One Shot, *Two* Shot, Red Shot, Blue Shot

Build Instructions

Target Console

1) Construct the target console from ½" thick plywood and ¾" thick select trim pieces. Use 1/8" plywood or hardboard for the 15° angled section serving as the target area floor. See **Figure 1** below for general dimensions and construction shape. An opening of 33" wide by 19" tall works well if using 5 targets to shoot at.

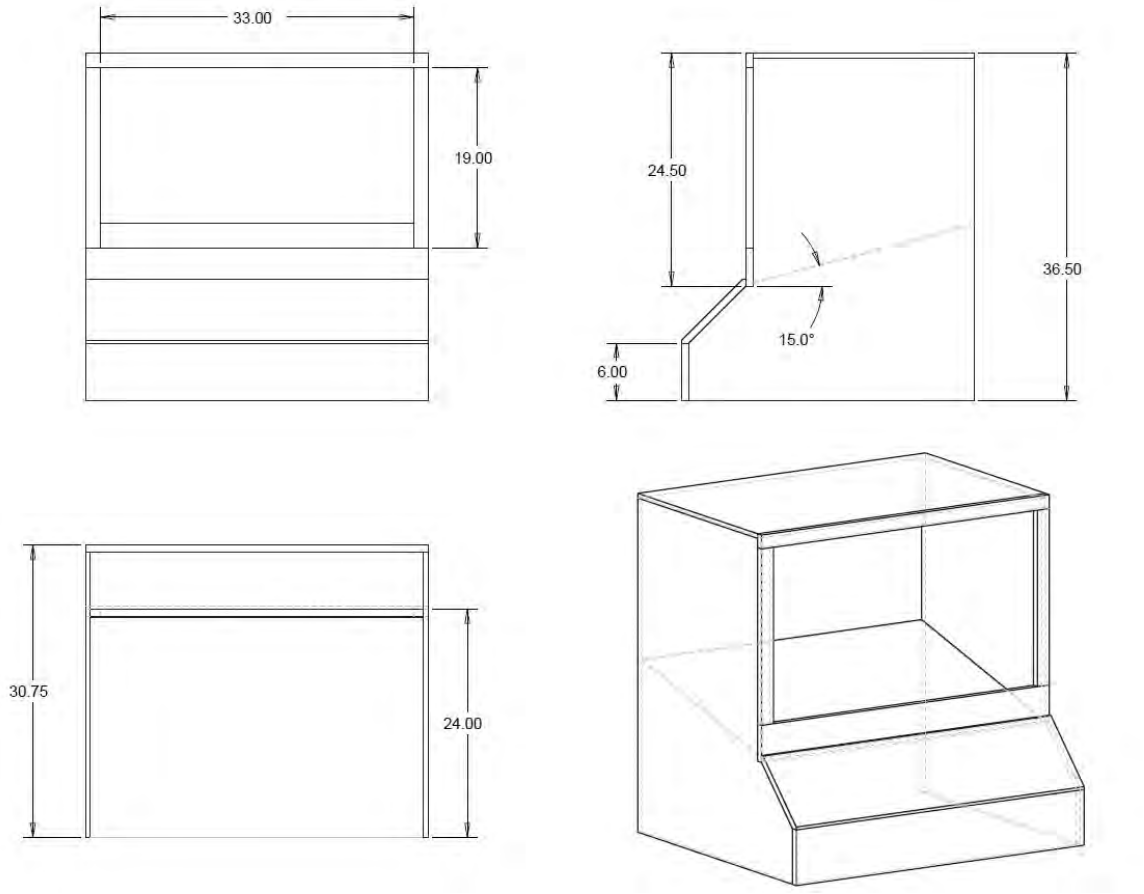


Figure 1 – Console Shape and Dimensions

2) Next, choose 5 pictures or figures to use as targets. Enlarge the pictures to an adequate size (Pictures that are roughly half of an 8.5x11 sheet work well).
3) Cut out the target pictures and trace them onto 1/8" plywood or hardboard.
4) Cut out the traced pictures, leaving roughly ¼" or room on all sides.
5) Paint the cutout wooden targets, and then attach the cutout pictures on top (hodgepodge works well for this).
6) Drill holes in the targets to fit LEDs, and mount LEDs in the holes.
7) Next, cutout 5 different length wooden beams to hang the targets with (1x2" wood boards work well for this).
8) Connect the targets to the shafts via hinges, and ensure they swing feely.

9) Attach a switch to the wooden beam of each target so that it is activated when the target swings.
10) Next, attach these newly constructed target assemblies to the console by screwing the wooden beams in from the top of the console. Be sure to space the targets out and position them in a random fashion.
11) Place a back drop to the target area that matches the theme of the game. In our case, we used the Lorax forest because of the Dr. Seuss theme.
12) Mount two 7-segment displays on the top of the console, and mount speakers on the inner part of the large bottom trip piece of the target area.

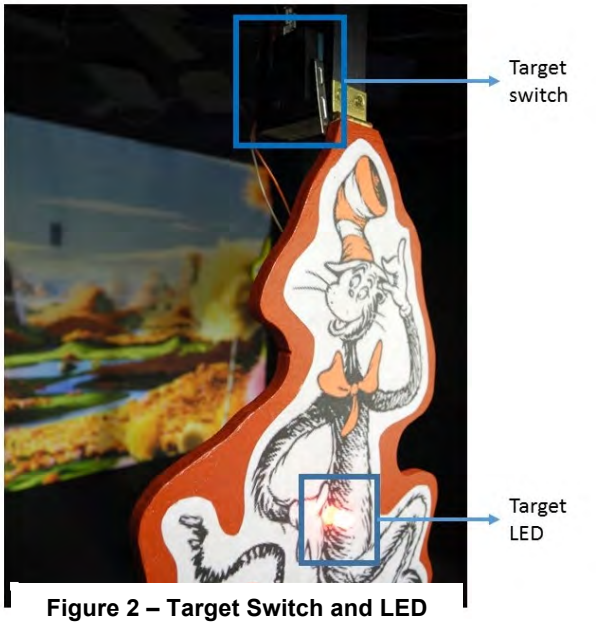


Figure 2 – Target Switch and LED

13) Mount an LCD interface screen and ultrasonic proximity sensor to the front of the cosole.
14) Lastly, run all of the wires of the 7-segment displays and targets (LED and switches) along the top and down the sides into the belly of the console.

Figure 3 – Five targets assembled and mounted with the Lorax forest as backdrop. From left to right – Thing 1 and Thing 2, Sneetch, Cat in the Hat, One Fish Two Fish Red Fish Blue Fish, and Horton hiding in

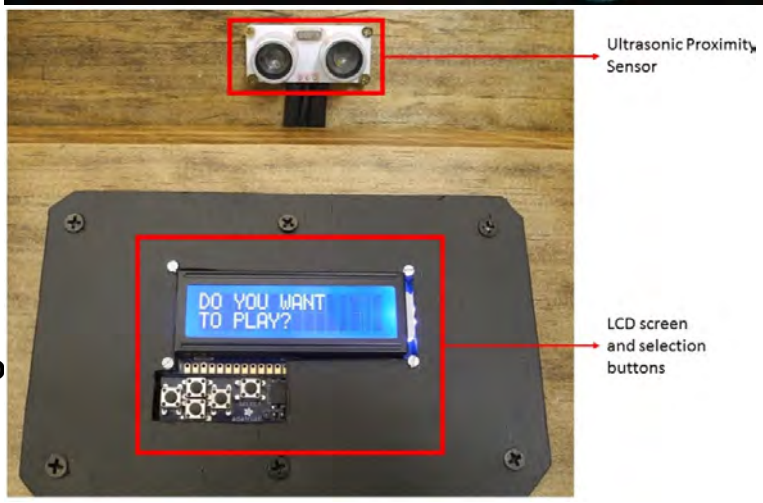


Figure 4 – LCD screen with user interface and Ultrasonic Proximity Sensor

Co

1) First, obtain a high power electric staple gun. We used an Arrow Fastener model ETFX50D electric staple gun.
2) Remove the staple reservoir on the front of the gun.
3) Open up the staple gun by removing the screws holding it together.
4) Disconnect the manual trigger switch that came with the gun. It will be connected to the coil driver board.
5) Locate the coil driver board, and solder lead wires across where the manual switch was connected.
6) Place a new electronic switch where the original trigger switch was connected.
7) Connect the new switch to the control side of a relay, and connect the lead wires from the coil driver board to the power side of the relay.
8) Position everything so that it fits nicely and close the staple gun back up.
9) Next, construct a revolving barrel to hold wooden dowels/darts. (We used a 3D printing method using an ABS plastic material).
10) Create a U shaped frame piece from 1/8" x ¾" metal to connect to the front of the gun to support the barrel, stepper motor, and infrared sensor.
11) Obtain a long slender rod that will be used as the axis for the barrel, and position it so that the barrels line up correctly.
12) Mount the stepper motor and barrel, and run a belt between the two pulleys.
13) Mount the infrared sensor below the stepper motor.
14) Lastly, mount 6 LEDs to the top of the gun. The best way to do this is to mount the LEDs to a separate piece of plastic and then mount the piece of plastic to the gun. Be sure to run a common power to the LEDs and keep the negative wires of the LEDs independently controlled.
15) Make darts to fit inside of the revolving barrel. Wooden dowels work well for this.
16) Lastly, run all of the wires down the side of the gun and secure them using tape or other means. Run the wires, along with the power cord for the coil that came with the staple gun, through a shielding hose to the console, leaving roughly 10 feet of length between the gun and console.

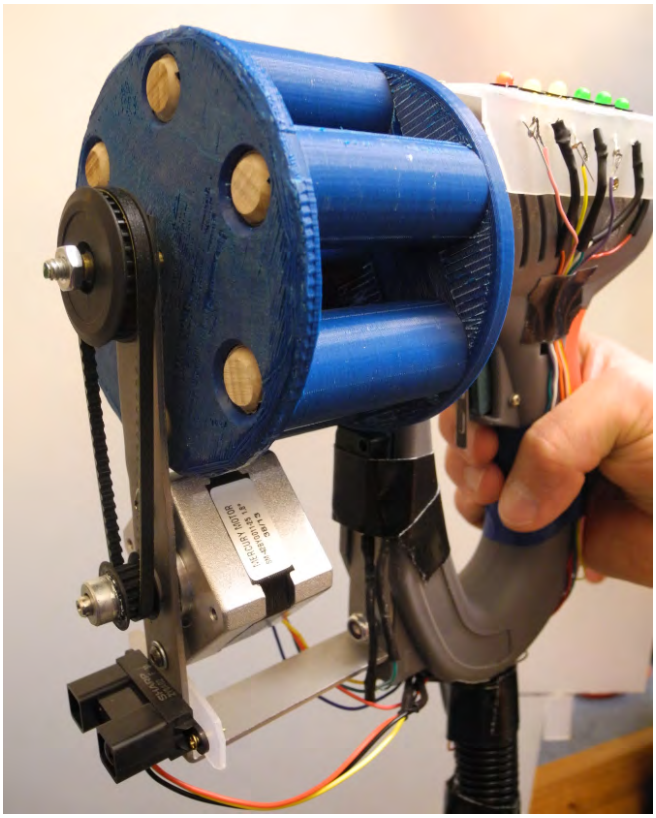


Figure 5 – Front View of Gun

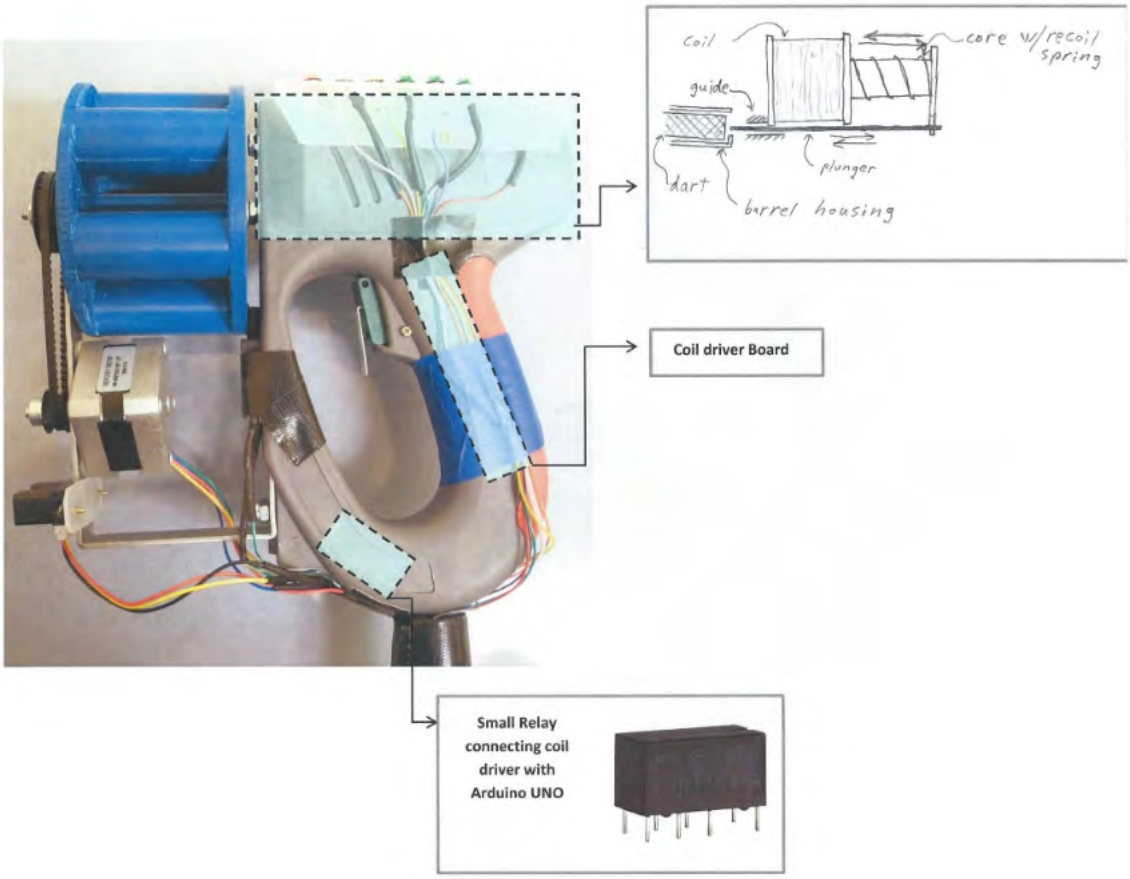


Figure 6 – Coil Gun Interior Break-down

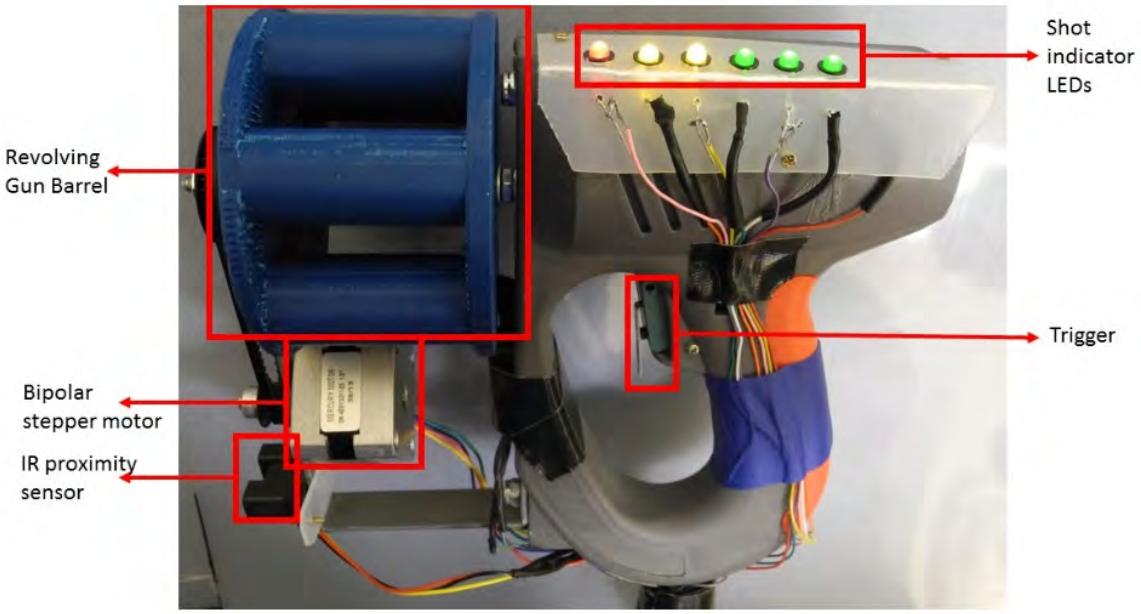


Figure 7 – Coil Gun Exterior Break-down

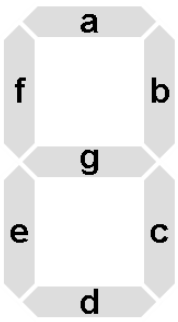
Electronics

***for any questions refer to Functional Diagram and Wiring Schematic for assistance (Figures 9 & 10)**

1) Connect pin 8 of the MEGA to the base of an NPN transistor. Also connect that same pin of the NPN transistor through a 1kohm resistor to ground. Connect the collector of the transistor to power, and connect the emitter of the transmitter to the power of the speakers. Connect the ground of the speakers to ground, and use a flyback diode.
2) Connect the communication lines of the LCD screen shield to ports SDA20 and SCL21 of the MEGA. Connect power and ground to the shield as well.
3) Connect pin 24 of the MEGA to pin 1 of both PICs using a 1 kohm pull down resistor.
4) Connect pin 27 of the MEGA to pin 4 of the UNO.
5) Connect pin 31 of the MEGA to pin 2 of the ONE's digit PIC using a 1 kohm pull down resistor.
6) Connect pin 36 of the MEGA to power.
7) Connect pin 40 of the MEGA to RESET of the UNO.
8) Connect pins 44 through 48 of the MEGA to the target switches using 1 kohm pull down resistors.
9) Connect pins 49 through 53 of the MEGA to the ground side of the target LEDs, using 330 ohm resistors in series.
10) Connect power and ground to the infrared proximity sensor, and connect the sensor wire to pin A0 of the UNO.
11) Connect the barrel reset switch to pin A4 of the UNO using a 1 kohm pull down resistor.
12) Connect pin 2 of the UNO to STEP of the stepper motor driver, and pin 3 of the UNO to DIR of the stepper motor driver.
13) Connect pin 5 of the UNO to the control side of the solenoid relay.
14) Connect pin 6 of the UNO to the trigger switch using a 1 kohm pull down resistor.
15) Connect pins 7 through 12 of the UNO to the ground side of the gun LEDs using 330 ohm resistors in series.
16) Connect power and ground to the ultrasound proximity sensor, and connect the sensor wire to pin 13 of the UNO.
17) Connect pin 4 of both PICs to power through a 1 kohm resistor.
18) Connect pin 5 of both PICs directly to ground.
19) Connect pins 6 through 12 of both PICs to their corresponding 7-Segment Display Leg through 330 ohm resistors using the following table:

Table 1 – Seven Segment Display Pin Numbers vs Leg

PIC PIN Number	7-Segment Display Leg
6	E
7	D
8	G
9	C
10	B
11	A



12	F

Figure 8 – Seven Segment Display Legs

20) Connect pin 13 of the ONEs digit PIC to pin 2 of the TENs digit PIC.
21) Connect pin 14 of both PICs directly to power, and connect a 0.1microfarad capacitor in parallel.
22) A separate 12V power source should connect to the common anode of both 7-segment displays.
23) Attach that same 12V power source to the stepper motor driver circuit, and be sure to attach ground of the 12V power source to the common ground.
24) Attach the ground of either the same high output 12V source or an additional high output source to the power side of the relay (the relay connected to pin 5 of the UNO). Ground both the power and control side of the relay.
25) Attach the power to a transistor that needs to be connected across the lead wires that were installed across where the manual trigger switch used to be connected to the staple gun driver board.

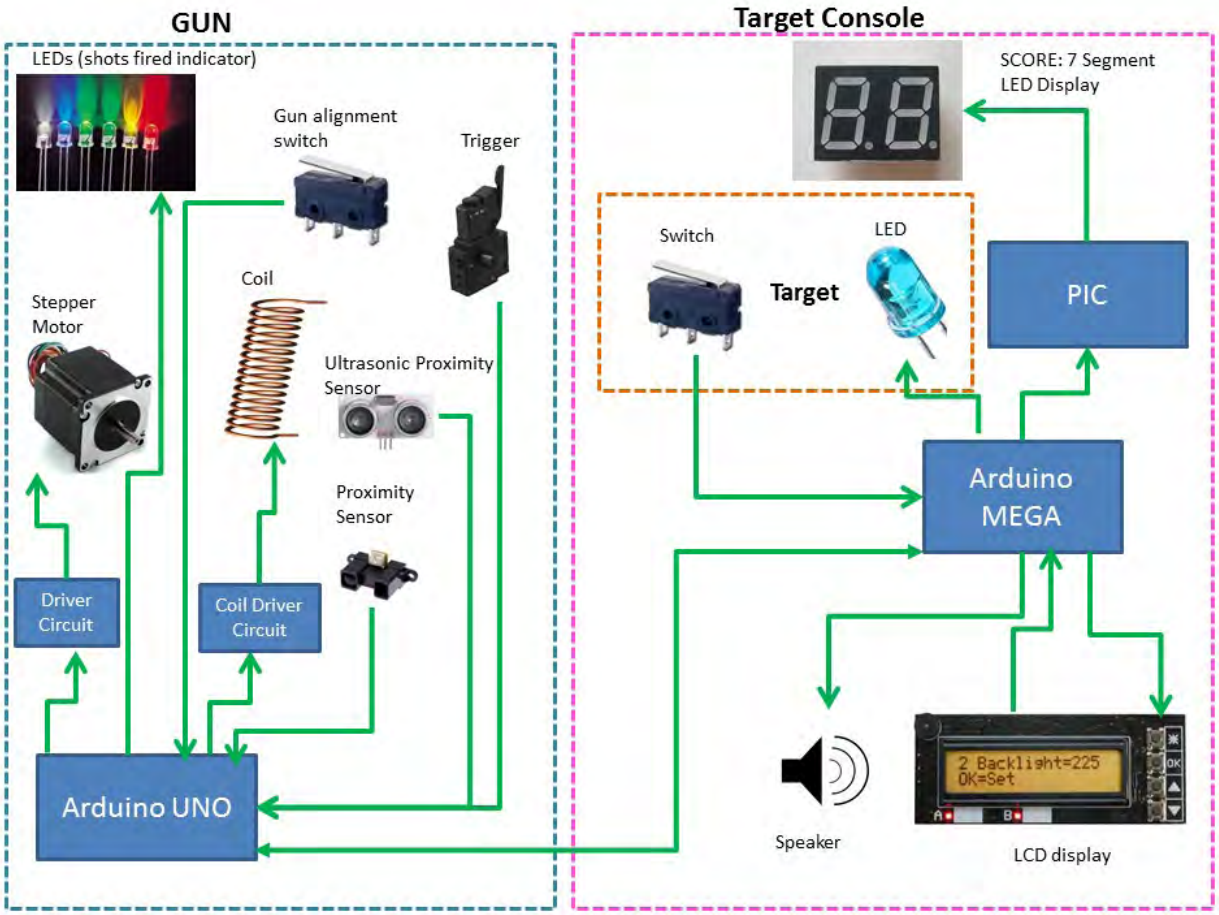


Figure 9 – Functional Diagram

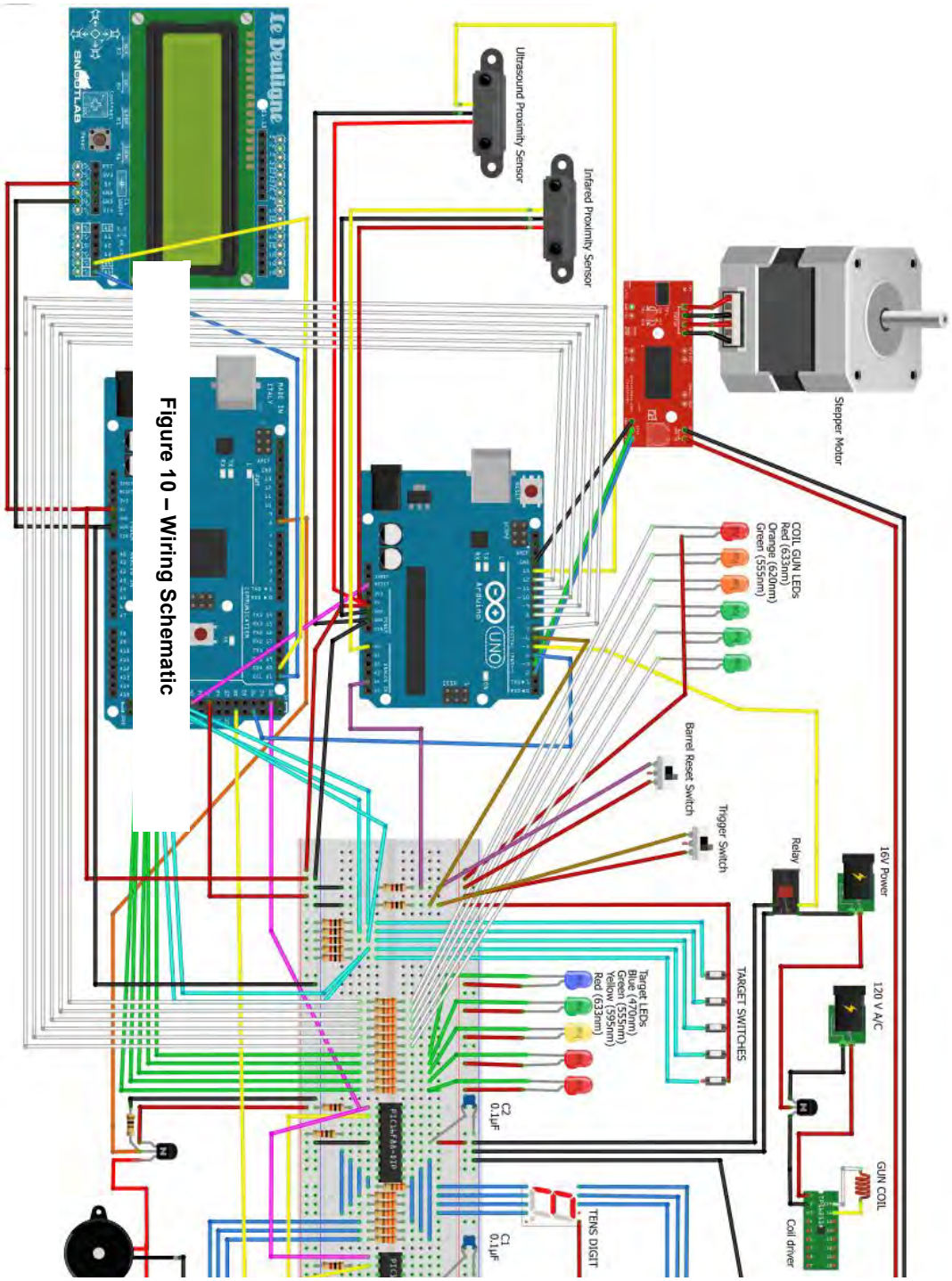


Figure 10 – Wiring Schematic

Arduino UNO Annotated Code

```
#include "Arduino.h"

int Pin2 = 2; // motor step
int Pin3 = 3; // motor direction
int Pin4 = 4; // end game signal
int Pin5 = 5; // coil activation
int Pin6 = 6; // trigger signal
int Pin7 = 7; // gun led #1
int Pin8 = 8; // gun led #2
int Pin9 = 9; // gun led #3
int Pin10 = 10; //gun led #4
int Pin11 = 11; //gun led #5
int Pin12 = 12; //gun led #6
int Pin13 = 13; //ultrasound proximity sensor
int PinA0 = A0; //infared proximity sensor
int PinA4 = A4; //barrell reset switch
int i=0;  // number of shots variable
int k=0;  // motor motion variable
int zach=0;  //prevent trigger during motor motion variable
int jake=0; // tied into end game signal
int duration; //duration for ultrasound sensor
int cm; //distance for ultrasound sensor
int ValueA0=LOW; //intitialize infared value low
int Value3=HIGH; //intitialize motor direction CCW
int Value6=LOW; //initialize trigger signal low
int ValueA4=LOW; //initial barrel reset switch low

void setup()
{
  pinMode(Pin2, OUTPUT);
  pinMode(Pin3, INPUT);
  pinMode(Pin4, OUTPUT);
  pinMode(Pin5, OUTPUT);
  pinMode(Pin6, INPUT);
  pinMode(Pin7, OUTPUT);
  pinMode(Pin8, OUTPUT);
  pinMode(Pin9, OUTPUT);
  pinMode(Pin10, OUTPUT);
  pinMode(Pin11, OUTPUT);
  pinMode(Pin12, OUTPUT);
  pinMode(PinA0, INPUT);
  pinMode(PinA4, INPUT);
}
```

```
void Reset()
{
 ValueA4=digitalRead(PinA4); // Barrel Reset
 if (ValueA4 == LOW)
 {
      digitalWrite(Pin3, HIGH);  //CCW
      digitalWrite(Pin2, HIGH);  // 1 step
      delayMicroseconds(500);
      digitalWrite(Pin2, LOW);
      delayMicroseconds(500);
 }
 if (ValueA4 == HIGH) //Barrel Reset complete
 {
  i++;
  zach++;
 }

//Turn on all Gun LEDs
digitalWrite (Pin7, LOW);
digitalWrite (Pin8, LOW);
digitalWrite (Pin9, LOW);
digitalWrite (Pin10, LOW);
digitalWrite (Pin11, LOW);
digitalWrite (Pin12, LOW);
}

void MotorMotion()
{
      digitalWrite(Pin3, LOW);  // Large Gear = 34,  Small Gear = 15
      digitalWrite(Pin2, HIGH);   // 1 step
      delayMicroseconds(500);
      digitalWrite(Pin2, LOW);
      delayMicroseconds(500);
}

void Always()
{

ValueA0=analogRead(PinA0);  //Infared Proximity Sensor
 if (ValueA0 <= 300) //Infared Sensor prevents shooting at close range
 {
  if (cm > 150)  //UltraSound Sensor prevents shooting at close range
  {
   ValueA4=digitalRead(PinA4);
  if (zach >= 1) //Prevent trigger during barrel reset
  {
```

```
   if (k==0) //Prevent trigger during motor motion
   {
Value6=digitalRead(Pin6);  //Activate Coil
if (Value6 == HIGH)
{
  digitalWrite(Pin5, HIGH);
  delay(25);
  digitalWrite(Pin5, LOW);
  delay(250);
  i++;

if (i==2) //Remove one LED from Gun
{
  digitalWrite (Pin7, HIGH);
  digitalWrite (Pin8, LOW);
  digitalWrite (Pin9, LOW);
  digitalWrite (Pin10, LOW);
  digitalWrite (Pin11, LOW);
  digitalWrite (Pin12, LOW);
}
if (i==3) //Remove one more LED from Gun
{
  digitalWrite (Pin7, HIGH);
  digitalWrite (Pin8, HIGH);
  digitalWrite (Pin9, LOW);
  digitalWrite (Pin10, LOW);
  digitalWrite (Pin11, LOW);
  digitalWrite (Pin12, LOW);
}
if (i==4) //Remove one more LED from Gun
{
  digitalWrite (Pin7, HIGH);
  digitalWrite (Pin8, HIGH);
  digitalWrite (Pin9, HIGH);
  digitalWrite (Pin10, LOW);
  digitalWrite (Pin11, LOW);
  digitalWrite (Pin12, LOW);
}
if (i==5) //Remove one more LED from Gun
{
  digitalWrite (Pin7, HIGH);
  digitalWrite (Pin8, HIGH);
  digitalWrite (Pin9, HIGH);
  digitalWrite (Pin10, HIGH);
  digitalWrite (Pin11, LOW);
  digitalWrite (Pin12, LOW);
```

```
  }
  if (i==6) //Remove one more LED from Gun
  {
    digitalWrite (Pin7, HIGH);
    digitalWrite (Pin8, HIGH);
    digitalWrite (Pin9, HIGH);
    digitalWrite (Pin10, HIGH);
    digitalWrite (Pin11, HIGH);
    digitalWrite (Pin12, LOW);
  }
  if (i==7) //Remove one more LED from Gun
  {
    digitalWrite (Pin7, HIGH);
    digitalWrite (Pin8, HIGH);
    digitalWrite (Pin9, HIGH);
    digitalWrite (Pin10, HIGH);
    digitalWrite (Pin11, HIGH);
    digitalWrite (Pin12, HIGH);
  }
    delay(100);
    k++;



  }    //End Activate Coil
     } //End prevent trigger during motor motion
   }  //End prevent trigger during barrel reset
   } //End Ultrasound Prox Sensor
} // End Proximity Sensor
} //End Always Function

void loop()
{
  if (i==0)  //initially set to 0
     {
    Reset ();
     }
  if (i<=6) //Active until 6 shots taken
     {
    Always();
    if (zach>0)
     {
    if (k==0) //Ultrasound sensor
     {

//UltraSound Sensor reading
```

```
  pinMode(Pin13, OUTPUT);
  digitalWrite(Pin13, LOW);
  delayMicroseconds(2);
  digitalWrite(Pin13, HIGH);
  delayMicroseconds(5);
  digitalWrite(Pin13, LOW);

  pinMode(Pin13, INPUT);
  duration = pulseIn(Pin13, HIGH); //UltraSound Time

  cm = duration/29/2;  //Convert UltraSound Time to Distance

  delay(100);

    }
    }
     if (k>0)
     {
      if(k<605) // 60 deg of motion
      {
      MotorMotion();
      k++;
      }
     }
    if (k==605)  // once 60 deg is complete
    {
    k=0; //reset k to 0
    }
    }
   else
   {
    if (jake == 0)  //only allow end game signal for 4 seconds
    {
    digitalWrite (Pin4, HIGH); // End Game Signal
    delay(4000);  //4 second delay
    digitalWrite (Pin4, LOW);
    jake++;
    }
   }
}
```

PicBasic ONES DIGIT Annotated Code

```
DEFINE OSC 8
OSCCON.4=1
OSCCON.5=1
OSCCON.6=1

ANSEL=0

pins    var     byte[16]
I       var     byte

TRISA=%00011100
TRISB=%00000000

pins[ 0] = %10000010
pins[ 1] = %10110111
pins[ 2] = %11000001
pins[ 3] = %10010001
pins[ 4] = %10110100
pins[ 5] = %10011000
pins[ 6] = %10001000
pins[ 7] = %10110011
pins[ 8] = %10000000
pins[ 9] = %10110000

I=0
Gosub Updatepins

myloop:

If(PORTA.2==0)Then 'reset
I=0
Gosub Updatepins
Pause 100
Endif

If(PORTA.3==1)Then 'increment
If(I==9)Then
I=0
Low PORTB.7 'send increment to tens digit
Pause 99
Else
I=I+1
```

```
Endif

Gosub Updatepins
Pause 100
Endif

Goto myloop

Updatepins:
PORTB=pins[I]
Return

End
```

PicBasic TENS DIGIT Annotated Code

```
DEFINE OSC 8
OSCCON.4=1
OSCCON.5=1
OSCCON.6=1

ANSEL=0

pins    var     byte[16]
I       var     byte

TRISA=%00011100
TRISB=%00000000

pins[ 0] = %10000010
pins[ 1] = %10110111
pins[ 2] = %11000001
pins[ 3] = %10010001
pins[ 4] = %10110100
pins[ 5] = %10011000
pins[ 6] = %10001000
pins[ 7] = %10110011
pins[ 8] = %10000000
pins[ 9] = %10110000

I=0
Gosub Updatepins

myloop:

If(PORTA.2==0)Then 'reset
I=0
Gosub Updatepins
Pause 100
Endif

If(PORTA.3==0)Then 'increment
If(I==9)Then
I=0
Else
I=I+1
Endif
```

```
Gosub Updatepins
Pause 100
Endif

Goto myloop

Updatepins:
PORTB=pins[I]
Return

End
```

Arduino MEGA Annotated Code

```
// include the library code:
#include <Wire.h>
#include <Adafruit_MCP23017.h>
#include <Adafruit_RGBLCDShield.h>
#include "pitches.h"
Adafruit_RGBLCDShield lcd = Adafruit_RGBLCDShield();
// These #defines make it easy to set the backlight color
#define RED 0x1
#define YELLOW 0x3
#define GREEN 0x2
#define TEAL 0x6
#define BLUE 0x4
#define VIOLET 0x5
#define WHITE 0x7
int selectval = 0; //variable for LCD screen selection options
int UNORESETpin = 40; //soft reset for UNO at game start
int ENDpin = 27;  //end of game signal
int valENDpin;
int melody1[] = {
  NOTE_C4, NOTE_G3,NOTE_G3, NOTE_A3, NOTE_G3,0, NOTE_B3, NOTE_C4}; // notes
in the start up melody
int melody2[] = {
  NOTE_C4, NOTE_D4, NOTE_F4};// notes in the hit target melody
int noteDurations1[] = {
  4, 8, 8, 4,4,4,4,4 };// note durations for melody 1: 4 = quarter note, 8 = eighth note, etc.:
int noteDurations2[] = {
  8, 8, 8}; // note durations for melody 2
int screenopen = 0;
int PICRESETpin = 22; //reset pin for PIC at beginning of game
int Pin31 = 31;
int Pin36 = 36; //on/off switch pin
int Pin43 = 43;
int Pin44 = 44;//target 1
int Pin45 = 45;//target 2
int Pin46 = 46;//target 3
int Pin47 = 47;//target 4
int Pin48 = 48;//target 5
int Pin49 = 49;//LED1
int Pin50 = 50;//LED2
int Pin51 = 51;//LED3
int Pin52 = 52;//LED4
int Pin53 = 53;//LED5
int j=0;
int repeat;
```

```
//////////////////////////VOID SETUP////////////////////////////
void setup() {
  // Debugging output
  Serial.begin(9600);
  // set up the LCD's number of columns and rows:
  lcd.begin(16, 2);
  //declare pin modes
  pinMode(ENDpin,INPUT);
  pinMode(UNORESETpin,OUTPUT);
  pinMode(PICRESETpin, OUTPUT);
  pinMode(Pin31, OUTPUT);
  pinMode(Pin36, INPUT);
  pinMode(Pin44, INPUT);  //target1
  pinMode(Pin45, INPUT);  //target2
  pinMode(Pin46, INPUT);  //target3
  pinMode(Pin47, INPUT);  //target4
  pinMode(Pin48, INPUT);  //target5
  pinMode(Pin49, OUTPUT);  //led1
  pinMode(Pin50, OUTPUT);  //led2
  pinMode(Pin51, OUTPUT);  //led3
  pinMode(Pin52, OUTPUT);  //led4
  pinMode(Pin53, OUTPUT);  //led5

  //set up starting values
  digitalWrite(UNORESETpin,HIGH);
  digitalWrite(PICRESETpin, LOW);
  digitalWrite(Pin49, HIGH);  //writing LEDs to ground powered
  digitalWrite(Pin50, HIGH);  //writing LEDs to ground powered
  digitalWrite(Pin51, HIGH);  //writing LEDs to ground powered
  digitalWrite(Pin52, HIGH);  //writing LEDs to ground powered
  digitalWrite(Pin53, HIGH);  //writing LEDs to ground powered

  // iterate over the notes of the melody:
   for (int thisNote = 0; thisNote < 8; thisNote++)
   {
    // to calculate the note duration, take one second
    // divided by the note type.
    //e.g. quarter note = 1000 / 4, eighth note = 1000/8, etc.
    int noteDuration = 1000/noteDurations1[thisNote];
    tone(8, melody1[thisNote],noteDuration);

    // to distinguish the notes, set a minimum time between them.
    // the note's duration + 30% seems to work well:
    int pauseBetweenNotes = noteDuration * 1.30;
    delay(pauseBetweenNotes);
    // stop the tone playing:
```

```
   noTone(8);
  }
  // Print a start up message to the LCD.
  lcd.setCursor(0,0);
    lcd.print("DO YOU WANT");
  lcd.setCursor(0,1);
    lcd.print("TO PLAY?");
  lcd.setBacklight(WHITE);
}
uint8_t i=0;
////////////////////////////VOID LOOP/////////////////////////////
void loop()
{
  while (screenopen==0)             //while the screen variable is 0, it will run the screen program.
  {
  screen();                         //run the screen program
  }
  delay(250);                       //delay 250 milliseconds
  int onoff = digitalRead(Pin36);    //initialize the On/off pin variable
  if (onoff == HIGH)                //if the console is on, it will go through the program
   {
    int theend = digitalRead(ENDpin); //set up end of game signal variable
    if (theend == LOW)              //if there is no end of game signal, it will go through the program
    {
    //if (tessa == 0)
     //{
     //PLAY STARTUP MUSIC
     //tessa++;
     //}
    delay(5000);                 //delay 5 seconds
    Always();                   //run the Always program
    screenopen = 0;              //set screen variable to 0
    }
   }
}
////////////////////////SCREEN PROGRAM/////////////////////////////
void screen()
{
  uint8_t buttons = lcd.readButtons();        // setting up the read buttons function
  int theend = digitalRead(ENDpin);           // setting up the end signal pin
  // Print a message to the LCD.
  if (selectval==3){                          // selectval is the selection variable for the LCD screen. Its
value lets the program know which message to display.
  lcd.setCursor(0,0);
    lcd.print("DO YOU WANT      ");
  lcd.setCursor(0,1);
```

```
   lcd.print("TO PLAY?");
  lcd.setBacklight(WHITE);
  }
 if (buttons) {                          // starting the IF statement for the buttons on the LCD screen
   lcd.clear();
   lcd.setCursor(0,0);

     if (buttons & BUTTON_UP) {     // pressing the "UP" button will display "YES!! I LIKE
FUN!" and change background color.
        lcd.print("YES!!");
        lcd.setCursor(0,1);
        lcd.print("I LIKE FUN!");
        lcd.setBacklight(TEAL);
        selectval=1;
     }
      if (selectval==1 & buttons & BUTTON_SELECT) {  //if the UP button has been hit, and the
select button is pressed, it will display "GOOD CHOICE" and send reset signals to
        lcd.print("GOOD CHOICE!");                 //the PICs and the UNO. It will remain on this
screen until the game has ended.
        digitalWrite(STARTpin, HIGH);
        digitalWrite(UNORESETpin,LOW);
        digitalWrite(PICRESETpin,HIGH);
        delay(50);
        digitalWrite(UNORESETpin,HIGH);
        digitalWrite(PICRESETpin,LOW);
        screenopen=1;
          }
   }
   if (buttons & BUTTON_DOWN) {                  //If the "DOWN" button is pressed, the screen
will display "NO I'M BORING".
     lcd.setCursor(0,0);
     lcd.print("NO");
     lcd.setCursor(0,1);
     lcd.print("I'M BORING :(");
     lcd.setBacklight(GREEN);
     selectval=2;
   }
        if (selectval==2 & buttons & BUTTON_SELECT) {  //If the down button has been
selected and the select button is pressed, the screen will display "BORING" and
          lcd.print("BORING!");                    //take the user back to the start screen.
          lcd.setBacklight(VIOLET);
          delay(2000);
          selectval = 3;

         }
```

```
   if (buttons & BUTTON_LEFT) {                    //If the left or right buttons are pressed, it will
display "CLICK UP OR DOWN", letting the user know they have hit the wrong buttons.
     lcd.print("CLICK UP OR DOWN");
     lcd.setBacklight(GREEN);
     selectval=0;
   }
   if (buttons & BUTTON_RIGHT) {
     lcd.print("CLICK UP OR DOWN");
     lcd.setBacklight(TEAL);
     selectval=0;
   }
     if (selectval==0 & buttons & BUTTON_SELECT) {    //If the left/right button has been
slected and the select button is pressed, it will display "WRONG BUTTON!" and take the user
back
        lcd.print("WRONG BUTTON!");                 //to the start screen.
        lcd.setBacklight(VIOLET);
        delay(3000);
        selectval=3;

          }

   if(theend == HIGH & selectval==1) {              //If the MEGA receives a high signal from the
UNO and the game has been started, the screen will display "THANKS 4 PLAYIN'"
      lcd.setCursor(0,0);                          //and take the user back to the start screen. A new game
can now be started.
      lcd.print("THANKS 4 PlAYIN'");
      delay(6000);
      selectval = 3;
    }
}
//////////////////////////score program////////////////////////////
void Score()
{
  digitalWrite (Pin31, HIGH);    //sends a HIGH signal to the PICs to increase the score on the 7-
segment displays.
  delay (100);
  digitalWrite (Pin31, LOW);     //writes the pin low again after score signal has been sent.
    //SCORE MUSIC
} // END OF VOID SCORE

//////////////////////////////ALWAYS////////////////////////////////
void Always()
{
  repeat = digitalRead(ENDpin); //The repeat variable allows for a new game to be played after
one has ended. It is dependent on the signal from the UNO in the ENDpin.
```

```
  while(repeat==LOW) //This loop will run while there is no high signal from the UNO, meaning the game is not over.
  //for (int x = 0; x <= 5; x++)
  {
    int randomselect=random(1,7);  //sets up the random LED program to select a number between 1 and 6.
    int theend = digitalRead(ENDpin); //setting up the end variable for the end of game signal from the UNO
    if (theend == HIGH)
    {
      loop();
    }
       if (randomselect==1) //random program selects a "1", which lights up LED on Target 1.
    {
      digitalWrite (Pin49, LOW); //send LED1 low to turn on LED.
      for (int k=0; k<=3000; k++) {
        boolean z = digitalRead(Pin44); //setting up the switch for target 1
        delay(1);
        if (z== HIGH) //If target 1 is hit, switch 1 will go high.
        {
          Score(); //target 1 will send 5 pulses to PIC, giving the user 5 points for hitting this target.
          Score();
          Score();
          Score();
          Score();
          tessa(); //melody will play telling the user a target was hit
          digitalWrite(Pin49, HIGH); //turn off the LED if the target is hit
          break; //breaks prevent someone from getting extra points from the target swinging and hitting the switch again.
          break;
        }
      }
      digitalWrite(Pin49, HIGH); //turn off the LED even if the target was not hit.
    }
   if (randomselect==6) //If random program selects "6", lights up LED on target 2.
    {
      digitalWrite (Pin50, LOW); //turn on target 2 LED
      for (int k=0; k<=3000; k++) {
        boolean z = digitalRead(Pin45); //setting up switch for target 2
        delay(1);
        if (z== HIGH) //if target 2 is hit, switch 2 will go high.
        {
          Score(); //target 2 will send 2 pulses to PIC, giving the user 2 points.
          Score();
          tessa();//melody will play telling the user a target was hit
          digitalWrite(Pin50, HIGH); //turn off LED if the target is hit
```

```
        break; //breaks prevent user from getting extra points from the target swinging and hitting the switch again.
        break;
       }
      }
     digitalWrite(Pin50, HIGH); //turn off LED even if target was not hit
    }

    if (randomselect==3) //If random program selects "3", lights up LED on target 3.
    {
     digitalWrite (Pin51, LOW); //turns on target 3 LED
     for (int k=0; k<=3000; k++) {
      boolean z = digitalRead(Pin46); //sets up switch for target 3
      delay(1);
      if (z== HIGH) //if target 3 is hit, switch 3 will go high
      {
       Score(); //target 3 will send one pulse to PIC, giving the user 1 point
       tessa(); //melody will play telling the user a target was hit
       digitalWrite(Pin51, HIGH); //turn off LED if the target it hit
       break;
       break; //breaks prevent user from getting extra points from the target swinging and hitting the switch again.
       }
      }
     digitalWrite(Pin51, HIGH); //turn off LED even if target was not hit
    }

    if (randomselect==4) //If random program selects "4", activates target 4
    {
     digitalWrite (Pin52, LOW); //turn on LED on target 4
     for (int k=0; k<=3000; k++) {
      boolean z = digitalRead(Pin47); //set up switch on target 4
      delay(1);
      if ( z== HIGH) //if target 4 is hit, switch 4 will go high
      {
       Score(); //target 4 will send 4 pulses to PIC, giving user 4 points.
       Score();
       Score();
       Score();
       tessa(); //melody will play telling the user a target was hit
       digitalWrite(Pin52, HIGH); //if target was hit, turn off LED
       break;
       break; //breaks prevent user from getting extra points from the target swinging and hitting the switch again.
       }
      }
```

```
    digitalWrite(Pin52, HIGH); //turn off LED even if target was not hit.
   }

   if (randomselect==5) //If random progam selects "5", activate target 5.
   {
    digitalWrite (Pin53, LOW); //turns on LED on target 5
    for (int k=0; k<=3000; k++) {
     boolean z = digitalRead(Pin48); //set up target 5 switch
     delay(1);
     if ( z== HIGH) //if target 5 is hit, switch will go high
       {
       Score(); //target 5 will send 3 pulses to PIC, giving user 3 points.
       Score();
       Score();
       tessa(); //melody will play telling the user a target was hit
       digitalWrite(Pin53, HIGH); //if taget was hit, turn off LED.
       break; //breaks prevent user from getting extra points from the target swinging and hitting
the switch again.
       break;
       }
     }
     digitalWrite(Pin53, HIGH); //turn off LED even if target was not hit
   }
   repeat = digitalRead(ENDpin); //checking for end of game signal
  }
}// END OF VOID ALWAYS
//////////////////SOUND//////////////////////////////
void tessa()
{
 //pin44
   for (int thisNote = 0; thisNote < 8; thisNote++)
   {
   int noteDuration = 1000/noteDurations2[thisNote];
   tone(8, melody2[thisNote],noteDuration);
   int pauseBetweenNotes = noteDuration * 1.30;
   delay(pauseBetweenNotes);
   noTone(8); // stop the tone playing
      }
 }
```