The following is a description of how the data is obtained for our Senior Design Compass module. All the references made can be found at http://www.phermans.com/sd.

All data is an average of 5 samples of the data obtained by the function “GetData” in ‘SensorComm.h’ of our Final Firmware build which can be found on our wiki. A brief summary of this function: (1) Puts the sensor into heading mode and gets the heading, (2) puts sensor in X Mag mode and gets the value and converts to a float, (3) puts the sensors into Y Mag mode and gets the value and converts to a float. These three values are then averaged for each sensor 5 times as indicated by the “AverageData” function found in ‘HeadingMath.h’.

Following the GetData sequence the data then needs to be aligned and normalized. This is all done in the “AlignData” function found in ‘HeadingMath.h’ as well. This function first centers the X and Y Mag voltages to zero and then normalizes the amplitude. These functions are described below. *Note this is all done in our firmware; nothing has been changed in the EEPROM of the actual sensor or anything of that nature

Centering the X & Y Mag voltages around zero is done by subtracting off the zero gauss chamber readings found. This makes the value zero actually represent zero, due to manufacturing inconsistencies and such a zero gauss field does not create an output of 0 as it should. The value outputted in a zero gauss field was found and is then subtracted such that when there is a zero gauss field the X or Y Mag data will actually be zero. As an example, in a 0 gauss field the sensor X Mag may be 12 when it should be 0 under perfect conditions; therefore, our firmware now subtracts 12 from all X Mag readings of that sensor to compensate. The values we found and are which are subtracted are called “Sensor_X_Zeros” and “Sensor_Y_Zeros” which can be found in ‘main.h’ and will be appended to this email.

Normalizing the X and Y Mag readings is done by multiplying the value by a predetermined amplitude offset. This offset was found by creating the sinusoidal waves forms that each sensors X and Y Mag voltages make when rotated 360 degrees. The wave forms where created by rotating 360 sampling at every 1 degree and then plotting the output. The amplitudes for each waveform (each sensor X and Y) where found and then normalized to the smallest amplitude. The process was done 3 times and the normalized values then averaged. They are stored in the respect X and Y arrays “Sensor_X_Amp” and “Sensor_Y_Amp” which can be found in ‘main.h’ and will also be attached to this email.

The description above is the first step of nearly all functions in our firmware, this way we get the same data in the same way for everything we do. The data described above is what can be found in the “AccuracyData” tab of Excel files posted on the wiki and the associated text file. It contains the data for a full rotation of 360 degrees sampled every degree for which the heading, X Mag and Y Mag data is recorded for every sensor. The PMC degree is also recorded as well as our processed heading, control heading, and Honeywell heading. These last three headings are found using further manipulation and functions all found in our final firmware build. The “RepeatData” tab of the Excel files contains this same data but is recorded 10 times at an interval of 10 degrees for a full roation of 360 degrees and is used to determine the repeatability of the compass. The “ModData” tab is simple an Excel tab that references the “AccuracyData” and “RepeatData” tabs to computed the accuracy, repeatability, and precision. The graphs are then made by referencing various results in the “ModData” tab.

I apologize that we do not have ‘raw’ data directly read from each sensor. However, we have provided you with data that can easily converted back to an average of 5 readings from the sensors by adding back the zero offsets and dividing out the amplitude normalization values. Furthermore, Beth still has all the hardware and software to obtain more data if that is required. Our firmware can be easily modified to output unadjusted and un-averaged data if so desired as well. I hope this makes the data we have found clearer and will aid you in further progress. If you have any further questions feel free to email me and I can answer your questions and if need be set up a phone interview.

Here are the offset referred to above.

```
const float Sensor_X_Zeros[11] =    {5.0699,
                                     6.2446,
                                     0.000,
                                     -1.7715,
                                     -0.7249,
                                     -2.0246,
                                     -1.9214,
                                     -3.4851,
                                     1.1142,
                                     -2.0805,
                                     1.2243};


const float Sensor_Y_Zeros[11] =    {-0.9654,
                                     2.4297,
                                     0.00,
                                     7.7352,
                                     -4.9016,
                                     -11.6652,
                                     0.3106,
                                     3.3111,
                                     2.2879,
                                     -5.4272,
                                     5.7407};

const float Sensor_X_Amp[11] =      {0.9924,
                                     1.000,
                                     0,
                                     0.9899,
                                     0.9896,
                                     0.9913,
                                     0.9886,
                                     0.9911,
                                     0.9899,
                                     0.9884,
                                     0.9873};

const float Sensor_Y_Amp[11] =      {0.9924,
                                     0.9780,
                                     0,
                                     0.9900,
                                     0.9924,
                                     0.9933,
                                     0.9877,
                                     0.9885,
                                     0.9913,
                                     0.9963,
                                     0.9895};
```