Berkleee College of Music

The M.A.F. Machine: A Solution for Live Performance in Changing Systems of Equal Division of Octave

Submitted in Partial Fulfillment of the Degree of
Master of Music in Music Technology, Production, and Innovation

Supervisor: Marta Verde

by Samuel Vincent

Valencia Campus, Spain

June 2020

Table of Contents

Acknowledgements

Thanks to everyone inside and outside the Berklee community who contributed ideas and feedback to the project, including the panelists at the AES Saul Walker Design competition, who awarded the M.A.F. Machine with a second place prize.

Thanks to my advisor, Marta Verde, whose guidance made the whole project possible. As someone who had never touched hardware development before, Marta opened up a world of possibilities for me, and I would have been totally lost if not for her instruction.

A big (perhaps even MACROTONAL) thanks to Tetyana Haraschuk and Michael Lucarelli, who contributed to the M.A.F. vision by lending their considerable instrumental talents. A big thanks to Ricky Young and Nicholas Phang for their engineering work, as well as to Ryan Tram and Kanishk Seth for their videography. Also thanks to Angel Rose, who in addition to helping me make my weird design look good was extremely supportive of the M.A.F. vision.

Abstract

Though many applications for microtonality exist, few allow for the flexible changing of systems of microtonality during live performance. This project implements the M.A.F. Machine, a hardware controller implementation designed to facilitate live performance in arbitrary, and changing, systems of equal temperament. The hardware is a 3D-printed casing containing circuitry programmed to detect how many plastic blocks, or “keys,” have been placed in a single octave of the unit. This data is then sent to the software application using MIDI retuning algorithms, and the software application uses the information about which “keys” are in which slots to determine the system of equal temperament. Finally, the project explores compositional techniques and systems of notation for microtonality, culminating in the composition and performance of original music using this instrument along with percussion, fretless instruments, and other instruments that are compatible with microtonality.

Keywords: microtonality, equal division of octave, interface design, hardware development

1. Introduction

Exploration of microtonal systems in western music has become rapidly more publicised in recent years, due in part to experiments from artists such as Adam Neely, Jacob Collier, and King Gizzard and the Lizard Wizard. As demand for playing in systems of microtonality grows, many artists find that existing instruments for live performance in systems of microtonality are clunky, limiting, and difficult to use. The project attempts to address this problem through the creation of a simple, easy-to-understand instrument for live performance in systems of microtonality.

Systems of microtonality are, by definition, infinitely continuous. It is therefore necessary to limit the scope of any project involving microtonality to a predefined metric by which to generate microtonal scales. The project features an implementation of a system that is limited to changes based on Equal Division of Octave (henceforth referred to as EDO). EDO, also known as Equal Temperament, is a system by which an octave is broken up into frequencies with equal distances between each other on a logarithmic scale, such that each note in the scale sounds equally spaced from the previous note. Western music is generally written in 12-tone EDO, the system that builds the 12 tone chromatic scale. The construction of this scale is not entirely arbitrary -- it contains rough, practical approximations of many intervals that sound “pleasing” to the human ear due to their mathematical relationship with the fundamental pitch of the octave[1]. However, constructing the scale in this way leaves out many possibilities, and

---

[1] “Why 12 Notes to the Octave?” by Michael Reubenstein, accessed December 2020. https://www.math.uwaterloo.ca/~mrubinst/tuning/12.html

other scales, such as 19-tone, 22-tone, or 60-tone EDO contain pleasing intervallic relationships that are left out by the 12-tone chromatic scale.

The solution developed for the purposes of the project is called the M.A.F. Machine, where M.A.F. stands for “Microtonal Acid Funk.” The M.A.F. Machine allows users to achieve up to 25 EDO by placing hexagonal keys into slots on the device. The device detects how many keys have been placed inside it and changes the system of EDO automatically, without the need to change values manually on a software or hardware interface. The device then sends information about key presses to a MIDI converter in Max/MSP, which acts as a MIDI device and can control any software or hardware synthesizer that has MIDI and pitch bend capability.

Finally, this project culminates in the composition and performance of musical pieces in changing systems of EDO using this instrument and other instruments that are compatible with microtonality. The music recorded is meant to be performed in a live scenario, and it features a quartet of drums, viola, lead/bass synthesizer, and synthesizer controlled by the M.A.F. Machine, for a set that is 11 minutes long in total and changes systems of EDO 4 times.

2. The State of the Art: Microtonal Literature Review

Much research and development has been done in the realm of microtonality, and numerous hardware and software applications have attempted to create alternate systems of tuning[2]. In the article "Microtonal piano roll for Ableton,"[3] musical artist Sevish writes about creating an interface using the fold function on Ableton's midi keyboard roll to simulate a 22-EDO[4] keyboard. Sevish's concept is similar to the goal of the M.A.F. Machine in that it creates an interface between hardware and software for performance in an alternate system of tuning; however, the M.A.F. Machine aims to abstract the equal division of octaves such that one can change them easily during a performance, rather than implement a solution for only one system of EDO. To date, software systems have created more or less seamless ways to accomplish this abstraction using regular, 12-EDO MIDI input (see Table 1), but hardware applications remain an emerging field. The most common hardware devices for the achievement of microtonality involve either 1) allowing the user to bend between notes using a plastic or rubber strip[5] or 2) using a boardlike instrument with a pre-programmed set of (generally hexagonal) polygons. Hexagonal polygons are usually chosen for this task because of their unique spatial properties -- each hexagon can be placed on the side of other hexagons in a self-replicating format, and their shape lends them playable by human fingers. Several varieties of such instruments exist (see Table 2), yet they are generally difficult to manage accurately in a live situation.

---

[2] See Table 1 for examples of software and Table 2 for examples of hardware.

[3] Sevish, "Microtonal piano roll for Ableton," by Sean Archibald, accessed November 2020, https://sevish.com/2016/microtonal-piano-roll-for-ableton-live/.

[4] Equal Division of Octave, also known as Equal Temperament

[5] Such as the Linnstrument or Roli Seaboard (see Table 2)

The M.A.F. Machine also involves a reinterpretation of MIDI protocol in order to interface the hardware controller with the software synthesizer. The General MIDI Specification, a standardization of MIDI events, does not currently support microtonal music without complex pitch bend workarounds[6], and it is generally impossible to change MIDI information sent from one key in a hardware MIDI controller. It is, however, possible to change the software's response to the data sent by the hardware by altering the MIDI note-to-frequency function to account for different systems of Equal Division of Octave:

$$2^{(n-h)/e} * s$$

Where n = MIDI Note Number

e = System of EDO (12 is standard)

s = starting pitch (in Hz, 440 is standard)

h = starting key on a MIDI-compatible keyboard (key number 69, or A4, is standard)

Figure 2.1: the MIDI note to Frequency Function[7]

The reinterpretation of this function allows a user to utilize a MIDI keyboard in standard tuning to send MIDI data to a software interface that can then convert the pitch to microtonal frequencies. Scala, a program designed to generate microtonal scales, utilizes MIDI mapping in a similar fashion, and its robust documentation and support has

---

[6] LilyPond Notation Reference, "Supported Notation for MIDI", accessed November 2020, http://lilypond.org/doc/v2.21/Documentation/notation/supported-notation-for-midi

[7] Joe Wolfe, "Note Names, MIDI Numbers, and Frequencies", University of New South Wales School of Physics. Accessed November 2020, http://newt.phys.unsw.edu.au/jw/notes.html

generated a community of microtonal plugin developers[8] (see “Microtonal Poly Worms” in Table 1).

Therefore, an analysis of the state of microtonal art demonstrates the steps the music industry has taken to address the musical frontier of microtonal performance. It also demonstrates that these steps can be taken further: as yet, a solution does not exist that is tailored for easily changing systems of EDO in a live scenario. This project attempts to implement such a solution by combining the software development methodologies presented above with elements of the design techniques used by current microtonal instrument designers.

---

8. Manuel Op de Coul, “Scala Features”, Scala. Accessed November 2020, http://www.huygens-fokker.org/scala/

Table 1: List of Existing Microtonal Software Synths / Plugins

Tool	Creator	Link	Comments
MTS-ESP	ODDSound / Aphex Twin	https://pitchfork.com/news/aphex-twin-develops-new-microtuning-synth-plugin-with-uk-startup/	A well-publicised, very recent (released in March 2021), microtonal tuning plugin with applications for hardware or software synthesizers.
MicroTuner	Yordi de Graaf	https://www.native-instruments.com/en/reaktor-community/reaktor-user-library/entry/show/11297/	Plugin with an extremely basic, easy-to-understand interface that outputs a frequency to an arbitrary EDO, similar to this project's idea.
Microposer	D5x610	https://gumroad.com/l/LwsghG	Basic plugin that "squeezes" midi frequencies across a regular keyboard to the user's specification.
Kontakt	Kontakt	https://soundbytesmag.net/technique-microtuning-in-kontakt-5-and-6/	A popular sample-driven synthesizer with support for scripting that allows achievement of microtonality.
Microtonal Poly Worms	BipTunia	https://biptunia.com/?p=3045	A robust microtonal software synth built using Scala, a programming environment that allows for development in systems of alternate tuning.
EP-MK1	mianmogra	https://patchstorage.com/ep-mk1/	A sample-based electronic piano plugin that allows for changes in temperament.
OffTonic Synth	OffTonic	https://offtonic.com/synth/	A browser-based microtonal synth.

Table 2: List of Existing Microtonal-Compatible Controllers / Hardware:

Tool	Creator	Link	Comments
Lumatone Isomorphic Keyboard	Lumatone	https://www.lumatone.io/	280-key “isomorphic” MIDI keyboard, currently in production as of June 2021.
AXiS 49 Hexagonal MIDI controller	C-Thru Music	https://reverb.com/au/item/150238-c-thru-music-axis-49-hexagonal-midi-controller	Mass-produced hexagonal controller used by microtonal electronic artist Sevish.
OPAL Chameleon	OPAL	http://www.shapeofmusic.com/	Hexagonal midi instrument with many features.
KORG Polyphonic Ensemble	KORG	http://www.vintagesynth.com/korg/pe1000.php	A vintage synth (built 1976) that lets users change the tuning of each key individually.
Tonal Plexus	h-pi	https://hpi.zentral.zone/tonalplexus	A very robust, very complicated piece of microtonal hardware. A 5-octave model that has 1300 keys is in development as of June 2021.
Linnstrument	Roger Linn	https://www.rogerlinndesign.com/linnstrument	A “strip” with a grid of squares that allow smooth transitions between corresponding squares, possible to program to play microtonality.
Seaboard	Roli	https://roli.com/products/seaboard	A “strip” with indentations that follow 12-tone equal temperament but allow the user to access tones between the notes, generally for a pitch bend effect, though microtonality is possible to achieve.

3. Description

3a. Hardware Development

The hardware prototype measures 375mm long, 127mm wide, and 87.5mm high. It contains 25 slots for keys, with 2 buttons to change the octave up or down and LED lights above each key to provide visual feedback. When a key is inserted, the LED corresponding to the slot in which the key has been placed will light up, and it will blink on and off rapidly when the key is pressed.

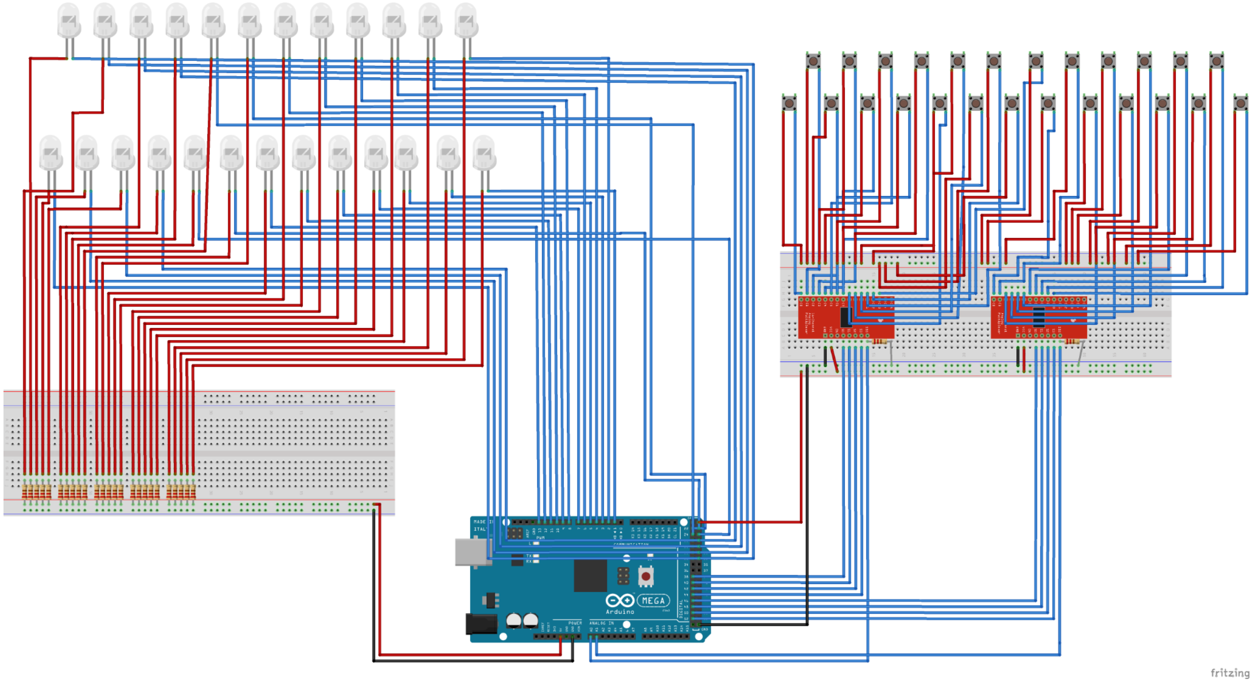


Figure 3.1. Schematic diagram of the M.A.F. Machine.

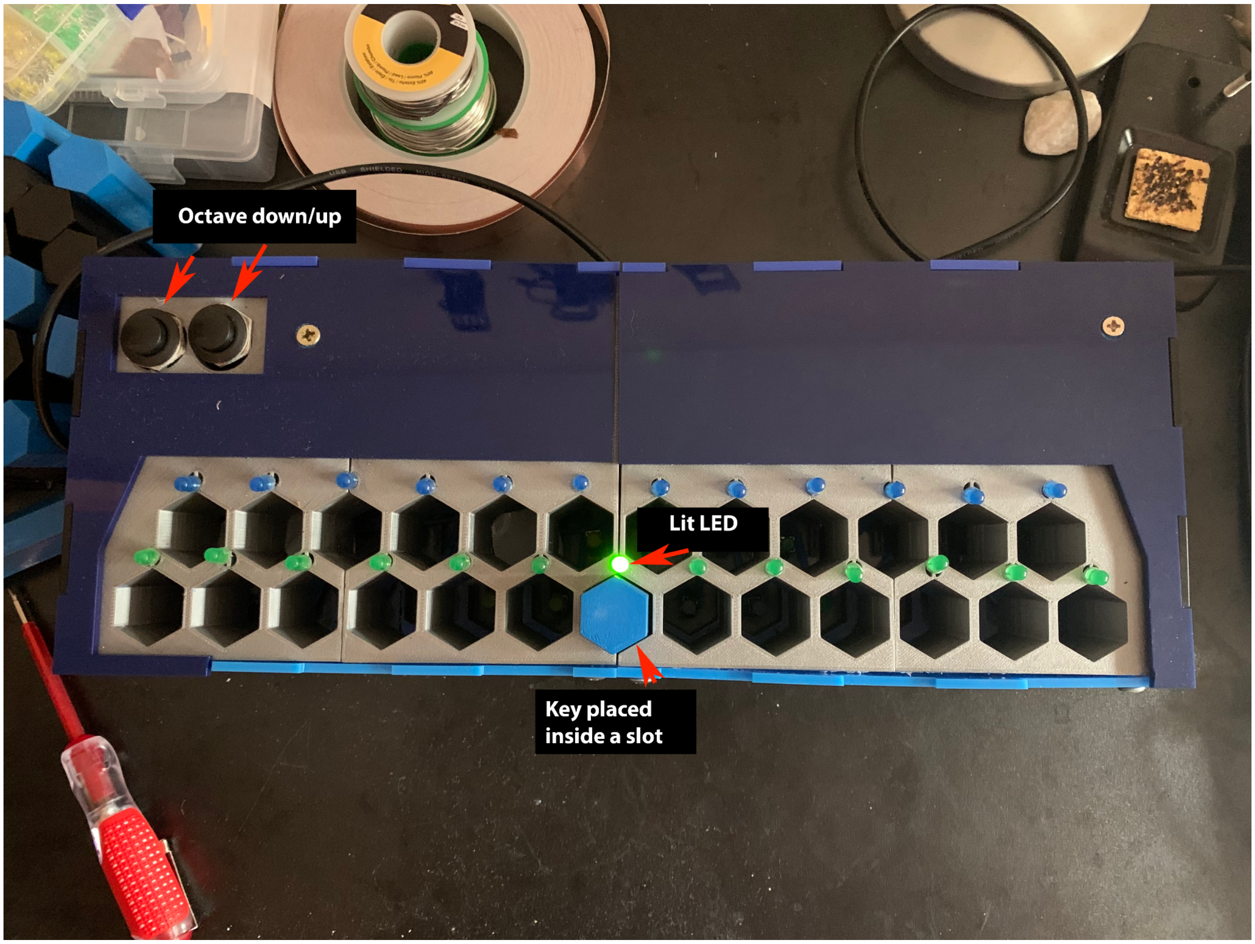


Figure 3.2. Diagram of the M.A.F. Machine with one key inserted.

Note the hexagonal shape of the keys, which is designed to be similar to other solutions for microtonal controllers, such as the Lumatone Isomorphic Keyboard (as demoed by Synthtopia)[9].

The casing for the controller consists of 8 3D printed sections fastened together using screws and metal fasteners, along with 10 panels for the front, back, and sides of the controller. The side panels are made of laser-cut acrylic and are designed to provide access to the inside of

---

[9] Synthtopia, "Lumatone Intros New Isomorphic Keyboard for Microtonal Music & More." 26 January, 2020. Accessed 23 May 2021. https://www.youtube.com/watch?v=LfNIxTiApig

the instrument as well as protection from outside elements that could damage the wiring inside the controller. Additionally, 25 hexagonal keys were printed to fit inside the slots.

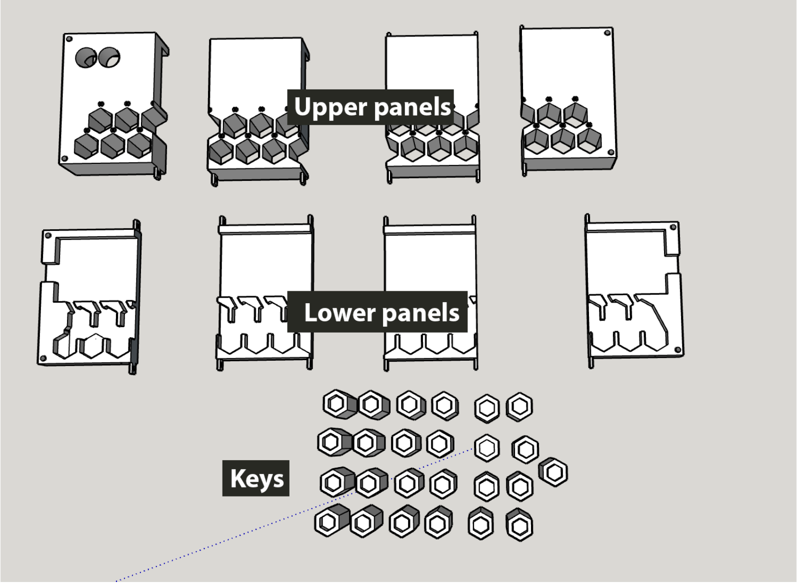


Figure 3.3. Diagram of the casing in 3D modeling software.

The casing was designed with a large tray running through the upper part of the instrument to handle routing of the wires, placement of the microcontroller and circuitry, and any other accessory that would need to be added to the controller as the prototype progressed. Upon implementation, the height of the controller needed to be raised in order to fit the amount of wiring and circuitry that had accumulated inside the controller; this was accomplished using the laser-cut acrylic panels previously discussed.

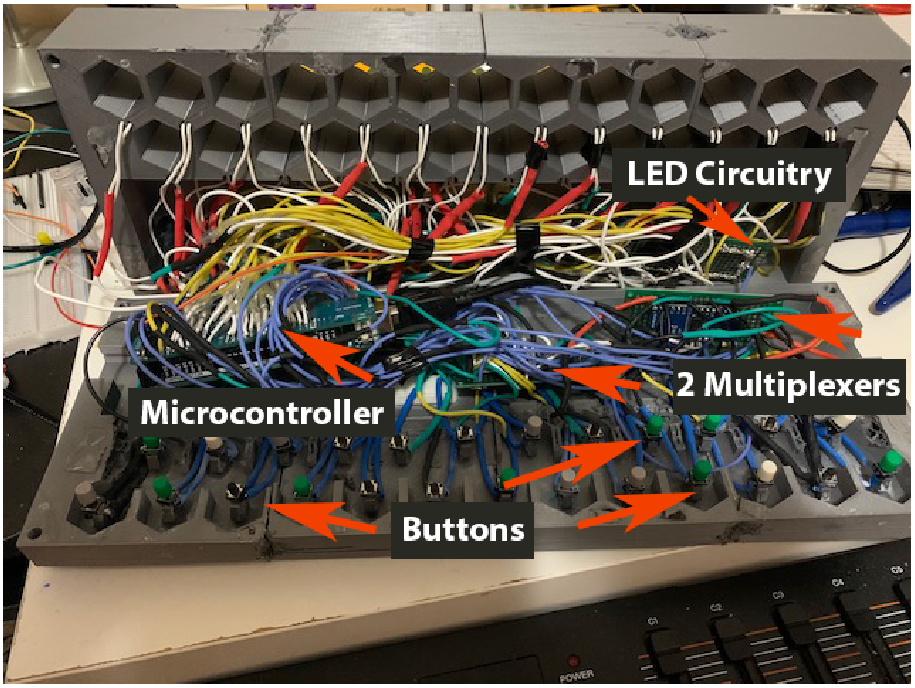


Figure 3.4. Circuitry inside the prototype.

Inside, the controller contains one large microcontroller (generically referred to as a "Mega") and 2 16-channel multiplexers which are sent to 2 digital pin inputs on the microcontroller. The 25 buttons at the front of the device are connected to the argument pins on the two multiplexers, and in each frame they are iterated over to catch key press events. Every button and LED light is soldered to a prototype board to receive 5V power, ground, and/or argument input/output. Each connection is also wired to a prototype shield specifically designed to interface with the Mega microcontroller for additional stability and security.

3b. Software Architecture

To keep latency low, the software for the microcontroller is mainly designed to catch key press events and to package and send their data elsewhere, without taxing the processing power of the microcontroller greatly. The first time a button is pressed, it will update the system of EDO on the microcontroller, which will in turn update the MIDI conversion software on the computer. Every subsequent time the same button is pressed, it will create a note-on event and send a midi note corresponding to that note's location in the sequence of keys that have been placed on the controller. This MIDI note to be sent is calculated using the following formula:

$$n = 48 + k + (12 * o)$$

(Function 3.1)

Where n is the note number to be sent to the software, k is the index of the key that is being pressed in relation to the keys to the left of it on the controller, and o is the number of octaves the note should be offset. Note that this implementation is built from MIDI note 48 (the C below middle C), as it is a useful starting point for building systems of EDO, especially when working with bassline synthesizers.

To take a note away from the controller, a user presses both octave buttons down and then presses the note they want to take away.

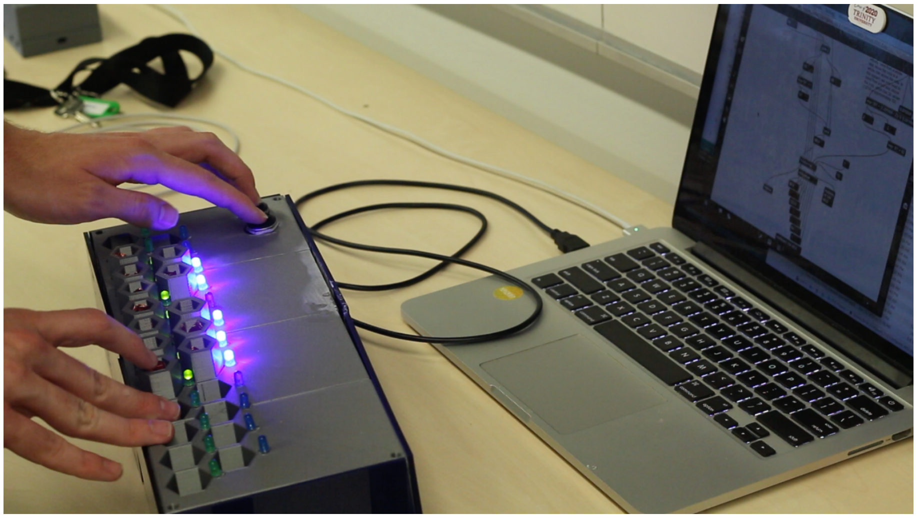
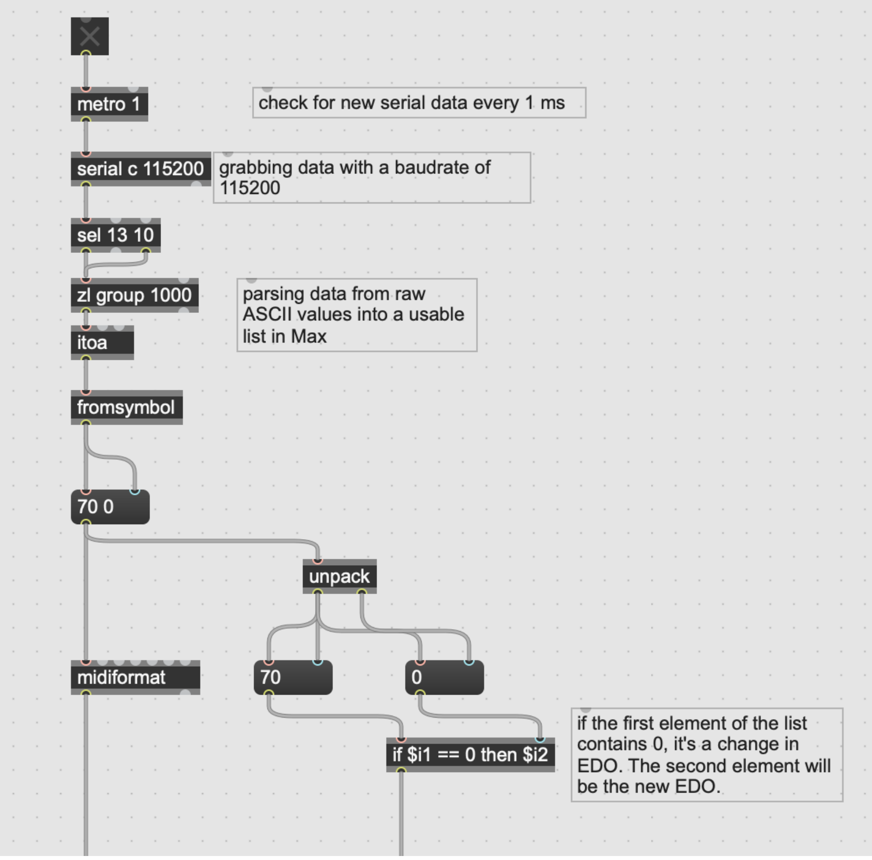


Figure 3.5. Removing a key from the controller.

When this occurs, the LED above the note will turn off and the note will be taken away from the array of available notes. The key can then be pulled out of its slot. The microcontroller also handles the LED feedback system by keeping information about which slots currently have keys and which are currently being pressed and updating the lighting accordingly.

While the microcontroller handles all aspects of the hardware input and output, the bulk of the computational processing occurs in Max/MSP, which reads changes in EDO and then converts incoming MIDI to outgoing MIDI with pitch bend that corresponds to the chosen scale. First, the patch must read and parse the data coming from the microcontroller in the serial protocol.



check for new serial data every 1 ms

grabbing data with a baudrate of 115200

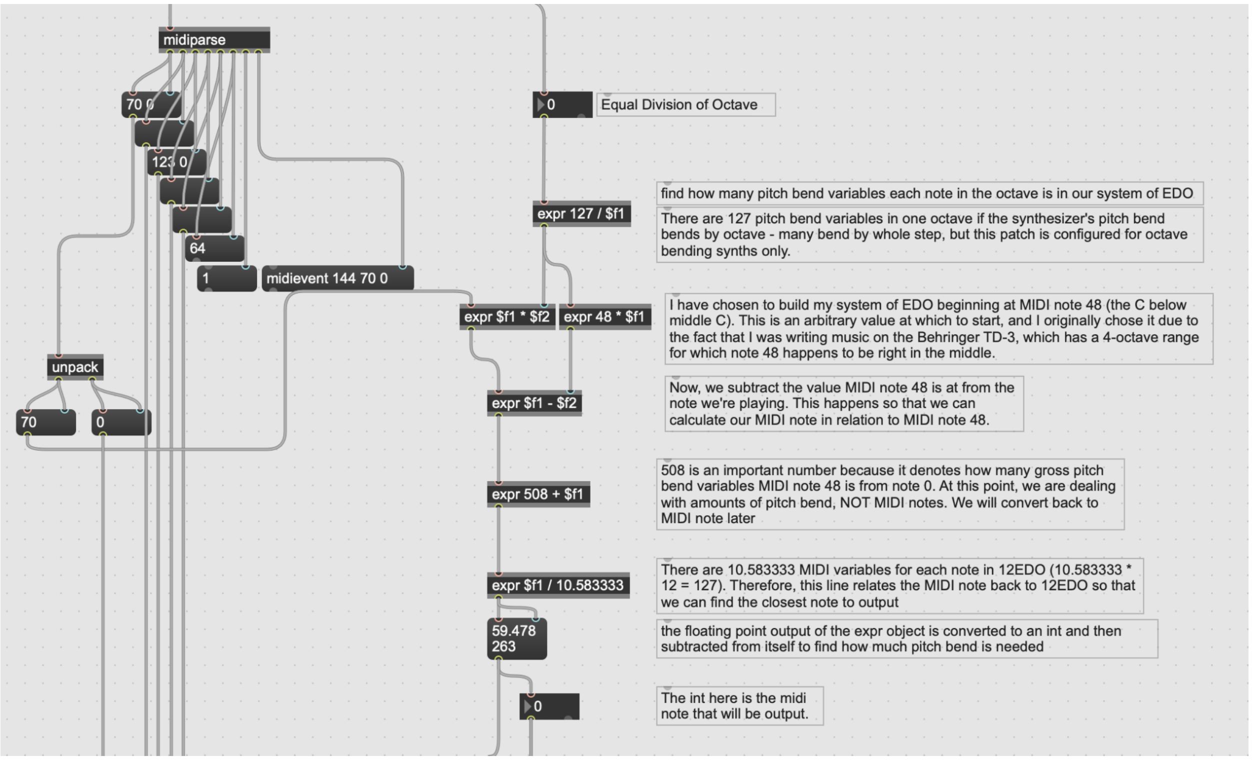
parsing data from raw ASCII values into a usable list in Max

if the first element of the list contains 0, it's a change in EDO. The second element will be the new EDO.

Figure 3.6. Serial input and parsing.

As shown in figure 3.1, the input from the microcontroller is parsed by grabbing data using the Serial protocol at a baudrate of 115200 and then determining whether the message contains regular MIDI note values or a special character that invokes a change in EDO. In this case, note 0 has been arbitrarily chosen to change the system of EDO; it is very unlikely that MIDI note 0 will be played in a musical context, as the frequency associated with it is below the range of human hearing.

After input is parsed, it must be converted from a MIDI note in 12 EDO to a MIDI note in the correct system of EDO.



find how many pitch bend variables each note in the octave is in our system of EDO

There are 127 pitch bend variables in one octave if the synthesizer's pitch bend bends by octave - many bend by whole step, but this patch is configured for octave bending synths only.

I have chosen to build my system of EDO beginning at MIDI note 48 (the C below middle C). This is an arbitrary value at which to start, and I originally chose it due to the fact that I was writing music on the Behringer TD-3, which has a 4-octave range for which note 48 happens to be right in the middle.

Now, we subtract the value MIDI note 48 is at from the note we're playing. This happens so that we can calculate our MIDI note in relation to MIDI note 48.

508 is an important number because it denotes how many gross pitch bend variables MIDI note 48 is from note 0. At this point, we are dealing with amounts of pitch bend, NOT MIDI notes. We will convert back to MIDI note later

There are 10.583333 MIDI variables for each note in 12EDO (10.583333 * 12 = 127). Therefore, this line relates the MIDI note back to 12EDO so that we can find the closest note to output

the floating point output of the expr object is converted to an int and then subtracted from itself to find how much pitch bend is needed

The int here is the midi note that will be output.

Figure 3.7. MIDI Conversion.

To achieve this conversion, an algorithm was developed that splits an octave into the amount of pitch bend variables in the octave divided by the system of EDO the user is in.

$$p = 127/E$$

(Function 3.2)

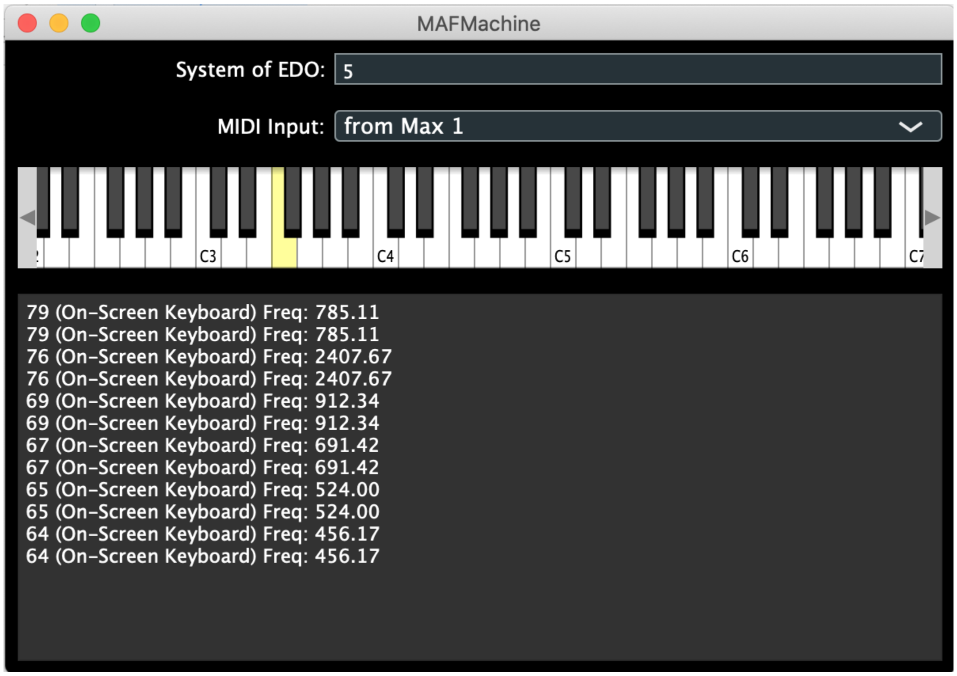
Where p is the raw amount of pitch bend variables that the note is in relation to the octave and E is the selected EDO. Then, p is used to convert the note into MIDI plus pitch bend by taking the difference of the note and MIDI note 48 (which, as previously mentioned, was chosen due to its placement on the keyboard), and calculate where the note is in relation to 12 EDO.

$$(508 + ((n * p) - (n * 48)))/10.583333$$

(Function 3.3)

Where n is the MIDI note, 508 is how many raw pitch bend variables MIDI note 48 is from 0, and 10.583333 is how many raw pitch bend variables there are for one note in 12 EDO (10.583333 * 12 = 127). This function will return a decimal; on the left of the decimal is the MIDI note to be output, and on the right is how much pitch bend is required as a percentage of the pitch bend variables allotted per note. From there, merely convert the right side of the decimal to pitch bend by mapping its scale to the amount of pitch bend variables in one note, and the value will be output as a MIDI object in Max/MSP to whatever MIDI-compatible hardware or software instrument that is desired.

While the Max/MSP implementation works well for sending monophonic MIDI to hardware or software synthesizers, it is not useful as a method for the achievement of polyphony. This is because it relies on pitch bend, which is a MIDI control channel attribute, rather than a note attribute. Therefore, sending pitch bend information to a chord will change a full MIDI instrument's pitch bend rather than the pitch bend of a singular note, and the chord will sound like it normally would in 12 EDO, displaced by the last pitch bend value received. To overcome this problem, a polyphonic software synthesizer was built using the C++ framework JUCE.



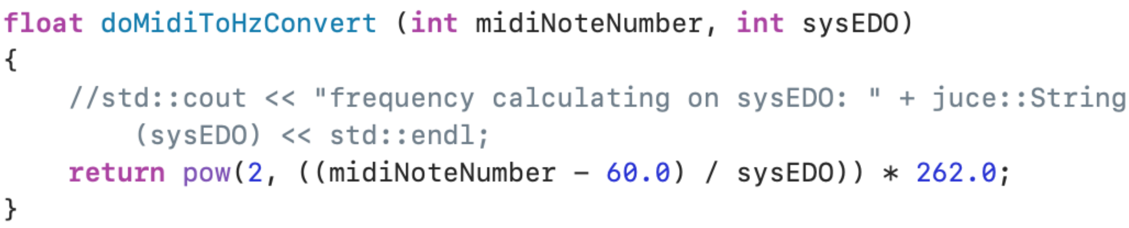
System of EDO: 5

MIDI Input: from Max 1

```
79 (On-Screen Keyboard) Freq: 785.11
79 (On-Screen Keyboard) Freq: 785.11
76 (On-Screen Keyboard) Freq: 2407.67
76 (On-Screen Keyboard) Freq: 2407.67
69 (On-Screen Keyboard) Freq: 912.34
69 (On-Screen Keyboard) Freq: 912.34
67 (On-Screen Keyboard) Freq: 691.42
67 (On-Screen Keyboard) Freq: 691.42
65 (On-Screen Keyboard) Freq: 524.00
65 (On-Screen Keyboard) Freq: 524.00
64 (On-Screen Keyboard) Freq: 456.17
64 (On-Screen Keyboard) Freq: 456.17
```

Figure 3.8. The GUI of the polyphonic synthesizer built in JUCE. Note the input box available to change systems of EDO.

This synthesizer relies on the MIDI to frequency function discussed in Figure 2.1 to achieve polyphonic output, and it works in an architecturally different fashion from the Max/MSP solution demonstrated earlier. Instead of altering the MIDI data sent to the program and then sending it along to the instrument, this application loads the voices of the instrument before it receives any MIDI data, and it will reload the voices of the instrument every time the system of EDO is changed. A working implementation of the instrument has been completed, and it is featured prominently at the beginning of the track “ur a meat sack, im a meat snacc.”



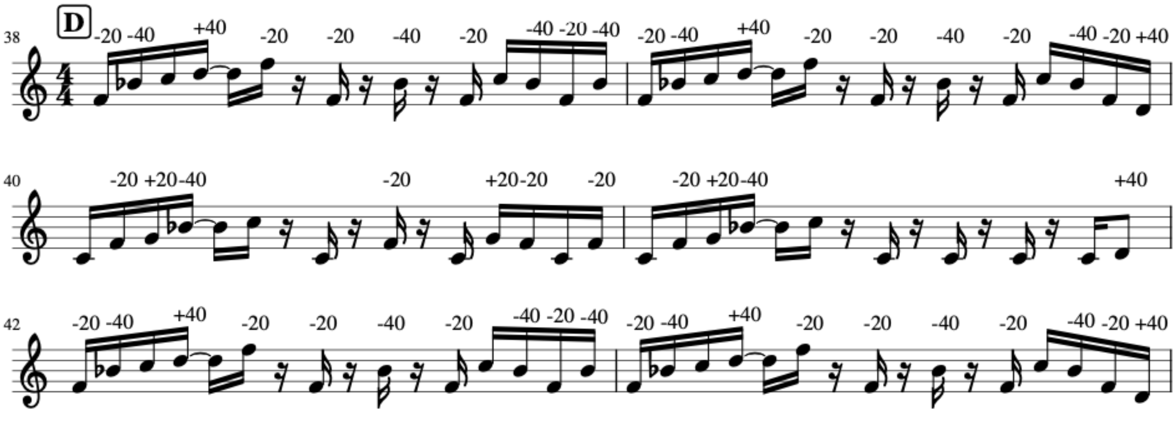
```
float doMidiToHzConvert (int midiNoteNumber, int sysEDO)
{
    //std::cout << "frequency calculating on sysEDO: " + juce::String
        (sysEDO) << std::endl;
    return pow(2, ((midiNoteNumber - 60.0) / sysEDO)) * 262.0;
}
```

Figure 3.9. The JUCE implementation of the function found in Figure 2.1.

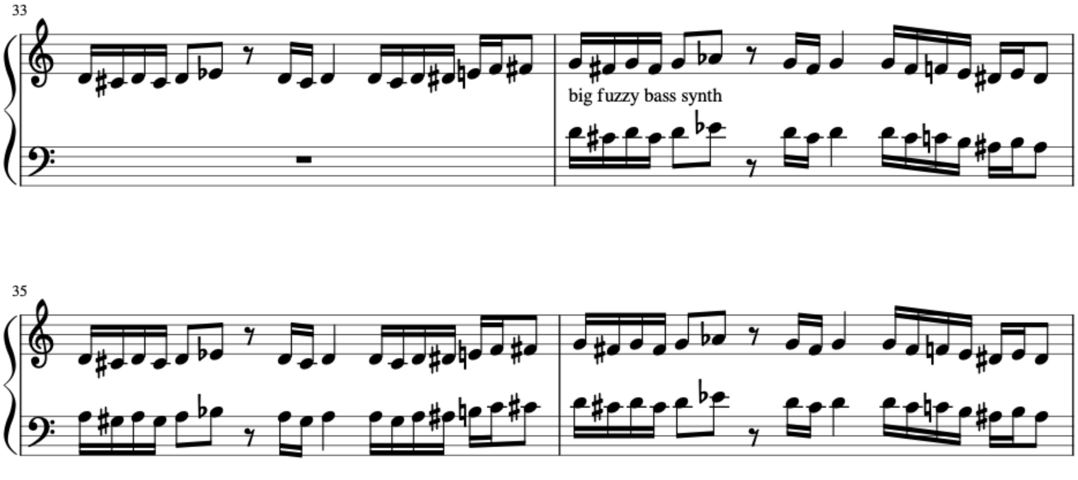
3c. The Music

To demo the M.A.F. Machine, an 11 minute set of microtonal music was written and recorded for an ensemble comprised of drum kit, viola, bass and lead synthesizers, and synthesizer controlled by the M.A.F. Machine. The bulk of the music was recorded at Berklee Valencia's Ann Kreis Scoring Stage over two dates in May 2021, and then a performance was overdubbed using the M.A.F. Machine to control Berklee Valencia's ARP 2600.

The difficulty of performing in alternate tuning systems meant that the set needed to be through-composed, with little room for improvisation. To notate the set, different methods were utilized for the viola and the synthesizers. For the viola, parts were notated using the traditional 12 tone system of notation, but text was written next to the note to indicate how many cents by which it should be altered.

Example 3.10. A part written for viola in 5 EDO

To achieve an accurate performance of the music, the viola player, Michael Lucarelli, applied colored electrical tape to the neck of his instrument at the correct intervals as defined by a mobile tuning application. Different colors of tape were used for different systems of EDO, and this system allowed him to keep track of where to place his fingers on the neck in much the same vein as a student learning to play the instrument for the first time. This method worked for the most part, though the tape occasionally created interference with the strings of the instrument, resulting in an unwanted “buzz.”

For the synthesizers controlled by traditional keyboards, notation was written using the traditional 12 tone system as well, but the notes performed on the keyboards were then sent through the retuning algorithm previously discussed to alter their frequency output in the synthesizer itself. Therefore, the notes heard by the listener were not necessarily the notes that would have been performed on the keyboards if they were using 12 EDO.

Example 3.11. The bass and lead synthesizer part of the piece demonstrated in Example 3.9.

No notation system was used for the performance on the M.A.F. Machine; its parts were, for the most part, composed during the overdub session due to limited time with the ARP 2600. A developed system of notation for the M.A.F. Machine, or a representation of its workings in a software environment, may be interesting elements to explore in future prototype iterations.

4. Innovative Aspects

The biggest innovation in the project lies in the construction of the hardware. The scheme by which a user can place “keys” into slots is designed to ease the process of changing systems of EDO, and it creates one possible ecosystem between software and hardware out of many ecosystems that have yet to be fully explored. The printing of the keys and the construction of the plastic hardware casing is designed to be intuitive and less complex than existing microtonal interfaces, while still retaining playability and user friendliness. Additionally, the notational styles and approaches to musical performance in microtonal tuning systems is innovative in that it attempts to define microtonal scales in a clean, readable, and accurate way.

5. New Skills Acquired

The M.A.F. Machine required skills in hardware prototyping that had to be acquired on the campus of Berklee Valencia. Soldering techniques, 3D modeling, 3D printing, laser cutting, and circuitry wiring were all skills that were developed during the construction of the instrument. These skills are beneficial for a career in many technical fields, and the glimpse into electrical engineering and its applications to music were helpful additions to an overall experience in music technology. Additionally, work was done in Max/MSP and the open source Arduino framework that had not been attempted prior to the project. On the musical side, new methods for the manipulation of analog synthesizers were explored, and the notational methods previously discussed had to be developed. Overall, the project involved many new technologies and approaches to musical expression that are relevant to a future career in music technology, hardware or software development, composition, and more.

6. Challenges

Expected challenges had to do with access to facilities, as well as difficulties in coordinating with musicians, recording engineers, and videographers to complete the musical side of the project. Of course, restrictions due to COVID-19 played a part in these logistical difficulties; however, most were resolved in a way which did not affect the project meaningfully.

Perhaps the largest unexpected challenge came from the hardware development process. What at first glance seemed to be a trivial, albeit labor-intensive, construction with well-defined specifications turned out to be a more complex process involving several different iterations and improvisations as processes that had been prototyped on paper proved ineffective in the real world, or as equipment (such as the 3D printer) broke down. Methods of overcoming these challenges included soliciting local makerspaces that had laser cutting and 3D printing options available, changing technologies used in the project -- including switching to an implementation of the device using buttons rather than pressure sensors to detect key press events -- and learning more about safe methods of conducting hardware development. Eventually, these solutions came together, and the project was completed, though the hardware part of the project took more time and was a more involved process than initially expected.

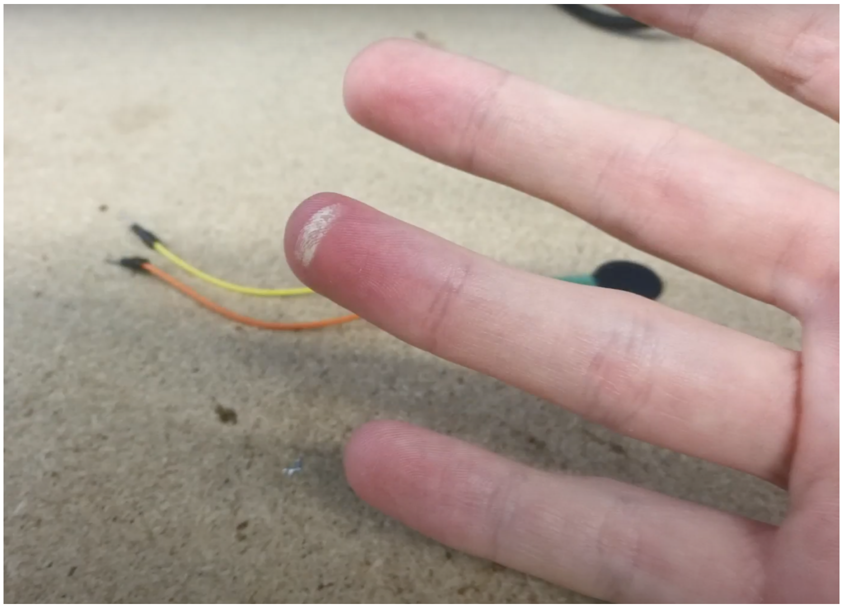


Figure 6.1. Burned finger early in the prototyping process.

7. Future Ramifications

The M.A.F. Machine has the potential for further refinement and addition to its functionality in subsequent prototype iterations. There are several additions to be made in future prototypes that would round out the instrument's functionality and add to its innovative qualities.

Earlier prototypes involved using analog pressure sensors to detect key placement and key press events. The sensors presented a logistical problem, however, in that professional ones are difficult to find at the quantities this project demanded, and homemade ones (built with velostat) were generally not accurate enough to be dependable. However, the use of pressure sensors in some capacity would enable the controller to, for example, detect the velocity of key press events, introduce modulation, create pitch bend options, and more.

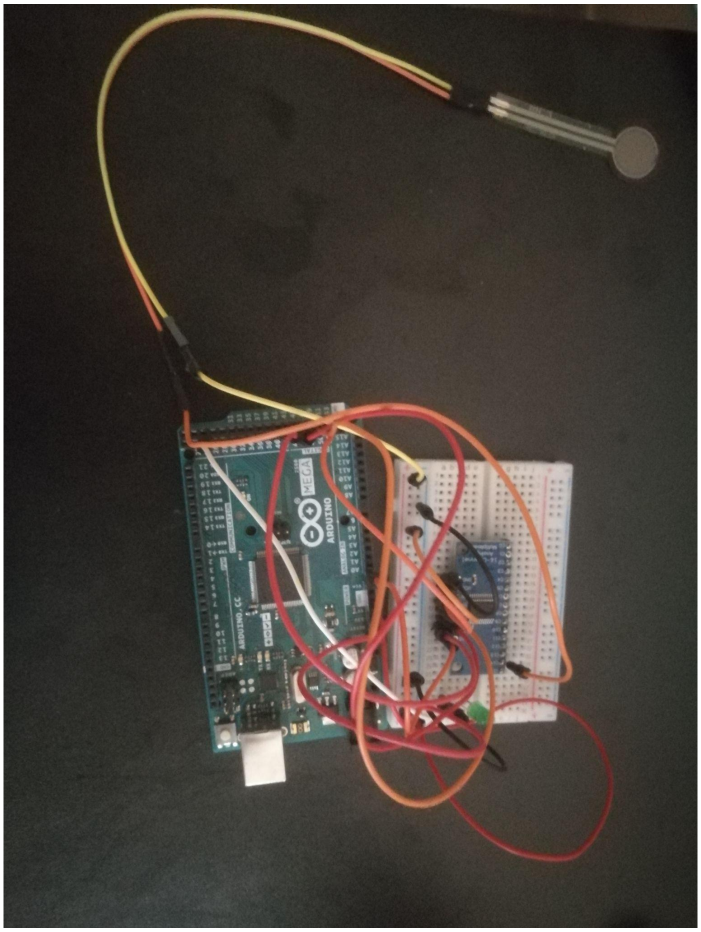


Figure 7.1. An early pressure sensor experiment.

Such options are vital to contemporary instrument design, and they would allow for integration with MPE or MIDI 2.0, which are frameworks that hold interesting implications not just for microtonality but for instrument development in general.

Another direction the prototype could take involves modularity. Early prints of the casing demonstrated that the geometric nature of the controller could allow for some interesting changes to the controller itself in a live scenario. One idea that was prototyped involved using magnets to connect different sections of the casing together, allowing a user to customize their controller to have greater or fewer sections.

Figure 7.2. Connecting different sections together to create a smaller controller.

This design could allow users to mix and match different sections to create a more unique controller, in the nature of a modular synthesizer. Different shapes of sections could be prototyped, such as rounded sections, or even vertically stacked sections. Adding this functionality would also allow users to play in microtonal systems much greater than 24EDO, which is vital functionality for any controller designed to facilitate performance in different systems of EDO -- many controllers geared towards this type of performance allow users to play scales with hundreds of notes.

8. Conclusions

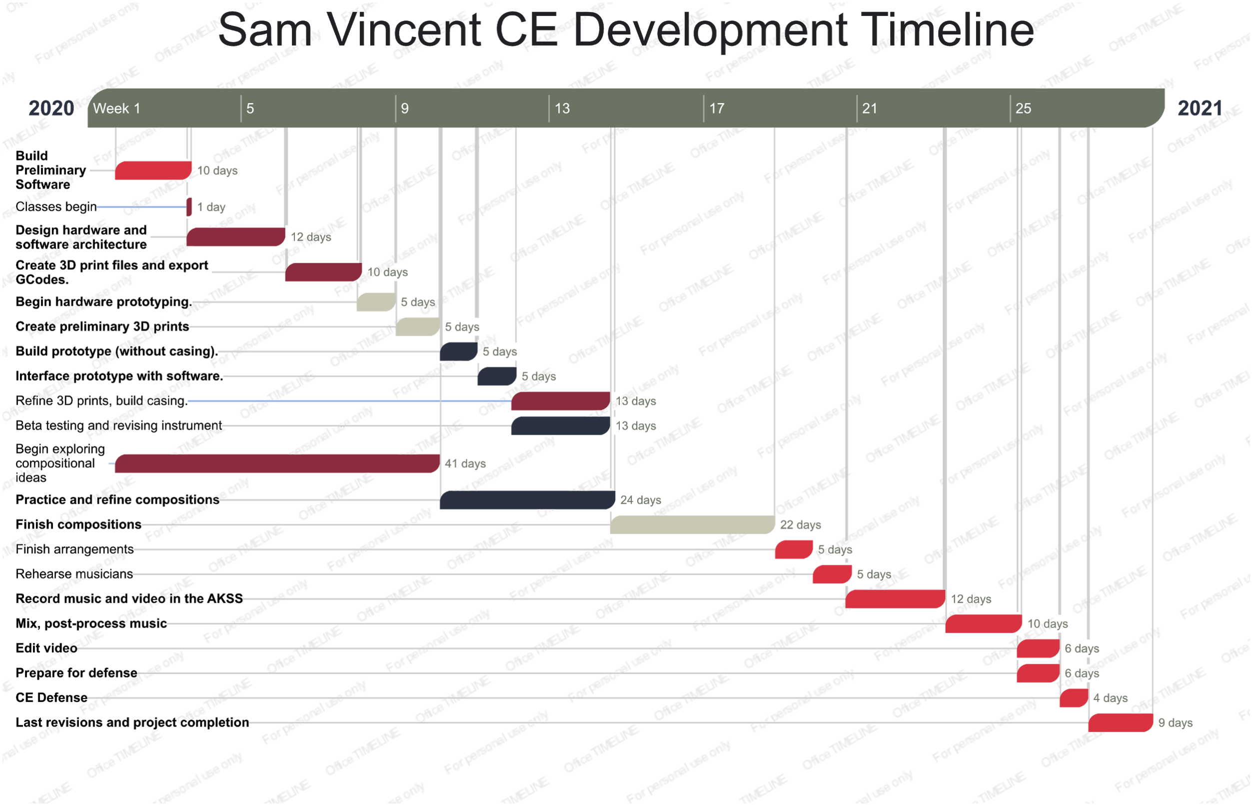
The M.A.F. Machine attempts to solve the problem of live performance in changing systems of EDO in a way that is both accurate and easy to understand. In doing so, it hopes to simplify the process of creating music in alternate tuning systems, and invite more musicians to explore the compositional and performative frontiers that microtonality contains. The solution implemented to achieve this goal acts as a well-rounded, working prototype, but there exists the opportunity to add more functionality in subsequent prototypes.

Appendix

1. Budget

Item	Proposed	Real
Prusa i3 MK3S 3D Printer	$932.42	$0.00
3D Printer Filament	$18.00	$0.00
Arduino Leonardo + Accessories	$50.00	$200.00
Raw Materials (Acrylic, wires, etc.)	$50.00	$75.00
Max/MSP (one year subscription)	$99.00	$0.00
Avid Pro Tools	$725.00	$0.00
Studio Musicians (3x 2 days of recording)	$1200	$0.00
Videographer (1x 2 days of recording)	$400	$0.00
Studio rental (2 days)	$800	$0.00
Camera rental (2 days)	$20	$0.00
Total	$4294.42	$275.00

2. Timeline


Sam Vincent CE Development Timeline
2020
Week 1
5
9
13
17
21
25
2021
Build Preliminary Software
10 days
Classes begin
1 day
Design hardware and software architecture
12 days
Create 3D print files and export GCodes.
10 days
Begin hardware prototyping.
5 days
Create preliminary 3D prints
5 days
Build prototype (without casing).
5 days
Interface prototype with software.
5 days
Refine 3D prints, build casing.
13 days
Beta testing and revising instrument
13 days
Begin exploring compositional ideas
41 days
Practice and refine compositions
24 days
Finish compositions
22 days
Finish arrangements
5 days
Rehearse musicians
5 days
Record music and video in the AKSS
12 days
Mix, post-process music
10 days
Edit video
6 days
Prepare for defense
6 days
CE Defense
4 days
Last revisions and project completion
9 days

Bibliography

Archibald, Sean: "Microtonal piano roll for Ableton," accessed November 2020, https://sevish.com/2016/microtonal-piano-roll-for-ableton-live/.

LilyPond Notation Reference, "Supported Notation for MIDI", accessed November 2020, http://lilypond.org/doc/v2.21/Documentation/notation/supported-notation-for-midi

Op de Coul, Manuel: "Scala Features", Scala. Accessed November 2020, http://www.huygens-fokker.org/scala/

Rubenstein, Michael: "Why 12 Notes to the Octave?" accessed December 2020. https://www.math.uwaterloo.ca/~mrubinst/tuning/12.html

Synthtopia, "Lumatone Intros New Isomorphic Keyboard for Microtonal Music & More." 26 January, 2020. Accessed 23 May 2021. https://www.youtube.com/watch?v=LfNIxTiApig

Various authors. "Collection of EDO Impressions," *Xenharmonic Wiki*, accessed 23 May 2021. https://en.xen.wiki/w/Collection_of_edo_impressions

Wolfe, Joe: "Note Names, MIDI Numbers, and Frequencies", University of New South Wales School of Physics. Accessed November 2020, http://newt.phys.unsw.edu.au/jw/notes.html