

Tileyard Education

Affiliated with: University of Wales Trinity Saint David

Student ID:
2006389

MA Commercial music production

LEVEL 7

2020-2021

Module Code: ADAD7005_LON

Element 2_2B - Essay

Word Count:4319

Title

SNAIL TRAIL SYNTH:

The discussion and build of a software synth blended with a hardware controller.

Acknowledgments

I would like to acknowledge the contribution and assistance of my friend David Cervinka, who was a vital and integral part of this build. He played a crucial role in solving problems particularly with the 3D printing process and coding.

Table of Contents

Title	2
Acknowledgments.....	3
Introduction	6
Introduction to the Snail Trail Synth (STS)	7
Research	9
What the F is FM?	9
Understanding Synths & The Role of Frequency	12
Introduction to an Arduino (microcontroller)	12
Coding	14
3D Printing	15
Design & Build.....	17
Prototyping Phase.....	17
Phase I.	17
Phase II	18
Phase III.....	18
Phase IV.....	19
Phase V	19
Phase VI.....	20
Discussion.....	21
Conclusion	24
Bibliography	26
Appendix I: STS Manual.....	29
Appendix II: Please see attached Video: Copying first synth to the other seven for uniform sequence	30
Appendix III: Please see attached Video: Design all 8 synths to individual sounds to sequence	30
Appendix IV: Please see attached Video: Shows the algorithm connection of the carrier and modulator.....	30
Appendix V: Sketches and 3D models showing parts housing.....	31
Appendix VI: Parts & Tool List	34
Parts.....	35
Tools	35
Appendix VII: Please see attached Video & Picture below.....	37
Appendix VIII: Sketches and 3D models showing synth box design	38
Appendix IX: Sketches of button housing	41

<i>Appendix X: Pictures of test and scrap parts</i>	<i>43</i>
<i>Appendix XI: Please see attached Video: Below are pictures of the wiring process.....</i>	<i>45</i>
<i>Appendix XII: Please see attached PDF: Code.....</i>	<i>48</i>
<i>Appendix XIII: Please see attached video: Walkthrough of the code.....</i>	<i>48</i>
<i>Appendix XIX: Please see attached PDF: Mapped out midi in Hz.....</i>	<i>48</i>
<i>Appendix XX: Picture of first Generation (Green) & Second Generation (Red) synth</i>	<i>49</i>
<i>Appendix XXI: Please see attached video: Octave selection</i>	<i>50</i>
<i>Appendix XXII: Please see attached video: Key selection.....</i>	<i>50</i>
<i>Appendix XXIII: Please see attached video: Reverse sequence & pulsing purple lights</i>	<i>50</i>
<i>Appendix XXIV: Please see attached video: Example of the pentatonic scale being played</i>	<i>50</i>
<i>Appendix XXV: Please see attached video: Buttons used in playback mode.....</i>	<i>50</i>

Introduction

The purpose of this essay is to explain my rationale for building and designing a FM synthesiser. The first half of this essay focuses on the *Research Phase* of the project while the second half discusses the *Design Process* and *Actual Build*, ending with a *Discussion* and *Conclusion* of my experience with the project.

When I conceptualised this project, my objective was to create a working, digital FM [frequency modulation] synthesiser that has an easy-to-use hardware controller built on top. In addition, I wanted to use this synthesiser like a midi controller, whereas once you have designed your required sound, you could easily use the buttons to play melodies with the scales provided. If this was not possible, the objective was to, at the very least, have a built-in variable speed sequencer.

I became intrigued with synths after I used a Korg Minilogue in one of my first productions (Minilogue - POLYPHONIC ANALOGUE SYNTHESIZER | KORG (USA), 2021). This was the first synth that I was exposed to and became familiar with. This experience made me realise that I really wanted to learn more about production and that I wanted to understand the inner workings of synthesisers.

Project STS [Snail Trail Synth] is a natural progression from my Korg experience. This project added numerous skills to my toolbox including conceptualisation, design, sourcing of parts, project management, electronic engineering, coding and sound design.

Introduction to the Snail Trail Synth (STS)

The *STS* is a digital synth with a hardware controller built on top, with a microcontroller being at the heart of it (Keim, 2019)(Gudino, 2018). This one computer calculates all the parameters that bring the sound and functionality of the digital FM synth to life.

The *STS* is a small box containing eight programmable synths, the sound can be shaped with control knobs [essentially potentiometers] to the users desired sound. Each of the eight synths has a singular carrier, which is capable of adjustment by six modulators. Appendix I illustrates the different modulators. The six modulators manipulate the parameters of:

- carrier wave
- modulator wave
- modulator frequency
- first intensity [the intensities manipulate the carrier and modulator above by increasing or decreasing the strength of the effects]
- second intensity
- modulation speed.

The problem with an FM synth is that the sound appears to be quite random and is not easy to save. This makes it difficult, if not impossible, to recreate a sound again (Schwartz, 2012).

In addition, there is an eight-synth sequencer programmed into the *STS*. The user can program the first synth sound and copy it to the remaining seven synths, enabling the user to have a sequence of the one sound [See Appendix II]. Additionally, the user can create an individual sound with each of the 8 synths and sequence them [See Appendix III].

There are two modes built into the *STS*, '*Edit mode*' and '*Playback mode*' [See Appendix I].

The selector knob allows the user to choose one of the eight synths for editing, mentioned above. Once the user has found their required sound, the user can play melodies with six playable buttons and change the octave and scale with two function buttons [See Appendix I].

In *Playback mode* you can use all eight buttons to trigger any synth during the sequence. To control the speed of the sequence you use the potentiometer on the top line, second from the right [Refer to Appendix I for more detail].

Research

The reason why I chose an FM synthesizer is that in my opinion, it is an extremely diverse form of synthesis, which gives the user the ability to create a range of sounds. Anything from warm key sounds to lush, modulated pads, to hard-hitting wobbly bass sounds and the weirdest noises can be created with this instrument.

Many synthesizers have some form of FM features built into them (Schwartz, 2012). Unlike the original FM synth DX7, the modern versions [such as FM8 or Dune 2] allow users to quickly familiarize with the instrument (FM Synthesis Explained - What is FM Synthesis?, n.d)(Crute, 2019). In my opinion, FM is the most creative form of synthesis and is why I set out to build an FM synth for myself, enabling me to better understand this field of sound design.

What the F is FM?

An FM synth is an acronym for *frequency modulation synthesis* and is defined as altering the modulation of sound wave/timbre with another. The sounds are created by an oscillator, which can be induced in an electrical [analogue] or digital manner (FM Synthesis Explained - What is FM Synthesis?, n.d).

Fundamentally, both analogue and digital use the same principles of creating an FM sound wave. John Chowning invented FM synthesis at Stanford University in the late 1960's (Hahn, 2020). It uses the principle of frequency modulation to create its unique sound.

Frequency modulation is not particularly interesting on its own, but when altered and modified the sound becomes more useful and captivating (FM Synthesis Explained - What is

FM Synthesis?, n.d). The simplest form of Frequency Modulation is vibrato, where an LFO [Low Frequency Oscillator] ‘modulator’ is used to periodically change the pitch of another ‘carrier’ signal (LANDR, 2016). The speed of the vibrato effect is determined by the frequency of the LFO with the amplitude of the frequency giving it a tremolo effect (Schwartz, 2012). The intensity is in turn determined by the modulator LFO level (Hahn, 2020). Majority of these LFO’s are in the sub audio range, which are inaudible for the human ear.

This is essentially the same process that takes place within an FM synthesizer, although there is a slight difference. The pitch of the modulator is increased to the point where the effect no longer sounds like vibrato, but morphs into a completely new sound (Hahn, 2020)(LANDR, 2016).

FM synths are based on four main components: *operators, algorithms, carriers and modulators*. With additional ratio and envelope components.

- **Operators** are small, tiny synths with their own oscillator and envelope, but only use basic wave oscillator shapes [sine, saw tooth, square, etc.] (FM Synthesis Explained - What is FM Synthesis?, n.d.)(Huang, 2018).
- **Carriers and Modulators** are important to FM synths since you require at least two sound sources. One that does the modulating [the *modulator*] and the other that is being modulated [the *carrier*] (FM Synthesis Explained - What is FM Synthesis?, n.d.)(Huang, 2018).

- **Algorithms:** these can get complex quite quickly, but fundamentally the algorithm is the arrangement of multiple carriers and modulators. [See Appendix IV] (Huang, 2018).

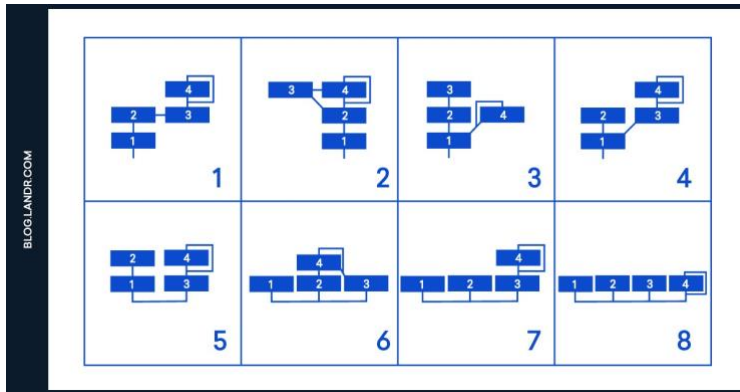


Fig.1 - FM Algorithms (LANDR, 2016)

In box 5 in the figure above, *Carriers* 1 & 3, each have their own operator modulating them. Setting these *modulators* to different parameters, will give the sounds a different *timbre* and the *carrier* will adjust the blend.

- **Envelope:** if amplitude of the waves did not alter over time, the sound would be quite stagnant. Within FM we use the envelope to control the level of the modulator. This is a common technique (LANDR, 2016).
- **Ratio:** FM synthesis is unpredictable and random. Modulated frequencies create a chaotic sound, which is difficult to understand. Instead, we use multiple *carriers*, making it easier to anticipate the sounds. The general rule is, the higher the ratio, the more complex the harmonics will be (LANDR, 2016)(FM Synthesis Explained - What is FM Synthesis?, n.d.)(Huang, 2018).

Understanding Synths & The Role of Frequency

In the audio field, synths are sound wave producers. The sound waves are created through an oscillating wave with the frequency being related to the period of oscillation [For example, 20Hz, meaning the waveform completes 20 cycles per second]. The amplitude and rapidness of a frequency creates a perception of pitch with the human ear only being able to hear between 20 Hz and 20,000 Hz (Walker, 2020). This means that notes [pitch] can be assigned to a specific Hz value. For example, to create the lowest note on a piano an oscillator has to generate a repeating waveform of $27\frac{1}{2}$ per second (27.5Hz) to get to the same note (Schwartz, 2012).

Though it may seem abstract to refer to a note [pitch] as frequencies there is a correlation in using the term frequency on a synth. This is due to the use of 'frequency' knobs to alter pitch. The amplitude [size of the wave] affects the level of the signal, and therefore some frequencies can be perceived as louder (Huang, 2018)(Schwartz, 2012).

Introduction to an *Arduino* (microcontroller)

An *Arduino* is a microcontroller; there are multiple types available in the market, each having different functionalities. For my synth, I used the *Arduino Mega 2560*, which is essentially the foundation and computer brain that powers the instrument. It is the bridge connection between the physical [potentiometers and buttons] and the digital [code] world (Arduino - Introduction, n.d.)(I Like To Make Stuff, 2015).

What is a microcontroller? It is a small computer processor, mounted on a board, with components that handle the input and output. This enables the user to assign controls, in the form of knobs and buttons on the synth. The microcontroller processes this data and via the code, it knows what frequency [notes], to produce with this data.

The advantage of the *Arduino* microcontroller is that it is open-ended; the implication is that you can have any input controlling any output, which gives the user the flexibility to design the functionality of the synth to one's own preference (Arduino - Introduction, n.d.).

During my research, I came to understand that the *Arduino* was the go-to microcontroller for the majority of custom-built synths (14 synthesizer Projects, n.d.). I did not want to reinvent the wheel, so I decided to use what has already been proven to work. It is easy to use and understand, especially for someone like me, who has no previous experience in electronics. Further research allowed me to understand its functionality and how to program the *Arduino* (Arduino Synth, n.d.).

Having researched and understood what FM synthesis is and armed with knowledge of how to use a microcontroller, input parameters and convert data to sound via the *Arduino*, what I now needed was to learn how to code for the *Arduino*.

Coding

From the outset, I realised that coding was going to be the biggest learning curve for me and difficult to master. However, the *Arduino* is user friendly since it uses open-source software called *Arduino IDE* and uses coding language “C” (Downloads, n.d.)(Eastwood, 2020).

Therefore, *Arduino* was in my opinion the correct choice since there are easy to understand tutorials on their website, with a big community behind the product. It was relatively easy to find existing code, which I could copy and create my own solid code foundation. It was necessary to customise and edit most parameters and lines of code to fit my particular requirements. The initial source code was adapted from the website *Mozzi* and Github (Barrass, 2020)(pcarver/Arduino, 2013).

The *Mozzi* website provided vital help as a resource in understanding and learning how to build both the software and the hardware of the particular synth that I wanted to build. What I found most pleasing was the systematic approach. In the end I learnt how to edit the code through a combination of YouTube videos, *Mozzi* and the *Arduino* forums and tutorials (Barrass, 2020) (Banas, 2018)(Whorter, 2019).

I also had external assistance when it came to solving bigger technical problems (Cervinka, 2021, personal communications). I simply did not have the technical knowledge for all the parameters. See Appendix XII for the full PDF of the written code.

3D Printing

I needed to select the material required to house my components and microcontroller. In the end, it was an easy decision. I had access to a friend's 3D printer, which is an efficient method to design and build prototypes including the synth (3D Printing for Prototyping: Why It's Changing for Good, 2020). To keep things easy and not get way in over my head I decided to keep the housing of my synth simple by opting for a standard rectangular box, however with my choice of pattern on top I did make it a little hard for myself. Incidentally, the name Snail Trail Synth [STS] is derived from this pattern printed on the housing of the synth.

I also came to realise that almost all 3D printing websites are open source and it was easy to find and download the necessary files to design the synth, for example the knob tips to house my potentiometers (How is called such a knob / potentiometer?, 2018)(gergreko, 2020). Adjustments to the 3D print design were required to fit all the parts. Before creating the sketch of the box, all the parts [the potentiometers, LEDs and buttons] were measured for accuracy [See Appendix V for the sketches].

The CAD [computer-aided design] software I used to sketch and build out the 3D design was *Fusion 360* (Gigante, 2019). This software is very popular and has a huge user base, which is useful when looking for video tutorials and information to solve problems (Christensen, 2016)(Fusion 360 for Personal Use | Fusion 360 | Autodesk, n.d.).

As previously mentioned, I used a friend's 3D printer, so I was fortunate enough to have assistance with learning about 3D designing and printing and was able to ask for assistance with solving printing problems. This project made me realise that 3D printers are not always accurate and in particular, minor offsetting is required due to the calibration of the printer. If this is not done there is a tendency for the parts to print incorrectly (Cervinka, 2021, personal communications).

Design & Build

Overall, the project took two months from conceptualisation, prototyping, problem solving to building. The main reason for the lengthy process was the amount of time it took for me to acquire the necessary skills to achieve the goal of building my own FM synth.

Prototyping Phase

The following section will be going into detail of the different phases of the project.

Phase I.

I purchased all the components needed [LEDS, potentiometers, *Arduino*, etc][See Appendix VI for parts list and pictures] (GM electronic, 2021). I used breadboard to make a mock-up of the wiring, and to understand the functionality of each component and how they integrated together with the *Arduino* and the foundation code [See Appendix VII]. Initially, I wanted to use a midi output, but quickly realised that it would be hard to code and difficult to connect to the *Arduino*. My final decision was to use a 3.5mm output jack, since it proved to be very simple to connect. However, this decision only became apparent after prototyping the synth on the breadboard. Fortunately, I did not need to worry about the power connectors as it was built into the microcontroller, these being a 5V standard USB and an adapter for a 7-12V power supply.

Phase II

I initially conceived, sketched and envisioned my FM synth as a sixteen-note sequencer, with eight buttons, which were intended to be programmed to notes or to be used for parameter functionality [See Appendix V for the sketch]. However, later I changed my idea and decided to have an eight-note sequencer to keep it simple and effective.

Phase III

During this phase I designed and prototyped the 3D parts with help from my friend (Cervinka, 2021, personal communications). I measured the size of each part so that they would fit correctly in the casing, which would contain the parts, cables and microcontroller. I sketched out the basic shape of the synth, which ended up being a box with dimensions 168mm by 106mm [See Appendix VIII]. The most difficult task to accomplish was the button-housing component [See Appendix IX]. It all came together only after the third print.

Having said that all of the components took on average three prints to get it right. This was because the printer required calibration to print the parts correctly [See Appendix X]. I needed to print the housing box three times to adjust for the opening of the 3.5 mm jack output, 12v input and the USB power supply [See Appendix V]. During this phase the dimensions of all the parts were double-checked to ensure they fit in the housing and I found a free download for the housing [knobs] of the potentiometers (gergreko, 2020).

Phase IV

At this point I built the first prototype. It was essentially a phase for the mock-up build to make sure all the parts fit in their respective positions and to determine the wiring. During this time, I learnt how to solder and use a digital multi-meter to ensure that there were no short circuits or loose connections. This was fundamentally a trial run to check if the electronics would work as planned [see Appendix XI].

Phase V

This phase brought the code and the hardware together and was the time where I would upload the code to the *Arduino* and determine if the foundation code was functioning seamlessly. I checked whether the inputs being received by the hardware and the sound output was correct and being manipulated and modulated correctly by its corresponding input.

Here I was also able to see how far I could push the microcontroller and what extra functionality could be added to the synth. For example, all the circle of fifths major and minor pentatonic scales were uploaded. See Appendix XIX for the midi table created, mapping out all the scales in Hz. The ability for the sequencer to play both forwards and back was added, as well as the new button combinations, which I will explain later [See Appendix I]. I established the functionality for the extra LEDs, as I still kept the original design of the 16 LEDs. As mentioned in phase II, I decided to go with an 8 note sequencer, thus I had 8 LEDs to spare. For more information refer to Appendix XIII, which is a video discussing the intricacies of the code and Appendix XII for PDF of the code.

Phase VI

Here I did final tweaks, design choices and usability features. This phase was the second-generation build, which was much cleaner, especially the soldering and wiring as I had the practice from phase IV. In this build, I changed aspects of the synth so that it would be more user friendly. The first-generation synth was symmetrical in colour, the LEDs and buttons were green and white [See Appendix XX]. I changed the colours of the LEDs and buttons in the second-generation to allow the user to understand its functionality better and create a sort of colour coding [See Appendix XX].

Discussion

I originally envisaged the synth as a sixteen-note sequencer. As I already mentioned, I altered the design to an eight-note sequencer, so that each LED in the sequence would correspond with its own synth. I thought this would make it easier for the user to know which synth they were editing and where it was in the sequence. Unlike the first-generation LED colour scheme, I decided to give the middle eight LEDs the colours of blue, red, yellow and green, which is repeated [Refer to Appendix XX]. I established that this was an easy visual way for the user to know which synth in the sequence was being edited [See Appendix III].

The reason for retaining the sixteen LEDs was the fact that the housing and parts were already designed, and 3D printed. In addition, I felt that the extra 8 LEDs could be used to display other parameters such as the octave, pulse and speed of the sequencer. This was not the initial plan, but in the end, was a useful change in design.

The LEDs on the far right and far left were changed to red and used to represent *power on*. The next three LEDs going inwards are purple on each side [See Appendix XX], which I linked to the octave selector. This enables the user to easily ascertain what octave they are in. [See Appendix XXI].

Of the three purple LEDs, the middle represents the key the user is in. The left side indicates major key, and the right side indicates the minor key [See Appendix XXII]. In addition, I felt that I not only wanted the sequence to go forwards but also backwards. By adding this functionality to the synth, I felt it would open up another creative avenue for a user to hear their designed sequenced sounds [See Appendix I & Appendix XXIII]. In playback mode the

purple LEDs were programmed to pulse at the same speed as the sequencers [See the end of Appendix XXIII]. This is just a fun visual element to indicate the sequencers speed.

One of my pre-requisites was to have buttons on the synth that made the FM synth tactile and playable. Initially, I only wanted the buttons to be programmed for major and minor scales. I decided against this in the end because I wanted to increase the capability of the synth. I decided to program six of the buttons to a one octave major or minor pentatonic scale with four octaves that can be selected with the user moving through the circle of fifths [See Appendix XXIV]. Starting at C major, through all the sharp keys then all the flat keys [See Appendix XXII]. This design choice made the playability of the synth more diverse, due to the nature of the pentatonic scale. This scale has six notes meaning it only took 6 of the 8 buttons, freeing up two. I used this design modification to use the buttons for other functionalities. This was necessary as my design had limited inputs [See Appendix XX]. The functionalities that were assigned are as follows:

- Octave selection
- Key selection
- Copy

I added some button combinations to add to the functionality [See Appendix I]. While holding down both the second and first button you can use the blue button to cycle through the keys and the green button to backtrack [Refer to Appendix XXII]. Pressing the green and blue buttons simultaneously copies the sound of the first synth onto the other seven for a uniform sequence [See Appendix II]. Pressing button one and eight saves the settings of the synth. Pressing the first, second and third buttons together wipes any custom settings and

returns the synth back to its origins [See Appendix I]. In playback mode, when the sequence is running, you can press any button and the corresponding synth will play immediately and then the sequence will continue. [Refer to Appendix XXV].

When looking back there is one major functionality which was overlooked and only realised after creating the videos for the appendances. This missing functionality was a way to control master volume, which is vital for some sounds that come out of the synth as they can be quite harsh with a high gain output. Due to the focus on the discussed functions above, it did not dawn on me until the project was coming to an end and is something that would be done differently in the future.

If I were to re-do this project with the knowledge I have now I would add a small display showing the information about what parameters are being changed. This would mean making the box bigger and hopefully, this added space would also allow for 2 extra potentiometers and 2 extra buttons. This would mean less compromise like not having to use a 6-note scale [pentatonic] and being able to add a volume knob as discussed above. After more research, I have come to the conclusion that I would change the microcontroller to a more powerful one to open up more functions as the Arduino Mega 2560 reached its computing capacity. I would opt for a teensy microcontroller with an audio adaptor board add-on (PJRC Store, n.d.). The reasoning behind this choice is that it is a specific hardware for audio processing, which enables more effects and heightens the audio quality to 16bit/44.1KHz sample rate, whereas currently I have 128kbps.

Conclusion

My initial objective was to create a fully functional digital FM synth, with an easy to use, hardware controller built on top. I wanted a playable instrument, with the functionality, that once the required sound was designed, the user could simply use buttons to play notes and if possible, have built-in programmed scales. If this was not possible, I planned at the very least, to have a variable speed sequencer built in.

Considering my initial objective, I am of the opinion that not only did I achieve my goals while at the same time I added valuable skills [project management, design, coding, 3D printing, etc] to my skills toolbox. In hindsight, I could have probably improved some aspects of the design, but these design flaws only became apparent, after gaining the required skillset and building the synth.

Overall, despite three re-builds, the process went well. I feel that I achieved my objective and in fact exceeded my expectations. I ended up creating a synth that had more functionality and programmable parameters than originally planned. Initially I set out to build a FM synth that could be manipulated. The result achieved was a FM hardware controller, with eight digital synths built in, that were capable of manipulation and that were easily playable.

In addition to my new skills, I have come out of this project feeling that I have gained a deeper understanding and appreciation of music synthesizers and sound design.

I enjoyed this project thoroughly, to the point that I am now considering building or buying my own 3D printer to design and build further projects. For example, I now want to build my own drum machine. The project has opened up to me the previously unknown world of the DIY instrument community. I was pleasantly surprised at the fact that the blueprints to build all of these items are open source and freely available online, making it an accessible project for all.

Bibliography

- Arduino.cc. n.d. Arduino - Introduction. [online] Available at: <<https://www.arduino.cc/en/guide/introduction>> [Accessed 31 March 2021].
- Arduino. n.d. Downloads. [online] Available at: <<https://www.arduino.cc/en/software>> [Accessed 1 April 2021].
- Autodesk. n.d. Fusion 360 for Personal Use | Fusion 360 | Autodesk. [online] Available at: <<https://www.autodesk.com/products/fusion-360/personal>> [Accessed 1 April 2021].
- Banas, D., 2018. Arduino Programming. [video] Available at: <https://www.youtube.com/watch?v=QO_Jlz1qpDw&t=1417s> [Accessed 1 April 2021].
- Barrass, T., 2020. Mozzi. [online] Mozzi. Available at: <<https://sensorium.github.io/Mozzi/learn/>> [Accessed 1 April 2021].
- Christensen, L., 2016. Fusion 360 Tutorial for Absolute Beginners— Part 1. [video] Available at: <<https://www.youtube.com/watch?v=A5bc9c3S12g>> [Accessed 1 April 2021].
- Crute, A., 2019. Understanding the origins of FM synthesis. [online] MusicTech. Available at: <<https://www.musictech.net/guides/essential-guide/history-of-fm-synthesis/>> [Accessed 31 March 2021].
- Eastwood, B., 2020. The 10 Most Popular Programming Languages to Learn in 2021. [online] Northeastern University Graduate Programs. Available at: <<https://www.northeastern.edu/graduate/blog/most-popular-programming-languages/>> [Accessed 1 April 2021].
- Electrical Engineering Stack Exchange. 2018. How is called such a knob / potentiometer?. [online] Available at: <<https://electronics.stackexchange.com/questions/342199/how-is-called-such-a-knob-potentiometer#342201>>(source) [Accessed 1 April 2021].
- gergreko, 2020. Volume Knob StandArt. [online] MakerBot Thingiverse. Available at: <<https://www.thingiverse.com/thing:4641222>> [Accessed 1 April 2021].
- Gigante, M., 2019. What Is CAD (+The Most Used Types of CAD Software). [online] Learning Hub. Available at: <<https://learn.g2.com/what-is-cad>> [Accessed 1 April 2021].
- GitHub. 2013. pcarver/Arduino. [online] Available at: <<https://github.com/pcarver/Arduino/blob/master/AdafruitDebounce/AdafruitDebounce.ino>> [Accessed 1 April 2021].
- GM Electronic. 2021. GM electronic. [online] Available at: <<https://www.gme.cz/>> [Accessed 1 April 2021].

Gudino, M., 2018. Introduction to Microcontrollers. [online] Arrow. Available at: <<https://www.arrow.com/en/research-and-events/articles/engineering-basics-what-is-a-microcontroller>> [Accessed 31 March 2021].

Hahn, M., 2020. FM Synthesis: Everything You Need to Know to Get Started. [online] LANDR. Available at: <<https://blog.landr.com/fm-synthesis/>> [Accessed 31 March 2021].

Huang, A., 2018. FM SYNTHS in under 4 minutes. [video] Available at: <<https://www.youtube.com/watch?v=vvBl3YUBUyY>> [Accessed 31 March 2021].

I Like To Make Stuff, 2015. What's an Arduino // Maker101 | I Like To Make Stuff. [video] Available at: <https://www.youtube.com/watch?v=_h1m6R9YW8c> [Accessed 31 March 2021].

Keim, R., 2019. What Is a Microcontroller? The Defining Characteristics and Architecture of a Common Component. [online] Allaboutcircuits.com. Available at: <<https://www.allaboutcircuits.com/technical-articles/what-is-a-microcontroller-introduction-component-characteristics-component/>> [Accessed 31 March 2021].

KORG Global. 2021. minilogue - POLYPHONIC ANALOGUE SYNTHESIZER | KORG (USA). [online] Available at: <<https://www.korg.com/us/products/synthesizers/minilogue/>> [Accessed 31 March 2021].

LANDR, 2016. LFO Tutorial: How to Use LFOs Creatively to Get a Better Mix | LANDR Mix Tips #14. [video] Available at: <<https://www.youtube.com/watch?v=7Kj2-3HBreI>> [Accessed 31 March 2021].

Music Production Nerds. n.d. FM Synthesis Explained - What is FM Synthesis?. [online] Available at: <<https://musicproductionnerds.com/what-is-fm-synthesis>> [Accessed 31 March 2021].

PJRC. n.d. *PJRC Store*. [online] Available at: <https://www.pjrc.com/store/teensy3_audio.html> [Accessed 2 April 2021].

Project Hub. n.d. 14 synthesizer Projects. [online] Available at: <<https://create.arduino.cc/projecthub/projects/tags/synthesizer>> [Accessed 1 April 2021].

Projects RaspberryPi. n.d. Arduino Synth. [online] Available at: <<https://projects.raspberrypi.org/en/projects/8bitsynth/5>> [Accessed 1 April 2021].

Sacred-texts.com. n.d. The Pythagorean Theory of Music and Color. [online] Available at: <<https://www.sacred-texts.com/eso/sta/sta19.htm>> [Accessed 31 March 2021].

Schwartz, P., 2012. On Synth Terminology: Frequency (Everybody Hertz, Sometimes). [online] Ask.Audio. Available at: <<https://ask.audio/articles/on-synth-terminology-frequency-everybody-hertz-sometimes>> [Accessed 2 April 2021].

Tulfa. 2020. 3D Printing for Prototyping: Why It's Changing for Good. [online] Available at: <<https://tulfa.com/why-3d-printing-changes-prototyping-forever/>> [Accessed 1 April 2021].

Walker, L., 2020. Audio Spectrum. [online] Teach Me Audio. Available at: <<https://www.teachmeaudio.com/mixing/techniques/audio-spectrum>> [Accessed 1 April 2021].

Whorter, P., 2019. Arduino Tutorial 1: Setting Up and Programming the Arduino for Absolute Beginners. [video] Available at: <<https://www.youtube.com/watch?v=fJWR7dBuc18&t=309s>> [Accessed 1 April 2021].

Wolfe, J., 2021. Note names, MIDI numbers and frequencies. [online] UNSW - Music Acoustics. Available at: <<http://newt.phys.unsw.edu.au/jw/notes.html>> [Accessed 31 March 2021].

Appendix I: STS Manual



Button Combinations:

Scale Select:

Scale and Octave reset:

Copy synth 1 setting to all other synths:

Save settings:

Reverse step direction:

Button 1 + Button 2 + (Button 7[Scale Down] or Button 8[Scale Up])

Button 1 + Button 2 + Button 3

Button 7 + Button 8

Button 1 + Button 8

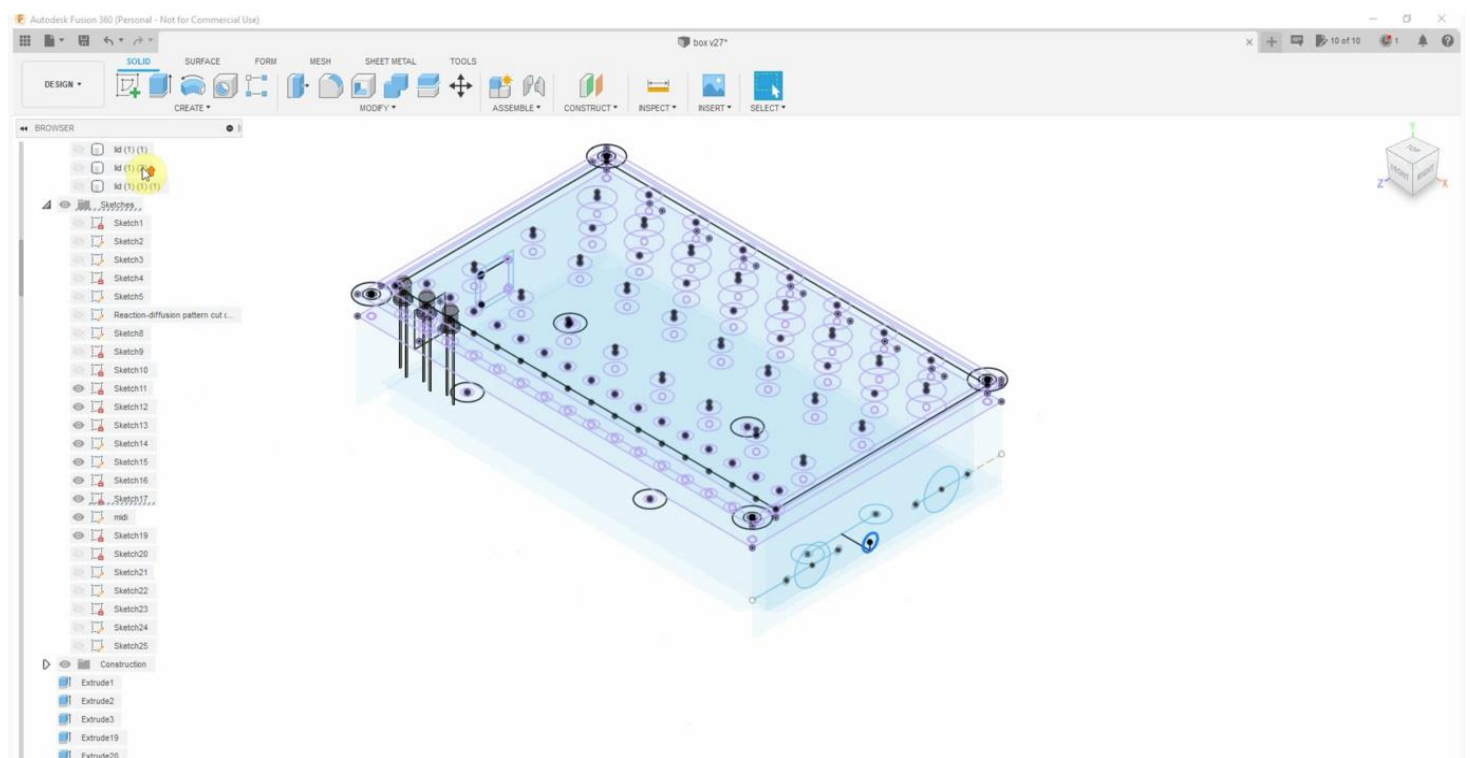
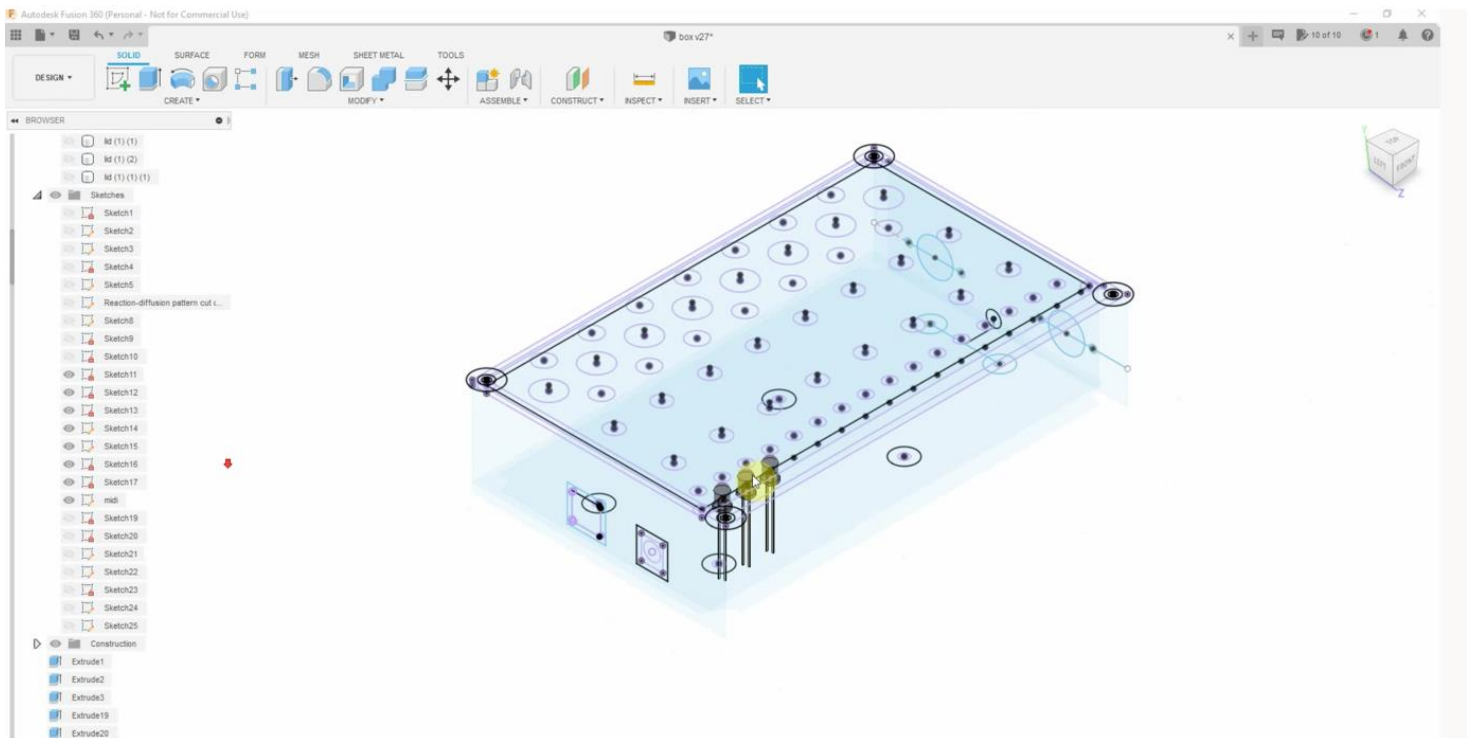
Move sequencer speed pot all the way to the right for 1 second

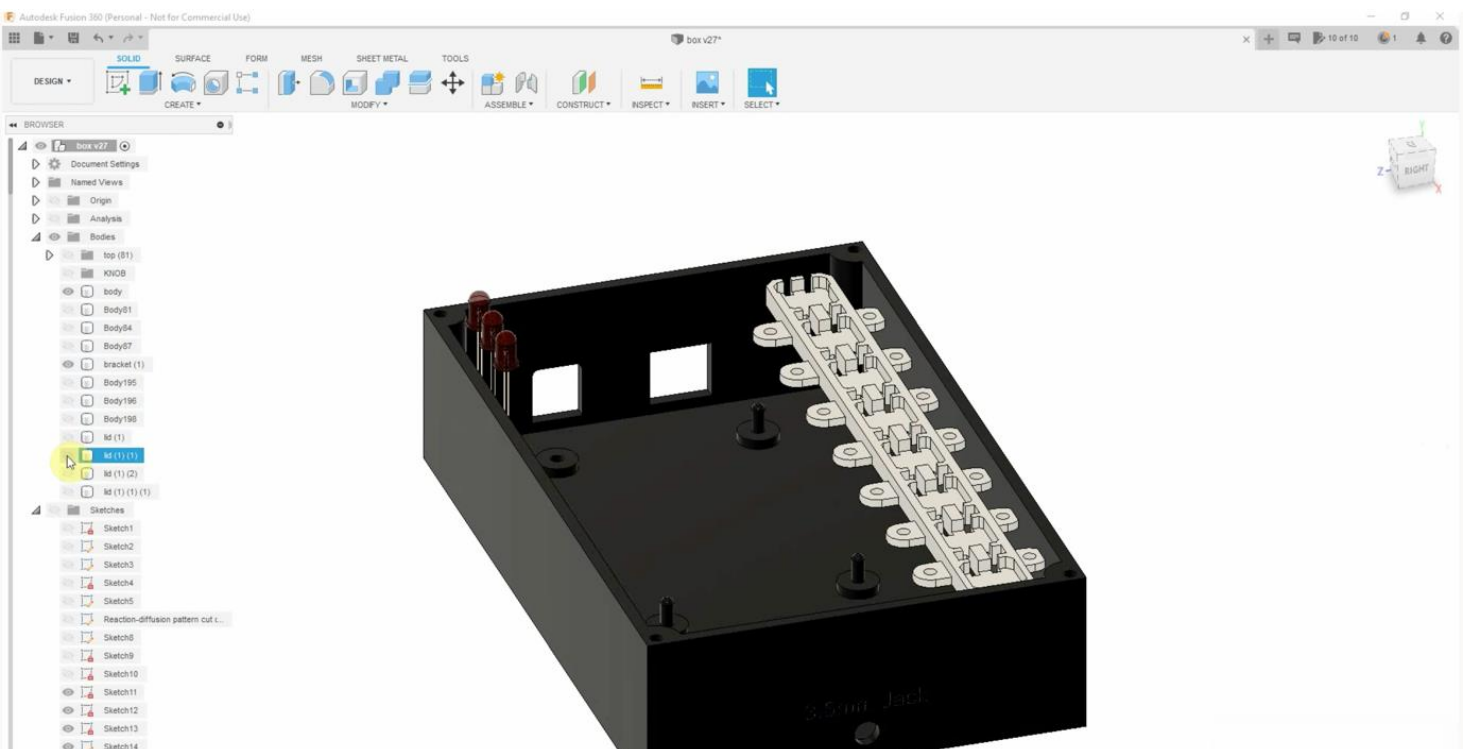
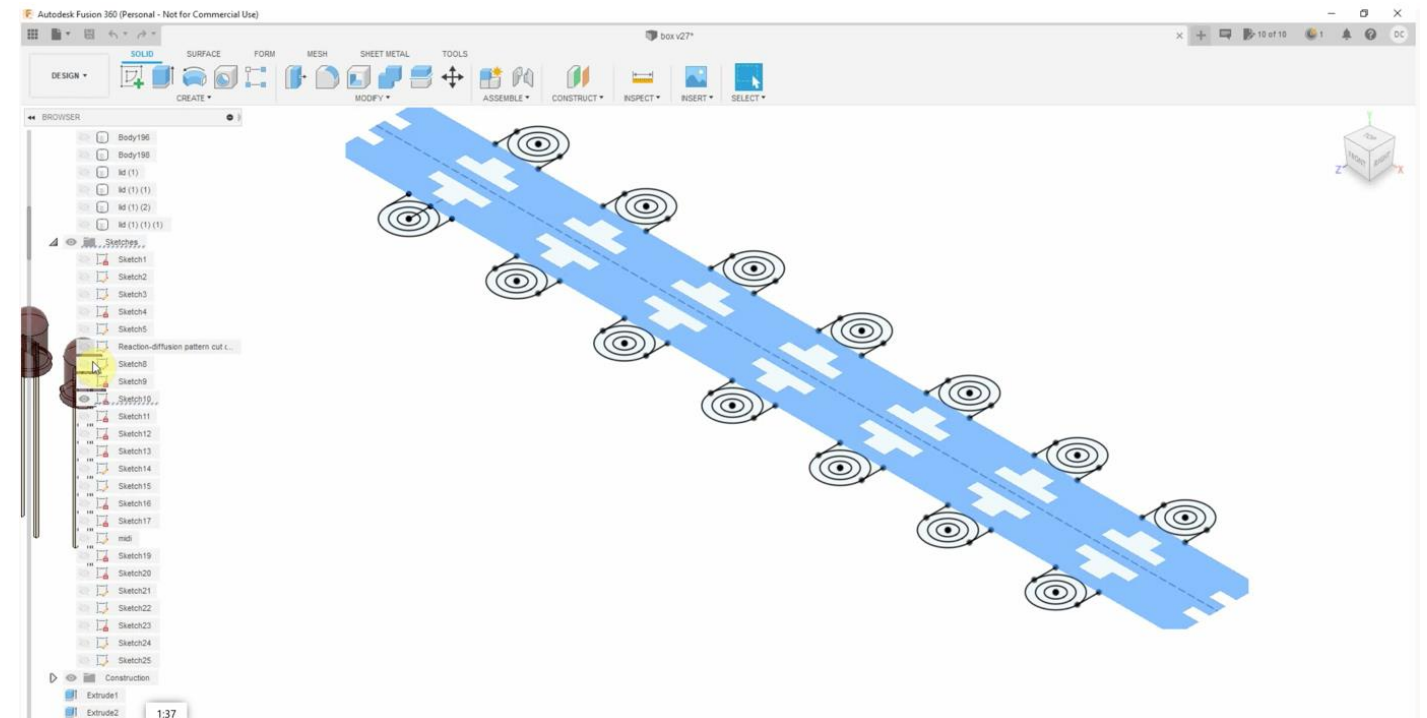
Appendix II: Please see attached Video: Copying first synth to the other seven for uniform sequence

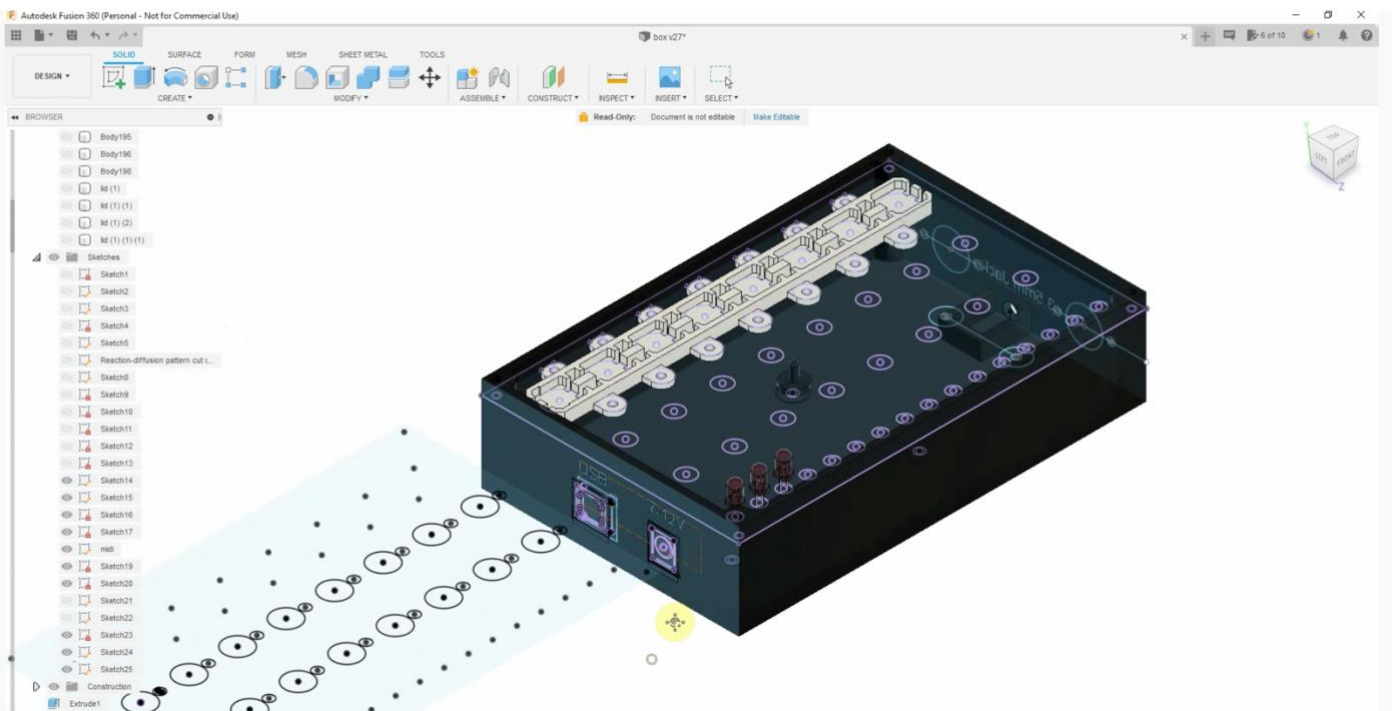
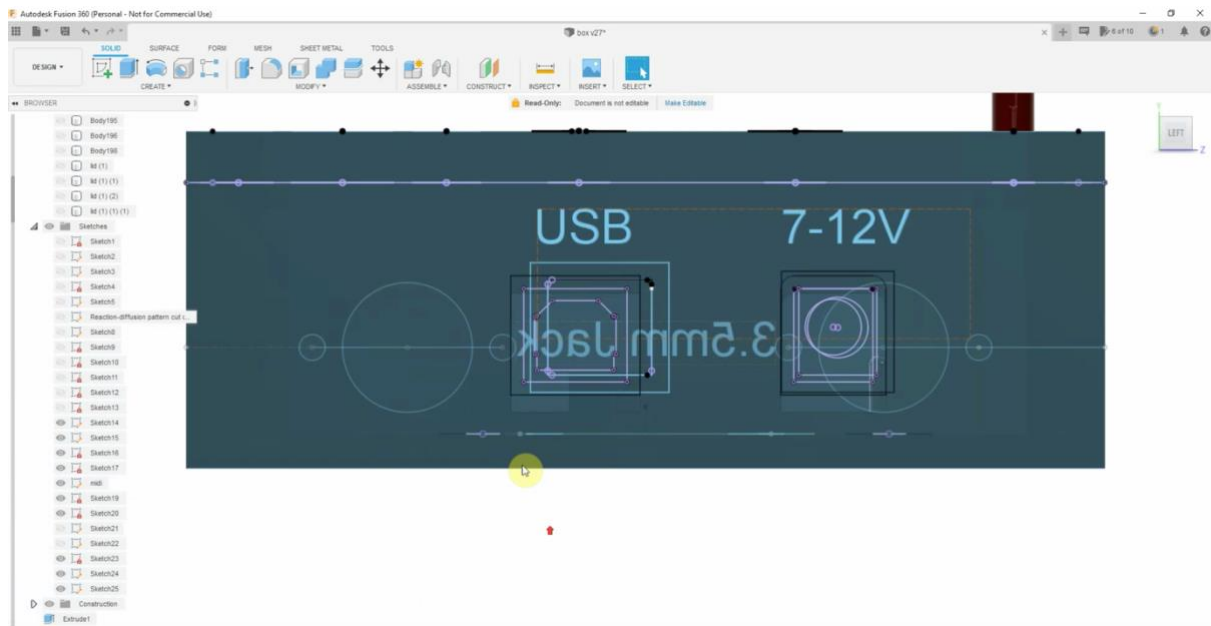
Appendix III: Please see attached Video: Design all 8 synths to individual sounds to sequence

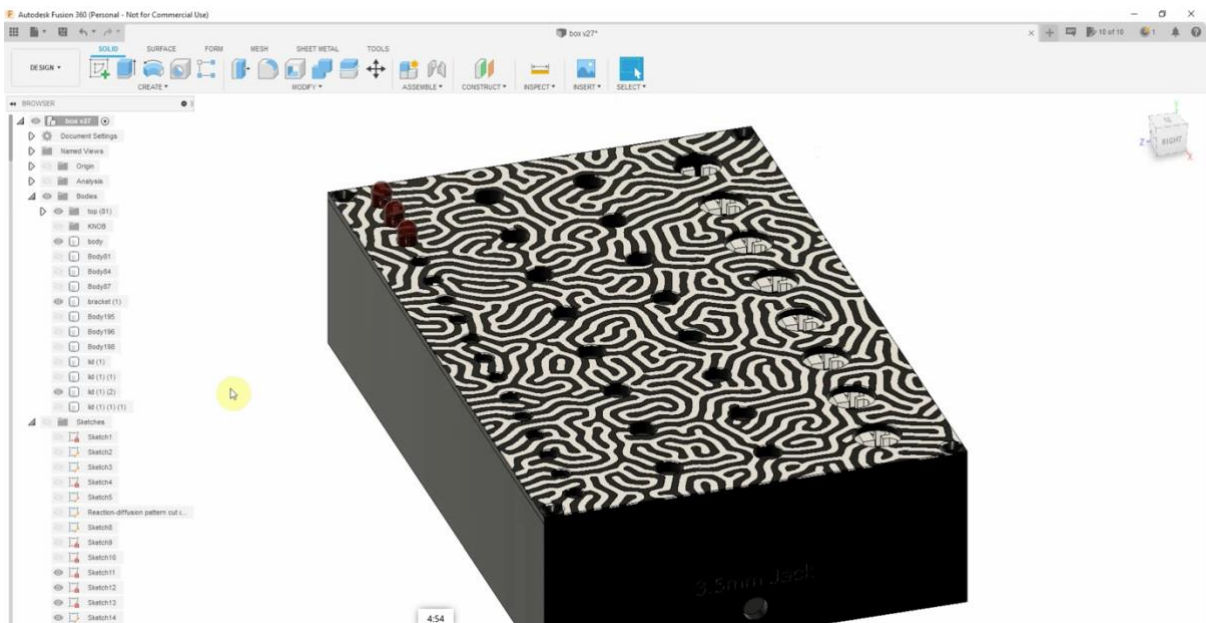
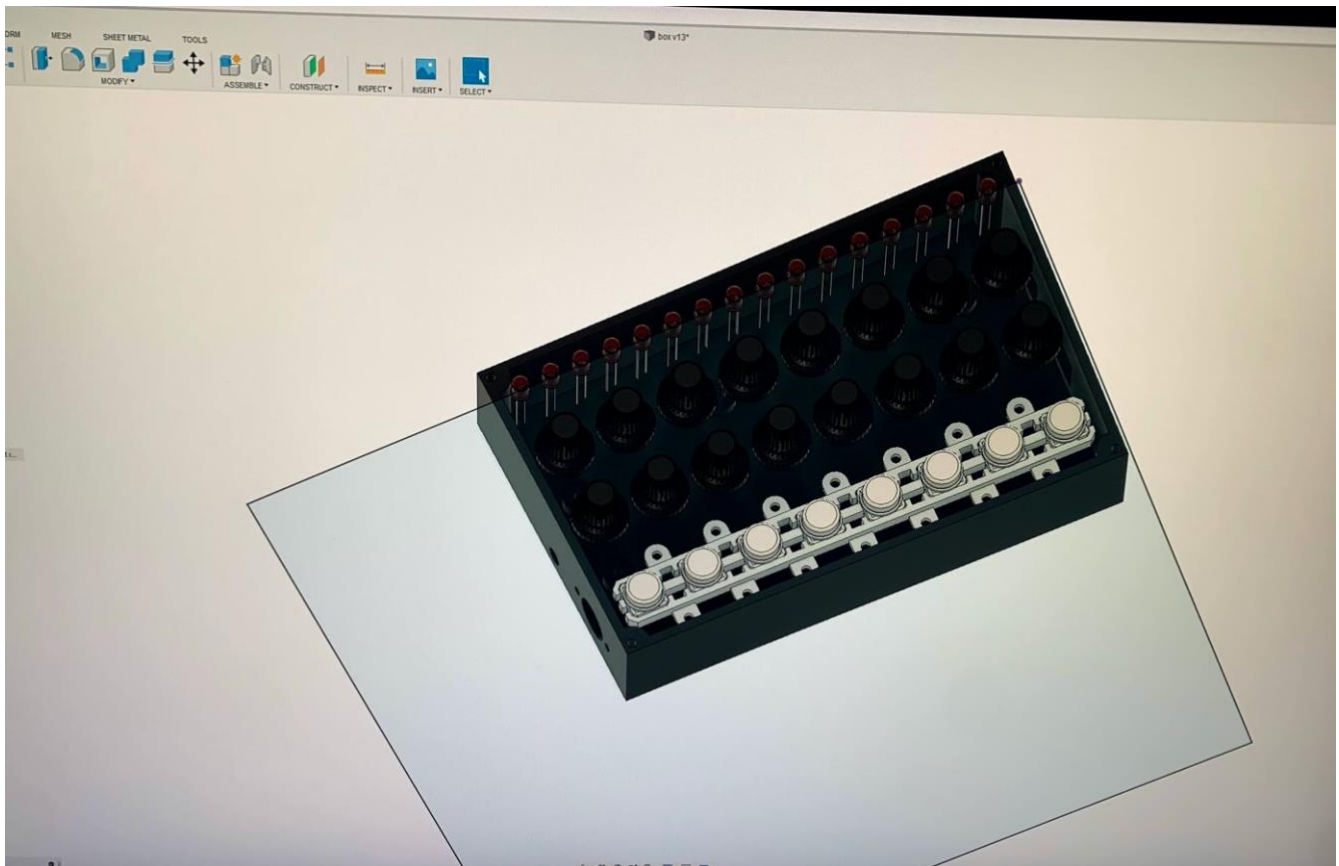
Appendix IV: Please see attached Video: Shows the algorithm connection of the carrier and modulator

Appendix V: Sketches and 3D models showing parts housing









Appendix VI: Parts & Tool List

Parts

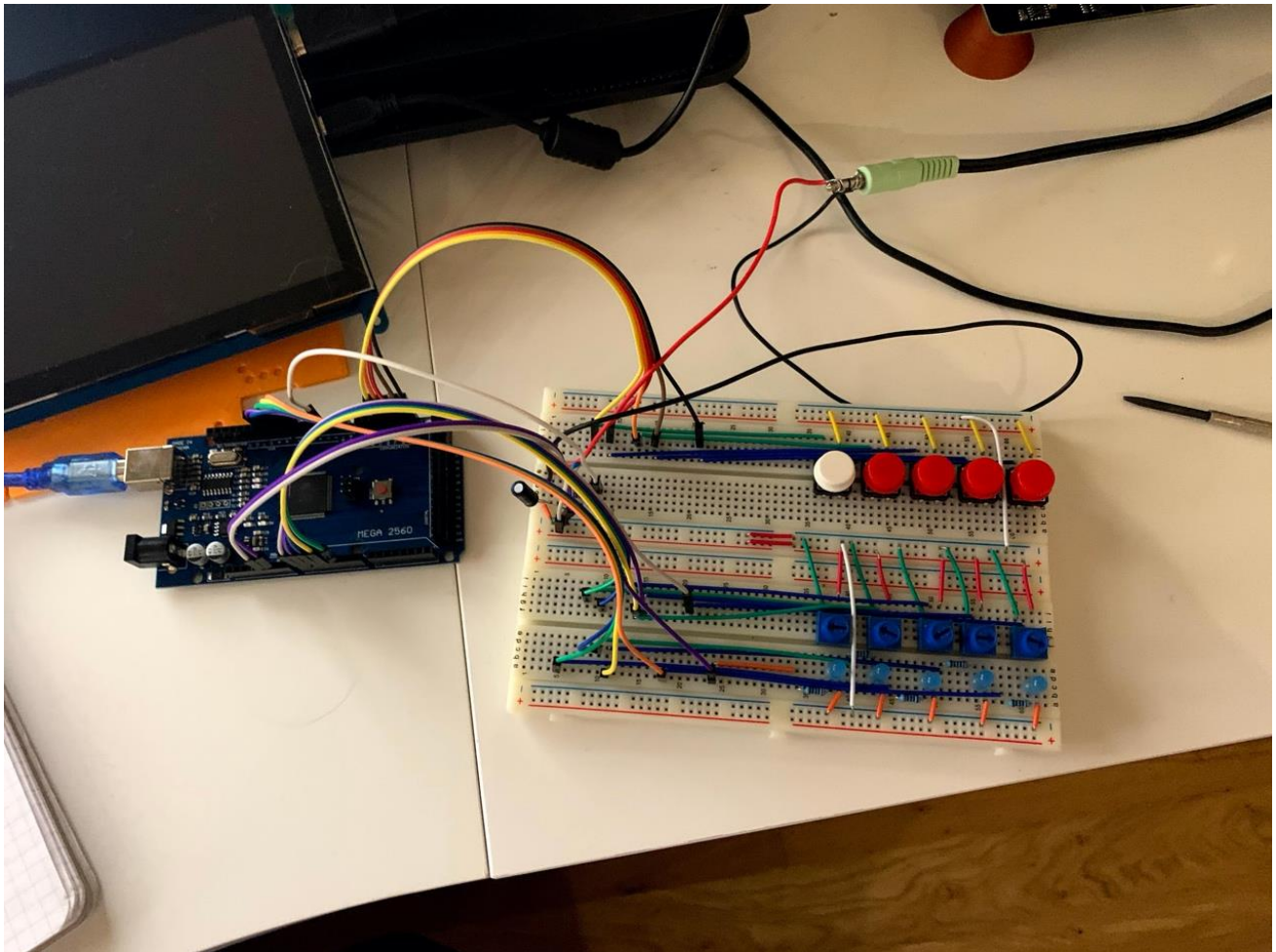
- 3 - Arduino Mega 2560
- 120 – 5mm LEDs - (20 Red, 20 Green, 20 Purple, 20 blue)
- 20 – 5k linear potentiometer. 280deg travel
- 20 – 12mm push buttons
- 20 – 280ohm resistors
- 20 – 100nF capacitor
- Insulation tape
- Bread board
- 5 – 3.5mm jack

Tools

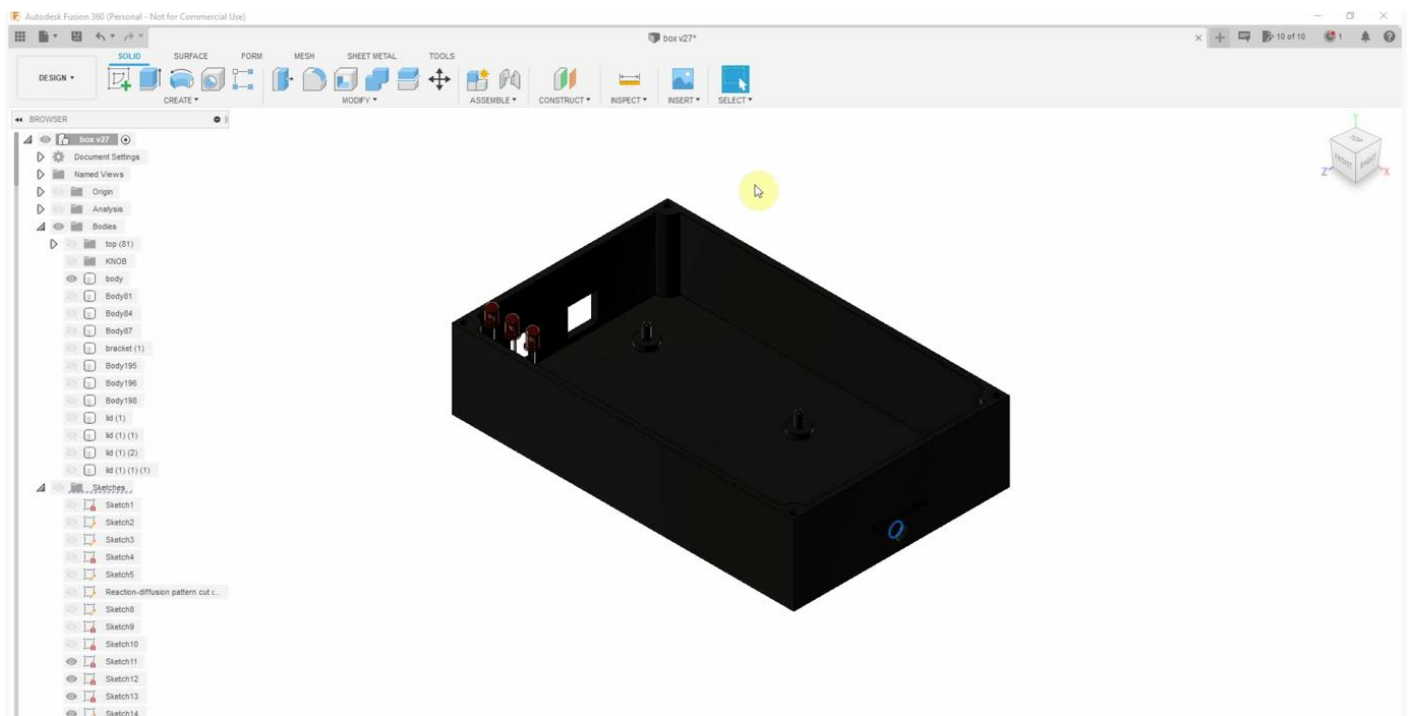
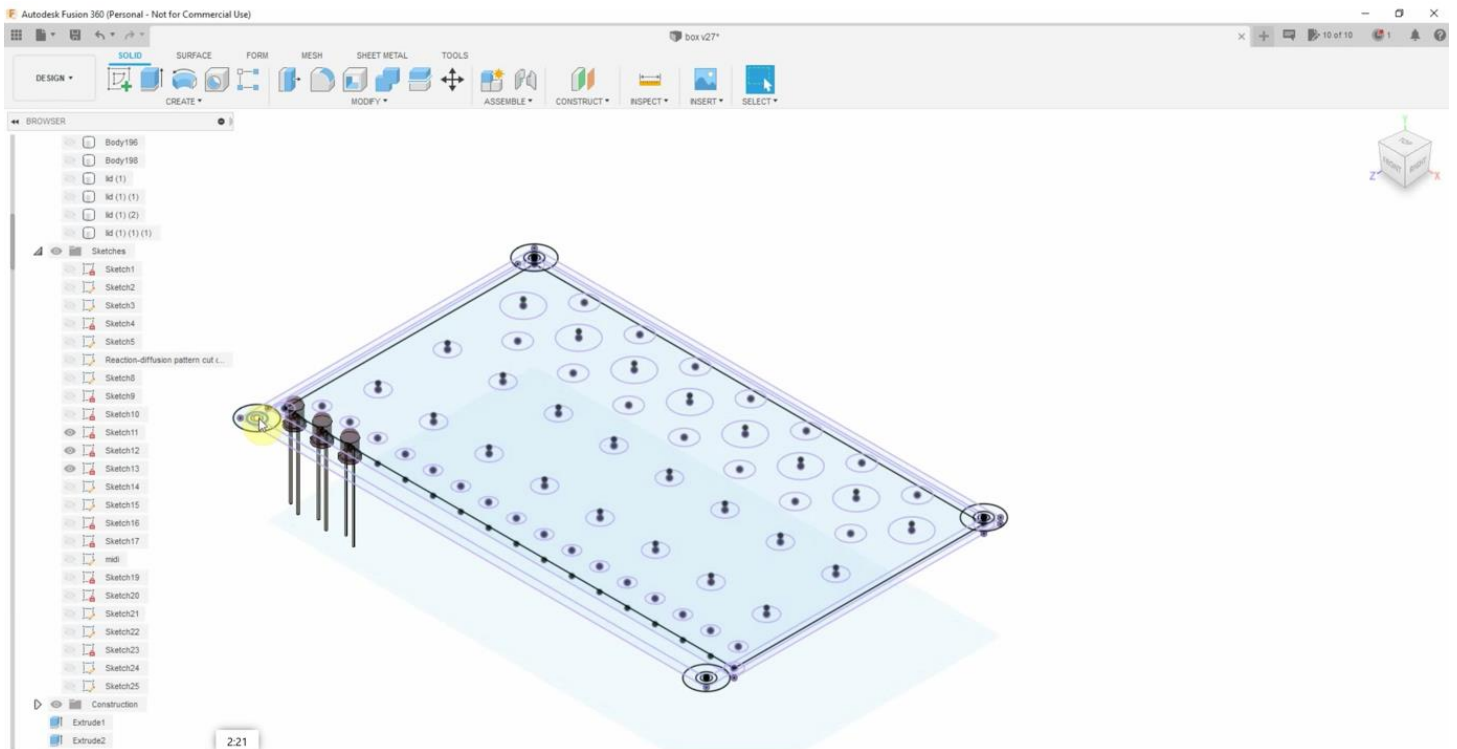
- Glue gun
- Soldering iron
- Digital multi-meter
- Wire pliers
- Wire stripper
- Screwdriver
- Wire
- Solder
- tweezers

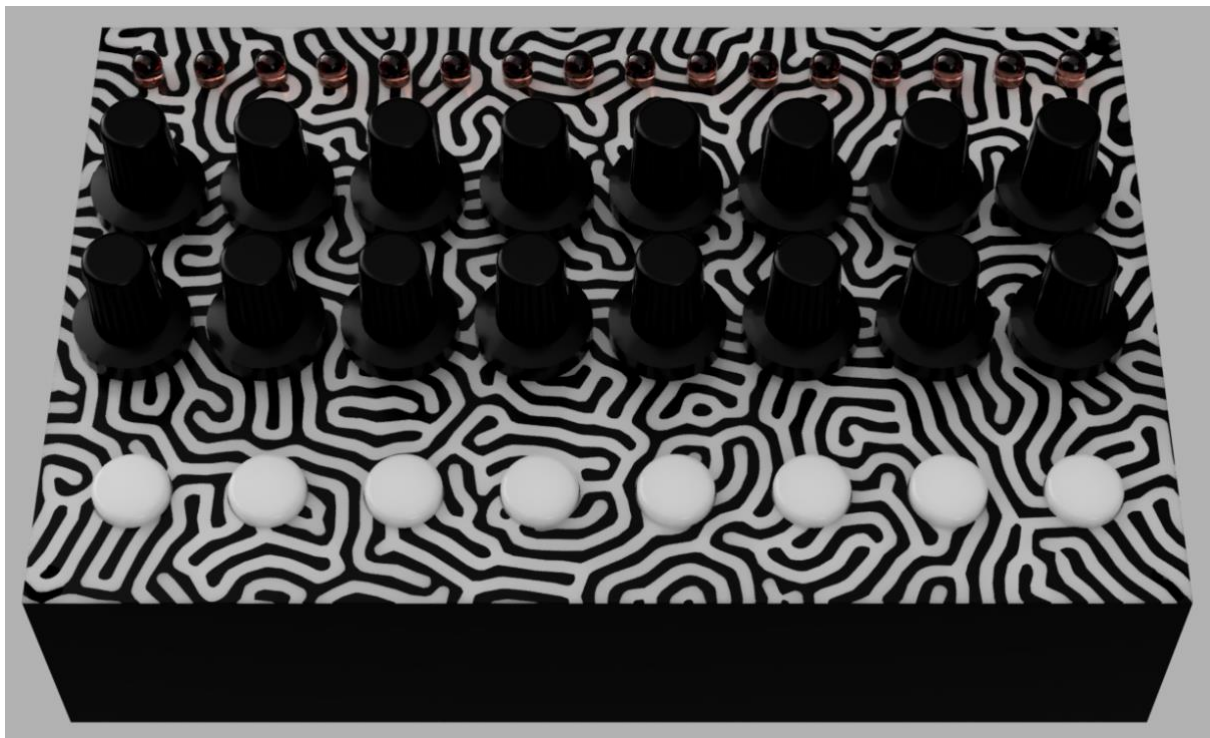
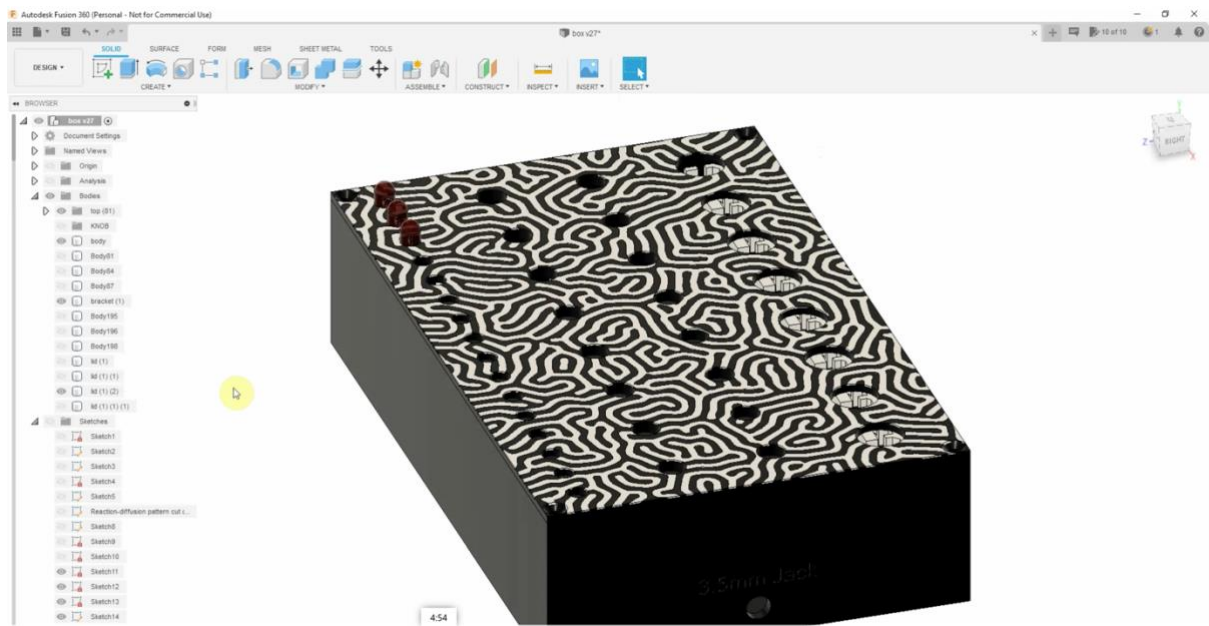


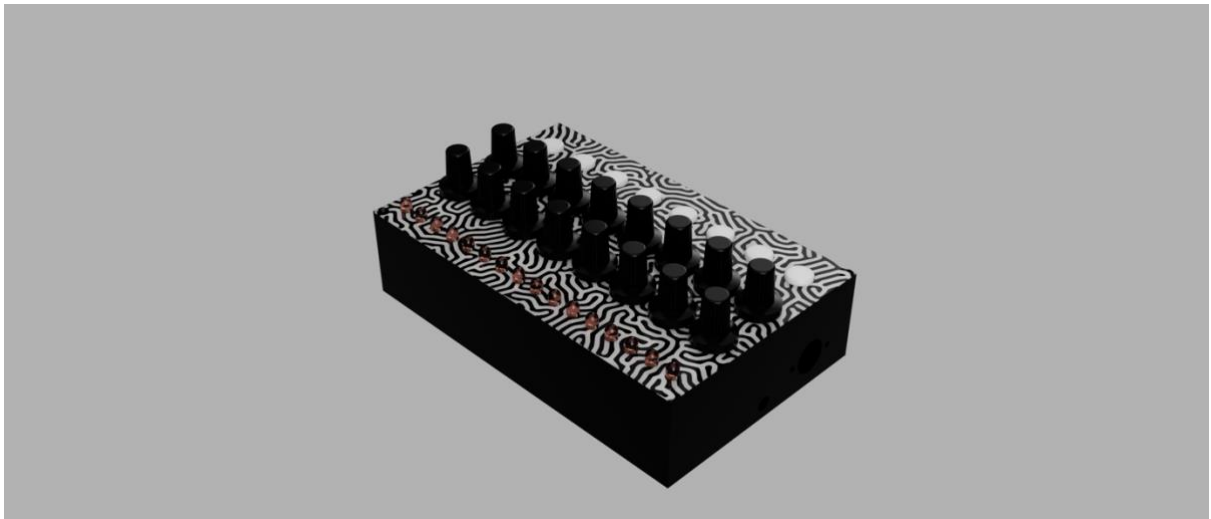
Appendix VII: Please see attached Video & Picture below

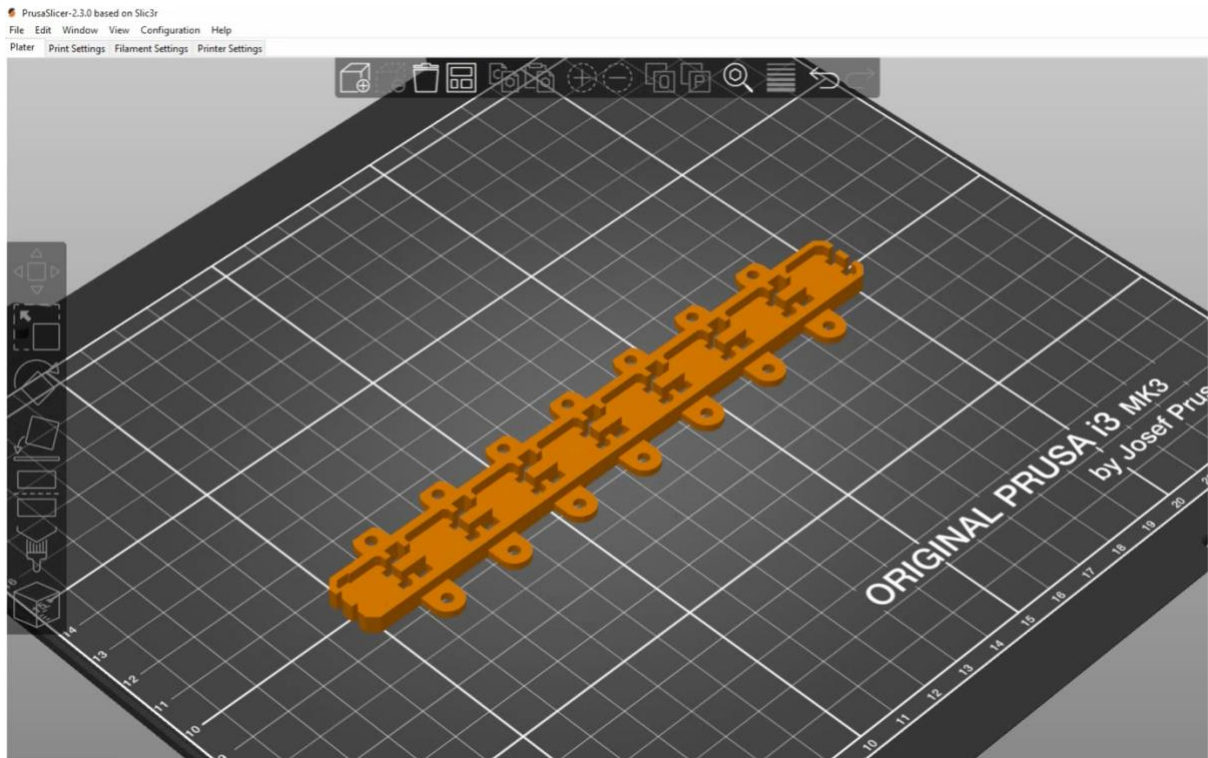


Appendix VIII: Sketches and 3D models showing synth box design



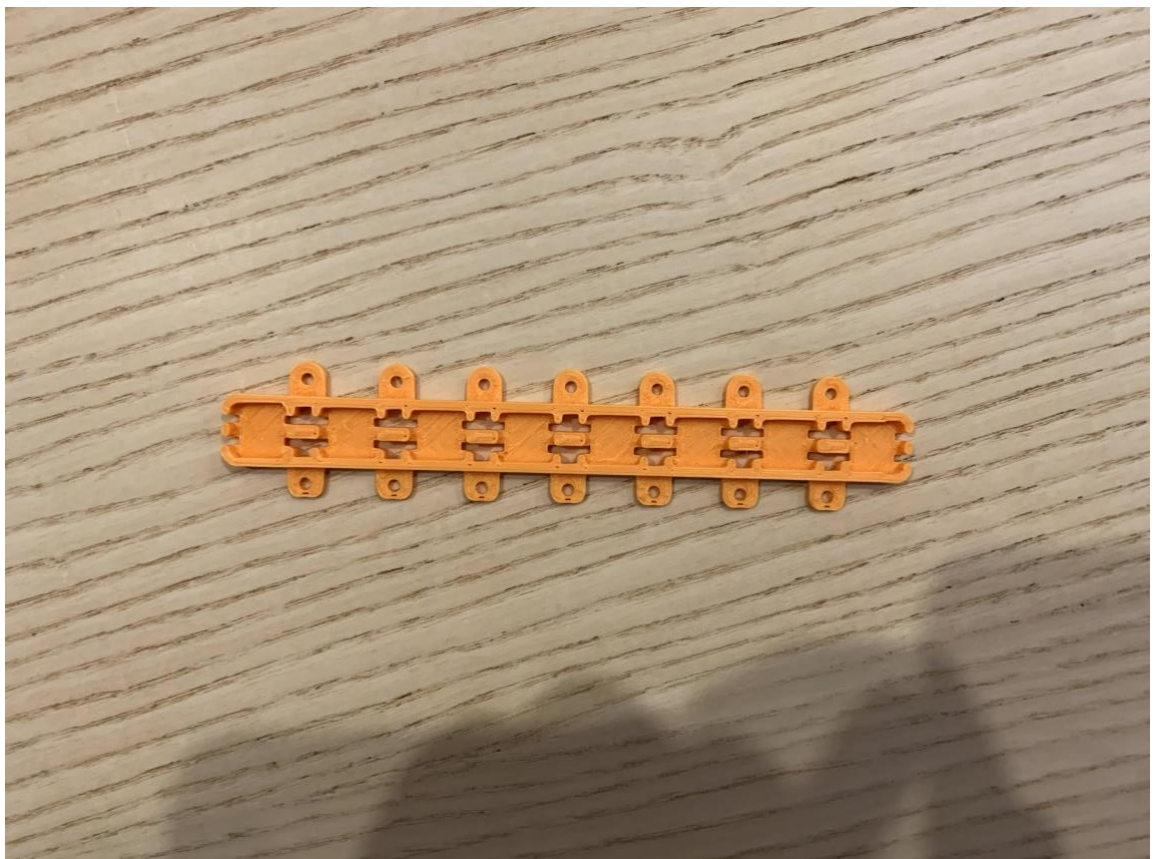
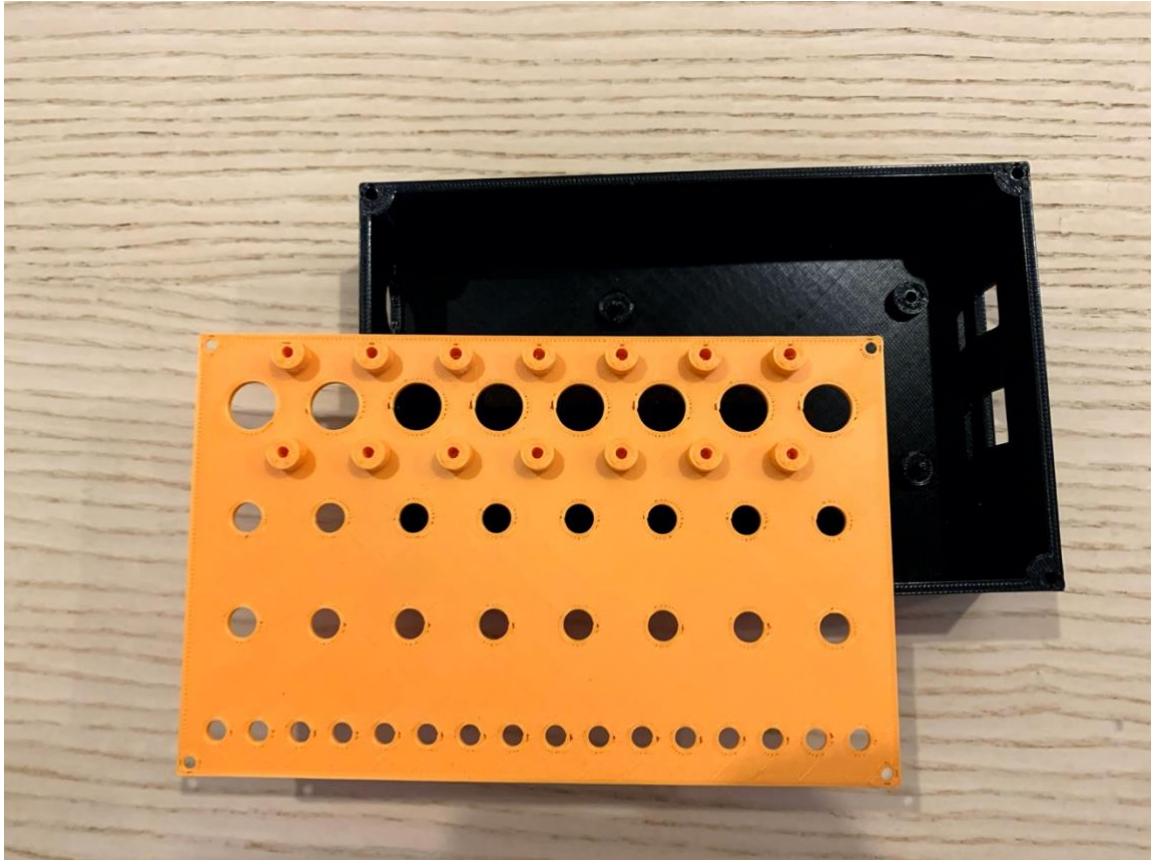




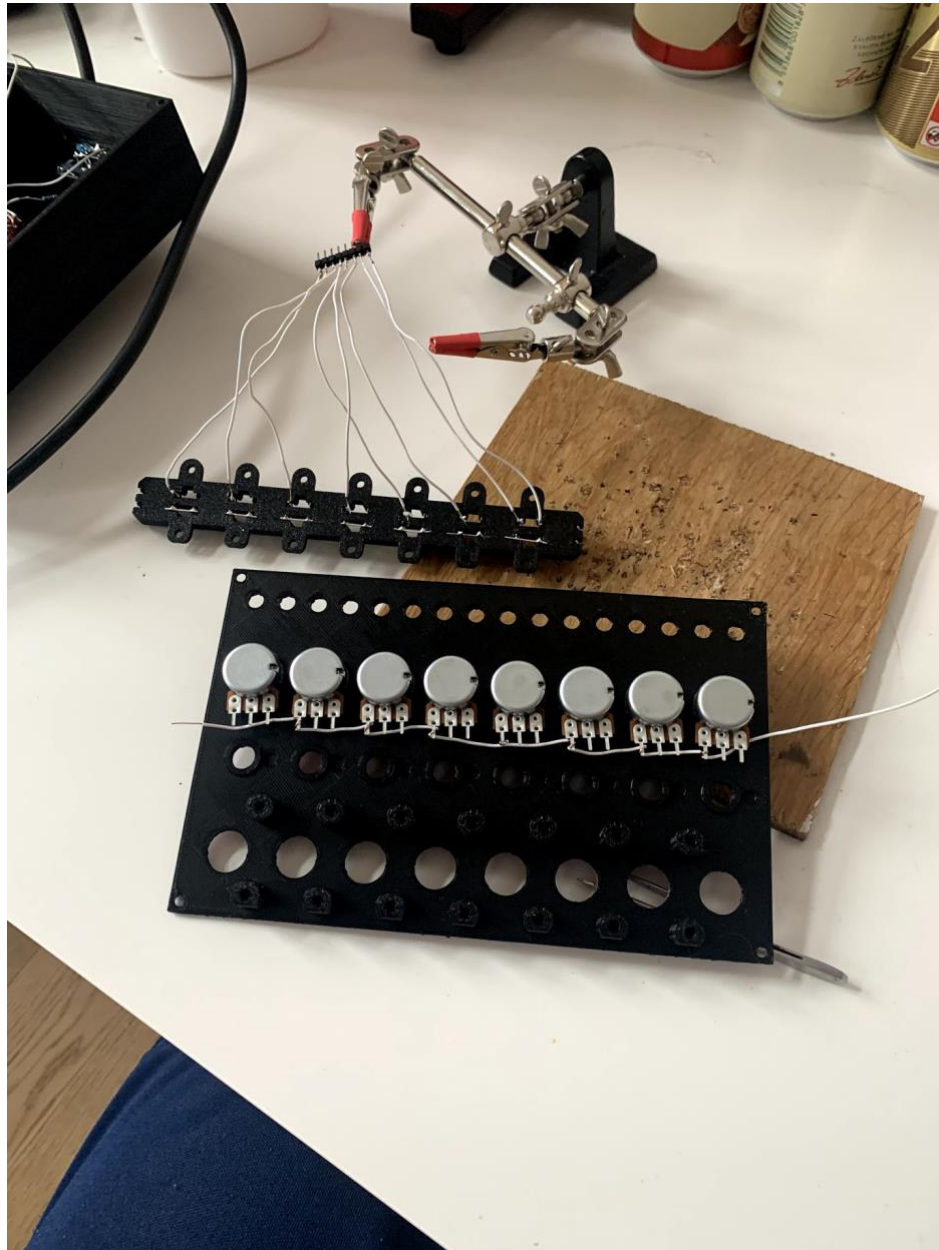


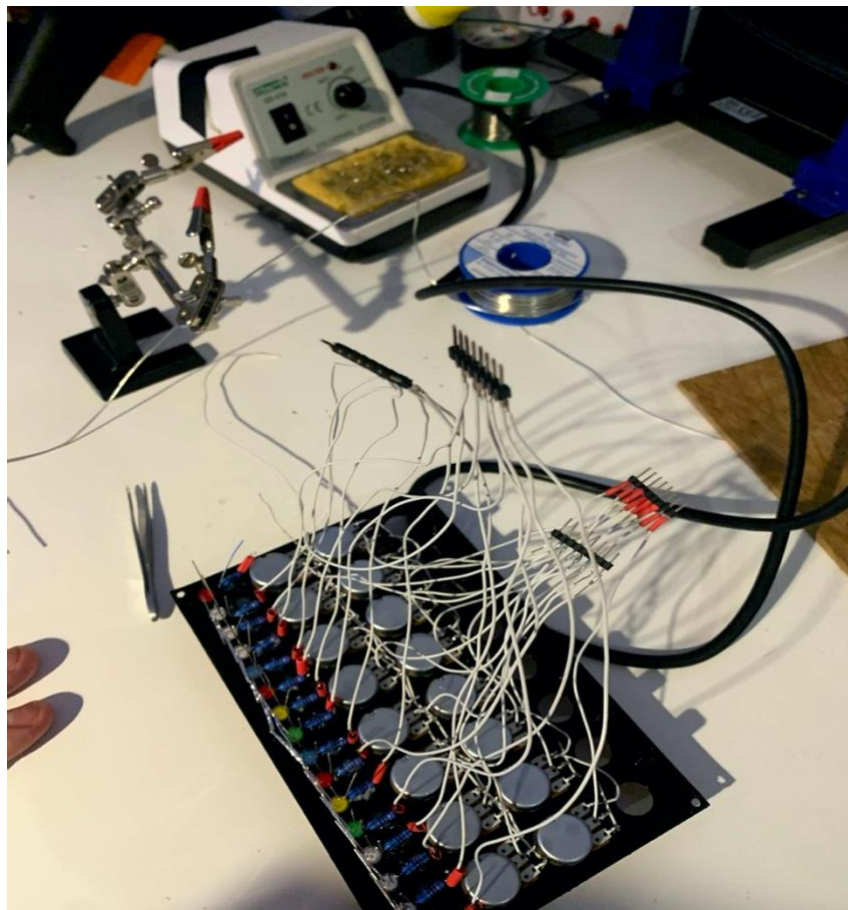
Appendix X: Pictures of test and scrap parts

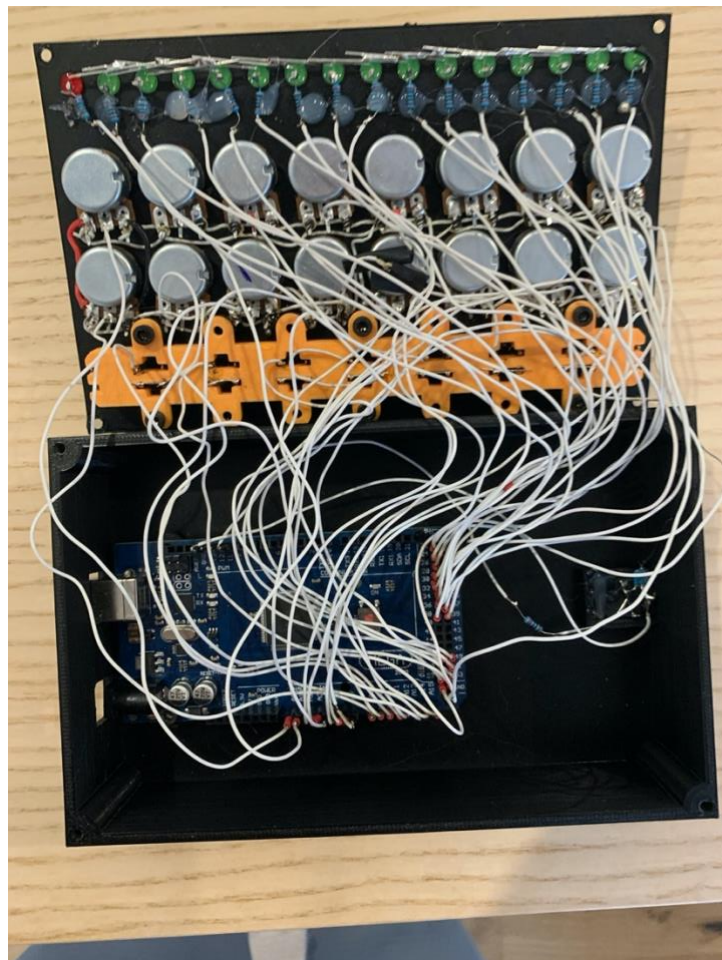
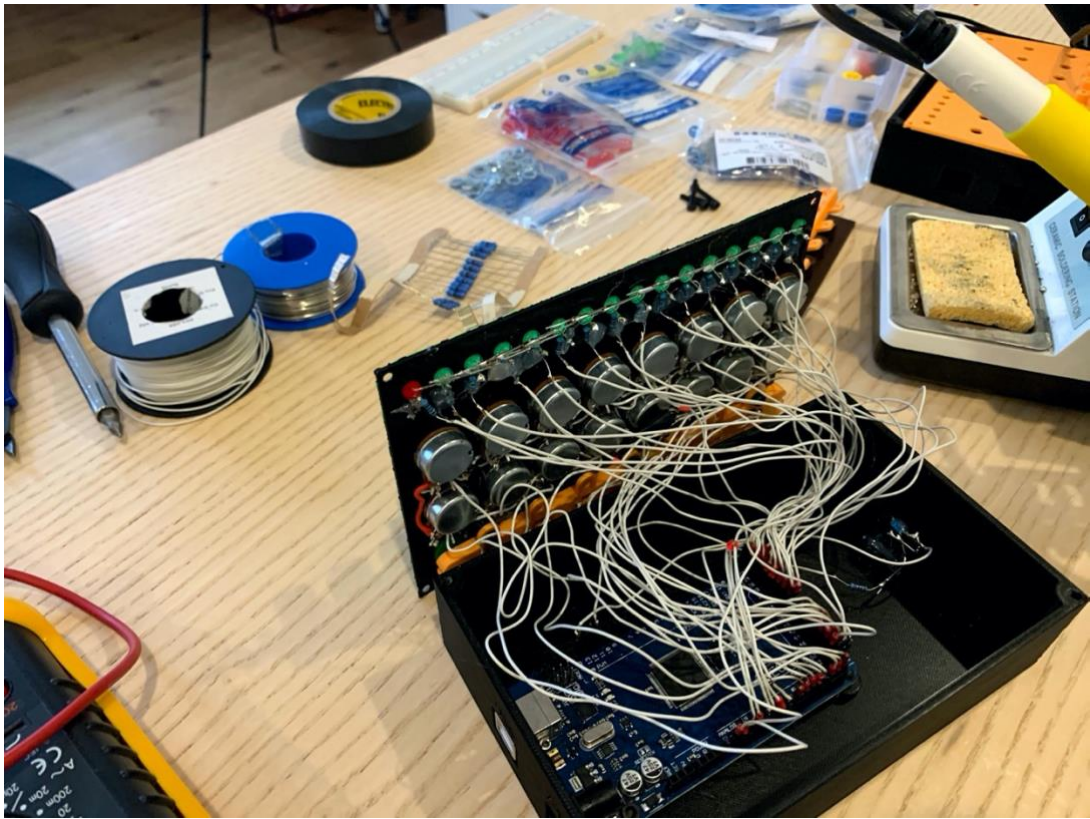




Appendix XI: Please see attached Video: Below are pictures of the wiring process





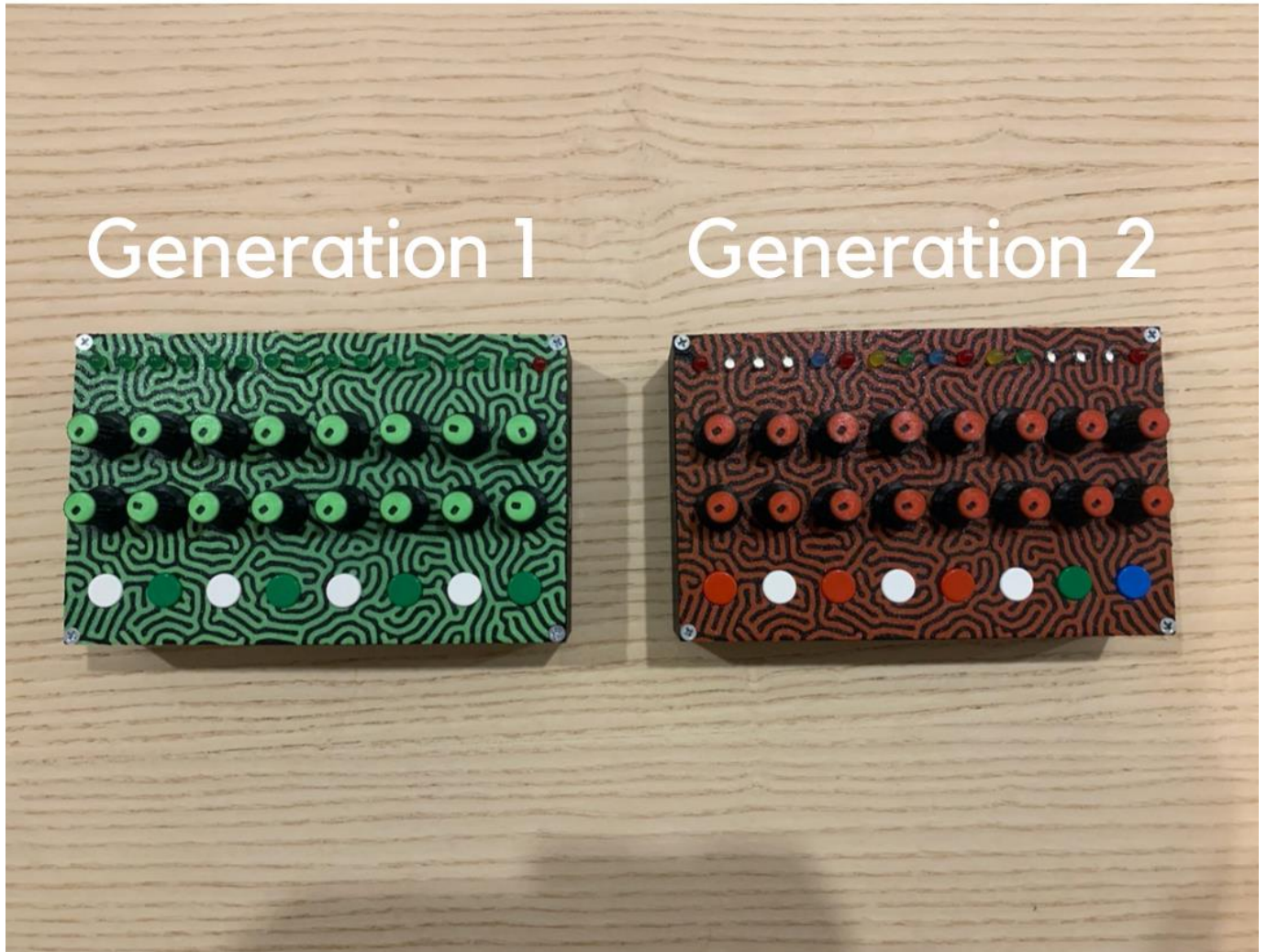


Appendix XII: Please see attached PDF: Code

Appendix XIII: Please see attached video: Walkthrough of the code

Appendix XIX: Please see attached PDF: Mapped out midi in Hz

Appendix XX: Picture of first Generation (Green) & Second
Generation (Red) synth



Appendix XXI: Please see attached video: Octave selection

Appendix XXII: Please see attached video: Key selection

Appendix XXIII: Please see attached video: Reverse sequence & pulsing purple lights

Appendix XXIV: Please see attached video: Example of the pentatonic scale being played

Appendix XXV: Please see attached video: Buttons used in playback mode