Technology Studio Final Project

Electronic Arm Wrestling Ring
Tutorial

Written By:

Mitch Patterson
Borys Petriczko

Overview

The objective of our project is to build a fully interactive arm-wrestling station that can be used by two competitors at a time. We hope to emulate each experience as if there is a crowd cheering them on in the background, and once the maximum threshold is hit, the crowd will reach maximum volume and a winner will be appointed.

How it works

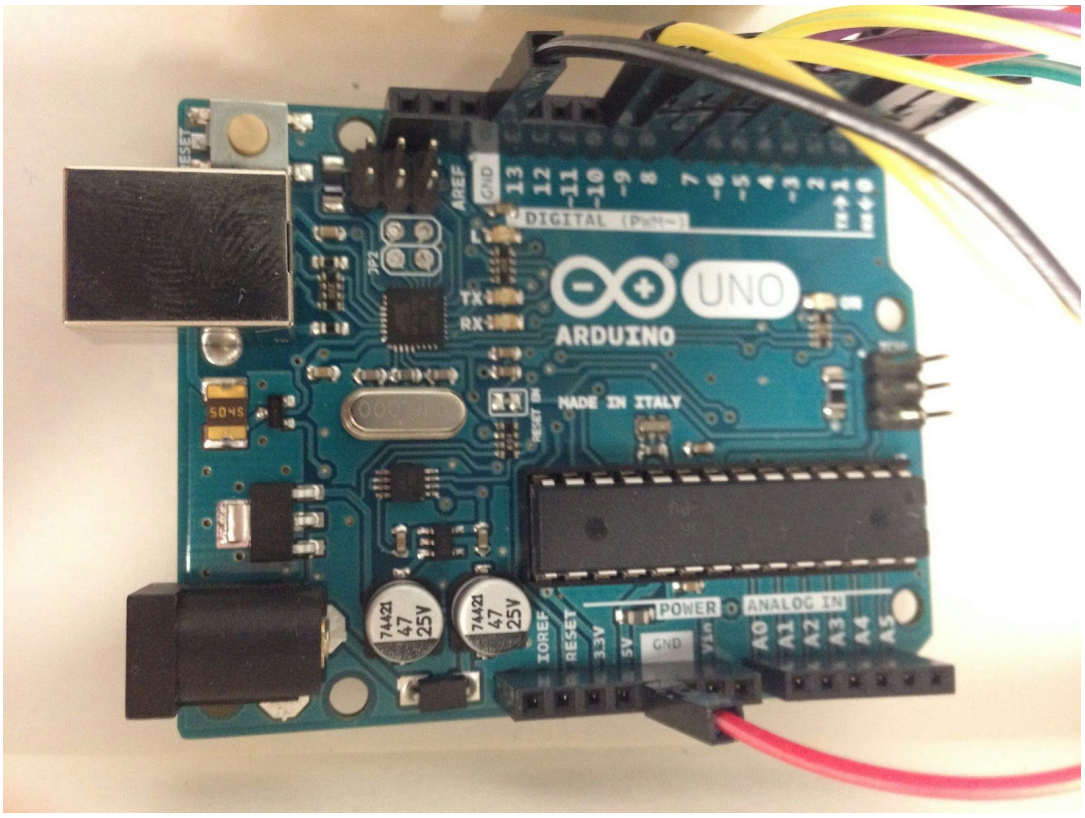
Competitors will begin by each placing their right (or left) elbow in the appropriately labelled area. As the competitors arm wrestle, ultrasonic sensors that are strategically placed within the constraints of the arm-wrestling station will measure the distance each player's hand is the the sensor. As a hand gets closer to a sensor, LED bulbs will also light up as the competitor's hand gets closer to the sensors. Additionally, an LED screen was installed to relay a message to the user about how they were doing in the competition.

Required Materials (for full board):

- 2 Arduino Boards (Uno)
- 2-4 HC-SR04 Ultrasonic Sensors
- 1 Potentiometer
- 6-8 LED Bulbs
- Styrofoam/Cardboard Platform 18” x 28” - rough estimate)

Step 1: Attaching the HC-SR04 Ultrasonic Sensor to the Ardunio Board

Since all circuitry in the project reacts differently based on position of a competitor’s hand from the sensor, it makes sense to attach the sensor first.

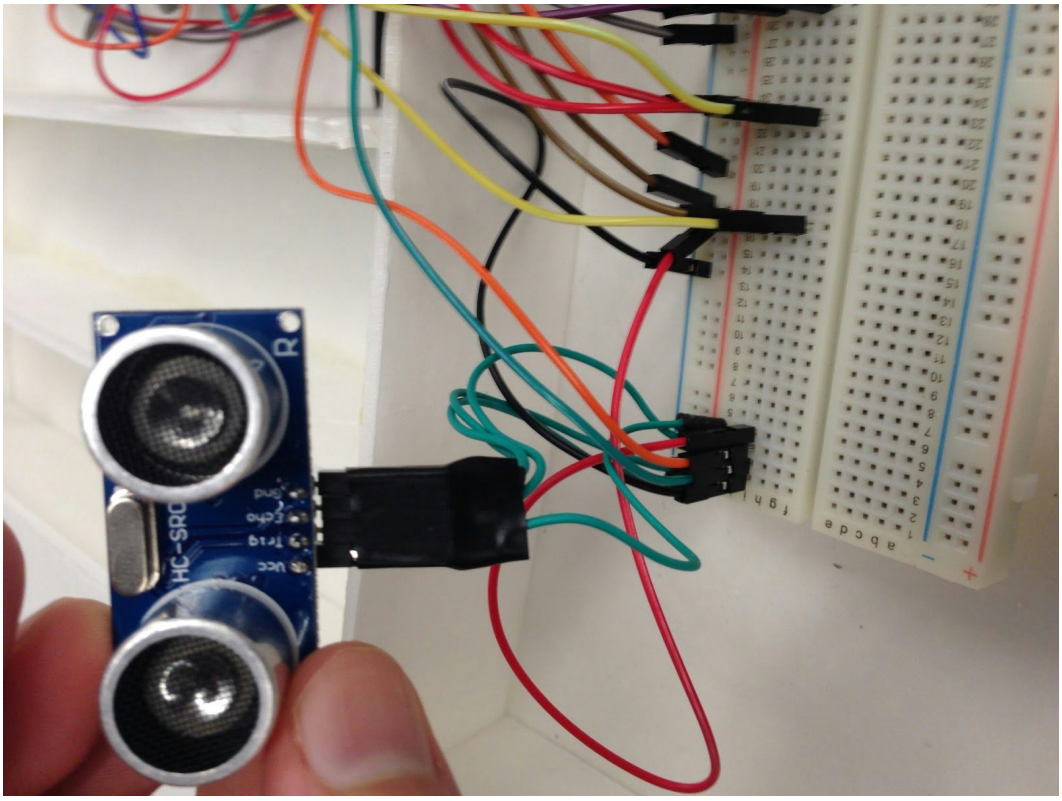


The first step is to take attach wires in two digital inputs in the arduino board, for our example we used input 2 & 3. These will then be attached to the trigPin & the echoPin on the sensor. Below you can see the arduino code we used to define the variables & setup up the void loop.

ArduinoArmWrestle

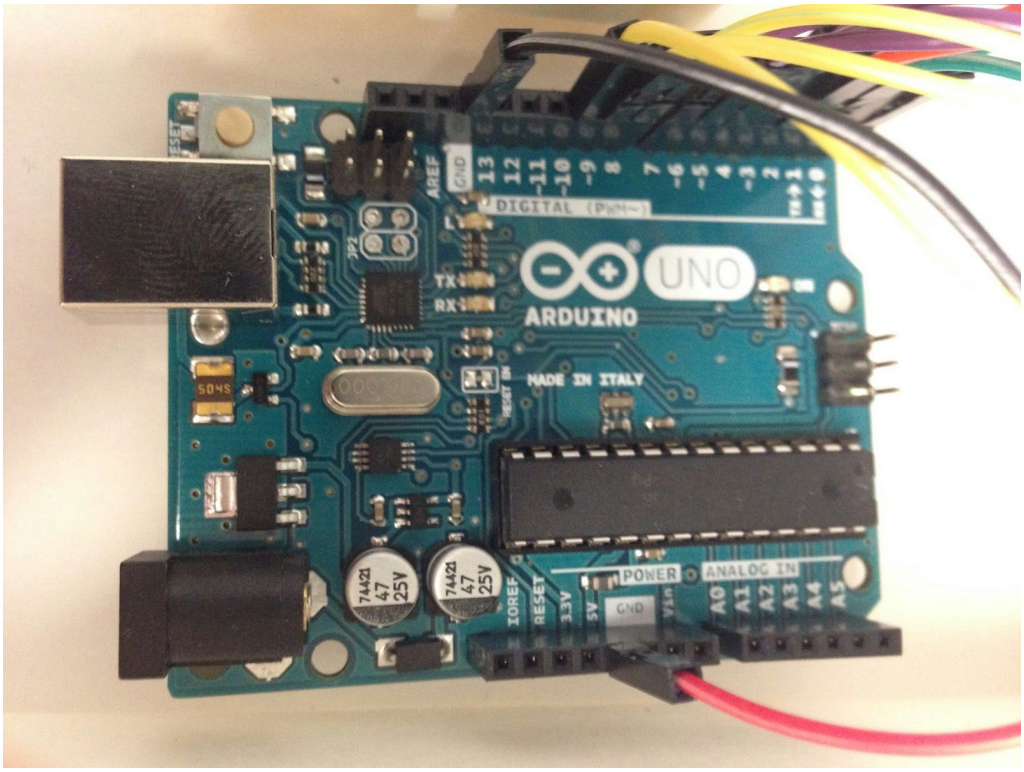
```
#define trigPin 3
#define echoPin 2
#define led 8
#define led2 9
#define led3 10

#include <LiquidCrystal.h> // includes the LiquidCrystal Library
LiquidCrystal lcd(0, 1, 4, 5, 6, 7); // Creates an LC object. Parameters: (rs, enable, d4, d5, d6, d7)

void setup() {
// Serial.begin (9600);
 pinMode(trigPin, OUTPUT);
 pinMode(echoPin, INPUT);
 pinMode(led, OUTPUT);
 pinMode(led2, OUTPUT);
 pinMode(led3, OUTPUT);
 lcd.begin(16,2); // Initializes the interface to the LCD screen, and specifies the dimensions (width and height) of the display

}
```

We then attached the trigPin & echoPin to the approriately labeled spaces of the sensor. Be sure to attach the power source & ground as well in order for the sensor to operate.



Step 2: Adding the LED bulbs to the circuit

Our next step was to add three different coloured bulbs to the circuit that would light up & turn off based on a user's proximity to the sensor. Adding the bulbs to the circuit is very simple. For simplicity, we added 3 bulbs to the Ardunio Board at inputs 8, 9, 10. (You can add as many bulbs as you want, we just used 3 in our example)



As you can see below, we defined the led bulbs at 8, 9, 10 & added the them to the void setup.

ArduinoArmWrestle

```
#define trigPin 3
#define echoPin 2
#define led 8
#define led2 9
#define led3 10

#include <LiquidCrystal.h> // includes the LiquidCrystal Library
LiquidCrystal lcd(0, 1, 4, 5, 6, 7); // Creates an LC object. Parameters: (rs, enable, d4, d5, d6, d7)

void setup() {
// Serial.begin (9600);
 pinMode(trigPin, OUTPUT);
 pinMode(echoPin, INPUT);
 pinMode(led, OUTPUT);
 pinMode(led2, OUTPUT);
 pinMode(led3, OUTPUT);
 lcd.begin(16,2); // Initializes the interface to the LCD screen, and specifies the dimensions (width and height) of the display

}
```

The bulbs then work by attached the wire from the Arduino input to the positive end of the bulb & making sure you attach a ground to complete the circuit.

Step 3: Coding the bulbs to react to the sensor

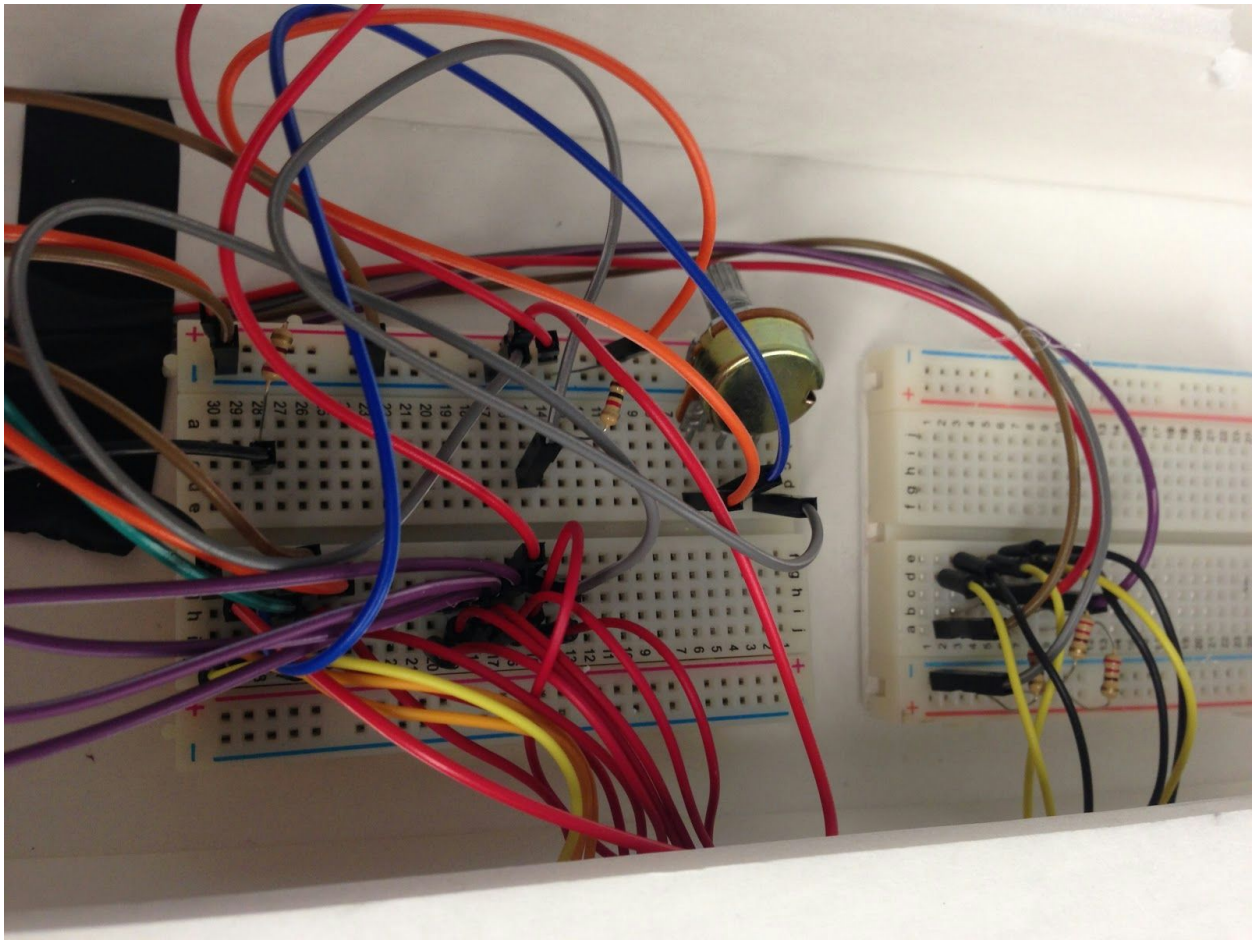
In order for the bulbs to react differently based on the distance returned by the sensor, we added if/else statements in the void loop.

```
void loop() {

 lcd.setCursor(0,1); // Sets the location at which subsequent text written to the LCD will be displayed

 long duration, distance;
 digitalWrite(trigPin, LOW);  // Added this line
 delayMicroseconds(2); // Added this line
 digitalWrite(trigPin, HIGH);
//  delayMicroseconds(1000); - Removed this line
 delayMicroseconds(10); // Added this line
 digitalWrite(trigPin, LOW);
 duration = pulseIn(echoPin, HIGH);
 distance = (duration/2) / 29.1;


 if (distance > 30) {  // This is where the LED On/Off happens
  digitalWrite(led,HIGH); // When the Red condition is met, the Green LED should turn off
  digitalWrite(led2,LOW);
  digitalWrite(led3,LOW);
  lcd.clear();
  lcd.print("FIGHT!"); // Prints "Arduino" on the LCD
}
 else if (distance > 15 && distance <= 29  ){
   digitalWrite(led,LOW);
   digitalWrite(led2,HIGH);
   digitalWrite(led3,LOW);
   lcd.clear();
   lcd.print("OHH DAMN");


 } else {
   digitalWrite(led,LOW);
   digitalWrite(led2,LOW);
   digitalWrite(led3,HIGH);
   lcd.clear();
   lcd.print("FINISH HIM!");
 }
```

Be sure to turn off the bulbs in each if statement as your turn one on in order to get the desired effect we accomplished. The variable “distance” returned the distance between an object and the sensor in centimeters. We then used the “digitalWrite” function built in to Arduino and added turned on & off the bulbs we wanted between each distance.

Step 4: Adding the LCD screen to the Arduino circuit

Adding the LCD screen is a rather complicated step, due to the number of wires required to be attached to complete the Arduino circuit. As you can see below, adding a potentiometer to the circuit is important in order to get the LCD screen to operate.



When defining our variables in the Arduino code, we included the liquid crystal library that is used for the LCD screen to operate properly. We then defined the lcd for the Arduino inputs 0, 1, 4, 5, 6, 7.

Here is a link where on the LCD screen to attach these inputs:
https://www.arduino.cc/en/Tutorial/LiquidCrystalDisplay

ArduinoArmWrestle

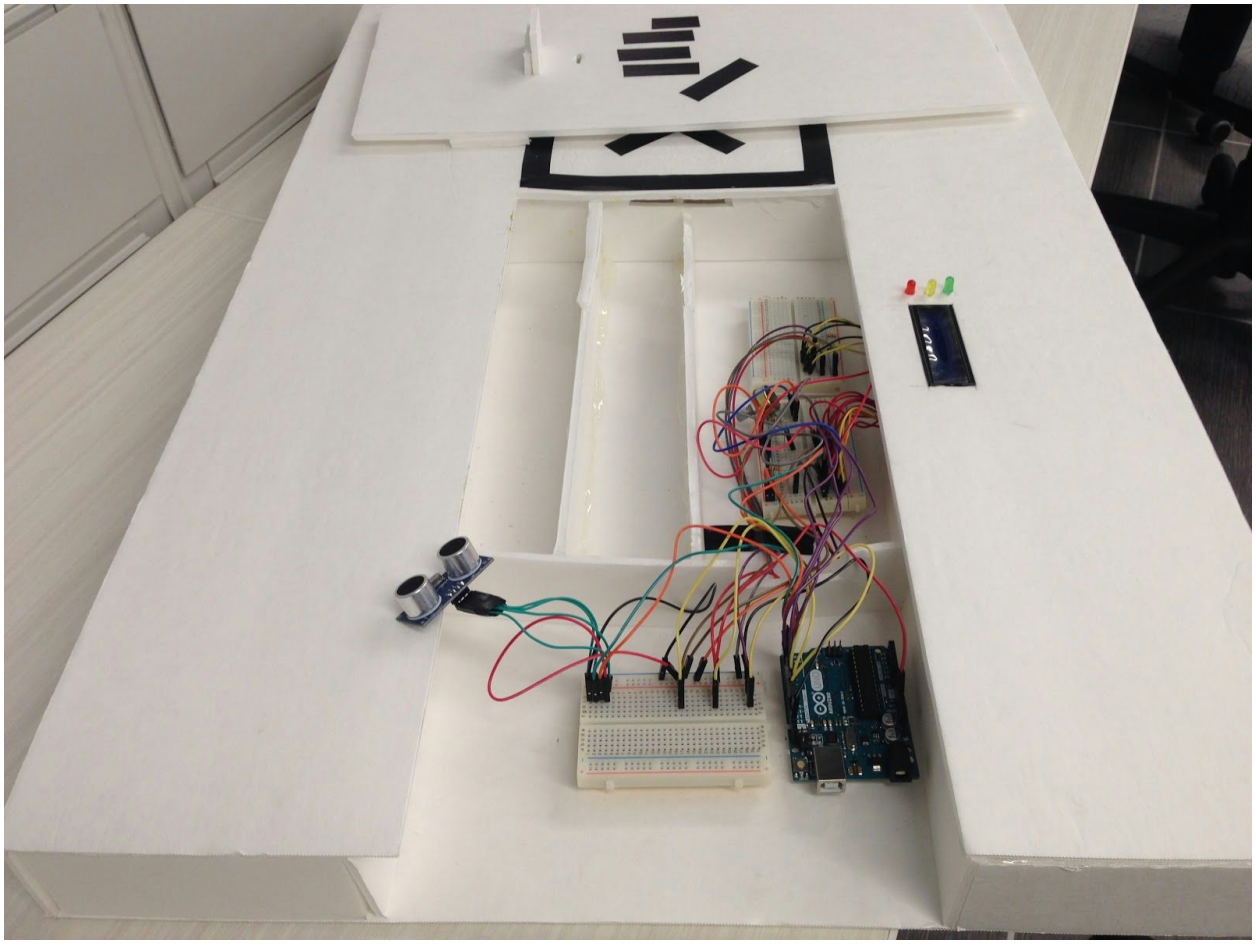
```
#define trigPin 3
#define echoPin 2
#define led 8
#define led2 9
#define led3 10

#include <LiquidCrystal.h> // includes the LiquidCrystal Library
LiquidCrystal lcd(0, 1, 4, 5, 6, 7); // Creates an LC object. Parameters: (rs, enable, d4, d5, d6, d7)

void setup() {
// Serial.begin (9600);
 pinMode(trigPin, OUTPUT);
 pinMode(echoPin, INPUT);
 pinMode(led, OUTPUT);
 pinMode(led2, OUTPUT);
 pinMode(led3, OUTPUT);
 lcd.begin(16,2); // Initializes the interface to the LCD screen, and specifies the dimensions (width and height) of the display

}
```

Here is the code that allows the lcd screen to operate, and display different text samples based on the distance returned from the sensor.

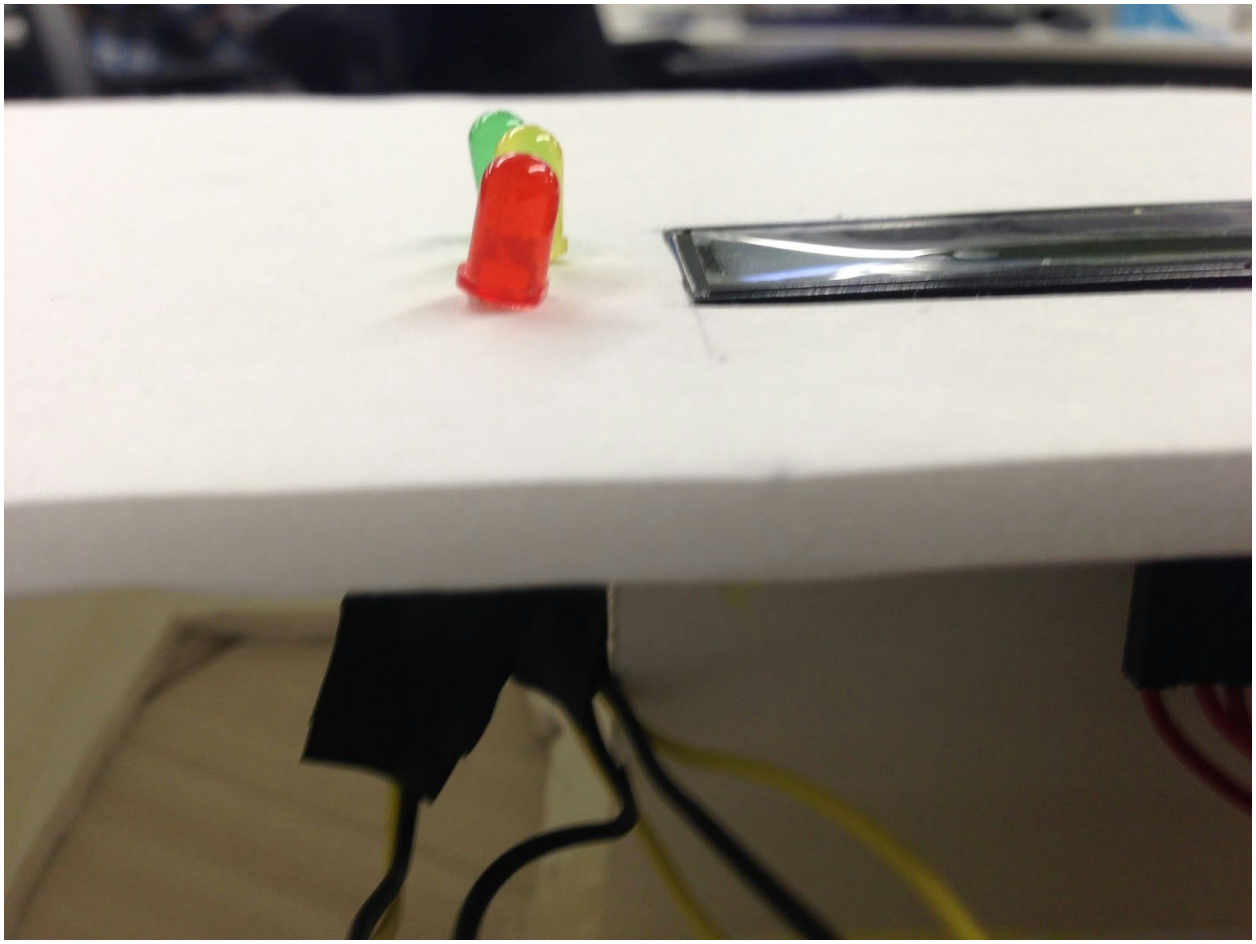
```
void loop() {
 lcd.setCursor(0,1); // Sets the location at which subsequent text written to the LCD will be displayed
 long duration, distance;
 digitalWrite(trigPin, LOW);  // Added this line
 delayMicroseconds(2); // Added this line
 digitalWrite(trigPin, HIGH);
//  delayMicroseconds(1000); - Removed this line
 delayMicroseconds(10); // Added this line
 digitalWrite(trigPin, LOW);
 duration = pulseIn(echoPin, HIGH);
 distance = (duration/2) / 29.1;


 if (distance > 30) {  // This is where the LED On/Off happens
  digitalWrite(led,HIGH); // When the Red condition is met, the Green LED should turn off
  digitalWrite(led2,LOW);
  digitalWrite(led3,LOW);
  lcd.clear();
  lcd.print("FIGHT!"); // Prints "Arduino" on the LCD
}
```

Step 5: Building the Armxr Station & casing for the Arduino

As you can see below, we built our prototype out of a strong styrofoam material that can easily be purchased at a dollar store. We then used an exacto knife & hot glue to build a strong foundation. We also built a removal casing area where we would be able to place our Adruino circuit & easily access it if we needed to make any changes or alter the way it works.



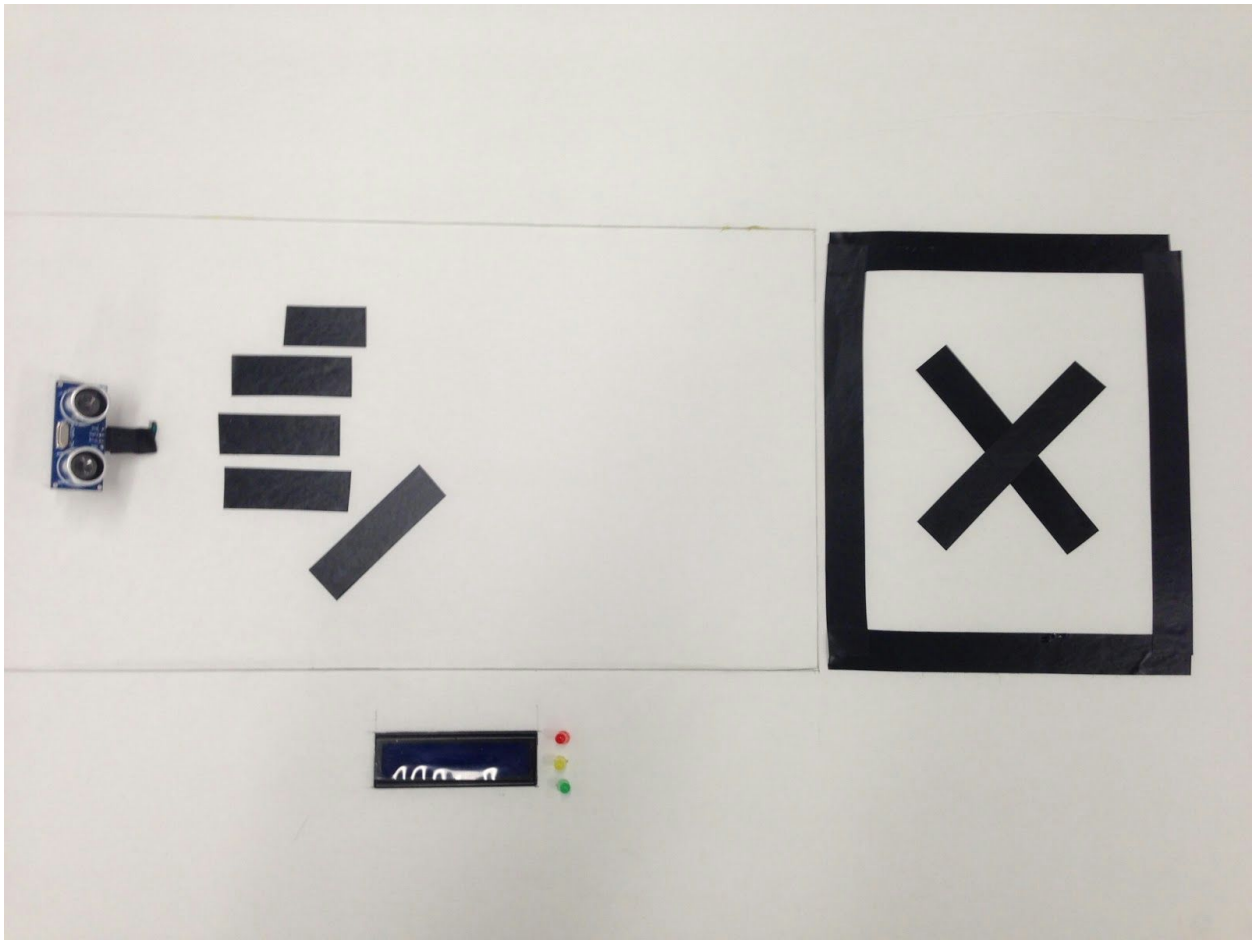
We then inserted the project within the specified casing area & then cut a hole the size of the LCD screen & built a support underneath it to keep it stable & visible to a user.



As shown above, we allowed space for the LCD screen and the LED bulbs to go through. However, we were able to still access these wires at any time to make alterations.

Step 6: Putting it all together & plugging it in

Our next step was to place our removal casing lid back on and placing the sensor on top of the casing as shown below. We added our logo & marked a reinforced area using electrical tape for where the user should place their elbow to begin.



VOILA! We were then able to power the Arduino with an extended USB cable from a computer. For a demonstration of our working project visit pic.twitter.com/nomHqc7XSc.