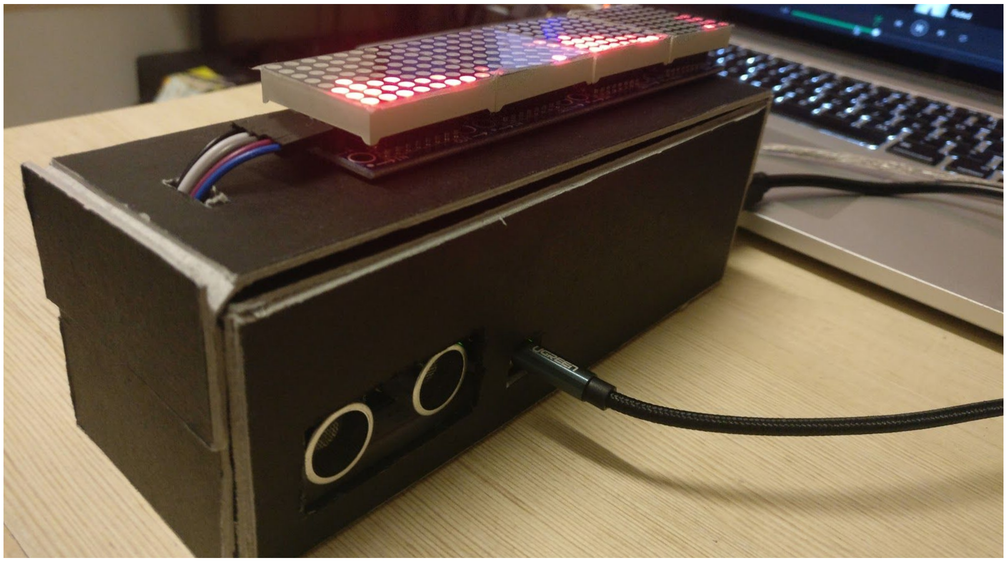
Tse Man Fung, Jason

Momentum

SM3610 Hardware Hacking - Final Project

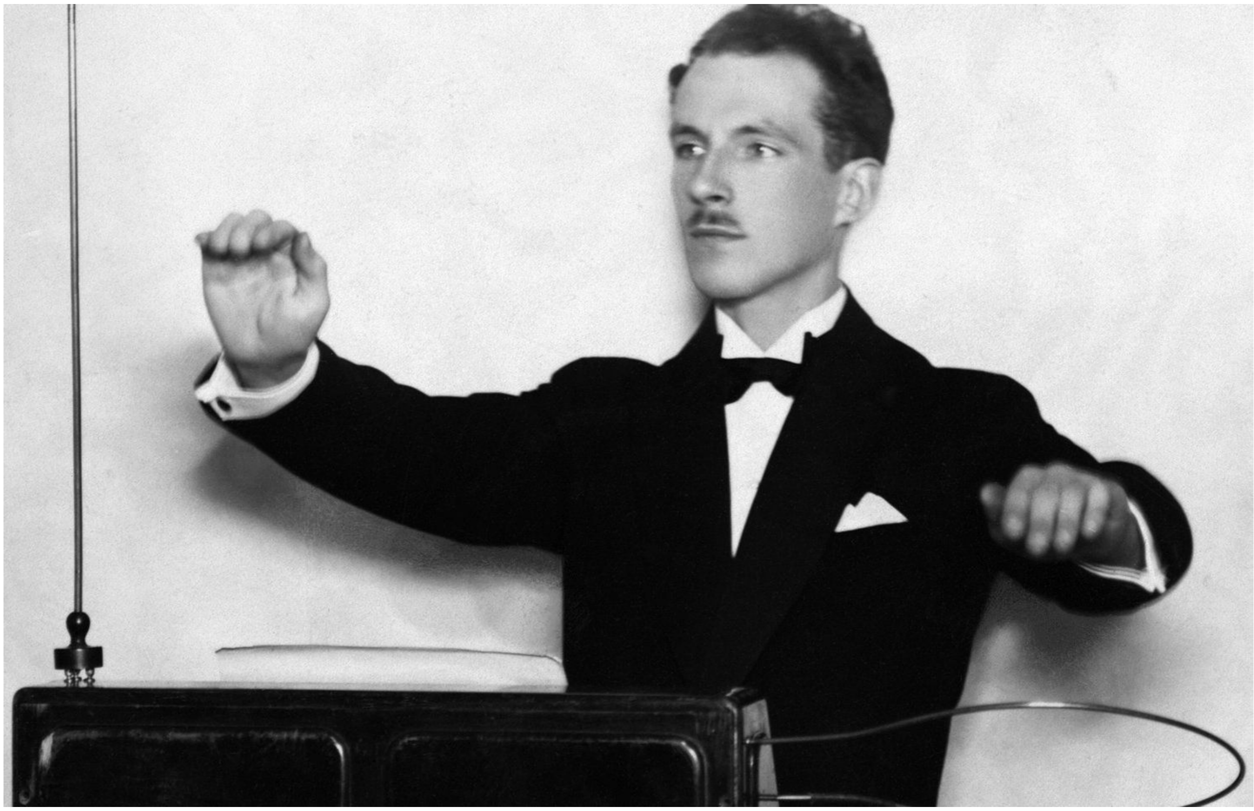


Project Thesis

“Momentum” is an arduino-powered MIDI music instrument and also an audio visualizer. Which is inspired by the the nostalgic electronic musical instrument, the theremin. The performer essentially performs with their hand movement dynamically, and elegantly like flowing water. The visualized audio on the LED screen reflects the volume and pitch, meanwhile reacts to the momentum of performer's movement.

Idea

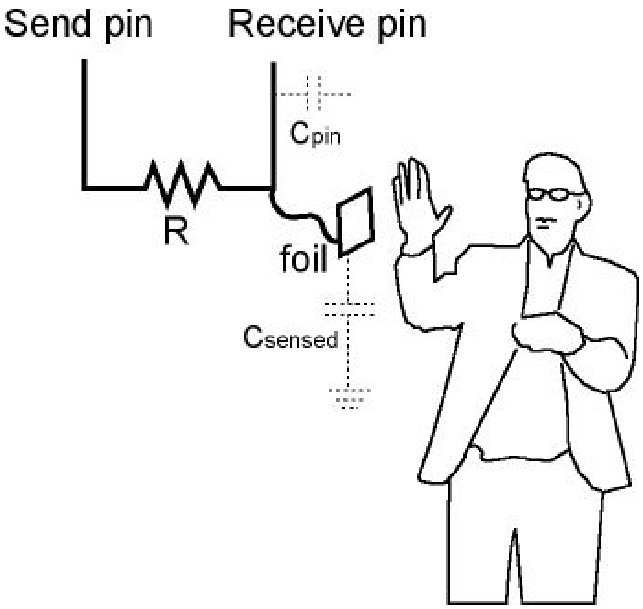
The Theremin



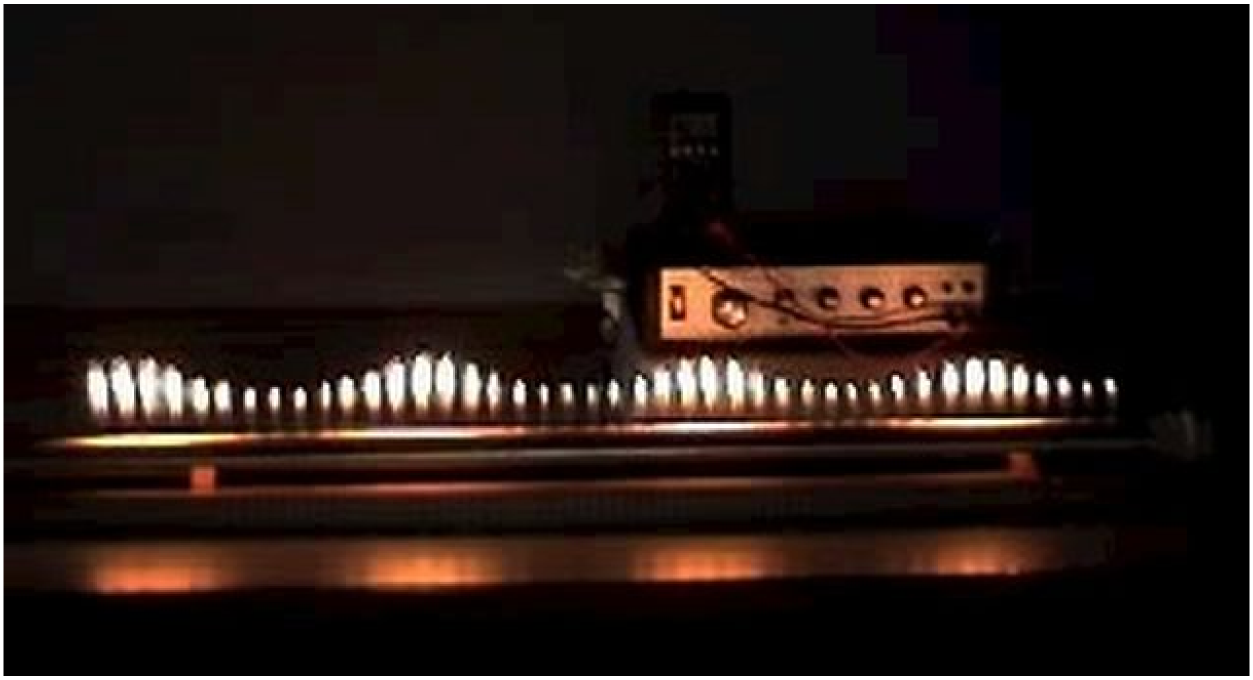
The Theremin is an Soviet invented electronic musical instrument that one can play without physically touching it. The main modules of such an instrument is essentially the body and 2 antennas, one controlling pitch, and one controlling volume. The thereminist perform with their hands moving around the the instrument to create electric signals, or say musical notes then amplified and sent to a speaker, synthesised a violin like sound.

https://youtu.be/K6KbEnGnymk

The science and physics behind such extraordinary and magical playing style is capacitive sensing, such form of detection allows control of instrument without touching it. However, DIY capacitive sensing is rather unreliable and unstable, from my previous experience. The selection of the foil (or antena) really matters, as the size and material affect the electric circuit.



Ruben’s Tube



The Ruben’s tube, a.k.a. The standing wave flame tube, is piece of German invented scientific apparatus that visualize acoustic wave. It graphically visualizes sound waves and sound pressure, just like another form of oscilloscope. Nowadays, It is typically used with musical instruments for music visualization.

I am inspired by how the moving firet, the performer’s motion and also the motion of the soundwave can be coherent and interactive. To further enhance the graphics, I would visualize audio spectrum rather than the oscilloscope that only visualize a certain frequencies’ wave, rather than the entire music spectrum.

https://youtu.be/Q3oItpVa9fs?t=3m32s

Implementation

It can be imagined as someone is playing with a crystal ball that shows reactive music.

Like I said, I've tried capacitive sensing before, and in my personal experience, the performance fluctuates a lot and the operation always goes out of control, due to bad choice of material and poor connection. To make it simple and controllable, I use 2 ultrasonic sensors instead. One for controlling the volume of a MIDI channel, another for controlling the MIDI note.

Since the project is fundamentally a MIDI device, the sound synthesis dirty work will be done by a third party digital music software, and my selection of choice is the free GarageBand. The MIDI device not only sends music notes to the computer, but also perform MIDI command, like volume control.

Midi Library

```
#include <MIDI.h>
MIDI.sendNoteOff(faded[constrain(oNoteNum, 0 ,7)], 0, 1);
MIDI.sendNoteOn(faded[constrain(noteNum, 0, 7)], 127, 1);
```

Distance

```
#include <MIDI.h>
 digitalWrite(trigPinA, LOW);  // Added this line
  delayMicroseconds(2); // Added this line
  digitalWrite(trigPinA, HIGH);
  delayMicroseconds(10); // Added this line
  digitalWrite(trigPinA, LOW);
  durationA = pulseIn(echoPinA, HIGH);
  distanceA = (durationA/2) / 29.1;
  if (distanceA >= mDistanceA){
    sDistanceA = 0;
  }else{
    sDistanceA = smoothValue(distanceA, sDistanceA);
  }
}
```

Smooth Algo

```
vol = map(sDistanceA, 0, 20, 0, 127);
 if(distanceA < mDistanceA) {
  if(oVol != vol){
   oVol = vol;
   MIDI.sendControlChange(7, 127-vol, 1);
  }
 }else if(oVol != 0){
   vol = 0;
   oVol = vol;
   MIDI.sendControlChange(7, vol, 1);
 }
```

Other than basic sound synthesis, I would like to visualize music in either hardware form or in processing. As processing is elementally designed to get visual work done, it opens up possibility for fancy graphical representation of the sound. However, I would like to put the device into a more self contained instrument. Since performing with such an instrument is already a performance, considering the gesture required to generate music, to me, it is better to centralize all the performing/visual elements, and place the visual element around the MIDI device. And LED matrix, or specifically 2D LED matrix can do the job.

Visual

```
#include <LedControl.h>
#include <FHT.h>
for (int i = 0 ; i < FHT_N / 2 ; i++) {
  if (fht_log_out[i] > 20)
```

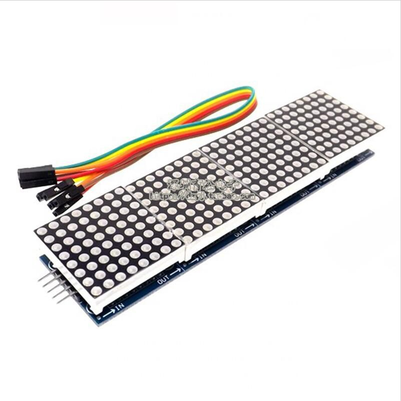
```
   fht_log_out[i] = fht_log_out[i] - 24;
 }

lc.setColumn(0, 7 - 0, bar(av(fht_log_out, 0, 8)));
 for (int i = 1; i < 8; i++) {
  lc.setColumn(0, 7 - i, bar(av(fht_log_out, 0 + i * 8, 8 + i * 8) * 2.4));
 }
 //        // Second matrix
 for (int i = 0; i < 8; i++) {
  lc.setColumn(1, 7 - i, bar(av(fht_log_out, 64 + i * 8, 72 + i * 8) * 2.4));
 }
```

Problem

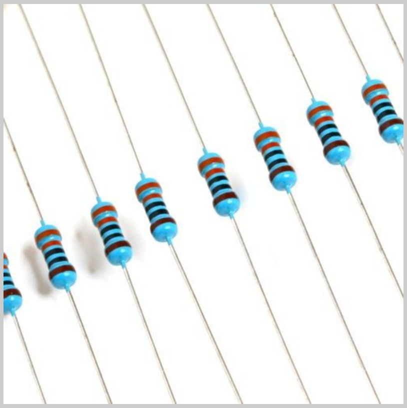
Since the audio visualization process requires the necessary audio signal from the computer, the typical approach is send serial data to the arduino, or set analog signal to the arduino with a physical audio cable from the computer. By the time of the actual development, I cannot find any possible software solution that does not exploit the serial port data parsing. Serial writing is not available as MIDI port has already occupy the TX port. Eventually I used the later method mentioned.

Components

Ultrasonic sensor - SR04 *2



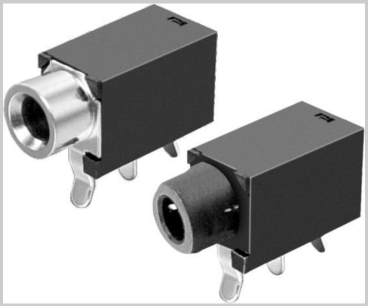
MAX7219 LED matrix



220 ohm resistors

7 pin midi socket

MIDI to USB cable

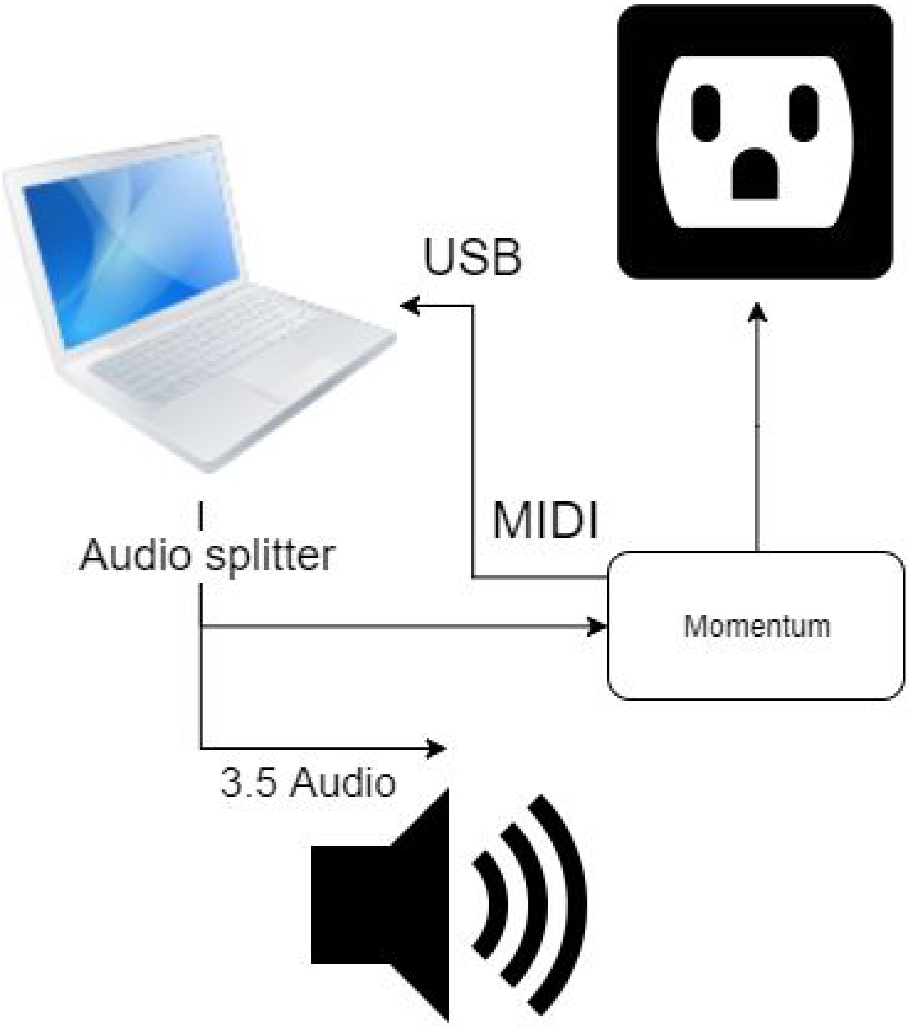


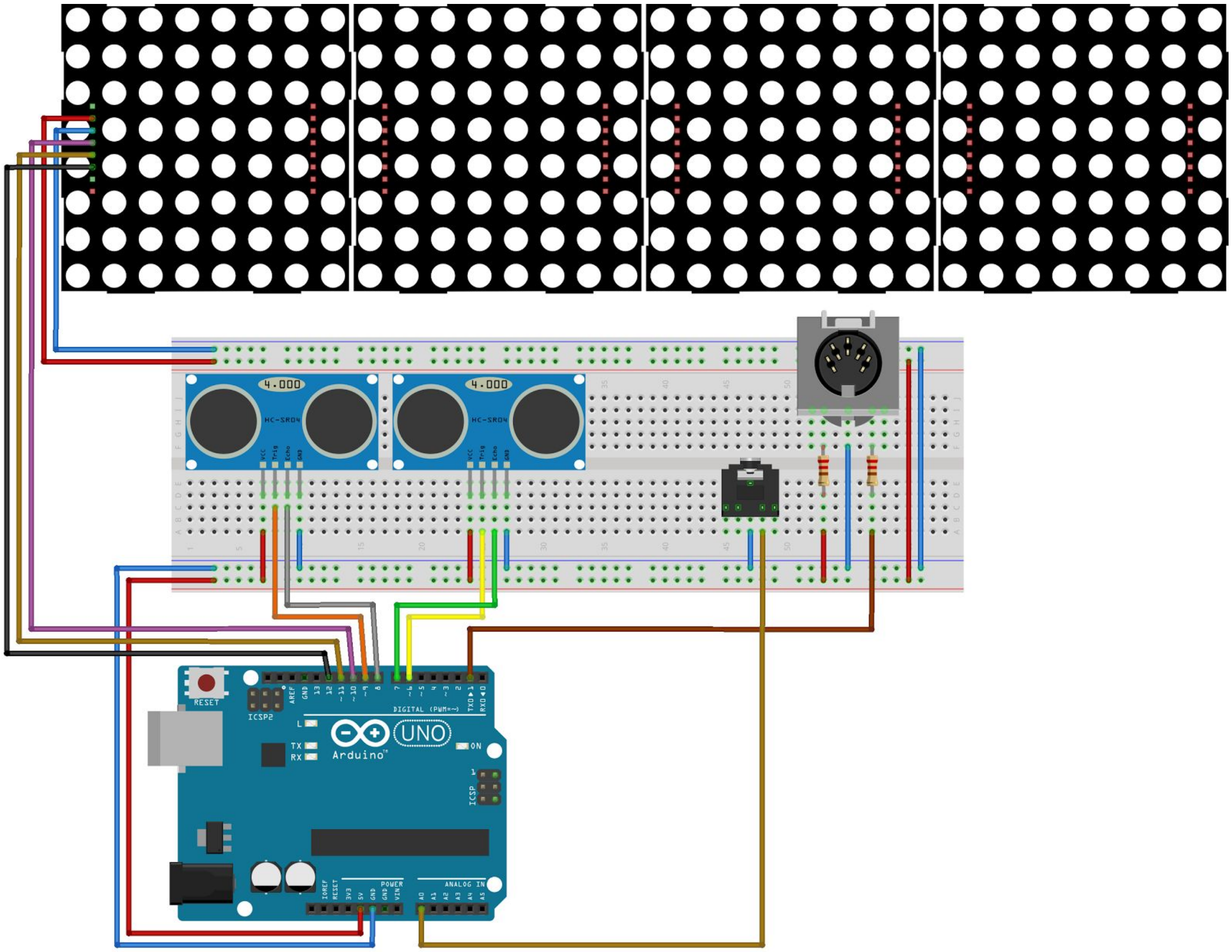
3 pin audio socket

Male to male 3.5 audio cable

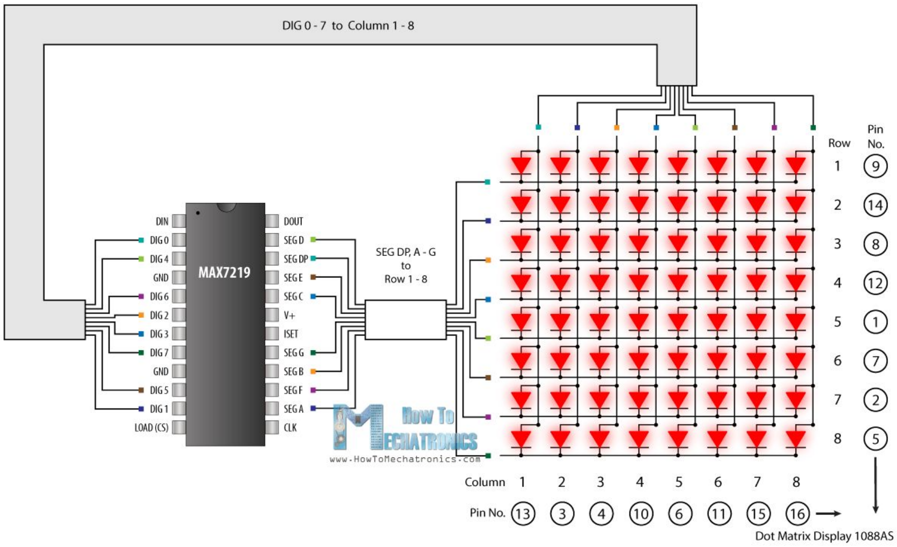
Audio splitter

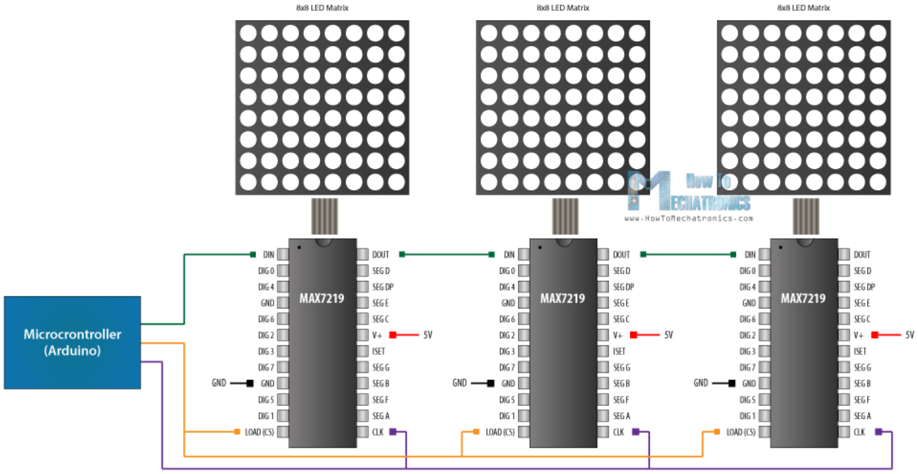
Circuit Design




4.000
HC-SR04
VCC
Trig
Echo
GND
4.000
HC-SR04
VCC
Trig
Echo
GND
RESET
ICSP2
AREF
GND
13
12
~11
~10
~9
8
7
~6
~5
4
~3
2
TX▶1
RX◀0
DIGITAL (PWM=~)
L
TX
RX
UNO
Arduino
ON
ICSP
IOREF
RESET
3V3
5V
GND
GND
VIN
POWER
ANALOG IN
A0
A1
A2
A3
A4
A5

fritzing


DIG 0 - 7 to Column 1 - 8
MAX7219
DIN
DIG 0
DIG 4
GND
DIG 6
DIG 2
DIG 3
DIG 7
GND
DIG 5
DIG 1
LOAD (CS)
DOUT
SEG D
SEG DP
SEG E
SEG C
V+
ISET
SEG G
SEG B
SEG F
SEG A
CLK
SEG DP, A - G
to
Row 1 - 8
Row
Pin No.
1 9
2 14
3 8
4 12
5 1
6 7
7 2
8 5
Column 1 2 3 4 5 6 7 8
Pin No. 13 3 4 10 6 11 15 16
Dot Matrix Display 1088AS
How To Mechatronics
www.HowToMechatronics.com


8x8 LED Matrix
8x8 LED Matrix
8x8 LED Matrix
How To Mechatronics
www.HowToMechatronics.com
Microcrontroller
(Arduino)
MAX7219
MAX7219
MAX7219
DIN
DIG 0
DIG 4
GND
DIG 6
DIG 2
DIG 3
DIG 7
GND
DIG 5
DIG 1
LOAD (CS)
DOUT
SEG D
SEG DP
SEG E
SEG C
V+
ISET
SEG G
SEG B
SEG F
SEG A
CLK
GND
5V

References

Arduino MIDI Library

http://arduinomidilib.sourceforge.net/a00001.html#a9664e523b35d8b42749e7bc3ba888943

Guide to the MIDI Software Specification

http://www.somascape.org/midi/tech/spec.html#ctrlnums

MIDI Note Numbers

https://www.midikits.net/midi_analyser/midi_note_numbers_for_octaves.htm

Arduino MIDI Controller: Potentiometers

http://www.notesandvolts.com/2016/03/arduino-midi-controller-potentiometers.html

Spectrum Analyser with Arduino UNO and three MAX7219 led matrices modules

https://www.youtube.com/watch?v=4Ukd18wbMoE

Arduino / Processing Audio Spectrum Analyze

http://www.instructables.com/id/Arduino-Processing-Audio-Spectrum-Analyzer/