Introduction

The problem that this design sought to fill was to fill the niche between toys like nerf, airsoft, and paintball. While all of these systems are relatively affordable to acquire and quite widespread, quality Laser tag has always remained out of reach. This is truly unfortunate as laser tag offers many advantages over these other technologies. There is also a large gap in the laser tag market between high quality but prohibitively expensive commercial systems and cheap but underwhelming ones. This design seeks to be that middle ground. With all of the advances in electronics since the last laser tag craze, perhaps successful consumer grade laser tag could finally become a viable product. Further this iteration of the design can be run on an Arduino which opens up many possibilities, namely the ability for the consumer to totally customize their experience with their own features and game types. Perhaps this could even be marketed as an educational product to teach students about creating devices like it. The design needs to be cheap, relatively simple, and high quality. It needs to have a long range, and easy ability to add new firmware features. The proof of concept will have two taggers but the technology should scale to many more.

Idea Development Process

The design of this project began with research into how laser tag has been previously done and what features I wanted to include. It would need to include sound effects, a screen, some kind of user control interface, and a flashlight. Quite a lot of research time went into understanding the various libraries that would be utilized especially Irlib2 which is quite complicated. Designs were all done using sketches on paper including circuit information and pin out documentation.

The design had multiple different possible solutions. The sound was initially done using a piezo speaker and tone but this was too underwhelming so the design incorporated a separate chip to play mp3 files from an sd card which was controlled with a serial interface. The choice between OLED and LCD screens were difficult but LCD matrix was used due to its affordability and simplicity. The tagger design itself went through a few design iterations but the final design was settled upon due to personal taste. The code was rewritten many times using differing techniques

but always including more if statement structures to allow the program to poll all of its sensors adequately.

Discussion

The final design incorporates the desired features . The taggers register and send hits, have team and tagger weapon type information in the sent signal to allow info to be sent over the shots, and it has a number pad with rudimentary menu system. It has a health, ammo, battery level, team, and tagger type LCD display. It runs off a supply of AA supplying 12 volts main power. It uses an Arduino Mega board for its extra serial and internal timers. This serves as the control center for the whole device. The actual shot signals are sent using an IR LED with a focusing lenses and received using an IR Receiver device. The lense and LED are housed in a PVC pipe. There are color LEDs that flash during firing. The components are driven using logic level mosfets on a handmade prototype board. There is a serial addressable DFPlayer mini device that plays mp3 files off an sd card to a small speaker. The IR LED and Vibration motor require 5 volts so a DC to DC converter was used to step down the 12 volts for them.

Conclusion

The design has a few successful features. The sound from the DFPlayer helps to bring the tagger to life. Without it the experience would be very underwhelming. Contrary to my initial suspicions it was far more important to the experience than an OLED screen would have been. It also adds another layer of customizability to the design because the sound files can all be swapped with a different theme by swapping an sd card. The flashlight feature driven with a mosfet switch control is extremely useful and convenient at night. The fact that code fuctions as complex as it's become is impressive to me. I have grown a lot in my engineering skills over this design process and semester.

The Design process was certainly not without difficulty. The addition of a logic gate to allow multiple IR Receivers was not functioning and was ultimately removed due to time constraints. The implemented solenoid force feedback drew to much power and interfered with the electronics and sound so it had to be removed. They were replaced with vibration motors. The

vibration motors have gone unused in the code because loose wires could get stuck in them. The cosmetic laser proved to be too strong to be eye safe and was omitted.

The Whole design would be given an overhaul with more time and resources. A new shell for the components with more room to be housed in and a more aesthetically pleasing design. Added complexity, features, and streamlining to the code. The implementation of a custom PCB for all the peripheral components. The implementation of things like voltage regulators and noise filters on the board as well to improve quality and reliability. A radio transceiver zigbee system to take this to a whole new level and truly rival a commercial system at an affordable price with user friendly design philosophy while remaining open to user customization.

Appendix A: Arduino Sketches

```
//----------------------------
//TRUE CODE
//-----------------------------------------------



#include <Wire.h> // For I2C

#include <EEPROM.h>


#include <LiquidCrystal_I2C.h>



#include <IRLib2.h>
#include <IRLibDecodeBase.h>
#include <IRLibGlobals.h>
#include <IRLibRecvBase.h>
#include <IRLibRecvLoop.h>



#include <Key.h>
#include <Keypad.h>

//DFPLAYER SHIT
#include <DFMiniMp3.h>

// implement a notification class,
// its member methods will get called
//
class Mp3Notify
{
  public:
    static void OnError(uint16_t errorCode)
    {
      // see DfMp3_Error for code meaning
      Serial.print("Com Error ");
      Serial.println(errorCode);
    }
    static void OnPlayFinished(uint16_t track)
    {
      Serial.print("Play finished for #");
      Serial.println(track);
    }
    static void OnCardOnline(uint16_t code)
    {
      Serial.println("Card online ");
    }
    static void OnCardInserted(uint16_t code)
    {
      Serial.println("Card inserted ");
    }
    static void OnCardRemoved(uint16_t code)
    {
      Serial.println("Card removed ");
    }
};

// instance a DFMiniMp3 object,
// defined with the above notification class and the hardware serial class
//
DFMiniMp3<HardwareSerial, Mp3Notify> mp3(Serial3);





IRrecvPCI myReceiver(2);
IRdecode myDecoder;
IRsend mySender;


//KeyBoard STUFF
int kb0 = 0;
int kb1 = 0;
```

```
int kb2 = 0;
int kb3 = 0;
int kb4 = 0;
int kb5 = 0;
int kb6 = 0;
int kb7 = 0;
int kb8 = 0;
int kb9 = 0;
int kbpound = 0;
int kbstar = 0;
int kba = 0;
int kbb = 0;
int kbc = 0;
int kbd = 0;


const byte ROWS = 4; // Four rows
const byte COLS = 4; // Four columns
// Define the Keymap
char keys[ROWS][COLS] = {
  {'1', '2', '3', 'A'},
  {'4', '5', '6', 'B'},
  {'7', '8', '9', 'C'},
  {'*', '0', '#', 'D'}
};
// Connect keypad ROW0, ROW1, ROW2 and ROW3 to these Arduino pins.
byte rowPins[ROWS] = { 39, 41, 43, 45 };
// Connect keypad COL0, COL1 and COL2 COL3 to these Arduino pins.
byte colPins[COLS] = { 47, 49, 51, 53 };
// Create the Keypad
Keypad kpd = Keypad( makeKeymap(keys), rowPins, colPins, ROWS, COLS );


#define ledpin 13

//LCDSTUFF
char array1[] = "array1";
LiquidCrystal_I2C lcd(0x27, 2, 1, 0, 4, 5, 6, 7, 3, POSITIVE);


byte sheild[8] = {  //CUSTOM CHARACTERS
  0b10001,
  0b00110,
  0b00100,
  0b01100,
  0b10001,
  0b00010,
  0b01110,
  0b11000
};

byte heartleft[] = {
  B00000,
  B00110,
  B01111,
  B01111,
  B01111,
  B00111,
  B00011,
  B00001
};

byte heartright[] = {
  B00000,
  B01100,
  B11110,
  B11110,
  B11110,
  B11100,
  B11000,
  B10000
};

byte ammogun1[8] = {
  B00000,
  B00010,
  B00110,
  B01110,
```

```
  B01110,
  B01110,
  B01110,
  B01110
};
byte ammogun2[8] = {
  B00000,
  B00000,
  B01001,
  B11011,
  B11011,
  B11011,
  B11011,
  B11011
};

byte ammogun3[8] = {
  0b00001,
  0b00011,
  0b00111,
  0b01111,
  0b11111,
  0b11111,
  0b11111,
  0b11111
};

byte ammogun4[8] = {
  0b00000,
  0b00100,
  0b01110,
  0b01110,
  0b01110,
  0b01110,
  0b00100,
  0b01010
};


byte checkers[] = {
  B00000,
  B00000,
  B00001,
  B00011,
  B00111,
  B01111,
  B11111,
  B11111
};






//CODE STUFF
unsigned long timestart;
int mag ;
int keyfreq = 800;
int k = 0;
int i = 1;
int flashstate = 0;
int reloaded = 1;
int reloadsound = 0;
unsigned long reloadbuttontime;
int triggerblockreload = 0;
int triggerblockreloading = 0;
int triggerblock = 0;
unsigned long triggerpull;
int gunshotsound = 0;
int healthlvl = 20;
int deathflag = 0;
int respawn = 0;
float batterylvl;

int lasthealth =  0;
int lastmag = 0;
int statechange = 1;
```

```
int startupDECIMAL = 0;

char key;

unsigned long laststandardscreen = 0;

int gunmagsize = 0;
int gun = 0;
int gundamage = 0;
int Menu = 0;
int reloadtime = 0;
int ROF = 0;
int dispmagsize = 0;
int dispmag = 0;

unsigned long reloadingtimer = 0;
unsigned long reloadingtimer2 = 0;

unsigned long strobetimestart;
unsigned long strobingstart;
int m = 0;

int team1vals[] = {1, 2, 3, 4};
int team2vals[] = {5, 6, 7, 8};
int myteamvals[] = {0, 0, 0, 0};
int otherteam1val[] = {0, 0, 0, 0};
int otherteam2val[] = {0, 0, 0, 0};

int team = 0;
int strobetimer = 0;
int strobestate = 0;

int semiautoindex=0;
//-----------------------------------------------------------------
int battprev=0;

unsigned long lastbatt=2;

unsigned long gun4chargeup;

unsigned long lastshot;
unsigned long lastgun2shot;


int triggerholddown = 0;

int gun4index = 0;
int gun4trueindex = 0;







//-----------------------------------------------------------------
int gun4fire = 0;


//Gun properties!1!!!!!!!!!1


int gun1damage=1;

int gun2damage=1;

int gun3damage=5;
int gun4damage=15;


void setup() {
  Serial.begin(9600);  //to debug

  mp3.begin(); //to DFPLAYER
  mp3.setVolume(0x25);
  mp3.playMp3FolderTrack(1);   // sd:/01/001.mp3 startup sound

  delay(500);
```

```
  //Pinmodes for mosfets and inputs ect
  //IRRECIEVER 2       IRSENDER 5
  pinMode(22, INPUT); //trigger
  pinMode(A0, INPUT); //batteryLVLsensor
  pinMode(26, INPUT); //Reload button
  pinMode(13, OUTPUT); //piezobuzzer
  pinMode(7, OUTPUT); //flashlight
  pinMode(4, OUTPUT); //LED muzzle flash
  pinMode(6, OUTPUT); //LED SESSY Mood lights
  pinMode(3, OUTPUT); //Vibrator
  pinMode(8, OUTPUT); //Solenoid

  // DONT FORGET TO ADD A RELOAD BUTTON CHRAS, FOR NOW ITS #

  digitalWrite(13, LOW);
  digitalWrite(7, LOW);

  digitalWrite(6, LOW);
  digitalWrite(4, LOW);
  digitalWrite(3, LOW);
  digitalWrite(8, LOW);

  if (EEPROM.read(1) == 1) {
    team = 1;

  }

  else if (EEPROM.read(1) == 2) {
    team = 2;
  }
  else {
    team = 1;


  }

//CHANGE THESE VALUES IN THE MENU SECTION AT THE END ASWELL
  gun = EEPROM.read(0);

  if (gun == 1) {
    gunmagsize = 15;
    gundamage = 1;
    reloadtime = 2000;
    ROF = 60;
  }
  if (gun == 2) {
    gunmagsize = 15;
    gundamage = 1;
    dispmagsize = 45;
    reloadtime = 2000;
    ROF = 60;
  }
  if (gun == 3) {
    gunmagsize = 5;
    gundamage = 5;
    reloadtime = 4000;
    ROF = 1200;
  }

  if (gun == 4) {
    gunmagsize = 2;
    gundamage = 15;
    reloadtime = 7000;
    ROF = 3500;
  }
  else {
    gun = 1;
    gunmagsize = 15;
    gundamage = 1;
    reloadtime = 2000;
    ROF = 60;
  }
```

```
mag = gunmagsize;




lcd.begin(20, 4);
lcd.clear();




//LCD STARTUP IMAGES------------------------------

i = 1;
while (i <= 5) {
  //lcd.clear();
  lcd.home(); // top left
  for (int i = 1; i <= 20; i++) {

    lcd.write(random(165, 222));
  }
  lcd.setCursor(0, 1); // bott left

  for (int i = 1; i <= 20; i++) {

    lcd.write(random(165, 222));
  }
  lcd.setCursor(0, 2); // bott left

  for (int i = 1; i <= 20; i++) {

    lcd.write(random(165, 222));
  }
  lcd.setCursor(0, 3); // bott left

  for (int i = 1; i <= 20; i++) {

    lcd.write(random(165, 222));
  }


  delay(100);

  i = i + 1;
}
i = 1;


while (i <= 4) {
  //lcd.clear();
  lcd.home(); // top left
  for (int i = 1; i <= 20; i++) {

    lcd.write(random(165, 222));
  }
  lcd.setCursor(0, 1); // bott left

  for (int i = 1; i <= 20; i++) {

    lcd.write(random(165, 222));
  }
  lcd.setCursor(0, 2); // bott left

  for (int i = 1; i <= 20; i++) {

    lcd.write(random(165, 222));
  }
  lcd.setCursor(0, 3); // bott left

  for (int i = 1; i <= 20; i++) {
```

```
      lcd.write(random(165, 222));
    }


    delay(180);

    i = i + 1;
  }
  i = 1;



  i = 1;
  while (i <= 3) {
    //lcd.clear();
    lcd.home(); // top left
    for (int i = 1; i <= 20; i++) {

      lcd.write(random(165, 222));
    }
    lcd.setCursor(0, 1); // bott left

    for (int i = 1; i <= 20; i++) {

      lcd.write(random(165, 222));
    }
    lcd.setCursor(0, 2); // bott left

    for (int i = 1; i <= 20; i++) {

      lcd.write(random(165, 222));
    }
    lcd.setCursor(0, 3); // bott left

    for (int i = 1; i <= 20; i++) {

      lcd.write(random(165, 222));
    }


    delay(230);

    i = i + 1;
  }
  i = 1;
  //lcd.clear();


  lcd.clear();
  lcd.setCursor(0, 1);
  lcd.print("   Greetings User.");
  lcd.setCursor(0, 2);
  lcd.print("");

  delay(1000);
  lcd.clear();
  lcd.home(); // top left


  mp3.playMp3FolderTrack(2);


  tone(13, 1000, 200);
  //END STARTUP
  lcd.clear();


  lcd.createChar(0, heartleft);
  lcd.createChar(1, heartright);
  lcd.createChar(2, ammogun1);
  lcd.createChar(3, ammogun2);
  lcd.createChar(4, ammogun3);
  lcd.createChar(5, ammogun4);
lcd.createChar(6, checkers);
  Serial.println("past starting up serial comm up");

  myReceiver.enableIRIn();
```

```
  timestart = millis();

}

void loop() {


  //mp3.loop();









  // Keyboard code----------------------------------------------------------------------
  key = kpd.getKey();
  if (key) // Check for a valid key.
  {
    switch (key)
    {
      case '0':


         mp3.playMp3FolderTrack(61);

        break;
      case '1':
         mp3.playMp3FolderTrack(62);

        break;
      case '2':
         mp3.playMp3FolderTrack(63);
        break;
      case '3':
         mp3.playMp3FolderTrack(64);
        break;
      case '4':
        mp3.playMp3FolderTrack(65);
        break;
      case '5':
        mp3.playMp3FolderTrack(66);
        break;
      case '6':
        mp3.playMp3FolderTrack(67);
        break;
      case '7':
         mp3.playMp3FolderTrack(68);
        break;
      case '8':
        mp3.playMp3FolderTrack(69);
        break;
      case '9':
        mp3.playMp3FolderTrack(70);
        break;
      case '*':
        //standard key beep
        strobetimer = strobetimer + 1;
        strobestate = 0;
        strobetimestart = millis();
        if (flashstate == 0) {
          //flashlight on
          digitalWrite(7, HIGH);
          flashstate = 1;
          statechange=1;
          k = 0;
          mp3.playMp3FolderTrack(2);
        }

        if (flashstate == 1 && k == 1) {
          //flashlight off
          digitalWrite(7, LOW);
          flashstate = 0;
          statechange=1;
          mp3.playMp3FolderTrack(2);
```

```
      }
      tone(13, keyfreq, 75);
      break;
    case '#': //reload button for NOW!!!
      mp3.playMp3FolderTrack(2);
      Menu = 1;


      break;
    case 'A':

      //Enter settings
      //standard key beep
      Menu = 1;

      break;
    case 'B':
      tone(13, keyfreq, 75); //standard key beep

      break;
    case 'C':
      tone(13, keyfreq, 75); //standard key beep

      break;
    case 'D':
      tone(13, keyfreq, 75); //standard key beep

      break;

    default:
      Serial.println(key);
  }
}
k = 1;







//Semi Auto trigger code

if (digitalRead(22) == HIGH) {
  if (triggerblockreload == 1 && gun!=4 && gun!=3) {
    mp3.playMp3FolderTrack(14);




  }


  if (triggerblockreload == 0) {


    if (triggerblockreloading == 0) {
      if (triggerblock == 0) {

        //mp3.playMp3FolderTrack(3);


        if (gun == 1  && semiautoindex==0) {

          if (team == 1) {
            mySender.send(NEC, 0x1, 56);
            Serial.println("shot signal of val 1 for blaster 1");
          }
          else if (team == 2) {
            mySender.send(NEC, 0x5, 56);
            Serial.println("shot signal of val 5 for blaster 1");
          }
          semiautoindex=1;
          myReceiver.enableIRIn();
          mp3.playMp3FolderTrack(5);
          lastshot=millis();
```

```
            digitalWrite(4, HIGH); //muzzle flash

            //digitalWrite(3, HIGH); //vibration

            triggerpull = millis();
            triggerblock = 1;
            gunshotsound = 1;
            mag = mag - 1;

          }

          else if (gun == 2 && semiautoindex==0) {
            if (team == 1) {
              mySender.send(NEC, 0x2, 56);
            }
            else if (team == 2) {
              mySender.send(NEC, 0x6, 56);
            }
            myReceiver.enableIRIn();
            mp3.playMp3FolderTrack(6);
            lastshot=millis();
            digitalWrite(4, HIGH); //muzzle flash

            //digitalWrite(3, HIGH); //vibration
semiautoindex=1;
            triggerpull = millis();
            triggerblock = 1;
            gunshotsound = 1;
            mag = mag - 1;
            dispmag=dispmag-3;


          }

          else if (gun == 3 && semiautoindex==0) {
            if (team == 1) {
              mySender.send(NEC, 0x3, 56);
            }
            else if (team == 2) {
              mySender.send(NEC, 0x7, 56);
            }
            myReceiver.enableIRIn();
            mp3.playMp3FolderTrack(7);
            semiautoindex=1;
            digitalWrite(4, HIGH); //muzzle flash

            //digitalWrite(3, HIGH); //vibration
lastshot=millis();
            triggerpull = millis();
            triggerblock = 1;
            gunshotsound = 1;
            mag = mag - 1;

          }
          else if (gun == 4 && gun4fire == 1) {
            if (team == 1) {
              mySender.send(NEC, 0x4, 56);
            }
            else if (team == 2) {
              mySender.send(NEC, 0x8, 56);
            }
            myReceiver.enableIRIn();
            mp3.playMp3FolderTrack(8);
            semiautoindex=1;
            gun4index = 0;
            gun4trueindex = 0;
            gun4fire = 0;
            digitalWrite(4, HIGH); //muzzle flash

            //digitalWrite(3, HIGH); //vibration

            triggerpull = millis();
            triggerblock = 1;
            gunshotsound = 1;
            mag = mag - 1;

          }
```

```
        }
      }
    }
  }


if(digitalRead(22)==LOW && millis()-lastshot>=ROF){
  semiautoindex=0;


}



hitcheck ();




  //Weird Stuff for guns 4




  //gun 4

  if (gun == 4) {
    if (digitalRead(22) == HIGH) {
      if (triggerblockreload == 1) {
        //mp3.playMp3FolderTrack(12);

      }


      if (triggerblockreload == 0) {


        if (triggerblockreloading == 0) {
          if (triggerblock == 0) {

            triggerholddown = 1;

            gun4index = 1;

            if (gun4trueindex == 0) {
              mp3.playMp3FolderTrack(9);
              gun4trueindex = 1;
              gun4chargeup = millis();
            }

            if (millis() - gun4chargeup >= 2900 && gun4index == 1) {
              gun4fire = 1;

            }




          }
        }
      }
    }



    if (digitalRead(22) == LOW) {

      triggerholddown = 0;

      if (gun4index == 1) {
        mp3.playMp3FolderTrack(10);
        gun4index = 0;
```

```
      gun4trueindex = 0;

    }

  }


}









//RELOAD PART

if (digitalRead(26) == HIGH) {
  mp3.playMp3FolderTrack(15);
  reloaded = 1;

  reloadsound = 1;
  reloadbuttontime = millis();
  triggerblockreload = 0;
  triggerblockreloading = 1;
  mag = gunmagsize;
  dispmag=dispmagsize;
}




//TIMER IF STATEMENTS------------------------------------------------------ =_\'Q'/_=

//timer trigger block
if (millis() - triggerpull > ROF) {
  triggerblock = 0;

}

//Muzzle flash and add laser part when it arrives timer turn em off
if (millis() - triggerpull > 600) {
  digitalWrite(4, LOW);
}

//gunshot sound play timer  simple
if (millis() - triggerpull < 230) {
  tone(13, random(300, 1500), 5);
}

//Solenoid Timer

if (millis() - triggerpull > 60) {
  digitalWrite(8, LOW);
}


//Vibration Timer!!!!!!!!!!!!!!!!!!!!!!!!!

/*
  if (gun == 1) {
    if (millis() - triggerpull > 60) {
      digitalWrite(3, LOW);
    }
  }
  if (gun == 2) {
    if (millis() - triggerpull > 60) {
      digitalWrite(3, LOW);
    }
  }
  if (gun == 3) {
```

```
      if (millis() - triggerpull > 80) {
        digitalWrite(3, LOW);
      }
    }

    if (gun == 4) {
      if (millis() - triggerpull > 90) {
        digitalWrite(3, LOW);
      }
    }

  */







  /*
    //sets a delay on the reload allowing the trigger to fire
    if(millis()-reloadbuttontime>150){
    triggerblockreloading=0;

    }
  */



  //ammunition counter to stop firing when mag = 0
  if (mag == 0) {
    triggerblockreload = 1;


  }









  //IMPORTANT LCD DISPLAY UPDATERS BELOW ------------------------------------------------------------
-  =_\'Q'/_=
  //---------
  //SETTINGS SCREEN CODE adding later empty for now...
  //---------

  //STANDARD, has health? ammo-------------------------------------------------------------------------
-------------------------------------------------------------------------
  //-------------------_______-------------------------------------------------------------------------
-------------------------------------------------------------------------
  //-------------------_______-------------------------------------------------------------------------
-------------------------------------------------------------------------
  //-------------------_______-------------------------------------------------------------------------
-------------------------------------------------------------------------
  //-------------------_______-------------------------------------------------------------------------
-------------------------------------------------------------------------
  //-------------------_______-------------------------------------------------------------------------
-------------------------------------------------------------------------
  //-------------------_______-------------------------------------------------------------------------
-------------------------------------------------------------------------


  //LIGHT DISPLAY
if(flashstate==1 && statechange==1 ){
lcd.setCursor(0, 0); // bott left

lcd.print("*");
```

```
}

if(flashstate==0 && statechange==1){
lcd.setCursor(0, 0); // bott left

lcd.print(" ");


}

//battery level
batterylvl = analogRead(A0);

if(millis()-lastbatt>=600 && batterylvl-battprev!=0 || statechange==1){

lcd.setCursor(6, 0); // bott left
lcd.print("Battery ");
lcd.setCursor(14, 0);
lcd.print("      ");
lcd.setCursor(14, 0);
lcd.print(batterylvl);
battprev=batterylvl;
lastbatt=millis();

}






  //TEAM DISPLAY
if(team==1 && statechange==1){
lcd.setCursor(0, 1); // bott left

lcd.print("Team 1");


}

if(team==2 && statechange==1){
lcd.setCursor(0, 1); // bott left

lcd.print("Team 2");


}

//Gun Display
if(gun==1 && statechange==1){
lcd.setCursor(8, 1); // bott left

lcd.print("Blaster");


}

if(gun==2 &&statechange==1){
lcd.setCursor(8,1); // bott left

lcd.print("Chain Gun");


}
if(gun==3 && statechange==1){
lcd.setCursor(8,1); // bott left

lcd.print("Bolt Rifle");


}

if(gun==4 && statechange==1){
lcd.setCursor(8,1); // bott left

lcd.print("Rail Gun");
```

```
}


if(statechange==1){
lcd.setCursor(0,3); // bott left
for (int i = 1; i <= 20; i++) {

      lcd.write(byte(6));
    }

}




  //Health display
  if (healthlvl - lasthealth != 0 || statechange == 1 ) {

    lcd.setCursor(0, 2); // bott left

    lcd.write(byte(0));
    lcd.write(byte(1));
    lcd.print(" ");
    lcd.print("  ");
    lcd.setCursor(3, 2);
    lcd.print(healthlvl, DEC);




    lasthealth = healthlvl;


  }



  //Ammo display
  if (mag - lastmag != 0 || statechange == 1 ) {

    if (gun == 1) {
      lcd.setCursor(11, 2);
      lcd.write(byte(2));
      lcd.print("        ");
      lcd.setCursor(13, 2);
      lcd.print(mag, DEC);

    }

    else if (gun == 2) {
      lcd.setCursor(11, 2);
      lcd.write(byte(3));
      lcd.print("        ");
      lcd.setCursor(13, 2);
      lcd.print(dispmag, DEC);
    }

    else if (gun == 3) {
      lcd.setCursor(11, 2);
      lcd.write(byte(4));
      lcd.print("        ");
      lcd.setCursor(13, 2);
      lcd.print(mag, DEC);

    }


    else if (gun == 4) {
      lcd.setCursor(11, 2);
      lcd.write(byte(5));
      lcd.print("        ");
      lcd.setCursor(13, 2);
      lcd.print(mag, DEC);

    }
```

```
    lastmag = mag;

  }

  //

  statechange = 0;


  //-------------------------------------------------------------------------------------------------------------
-------------------------------------------------------------------
  //-------------------------------------------------------------------------------------------------------------
-------------------------------------------------------------------
  //-------------------------------------------------------------------------------------------------------------
-------------------------------------------------------------------
  //-------------------------------------------------------------------------------------------------------------
-------------------------------------------------------------------
  //-------------------------------------------------------------------------------------------------------------
-------------------------------------------------------------------
  //-------------------------------------------------------------------------------------------------------------
-------------------------------------------------------------------
  //-------------------------------------------------------------------------------------------------------------
-------------------------------------------------------------------




  //DEATH SCREEEEEEN BIIIIIIIITCH(pauses code)
  if (healthlvl <= 0) {
    digitalWrite(4, LOW);
    digitalWrite(3, LOW);
    digitalWrite(8, LOW);
    mp3.playMp3FolderTrack(13);

    key = 'E';
    while (healthlvl == 0) {

      //lcd.clear();
      lcd.home(); // top left
      lcd.print("DEAD==DEAD==DEAD==DE");
      lcd.setCursor(0, 1); // bott left
      lcd.print("AD==DEAD==DEAD==DEAD");
      lcd.setCursor(0,2); // bott left
      lcd.print("DEAD==DEAD==DEAD==DE");
      lcd.setCursor(0, 3); // bott left
      lcd.print("AD==DEAD==DEAD==DEAD");


      tone(13, 300, 25);
      key = kpd.getKey();
      delay(400);
      //lcd.clear();
      lcd.home(); // top left
      lcd.print("=DEAD==DEAD==DEAD==D");
      lcd.setCursor(0, 1); // bott left
      lcd.print("D==DEAD==DEAD==DEAD=");
      lcd.setCursor(0, 2); // bott left
lcd.print("=DEAD==DEAD==DEAD==D");
      lcd.setCursor(0, 3); // bott left
      lcd.print("D==DEAD==DEAD==DEAD=");



      tone(13, 300, 25);
      key = kpd.getKey();
      delay(400);
      //lcd.clear();
      lcd.home(); // top left
      lcd.print("==DEAD==DEAD==DEAD==");
      lcd.setCursor(0, 1); // bott left
      lcd.print("==DEAD==DEAD==DEAD==");
      lcd.setCursor(0, 2); // bott left
      lcd.print("==DEAD==DEAD==DEAD==");
      lcd.setCursor(0, 3); // bott left
      lcd.print("==DEAD==DEAD==DEAD==");
      tone(13, 300, 25);
      key = kpd.getKey();
      delay(400);
```

```
      //lcd.clear();
      lcd.home(); // top left
      lcd.print("D==DEAD==DEAD==DEAD=");
      lcd.setCursor(0, 1); // bott left
      lcd.print("=DEAD==DEAD==DEAD==D");
      lcd.setCursor(0, 2);
lcd.print("D==DEAD==DEAD==DEAD=");
      lcd.setCursor(0,3); // bott left
      lcd.print("=DEAD==DEAD==DEAD==D");


      tone(13, 300, 25);
      key = kpd.getKey();
      delay(400);
      //lcd.clear();
      lcd.home(); // top left
      lcd.print("ED==DEAD==DEAD==DEAD");
      lcd.setCursor(0, 1); // bott left
      lcd.print("DEAD==DEAD==DEAD==DE");
      lcd.setCursor(0, 2);
      lcd.print("ED==DEAD==DEAD==DEAD");
      lcd.setCursor(0, 3); // bott left
      lcd.print("DEAD==DEAD==DEAD==DE");
      tone(13, 300, 25);
      key = kpd.getKey();
      delay(400);
      //lcd.clear();
      lcd.home(); // top left
      lcd.print("EAD==DEAD==DEAD==DEA");
      lcd.setCursor(0, 1); // bott left
      lcd.print("EAD==DEAD==DEAD==DEA");
       lcd.setCursor(0, 2);
      lcd.print("EAD==DEAD==DEAD==DEA");
      lcd.setCursor(0, 3); // bott left
      lcd.print("EAD==DEAD==DEAD==DEA");
      tone(13, 300, 25);
      delay(400);
      //lcd.clear();

      respawn = 1;
      healthlvl = 20;

    }


  }



  //RESPAWNING SCREEN(pauses code)
  if (respawn == 1) {
    i = 1;
    while (i <= 4) {

      //lcd.clear();
      lcd.home(); // top left
      lcd.print(".....RESPAWNING....");
      lcd.setCursor(0, 1); // bott left
      lcd.print("-~-~-~-~-~-~-~-~-~-~-");
      lcd.setCursor(0, 2); // bott left
      lcd.print("-~-~-~-~-~-~-~-~-~-~-");
      lcd.setCursor(0, 3); // bott left
      lcd.print("-~-~-~-~-~-~-~-~-~-~-");
      tone(13, 600, 25);
      delay(350);
      //lcd.clear();
      lcd.home(); // top left
      lcd.print(".....__________....");
      lcd.setCursor(0, 1); // bott left
      lcd.print("-~-~-~-~-~-~-~-~-~-~-");
      lcd.setCursor(0, 2); // bott left
      lcd.print("-~-~-~-~-~-~-~-~-~-~-");
      lcd.setCursor(0, 3); // bott left
      lcd.print("-~-~-~-~-~-~-~-~-~-~-");
      tone(13, 900, 25);
      delay(350);

      i = i + 1;
```

```
  }
  i = 1;
  lcd.clear();

  myReceiver.enableIRIn();
  healthlvl = 20;
  mag = gunmagsize;
  dispmag = dispmagsize;
  respawn = 0;
  statechange = 1;
}


//LOW BATTERY SCREEN WARNING(pauses code) add later using separate test code
batterylvl = analogRead(A0);


//BATTERY LEVEL DISPLAY add later using separate test code


//RELOADING SCREEN pauses code so health gets recieved here also

if (triggerblockreloading == 1) {

  lcd.clear();
  reloadingtimer = millis();
  reloadingtimer2 = millis();
  mp3.playMp3FolderTrack(15);
  int h = 1;

  while (millis() - reloadingtimer <= reloadtime) {

    hitcheck ();







    if (millis() - reloadingtimer2 >= 400) {
      lcd.home(); // top left
      lcd.print("......RELOADING.....");
      lcd.setCursor(0, 1); // bott left
      lcd.print("~-~-~-~-~-~-~-~-~-~-~-~-~-~-");
      lcd.setCursor(0, 2); // bott left
      lcd.print("~-~-~-~-~-~-~-~-~-~-~-~-~-~-");
      lcd.setCursor(0, 3); // bott left
      lcd.print("~-~-~-~-~-~-~-~-~-~-~-~-~-~-");
      reloadingtimer2 = millis();
    }



  }
  mp3.playMp3FolderTrack(16);
  triggerblockreloading = 0;

  mag = gunmagsize;
  lcd.clear();
  statechange = 1;
}









// MENU CODE PAUSES CODE

while (Menu == 1) {
  statechange = 1;
```

```
    lcd.clear();
    lcd.setCursor(0, 0); // bott left
    lcd.print("MAIN MENU");
    lcd.setCursor(0, 1); // bott left
    lcd.print("B: Team Select");
    lcd.setCursor(0, 2); // bott left
    lcd.print("C: Tagger Select");
    lcd.setCursor(0, 3); // bott left
    lcd.print("D: Settings");

    m = 0;

    while (m == 0) {
      k = 1;
      key = kpd.getKey();
      statechange = 1;

      if (key) // Check for a valid key.
      {
        switch (key)
        {

          case '*':
            //standard key beep
            if (flashstate == 0) {
              //flashlight on
              digitalWrite(7, HIGH);
              flashstate = 1;
              k = 0;
              mp3.playMp3FolderTrack(2);
            }

            if (flashstate == 1 && k == 1) {
              //flashlight off
              digitalWrite(7, LOW);
              flashstate = 0;
              mp3.playMp3FolderTrack(2);
            }
            tone(13, keyfreq, 75);
            break;


          case '#': //reload button for NOW!!!
            mp3.playMp3FolderTrack(2);
            Menu = 0;
            m = 1;
            lcd.clear();




            break;
  case '9':
mp3.playMp3FolderTrack(70);
  break;
             case '8':
mp3.playMp3FolderTrack(69);

break;


          case 'B':
            mp3.playMp3FolderTrack(2);

            lcd.clear();
            lcd.setCursor(0, 0); // bott left
            lcd.print("TEAM SELECT");
            lcd.setCursor(0, 1); // bott left
            lcd.print("1: Team 1");
            lcd.setCursor(0, 2); // bott left
            lcd.print("2: Team 2");
            lcd.setCursor(0, 3); // bott left
            lcd.print("  ");

            while (m == 0) {
              key = kpd.getKey();
              if (key) // Check for a valid key.
```

```
    {
      switch (key)
      {

        case '1':
          mp3.playMp3FolderTrack(2);
          EEPROM.write(1, 1);
          team = 1;
          m = 1;
          statechange = 1;







          break;
        case '2':
          mp3.playMp3FolderTrack(2);

          EEPROM.write(1, 2);
          team = 2;

          m = 1;
          statechange = 1;
          break;

        case '*':
          //standard key beep
          statechange = 1;
          if (flashstate == 0) {
            //flashlight on
            digitalWrite(7, HIGH);
            flashstate = 1;
            statechange=1;
            k = 0;
            mp3.playMp3FolderTrack(2);
          }

          if (flashstate == 1 && k == 1) {
            //flashlight off
            digitalWrite(7, LOW);
            flashstate = 0;
            statechange=1;
            mp3.playMp3FolderTrack(2);
          }
          tone(13, keyfreq, 75);
          break;
        case '#':
          mp3.playMp3FolderTrack(2);
          m = 1;
          break;

        default:
          Serial.println(key);
      } k = 1;
    }
  }



  break;
case 'C':
  mp3.playMp3FolderTrack(2);

  lcd.clear();
  lcd.setCursor(0, 0); // bott left
  lcd.print("TAGGER SELECT");
  lcd.setCursor(0, 1); // bott left
  lcd.print("1:Blaster 2:ChainGun");
  lcd.setCursor(0, 2); // bott left
  lcd.print("3:BoltRifle ");
  lcd.setCursor(0, 3); // bott left
  lcd.print("4:RailGun  ");

  while (m == 0) {
    key = kpd.getKey();
    if (key) // Check for a valid key.
```

```
{
  switch (key)
  {

    case '1':
      mp3.playMp3FolderTrack(4);
      EEPROM.write(0, 1);
      gunmagsize = 15;
      mag = gunmagsize;
      gundamage = 1;
      reloadtime = 2000;
      ROF = 60;
      gun = 1;

      m = 1;
      statechange = 1;







      break;
    case '2':
      mp3.playMp3FolderTrack(4);

      EEPROM.write(0, 2);
      gunmagsize = 15;
      mag = gunmagsize;

      gundamage = 1;
      dispmagsize = 45;
      dispmag=dispmagsize;
      reloadtime = 2000;
      ROF = 60;
      m = 1;
      gun = 2;
      statechange = 1;
      break;
    case '3':
      mp3.playMp3FolderTrack(4);
      EEPROM.write(0, 3);
      gunmagsize = 5;
      mag = gunmagsize;
      gundamage = 5;
      reloadtime = 4000;
      ROF = 1200;
      gun = 3;
      m = 1;
      statechange = 1;







      break;
    case '4':
      mp3.playMp3FolderTrack(4);
      EEPROM.write(0, 4);
      gunmagsize = 2;
      mag = gunmagsize;
      gundamage = 15;
      reloadtime = 7000;
      ROF = 3500;
      gun = 4;
      m = 1;
      statechange = 1;







      break;

    case '*':
      //standard key beep
```

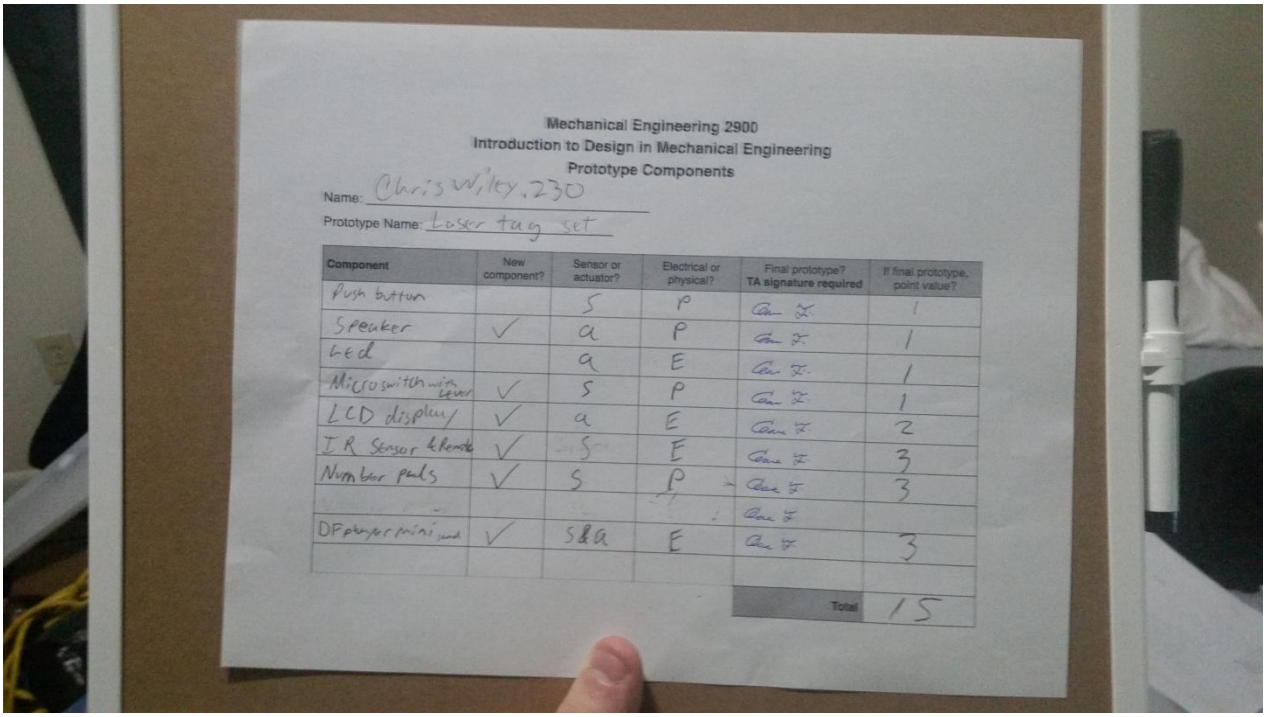
```
                    statechange = 1;
                    if (flashstate == 0) {
                      //flashlight on
                      digitalWrite(7, HIGH);
                      flashstate = 1;
                      statechange=1;
                      k = 0;
                      mp3.playMp3FolderTrack(2);
                    }

                    if (flashstate == 1 && k == 1) {
                      //flashlight off
                      digitalWrite(7, LOW);
                      flashstate = 0;
                      statechange=1;
                      mp3.playMp3FolderTrack(2);
                    }
                    tone(13, keyfreq, 75);
                    break;
                  case '#':
                    mp3.playMp3FolderTrack(2);
                    m = 1;
                    break;

                  default:
                    Serial.println(key);

                }

              }
            }




            break;
          case 'D':
            mp3.playMp3FolderTrack(2);
            lcd.clear();
            lcd.setCursor(0, 0); // bott left
            lcd.print("SETTINGS");
            lcd.setCursor(0, 1); // bott left
            lcd.print("TBD");
            lcd.setCursor(0, 2); // bott left
            lcd.print("");
            lcd.setCursor(0, 3); // bott left
            lcd.print("");
            delay(600);

            break;

          default:
            Serial.println(key);
        }
      }







    }


  }
}



//USER DEFINED FUNCTIONS____-----__-_---__-___--------------------______---------------__-_____----
-----_-______-----_-------------------------------------------------------

float mapf(float x, float in_min, float in_max, float out_min, float out_max)
{
  return (x - in_min) * (out_max - out_min) / (in_max - in_min) + out_min;
```

```
}



void hitcheck (){
 if (myReceiver.getResults()) {
    myDecoder.decode();           //Decode it
    Serial.println("Entered Hit Check and value is");
    Serial.println(myDecoder.value);

    if (team == 1) {
      Serial.println("Team is 1");
if(myDecoder.value==0){
  Serial.println("That was a miss val == 0");
}

else if(myDecoder.value==5){
   mp3.playMp3FolderTrack(14);
  Serial.println("HIT BLASTER 1");
  healthlvl=healthlvl-gun1damage;
}
else if(myDecoder.value==6){
   mp3.playMp3FolderTrack(14);
  Serial.println("HIT CHAINGUN 2");
  healthlvl=healthlvl-gun2damage;
}
else if(myDecoder.value==7){
   mp3.playMp3FolderTrack(14);
  Serial.println("HIT BOLTRIFLE 3");
  healthlvl=healthlvl-gun3damage;
}
else if(myDecoder.value==8){
   mp3.playMp3FolderTrack(14);
  Serial.println("HIT RAILGUN 4");
  healthlvl=healthlvl-gun4damage;
}

else{
  Serial.println("went through the extra else statement");
  }

myReceiver.enableIRIn();
Serial.print("NEW HEALTH =  ");
Serial.println(healthlvl);
    }


else if (team == 2) {
      Serial.println("Team is 1");
if(myDecoder.value==0){
  Serial.println("That was a miss val == 0");
}

else if(myDecoder.value==1){
   mp3.playMp3FolderTrack(14);
  Serial.println("HIT BLASTER 1");
  healthlvl=healthlvl-gun1damage;
}
else if(myDecoder.value==2){
   mp3.playMp3FolderTrack(14);
  Serial.println("HIT CHAINGUN 2");
  healthlvl=healthlvl-gun2damage;
}
else if(myDecoder.value==3){
   mp3.playMp3FolderTrack(14);
  Serial.println("HIT BOLTRIFLE 3");
  healthlvl=healthlvl-gun3damage;
}
else if(myDecoder.value==4){
   mp3.playMp3FolderTrack(14);
  Serial.println("HIT RAILGUN 4");
  healthlvl=healthlvl-gun4damage;
}

else{
  Serial.println("went through the extra else statement");
  }
```

```
myReceiver.enableIRIn();
Serial.print("NEW HEALTH =  ");
Serial.println(healthlvl);
    }









 }







}


void hitchecktest(){
if (myReceiver.getResults()) {
    myDecoder.decode();           //Decode it

    Serial.println(myDecoder.value);
    if(team==1){
    if(myDecoder.value!=0 && myDecoder.value!=2){
healthlvl=healthlvl-1;
Serial.println("Got Shot!");
   Serial.println(healthlvl);
    }

    }

    myReceiver.enableIRIn();

}
}







//-----------------------------
//TEST CODE
//-----------------------------------------------

#include <IRLib2.h>
#include <IRLibDecodeBase.h>
#include <IRLibGlobals.h>
#include <IRLibRecvBase.h>
#include <IRLibRecvLoop.h>

IRrecvPCI myReceiver(2);
IRdecode myDecoder;
IRsend mySender;


void setup() {
 Serial.begin(9600);
Serial.println("Starting");
pinMode(22,INPUT);

myReceiver.enableIRIn();




}

void loop() {
```

```
  if(digitalRead(22)==HIGH){
Serial.println("Sending");
mySender.send(NEC, 0x5, 56);
myReceiver.enableIRIn();


  }



 if (myReceiver.getResults()) {
    myDecoder.decode();           //Decode it
    Serial.println(myDecoder.value);
    myReceiver.enableIRIn();

 }



}
```

IMAGES:

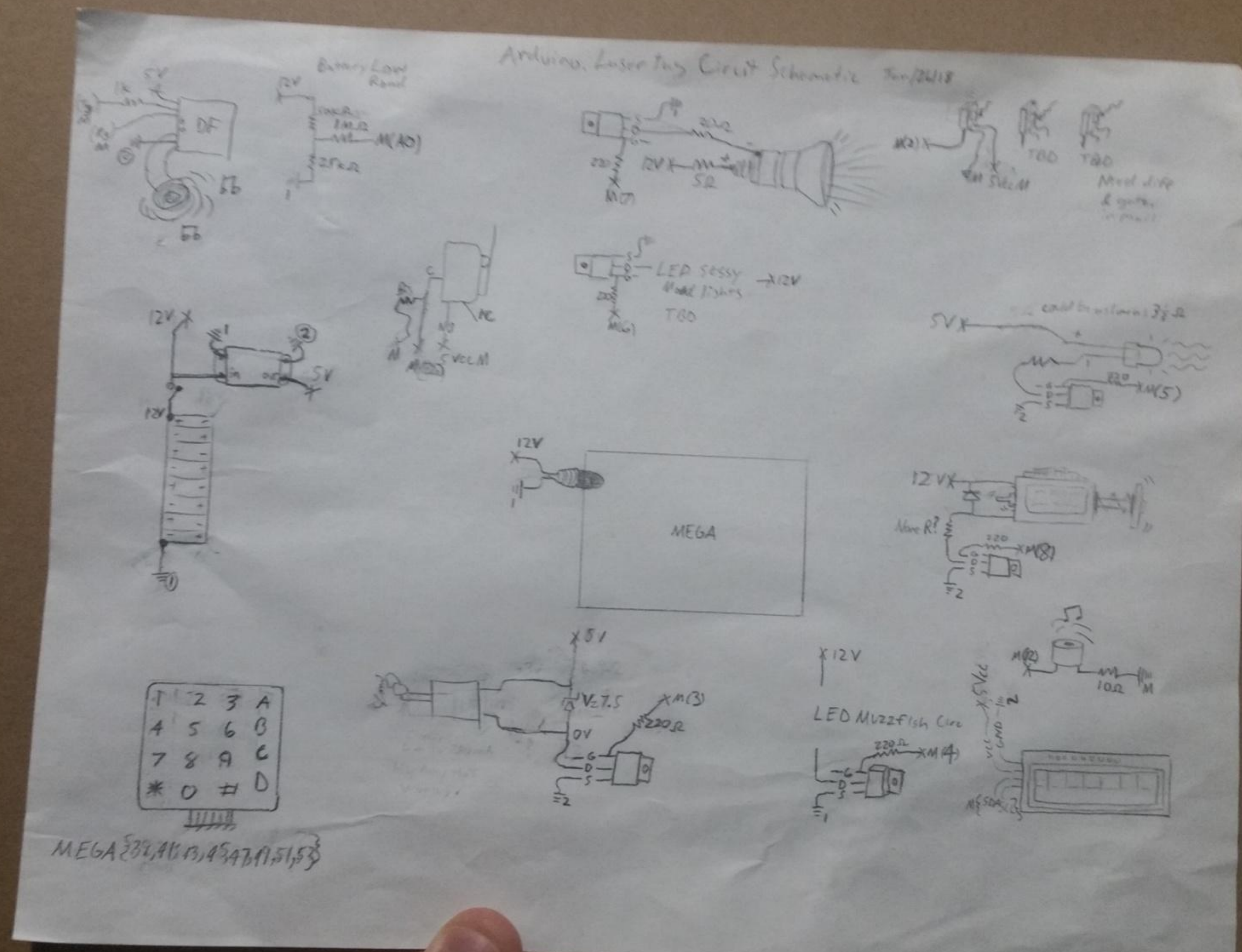


Mechanical Engineering 2900
Introduction to Design in Mechanical Engineering
Prototype Components

Name: Chris Wiley, 230

Prototype Name: Laser tag set

Component	New component?	Sensor or actuator?	Electrical or physical?	Final prototype? TA signature required	If final prototype, point value?
Push button		S	P		1
Speaker	✓	a	P		1
Led		a	E		1
Microswitch with lever	✓	S	P		1
LCD display	✓	a	E		2
IR Sensor & Remote	✓	S	E		3
Number pads	✓	S	P		3
DFplayer mini and	✓	S&a	E		3
				Total	15

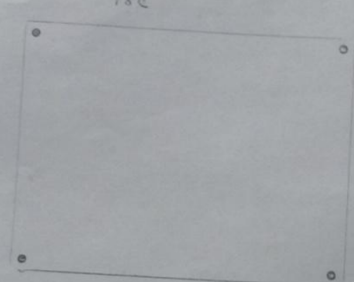


eprom 0 = gun slot
1 = team
2 = motor enabled

Code notes/needs:
IRLib send from 3 pin 5
DFplayer 10, 11

TBC

x6 NMOS

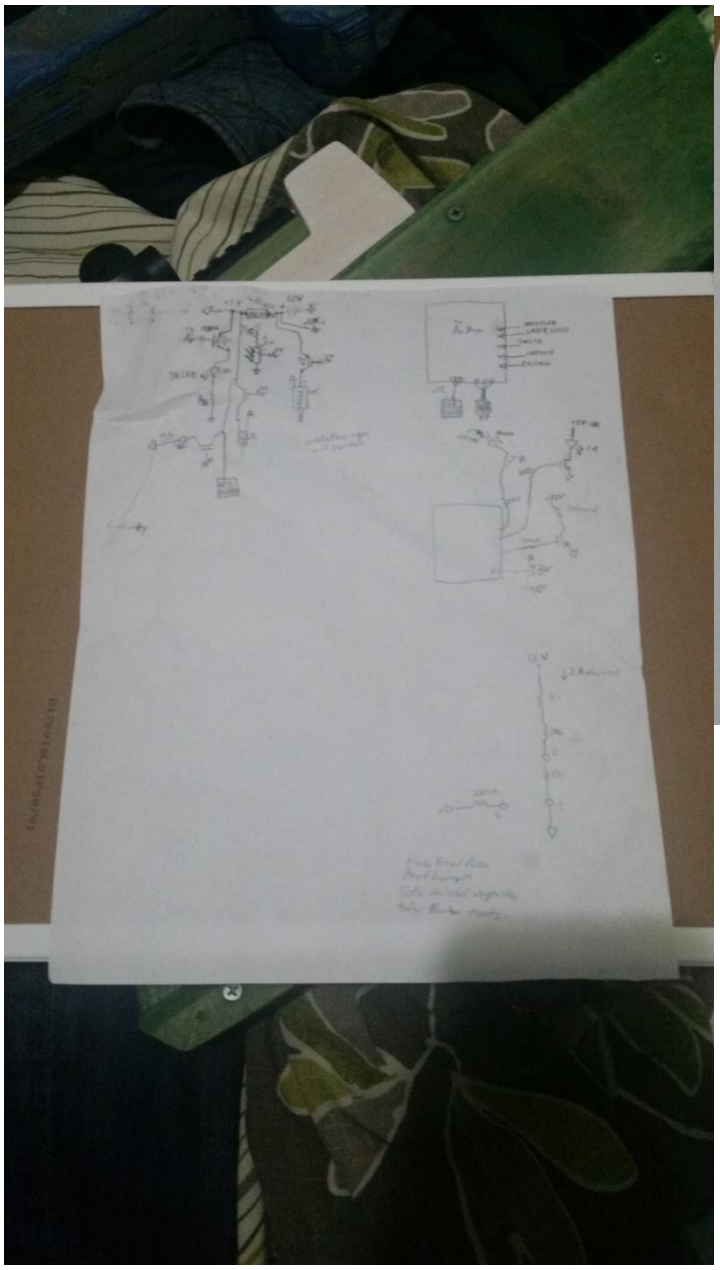


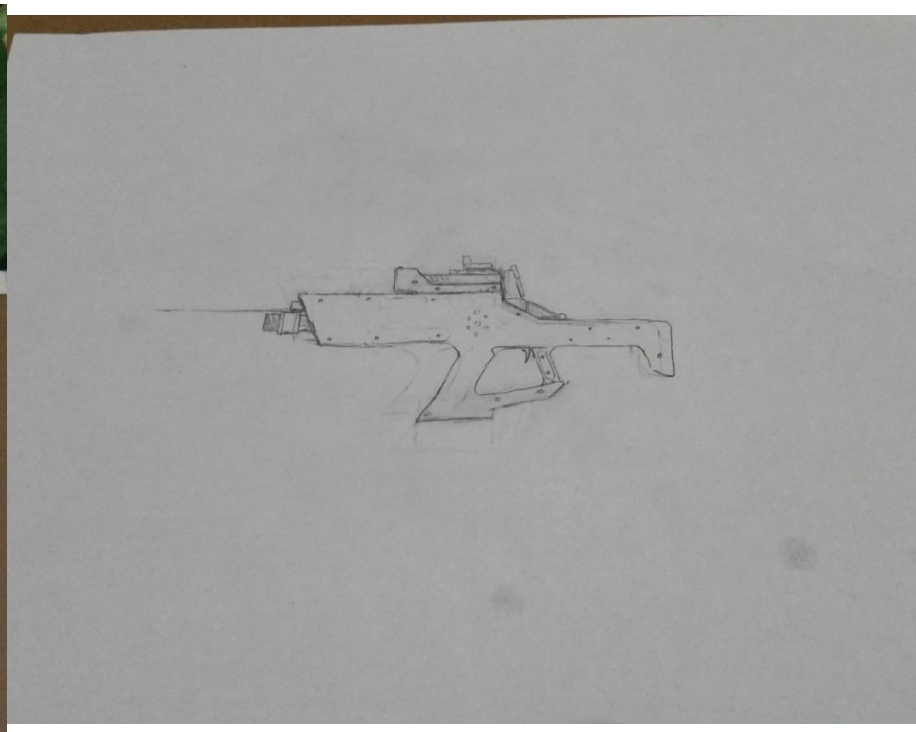
MEGA Pins
5 IR mos Sender
2 IR Rec
22 Trigger
13 piezo buzzer
A0 Battery sensor.
7 Flashlight
4 LEDMuzz
6 LED Sessy
3 VibMotor
8 Solenoid

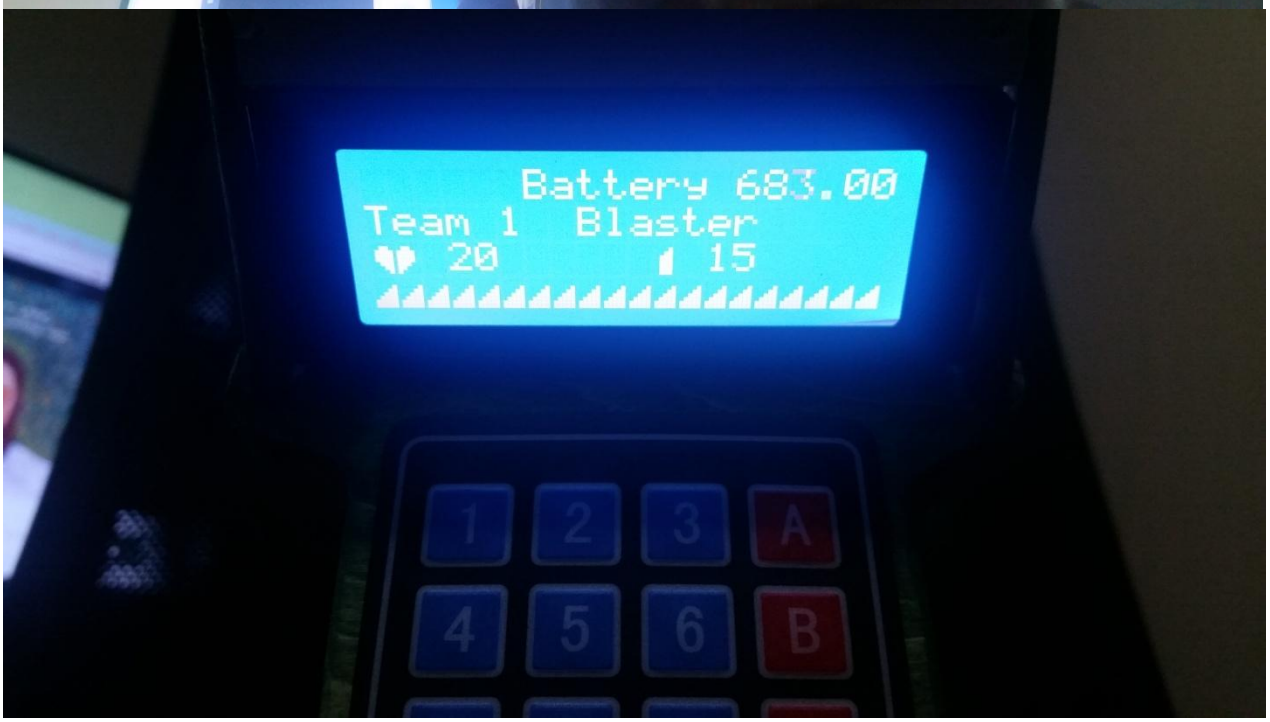
Sound map
0001.mp3 Startup
2 menu1
3 menu2
4 menu3
5 Fire gun1 "Standard"
6 Fire gun 2 "fullauto"
7 Fire gun3 "Sniper"
8 Fire gun4 "charge"
9 Charge gun4
10 decharge gun4
11 F fail 1
12 ? fail 2
13 Fail 3
14 Fail 4
15 Reload 1
16 Reload 2

dec
1 gun 1
2 gun 2 Team 1
3 gun 3
4 gun4

5 gun1
6 2 Team 2
7 3
8 4






Battery 683.00
Team 1 Blaster
20 15
1 2 3 A
4 5 6 B