Open Sourcing: Metallophone for a Severely Disabled Child

USU Senior Design
Jacob Jewkes
Chet Pettingill
Beto Salas
Jaden Scott

The purpose of this document is to provide clear instructions and examples of how to construct a metallophone that can be played by a severely disabled child. The team listed above designed and tested this system to ensure it can play the metallophone according to the requirements set by the team sponsor My Breath My Music. It is assumed at this point that you have received access to all of the part files listed to complete the design.

The Base Structure

The following materials were used to create the Base:

- 8ft x 4ft Maple Plywood
    - 4 L-Shaped Pieces
    - 2 Solenoid Boards
    - 8 Clamp Blocks
    - 1 Base Board
- Wood Glue
- Wood Stain
- Wood Screws
- 2 Bar Stands (3D Printed)
- One 15 Tonebell Metallophone

Assembly

First, the Plywood needs to be cut into the separated wooden pieces described in the Drawing Files.

Then the duplicate pieces should be glued together. There should now be 2 L-Shaped pieces, 1 Solenoid Board, 4 Clamp Blocks, and 1 Base Board.

The wood pieces should then be sanded until smooth and stained.

After the staining is complete, the two L-Shaped pieces can be attached to the Solenoid Board. Use at least two 3 in. wood screws to attach each L piece to the Solenoid Board. Be sure that the longer LED gap appears on the left-hand side of the Solenoid Board when right side up. This ensures the larger side of the metallophone will appear on the left.

Longer LED Gap

Stagger the screws so there is a screw entering both wood pieces of the glued Solenoid Board. If necessary, use more screws at this connection to ensure the pieces have a tight connection.

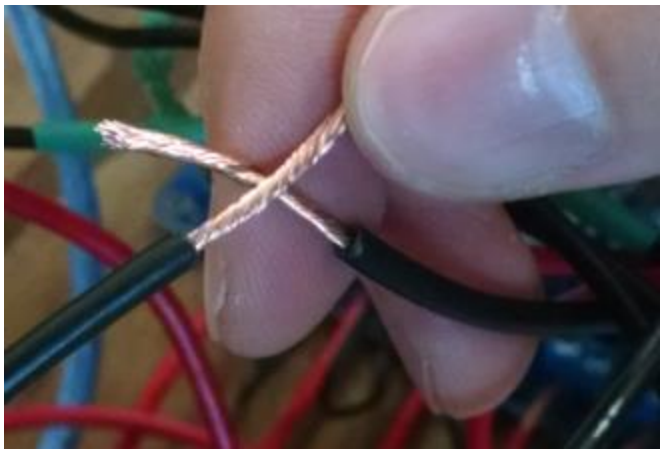
Solenoid Housing and Placement

The following materials will be needed for the solenoid housing and placement.

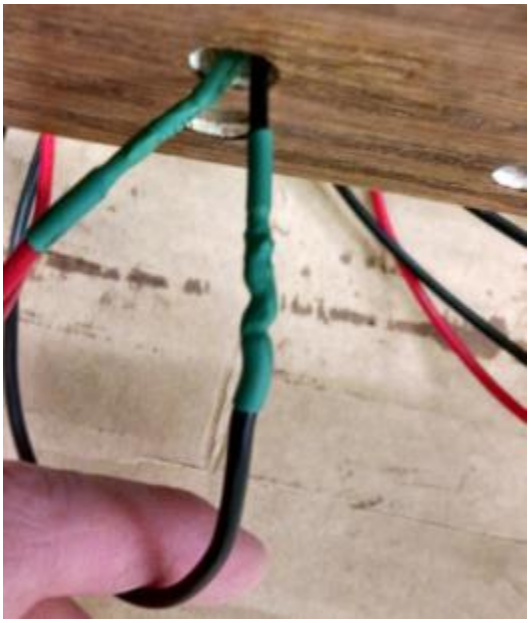
- 15 solenoids
- 5 solenoid plates (3-D printed piece)
- 60 solenoid fastener screw (ensure that these can fasten into the rear end of the solenoids)
- Red 16 mm wire
- Black 16 mm wire
- Heat shrink
- Felt fabric
- Soldering Equipment

Assembly

First, the wires coming off the end of the solenoid need to be lengthened. There should be two wires on the end of each solenoid. Cut 15 red and 15 black wires to approximately 30 cm in length. Using the soldering equipment, solder a red wire and a black wire onto the ends of the wires on the solenoid. It does not matter which wire is red or black at this point, but that distinction will become important later on.



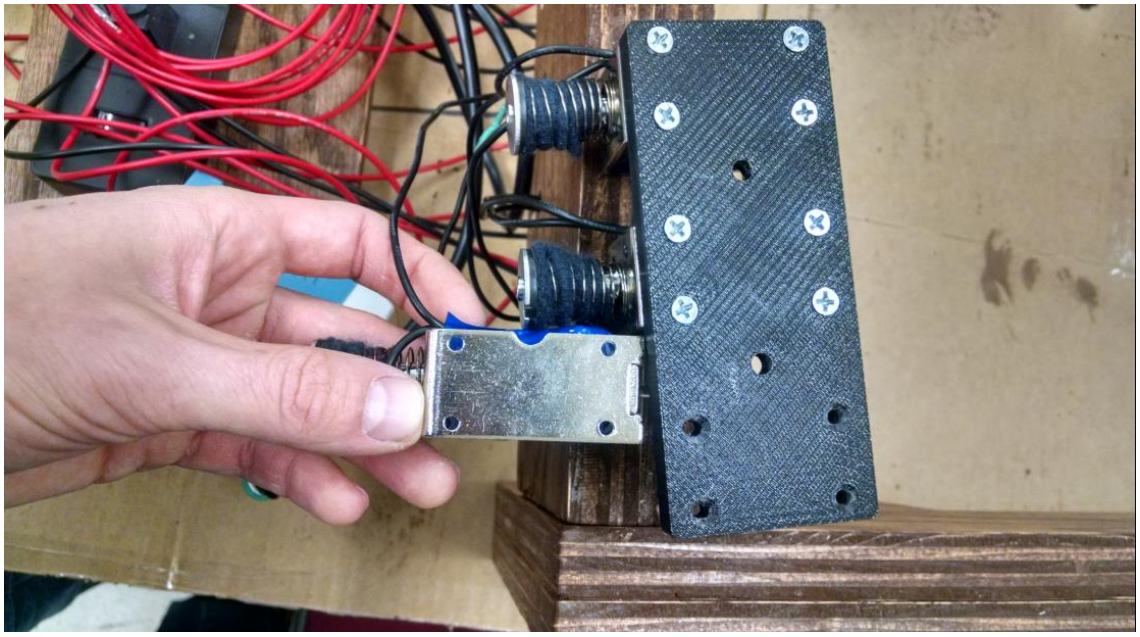
Slide a piece of heat shrink over the exposed wire connection and, using a heat gun, melt the heat shrink over the exposed wire.



Take the felt fabric, cut and sew it into 15 individual socks of slight larger diameter than the solenoid bar where the spring rests on the lower end of the solenoid. A simple cross stitch is fine.

Next, remove the O-ring and spacer on the tip of the solenoid bar (You will be putting these back on so do not lose them) and slide the bar out of the solenoid. Remove the spring from the bar and place the felt sock over the bar. Then replace the spring and adjust it so the sock is between the bar and the spring. This will help dampen the sound from the solenoid. Place the bar back in the solenoid and replace the spacer and the O-ring.

The Solenoids are now ready to be placed on the Solenoid Plate. Place four screws through the plate and screw them into the screw slots on the back of the solenoid. Place three solenoids on each plate.



The plates can now be placed on the base structure. Place the solenoids plate level with the top of the base and align the plate so the solenoid bars are directly above the holes in the base structure. Place two 1.5 in. screws through the solenoid plate into the base structure.

Electronic System

The following materials will be required for this section.

- 1 Arduino Mega
- 1 Arduino Mega screw terminal board
- 1 PCB Prototyping Board (18x24 Hole)
- 2 8-Channel Solid-State Relay Boards

- 2 220 Ω Resistors
- MIDI Pin Connection Piece
- 16 mm Red Wire
- 16 mm Black Wire
- Jumper Wires
- Wire Pin Connections
- 2 8-Position 1-Pole Rotary Switches
- 1 Potentiometer
- 4 1/8 in. Audio jacks
- Soldering Equipment
- 19 7 in. Male to Female Wires
- 1 24 x 18 pin Printed Circuit Board (PCB)
- Heat-Shrink
- 1 Adafruit LED Strip
- 12 V Direct Current Converter
- Disposable Extension Cord

IMPORTANT: This section is highly detailed and must be done correctly or the system will not work. Be sure to keep track of all wire connections and double check each step is completed before testing the metallophone. A table is provided at the end of the section listing all of the Arduino pin connections and the PCB connections.

Individual Part Wiring

Intro: Below are instructions about how to wire and connect all of the electrical components. The following tables specify where each wire/pin needs to go. 12V Positive, 12V Ground, and 5V Positive location designations have to do with the PCB board. More information is found in the PCB board section below.

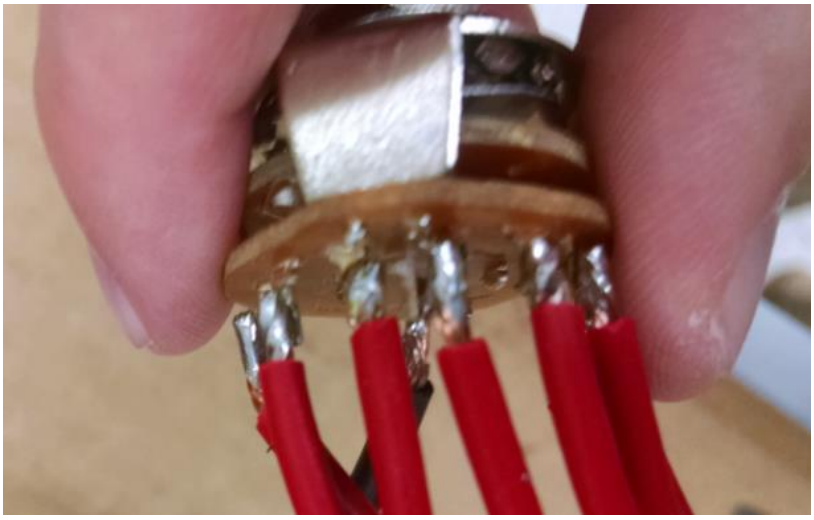
Solenoid: Connect the red wire to the right port of the relay, and the black wire to 12V ground. Repeat for all 15 solenoids.

	Location
Red Wire	Right port of the relay
Black Wire	12V Ground

Potentiometer: The potentiometer has three pin connections: one ground, one 5V supply, and one that connects to the metallophone. Solder a 20 cm wire onto the end of the pins. Any color arrangement is fine, just keep track of which wire does what.

	Location
Red Wire	5V Positive
Black Wire (Middle)	12V Ground
Black Wire (Left)	Arduino Pin A14

Rotary Switch: The rotary switches require nine pin connections: one for each rotary position, and one for 5V supply. Solder a 20 cm black wire onto the center 5V supply pin, and solder a 20 cm wire onto each pin of the switch. (we chose to only use red wire for these pins but a different color combination can be used if desired)



Rotary Switch (1)

	Location
Position 1	Arduino Pin 13
Position 2	Arduino Pin 12
Position 3	Arduino Pin 11
Position 4	Arduino Pin 10
Position 5	Arduino Pin 9
Position 6	Arduino Pin 8
Position 7	Arduino Pin 7
Position 8	Arduino Pin 6
Power Pin	5V Positive

Rotary Switch (2)

	Location
Position 1	Arduino Pin 53
Position 2	Arduino Pin 51
Position 3	Arduino Pin 49
Position 4	Arduino Pin 47
Position 5	Arduino Pin 45
Position 6	Arduino Pin 43
Position 7	Arduino Pin 41
Position 8	Arduino Pin 39
Power Pin	5V Positive

Audio Jack: The audio jacks have three pins but only two are used in this setup. Looking at the audio jack so that the center pin is on the bottom, solder a 20 cm wire onto the middle pin and the right-side pin. The right-side wire will be connected to 5V, the center wire will be connected to the arduino.

Audio Jack Plug (1)

	Location
Red Wire	Arduino Pin 37
Black Wire	5V Positive

Audio Jack Plug (2)

	Location
Red Wire	Arduino Pin 35
Black Wire	5V Positive

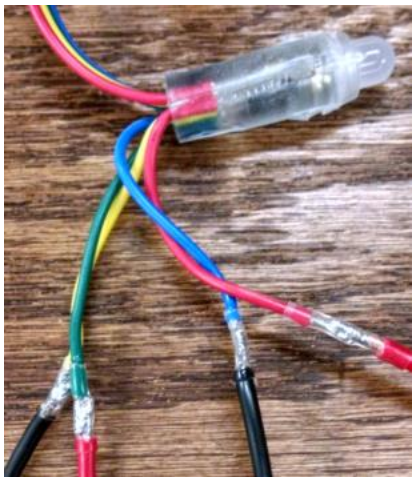
Audio Jack Plug (3)

	Location
Red Wire	Arduino Pin 33
Black Wire	5V Positive

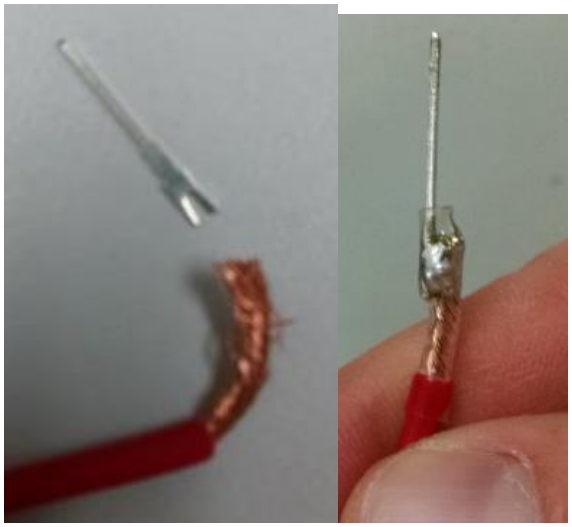
Audio Jack Plug (4)

	Location
Red Wire	Arduino Pin 31
Black Wire	5V Positive

LEDs: Solder a 40 cm wire onto the end of each of the four LED wires. Melt heat shrink over the exposed wire connections.



Solder a metal pin onto the opposite end of each of the wires on the potentiometer, rotary switches, and the audio jacks. Also solder pins onto the end of the red and green LED wire. Place heat shrink over the exposed pin connection but be sure to leave a portion of the pin exposed for later connections.



Pull both of the solenoid wires through the first hole directly beneath the solenoid. Solder a pin onto the black wire of each solenoid. The red wire does not require a pin.

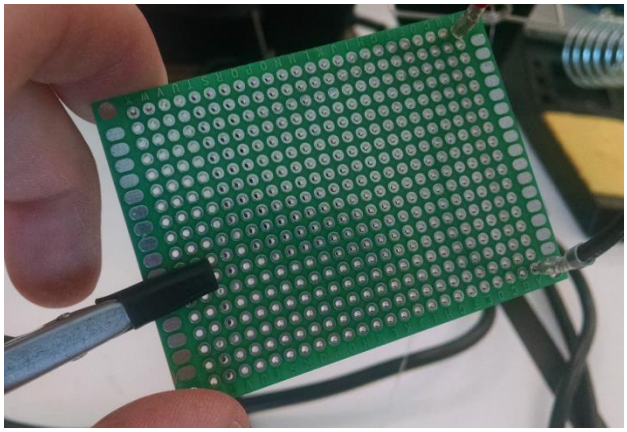
Screw a 45 cm black wire in the V- pin of the 12 V power supply. Screw a 45 cm red wire in the V+ pin of the 12 V power supply. Solder a metal pin onto the end of each of those wires. Melt heat shrink over the exposed section.

Cut the head of the extension cord off (not the part that connects to the wall) to expose the three interior wires. Screw those three interior wires into the back end of the 12 V power supply. The green wire goes into the ground port. The black wire goes into the L (Live) port. The white wire goes into the N (Neutral) port.

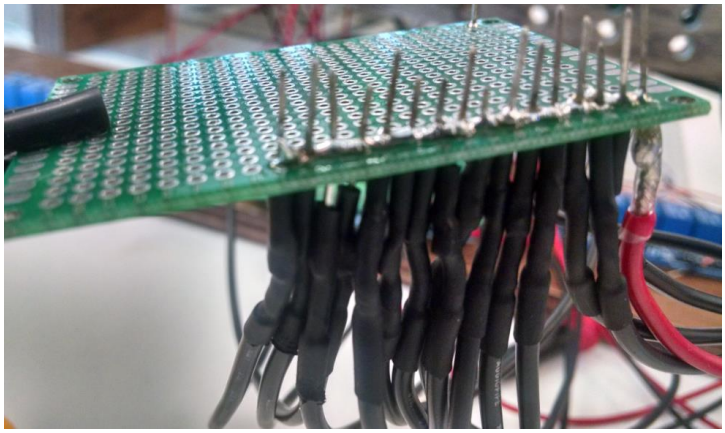
PCB Wiring

Setup the PCB to begin connecting wires to it. The top row of the PCB will be used for the 12 V wire connections. The middle row will be used for the 5 V wire connections. The bottom row will

be used for the ground connections. The left side of the PCB will also be used to connect the MIDI device.



Begin by placing the 12 V wire connection (the red V+ wire from the 12 V supply) through the bottom right pin. Next place all of the black solenoid wires into the same row connections, side by side, with the first one next to the red V+ 12 V wire.



As noted in the above image, the pins should first be soldered onto the PCB and then a line of solder should connect all of the pins on the bottom of the PCB. This allows the 12 V current to run directly to the solenoids. After that line of solder is complete, the ends of the pins can be snipped just below the solder line.

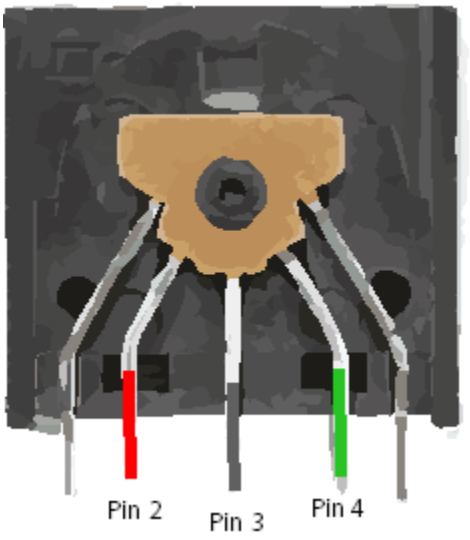
Next, complete the 5V line of pins in the middle row, right side of the PCB. Use the tables above to know what all goes in the 5V line.

After completing the 5V pins, refer to the tables and complete the Ground pins on the bottom row, right side of the PCB.

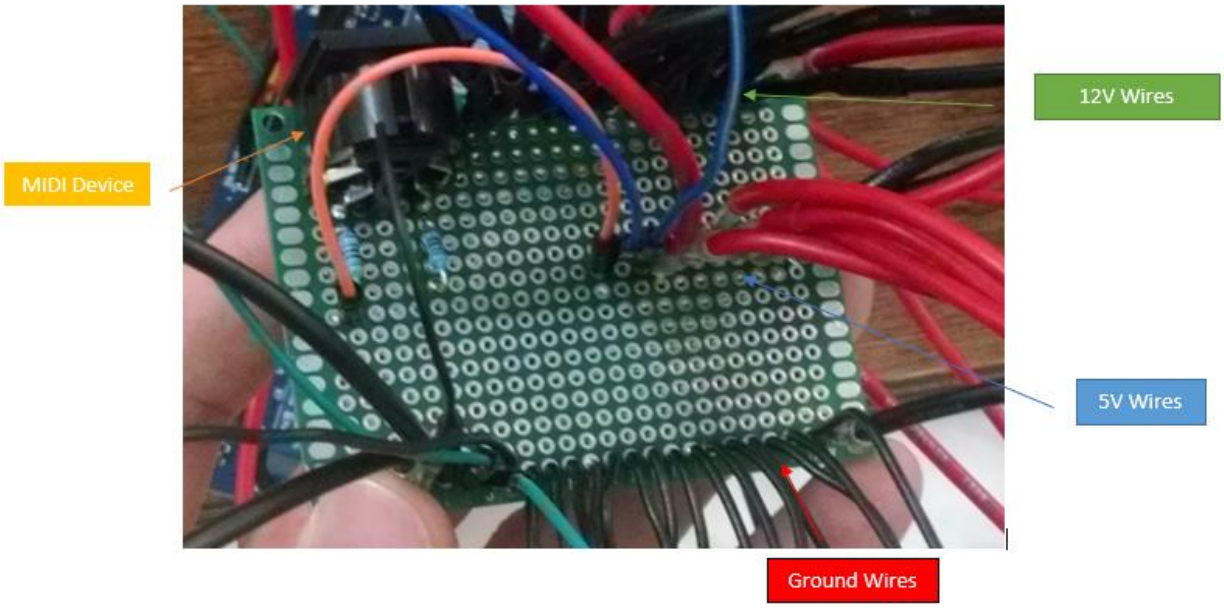
Finally connect the MIDI device to the left end of the PCB. The MIDI device has 3 pins that are significant on the back side.

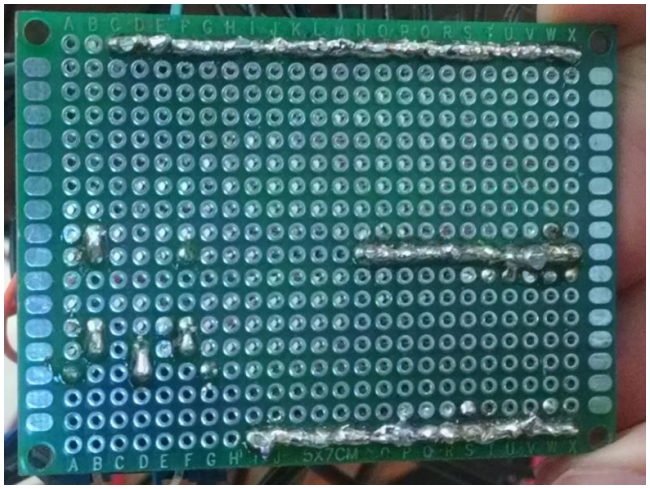
Connect pin 2 to a 220 ohm resistor and then to 5V Positive

Connect pin 3 to 12V Ground
Connect pin 4 to a 220 ohm resistor and then to the TX2 pin on the Arduino



After soldering on the MIDI device, double check all of your connections to ensure everything is laid out as directed in the PCB table. The final design should look like something like the following images.





Arduino Wiring
Place the screw terminal protection over the Arduino Mega. This will ensure that once pins are screwed in place, they will stay where they should be.

Next align the following parts on the base board as shown. From left to right, one 8 Channel Solid-State Relay with the relays on the left side, the Arduino mega, the PCB, and then the other 8 Channel Solid-State Relay but this one should have the relays on the right.

Begin by placing the Red solenoid wires onto every third screw terminal of the solid-state relays. See the solenoid table above to know what pin each wire connects to. Since there are only 15 solenoids, one relay will not have anything screwed into it. (that will be the top relay on the left side of the Arduino) Next the ground relay wires should be screwed into the middle relay terminals. (The top relay should still be completely empty)

Now the Male to Female jumper wires should be placed on the pins on the opposing side of the relay boards. Since there are only 15 solenoids, the relay with nothing plugged into it will not need a wire connected to it.

Refer to the two relay tables below to know what pin goes where. Begin with the solenoid male to female wires you just placed. All the remaining solenoid, rotary switch, potentiometer, LED, audio jack, MIDI, ground, and voltage wires will be screwed into the Arduino.

Relay (1)

	Location
In1	Arduino Pin 22
In2	Arduino Pin 24
In3	Arduino Pin 26
In4	Arduino Pin 28
In5	Arduino Pin 30
In6	Arduino Pin 32
In7	Arduino Pin 34

In8	Arduino Pin 36
VCC	5V Arduino
GND	12V Ground
R-VCC	12V Positive

Relay (2)

	Location
In1	Arduino Pin 38
In2	Arduino Pin 40
In3	Arduino Pin 42
In4	Arduino Pin 44
In5	Arduino Pin 46
In6	Arduino Pin 48
In7	Arduino Pin 50
VCC	5V Arduino
GND	12V Ground
R-VCC	12V Positive

The rotary switches will be designated either the scale functions, or the switchensemble functions depending on which you put where. Be sure to put the wire from the rotary switches from right to left order onto the Arduino board.

Keep track of which audio jack you put in which location. The audio jacks will be the switch connections, and one will always function as the first switch and so on.

Double check that all pins are in the right locations. Congratulations! You have now completed the electrical part of the setup.

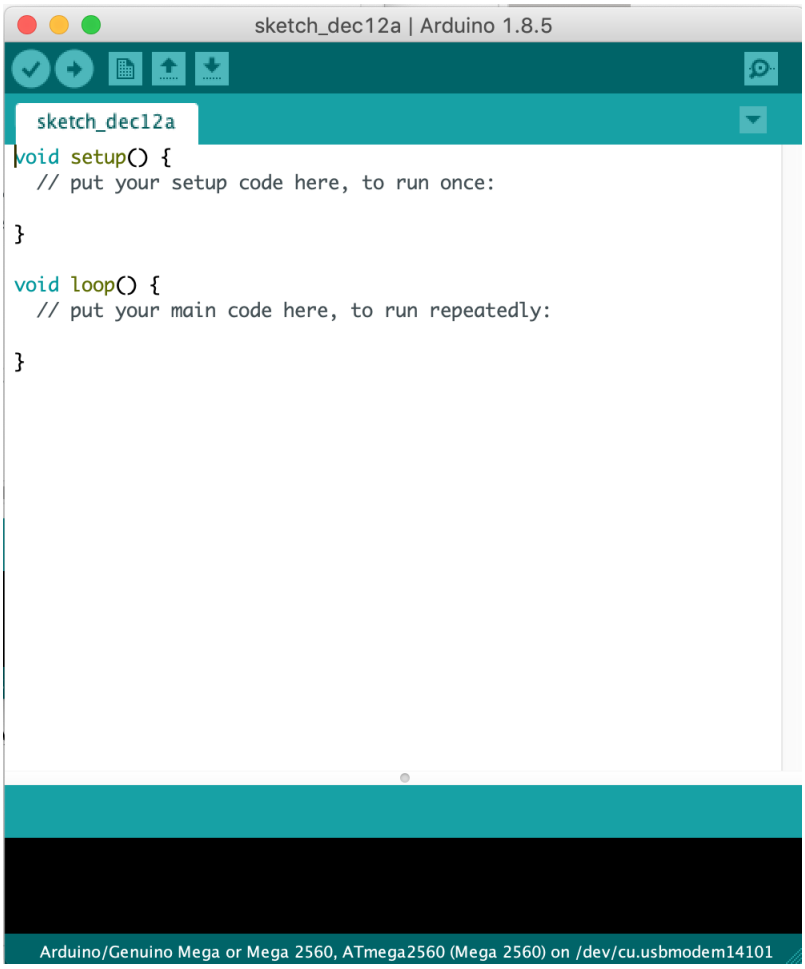
Arduino Code

Code Body

Copy the code found at the end of this document and paste it to an empty Arduino sketch. If you have not used an Arduino before, please go to the following website and download the Arduino software for your appropriate operating system.

https://www.arduino.cc/en/Main/Software

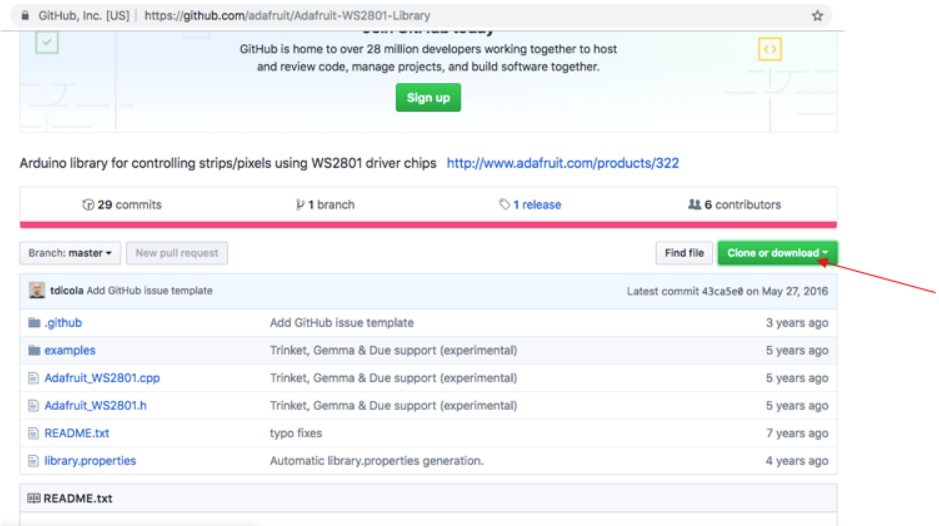
Open the Arduino software. After some initial setup, you should see a screen that looks like this.



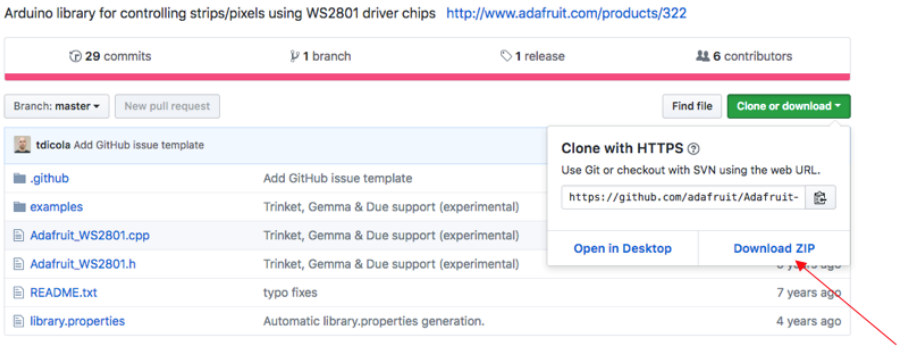
Delete everything inside of the sketch so that you are left with only a white workspace. This is where you will paste the code from the bottom of this document.

It is now necessary to upload the appropriate libraries that pertain to the LED's.

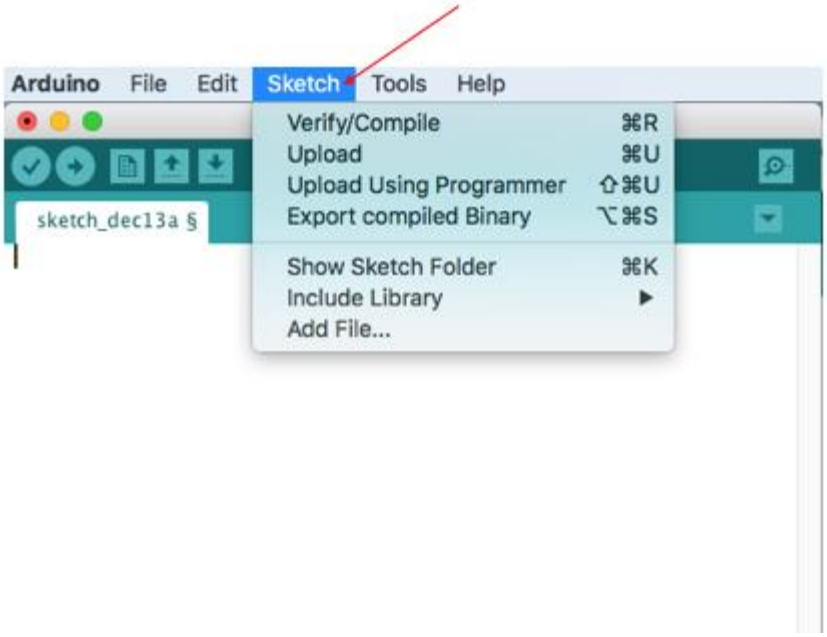
Type this into the URL: https://github.com/adafruit/Adafruit-WS2801-Library
You will be redirected to the GitHub files page. Select "Clone or Download".



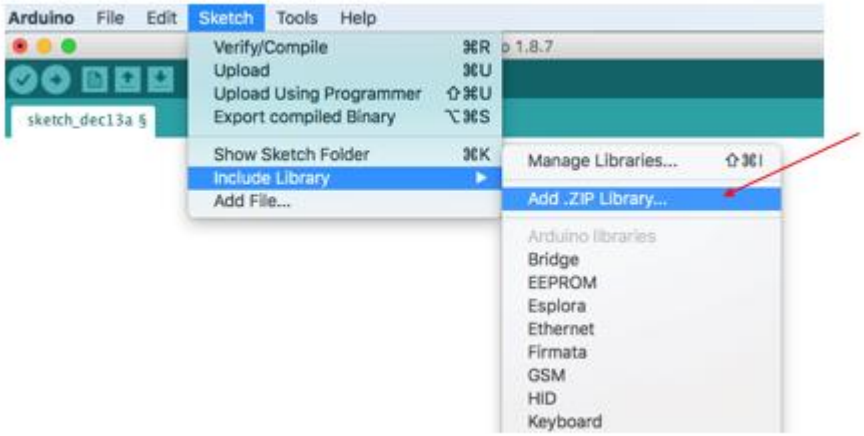
Next select “Download Zip”



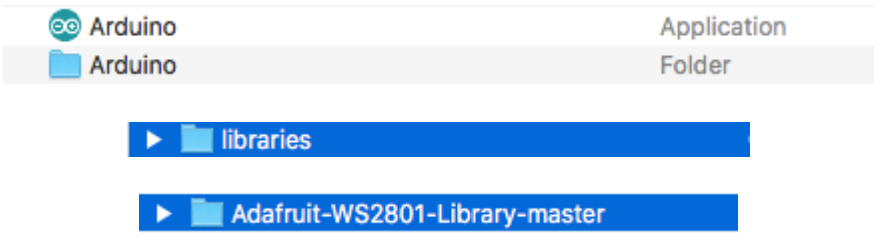
Once downloaded, it is necessary to add the libraries onto the Arduino software. Once inside the Arduino sketch, select the “Sketch” tab.

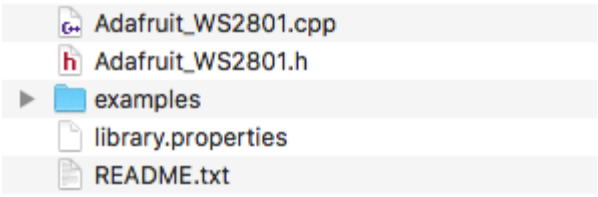


Next select “Include library” and then “Add .ZIP Library”.

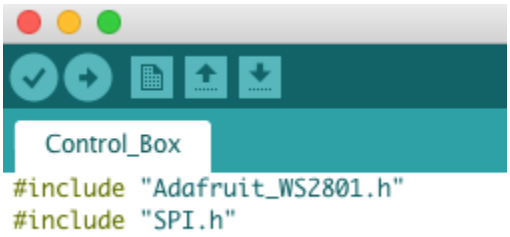


Now, within your computer, you must create a folder named “Arduino”. Inside the Arduino folder, create a “Libraries” folder; find the downloaded zip folder and drag it into the Arduino folder!

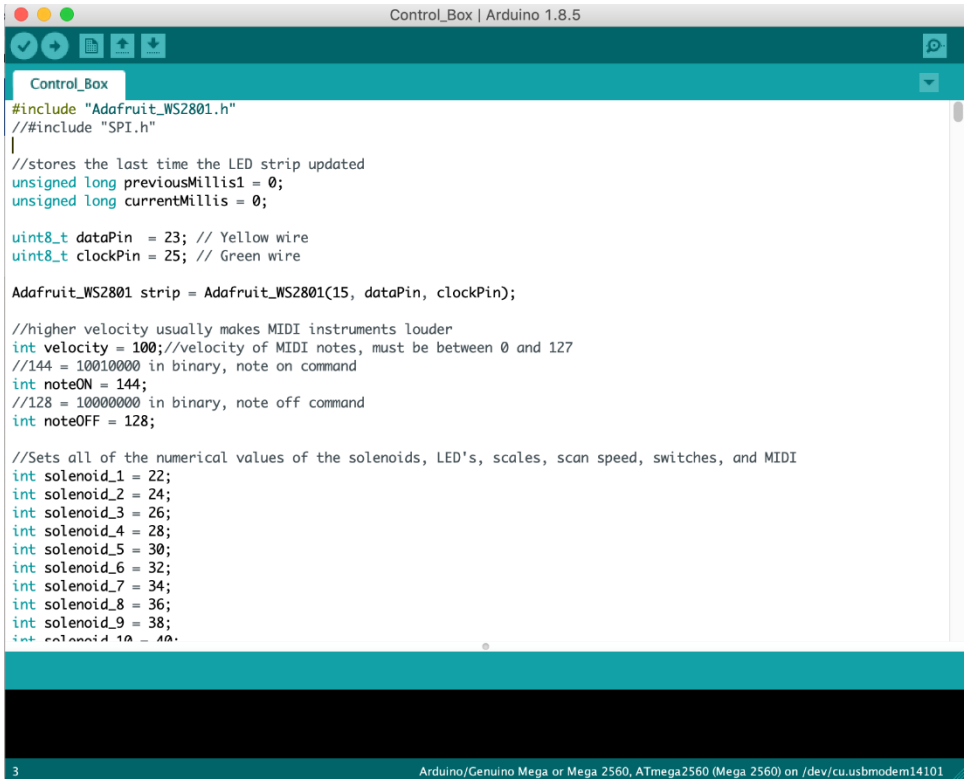




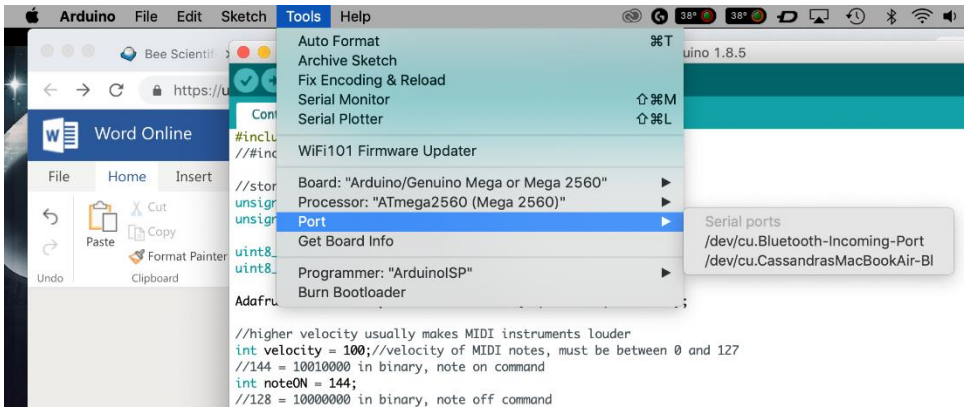
Once this is created, the files necessary may be included in the Arduino code by doing the following:



Once you have copied the code and downloaded the proper LED libraries, your Arduino sketch window should look like the following.



Now connect your computer to the Arduino Mega via USB. Make sure that you are not plugged into the 12V power supply to avoid causing unexpected damage to the board. Once connected, click on tools from the toolbar at the top of your window and mouse down to where it says ports.



From the list provided, select the port that says Arduino Mega or something similar. Also, be sure to select the proper board from the board side menu found just above the port menu. Once that is done, click on the check mark at the top left of the Arduino sketch window. This will compile the code and make sure everything is running accordingly.

If there is an error, double-check that you downloaded and included the correct LED library. Try to re-copy and paste the code in case you accidentally missed something.

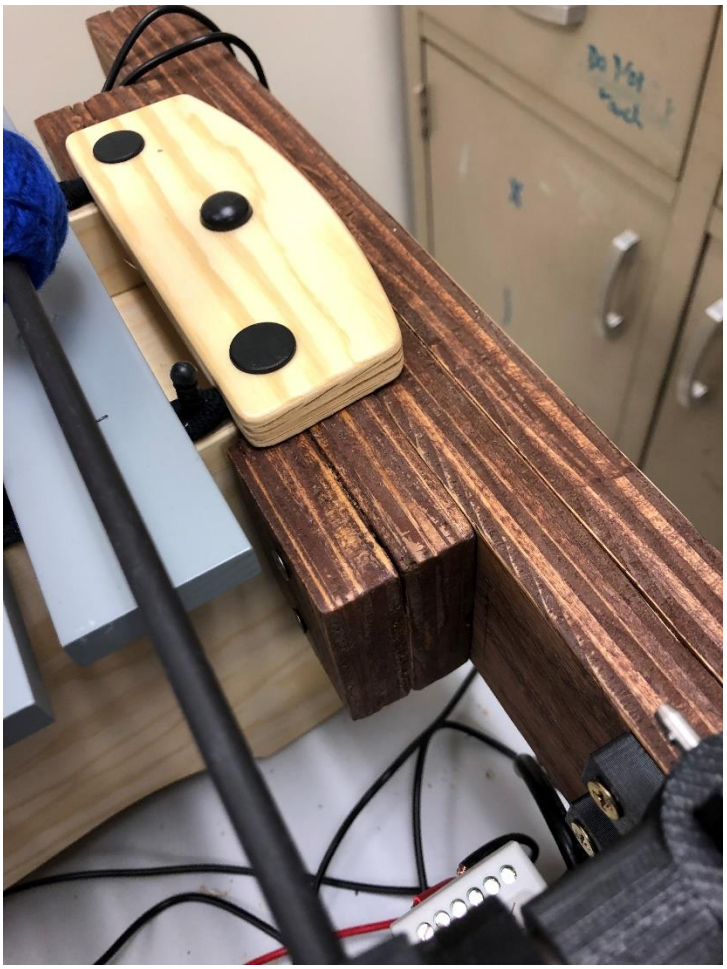
Once the code compiles like it should, click on the arrow right next to the check mark. This will upload the code to the Arduino. Once it is finished loading, disconnect the USB cable from your computer and connect it to the 5V power source on the power strip.

Metallophone Placement

To secure the metallophone to the wood base use one of the assembled clamping blocks shown in the picture below (two of the individual clamping blocks glued together).

Screw the block into the wood such that when you slide the metallophone in the mallets line up with the middle of the tone bells. Once the first block is placed, do the same on the opposite side, taking care to line them up with one another.

Next, slide the metallophone up to the stopping blocks. Place another set of clamping blocks on the other side of the metallophone to clamp it into place.



When you are finished placing the blocks, the placement of the metallophone should look something like this on both sides.

Control Box Cover

Cut a piece of acrylic so that it covers the front portion of the control box (about 34x3.5 inches). Mark and cut holes into the acrylic sheet to accommodate the four 1/8 audio jack plugs, the two rotary switches, and the potentiometer (measure the diameter of the threaded pieces and cut the wholes to be slightly larger).

Once the holes are cut, remove the paper from the acrylic and insert the components. Tighten them down using the provided locking nuts. Stickers can be used to help you remember what each of the switches do.

Screw the acrylic sheet to the wooden base. Use pilot holes to avoid cracking the acrylic.

Mallets System

We are first going to connect the mallets to the metal pivot bar. This is done using the Mallet Pivot Groove and Mallet Pivot Lip 3D printed parts. Consider sanding the sections that are going to be interlocking so they are not so tight. First, take one of the pieces and slide the mallet into the small channel.

Next, take the other half of the 3D printed part and line up the groove slots. Push the two together slightly so that they stay but don’t push them in all of the way.

Ensure that both pieces are lined up the way you want them, then force them together until there is little to no gap.

Next, take the Z-Height Adjust 3D printed parts and screw them into the inside of the wooden base close behind the clamping blocks. Take care to leave enough distance so that when you place the bar onto the 3D printed part, there is enough space between the rubber part of the mallet and the Mallet Pivot assembly.

Next, slide the male Z-Height Adjust 3D printed part onto the bar, then onto the 3D printed part just placed. Use a metal pin to hold the 3D printed assembly at the desire height. Repeat this exact same process for the other side.

Once those pieces are assembled, we are going to assemble the parts that connect the mallets to the solenoids. This is done by first connecting the ‘Solenoid Pivot’ and the ‘Mallet Solenoid Pivot’ using pin connections. The before and after assembly of this piece is as follows:

Next, slide on the previous assembled piece to the end of the mallet and also to the top of the solenoid. The end that connects to the mallet should be a press fit, but the piece that is connected to the solenoid should be glued to hold it in place.

Repeat the above processes for all of the solenoids and mallets. It should look like the following image when complete.

Arduino Mega Code

```
#include "Adafruit_WS2801.h"
//#include "SPI.h"

//stores the last time the LED strip updated
unsigned long previousMillis1 = 0;
unsigned long currentMillis = 0;

uint8_t dataPin  = 23; // Yellow wire
uint8_t clockPin = 25; // Green wire

Adafruit_WS2801 strip = Adafruit_WS2801(15, dataPin, clockPin);

//higher velocity usually makes MIDI instruments louder
int velocity = 100;//velocity of MIDI notes, must be between 0 and 127
//144 = 10010000 in binary, note on command
int noteON = 144;
//128 = 10000000 in binary, note off command
```

```
int noteOFF = 128;

//Sets all of the numerical values of the solenoids, LED's, scales, scan speed, switches, and MIDI
int solenoid_1 = 22;
int solenoid_2 = 24;
int solenoid_3 = 26;
int solenoid_4 = 28;
int solenoid_5 = 30;
int solenoid_6 = 32;
int solenoid_7 = 34;
int solenoid_8 = 36;
int solenoid_9 = 38;
int solenoid_10 = 40;
int solenoid_11 = 42;
int solenoid_12 = 44;
int solenoid_13 = 46;
int solenoid_14 = 48;
int solenoid_15 = 50;

int step_ = 13;
int random_ = 12;
int auto_scan = 11;
int auto_step_scan = 10;
int step_scan = 9;
int up_down = 8;
int up_down_repeat = 7;
int up_down_repeat_skip = 6;
int scale_1 = 53;
int scale_2 = 51;
int scale_3 = 49;
int scale_4 = 47;
int scale_5 = 45;
int scale_6 = 43;
int scale_7 = 41;
int scale_8 = 39;
int scan_speed = A14;
int button_1 = 37;
int button_2 = 35;
int button_3 = 33;
int button_4 = 31;

int button_condition1 = 0;
int button_condition2 = 0;
int button_condition3 = 0;
int button_condition4 = 0;
int const1 = 0;
int const2 = 0;
```

```
int const3 = 0;
int const4 = 0;
int const5 = 0;
int const6 = 0;
int const7 = 0;
int const8 = 0;
int const9 = 0;
int const10 = 0;
int const11 = 0;
int const12 = 0;
int const13 = 0;
int const14 = 0;
int const15 = 0;
int const16 = 0;
int n = 0;
int m = 0;
float q = 0;
int i = 0;
int j = 0;
int g = 0;
int h = 0;
int fire1 = 0;
int fire2 = false;
int fire3 = false;
int fire4 = false;
float pot = 0;
int timeStep = 0;
int LEDdelay = 0;
int scaleNumber = 0;

int solenoid_scale[8][15] = {
  {22, 24, 26, 28, 30, 32, 34, 36, 38, 40, 42, 44, 46, 48, 50}, //Full Scale
  {22, 24, 28, 30, 32, 36, 38, 42, 44, 46, 50, 22, 22, 22, 22}, //F Major Penatonic
  {22, 24, 28, 30, 34, 36, 38, 42, 44, 48, 50, 22, 22, 22, 22}, //Bb Penatonic
  {22, 24, 26, 30, 32, 36, 38, 40, 44, 46, 50, 22, 22, 22, 22}, //C Scale
  {24, 26, 30, 32, 34, 38, 40, 44, 46, 48, 50, 22, 22, 22, 22}, //G Scale
  {22, 24, 26, 30, 32, 36, 38, 40, 44, 46, 50, 22, 22, 22, 22}, //Egyptian 1
  {22, 26, 28, 32, 34, 36, 40, 42, 46, 48, 50, 22, 22, 22, 22}, //Japanese
  {22, 24, 28, 30, 34, 36, 38, 42, 44, 48, 50, 22, 22, 22, 22}, //Egyptian 2
};

int LED_scale[8][15] = {
  {0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14},
  {0, 1, 3, 4, 5, 7, 8, 10, 11, 12, 14, 22, 22, 22, 22},
  {0, 1, 3, 4, 6, 7, 8, 10, 11, 13, 14, 22, 22, 22, 22},
  {0, 1, 2, 4, 5, 7, 8, 9, 11, 12, 14, 22, 22, 22, 22},
  {1, 2, 4, 5, 6, 8, 9, 11, 12, 13, 14, 22, 22, 22, 22},
```

```
  {0, 1, 2, 4, 5, 7, 8, 9, 11, 12, 14, 22, 22, 22, 22},
  {0, 2, 3, 5, 6, 7, 9, 10, 12, 13, 14, 22, 22, 22, 22},
  {0, 1, 3, 4, 6, 7, 8, 10, 11, 13, 14, 22, 22, 22, 22},
};

int MIDI_scale[8][15] = {
  {36, 38, 40, 41, 43, 45, 46, 48, 50, 52, 53, 55, 57, 58, 60},
  {36, 38, 41, 43, 45, 48, 50, 53, 55, 57, 60, 36, 36, 36, 36},
  {36, 38, 41, 43, 46, 48, 50, 53, 55, 58, 60, 36, 36, 36, 36},
  {36, 38, 40, 43, 45, 48, 50, 52, 55, 57, 60, 36, 36, 36, 36},
  {38, 40, 41, 45, 46, 50, 52, 55, 57, 58, 60, 36, 36, 36, 36},
  {36, 38, 43, 43, 45, 48, 50, 52, 55, 57, 60, 36, 36, 36, 36},
  {36, 40, 41, 45, 46, 48, 52, 53, 57, 58, 60, 36, 36, 36, 36},
  {36, 38, 41, 43, 46, 48, 50, 53, 55, 58, 60, 36, 36, 36, 36},
};

void setup() {
//initiate serial communication
  Serial.begin(9600);
  Serial2.begin(31250);

//delay time step
  timeStep = 150;

//initialize LED strip
  strip.begin();
  strip.show();

//solenoid numbering - sets all the pins to outputs
  pinMode(solenoid_1, OUTPUT); //C
  pinMode(solenoid_2, OUTPUT); //D
  pinMode(solenoid_3, OUTPUT); //E
  pinMode(solenoid_4, OUTPUT); //F
  pinMode(solenoid_5, OUTPUT); //G
  pinMode(solenoid_6, OUTPUT); //A
  pinMode(solenoid_7, OUTPUT); //Bb
  pinMode(solenoid_8, OUTPUT); //C
  pinMode(solenoid_9, OUTPUT); //D
  pinMode(solenoid_10, OUTPUT); //E
  pinMode(solenoid_11, OUTPUT); //F
  pinMode(solenoid_12, OUTPUT); //G
  pinMode(solenoid_13, OUTPUT); //A
  pinMode(solenoid_14, OUTPUT); //Bb
  pinMode(solenoid_15, OUTPUT); //C

//Switch Ensamble Functions - sets all the pins to inputs
  pinMode(step_, OUTPUT);
```

```
  pinMode(random_, OUTPUT);
  pinMode(auto_scan, OUTPUT);
  pinMode(auto_step_scan, OUTPUT);
  pinMode(step_scan, OUTPUT);
  pinMode(up_down, OUTPUT);
  pinMode(up_down_repeat, OUTPUT);
  pinMode(up_down_repeat_skip, OUTPUT);

//Scales - sets all the pins to inputs
  pinMode(scale_1, OUTPUT);
  pinMode(scale_2, OUTPUT);
  pinMode(scale_3, OUTPUT);
  pinMode(scale_4, OUTPUT);
  pinMode(scale_5, OUTPUT);
  pinMode(scale_6, OUTPUT);
  pinMode(scale_7, OUTPUT);
  pinMode(scale_8, OUTPUT);

//Scan Speed - sets the pin to input
//  pinMode(scan_speed, INPUT);

//Switches - sets all the pins to inputs
  pinMode(button_1, OUTPUT);
  pinMode(button_2, OUTPUT);
  pinMode(button_3, OUTPUT);
  pinMode(button_4, OUTPUT);

//sets all of the relays to be on
  digitalWrite(solenoid_1, HIGH);
  digitalWrite(solenoid_2, HIGH);
  digitalWrite(solenoid_3, HIGH);
  digitalWrite(solenoid_4, HIGH);
  digitalWrite(solenoid_5, HIGH);
  digitalWrite(solenoid_6, HIGH);
  digitalWrite(solenoid_7, HIGH);
  digitalWrite(solenoid_8, HIGH);
  digitalWrite(solenoid_9, HIGH);
  digitalWrite(solenoid_10, HIGH);
  digitalWrite(solenoid_11, HIGH);
  digitalWrite(solenoid_12, HIGH);
  digitalWrite(solenoid_13, HIGH);
  digitalWrite(solenoid_14, HIGH);
  digitalWrite(solenoid_15, HIGH);
}

void loop() {
```

```
//creates constant values for the if statements below
  const1 = digitalRead(scale_1);
  const2 = digitalRead(scale_2);
  const3 = digitalRead(scale_3);
  const4 = digitalRead(scale_4);
  const5 = digitalRead(scale_5);
  const6 = digitalRead(scale_6);
  const7 = digitalRead(scale_7);
  const8 = digitalRead(scale_8);

//sets the m value to specifiy which scale to use
//F major scale/ D minor scale
  if (const1 == HIGH){
    //F G A Bb C D E F G
    m = 0;
  }
//Egypt 2
  else if (const2 == HIGH){
    //C D F G Bb C
    m = 1;
  }
//F pentatonic
  else if (const3 == HIGH){
    //F G A C D F G
    m = 2;
  }

//Bb pentatonic
  else if (const4 == HIGH){
    //Bb C D F G Bb C
    m = 3;
  }

//C pentatonic
  else if (const5 == HIGH){
    //C D E G A C D
    m = 4;
  }

//G pentatonic
  else if (const6 == HIGH){
    //G A Bb D E G A
    m = 5;
  }

// Egyptian pentatonic
  else if (const7 == HIGH){
```

```
    //D E G A C D E
    m = 6;
  }

// Japanese pentatonic
  else if (const8 == HIGH){
    //E F A Bb C E F
    m = 7;
  }

//creates constant values for the if statements below
  const9 = digitalRead(step_);
  const10 = digitalRead(random_);
  const11 = digitalRead(auto_scan);
  const12 = digitalRead(auto_step_scan);
  const13 = digitalRead(step_scan);
  const14 = digitalRead(up_down);
  const15 = digitalRead(up_down_repeat);
  const16 = digitalRead(up_down_repeat_skip);

// sets the n value specifiy which Switch Ensamble function to use
  if (const9 == HIGH){
    n = 0;
  }
  else if (const10 == HIGH){
    n = 1;
  }
  else if (const11 == HIGH){
    n = 2;
  }
  else if (const12 == HIGH){
    n = 3;
  }
  else if (const13 == HIGH){
    n = 4;
  }
  else if (const14 == HIGH){
    n = 5;
  }
  else if (const15 == HIGH){
    n = 6;
  }
  else if (const16 == HIGH){
    n = 7;
  }
  //Serial.print(n);
```

```
//switch cases for each of the 8 Switch Ensamble Functions with input n
  switch(n) {
    i = 0;
    j = 0;
    g = 0;

//step
    case 0:
      for (g = 0; g < 15; g++){
        strip.setPixelColor(g,0,0,0);
        strip.show();
      }
      if (m == 0){
        scaleNumber = 14;
      }
      else{
        scaleNumber = 10;
      }
      if (i <= scaleNumber){
        button_condition1 = digitalRead(button_1);
        if (button_condition1 == LOW){
          fire1 = 0;
        }
        if (button_condition1 == HIGH && fire1 == 0){
          digitalWrite(solenoid_scale[m][i], LOW); //turns on the solenoid (i) using scale (m)
          MIDImessage(noteON, MIDI_scale[m][j], velocity);
          delay(100);
          digitalWrite(solenoid_scale[m][i], HIGH);  //turns off the solenoid
          MIDImessage(noteOFF, MIDI_scale[m][j], velocity);
          delay(timeStep);
          i++;
          j++;
          fire1 = 1;
          //Serial.print(i);
        }
        else{
          digitalWrite(solenoid_scale[m][i], HIGH);
          MIDImessage(noteOFF, MIDI_scale[m][j], velocity);
          strip.setPixelColor(i,0,0,0);
          strip.show();
        }
      }
      else{
        i = 0;
        j = 0;
      }
```

```
      break;

//random
    case 1:
      for (g = 0; g < 15; g++){
        strip.setPixelColor(g,0,0,0);
        strip.show();
      }
      if (m == 0){
          scaleNumber = 14;
        }
        else{
          scaleNumber = 10;
        }
      i = rand()%scaleNumber;
        button_condition1 = digitalRead(button_1);
        if (button_condition1 == LOW){
          fire1 = 0;
        }
        if (button_condition1 == HIGH && fire1 == 0){
          digitalWrite(solenoid_scale[m][i], LOW);
          MIDImessage(noteON, MIDI_scale[m][i], velocity);
          delay(100);
          digitalWrite(solenoid_scale[m][i], HIGH);
          MIDImessage(noteOFF, MIDI_scale[m][i], velocity);
          delay(timeStep);
          fire1 = 1;
          //Serial.print(i);
        }
        else{
          digitalWrite(solenoid_scale[m][i], HIGH);
          MIDImessage(noteOFF, MIDI_scale[m][i], velocity);
        }
      break;

//auto scan
    case 2:
      q = analogRead(scan_speed);
      //Serial.println(q);
      pot = q/1023;
      LEDdelay = 5000-(pot*4500);
      if (m == 0){
        scaleNumber = 14;
      }
      else{
        scaleNumber = 10;
      }
```

```
strip.setPixelColor(LED_scale[m][g],0,0,0);
strip.show();
button_condition1 = digitalRead(button_1);
if (button_condition1 == LOW){
   fire1 = 0;
}
if (button_condition1 == HIGH && fire1 == 0){
  delay(500);
  currentMillis = millis();
  previousMillis1 = currentMillis;
  button_condition1 = digitalRead(button_1);
  fire1 = 1;
  while (button_condition1 == LOW || fire1 == 1){
    if (g <= scaleNumber){
      strip.setPixelColor(LED_scale[m][g],255,255,255);
      strip.show();
      currentMillis = millis();
      if (currentMillis-previousMillis1 > LEDdelay){
        strip.setPixelColor(LED_scale[m][g],0,0,0);
        strip.show();
        previousMillis1 = currentMillis;
        g++;
      }
      button_condition1 = digitalRead(button_1);
      if (button_condition1 == LOW){
        fire1 = 0;
      }
    }
    else{
      g = 0;
    }
  }
  strip.setPixelColor(LED_scale[m][g],0,0,0);
  strip.show();
  digitalWrite(solenoid_scale[m][g], LOW);
  MIDImessage(noteON, MIDI_scale[m][g], velocity);
  delay(100);
  digitalWrite(solenoid_scale[m][g], HIGH);
  MIDImessage(noteOFF, MIDI_scale[m][g], velocity);
  delay(timeStep);
  g = 0;
  fire1 = 1;
}
else{
  digitalWrite(solenoid_scale[m][g], HIGH);
  MIDImessage(noteOFF, MIDI_scale[m][g], velocity);
```

```
      }
      break;

//auto step scan
    case 3:
      q = analogRead(scan_speed);
      //Serial.println(q);
      pot = q/1023;
      LEDdelay = 5000-(pot*4500);
      if (m == 0){
        scaleNumber = 14;
      }
      else{
        scaleNumber = 10;
      }
      strip.setPixelColor(LED_scale[m][g],0,0,0);
      strip.show();
      button_condition1 = digitalRead(button_1);
      if (button_condition1 == LOW){
         fire1 = 0;
        }
      if (button_condition1 == HIGH && fire1 == 0){
        delay(500);
        currentMillis = millis();
        previousMillis1 = currentMillis;
        button_condition1 = digitalRead(button_1);
        fire1 = 1;
        while (button_condition1 == LOW || fire1 == 1){
          if (g <= scaleNumber){
            strip.setPixelColor(LED_scale[m][g],255,255,255);
            strip.show();
            currentMillis = millis();
            if (currentMillis-previousMillis1 > LEDdelay){
              strip.setPixelColor(LED_scale[m][g],0,0,0);
              strip.show();
              previousMillis1 = currentMillis;
              g++;
            }
            button_condition1 = digitalRead(button_1);
            if (button_condition1 == LOW){
              fire1 = 0;
            }
          }
          else{
            g = 0;
          }
```

```
        }
        delay(250);
        fire1 = 1;
        button_condition1 = digitalRead(button_1);
        while (button_condition1 == LOW || fire1 == 1){
          strip.setPixelColor(LED_scale[m][g],255,0,0);
          strip.show();
          button_condition1 = digitalRead(button_1);
          if (button_condition1 == LOW){
          fire1 = 0;
          }
        }
        strip.setPixelColor(LED_scale[m][g],0,0,0);
        strip.show();
        digitalWrite(solenoid_scale[m][g], LOW);
        MIDImessage(noteON, MIDI_scale[m][g], velocity);
        delay(100);
        digitalWrite(solenoid_scale[m][g], HIGH);
        MIDImessage(noteOFF, MIDI_scale[m][g], velocity);
        delay(timeStep);
        g = 0;
        fire1 = 1;
      }
      else{
        digitalWrite(solenoid_scale[m][g], HIGH);
        MIDImessage(noteOFF, MIDI_scale[m][g], velocity);
      }
      break;

//step scan
    case 4:
      q = analogRead(scan_speed);
      Serial.println(q);
      pot = q/1023;
      LEDdelay = 5000-(pot*4500);
      if (m == 0){
        scaleNumber = 14;
      }
      else{
        scaleNumber = 10;
      }
      if (g <= scaleNumber){
        button_condition1 = digitalRead(button_1);
        if (button_condition1 == LOW){
          fire1 = 0;
        }
```

```
      button_condition2 = digitalRead(button_2);
      if (button_condition2 == LOW){
        fire2 = 0;
      }
      if (button_condition1 == HIGH && fire1 == 0){
        strip.setPixelColor(LED_scale[m][g-1],0,0,0);
        strip.show();
        if (g == 0){
          strip.setPixelColor(14,0,0,0);
          strip.show();
        }
        strip.setPixelColor(LED_scale[m][g],255,255,255);
        strip.show();
        g++;
        h = g-1;
        delay(timeStep);
        fire1 = 1;
      }
      else if (button_condition2 == HIGH && fire2 == 0){
        digitalWrite(solenoid_scale[m][h], LOW);
        MIDImessage(noteON, MIDI_scale[m][h], velocity);
        delay(100);
        digitalWrite(solenoid_scale[m][h], HIGH);
        MIDImessage(noteOFF, MIDI_scale[m][h], velocity);
        delay(timeStep);
        fire2 = 1;
        //Serial.print(i);
      }
      else{
        digitalWrite(solenoid_scale[m][h], HIGH);
        MIDImessage(noteOFF, MIDI_scale[m][h], velocity);
      }
    }
    else{
      g = 0;
    }
    break;

//up down
  case 5:
    for (g = 0; g < 15; g++){
      strip.setPixelColor(g,0,0,0);
      strip.show();
    }
    if (m == 0){
      scaleNumber = 14;
```

```
}
else{
  scaleNumber = 10;
}
button_condition1 = digitalRead(button_1);
if (button_condition1 == LOW){
    fire1 = 0;
  }
button_condition2 = digitalRead(button_2);
if (button_condition2 == LOW){
    fire2 = 0;
  }
if (button_condition1 == HIGH && fire1 == 0){
  i++;
  if (i >= scaleNumber+2){
    i = 1;
  }
  h = i-1;
  digitalWrite(solenoid_scale[m][h], LOW);
  MIDImessage(noteON, MIDI_scale[m][h], velocity);
  delay(100);
  digitalWrite(solenoid_scale[m][h], HIGH);
  MIDImessage(noteOFF, MIDI_scale[m][h], velocity);
  delay(timeStep);
  fire1 = 1;
  //Serial.print(i);
}
else if (button_condition2 == HIGH && fire2 == 0){
  i--;
  h--;
  if (h <= -1){
    i = scaleNumber+1;
    h = scaleNumber;
    digitalWrite(solenoid_scale[m][h], LOW);
    MIDImessage(noteON, MIDI_scale[m][h], velocity);
    delay(100);
    digitalWrite(solenoid_scale[m][h], HIGH);
    MIDImessage(noteOFF, MIDI_scale[m][h], velocity);
    delay(timeStep);
    fire2 = 1;
  }
  else{
    digitalWrite(solenoid_scale[m][h], LOW);
    MIDImessage(noteON, MIDI_scale[m][h], velocity);
    delay(100);
    digitalWrite(solenoid_scale[m][h], HIGH);
```

```
        MIDImessage(noteOFF, MIDI_scale[m][h], velocity);
        delay(timeStep);
        fire2 = 1;
       }
      }
      else{
       digitalWrite(solenoid_scale[m][h], HIGH);
       MIDImessage(noteOFF, MIDI_scale[m][h], velocity);
      }
      break;

//up down repeat
    case 6:
      for (g = 0; g < 15; g++){
       strip.setPixelColor(g,0,0,0);
       strip.show();
      }
      if (m == 0){
       scaleNumber = 14;
      }
      else{
       scaleNumber = 10;
      }
      button_condition1 = digitalRead(button_1);
      if (button_condition1 == LOW){
         fire1 = 0;
       }
      button_condition2 = digitalRead(button_2);
      if (button_condition2 == LOW){
         fire2 = 0;
       }
      button_condition3 = digitalRead(button_3);
      if (button_condition3 == LOW){
         fire3 = 0;
       }
      if (button_condition1 == HIGH && fire1 == 0){
       i++;
       if (i >= scaleNumber+2){
        i = 1;
       }
       h = i-1;
       digitalWrite(solenoid_scale[m][h], LOW);
       MIDImessage(noteON, MIDI_scale[m][h], velocity);
       delay(100);
       digitalWrite(solenoid_scale[m][h], HIGH);
       MIDImessage(noteOFF, MIDI_scale[m][h], velocity);
```

```
      delay(timeStep);
      fire1 = 1;
      //Serial.print(i);
    }
    else if (button_condition2 == HIGH && fire2 == 0){
      i--;
      h--;
      if (h <= -1){
        i = scaleNumber+1;
        h = scaleNumber;
        digitalWrite(solenoid_scale[m][h], LOW);
        MIDImessage(noteON, MIDI_scale[m][h], velocity);
        delay(100);
        digitalWrite(solenoid_scale[m][h], HIGH);
        MIDImessage(noteOFF, MIDI_scale[m][h], velocity);
        delay(timeStep);
        fire2 = 1;
      }
      else{
        digitalWrite(solenoid_scale[m][h], LOW);
        MIDImessage(noteON, MIDI_scale[m][h], velocity);
        delay(100);
        digitalWrite(solenoid_scale[m][h], HIGH);
        MIDImessage(noteOFF, MIDI_scale[m][h], velocity);
        delay(timeStep);
        fire2 = 1;
      }
    }
    else if (button_condition3 == HIGH && fire3 == 0){
      digitalWrite(solenoid_scale[m][h], LOW);
      MIDImessage(noteON, MIDI_scale[m][h], velocity);
      delay(100);
      digitalWrite(solenoid_scale[m][h], HIGH);
      MIDImessage(noteOFF, MIDI_scale[m][h], velocity);
      delay(timeStep);
      fire3 = 1;
    }
    else{
      digitalWrite(solenoid_scale[m][h], HIGH);
      MIDImessage(noteOFF, MIDI_scale[m][h], velocity);
    }
    break;

//up down repeat skip
  case 7:
    for (g = 0; g < 15; g++){
```

```
    strip.setPixelColor(g,0,0,0);
    strip.show();
  }
if (m == 0){
  scaleNumber = 14;
}
else{
  scaleNumber = 10;
}
button_condition1 = digitalRead(button_1);
if (button_condition1 == LOW){
    fire1 = 0;
  }
button_condition2 = digitalRead(button_2);
if (button_condition2 == LOW){
    fire2 = 0;
  }
button_condition3 = digitalRead(button_3);
if (button_condition3 == LOW){
    fire3 = 0;
  }
button_condition4 = digitalRead(button_4);
if (button_condition4 == LOW){
    fire4 = 0;
  }
if (button_condition1 == HIGH && fire1 == 0){
  i++;
  if (i >= scaleNumber+2){
    i = 1;
  }
  h = i-1;
  digitalWrite(solenoid_scale[m][h], LOW);
  MIDImessage(noteON, MIDI_scale[m][h], velocity);
  delay(100);
  digitalWrite(solenoid_scale[m][h], HIGH);
  MIDImessage(noteOFF, MIDI_scale[m][h], velocity);
  delay(timeStep);
  fire1 = 1;
  //Serial.print(i);
}
else if (button_condition2 == HIGH && fire2 == 0){
  i--;
  h--;
  if (h <= -1){
    i = scaleNumber+1;
    h = scaleNumber;
```

```
    digitalWrite(solenoid_scale[m][h], LOW);
    MIDImessage(noteON, MIDI_scale[m][h], velocity);
    delay(100);
    digitalWrite(solenoid_scale[m][h], HIGH);
    MIDImessage(noteOFF, MIDI_scale[m][h], velocity);
    delay(timeStep);
    fire2 = 1;
   }
   else{
    digitalWrite(solenoid_scale[m][h], LOW);
    MIDImessage(noteON, MIDI_scale[m][h], velocity);
    delay(100);
    digitalWrite(solenoid_scale[m][h], HIGH);
    MIDImessage(noteOFF, MIDI_scale[m][h], velocity);
    delay(timeStep);
    fire2 = 1;
   }
  }
  else if (button_condition3 == HIGH && fire3 == 0){
   digitalWrite(solenoid_scale[m][h], LOW);
   MIDImessage(noteON, MIDI_scale[m][h], velocity);
   delay(100);
   digitalWrite(solenoid_scale[m][h], HIGH);
   MIDImessage(noteOFF, MIDI_scale[m][h], velocity);
   delay(timeStep);
   fire3 = 1;
  }
  else if (button_condition4 == HIGH && fire4 == 0){
   i = i+2;
   if (i == scaleNumber+2){
    i = 1;
   }
   else if(i >= scaleNumber+3){
    i = 2;
   }
   h = i-1;
   digitalWrite(solenoid_scale[m][h], LOW);
   MIDImessage(noteON, MIDI_scale[m][h], velocity);
   delay(100);
   digitalWrite(solenoid_scale[m][h], HIGH);
   MIDImessage(noteOFF, MIDI_scale[m][h], velocity);
   delay(timeStep);
   fire4 = 1;
  }
  else{
   digitalWrite(solenoid_scale[m][h], HIGH);
```

```
      MIDImessage(noteOFF, MIDI_scale[m][h], velocity);
     }
     break;
  }
}

//send MIDI message
void MIDImessage(int command, int MIDInote, int MIDIvelocity) {
  Serial2.write(command);//send note on or note off command
  Serial2.write(MIDInote);//send pitch data
  Serial2.write(MIDIvelocity);//send velocity data
}
```