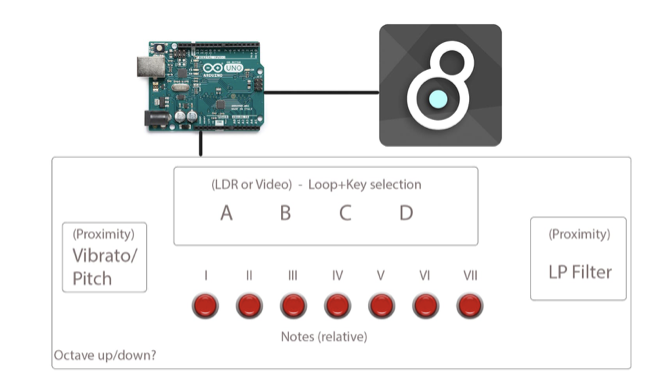
Gathering the Group

A WhatsApp group chat was set up between all members so we could communicate efficiently. We started by brain storming ideas for our presentation. A musical device with sensors and was suggested followed by a device that could be a learning tool to help with musical scales, the device change go up and down octaves to add functionality. By having this learning aspect, the device could be unique to just a midi device.

During our pitch we showed a diagram which really helped to get our point across. The feedback we received was that we should very much focus on the education aspect as it could lead to an interesting project and final design.

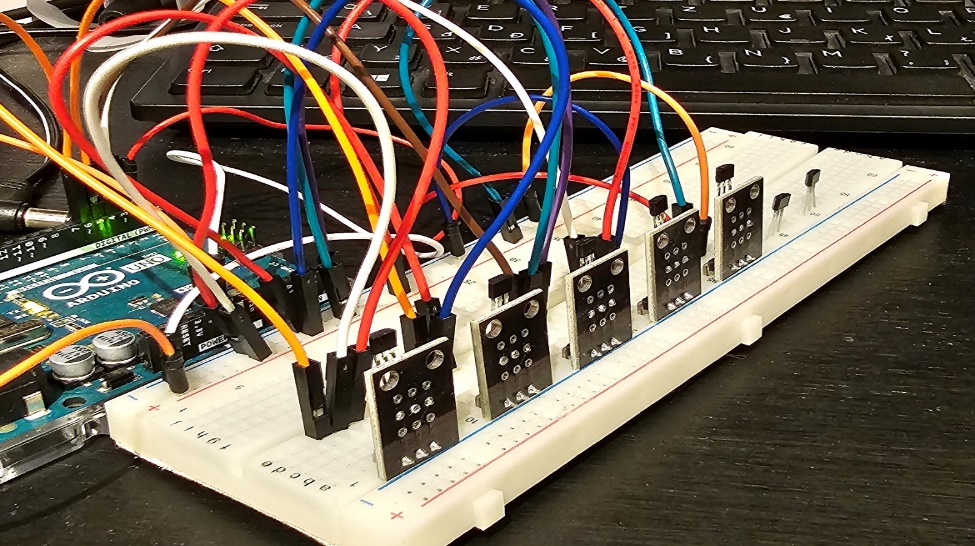


Recognising our feedback, we believed having a learning tool for musical scales, which doubled as a synthesizer, and that could be experimented with both in hardware and software relatively easily.

We decided to opt for a magnetic synthesiser for users to interactive with. This would make use of hardware sensors which pick up changes in magnetic fields. These changes can be picked up and translated into digital data types we can use to create sound. Creating a sense of novelty around the project

The Build

When the first batch of magnetic sensors arrived although they were sold as linear, they were all bipolar sensors. We wanted to have linear sensors for expressive playing experience instead of just on or off. We could still use one of them as a trigger for a percussive sound.

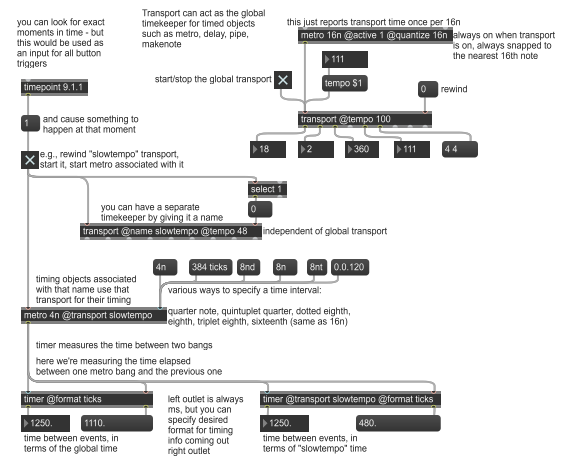


We also wanted to have manual control over certain parameters of the synth such as waveform, cutoff frequency, and modulation. For these inputs we used potentiometers, an expression pedal (that really is just another potentiometer) an ultrasound sensor and an LDR.

After considering all the inputs it was clear that 1 arduino was not enough and borrowing another one was easier than researching how to expand it with i2c modules. The next challenge was to modify the arduino code and the max patch to allow 2 microcontrollers to communicate with max through serial ports

Fun&Games

The idea for the timing game was that something basic like a 4x4 loop would play, and then stop. A counter would begin counting in time with the previously played loop, if the operator played a sample through our inputs within a certain time frame the counter would register it as being "in time".  The counter was to be visualised through jitter objects where the user could see a literal count of the tempo which would serve as a guide for them to follow.



The above patch was the starting point for the timing game design, it doesn’t do anything musical, simply measures the time between two events. The start of the patch has a start for the global transport which turns on and off the overall clock, this acts as a global Bpm or tempo for the patch. As you can the first metro is set to the 4th note and the second to the 16th. the 16th note is equivalent to 120 ticks. The timer object seen at the bottom of the patch is used to measure the time between the bang received in the left inlet and the bang in the right inlet, the time measurement is outputted as milliseconds but this can be declared as ticks or bpm.

As this idea was difficult to implement alongside our existing patch, the group decided we were better suited to build upon the idea of a timing interval game where the user had to guess the notes played/to be played based upon a previously outputted note selection or melody.

Aesthetics

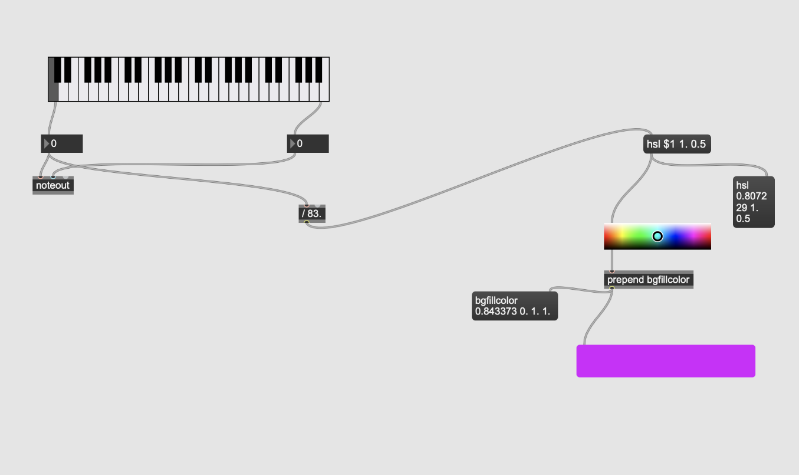
After creating the general schematic, it was time to add more depth to our project we believe that adding colour changing elements would keep the user interested when interacting with the device. The only problem we now faced was how we would get this to work. The idea was for us to have it detect different frequencies in which the colour would change on the different frequencies. We were finding this rather challenging when it came to figuring this out.

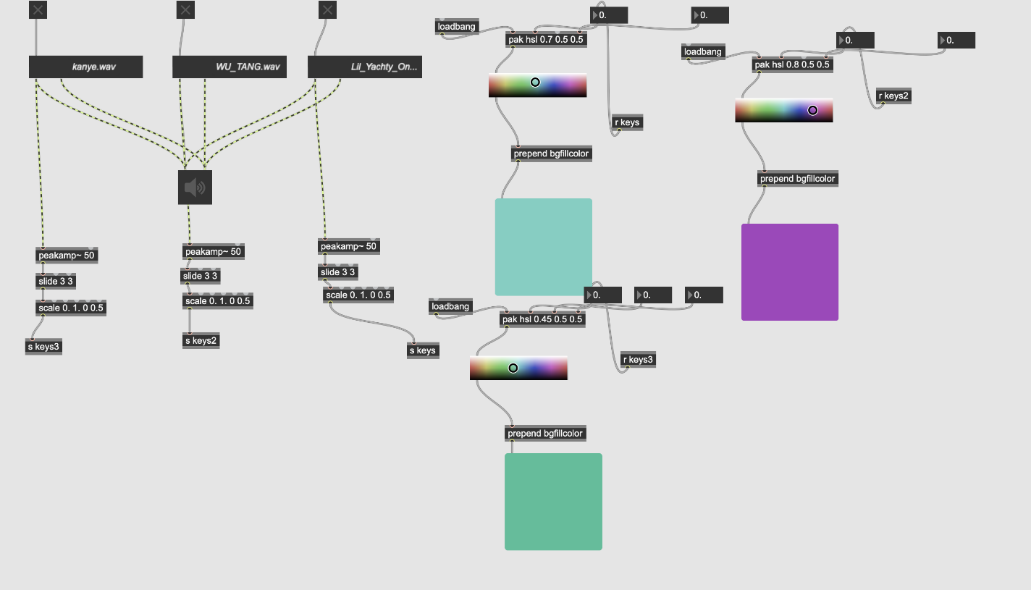
We went back to the drawing board on this. From research we found that by taking the amplitude change in the tracks we could then program using some of the max objects to do it for us by giving the conditions. It means for when the song plays the colour is ever changing with the music.

By adjusting the amplitude, we then change the brightness of the music in sync. We have given it random values that then get put into the scatter object by changing these values we can change the colour.

To have this all connected we use peak amp function. We then set it to fifty which represents seconds, it gives a value of the sample every fifty seconds we can change this if needed. To the music we use the scale as we can change the range in which we receive values again giving us customisation within the project.

We then decided to do something slightly different. We used the same premise of adding the scatter object as well as a condition to change the colour of the panel object. The hsl object has conditions that changes the saturation, brightness to actually c change the colour of the panel we take the two outputs of the midi keyboard and divide it by the Total keys on the midi keyboard eighty-three. It will then select the colour.

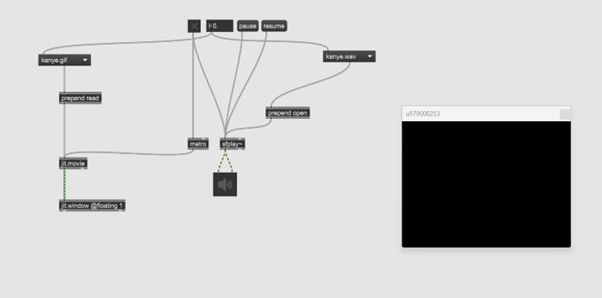




Video Elements   
  
This week it was time to introduce some visuals to our project, this would be done using jitter (video) on max8, this required some research due to our lack of understanding of jitter within max8 as we had previously only had experience in audio on max8.

We had the idea of having a visual representation of what notes are being played using the product preferably using colour or something easily recognisable. This also meant we needed some form of visual representation of the loops which were selected. We done this through max8 using jitter objects.

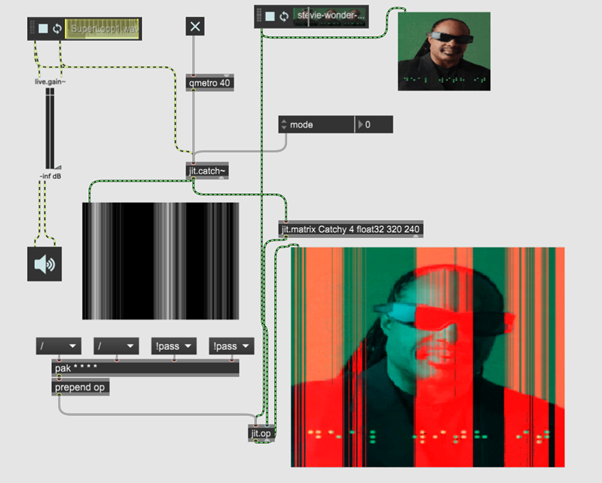
We tried to have a suitable visual going along with each loop to make the max patch more exciting rather than just what was happening, this is where we introduced jitter to the project. YouTube videos and online blogs were referenced to learn about how jitter works and how we can implement it into our patch.



This is the initial patch which gave the group an understanding of how jitter works and was a good introduction. Here we have an instrumental and a matching gif, when pressing the toggle at the top you have an instrumental play along with the maching gif, the flonum next to the toggle changes the instrumental which in turn also changes the accociated gif to go along with it. These songs and gifs were not used in the final patch as we didnt know what loops were going to be used at this point but these were just examples to help us get a start and understanding of jitter.

Gifs were used instead of videos as the music playing was a loop, and obviously, a gif is endless and would work perfectly with a musical loop as when the loop stops the gif stops, the gifs used were just gifs I thought would match the vibe of the song.

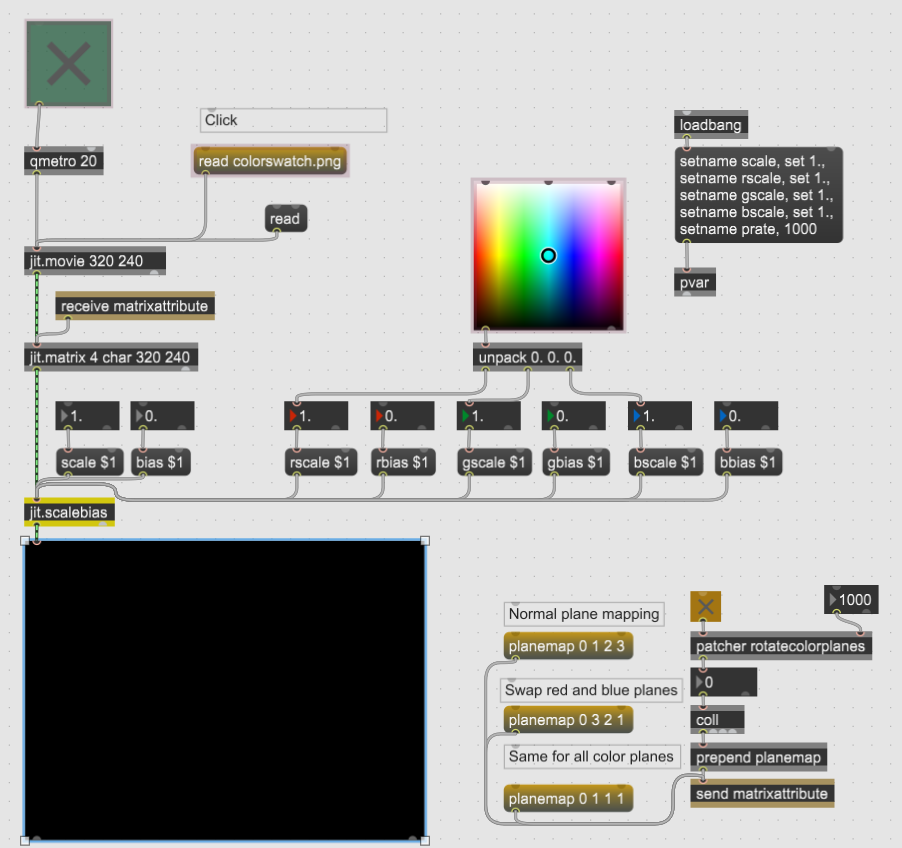
These loops were mixed with the audio using max to create a cool strobe-like effect, where the noise from the loop playing would play and was overlaying the gif to give it an interesting effect, that patch is shown below.



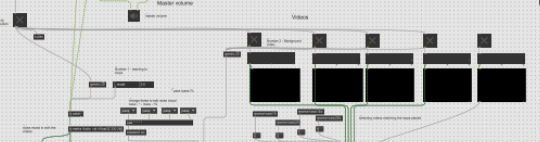
The first jit window shown on the left is the visual noise from the loop, and the other jit window at the top is just the gif, the thirst jit window shown at the bottom is the combination of both windows creating the strobe effect as mentioned above, this was done just to make the visuals look a bit more interesting.

**The musical loops were matched with their corresponding gifs on the final patch using toggles and load bangs that play the tracks, all four gifs actually play at once, but only the one matching the loop selected is playing on the presentation screen on the jit window, these gifs were played using the same signal that played the loops.**

Once consolidated we managed to achieve something that looked like this



**Alongside are the 4 jit windows which all play a different gif to match one of the 4 loops, (the 5th was an experiment and serves no purpose in final build).**



WE MADE !!!!!

After weeks of research, developing, hardware, and programming! We have a working prototype that has all our intended functions. It is housed in a cardboard box that was modified to fit all the hardware and to be small enough to protect it during transport.

Product specification:

2x Arduino Uno boards (connected to PC via USB)  
2x breadboards  
6x linear magnetic sensors 49E  
1x bipolar magnetic sensor 44E  
2x 5K potentiometers  
1x Bespeco expression pedal  
1x ultrasonic sensor  
1x LDR  
3x LED (2 of them with no function other than ambient lighting for the sensors)  
3x 200Ohm resistors (for the LEDs)  
1x 20KOhm resistor for the LDR  
1x stereo jack socket (for the expression pedal)  
1x cardboard box  
patch leads

Software used to write the Arduino code is Arduino IDE  
Max patch developed on Cycling74 MAX 8

Synthesiser functions:

The notes can be played by holding the magnet close to the magnetic sensors. The movement of the magnet controls the sound envelope. The synth has 6 discrete pitches in one key, however between the 3rd and the 4th, as well as between the 7th and the octave the pitch is continuous.  
6 notes are playable in both keys that covers an octave from A to A or from Eb to Eb. It can be extended with the use of the whammy pedal that slides the note played down 1 octave, doubling the range.

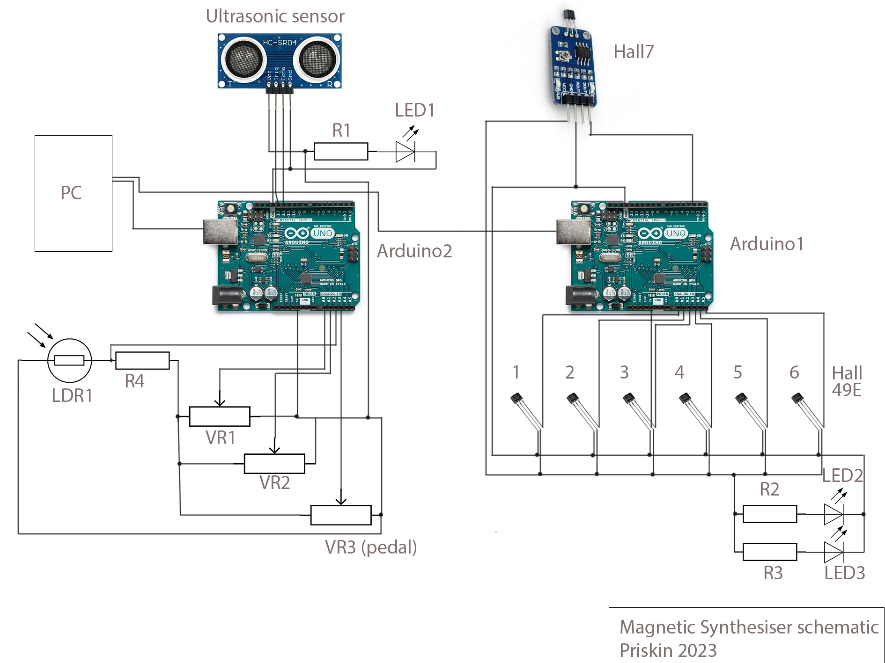
The oscillators have 6 waveforms based on the Minimoog Model D, that are selectable with the first potentiometer. The 2nd potentiometer controls the cutoff frequency of the LPF.  
The ultrasonic sensor controls the amount of pitch modulation. The red LED's brightness increases as the output value of the sensor increases.

The LDR controls the amount of the stereo flanger effect.  
The synth also has more controls accessible only on the computer such as pitch detune (by default there is detune), separate volume for the synth and the loops, a master volume for the patch, etc.

The synth has 2 modes: Jam mode and Game mode:  
In Jam mode all the counting and feedback functions are disabled. It offers a choice of 4 backing track loops. 2 in A minor 2 in Eb minor. By selecting a track it selects the key and assigns the appropriate note numbers for the 6 notes playable. In Am it is an A minor pentatonic scale: A, C, D,, E, G, A. In Eb it is and Eb minor pentatonic scale: Eb, Gb, Ab, Bb, Db, Eb. Upon starting the selected track it plays a video connected to the genre of the loop.

In Game mode the loops are disabled, Upon clicking on the interval button on the screen the synth will play the root note followed by one of the available six notes chosen randomly. The player has to guess the interval by playing the correct note on the instrument. The computer will then decide if the generated and the played notes are the same and play a negative or a positive sound accordingly and update the score. After 10 rounds the counter resets and displays the score out of 10.

Schematic



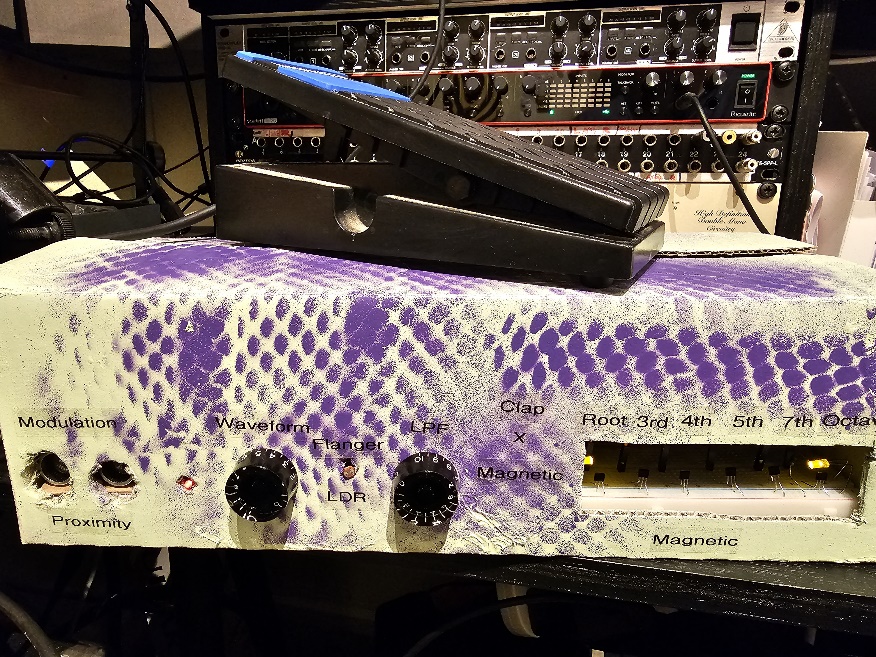
Component list   
  
R1,2,3: 220 Ohm  
R4: 20KOhm  
VR1,5: B5KOhm  
VR3: 20KOhm  
Hall sensor 1-6: 49E  
Hall sensor7: 44E  
Led 1: red  
Led 2,3: yellow  
LDR1  
Arduino1,2: Arduino Uno

Video Demonstration –

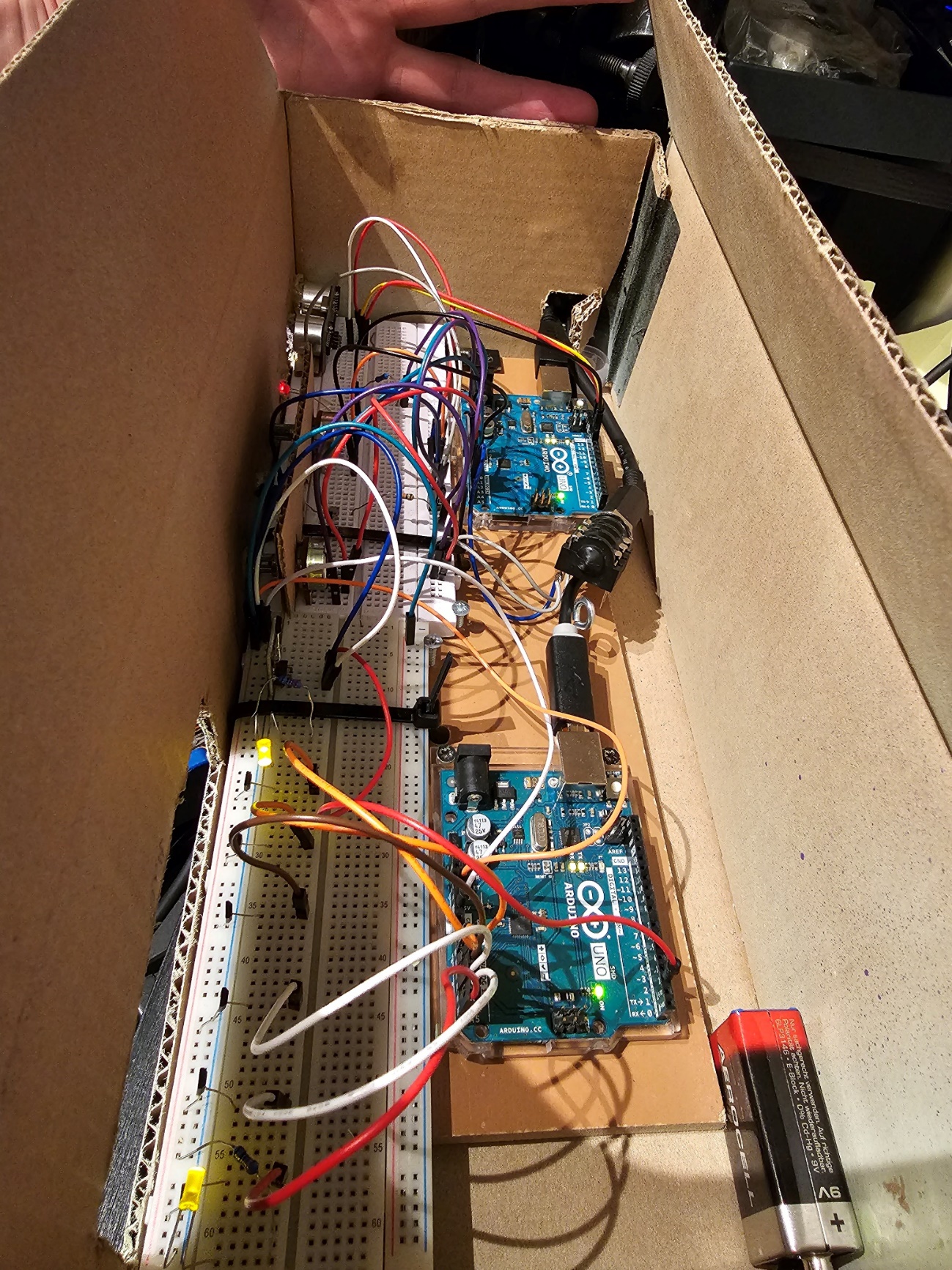
Images:

Prototype Iterations

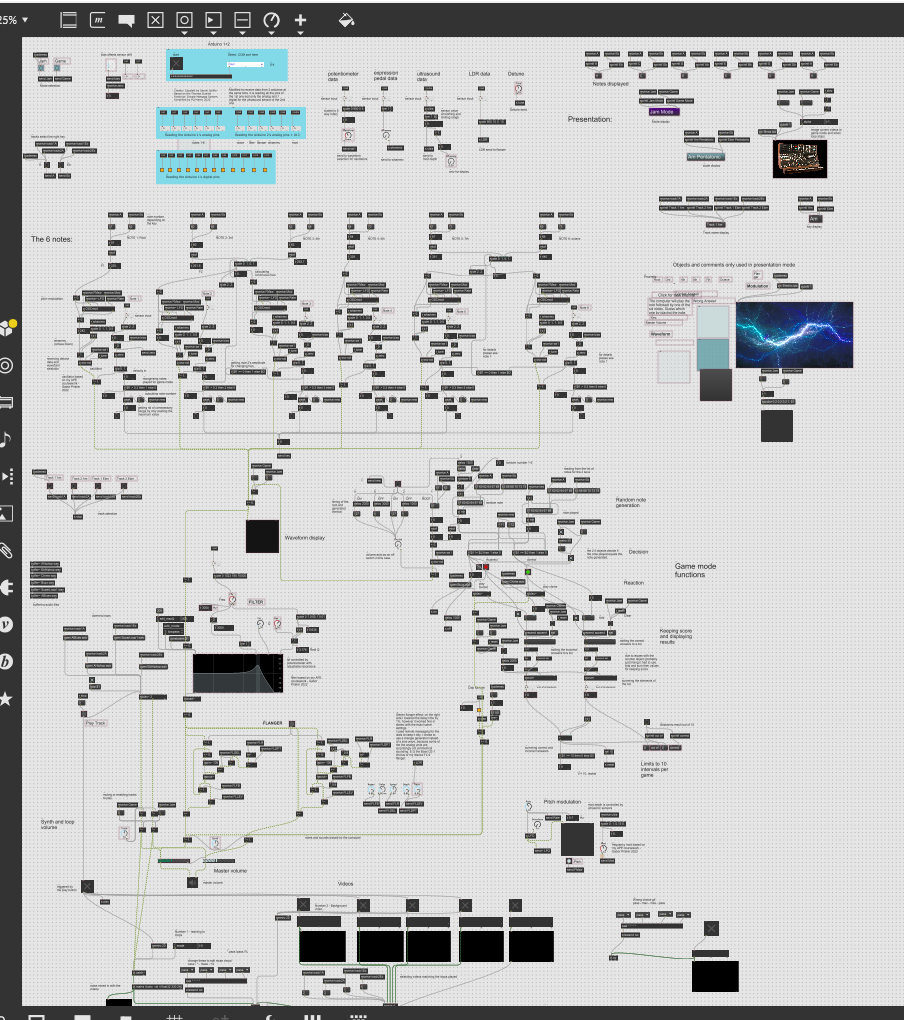


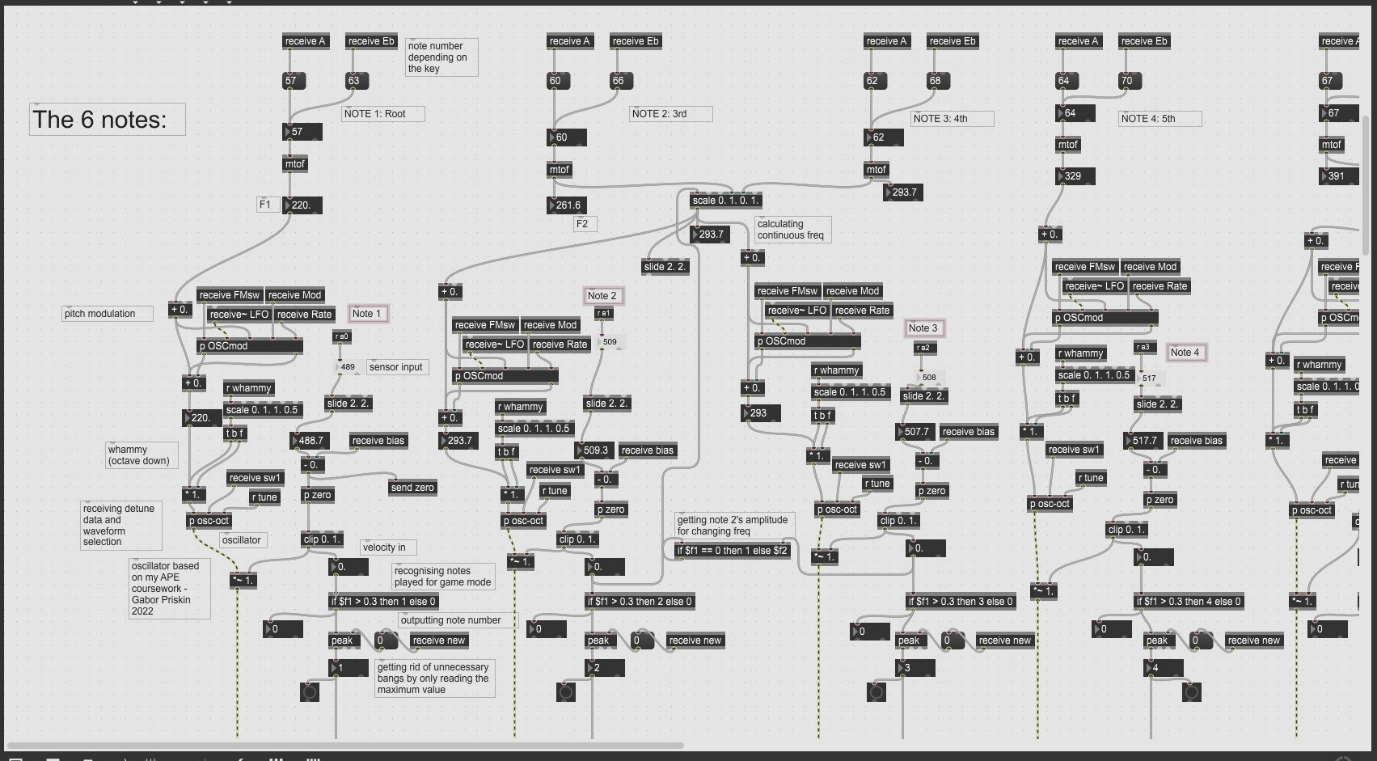


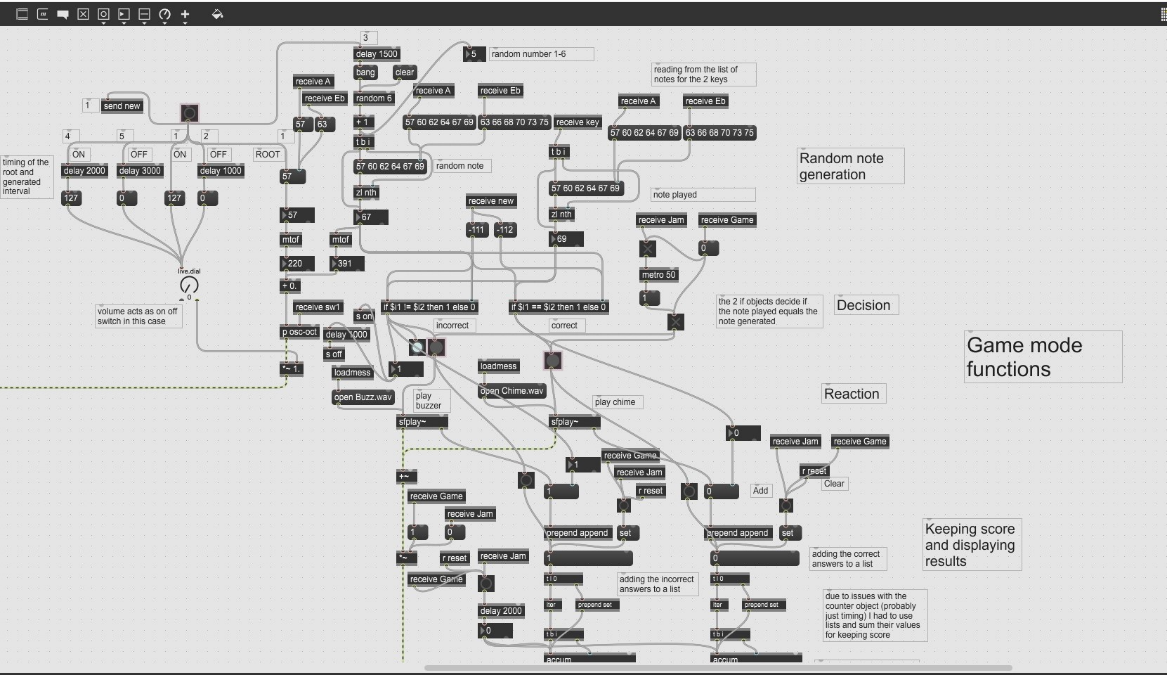
The Guts



Max Programming







References:

M3H623530 Audio and Interactivity Practical Lab 6 (Arduino code) Quinn P. Glagow Caledonian University

Max patch set for communication with Arduino: Copyleft by Daniel Jolliffe Based on the Thomas Ouellet Fredricks' Simple Message System Simplified by PQ March 2020

MAX MSP Request: Serial Arduino Live Data Stream MCKELLAR M. 2016 [video] accessed: 6.5.2023. available from: [Max MSP Request: Serial Arduino Live Data Stream - YouTube](https://www.youtube.com/watch?v=tOPObpGnrIA)

Fading LED On Breadboard with PWM - Arduino tutorial #3 Bas on Tech 2018 [video] accessed: 8.5.2023. available from: [FADING LED ON BREADBOARD WITH PWM - Arduino tutorial #3 - YouTube](https://www.youtube.com/watch?v=1fhRf-0RxyQ)

Cycling 74 MAX 8 Tutorials and examples. Cycling 74

Haken Continuum. (Inspiration from my AAI coursework 1 for using hall sensors and continuous pitch). Haken L.

<https://www.youtube.com/watch?v=s8yOsVSpDsk> - Max 8 Tutorial #26: Intro to Jitter, by dearjohnreed

<https://www.youtube.com/watch?v=t_pcOOvPWvw> - Max 8 Tuturial #33: Audio Becomes Video, by deerjohnreed

Listening references to review and attempt to draw relevant concepts.

<http://www.music.mcgill.ca/~ich/classes/mumt306/P.Elsea%27s_Max_Tutors/MaxTutors/Max&Sequences.pdf>

*(Max 8 & Sequences, Peter Elsea University of California)*

<http://www.music.mcgill.ca/~ich/classes/mumt306/P.Elsea%27s_Max_Tutors/MaxTutors/Max&Chords.pdf>

*(Max 8 & Chords, Peter Elsea University of California)*

<http://www.music.mcgill.ca/~ich/classes/mumt306/P.Elsea%27s_Max_Tutors/MaxTutors/Max&Time.pdf>

*(Max 8 & Timing, Peter Elsea University of California)*

<http://www.music.mcgill.ca/~ich/classes/mumt306/P.Elsea%27s_Max_Tutors/MaxTutors/Max&Rhythm.pdf>

*(Max 8 & Rhythm, Peter Elsea University of California)*

<http://www.music.mcgill.ca/~ich/classes/mumt306/P.Elsea%27s_Max_Tutors/MaxTutors/MSPfilters.pdf>

*(Max 8 & Filters, Peter Elsea University of California)*

Colour changing keys

<https://www.youtube.com/watch?v=TkQGoGg5VHA>

<https://www.youtube.com/watch?v=gbLRFn_-ygY>