Gateway To Space
ASEN 1400 / ASTR 2500

Class #10

Colorado Space Grant Consortium
Announcements:

- Hardware from last Tuesday, HW 4 and HW 5 needed today
Arduino Part:
Catch Up and microSD

Those not interested in Arduinos may leave now

Please return at 10:35 to pick up proposal comments

Colorado Space Grant Consortium
Let’s Verify Some Items:

- When doing math in code, use float variables

- When doing math, you must show zero before and after decimal point
Let’s Verify Some Items:

- 5V (pin 1) = E3
- Data (pin 2) = E2
- GND (pin 3) = E1

Please VERIFY Orientation now

PIN 1, +Vs; PIN 2, Vout; PIN 3, GND
Let’s Verify Some Items:

- Verify this wiring
Let’s Verify Some Items:

- 3.3 V power and ground for Accelerometer
Let’s Verify Some Items:

- Connect Vcc to SLP
- Forces the accelerometer to always stay on

Sleep Mode

The 3 axis accelerometer provides a Sleep Mode that is ideal for battery operated products. When Sleep Mode is active, the device outputs are turned off, providing significant reduction of operating current. A low input signal on pin 7 (Sleep Mode) will place the device in this mode and reduce the current to 3 μA typ. For lower power consumption, it is recommended to set g-Select to 1.5g mode. By placing a high input signal on pin 7, the device will resume to normal mode of operation.
Let’s Verify Some Items:

- Connect Vcc to GSEL

- Sets range to ±6 G
Let’s Verify Some Items:

- Connect XOUT to A1, YOUT to A2, & ZOUT to A3 on Arduino
**Accelerometers:**

- Connect XOUT to A1, YOUT to A2, & ZOUT to A3 on Arduino
Let’s Verify Some Items:

- Variables used to store analog read, should be `int`
Let’s Verify Some Items:

- Convert to Voltage = \(\text{accelX} \times (5.0/1023)\)

- Subtract offset and convert to Gs
  \(\text{Gs} = (\text{voltage} - 1.65) / (0.206)\)
Let’s Verify Some Items:

- These should be float

```c
// int led = 9;
int sensorValue;
float sensorVoltage;
float tempC;
int accelX;
int accelY;
int accelZ;

float accelXV;
float accelYV;
float accelZV;
float accelXG;
float accelYG;
float accelZG;
```
Let’s Verify Some Items:

- Use your zeros before and after decimal point

```cpp
void loop() {
  // put your main code here, to run repeatedly:

  sensorValue = analogRead(A0);
  sensorVoltage = sensorValue*(5.0/1023);
  tempC = (sensorVoltage - 0.5)/(0.01);  
  accelX = analogRead(A1);
  accelXV = accelX*(5.0/1023);
  accelXG = (accelXV - 1.65)/(0.206);
  accelY = analogRead(A2);
  accelYV = accelY*(5.0/1023);
  accelYG = (accelYV - 1.65)/(0.206);
  accelZ = analogRead(A3);
  accelZV = accelZ*(5.0/1023);
  accelZG = (accelZV - 1.65)/(0.206);

  Serial.print("accel XG ");
  Serial.print(accelXG);
  Serial.print(" \t accel YG ");
  Serial.print(accelYG);
  Serial.print(" \t accel ZG ");
  Serial.print(accelZG);
}
Let’s Verify Some Items:

- Modify the print statements

```c
accelZG = (accelZV - 1.65)/(0.206);

Serial.print("accelXG");
Serial.print(acceXG);
Serial.print("\t accelYG");
Serial.print(acceYG);
Serial.print("\t accelZG");
Serial.print(acceZG);
Serial.print(sensorValue);
Serial.print("\t Voltage ");
Serial.print(sensorVoltage);
Serial.print("\t Temp C ");
Serial.println(tempC);
```
Let’s Verify Some Items:

- Compile and Upload
- Who is not getting these results?

PLEASE SAVE YOUR SKETCH FILE
Let’s Verify Some Items:

- Verify Humidity Sensor Wiring

5V to 5 V power Rail

GND to GND Rail

OUT to A5 on the Arduino
Humidity Sensor:

- Add your humidity sensor as shown

5V to 5 V power Rail

GND to GND Rail

OUT to A5 on the Arduino
Humidity Sensor:

- Define your variables

```c
// Definitions
int humidity;
float humidityV;
float RH;
int sensorValue;
```
### Humidity Sensor:

- Look at the data sheet to understand output of the sensor
- We know $V_{out}$ and $V_{supply}$ so using algebra

<table>
<thead>
<tr>
<th>Voltage output (1st order curve fit)</th>
<th>$V_{out} = (V_{supply})(0.0062(\text{sensor RH}) + 0.16)$, typical at 25 °C</th>
</tr>
</thead>
<tbody>
<tr>
<td>Temperature compensation</td>
<td>True RH = $(\text{Sensor RH})/(1.0546 - 0.00216T)$, $T$ in °C</td>
</tr>
</tbody>
</table>
**Humidity Sensor:**

- Here’s the algebra and the equation to code

\[
V_{\text{OUT}} = \left(V_{\text{SUPPLY}}\right)\left(0.0062\left(sensor\,RH\right) + 0.16\right)
\]

\[
\left(\frac{V_{\text{OUT}}}{\left(V_{\text{SUPPLY}}\right)} - 0.16\right) = sensor\,RH
\]

\[
sensor\,RH = \left(\frac{V_{\text{OUT}}}{\left(5.0\,V\right)} - 0.16\right)\]

\[
= \frac{0.0062}{0.0062}
\]
**Humidity Sensor:**

- Read the analog value, convert to voltage and then convert to RH%

- Serial print the RH to the monitor

```
accelZG = (accelZV - 1.65)/(0.206);

humidity = analogRead(A5);
humidityV = humidity*(5.0/1023);
RH = ((humidityV/5.0)-0.16)/0.0062);

//  Serial.print("Heading ");
//  Serial.print(currentVector);
Serial.print("RH ");
Serial.print(RH);
Serial.print("t accel XG ");
```
Humidity Sensor:

- Set the range for your LED Visual Display
- RH values for 10, 20, 30 and 40%
- Verify and upload your code
- Launch serial monitor

```cpp
if(RH > 10) {
    digitalWrite(2, HIGH);
}
if(RH > 20.0) {
    digitalWrite(3, HIGH);
}
if(RH > 30.0) {
    digitalWrite(4, HIGH);
}
if(RH > 40.0) {
    digitalWrite(5, HIGH);
}
delay(100);
```
Humidity Sensor:

- Breathe like Darth Vader on your humidity sensor
- See the changes on your monitor and LEDs
Humidity Sensor:

- While waiting for the rest of the group, play with your new sensor
- Also, look at the data sheet and determine the voltage at maximum humidity

PLEASE SAVE YOUR SKETCH FILE
Humidity Sensor:

- % RH is linear as function of voltage
- 100% RH looks like ~3.7 V
Part 3 – Arduino Road Trip

A. Temperature Sensor
B. Accelerometers
C. Humidity Sensor
D. Pressure Sensor
E. Compass
Pressure Sensor:

Arduino Uno

Monitor ↔ PC/Mac

LEDs

Accels
Pressure
Temp
Humidity
Pressure Sensor:

- Pressure is our last sensor we will use for our BalloonSat
- Can use it to determine pressure/altitude of payload
- Fragile sensor and $$$
Pressure Sensor:

- Pressure sensor orientation
Pressure Sensor:

- Pressure sensor orientation

<table>
<thead>
<tr>
<th>Pin 1</th>
<th>Pin 2</th>
<th>Pin 3</th>
<th>Pin 4</th>
</tr>
</thead>
<tbody>
<tr>
<td>NC</td>
<td>$V_{\text{supply}}$</td>
<td>OUTPUT+</td>
<td>GND</td>
</tr>
</tbody>
</table>
Pressure Sensor:

- Connect GND to Pin 4, 5V to Pin 2, and Pin 2 to A4 on the Arduino

<table>
<thead>
<tr>
<th>Pin 1</th>
<th>Pin 2</th>
<th>Pin 3</th>
<th>Pin 4</th>
</tr>
</thead>
<tbody>
<tr>
<td>NC</td>
<td>V_{supply}</td>
<td>OUTPUT+</td>
<td>GND</td>
</tr>
</tbody>
</table>
**Pressure Sensor:**

- Connect GND to Pin 4, 5V to Pin 2, and Pin 2 to A4 on the Arduino
Pressure Sensor:

- Overall picture of BalloonSat Sensor Suite
**Pressure Sensor:**

- Open your Humidity Sketch and save as Pressure
- Modify as follows:

- Add 3 new variables

```cpp
int pressure;
float pressureV;
float psi;

int humidity;
float humidityV;
float RH;
```
Pressure Sensor:

- Look at the data sheet to understand output of the sensor

- Known:
  V_{supply} = 5.0 \, \text{V}
  P_{max} = 15.0 \, \text{psi}
  P_{min} = 0.0 \, \text{psi}
  Output(V) = \text{measured}
  Pressure \text{ applied} = \text{solve}

\[
Output\, (V) = \frac{0.8 \times V_{supply}}{P_{max} - P_{min}} \times (Pressure_{applied} - P_{min}) + 0.10 \times V_{supply}
\]
Pressure Sensor:

- Here’s the algebra and the equation to code

\[
\text{Output}(V) = \frac{0.8 \times V_{\text{SUPPLY}}}{(P_{\text{max}} - P_{\text{min}})} \times (\text{pressure}_{\text{applied}} - P_{\text{min}}) + 0.10 \times V_{\text{SUPPLY}}
\]

\[
\text{Output}(V) = \frac{(0.8 \times 5.0)}{(15.0 - 0.0)} \times (\text{pressure}_{\text{applied}} - 0.0) + 0.10 \times 5.0
\]

\[
\text{Output}(V) = \frac{(4.0)}{(15.0)} \times (\text{pressure}_{\text{applied}}) + 0.5
\]

\[
\frac{15.0}{4.0} \times (-0.5 + \text{Output}(V)) = \text{pressure}_{\text{applied}}
\]
**Pressure Sensor:**

- Read the analog value, convert to voltage and then convert to psi

- Serial print the psi to the monitor

```cpp
accelZg = (accelZV - 1.65)/(0.206);

pressure = analogRead(A4);
pressureV = pressure*(5.0/1023);
psi = (pressureV-0.5)*(15.0/4.0);

humidity = analogRead(A5);
humidityV = humidity*(5.0/1023);
RH = (((humidityV/5.0)-0.16)/0.0062);

// Serial.print("Heading ");
// Serial.print(currentVector);
Serial.print("PSI ");
Serial.print(psi);
Serial.print(" \t RH ");
```
**Pressure Sensor:**

- Adjust the ranges in your code to use your LED Visual Display

- Suggest 13, 9, 5, and 1.0 psi

- Change from > to <

```c
if(psi < 13.0) {
    digitalWrite(2, HIGH);
}
if(psi < 9.0) {
    digitalWrite(3, HIGH);
}
if(psi < 5.0) {
    digitalWrite(4, HIGH);
}
if(psi < 1.0) {
    digitalWrite(5, HIGH);
}
delay(100);
```
**Pressure Sensor:**

- Carefully push tubing on end of sensor and suck

- **DO NOT BLOW** or **DO NOT APPLY PRESSURE**; it will break the sensor

- See the changes on your monitor and LEDs
Pressure Sensor:

- While waiting for the rest of the group, play with your new sensor

- Try to get your sensor to zero

PLEASE SAVE YOUR SKETCH FILE
To Add More Sensors…

…to the Arduino, what do we do?

We are out of analog space
To Add More Sensors…

- To complicate matters, the compass requires two analog channels

- One solution is to use digital temp sensors

- Another solution is to use an Arduino with more analog channels

- We will address this issue at the evening session tonight
Part 3 – Arduino Road Trip

A. Temperature Sensor  
B. Accelerometers  
C. Humidity Sensor  
D. Pressure Sensor  
E. Compass
Part 4 – Arduino Data Storage

A. SHIELD Integration
B. SD Card Code Integration
C. Data Retrieval
D. Accelerometer Calibration
E. Compass
Micro SD Card Shield:

- Micro SD Card Shield
- Arduino Uno
- Monitor
- PC/Mac
- LEDs
- Accels
- Pressure
- Temp
- Humidity
Micro SD Card Shield:

- Disconnect USB cable
- One wire at a time, disconnect wires from Arduino and reconnect to micro SD card shield
Micro SD Card Shield:

- With Arduino unpowered, connect the SD card shield (wires not shown)
Micro SD Card Shield:

- With Arduino unpowered, connect the SD card shield (wires not shown)
Micro SD Card Shield:

- Insert the micro SD card as shown (wires not shown)
**Micro SD Card Shield:**

- Insert the micro SD card as shown (wires not shown)
Micro SD Card Shield:

- Reconnect USB and rerun same code
Micro SD Card Shield:

- Everyone should get the same results
Part 4 – Arduino Data Storage

A. SHIELD Integration
B. SD Card Code Integration
C. Data Retrieval
D. Accelerometer Calibration
E. Compass
**SD Card Code:**

- Everyone should get the same results
SD Card Code:

- Go to the class website and download Gateway Sensor Code
SD Card Code:

- My version of your code but gives us a starting point before going down the rabbit hole.
- My version of your code but gives us a starting point before going down the rabbit hole.

**SD Card Code:**

```cpp
void setup() {
  // put your setup code here, to run once:

  Serial.begin(9600);

  // setup the LED Visual Display
  pinMode(2, OUTPUT);
  pinMode(3, OUTPUT);
  pinMode(4, OUTPUT);
  pinMode(5, OUTPUT);
}

void loop() {
  // put your main code here, to run repeatedly:

  temp = analogRead(A0);
  tempV = temp*(5.0/1024);
  tempC = (tempV - 0.5)/(0.01);

  accelX = analogRead(A1);
  accelXV = accelX*(5.0/1024);
  accelXG = (accelXV - 1.65)/(0.206);

  accelY = analogRead(A2);
  accelYV = accelY*(5.0/1024);
  accelYG = (accelYV - 1.65)/(0.206);

  accelZ = analogRead(A3);
  accelZV = accelZ*(5.0/1024);
  accelZG = (accelZV - 1.65)/(0.206);
```
My version of your code but gives us a starting point before going down the rabbit hole.

```
pressure = analogRead(A4);
pressureV = pressure*(5.0/1024);
psi = (pressureV-0.5)*(15.0/4.0);

humidity = analogRead(A5);
humidityV = humidity*(5.0/1024);
RH = (((humidityV/5.0)-0.16)/0.0062);

Serial.print("PSI ");
Serial.print(psi);
Serial.print(" \t RH ");
Serial.print(RH);
Serial.print(" \t accel XG ");
Serial.print(accelXG);
Serial.print(" \t accel YG ");
Serial.print(accelYG);
Serial.print(" \t accel ZG ");
Serial.print(accelZG);
Serial.print(" \t Temp C ");
Serial.println(tempC);

digitalWrite(2, LOW);
digitalWrite(3, LOW);
digitalWrite(4, LOW);
digitalWrite(5, LOW);
```
SD Card Code:

- My version of your code but gives us a starting point before going down the rabbit hole.

```cpp
if(psi < 13.0) {
    digitalWrite(2, HIGH);
}
if(psi < 9.0) {
    digitalWrite(3, HIGH);
}
if(psi < 5.0) {
    digitalWrite(4, HIGH);
}
if(psi < 1.0) {
    digitalWrite(5, HIGH);
}
delay(100);
```
- Upload code and verify results look similar
SD Card Code:

- Now let’s explore the code needed to record this data to an SD Card

“This is your last chance ... After this, there is no turning back. You take the blue pill - the story ends, you wake up in your bed, and believe whatever you want to believe.

You take the red pill, ... you stay in Wonderland, and I show you, how deep the rabbit-hole goes.”

~ Morpheus’ Warning To Neo (From The Film; “The Matrix”) ~
SD Card Code:

- SD card writing uses libraries built into the Arduino software

- Code gets a little more complicated at this point
SD Card Code:

- New Stuff...

```c
// Include Files
#include <SD.h>

// Controls for the data logging system
const int LOG_INTERVAL = 50;  // milli seconds between entries

// Time keeper
uint32_t timeStamp = 0;      // The time stamp used when recording data points

const int chipSelect = 8;    // This is set to 8 for the SparkFun uSDCard shield

// Variable for file name
char logFileName[16];

// Use this LED for your writing indicator
int led = 5;
int ledState = 0;
```
**SD Card Code:**

- Define character variables for data storage

```c
// Analog temperature sensor variables
int temp;
float tempV;
float tempC;
char tempCString[?];

// Analog humidity sensor variables
int humidity;
float humidityV;
float RH;
char RHString[?];

// Analog accelerometers variables
int accelX;
int accelY;
int accelZ;
float accelXV;
float accelYV;
float accelZV;
float accelXG;
float accelYG;
float accelZG;
char accelXGString[?];
char accelYGString[?];
char accelZGString[?];
```
**SD Card Code:**

- Define character variables for data storage

```c
// Analog pressure sensor variables
int pressure;
float pressureV;
float psi;

char psiString[7];

// This variable will hold the data from a read cycle
String dataString = "";
```
**SD Card Code:**

- Define a string variable to contain data headers

```java
// Titles for column headings
String sensorNames = "Time Stamp (ms), Temp (°C), AccelX (g), AccelY (g), AccelZ (g), Pressure (psi), Humidity (%)";
```
SD Card Code:

- New Void Setup stuff...

```c
void setup()
{
  // put your setup code here, to run once:
  // initialize the digital LED pin as an output.
  pinMode(led, OUTPUT);

  // initialize serial communication at 9600 bits per second:
  Serial.begin(9600);

  // SD Card setup
  // Make sure that the default chip select pin is set to output, even if you don't use it:
  pinMode(10, OUTPUT);  // To make the SPI (microSD card) interface work Set pin 10 to an Output.
  pinMode(chipSelect, OUTPUT);  // Set the Chip Select pin for the SparkFun SD card to an Output.

  // This function will set up the SD card so we can write to it.
  // Also, the header of the log file will be written to the file.
  SDCardInit();
}
```
**SD Card Code:**

- New Void Loop stuff...

- Delay loop start

- Record the time and print it

- Convert sensor value to character and store in string

```java
void loop()
{
    delay(LOG_INTERVAL);

    // Clear out dataString to start a new record.
    dataString = "";

    // Log the time
    timestamp = millis();
    dataString = String(timestamp);
    Serial.print(timestamp);

    // Read 1st sensor A0 - Temp Sensor
    temp = analogRead(A0);
    tempV = temp*(5.0/1024);
    tempC = (tempV - 0.5)/(0.01);
    Serial.print(',')
    Serial.print(tempC, 2);
    dtostrf(tempC, 6, 2, tempCString);
    dataString = dataString + "," + tempCString;
}```
SD Card Code:

- Convert sensor value to character and store in string

```
// Read 2nd Sensor A1 - Accel X
accelX = analogRead(A1);
accelXV = accelX*(5.0/1024);
accelXG = (accelXV - 1.65)/(0.206);
Serial.print(',,');
Serial.print(accelXG,3);
dtostrf(accelXG, 6, 2, accelXGString);
dataString = dataString +"," + accelXGString;

// Read 3rd Sensor A2 - Accel Y
accelY = analogRead(A2);
accelYV = accelY*(5.0/1024);
accelYG = (accelYV - 1.65)/(0.206);
Serial.print(',,');
Serial.print(accelYG,3);
dtostrf(accelYG, 6, 2, accelYGString);
dataString = dataString +"," + accelYGString;

// Read 4th Sensor A3 - Accel Z
accelZ = analogRead(A3);
accelZV = accelZ*(5.0/1024);
accelZG = (accelZV - 1.65)/(0.206);
Serial.print(',,');
Serial.print(accelZG,3);
dtostrf(accelZG, 6, 2, accelZGString);
dataString = dataString +"," + accelZGString;
```
**SD Card Code:**

- Convert sensor value to character and store in string

- `dataString` now contains all data for all sensors

- Time to write it to the flash
**SD Card Code:**

- Call the write to sd magic

- If successful, blink led

- Loop

```cpp
Serial.println();

// Write to the SD card and if a write to SD is successful
// The function writeDataToSD returns TRUE if a write was succesful
if (writeDataToSD()) {
  // Change the state of the led from OFF->ON, or ON->OFF
  ++ledState;
  // Handle the case where the LED state is now 2 and set
  if (ledState > 1) {
    ledState = 0;
  }
}

// Show a successful write cycle
digitalWrite(led, ledState);
// Change the LED state
```
SD Card Code:

- If you were to upload the code, would it work?
**SD Card Code:**

- If you were to upload the code, would it work?
- No because functions are not defined yet...

```c
pinMode(10, OUTPUT);  //
pinMode(chipSelect, OUTPUT

// This function will set
// Also the header of the
SDCardInit();
}

// Write to the SD card and if
// The function writeDataToSD is
if(writeDataToSD())
{
  // Change the State of the LE
```
SD Card Code:

- Here is the code for SDCardInit();

```java
// This function is called when the system starts or after a power reset to setup
// SDCard

boolean SDCardInit()
{
  Serial.println("Initializing SD card...");

  // see if the card is present and can be initialized:
  if (!SD.begin(chipSelect))
  {
    Serial.println("Card failed, or not present");
    delay(100);
    // Can't do anything more. Try another time....
    return 0;
  }

  // The Card is present so find an unused file name

  Serial.println("searching for an unused file name...");
  // Start with LOG0.CSV and count up until an unused file name is found.
```
SD Card Code:

- Here is the code for `SDCardInit()`;

- But wait there is more!

- We still have the other function

```c
for (long i=0; i < 1000; i++)
{
    sprintf(logFileName, "LOG%d.CSV", i);
    if (!SD.exists(logFileName))
    {
        break;
    }
}
Serial.print("The Log filename is: ");
Serial.println(logFileName);

File dataFile = SD.open(logFileName, FILE_WRITE);
// Write the header including sensor names to the newly opened file
dataString = sensorNames;

if (dataFile)
{
    dataFile.println(dataString);
    dataFile.close();
    Serial.println("SD Card initialized and data written.");
    return(1);
    // Able to write to SD card
}
else
{
    Serial.println("SD card present but unable to write to file");
    sprintf(logFileName, "LOG0.CSV");
    // Clear out the file name
    return(0);
    // Unable to write to SD card
}
```
**SD Card Code:**

- Here is the code for `writeDataToSD`;

- So where does all this code go?

- TABS…

```java
boolean writeDataToSD ()
{
    File dataFile = SD.open(logFileName, FILE_WRITE);

    // if the file is available, write to it:
    if (dataFile)
    {
        dataFile.println(dataString);
        dataFile.close();
        return(1);
    }

    // if the file isn't open, notify that there was an error
    else {
        Serial.println();
        Serial.print("error writing to file: ");
        Serial.println(logFileName);
        dataFile.close();
        delay(100);
        Serial.println("Re - Initializing SD card...");
        return(SDCardInit());
    }
}
```
**SD Card Code:**

- Creating tabs is a great to organize your code

```c
/* Gateway to Space 2014 - Chris Koehnle 
Sensor Code for: 
- Temperature 
- Accelerometers 
- Humidity 
- Pressure 
- 4 LED Visual Display */

// Definitions 
int pressure; 
float pressureV; 
float psi; 

int humidity; 
float humidityV; 
float RH;
```
**SD Card Code:**

- Creating one and press OK
**SD Card Code:**

- Brand new real estate to type code!
SD Card Code:

- Create another one
**SD Card Code:**

- Every tab you create, creates a new *.ino file in the folder for your main tab or sketch you were working
- Must keep all *.ino files in the folder for the code to work properly
- Can only open the main *.ino file directly
**SD Card Code:**

- Now start typing
**SD Card Code:**

- No. Just kidding. Go to the website and download sd +Sensor Code (zip file)
**SD Card Code:**

- Unzip and use directly or open tabs, copy and paste content from one tab to your tab

- Build and check for errors

- Upload and watch LED blink

- Raise hand for help
Part 4 – Arduino Data Storage

A. SHIELD Integration
B. SD Card Code Integration
C. Data Retrieval
D. Accelerometer Calibration
E. Compass
Data Retrieval:

- Press the reset button on your microSD card Shield

- Record some easy to recognize data

- Breath on your humidity sensor twice

- Use solder sucker to give pressure sensor data

- Touch the temp sensor

- Rotate your accelerometer and hold it steady
Data Retrieval:

- Rotate your accelerometer like…

4. X Down  5. X Up
Data Retrieval:

6. Y Down

7. Y UP
Data Retrieval:

8. Z Down

9. Z UP
Data Retrieval:

Record end of file markers

10. Touch your temp sensor for 5 seconds

12. Suck on your pressure sensor for about a 2 seconds

- Pull USB from Uno
Data Retrieval:

- Remove microSD card from Uno and insert into SD card adapter
Data Retrieval:

- Remove microSD card from Uno and insert into SD card adapter
Data Retrieval:

- Insert SD card adapter into your laptop
Data Retrieval:

- Navigate to card and copy last LOGGER file to your desktop

- Open this file with Excel
Data Retrieval:

- Graph all data minus the time stamp
Data Retrieval:

- Graph all data minus the time stamp
Data Retrieval:

- Do you see your data markers?
Data Retrieval:

- Re-plot just your accel data
Data Retrieval:

- How can you use this data?
Data Retrieval:

- So that’s the SD card stuff

- Questions?
Part 4 – Arduino Data Storage

A. SHIELD Integration
B. SD Card Code Integration
C. Data Retrieval
D. Accelerometer Calibration
E. Compass
Acceleration Calibration:

- Adding calibration information to code

- Look at values for X, Y, and Z at -1 and 1 G
Acceleration Calibration:

- Record the differences

- Say X is -0.73 when supposed to be -1.0 AND 1.24 when it should be 1.00

- The difference is roughly 0.24

- Go into the code and subtract it out...
Part 4 – Arduino Data Storage

A. SHIELD Integration
B. SD Card Code Integration
C. Data Retrieval
D. Accelerometer Calibration
E. Compass