OVERVIEW

This is a breakout board for Honeywell's HMC5883L, a 3-axis magnetometer. Magnetometers have a wide range of uses. The most common include using the chip as a digital compass to sense direction or using them to detect ferrous (magnetic) metals.

Magnetic fields and current go hand-in-hand. When current flows through a wire, a magnetic field is created. This is the basic principle behind electromagnets. This is also the principle used to measure magnetic fields with a magnetometer. The direction of Earth's magnetic fields affects the flow of electrons in the sensor, and those changes in current can be measured and calculated to derive a compass heading or other useful information.

The breakout board includes the HMC5883 sensor and all filtering capacitors necessary. There are two unpopulated pads in case you need pull up resistors (no need for them if you're using the Arduino/ATmega328).

OPERATION

Communication with the HMC5883 is simple and all done through an I2C interface.

1. Wire:
   - attach the SDA line to A4
   - attach the SCL line to A5
   - attach Vcc to 3.3V
   - attach GND to GND.
2. Download and install the HMC5883L library (unpack to Arduino libraries):
   http://faculty.unlv.edu/eelabs/soft/HMC5883L.zip

**CODE 1**

```c
// Reference the I2C Library
#include <Wire.h>
// Reference the HMC5883L Compass Library
#include <HMC5883L.h>

// Store our compass as a variable.
HMC5883L compass;
// Record any errors that may occur in the compass.
int error = 0;

// rounding angle
int RoundDegreeInt;
int PreviousDegree = 0;

// Out setup routine, here we will configure the microcontroller and compass.
void setup()
{
  // Initialize the serial port.
  Serial.begin(9600);

  Wire.begin(); // Start the I2C interface.

  compass = HMC5883L(); // Construct a new HMC5883 compass.
  error = compass.SetScale(1.3); // Set the scale of the compass.
  if(error != 0) // If there is an error, print it out.
```
Serial.println(compass.GetErrorText(error));

error = compass.SetMeasurementMode(Measurement_Continuous); // Set the measurement mode to Continuous
if(error != 0) // If there is an error, print it out.
   Serial.println(compass.GetErrorText(error));
}

// Our main program loop.
void loop()
{
   // Retrieve the raw values from the compass (not scaled).
   MagnetometerRaw raw = compass.ReadRawAxis();
   // Retrieved the scaled values from the compass (scaled to the configured scale).
   MagnetometerScaled scaled = compass.ReadScaledAxis();

   // Values are accessed like so:
   int MilliGauss_OnThe_XAxis = scaled.XAxis;// (or YAxis, or ZAxis)

   // Calculate heading when the magnetometer is level, then correct for signs of axis.
   float heading = atan2(scaled.YAxis, scaled.XAxis);

   // Once you have your heading, you must then add your 'Declination Angle', which is the 'Error' of the magnetic field in your location.
   // Find yours here: http://www.magnetic-declination.com/
   // Mine is: 61deg 22' W, which is 61.37 Degrees, or (which we need)
   // 0.0456752665 radians, I will use 0.0457
   // If you cannot find your Declination, comment out these two lines, your compass will be slightly off.
   float declinationAngle = 1.071;
   heading += declinationAngle;

   // Correct for when signs are reversed.
   if(heading < 0)
      heading += 2*PI;

   // Check for wrap due to addition of declination.
   if(heading > 2*PI)
      heading -= 2*PI;

   // Convert radians to degrees for readability.
   float headingDegrees = heading * 180/M_PI;

   //correcting the angle issue
   if (headingDegrees >= 1 && headingDegrees < 240)
   {
      headingDegrees = map(headingDegrees,0,239,0,179);
   }
   else if (headingDegrees >= 240)
   {
      headingDegrees = map(headingDegrees,240,360,180,360);
   }

   // rounding the angle
RoundDegreeInt = round(headingDegrees);

// smoothing value
if( RoundDegreeInt < (PreviousDegree + 3) && RoundDegreeInt >
(PreviousDegree - 3) ) {
    RoundDegreeInt = PreviousDegree;
}
Output(RoundDegreeInt);

PreviousDegree = RoundDegreeInt;

// Normally we would delay the application by 66ms to allow the loop
// to run at 15Hz (default bandwidth for the HMC5883L).
// However since we have a long serial out (104ms at 9600) we will let
// it run at its natural speed.
// delay(66);

// Output the data down the serial port.
void Output(int RoundDegreeInt)
{
    //Serial.println();
    Serial.println(RoundDegreeInt);
    delay(150);
}

CODE 2

import processing.serial.*;

Serial myPort;
PFont b;

int lf = 10;  // Linefeed in ASCII
String myString = null;
float angle;

void setup(){
    size(600,400);
    b = loadFont("Arial-BoldMT-48.vlw");
    myPort = new Serial(this, "COM10", 9600);
}

void draw(){
    background(255);

    while (myPort.available() > 0) {
        myString = myPort.readStringUntil(lf);
        if (myString != null) {
            print(myString);  // Prints String
            angle = float(myString);  // Converts and prints float
println(angle);
}
translate(160, 50);

// draw the compass background
ellipseMode(CENTER);
fill(50);
stroke(10);
strokeWeight(2);
ellipse(150, 150, 300, 300);

// draw the lines and dots
translate(150, 150); // translate the lines and dots to the middle of the compass
float CompassX = -angle;
rotate(radians(CompassX));
noStroke();
fill(51, 255, 51);

int radius = 120;

for( int degC = 5; degC < 360; degC += 10) //Compass dots
{
    float angleC = radians(degC);
    float xC = 0 + (cos(angleC)* radius);
    float yC = 0 + (sin(angleC)* radius);
    ellipse(xC, yC, 3, 3);
}

for( int degL = 10; degL < 370; degL += 10) //Compass lines
{
    float angleL = radians(degL);
    float x = 0 + (cos(angleL)* 145);
    float y = 0 + (sin(angleL)* 145);

    if( degL==90 || degL==180 || degL==270 || degL==360) {
        stroke(51, 255, 51);
        strokeWeight(4);
    } else {
        stroke(234, 144, 7);
        strokeWeight(2);
    }
    line(0, 0, x, y);
}

fill(102, 102, 102);
noStroke();
ellipseMode(CENTER);
ellipse(0, 0, 228, 228); //draw a filled circle to hide the lines in the middle

b = loadFont("Arial-BoldMT-48.vlw");
textAlign(CENTER);
EXPERIMENTS

Experiment 1
1. Wire the circuit and sensor as mentioned in “Operation” section
2. Run code1 to observe magnet operation
3. Demonstrate the results to the TA

Experiment 2
1. Modify:
   a. add a code to Output function, that together with the angle it also prints the direction in words ("north", "south", "west", "east")
2. Demonstrate the results to the TA

Experiment 3
1. Modify:
   a. Add 3 leds: left, middle, center
   b. Light up center LED when heading North
   c. Light up left LED when heading North-West
   d. Light up right LED when heading North-East
   e. Output "N", "NW", "NE" to the Serial Monitor, according to current angle
2. Demonstrate the results to the TA
POSTLAB REPORT DELIVERIES

Include the following elements in your postlab report:

1. Theory of operation
   a. Describe how the magnet sensor works
   b. List 3 different devices, where the electrical magnet sensor is used

2. Results of the experiments
   For each experiment, include:
   a. The code that you developed for the experiment. Each line that was added must be highlighted and commented with the explanation of what is its meaning.
   b. Brief explanation how the goal of the experiment was reached
   c. Screenshots of the serial monitor with the values, presenting the operation of your code
   For experiment 4, explain your method. Write your conclusions about the differences between library and your results.

3. Answer the questions:
   a. Is the output of the IR sensor analog or digital?
   b. Find out the technical data about the sensor used:
      i. Detecting distance
      ii. Operating temperature
      iii. Operating supply voltage

4. Conclusions
   a. Write down your conclusions, things learned, problems encountered during the lab and how they were solved, etc.

REFERENCES

https://www.sparkfun.com/tutorials/301