The Welcomer

By Vanessa Hernandez-Cruz

Product Description:

Since a young age, I always found the devices that shops have very intriguing. I am referring to the devices that alert the shop owner when a customer walks into the shop. I remember purposely waving my hand or leg in the way of the sensor just to make it produce the sound. I now realize that this might have been somewhat disturbing to the owners. I simply enjoyed the action of those devices as a kid! As an adult with engineering knowledge now, I decided to replicate the same action with an extra feature of having a hand wave at the customer thus being called *The Welcomer*. I have frequently encountered this device at stores throughout my life, as I am sure many other people have as well. I am happy that I now understand how these devices work. This project allowed me to also learn how to work with a speaker and an ultrasonic sensor in Arduino software. Now I have the opportunity to share how devices, like *The Welcomer*, work with other people I happen to walk into a store with.

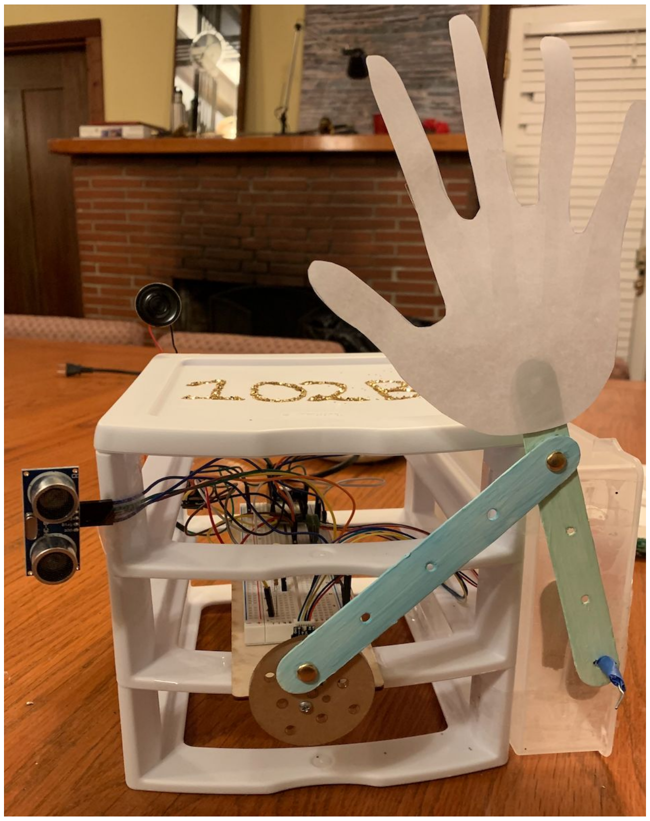


Figure 1: *The Welcomer* exhibiting its components without power yet.

Electromechanical Details:

The Housing: A Sterilite 3 drawer small countertop was used to build the base for my project. The middle section stores the breadboard, wires, microcontroller, and motor. The ultrasonic sensor is taped to the outer left side so that it does not confuse the waving mechanism (crank rocker four bar linkage and hand cutout) as a customer since the ultrasonic sensor will detect it if it crosses in front. The power cable and USB plug are handled by the open back of the housing so they do not get in the way of possibly being mistaken as a customer by the ultrasonic sensor. The mini loudspeaker is attached to the back of the housing but is still visible from any view.

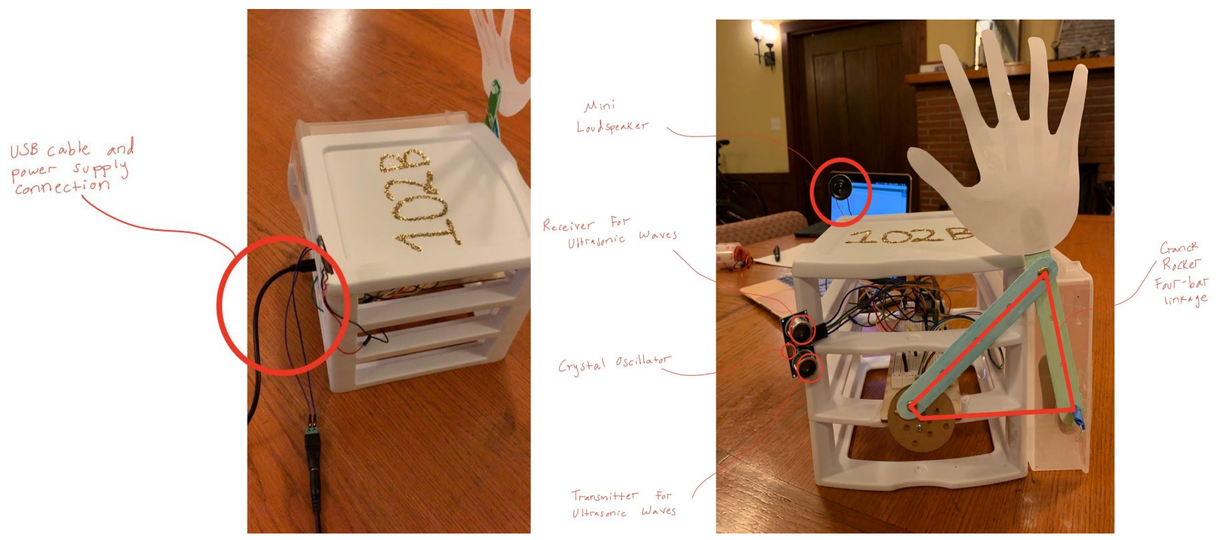

USB cable and power supply connection
Mini Loudspeaker
Receiver for Ultrasonic Waves
Crystal Oscillator
Transmitter for Ultrasonic Waves
Crank Rocker Four-bar linkage
102B

Figure 2: The circuit sits on the middle shelf shown on the right picture to prevent the four bar linkage from getting caught onto any wires.The left picture shows the connections needed to power *The Welcomer*. To allow the links to rotate without obstruction, the bottom right side of the crank rocker is installed into a disformed paperclip and kept in place with blue duct tape. Details on the ultrasonic sensor can be found on the picture on the right. The motor drives the four bar linkage, and can be found directly behind the left side of the crank rocker where it is mounted onto the same wooden surface as the breadboard.

Circuit:

The purpose of *The Welcomer* was for me to learn how to prototype a mechatronics system from start to finish using the concepts learned in ME 102B. The main components of my project are (1) the ultrasonic sensor, (2) the mini loudspeaker, and a (3) DC motor attached to a crank rocker four bar linkage.

(1) Ultrasonic sensor (US): I used the US (HC-SR04 Ultrasonic Sonar Distance Sensor) provided in the microkit. The US comes with four pins: Vcc is connected to the microcontroller's 3.3V, Trig and Echo pins are connected to a pin on the microcontroller, and the Gnd pin is grounded. Through research and trial and error, I was able to effectively calibrate the US to read distance in inches using a measuring stick and Arduino to check the values on the serial monitor. The US reads values the entire time power is on because there is always something within its measurable range.

(2) Mini loudspeaker (ML): The ML (Yootop 1W 8 Ohm Round Internal Magnet ML) was also provided in the kit. Before connecting the speaker to power, a 100 ohm resistor was added to avoid blowing it out. The other end was connected to pin A6 on the microcontroller. Distances correlating to the functionality of the mini loudspeaker can be found in Figure 4.

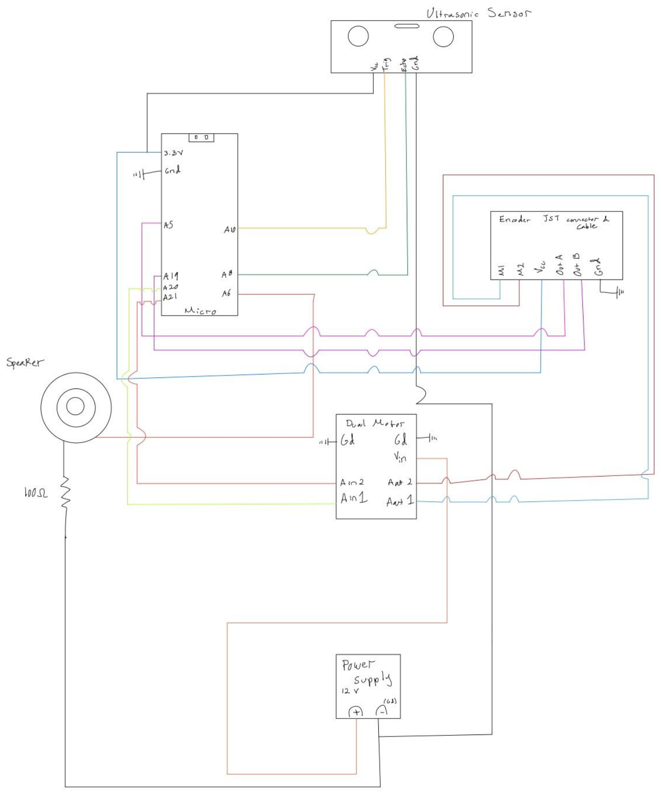


Figure 3: Circuit diagram for *The Welcomer.*

(3) DC motor with Crank rocker four bar linkage (CR): I made a crank rocker using the motor as the input crank and satisfying Grashof's condition that the sum of the shortest and longest link is less than the sum of the other two intermediate links. This CR is exemplified in Figure 2. The DC motor turns when an object/customer is detected within ten inches of the US making the CR turn and appear as a waving hand to the customer/object.

I also included the use of the H-bridge, included in our kit as well, to have the liberty to run the DC motor forwards and backwards. This is not essential to the functionality of my project; however, I enjoyed being able to physically see my DC motor turn clockwise and counterclockwise depending on the code ran. In addition, the encoder JST connector and cable (Magnetic Encoder Kit with Side-Entry Connector for Micro Metal Gearmotors, 12 CPR, 2.7-18V and 6-Pin Female JST SH-Style Cable 30cm) were used to complement the H-bridge and the rest of the circuit shown in Figure 3.

Finite State Machine:

The machine here is my project. It only has two states: waving or idling, depending on the distance detected from the US. The result of the finite state machine is shown below, and its code can be found in the appendix. The DC motor is run with open loop control meaning that it does not rectify errors or deviations from the desired angular velocity. I chose this type of control because the hand cutout waves regardless of the speed that the motor is turning (as long as it is not zero), and the waving motion, along with the speaker making a high pitched sound, is the key part for my project.

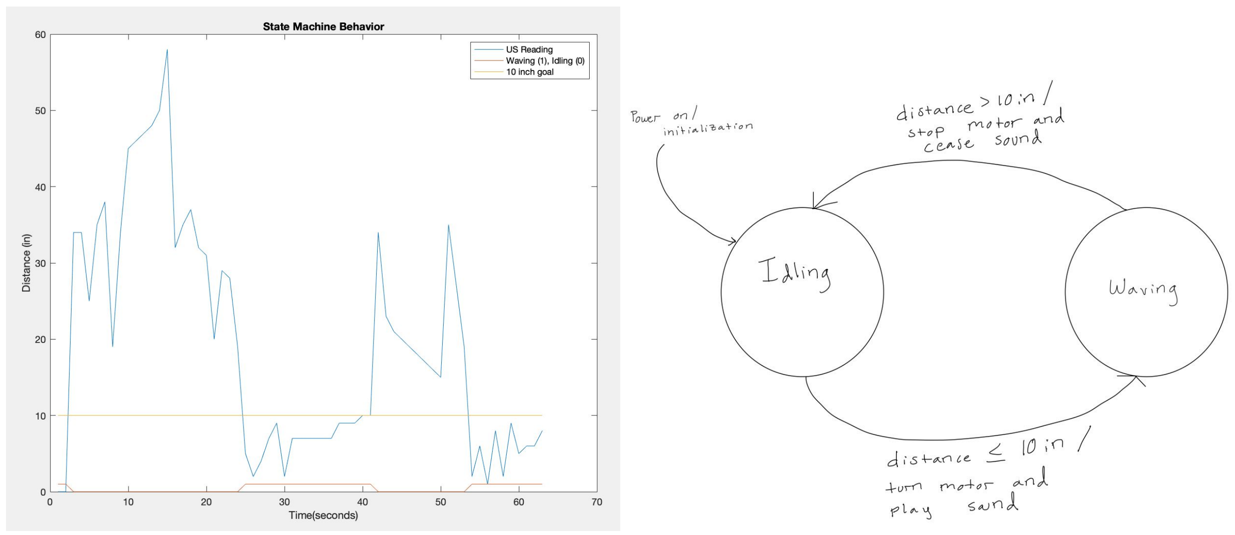
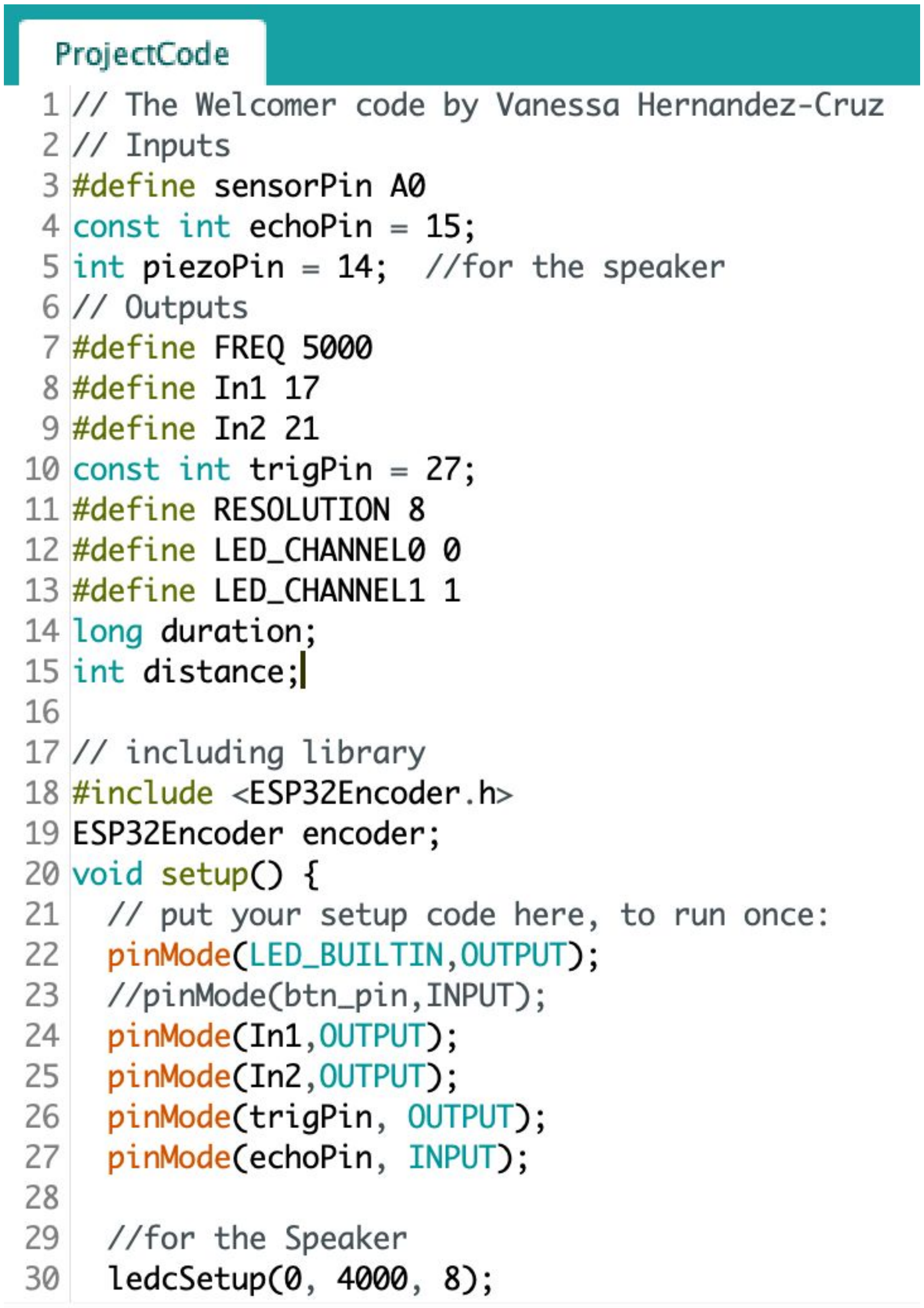


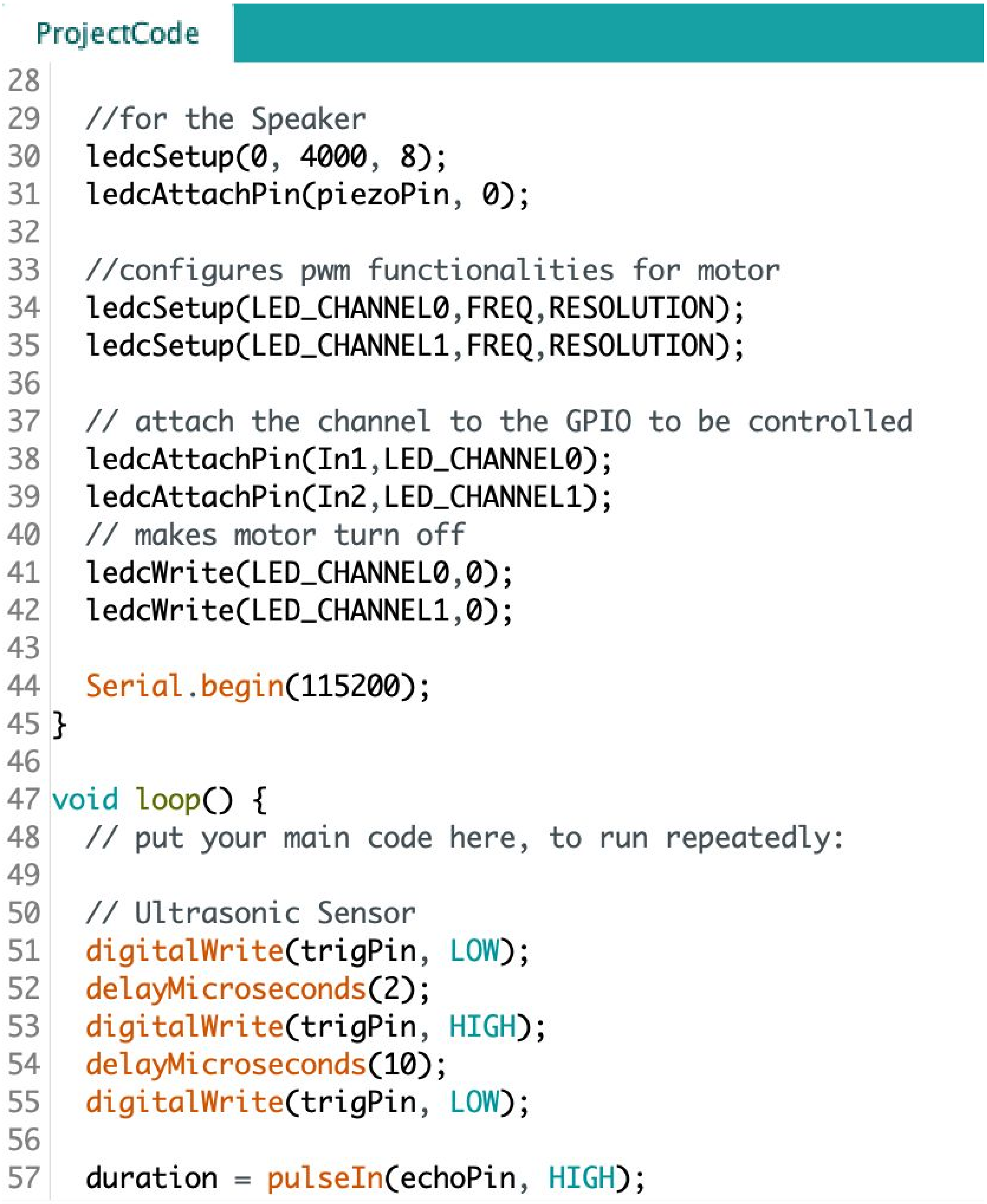
Figure 4: Finite State Diagram shown on the right. Turn motor means the motor is on and rotating while stop motor means the motor is not spinning. The left shows the behavior of the FSD. The yellow line is the threshold at 10 inches. The orange line shows what state the machine is in: 1in for waving and 0 in for idling. The blue line shows the readings of the ultrasonic sensor over time. This data was taken by manually moving the object/customer shown in the video.

The code for the functionality of *The Welcomer* can be found in the Appendix, and a physical representation can be found on the video. I created the two functions shown at the bottom of the code to be able to hear and turn off sound from the speaker since "tone" is not built into my microcontroller.

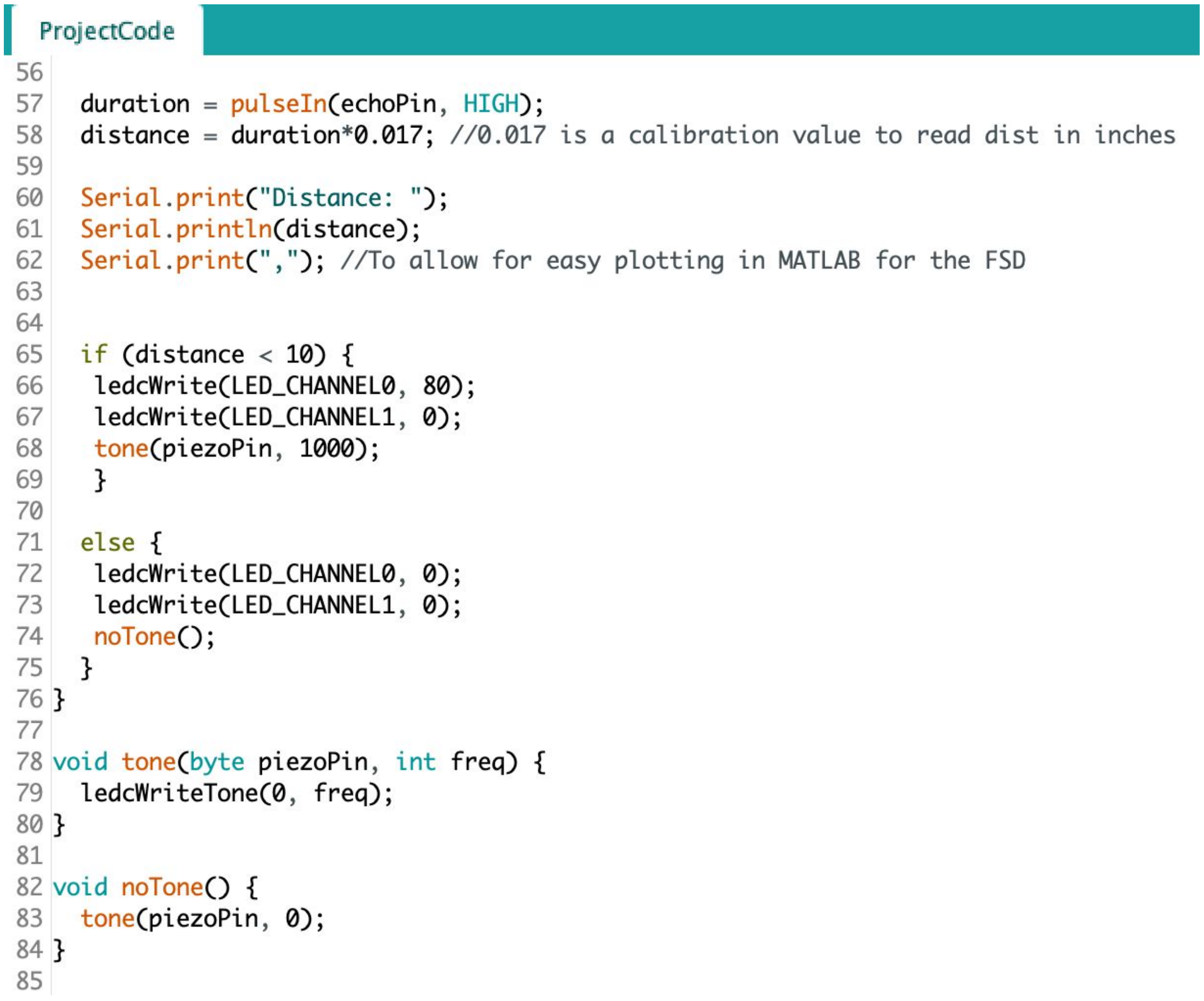
Appendix :



```
// The Welcomer code by Vanessa Hernandez-Cruz
// Inputs
#define sensorPin A0
const int echoPin = 15;
int piezoPin = 14;  //for the speaker
// Outputs
#define FREQ 5000
#define In1 17
#define In2 21
const int trigPin = 27;
#define RESOLUTION 8
#define LED_CHANNEL0 0
#define LED_CHANNEL1 1
long duration;
int distance;

// including library
#include <ESP32Encoder.h>
ESP32Encoder encoder;
void setup() {
  // put your setup code here, to run once:
  pinMode(LED_BUILTIN,OUTPUT);
  //pinMode(btn_pin,INPUT);
  pinMode(In1,OUTPUT);
  pinMode(In2,OUTPUT);
  pinMode(trigPin, OUTPUT);
  pinMode(echoPin, INPUT);

  //for the Speaker
  ledcSetup(0, 4000, 8);
```


ProjectCode

```

  //for the Speaker
  ledcSetup(0, 4000, 8);
  ledcAttachPin(piezoPin, 0);

  //configures pwm functionalities for motor
  ledcSetup(LED_CHANNEL0,FREQ,RESOLUTION);
  ledcSetup(LED_CHANNEL1,FREQ,RESOLUTION);

  // attach the channel to the GPIO to be controlled
  ledcAttachPin(In1,LED_CHANNEL0);
  ledcAttachPin(In2,LED_CHANNEL1);
  // makes motor turn off
  ledcWrite(LED_CHANNEL0,0);
  ledcWrite(LED_CHANNEL1,0);

  Serial.begin(115200);
}

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

  // Ultrasonic Sensor
  digitalWrite(trigPin, LOW);
  delayMicroseconds(2);
  digitalWrite(trigPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPin, LOW);

  duration = pulseIn(echoPin, HIGH);
```



```

  duration = pulseIn(echoPin, HIGH);
  distance = duration*0.017; //0.017 is a calibration value to read dist in inches

  Serial.print("Distance: ");
  Serial.println(distance);
  Serial.print(","); //To allow for easy plotting in MATLAB for the FSD


  if (distance < 10) {
   ledcWrite(LED_CHANNEL0, 80);
   ledcWrite(LED_CHANNEL1, 0);
   tone(piezoPin, 1000);
   }

  else {
   ledcWrite(LED_CHANNEL0, 0);
   ledcWrite(LED_CHANNEL1, 0);
   noTone();
  }
}

void tone(byte piezoPin, int freq) {
  ledcWriteTone(0, freq);
}

void noTone() {
  tone(piezoPin, 0);
}

```