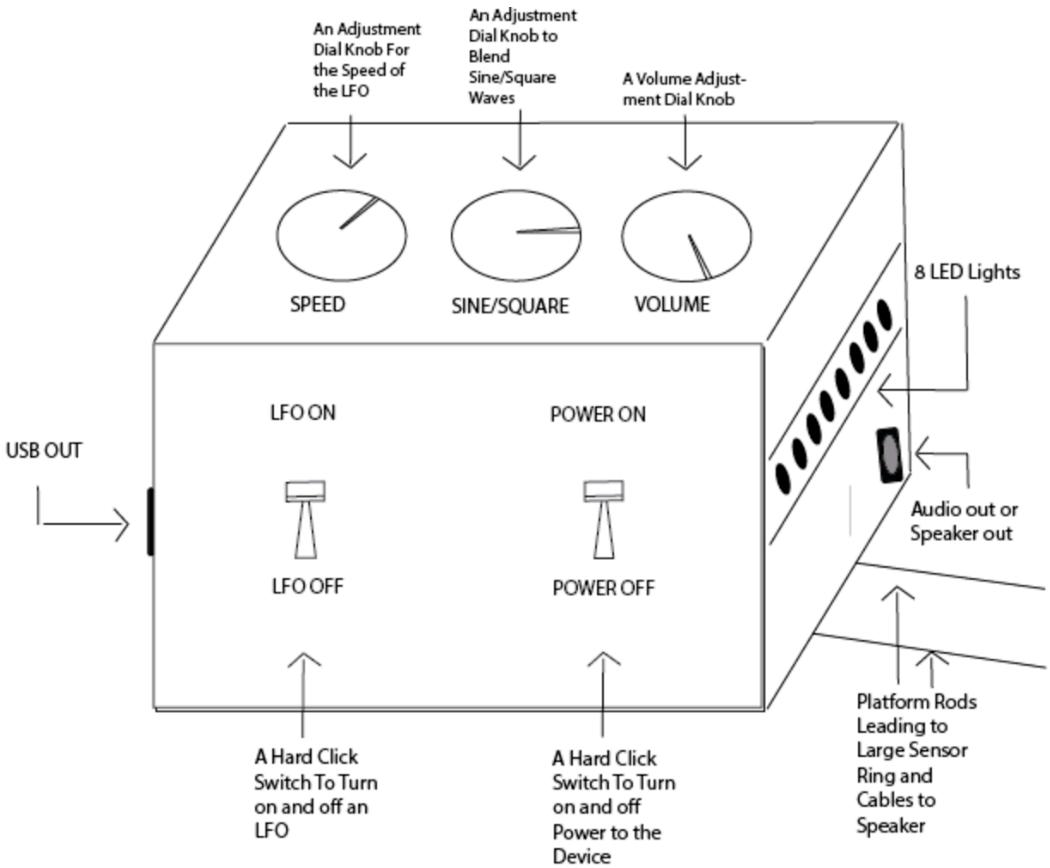
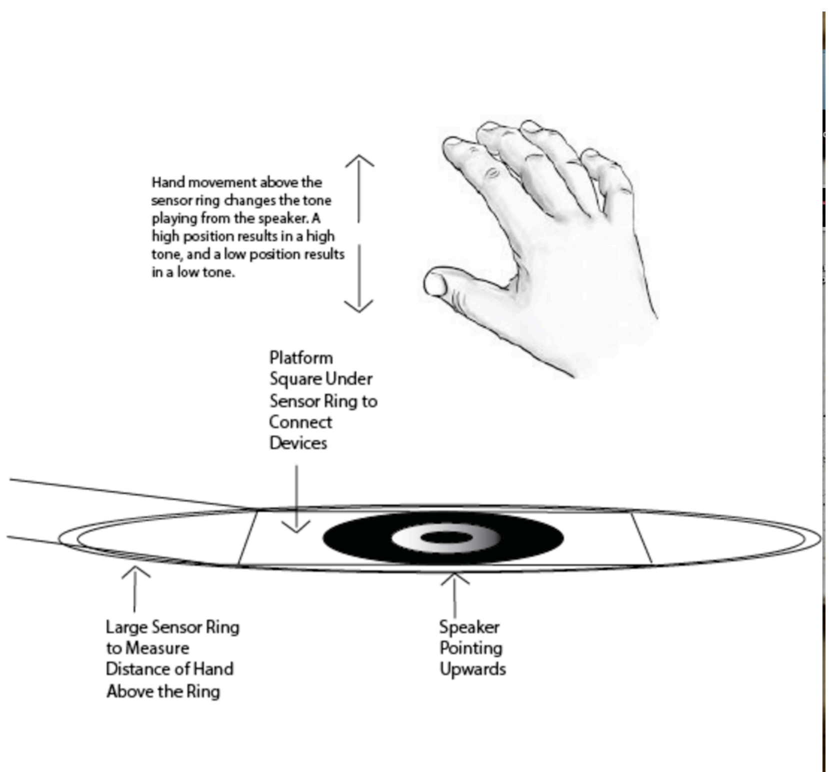
Ben Leger
Technology Design Studio
Professor Andrew Smyk
December 1st 2016.
Statement of Breaththroughs, Stumbles and Victories.

For my technology design studio project, I wanted to tackle an arduino-based prototype that included sensors, and sound manipulation. The goal was to create a sensor based frequency generator, allowing a user to manipulate outputted sound with their body as opposed to dials and knobs. A Theremin is a music instrument that uses hand movement to manipulate frequency sounds, so essentially an arduino based Theremin was what I created as my prototype.

Through the planning process, I was very optimistic with the direction I wanted to go. I considered different sensors and how they could be used in application, but it was difficult to put a particular sensor into my plan having never used the sensors for sound manipulation. With this in mind, I decided to build 2 prototypes: one with a light sensor, that would manipulate sound based on how much light was in the room, and in a separate project using the HC-SR04 sensor, that measured an objects distance from it and would manipulate sound based on how close the object was. I was also hoping I would be able to allow the user to have multiple controls for sound manipulation in a low filter oscillator effect, and a mix knob to change between sine and square waves. These ideas were beyond the basic goal of the device, which was to just have a user manipulate audio sounds, but I was unsure how complex the total build process was going to be, and wanted to leave my options open to enhance the user experience by including these additional user features. Below is my initial design sketch:


An Adjustment Dial Knob For the Speed of the LFO
An Adjustment Dial Knob to Blend Sine/Square Waves
A Volume Adjustment Dial Knob
SPEED
SINE/SQUARE
VOLUME
8 LED Lights
USB OUT
LFO ON
LFO OFF
POWER ON
POWER OFF
Audio out or Speaker out
A Hard Click Switch To Turn on and off an LFO
A Hard Click Switch To Turn on and off Power to the Device
Platform Rods Leading to Large Sensor Ring and Cables to Speaker


Hand movement above the sensor ring changes the tone playing from the speaker. A high position results in a high tone, and a low position results in a low tone.
Platform Square Under Sensor Ring to Connect Devices
Large Sensor Ring to Measure Distance of Hand Above the Ring
Speaker Pointing Upwards

After creating my prototype, I found both devices to have their advantages, but my stumbles were consistent through both prototypes in manipulating the code to smoothen out the transition between frequencies. I found it easier to do this with the distance sensor to create a step-by-step pitch increase by way of ‘if statements’ that included 3 set frequencies of defined sound variables at different “greater than” distance points. Successfully writing the ‘if statements’ in this fashion was definitely a victory, as it was a tedious process getting the distances at comfortable ranges for application, and took a fair amount of time via trial and error. This however was very rewarding once I got the distances in a comfortable placement for the user. After generating my tones successfully, it was fairly straightforward to create a loop of the 3 sounds, as I just applied a short delay variable to each tone. I also was able to include LED lights that measured in HIGH and LOW in the same statement, creating a joined auditory and visual interactive experience. This was definitely a breakthrough as an interactive experience for both audio and visual was one of my biggest goals for the project. I did not, however, include the additional features of sin/square wave mixing and a low filter oscillator. I did some research on it and found that it was a project in itself to create the effects, and I would risk serious volume and stability problems by sending additional manipulations to the buzzer (speaker). This was frustrating, but I did find some success in attempting the low filter oscillator, which was a breakthrough for a future project; I was able to open and close the send signal, but I had to copy and paste some code from an external library, and admittedly could not make much sense of it and didn’t feel comfortable including it in my own project. I plan to revisit the idea at a later date. Below is a screenshot of the code I ended up using for this device.

```
#define trigPin 7
#define echoPin 6
#define led 9
#define led3 11
#define led5 13
#define buzzer 3
//#define buzzer2 4

int sound = 261;
int sound2 = 501;
int sound3 = 900;
int sound4 = 10;


void setup() {
 Serial.begin (9600);
 pinMode(trigPin, OUTPUT);
 pinMode(echoPin, INPUT);
 pinMode(led, OUTPUT);
 pinMode(led3, OUTPUT);
 pinMode(led5, OUTPUT);
 pinMode(buzzer, OUTPUT);

}


void loop() {
 long duration, distance;
 digitalWrite(trigPin, LOW);
 delayMicroseconds(2);
 digitalWrite(trigPin, HIGH);
 delayMicroseconds(10);

 duration = pulseIn(echoPin, HIGH);
 distance = (duration/2) / 29.1;
 Serial.print(distance);
 Serial.println(" cm");

 if (distance < 45) {
   digitalWrite(led,HIGH);
   digitalWrite(led3,LOW);
   digitalWrite(led5,LOW);
   delay(50);
   tone(buzzer,sound);

 }
 if (distance < 32) {
   digitalWrite(led,LOW);
   digitalWrite(led3,HIGH);
   digitalWrite(led5,LOW);
   delay(50);
   tone(buzzer,sound2);
 }
  if (distance < 15) {
   digitalWrite(led,LOW);
   digitalWrite(led3,LOW);
   digitalWrite(led5,HIGH);
   delay(50);
   tone(buzzer,sound3);
 }
```

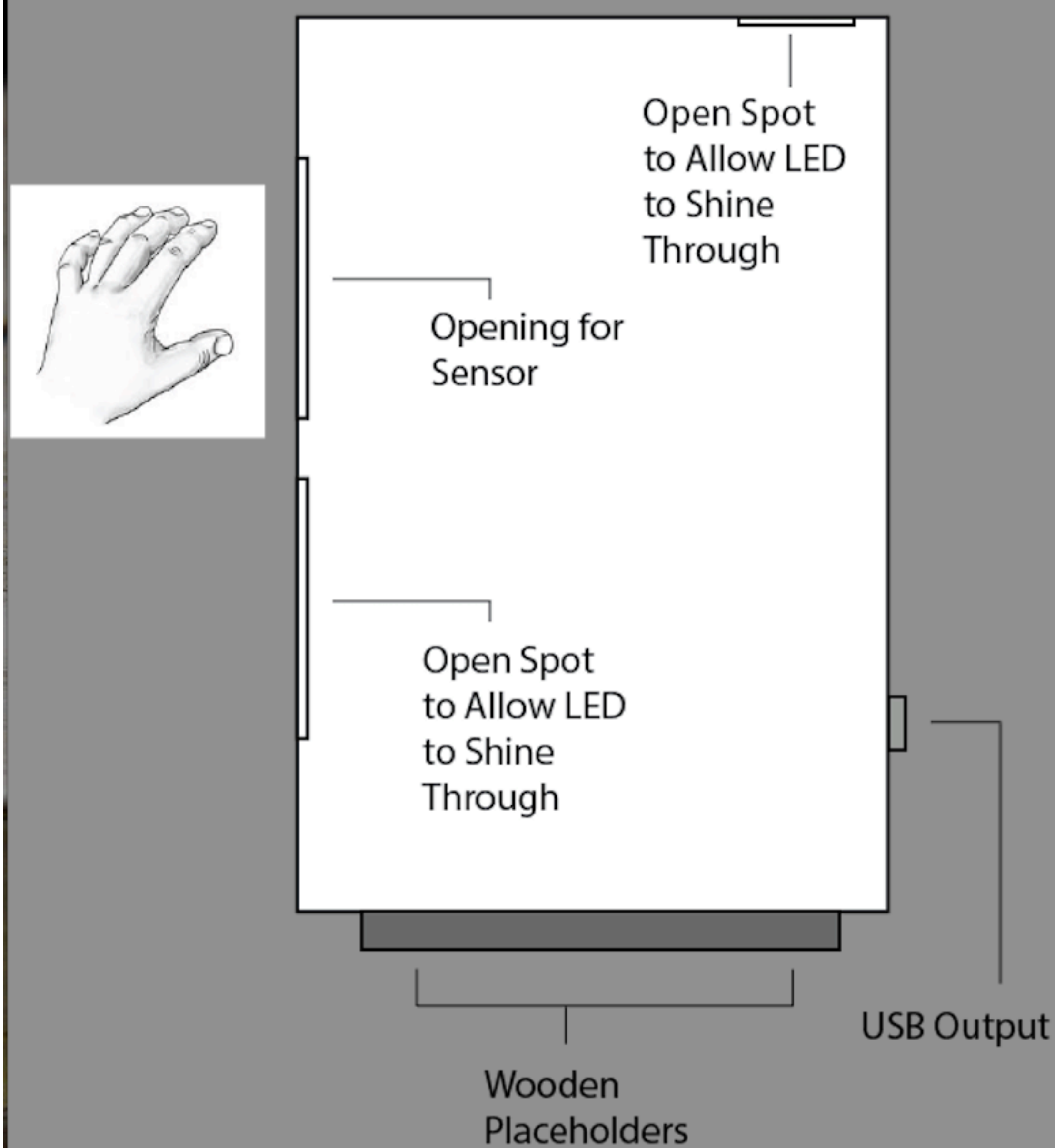
For the light Theremin, I stumbled through the code setting up the sensor to measure the distance of my hand from it (which would cover light and manipulate the frequency send settings), but once I was able to measure the light input, incorporating the sound playing aspect was relatively painless. The light sensor however was difficult to set up on the hardware side and took more project time than was ideal. The stumble occurred until I discovered halfway through testing that the sensors light intensity had to be generated on the loading process; in other words, I had to set the amount of light in its current state on load, in accordance with the sensor in order to create a measurable gage for the sound manipulation. This was a tedious process. However, The nice thing about this stumble was, there was a breakthrough in discovering the problem, and a victory in solving it. Once I found a comfortable amount of light to calibrate the device to on load, I got very good results. Below is a screenshot of the code I ended up using for this device:

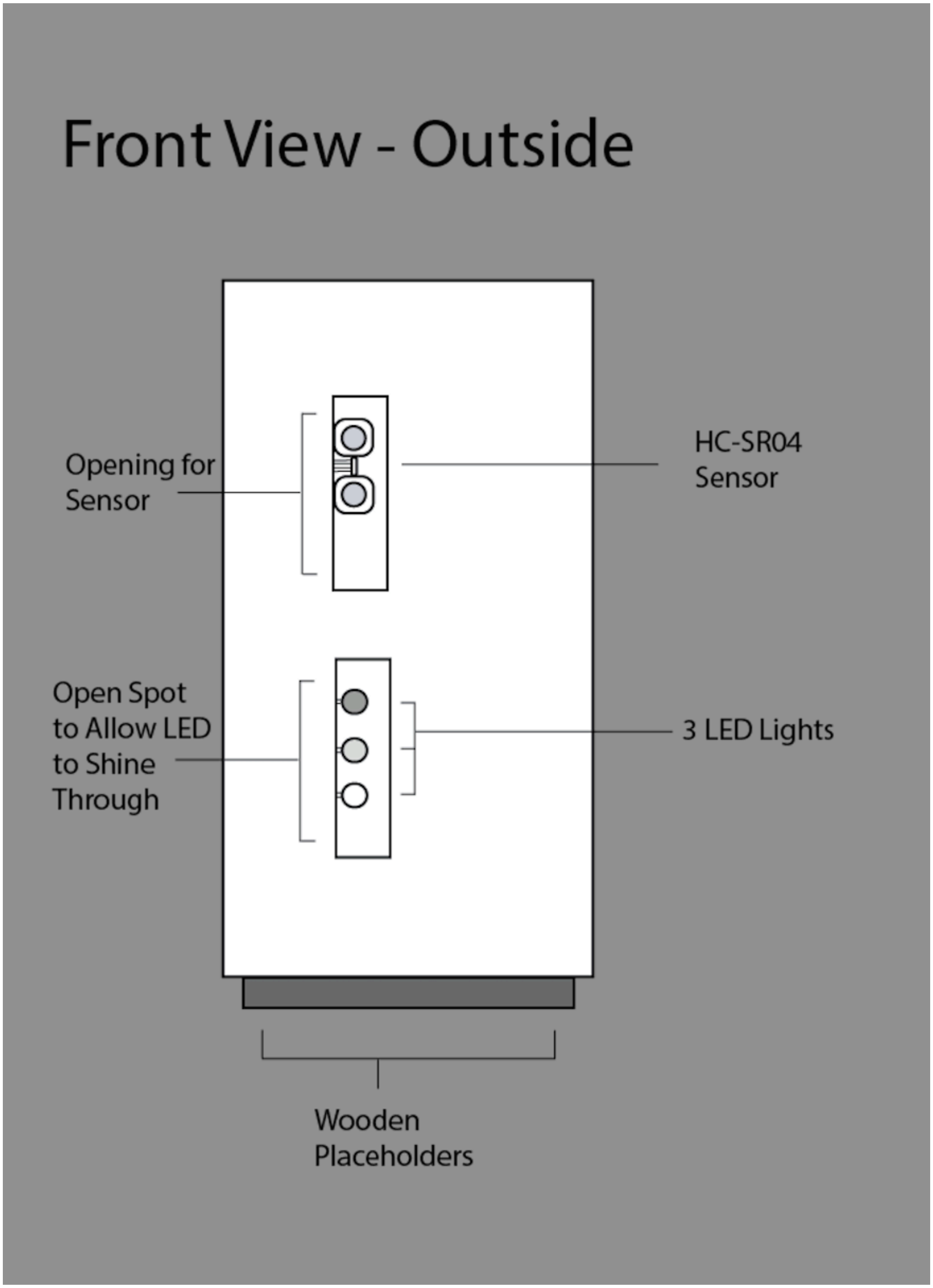
```
int sensorValue;
int sensorLow = 23;
int sensorHigh = 0;
const int ledPin = 13;

void setup() {
  // Make the LED pin an output and turn it on
  pinMode(ledPin, OUTPUT);
  digitalWrite(ledPin, HIGH);


  while (millis() < 5000) {
    sensorValue = analogRead(A0);
    if (sensorValue > sensorHigh) {
      sensorHigh = sensorValue;
    }

    if (sensorValue < sensorLow) {
      sensorLow = sensorValue;
    }
  }
  digitalWrite(ledPin, LOW);
}

void loop() {

  sensorValue = analogRead(A0);

  int pitch = map(sensorValue, sensorLow, sensorHigh, 50, 4000);

  // play the tone on pin 8
  tone(8, pitch, 20);

  // wait for 100ms
  delay(100);
}
```

It is also worth noting that I did at one point try to bring both devises together. In the end, I believe the major stumble was placing devices far enough apart to not have them affect each other, and figuring out a way to arrange separate speakers on the single arduino board. I assumed this would be easier than it ended up being, and didn't really find a solution to creating a single filed and hardware device. Ultimately, I could only generate one buzzer speaker at a time and not trigger them individually, which compromised sound clarity and quality, so I decided to leave them separate to allow for more specific usage and manipulation down the road.

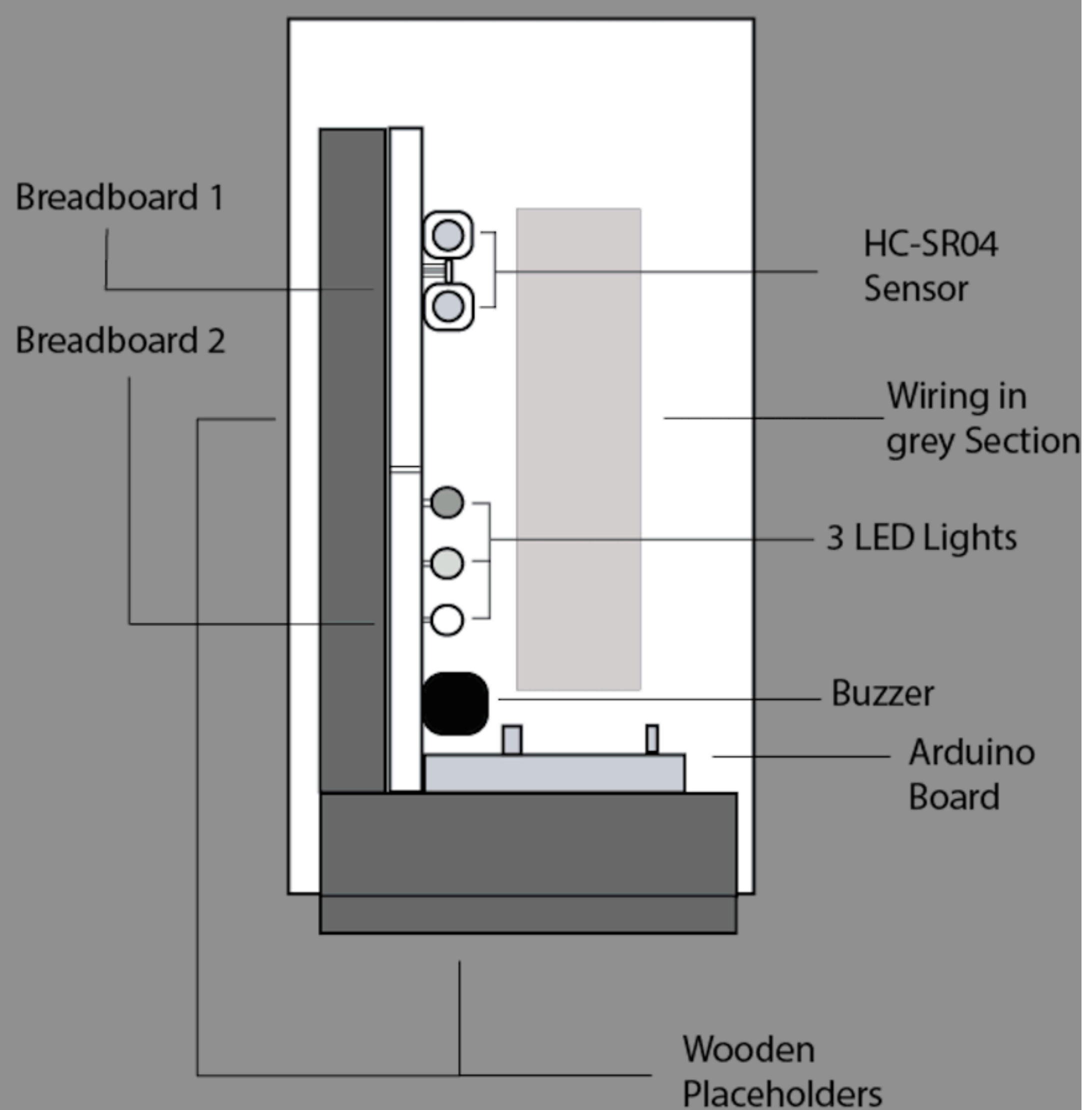
After completing the wiring and assembly of the devices, it was time to give some casing and styling to them. Casing the device would allow for it to be protected from wires being unplugged or any other damage to the device. To do this I had to consider a new design plan, as my initial sketches looked little like my prototype device. This was a stumble moment, so I had to consider a new casing approach. I decided to cover only the HC-SR04 sensor prototype, and leave the light uncovered, as I found when I started to cover the device, the sound was greatly compromised. If I had taken some more time on it, I might have been able to come up with a better design, but for the purpose of this study, I assumed it better to leave it uncovered and place it on top of the HC-SR04 sensor device. Ultimately, the two together looked very sci-fi, and I kind of liked that anyways! Below are some rough sketches of the then new design idea of the HC-SR04 sensor casing: the light Theremin is designed to sit on top.

Side View

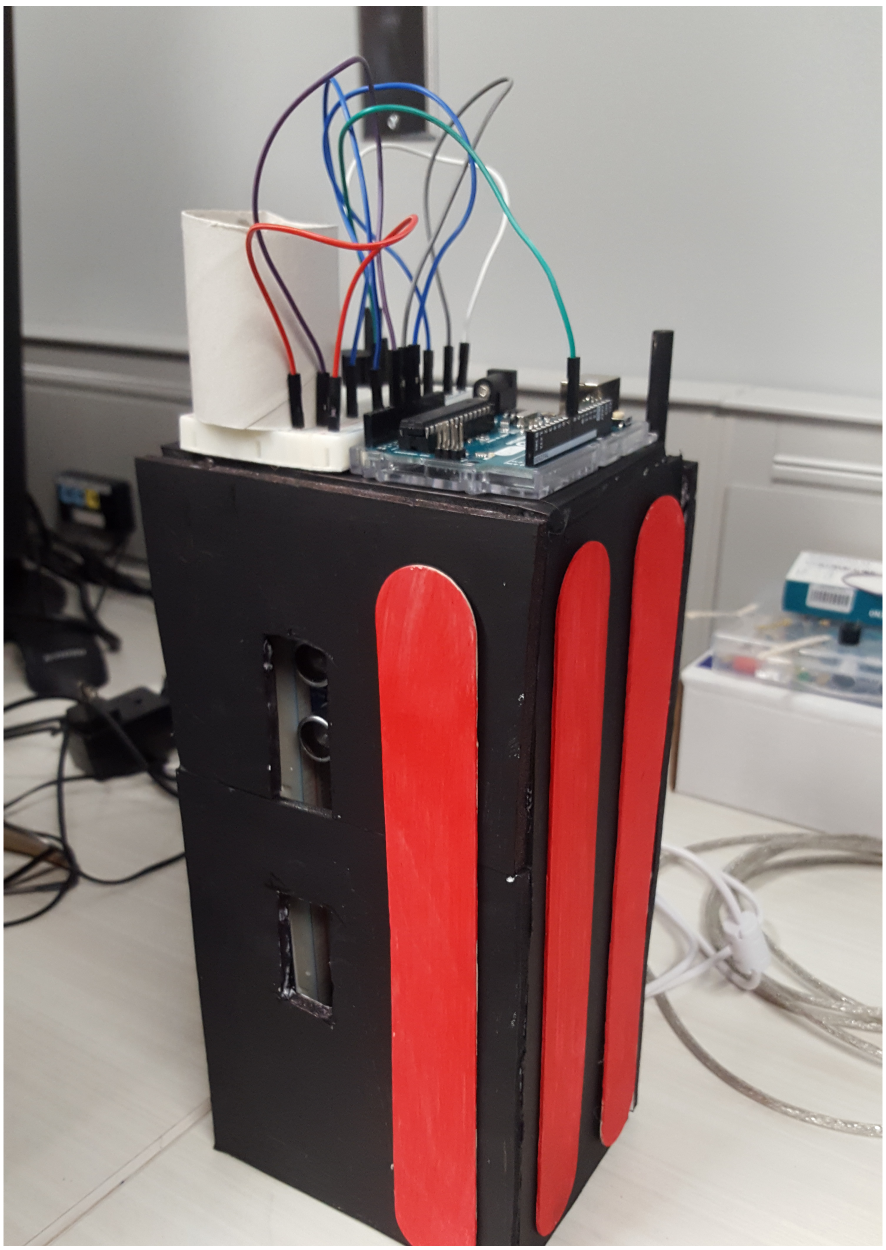

Open Spot
to Allow LED
to Shine
Through
Opening for
Sensor
Open Spot
to Allow LED
to Shine
Through
USB Output
Wooden
Placeholders

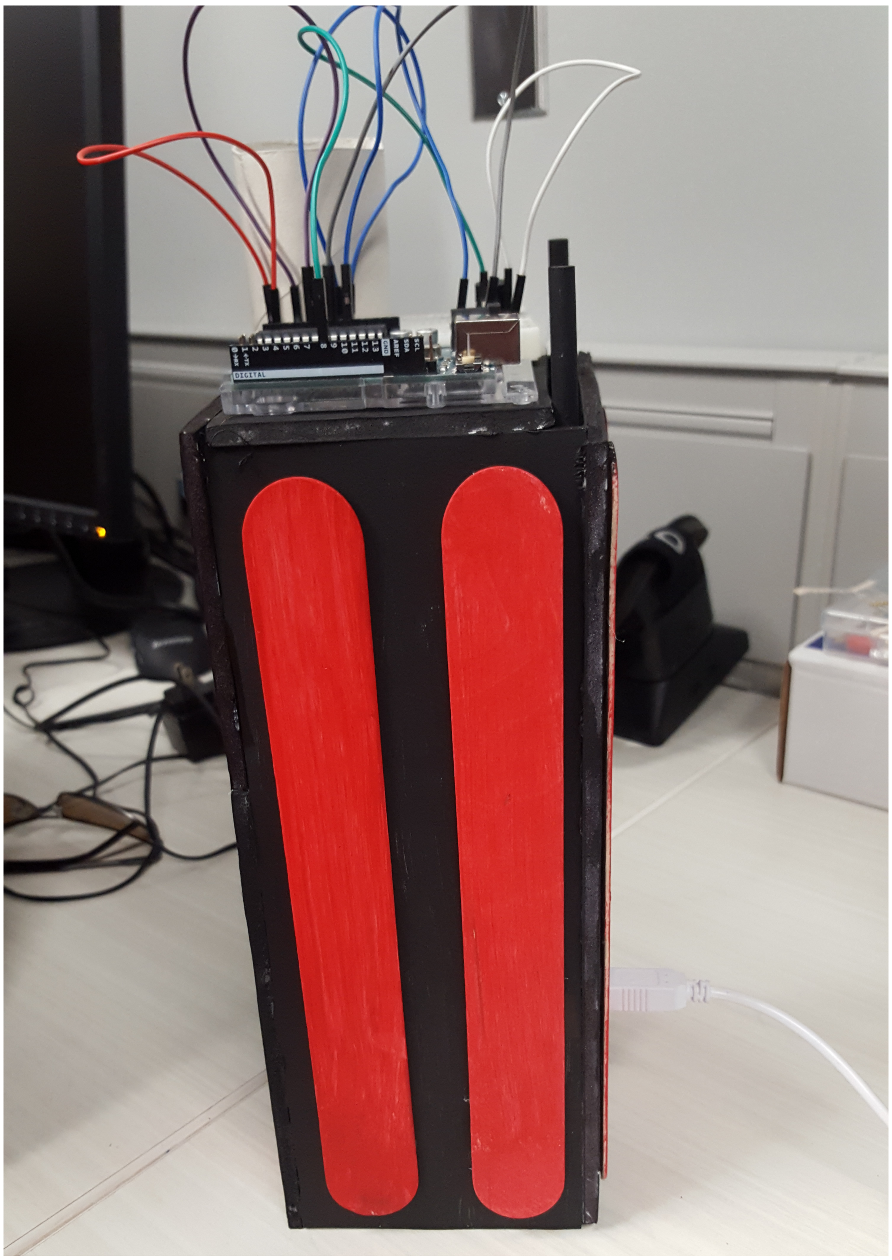

Front View - Outside
Opening for Sensor
HC-SR04 Sensor
Open Spot to Allow LED to Shine Through
3 LED Lights
Wooden Placeholders

Front View - Inside Hardware Placement



My biggest victory for this assignment was successfully using two different kinds of sensors, and learning how to use them in their proper applications. My biggest stumbles were coding, as I am more comfortable using hardware than I am software. I found the purpose of this study was to go through the process of hills and valleys that come with designing a device. Overall I had many successes on both sides of the spectrum, and recognize that chasing a design requires the builder to problem solve, and compromise initial ideas. Below are some pictures of my final design, and attached in external files are videos of the devices in use.




DIGITAL