Announcements:

- Hardware from last Tuesday, HW 4 and HW 5 needed today
Announcements:

- All proposals are graded

- Feedback at end of class

- Scores were…
  63.2, 64.7, 69.3, 75.7, 77.2, 83.2, 85.0, 85.2

- Grades posted by Tuesday – sorry for the delay

- One minute reports – I am reading them and I appreciate your feedback
Announcements:

Friday Meetings: My Office DLC 270A

- Bring proposals, comments, and grade sheets
- Bring questions and comments
- Bring homework #6 – multiple copies if ordering from different locations
- Bring budget
- Whole team does not need to be there
### Meetings Friday:

<table>
<thead>
<tr>
<th></th>
<th>8:30</th>
<th>9:45</th>
<th>10:30</th>
<th>11:30</th>
<th>12:15</th>
<th>1:00</th>
<th>2:00</th>
<th>2:30</th>
<th>3:15</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>8:30</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
<td>9:45</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
<td>10:30</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td></td>
<td>11:30</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td></td>
<td>12:15</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td></td>
<td>1:00</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td></td>
<td>2:00</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td></td>
<td>2:30</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td></td>
<td>3:15</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- Team 6
- Team 2
- Team 7
- Team 3
- Team 5
- Team 4
- Team 8
- Team 1

---

**Announcements:**
Announcements:

Arduino Final Session Tonight from 6 – 8ish

- **DLC 1B70**
  - All teams must be represented
  - Expect 2 at a minimum but 4 is preferred
  - Pizza will be provided for those who RSVP on today’s attendance sheet
  - Please bring your own drinks
  - By end of session, each team will know how to write sensor data to SD card and then read that data in Excel
  - Will also cover compass
Thursday Night

Arduinos Part 4 (MicroSD cards)

DLC 1B70

Bring all hardware

Food provided for RSVPs

Colorado Space Grant Consortium
Friday...

One on One meetings

Hardware ordering

Chris’s office

Colorado Space Grant Consortium
Next Tuesday...

Requirements and HASP Student presentation

Colorado Space Grant Consortium
Arduino Part 3:
Analog Sensors

Those not interested in Arduinos may leave now

Please return at 10:35 to pick up proposal comments
Congratulations…

- You have completed your first sensor integration
- They get easier now
- Why?

**Potentiometer:**

PLEASE SAVE YOUR SKETCH FILE
Part 2 – Arduino Test Drive

A. LED Visual Display
B. Analog vs. Digital
C. Potentiometer
D. Infrared Sensor
Infrared Sensor:

- Arduino Uno
- Infrared
- Monitor
- PC/Mac
- LEDs
- Pot/Other
Infrared Sensor:

- Let’s replace the potentiometer with a different type of sensor that will detect motion and use our 4 LED to tell us how close an object is to our sensor
Infrared Sensor:

- This is an Infrared Proximity Sensor by Sharp (GP2Y0A21YK)

- Emits and detects IR similar to a RADAR
Infrared Sensor:

- Connect the 3 wire harness to the IR sensor as shown

- Connector is “keyed” so it will only install one way
**Infrared Sensor:**

- Connect the 3 wire harness to the IR sensor as shown

- Connector is “keyed” so it will only install one way

**Key Feature**
**Infrared Sensor:**

- Remove the potentiometer from the Arduino and replace with IR Sensor

- Yellow wire goes to A0, Red to 5V and Black to GND
Infrared Sensor:

- At a distance of 5 cm from the "eyes" the detector will reach its max response voltage of ~3.2 volts
**Infrared Sensor:**

- What do you have to change in the code?

- Just the levels since ~3.2V is the max value

- Using the graph, set levels for 5 cm, 10 cm, 20 cm and 30 cm

- Like the Potentiometer, the IR device is just another analog sensor
**Infrared Sensor:**

- Use the following values for the IR sensor

- Compile and Upload and test!

- Verify LED Visual Display is working by comparing with Serial Monitor

```c
if(sensorVoltage > 0.9) {
    digitalWrite(2, HIGH);
}
if(sensorVoltage > 1.25) {
    digitalWrite(3, HIGH);
}
if(sensorVoltage > 2.25) {
    digitalWrite(4, HIGH);
}
if(sensorVoltage > 3.2) {
    digitalWrite(5, HIGH);
}
delay(100);
```
**Infrared Sensor:**

- Tinker with the distances until everyone is at the same place

**Challenge…**
- Come up with a way to see 0.5 volt increments with your LED Visual Display

PLEASE SAVE YOUR SKETCH FILE
Part 2 – Arduino Test Drive

A. LED Visual Display
B. Analog vs. Digital
C. Potentiometer
D. Infrared Sensor
Part 3 – Arduino Road Trip

A. Temperature Sensor
B. Accelerometers
C. Humidity Sensor
D. Pressure Sensor
E. Compass
Part 3 – Arduino Road Trip

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

- Both the potentiometer and the IR sensors are analog sensors

- Almost all analog sensors work and are connected in the same way

- Let’s use hook up one more

- To do so will require another breadboard, with nicer features
Temperature Sensor:

Temperature sensor is the TMP36 - Temperature Sensor
Temperature Sensor:

- This breadboard has power and ground rails
- Individual points on rails (rows) are connected
- One rail, and its points, are independent of other rails
Temperature Sensor:

- Also has numbers and letters to coordinate builds
Temperature Sensor:

- Place in new breadboard **exactly** as shown
Temperature Sensor:

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

Note orientation of sensor package

PIN 1, +Vs; PIN 2, Vout; PIN 3, GND
Temperature Sensor:

- Connect 5V from Arduino to “+” power rail
- Connect GND from Arduino to “-” rail
Temperature Sensor:

- Connect 5V from Arduino to “+” power rail

- Connect GND from Arduino to “-” rail
Temperature Sensor:

- Now all pins on “+” rail are connected to 5 Volts
- All “-” are connected to GND
Temperature Sensor:

- Connect 5V and GND from rails to sensor

5V to A3

GND to A1
Temperature Sensor:

- Connect A0 from Arduino to A2 on Breadboard.
Temperature Sensor:

- What do we need to change in the Sketch?
Temperature Sensor:

- Nothing! But the Temp Sensor Voltage commands all LEDs off

- Build and Upload the code and look at serial monitor

- Should see ~0.73 Volts, which is well below our first LED turn on value of 0.9 Volts in the current IR sensor Sketch
Temperature Sensor:

- To understand and set the proper range, let’s consult the data sheet for the sensor

- **10 mV/C (0.010V/C)**

The TMP35 is functionally compatible with the LM35/LM45 and provides a 250 mV output at 25°C. The TMP35 reads temperatures from 10°C to 125°C. The TMP36 is specified from −40°C to +125°C, provides a 750 mV output at 25°C, and operates to 125°C from a single 2.7 V supply. The TMP36 is functionally compatible with the LM50. Both the TMP35 and TMP36 have an output scale factor of 10 mV/°C.
Temperature Sensor:

- Data sheet also says there is an offset

- For TMP36, Offset = 0.5 Volts
Temperature Sensor:

- So to understand the data, we need to do some math to convert voltage to °C and set our ranges

\[
TempC = \frac{(\text{sensorVoltage} - 0.5)}{0.01}
\]

Using what we are seeing from our serial monitor, 0.73 Volts, we would get...

\[
TempC = \frac{(0.73 - 0.5)}{0.01} = 23 \degree C
\]
Temperature Sensor:

- Change the Sketch to do the math and change LED turn on ranges to temperatures of 21, 22, 23, and 24 C.

- Will need a new variable (float)
Temperature Sensor:

- New variable, tempC (float)

- Add your math

- Modify your print statements
Temperature Sensor:

- Adjust your range settings for the LEDs
- Build and Upload
- Test by touching your temp sensor
- Challenge while you wait…

Convert to Fahrenheit

PLEASE SAVE YOUR SKETCH FILE

digitalWrite(2, LOW);
digitalWrite(3, LOW);
digitalWrite(4, LOW);
digitalWrite(5, LOW);

if(tempC > 21.0) {
digitalWrite(2, HIGH);
}
if(tempC > 22.0) {
digitalWrite(3, HIGH);
}
if(tempC > 23.0) {
digitalWrite(4, HIGH);
}
if(tempC > 24.0) {
digitalWrite(5, HIGH);
}
delay(100);
Part 3 – Arduino Road Trip

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

- Arduino Uno
- Monitor
- PC/Mac
- LEDs
- Accels
- Temp
Accelerometers:

- Detect forces acting upon the sensor/payload
- MMA7361 3 axis accelerometer
- 3.3 V input & Analog Output

Low Range ±1.5 G or High Range ±6 G
Accelerometers:

- How it works…

Figure 3. Simplified Transducer Physical Model
**Accelerometers:**

- Place Accelerometer as shown

- Pin ST at F30

- Pin Vcc at F22
Accelerometers:

- Connect 3.3 V from Arduino to new “+” power rail

- Connect 3.3 V to VCC on accelerometer
Accelerometers:

- Connect GND from GND rail used for Temp Sensor as new GND rail

- Connect GND to GND on accelerometer
Accelerometers:

- 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.
Accelerometers:

- Connect Vcc to GSEL
- Sets range to ±6 G
Accelerometers:

- 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
Accelerometers:

- How do we modify the sketch?

- 3 new Analog lines (inputs)

- 3 to 9 new variables

- Changes to print statements
Accelerometers:

- Add new variables

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

- Add new read statements
- Add new print statements
- Compile and Upload
- Launch serial monitor

```c
void loop() {
  // put your main code here, to run repeat
  sensorValue = analogRead(A0);
  sensorVoltage = sensorValue*(5.0/1024);
  tempC = (sensorVoltage - 0.5)/(0.01);

  accelX = analogRead(A1);
  accelY = analogRead(A2);
  accelZ = analogRead(A3);

  Serial.print("accel X ");
  Serial.print(accelX);
  Serial.print(" \t accel Y ");
  Serial.print(accelY);
  Serial.print(" \t accel Z ");
  Serial.print(accelZ);
  //Serial.print(sensorValue);
  //Serial.print(" \t Voltage ");
  //Serial.print(sensorVoltage);
  Serial.print(" \t Temp C ");
  Serial.println(tempC);
}
```
Accelerometers:

- Carefully rotate your breadboard to see different accel values
Accelerometers:

- Carefully rotate your breadboard to see different accel values
Accelerometers:

- Let’s convert decimal value to engineering units

<table>
<thead>
<tr>
<th>Output Signal</th>
<th>V_{OFF}</th>
<th>1.485</th>
<th>1.65</th>
<th>1.815</th>
<th>V</th>
</tr>
</thead>
<tbody>
<tr>
<td>Zero-g (T_{A} = 25°C, V_{DD} = 3.3 V)</td>
<td>V_{0g}</td>
<td>1.65</td>
<td>1.815</td>
<td>V</td>
<td></td>
</tr>
<tr>
<td>Zero-g (4)</td>
<td>V_{0g}</td>
<td>1.65</td>
<td>1.815</td>
<td>V</td>
<td></td>
</tr>
<tr>
<td>Sensitivity (T_{A} = 25°C, V_{DD} = 3.3 V)</td>
<td>S_{6g}</td>
<td>190.6</td>
<td>206</td>
<td>221.5</td>
<td>mV/°g</td>
</tr>
<tr>
<td>1.5g</td>
<td>S_{6g}</td>
<td>190.6</td>
<td>206</td>
<td>221.5</td>
<td>mV/°g</td>
</tr>
<tr>
<td>6g</td>
<td>S_{1g}</td>
<td>0.807</td>
<td>0.807</td>
<td>0.807</td>
<td>mV/°g</td>
</tr>
<tr>
<td>Sensitivity</td>
<td>S_{1g}</td>
<td>0.807</td>
<td>0.807</td>
<td>0.807</td>
<td>mV/°g</td>
</tr>
<tr>
<td>Bandwidth Response</td>
<td>f_{3dBY}</td>
<td>400</td>
<td>—</td>
<td>—</td>
<td>Hz</td>
</tr>
<tr>
<td>XY</td>
<td>f_{3dBZ}</td>
<td>300</td>
<td>—</td>
<td>—</td>
<td>Hz</td>
</tr>
<tr>
<td>Z</td>
<td>Z_{O}</td>
<td>32</td>
<td>—</td>
<td>—</td>
<td>kΩ</td>
</tr>
<tr>
<td>Output Impedance</td>
<td>Q_{detect}</td>
<td>-0.4</td>
<td>0</td>
<td>+0.4</td>
<td>g</td>
</tr>
<tr>
<td>0g-Detect</td>
<td>Q_{detect}</td>
<td>-0.4</td>
<td>0</td>
<td>+0.4</td>
<td>g</td>
</tr>
</tbody>
</table>
Accelerometers:

- Convert to Voltage = \( \text{accelX} \times \frac{5.0}{1024} \)

- Subtract offset and convert to Gs
  \( \text{Gs} = \frac{\text{voltage} - 1.65}{0.206} \)
Accelerometers:

- So modify the sketch
**Accelerometers:**

- Add some new variables

```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;
```
Accelerometers:

- Do the math

```c
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);
}```
Accelerometers:

- Modify the print statements
Accelerometers:

- Compile and Upload
- Start serial monitor

- While waiting…

Challenge: Use your LED Visual Display with your accelerometer

PLEASE SAVE YOUR SKETCH FILE
Part 3 – Arduino Road Trip

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

- Humidity sensor (or the Darth Vader Sensor)

- It measures moisture in the air, which is great for balloon flights (condensation failures)
**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:**

- Add your humidity sensor as shown

5V to 5 V
power Rail

GND to
GND Rail

OUT to A5
on the
Arduino
Humidity Sensor:

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

- Comment out all the compass code by using `//`

- Add 3 new variables

```cpp
#include <Wire.h>

// I2C Address for Compass Module HMC
// int compassAddress = 0x42 >> 1;
// shift the address 1 bit right, the
// most significant bits for the address

// int currentVector = 0;

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

- You will also need to delete the Compass Magic tab or comment all code in that tab

- With that tab selected, click down arrow and select Delete
Humidity Sensor:

- Comment out all the compass code in void setup and void loop by using //

```c
void setup() {
  // put your setup code here, to run once:
  // Start of I2C
  // join i2c bus (address optional for master)
  Wire.begin();

  Serial.begin(9600);
  // pinMode(led, OUTPUT);
  pinMode(2, OUTPUT);
  pinMode(3, OUTPUT);
  pinMode(4, OUTPUT);
  pinMode(5, OUTPUT);
}

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

  // Call the Function getVector and save the results
  // currentVector = getVector();
  // Print the compass reading to the serial port
```
Humidity Sensor:

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

- We know Vout and Vsupply so using algebra

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

- Here’s the algebra and the equation to code

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

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

\[
\text{sensorRH} = \left( \frac{V_{OUT}}{5.0 \text{ V}} - 0.16 \right) \frac{0.0062}{0.0062}
\]
Humidity Sensor:

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

- Serial print the RH to the monitor

```cpp
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("\taccel 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

```c
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