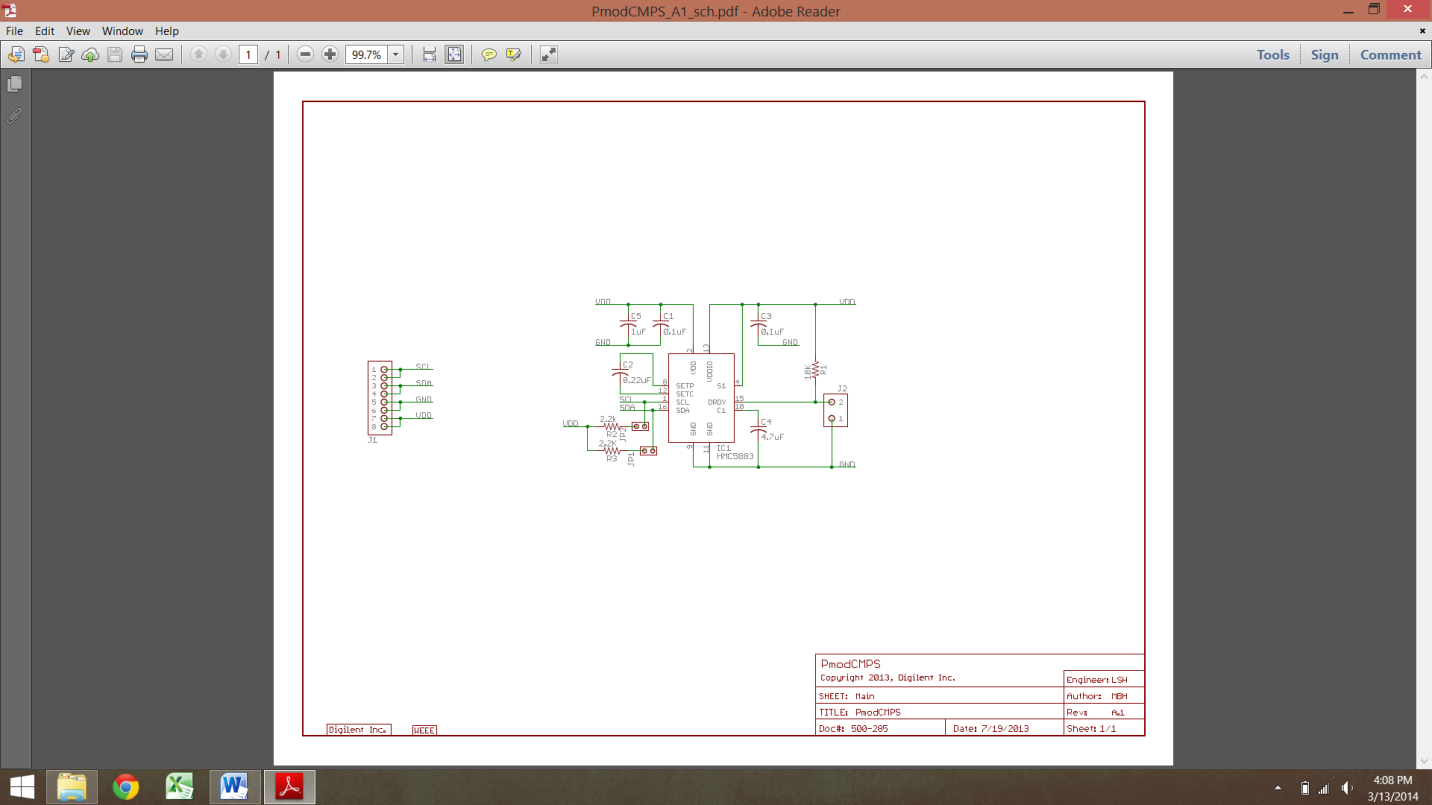
Husain Rahim - Quadcoper

**PmodCMPS – 3-Axis Digital Compass**

Our digital compass features the Honeywell HMC5883L function which can be added to our MPIDE by adding the library and example code. Our Pmod CMPS can be configured to run in single measurement or continuous measurement mode. Basically a 3 axis digital compass sensor is designed for low field magnetic sensing. The magnetic field measured on each axis is represented as a signed 16-bit word with the most significant bits being the sign bits. There are two eight bit registers for each axis. It can also detect location using the three axis x, y, & z. Our sensor has a 1 to 2 degrees compass heading accuracy. The Pmod CMPS connects to the chip kit through an I2C interface. This sensor is capable of 160 Hz maximum data output rate.

**Connecting 3-Axis Digital Compass to ChipKit**

They’re four pins that we are utilizing on our compass sensor. The four pins are SDA, SCL, VIN, & GND. These pins are connected to data, clock, voltage in, and ground on the ChipKit respectively. Once these pins are connected you can then run the example code below to output the results for your sensor. The sensor should output values for x, y, & z depending on the location of the sensor, the values will differ when the sensor is moved to different locations.



// 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;

// Out setup routine, here we will configure the microcontroller and compass.

void setup()

{

// Initialize the serial port.

Serial.begin(9600);

Serial.println("Starting the I2C interface.");

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

Serial.println("Constructing new HMC5883L");

Serial.println("Setting scale to +/- 1.3 Ga");

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));

Serial.println("Setting measurement mode to continous.");

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()

{

// Retrive the raw values from the compass (not scaled).

MagnetometerRaw raw = compass.readRawAxis();

// Retrived 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: -2¡ã37' which is -2.617 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 = -0.0457;

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;

// Output the data via the serial port.

Output(raw, scaled, heading, headingDegrees);

// 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);//of course it can be delayed longer.

}

// Output the data down the serial port.

void Output(MagnetometerRaw raw, MagnetometerScaled scaled, float heading, float headingDegrees)

{

Serial.print("Raw:\t");

Serial.print(raw.XAxis);

Serial.print(" ");

Serial.print(raw.YAxis);

Serial.print(" ");

Serial.print(raw.ZAxis);

Serial.print(" \tScaled:\t");

Serial.print(scaled.XAxis);

Serial.print(" ");

Serial.print(scaled.YAxis);

Serial.print(" ");

Serial.print(scaled.ZAxis);

Serial.print(" \tHeading:\t");

Serial.print(heading);

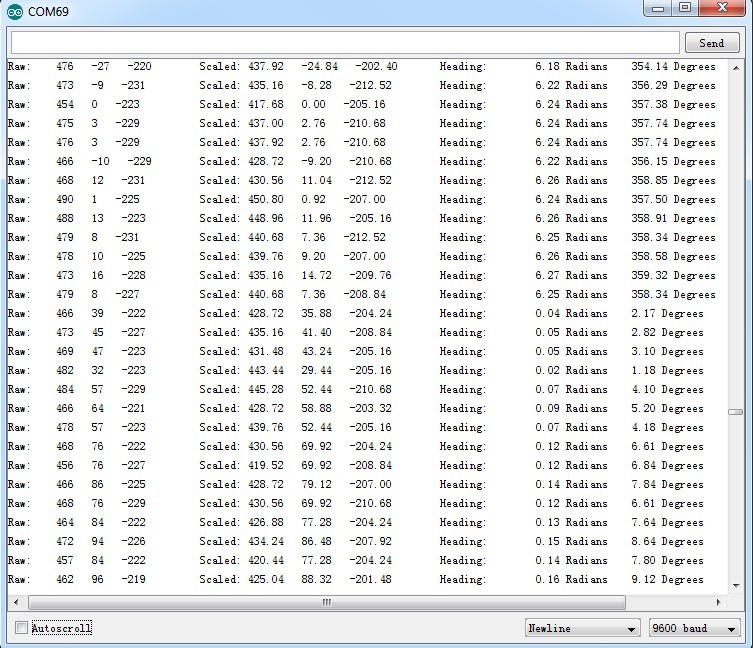
Serial.print(" Radians \t");

Serial.print(headingDegrees);

Serial.println(" Degrees \t");

}

After running the example code you should then open the serial monitor to see your results. In the picture below you can see the values for x, y, &z. It also displays the heading in degrees and radians. These results display different values depending on the location of your sensor.



**Quad-copter**

The 3 axis digital compass is a useful sensor for our quad-copter because it tells us our compass heading. We are able to determine which direction our copter is flying based on the compass heading. Although it’s not necessary in our project it can also detect low magnetic fields. In our project we are mostly using this sensor for compass heading and keeping track of our axis values. This sensor will also be combined with an accelerometer in our project for more accurate readings.