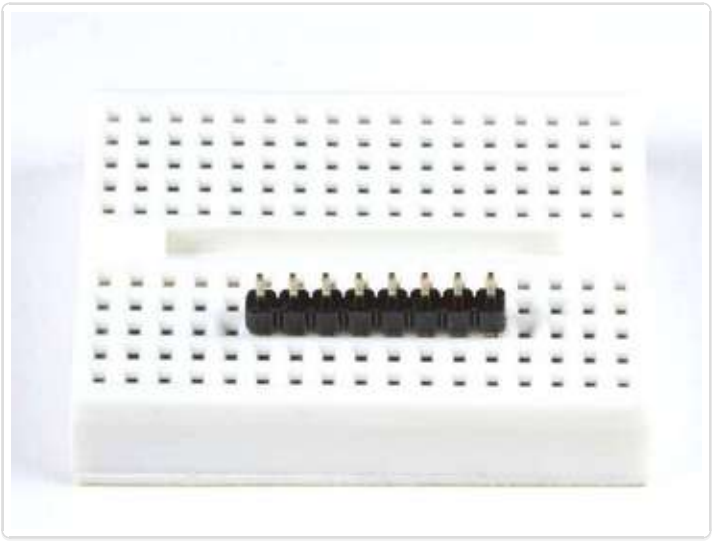
These structures are similar to the accelerometer structures, but are etched with microscopic coils. An excitation current is passed through the coils, and the Lorentz Force (http://adafru.it/c25) due to the magnetic field causes the structure to deflect. Once again the deflection is converted to an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

Assembly and Wiring



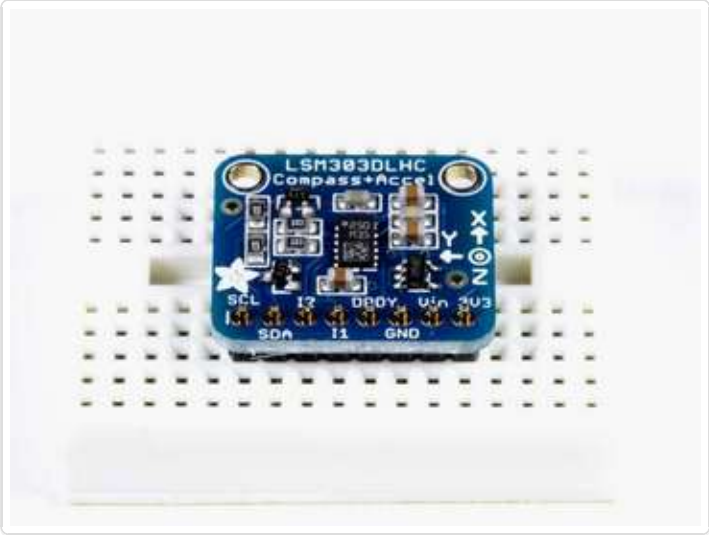
Board Assembly:

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.



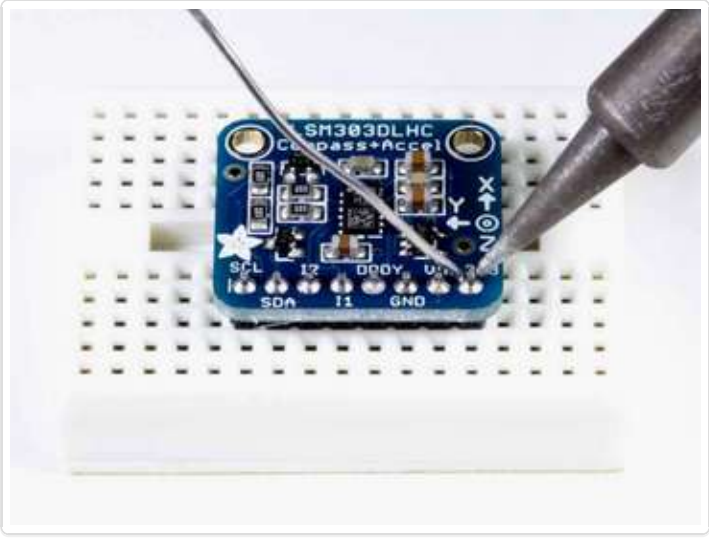
Position the header

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!

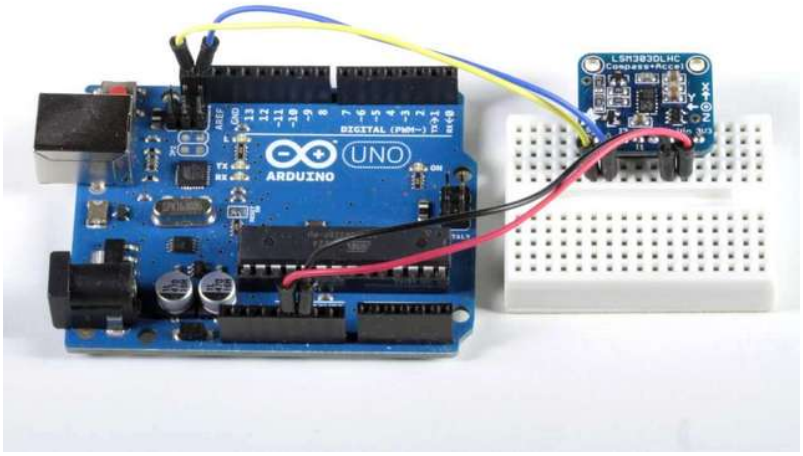
Solder each pin to assure a good electrical connection.

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

Connect to the Arduino:

The LSM303 breakout requires only 4 wires:

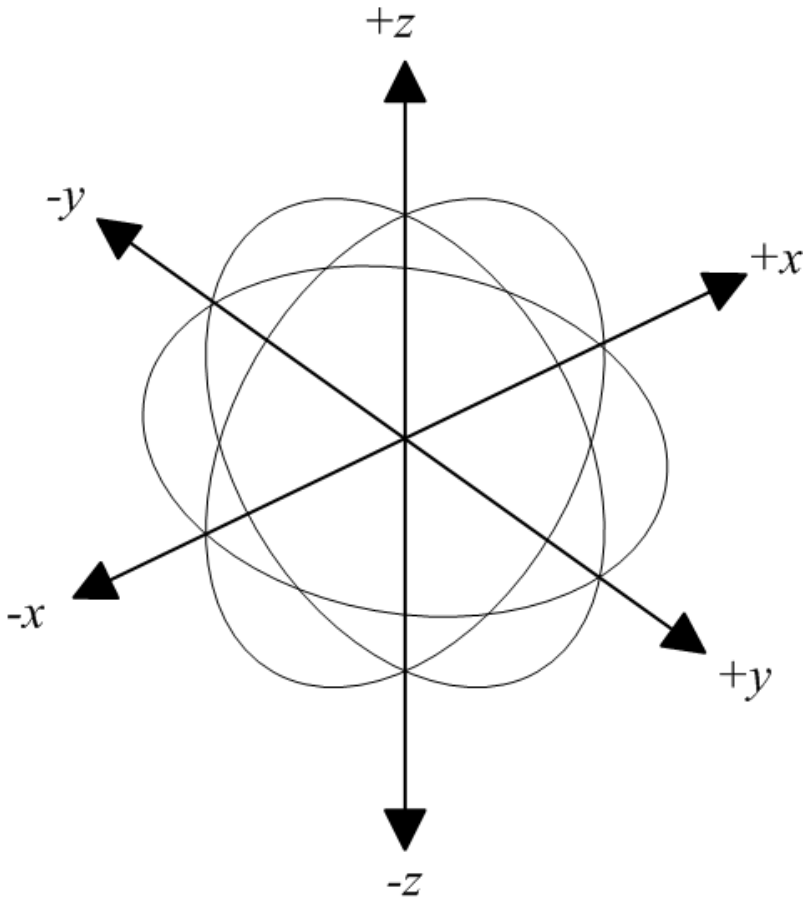
- GND->GND
- VIN->5v (It will also work on 3.3v)
- SDA->SDA (A4 on 'classic' Arduinos)
- SCL->SCL (A5 on 'classic' Arduinos)


LSM303DLHC
Compass+Accel
ARDUINO
UNO
DIGITAL (PWM~)

Using the LSM303

Download the Library

To get started with the LSM303, first download and install (http://adafru.it/aYM) the Adafruit_LSM303DLHC library (http://adafru.it/aYU) from Github. Since this driver is part of the Adafruit Unified Sensor system, you will also need to download and install the Adafruit_Sensor library (http://adafru.it/aZm).



Basic Accelerometer Readings

The Adafruit_LSM303_Accel sensor class in the Adafruit_LSM303 library reports X, Y and Z axis accelerometer readings directly in meters per second squared (http://adafru.it/c26). The AccelSensor example code (http://adafru.it/c27) 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 (http://adafru.it/c28).

Basic Magnetometer Readings

The Adafruit_LSM303_Mag sensor class in the Adafruit_LSM303 library reports X, Y and Z axis magnetometer readings directly in micro-Teslas (http://adafru.it/c29). The MagSensor example code (http://adafru.it/c2a) in the library 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

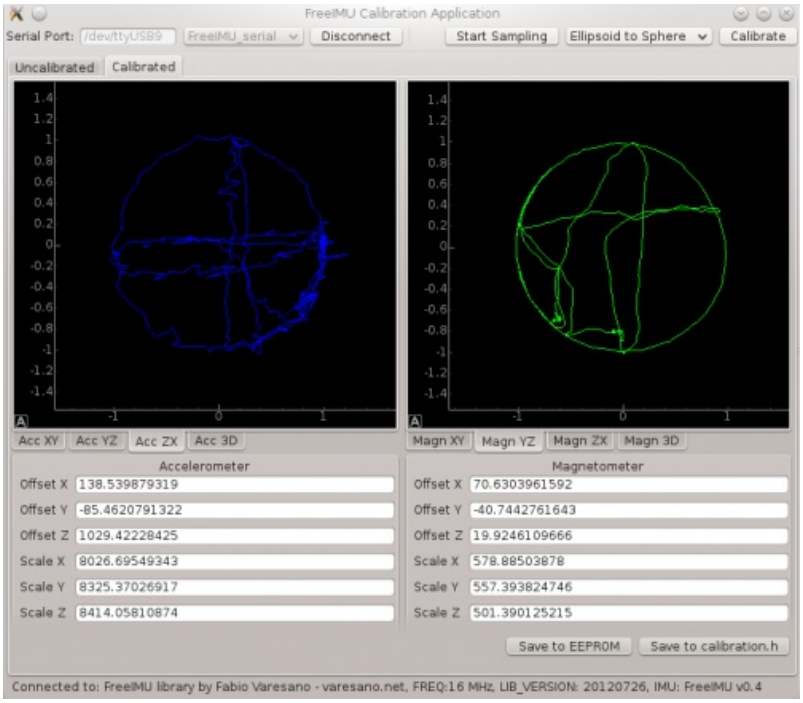
To convert the microTesla readings into a 0-360 degree compass heading, we can use the atan2() function (http://adafru.it/c2b) to compute the angle of the vector defined by the Y and X axis readings. The result will be in radians, so we multiply by 180 degrees and divide by Pi to convert that to degrees.

The following sketch will print the compass heading in degrees to to the serial monitor:

```
#include <Wire.h>
#include <Adafruit_Sensor.h>
#include <Adafruit_LSM303.h>

/* Assign a unique ID to this sensor at the same time */
Adafruit_LSM303_Mag mag = Adafruit_LSM303_Mag(12345);

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);
  }
}

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);
}
```

Calibration



(Calibration GUI image by Fabio Varesano (http://adafru.it/c2c))

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 calibration of the LSM303, you will want to check out the FreeIMU Magnetometer and Accelerometer GUI (http://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.

Calibration Sketch:

```
#include <Wire.h>
#include <Adafruit_Sensor.h>
#include <Adafruit_LSM303.h>

/* Assign a unique ID to these sensors */
Adafruit_LSM303_Accel accel = Adafruit_LSM303_Accel(54321);
Adafruit_LSM303_Mag mag = Adafruit_LSM303_Mag(12345);

float AccelMinX, AccelMaxX;
float AccelMinY, AccelMaxY;
float AccelMinZ, AccelMaxZ;

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

long lastDisplayTime;

void setup(void)
{
  Serial.begin(9600);
  Serial.println("LSM303 Calibration"); Serial.println("");

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

  accel.getEvent(&accelEvent);
  mag.getEvent(&magEvent);

  if (accelEvent.acceleration.x < AccelMinX) AccelMinX = accelEvent.acceleration.x;
  if (accelEvent.acceleration.x > AccelMaxX) AccelMaxX = accelEvent.acceleration.x;
```

```
  if (accelEvent.acceleration.y < AccelMinY) AccelMinY = accelEvent.acceleration.y;
  if (accelEvent.acceleration.y > AccelMaxY) AccelMaxY = accelEvent.acceleration.y;

  if (accelEvent.acceleration.z < AccelMinZ) AccelMinZ = accelEvent.acceleration.z;
  if (accelEvent.acceleration.z > AccelMaxZ) AccelMaxZ = accelEvent.acceleration.z;

  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("Accel Minimums: "); Serial.print(AccelMinX); Serial.print("  ");Serial.print(AccelMinY); S
    Serial.print("Accel Maximums: "); Serial.print(AccelMaxX); Serial.print("  ");Serial.print(AccelMaxY)
    Serial.print("Mag Minimums: "); Serial.print(MagMinX); Serial.print("  ");Serial.print(MagMinY); Serial.
    Serial.print("Mag Maximums: "); Serial.print(MagMaxX); Serial.print("  ");Serial.print(MagMaxZ); Ser
    lastDisplayTime = millis();
  }
}
```

Making Tracks!

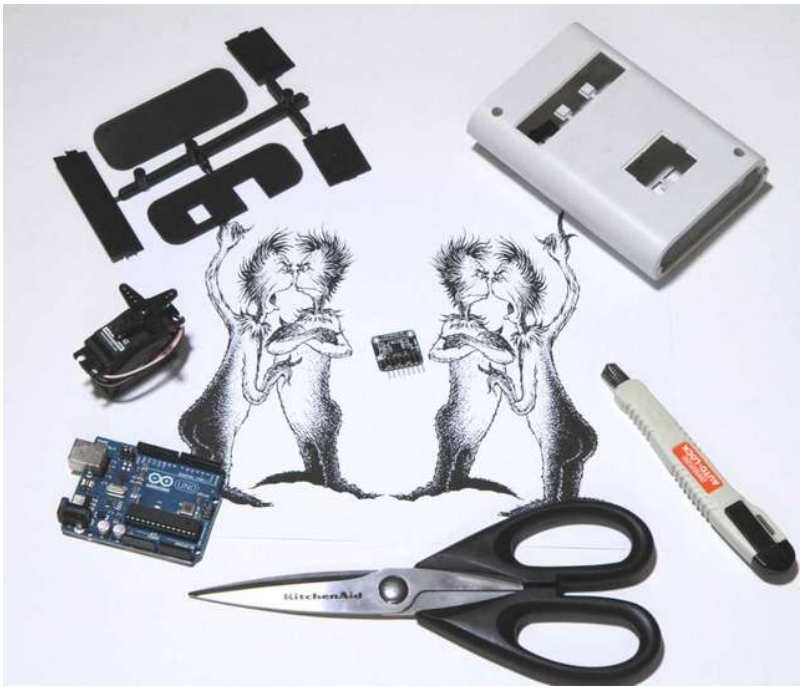
Now let's use the LSM303 module to do some simple navigation!

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 (http://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:

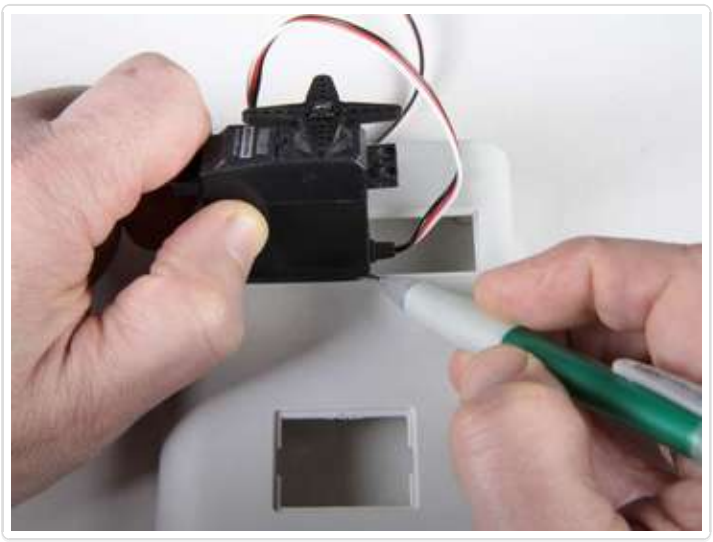
- Adafruit LSM303 Breakout Board (http://adafru.it/1120)
- Arduino Uno (http://adafru.it/50)
- Arduino Enclosure (http://adafru.it/271)
- Continuous Rotation Servo (http://adafru.it/154)
- Jumper Wires (http://adafru.it/aM5)
- Card-stock
- Cardboard or Foam Core

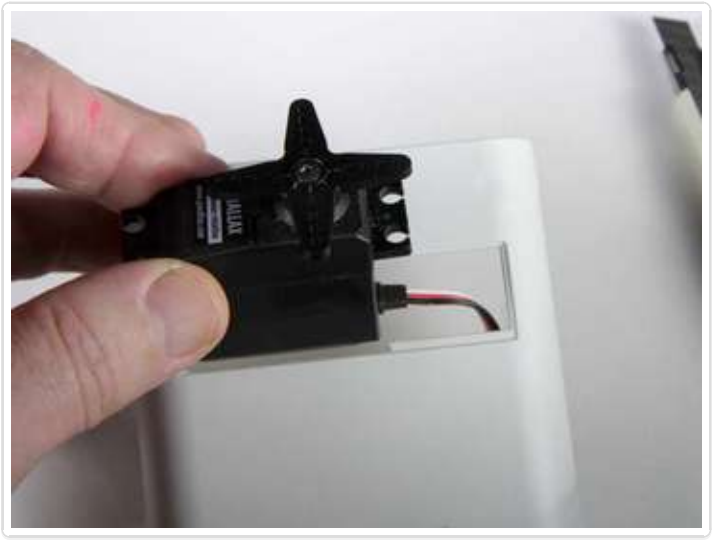
Calibrate your servo:

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;
```

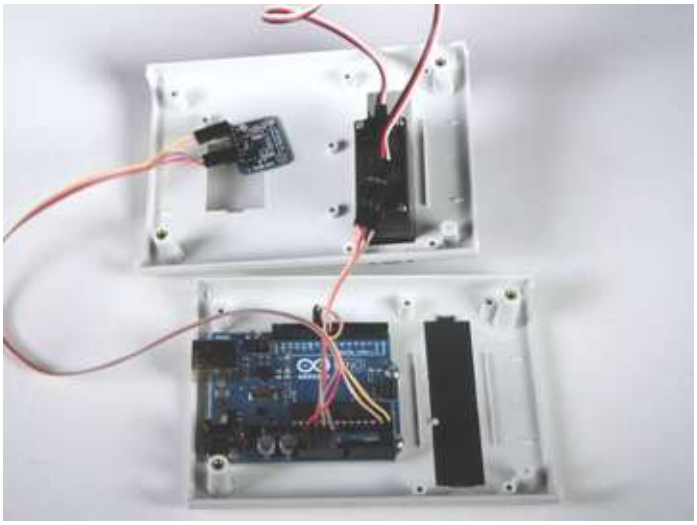
```
void setup()
{
  servo.attach(9);  // attaches the servo on pin 9 to the servo object
  servo.write(90);  // change this value to achieve minimum rotation!
}

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:

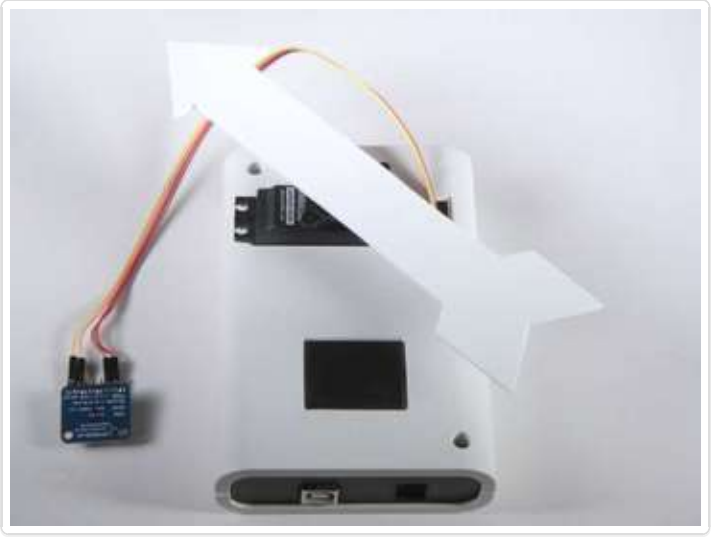
Servo:

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

LSM303:

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

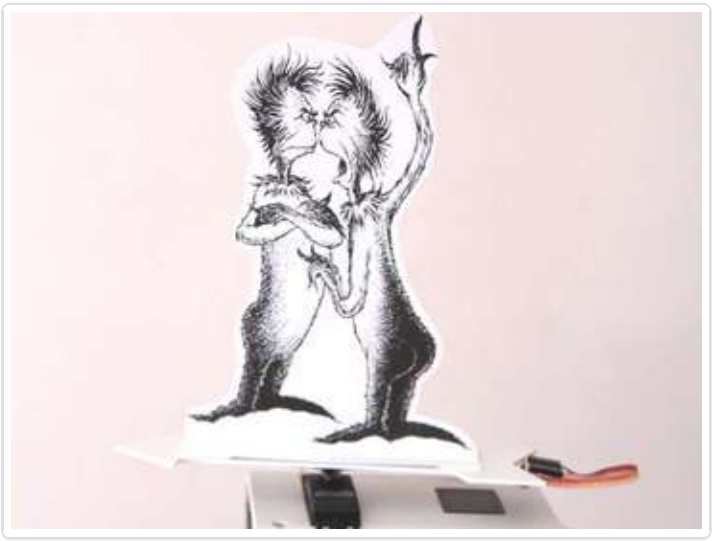
Then route the sensor wire through the opening next to the servo and close up the enclosure.

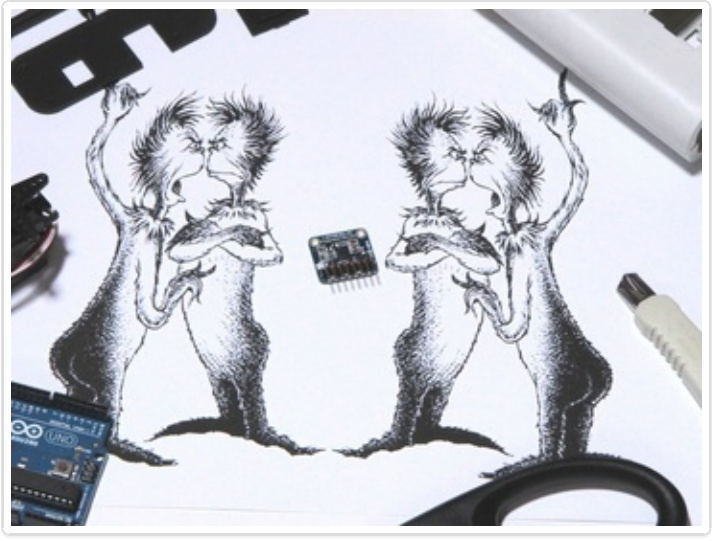


Add a pointer

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 (http://adafru.it/c2f). 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.h>
#include <Servo.h>

/* Assign a unique ID to this sensor at the same time */
Adafruit_LSM303_Mag mag = Adafruit_LSM303_Mag(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);
}
```