Matrix Game Board

Technical Documentation

A Engineering Project by: Dorian Todd
www.doriantodd.com

Background Information

LED matrix displays are two-dimensional arrays of light-emitting diodes (LEDs) arranged in rows and columns to create a grid. These displays can show text, images, animations, and other visual content by selectively illuminating individual LEDs within the matrix. LED matrices operate on the principle of multiplexing, where LEDs are not all powered simultaneously but rather in rapid sequence that appears continuous to the human eye.

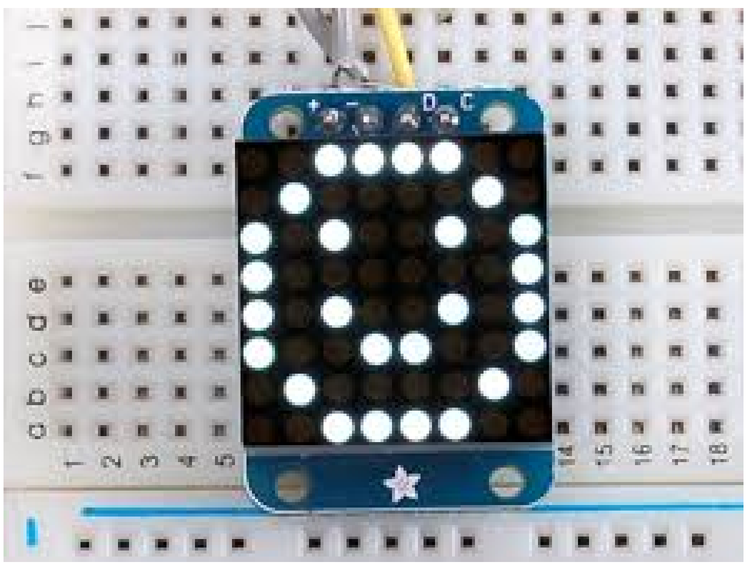


Image Source: Adafruit Industries et al. "Adafruit Mini 8x8 LED Matrix w/I2C Backpack - Ultra Bright White"

The Multi-Matrix Tic Tac Toe Board Is an original design created and programmed by Dorian Todd. The following documentation is intended to demonstrate the process in which the project was created, tested and optimized, and does not intend to act as an instruction manual, or guide of any type. If you want to make one yourself, please see the open source documentation on github.

Phase #1: Research

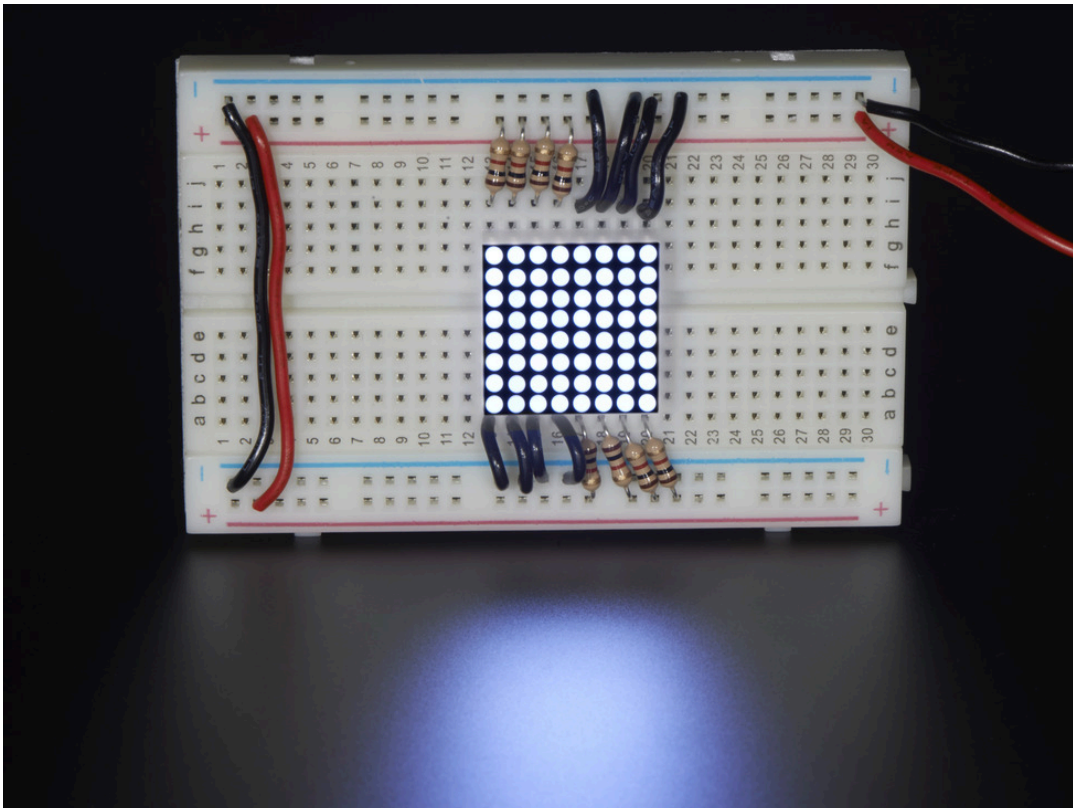
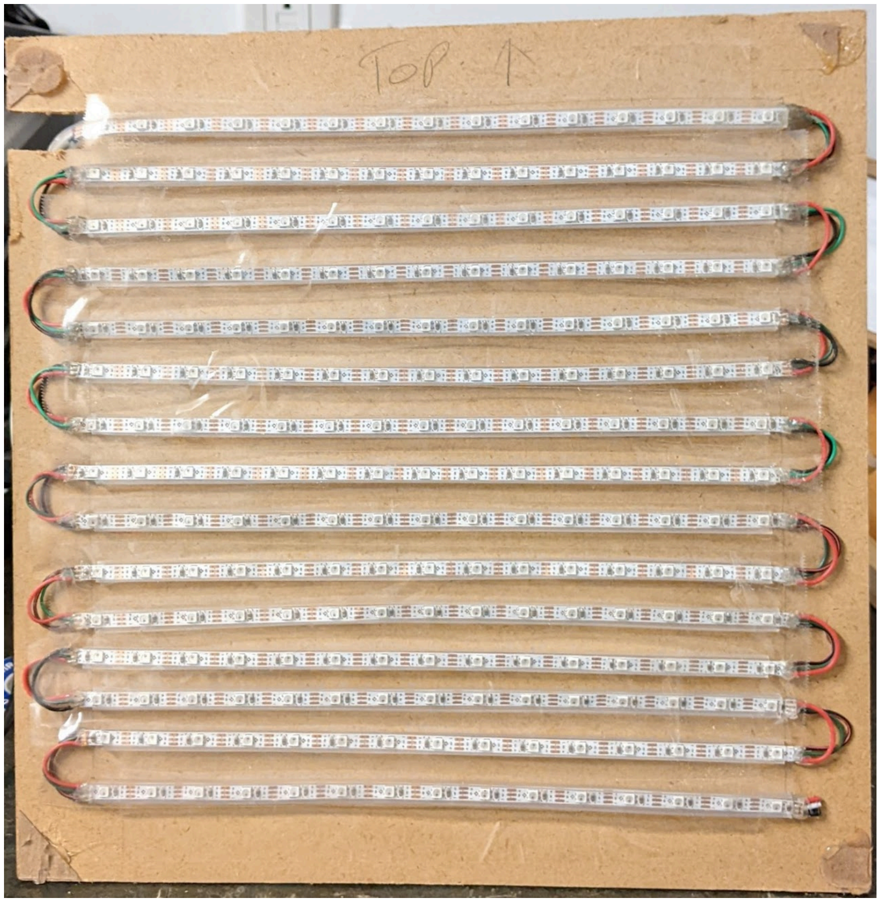


Image Source: Adafruit Industries et al. “Miniature Ultra-Bright 8x8 White LED Matrix.”

Throughout early to mid-March, I focused on the research phase for the project. When the SVMA Tic Tac Toe board design competition was initially announced, I had immediately envisioned creating an interactive tic-tac-toe board utilizing an LED matrix display to represent the game symbols. While the idea was compelling, I temporarily shelved it due to other priorities, only to revive it during this dedicated research period.



Having previously built an 15x15 pixel [LED matrix display for a different project](), I was already familiar with the fundamental concepts and implementation challenges. However, for this tic-tac-toe board, I wanted to explore new driving techniques and display configurations that would improve efficiency and visual appeal. Especially since the board needed to fit within a 6.5in^3 area. My prior experience provided a solid foundation, but I was eager to push beyond conventional approaches and experiment with alternative multiplexing methods.

Image Source: Hackaday et al. "An Impressively Large LED Matrix"

The concept of using matrix displays was particularly appealing because it combined digital display technology with a classic game. During my research, I explored various LED matrix types, microcontroller options, and power management solutions that would best suit a portable gaming device. I investigated similar projects online to understand common implementation challenges and innovative approaches to user interaction.

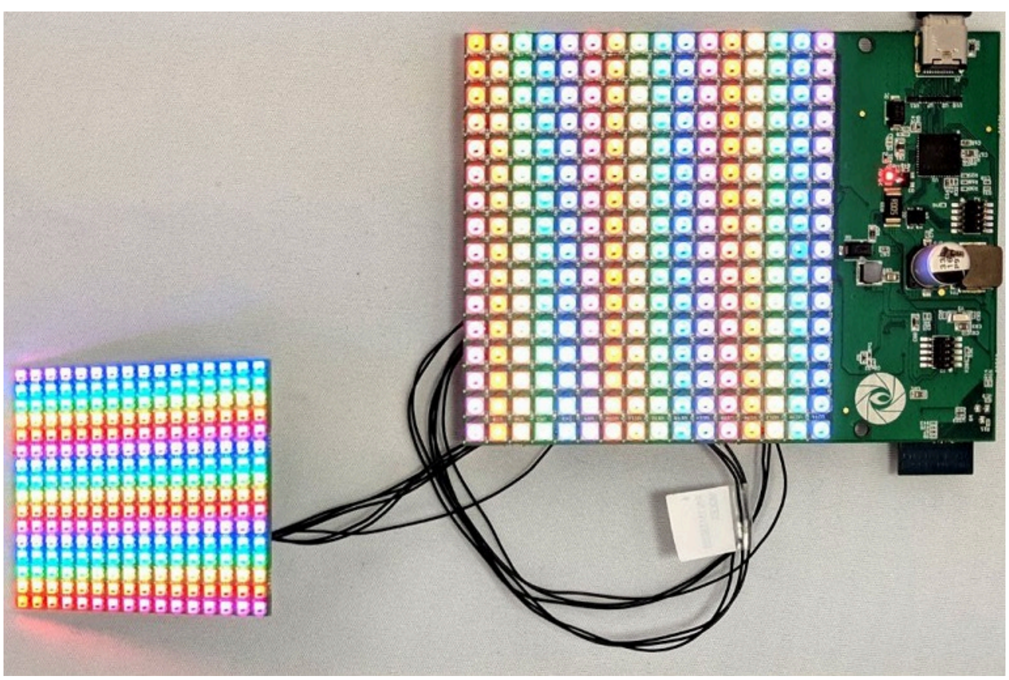


Image Source: Altium et al. "How to Add an RGB Matrix Display to Your Board in Upverter"

At the time of undertaking this project, my workload was already substantial with three other major projects in progress. Regardless, I’ve been inspired by Adam Savage's renowned "One Day Builds" methodology—a practice I've long admired and aspired to emulate in my own work. So I’ll be trying to condense the timeline down on this project. Maybe not just one day, but definitely less than a month.

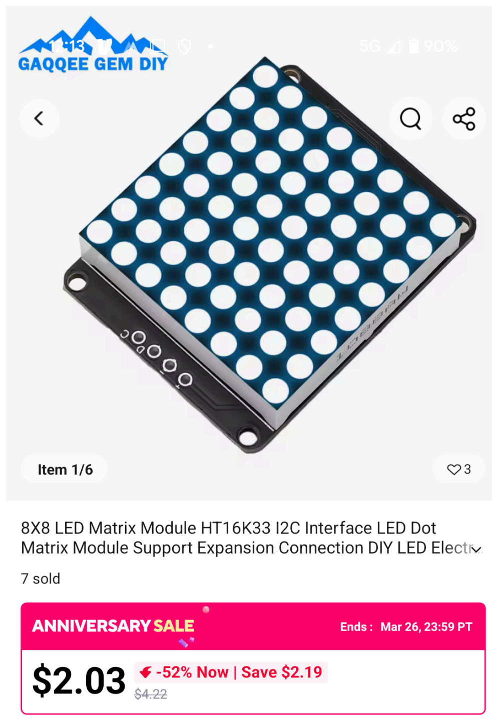


Image Source: Gaqqee Gem DIY listing on Aliexpress et al. “8x8 LED Matrix Module”

I ended up deciding on these small size lightweight white LED matrix boards from AliExpress. They fit the build constraints and are much cheaper than the options on Amazon and Adafruit. They come with a built-in matrix display driver chip, which makes it much easier to control the nine displays with a microcontroller. While I waited for the displays to arrive, I worked on the design for the casing and the button mechanism.

Image Source: Sac Valley Manufacturing et al. "SVMA is organized by Manufacturers for Manufacturers"

The rules of the SVMA Manufacturing Challenge state that at least one subtractive process must be used, so I'll plan on using either a laser cut wood or acrylic for the top panel which holds all of the electronics inside. They also state that electronics are optional, so I want to focus on making the actual construction of the board super high quality, so that the electronics are not the only shining feature. Both literally and figuratively.

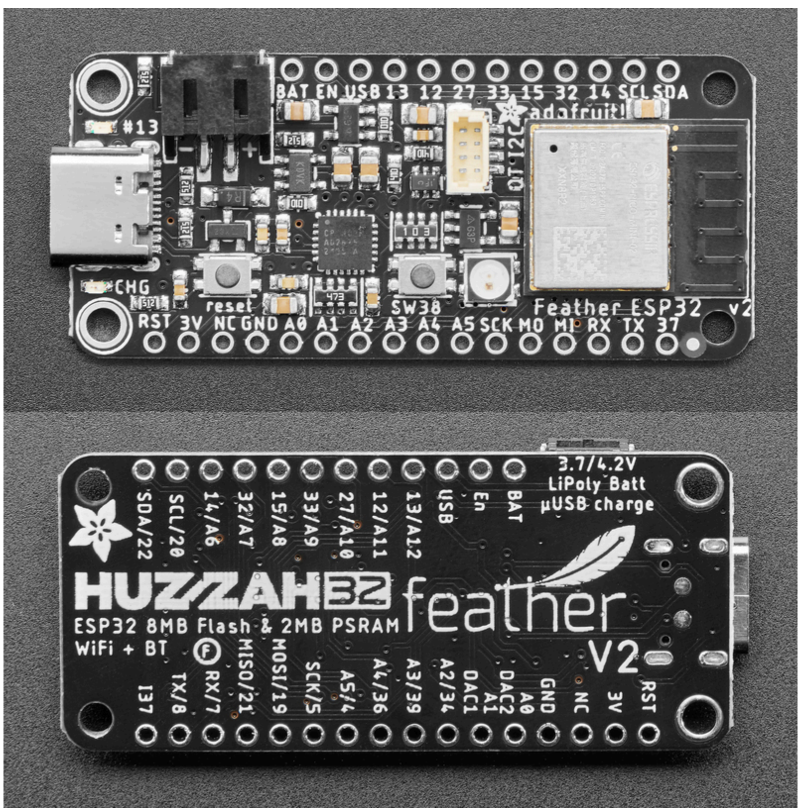
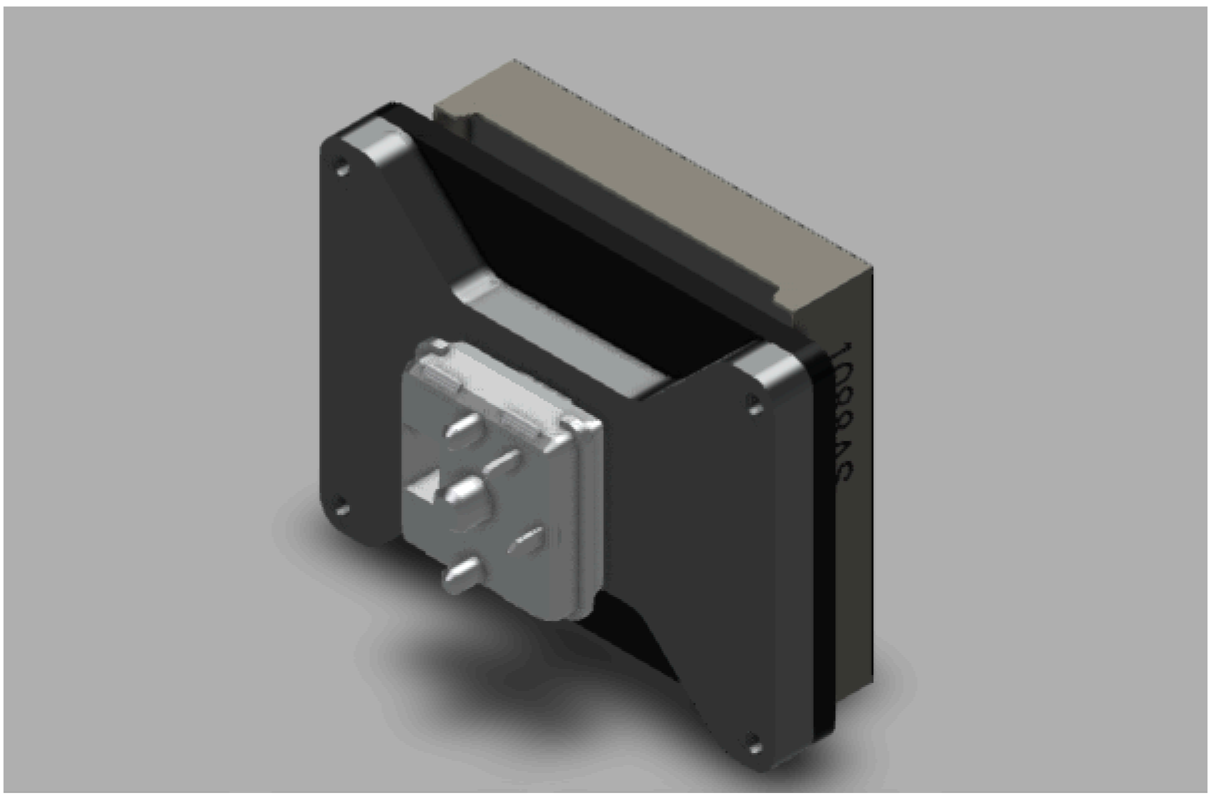


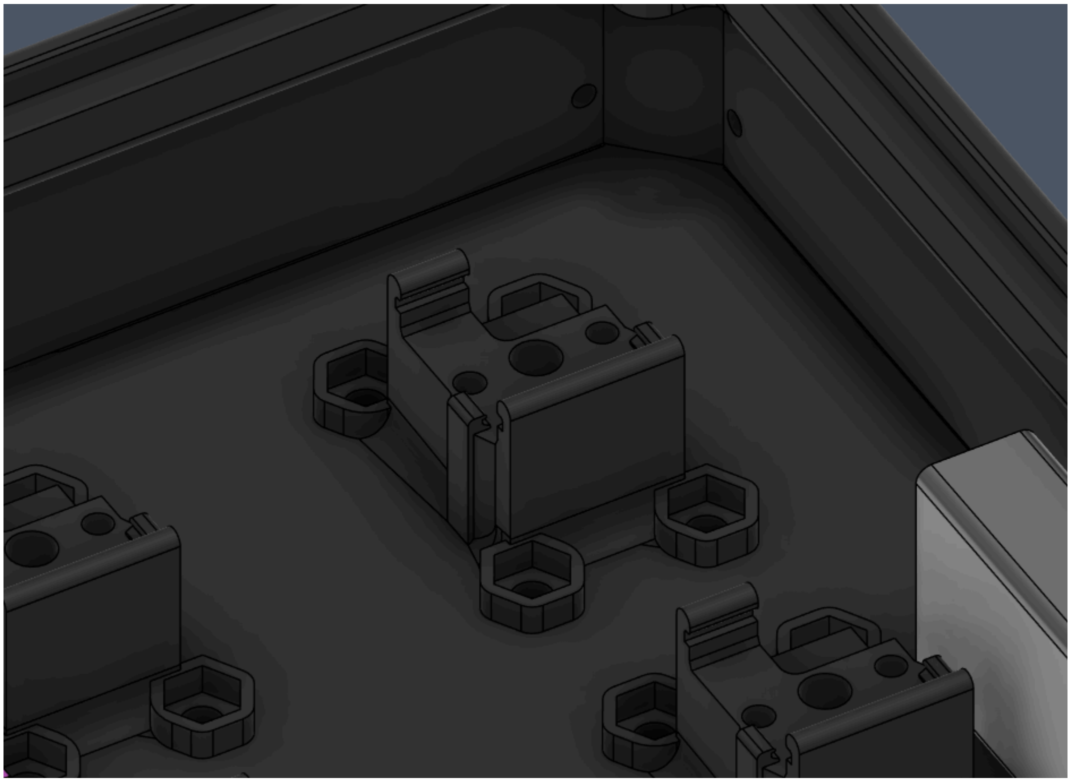
Image Source: Adafruit Industries et al. "ESP32 Feather V2 - Adafruit Learning System"

As far as microcontrollers go, an Adafruit Feather ESP32 is my board of choice; primarily for its battery input port, decent performance, and small form factor. Doing the napkin math I'm going to need at least 18 digital output pins and 18 digital input pins. Not many boards have this many pins, and also fit inside the small casing, so I'll end up needing to add in a multiplexing board that can handle all of the analog and digital inputs and outputs. Additionally, I may also need to introduce an additional source of power for the matrix panels such as a USB-C Power Delivery breakout board. In the past, I've been able to power large numbers of LEDs with just the 3.3-4.2v from an ESP32, but I'll have to do testing later on in the process to find out.

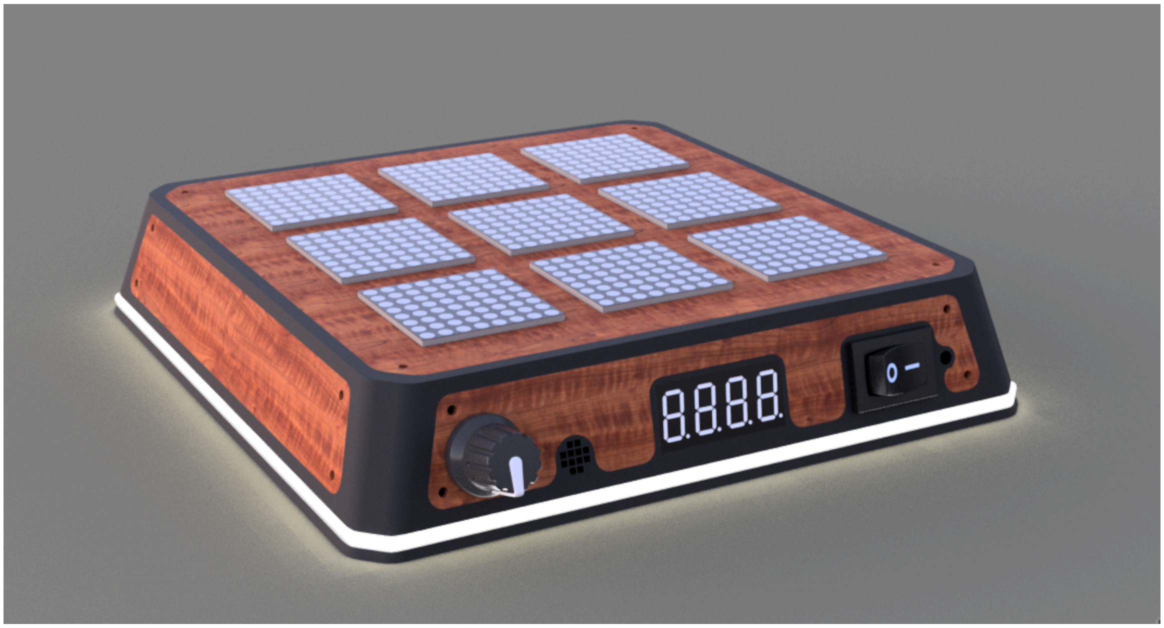
Phase #2: CAD Design



I'm designing the whole project in Fusion360, and I'm using designs from GrabCad for the PCBs, switches, and other electronic components. The first part of the design I tackled was the actuation of the matrix panels as buttons. I wanted to give the pressing action a tactile and premium feeling. At first I was looking at using cheap micro buttons that come in arduino kits, but I quickly realised that the surface area on the top of those buttons is too small to press evenly. Then I found that I could use low profile mechanical keyboard switches and get an even better tactile interface. The image above is the first revision of the connection from the matrix controller board and the low-profile mechanical keyboard switch.

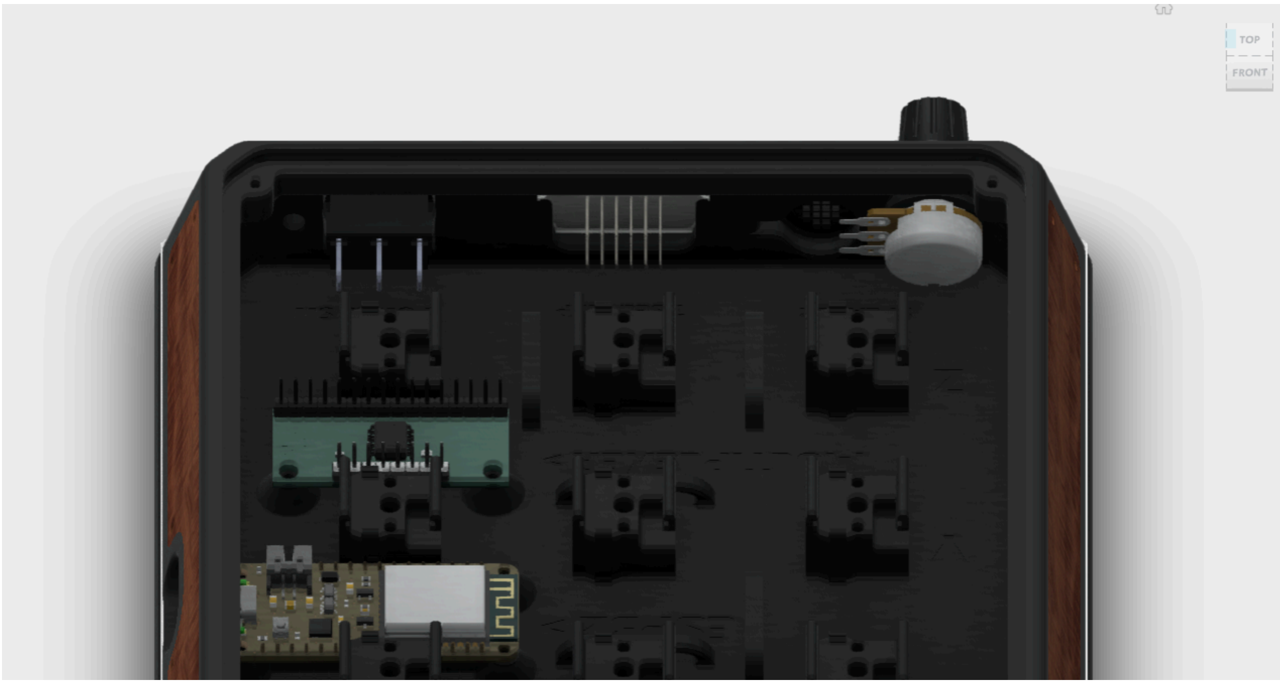


After making the switch to matrix interface, I designed a locking mount for all of the switches to go onto. Each “Matrix Cell” can easily snap into the locking mount 3D printed into the base of the device. One of my goals for the internals is neat cable management, so as the design progresses I intend to add dedicated channels for wires to pass through. Each matrix cell is exactly 1.6 inches away from each other, which gives just enough space to route wires and keeps the entire board just under 6.5 inches squared. (the maximum for the competition.)

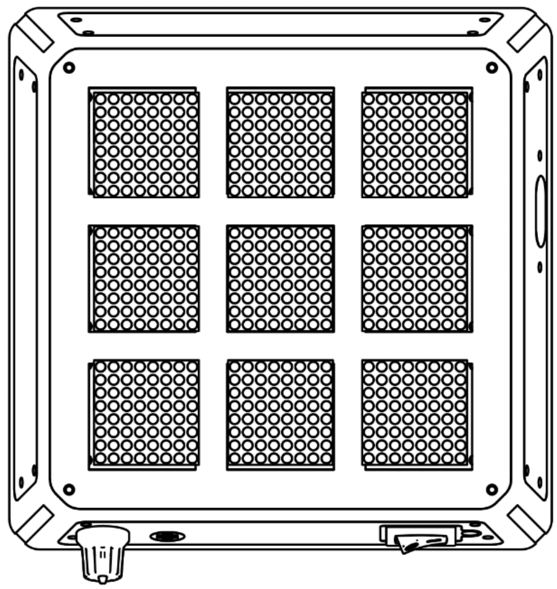


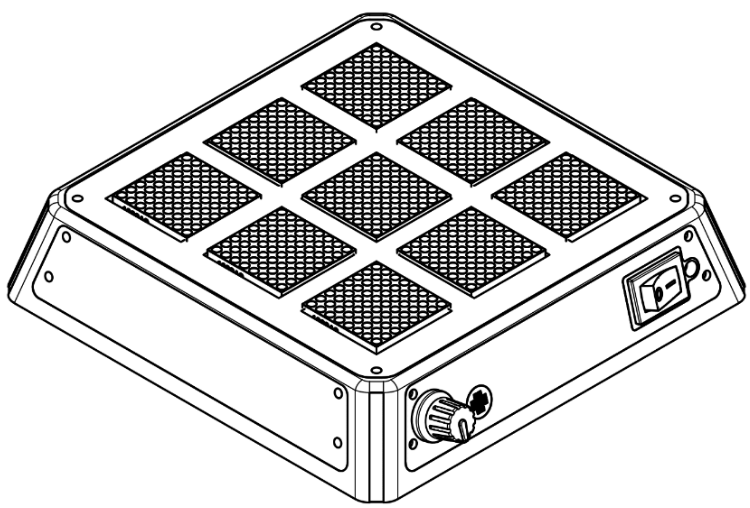
Above is the CAD design for V1. This design uses laser cut wood veneer on the sides to add in a rustic bespoke contrast to the industrial techy design. All of the black material seen is simply 3D printed PLA plastic. On the front panel, I've added in a power switch with status LED, a four digit 7-segment display, then a piezo speaker with volume control. This is all wired with connections behind the front panel. There is also a channel for a LED filament tube, which I'm still deciding if I should keep installed, since I'm not sure if it will look good. For the sake of project scope, I do end up removing a few of the features seen in this render.

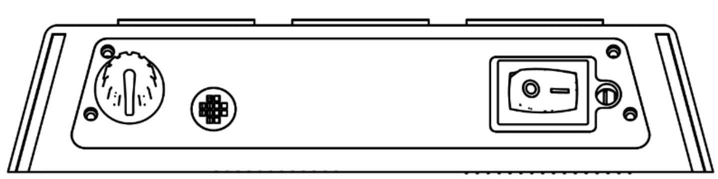
At this point I’ve already ordered all of the parts I need, or have them in my electronics kit, so it's just about getting the design optimized, and manufactured. As seen in the design, there is a dedicated slot for the ESP32 Feather board and a slot for the Multiplexer board.

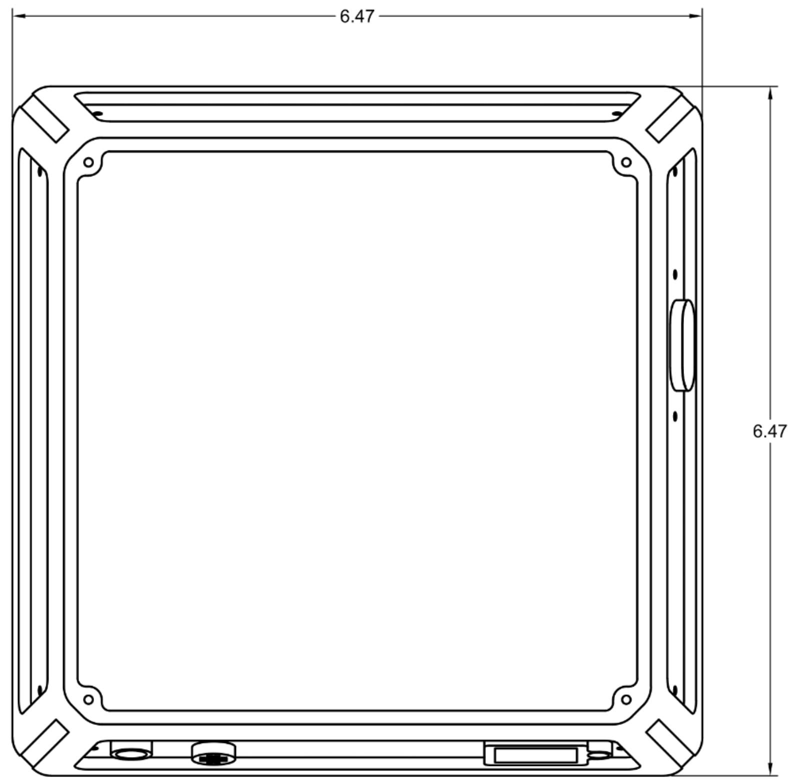


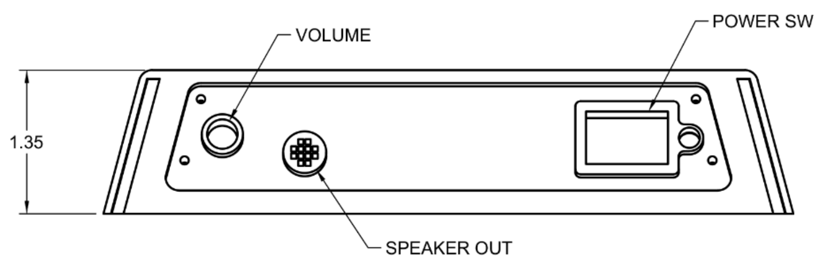
In this view you can see the four main modules in the front panel. Power switch, 7-segment display, piezo speaker, and volume knob. Each of these will be wired in with the rest of the inputs.

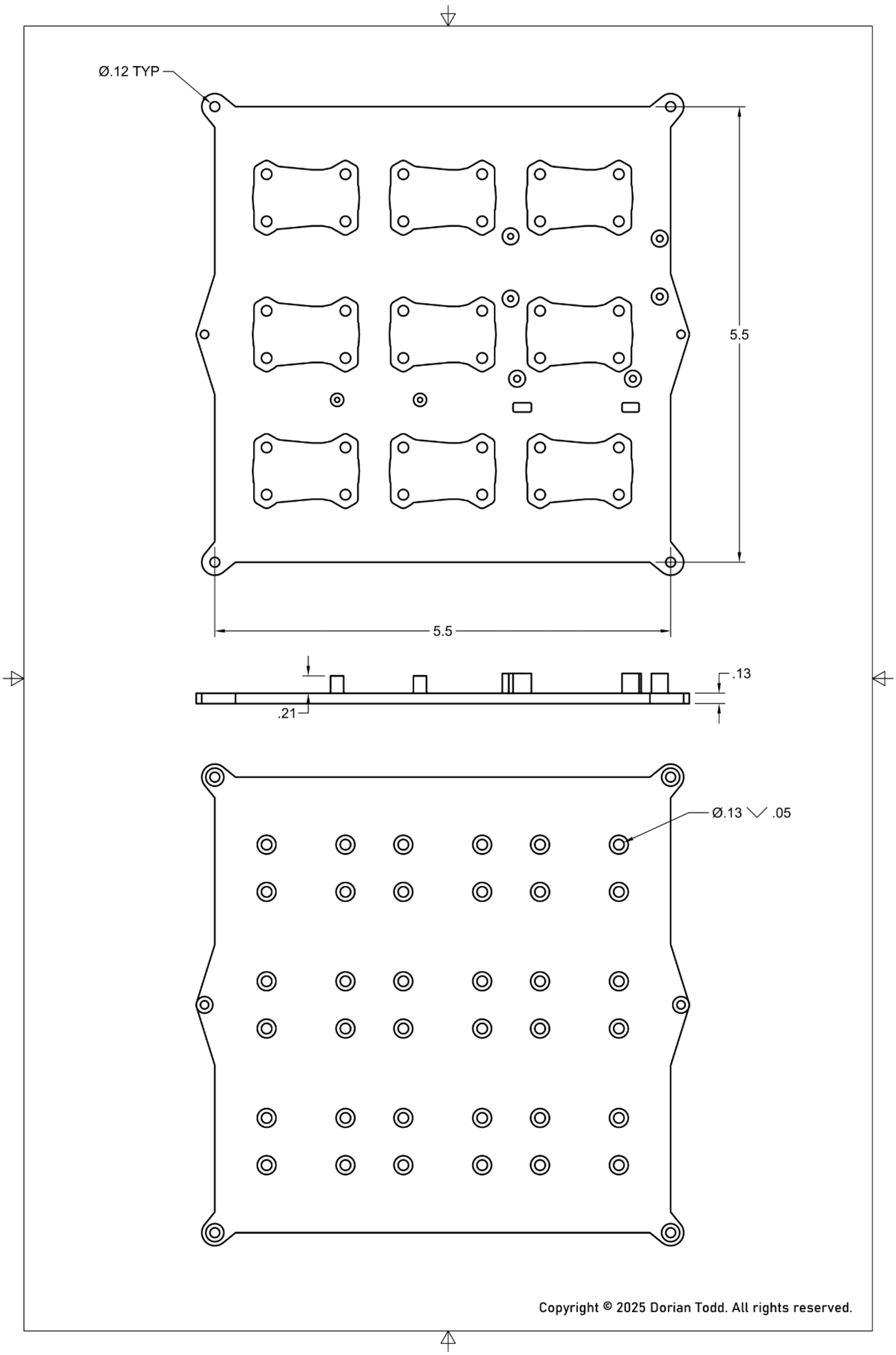


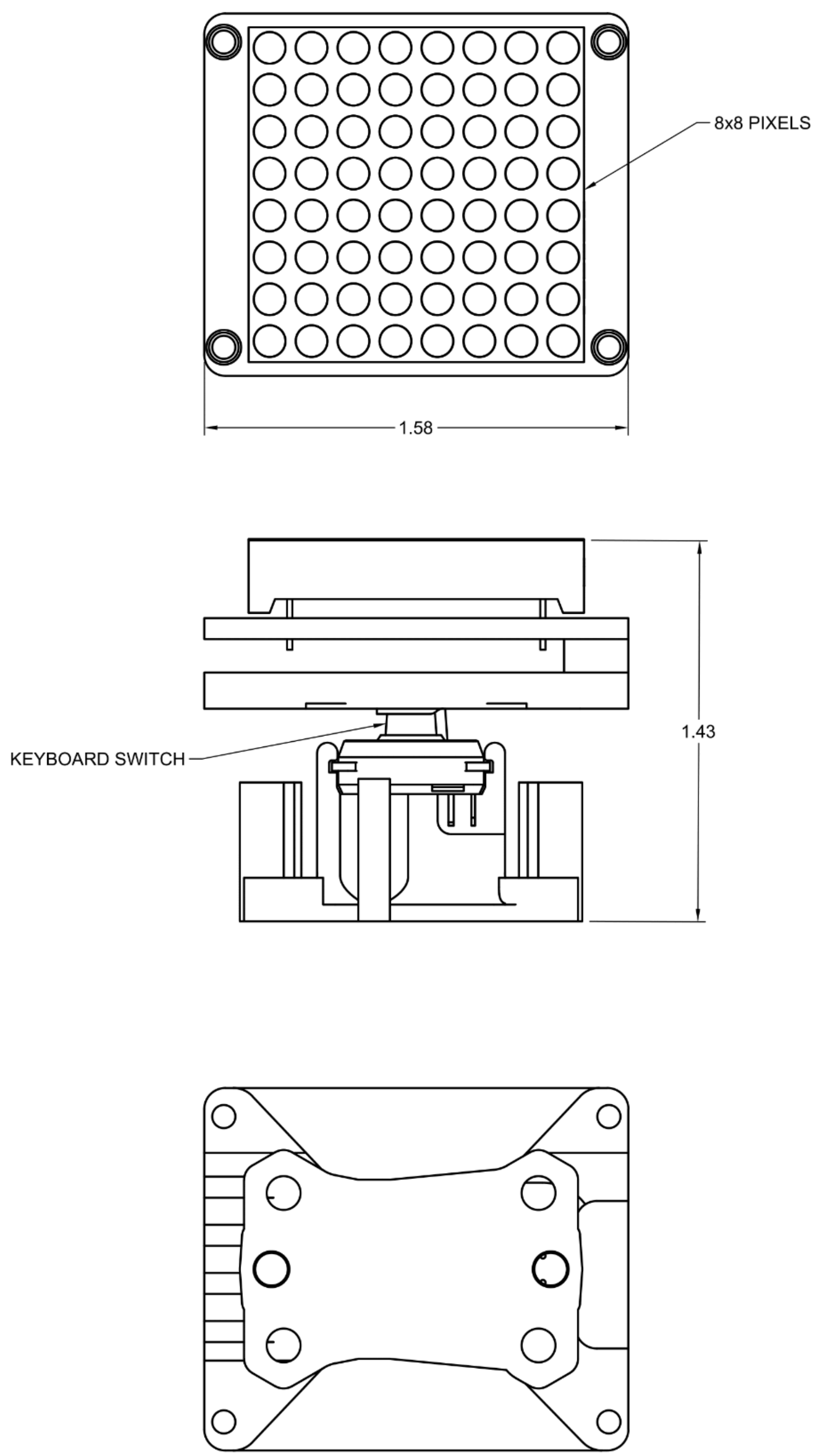




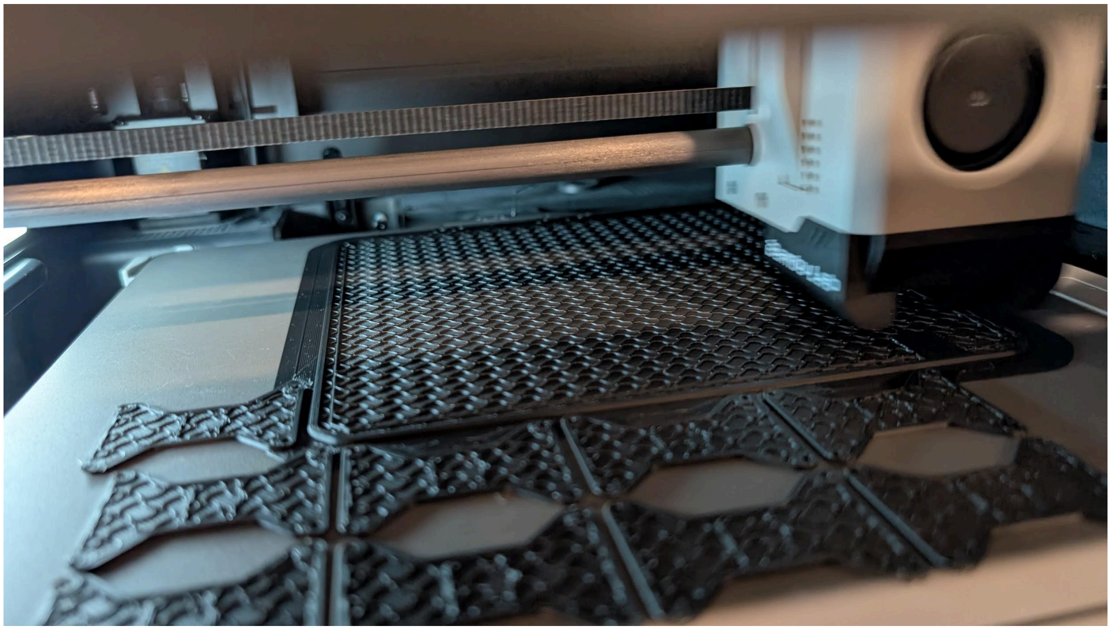

6.47
6.47


VOLUME
POWER SW
1.35
SPEAKER OUT

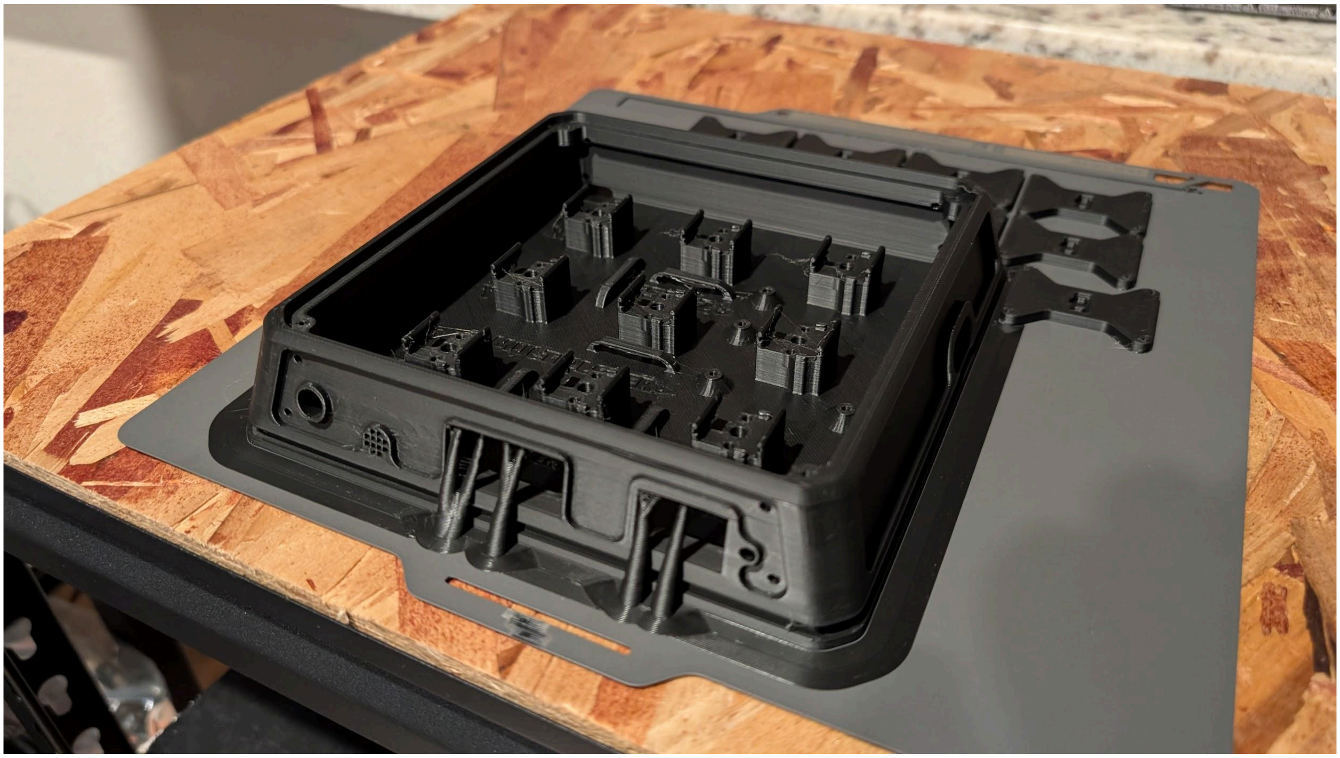

Ø.12 TYP
5.5
5.5
.13
.21
Ø.13 ⌵ .05


8x8 PIXELS
1.58
KEYBOARD SWITCH
1.43

Phase #3: Manufacturing



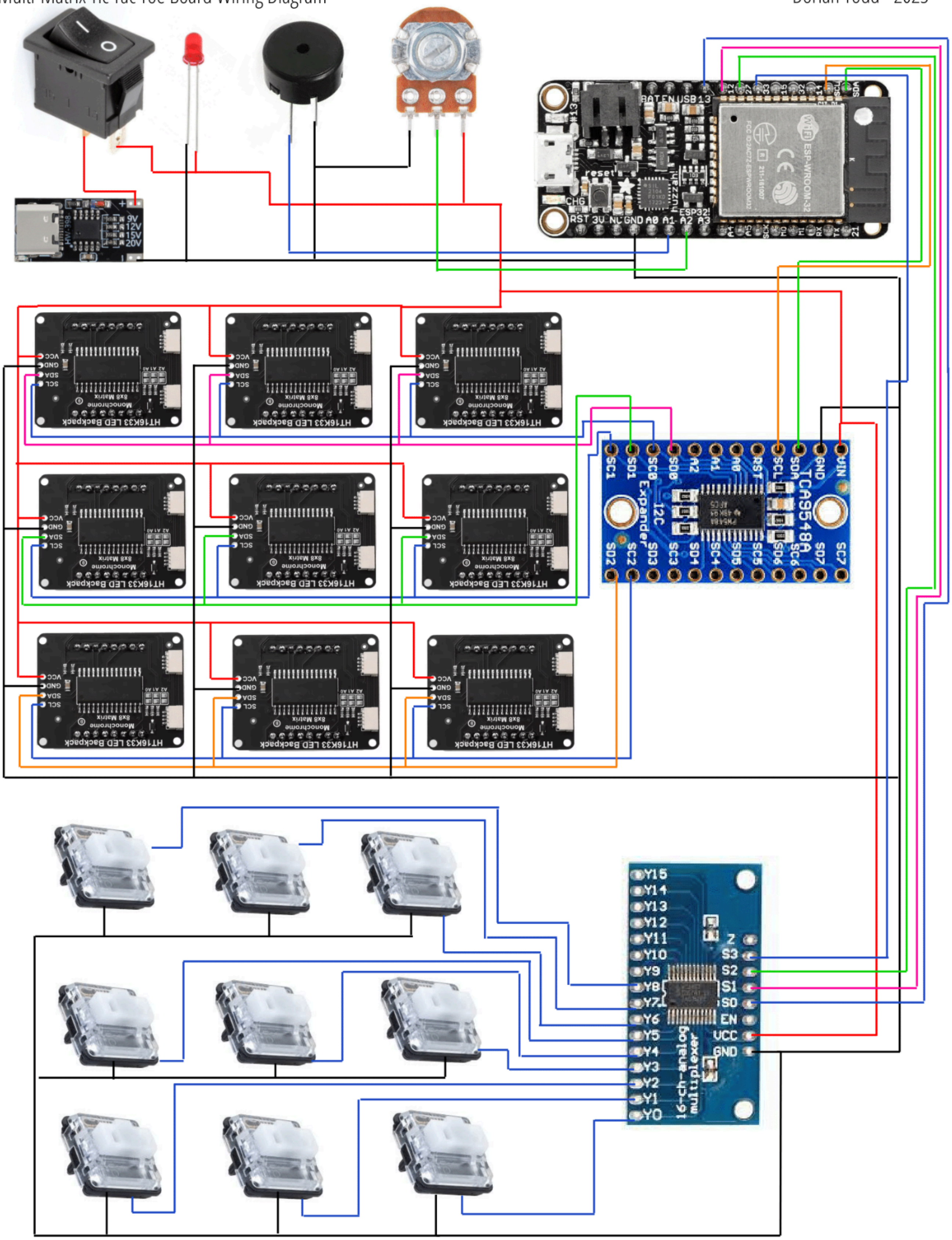
Starting out on manufacturing, I sent the design files from Fusion360 over to the 3D printer using the slicer software. The printer is a Bambu Lab P1S, which has great print speed and quality. This first prototype is printed in PLA filament, also known as polylactic acid. This is “a thermoplastic monomer derived from renewable, organic sources such as corn starch or sugar cane” *(TWI Global, 2023)*. It's also biodegradable within one lifetime. All of this is a long winded way of saying, I’m not too worried about waste if this first prototype does not work as intended.



Here is a finished print, right off of the printer. It took roughly 5 hours, and came out better than I was expecting. Now I need to cut out the wooden pieces, but first I'll make some progress on electronics fitment and wiring.

Multi-Matrix Tic Tac Toe Board Wiring Diagram

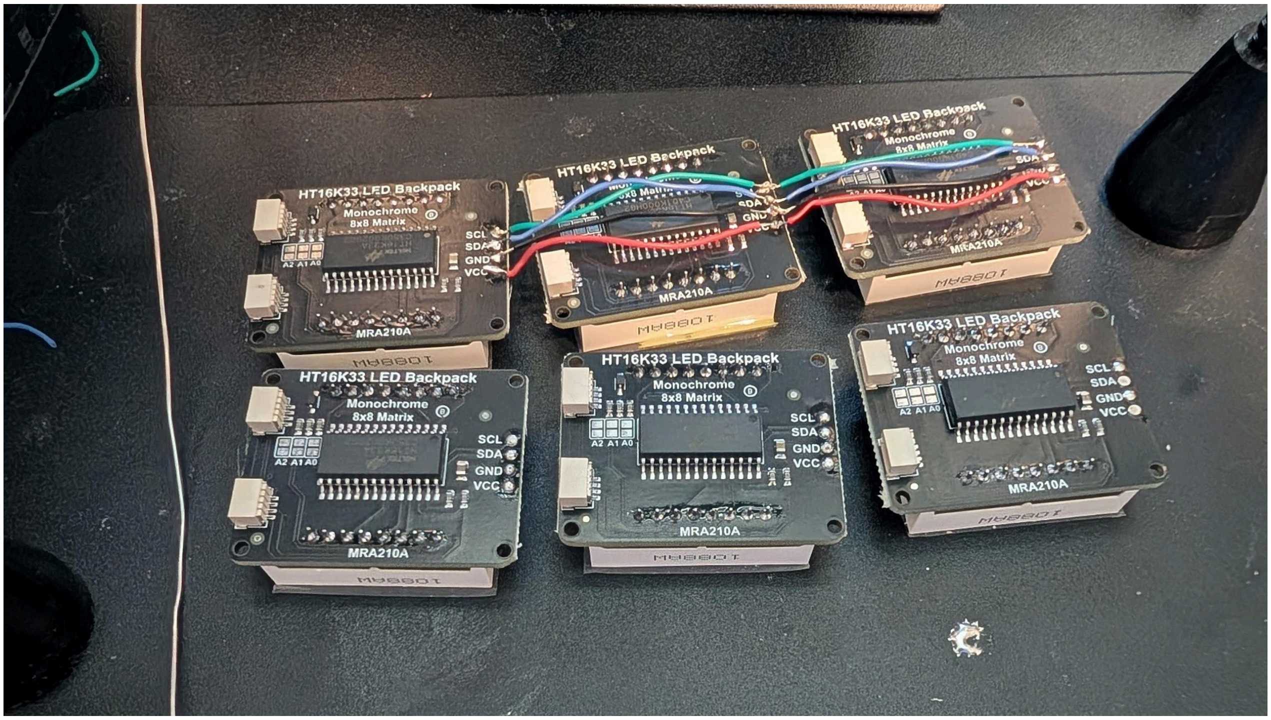
Dorian Todd - 2025



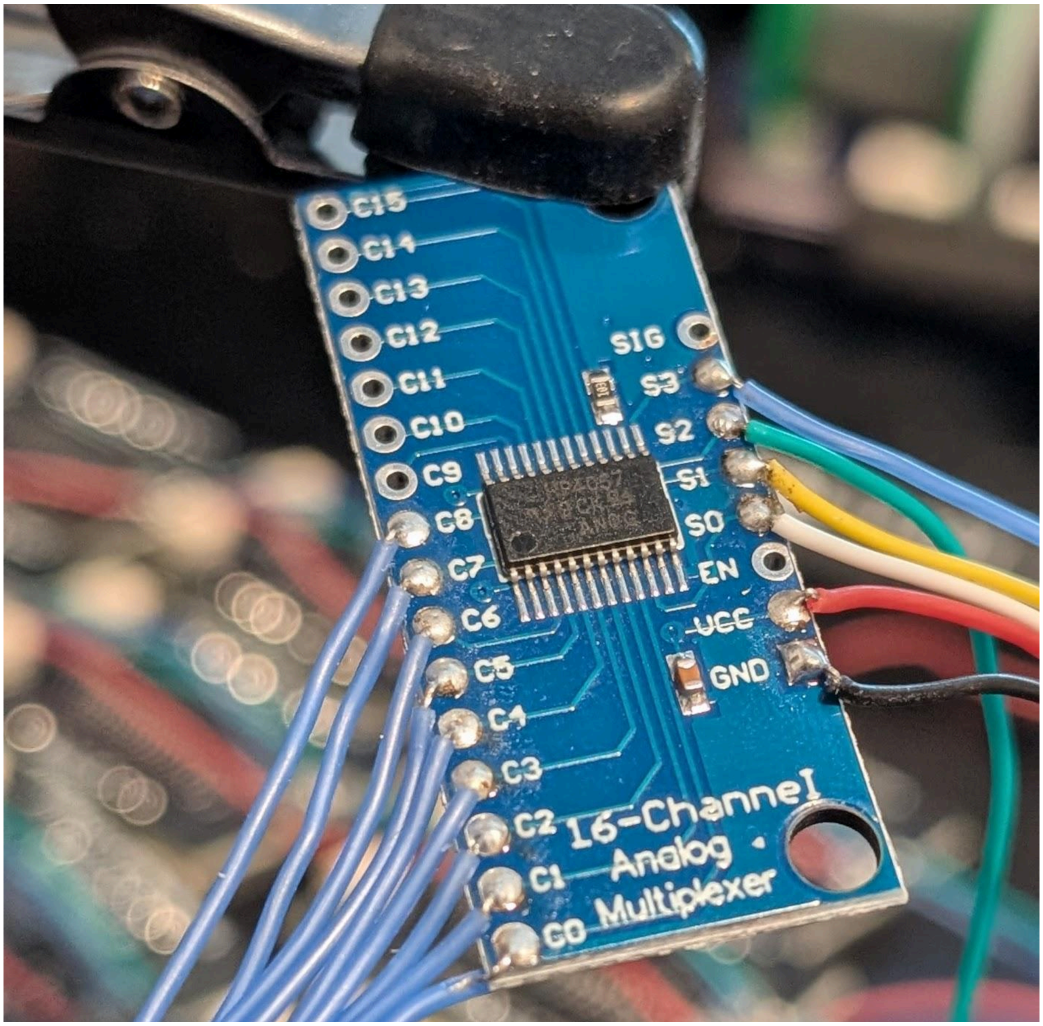
It turns out there are many more connections than initially anticipated, so we'll have to use two different multiplexer boards. The one connected to the matrix driver boards is an I2C expander board, and the one connected to the switch is a 16 channel analog multiplexer that handles the signals from all nine switches.

While this wiring diagram proved extremely useful, I did end up making many changes when actually wiring the boards, particularly with the data out pins. More details on the exact items used can be found on Matrix Game Board Bill of Materials, but here is a general list:

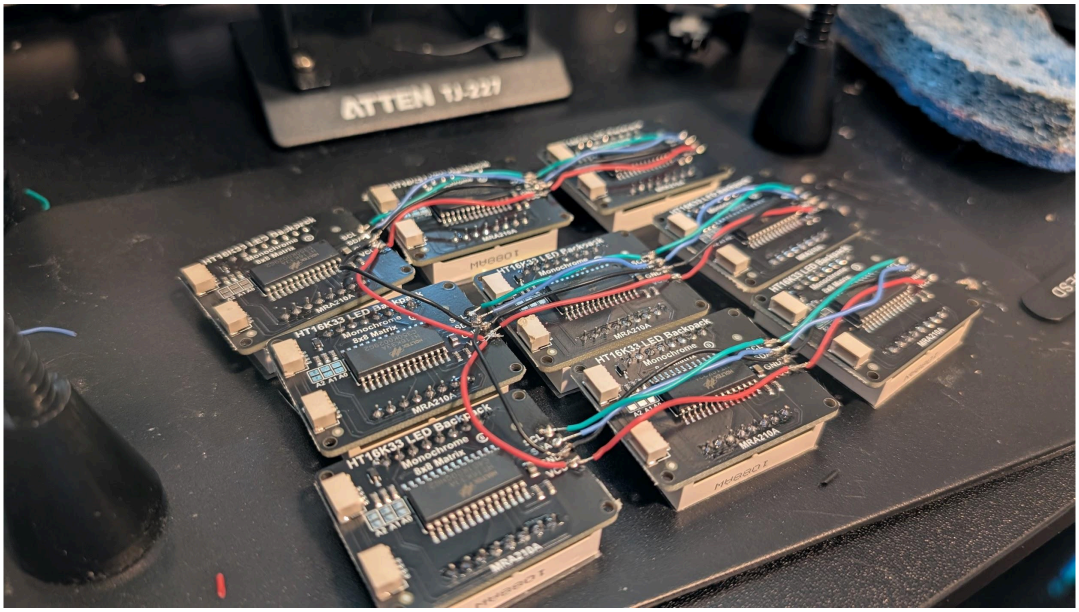
- - Adafruit ESP32 Feather V2
- TCA9548A I2C Multiplexer
- 9x 8x8 LED Matrix Modules (HT16K33)
- 9x Kailh Keyboard Switches
- 1x CD74HC4067 Analog Multiplexer
- 1x Volume Potentiometer
- 1x Piezo Buzzer
- 1x Power Switch
- 1x Power LED
- 1x Lithium Polymer 1000mah 3.7v battery



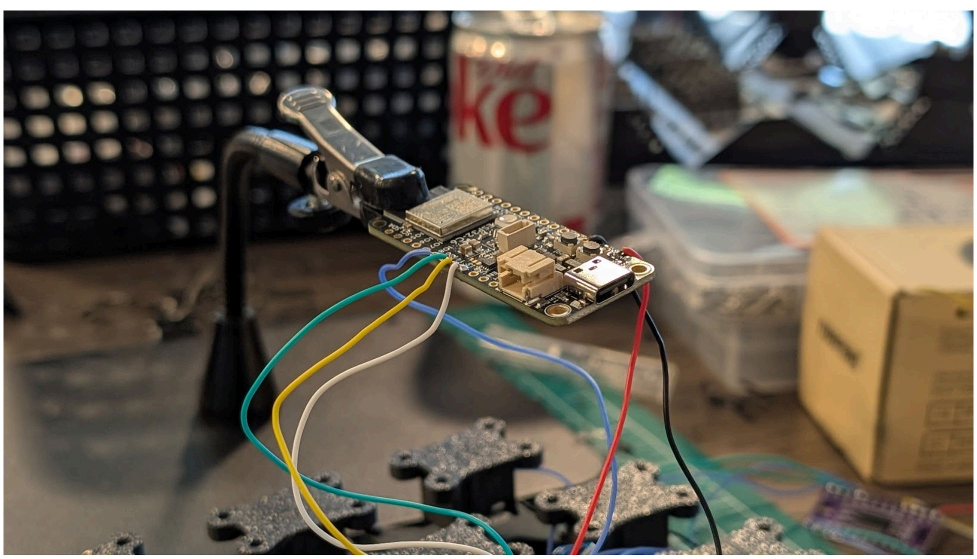
Given the configuration of the displays, we can wire them each in series and utilize the i2c memory addresses to drive signals to each display individually. The displays are soldered into rows of SCL and SDA connections, and have shared VCC and GND across all displays.



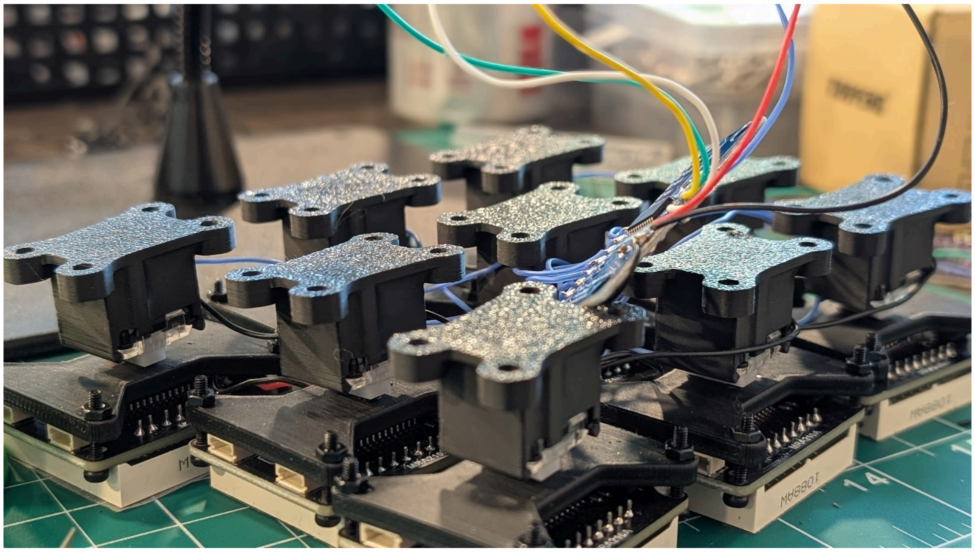
The keyboard switches which are actuated by the pressing on each matrix display all have signal wires that go to this 16-channel multiplexer, which handles the button states, decreasing the 9 inputs to the microcontroller down to just four.



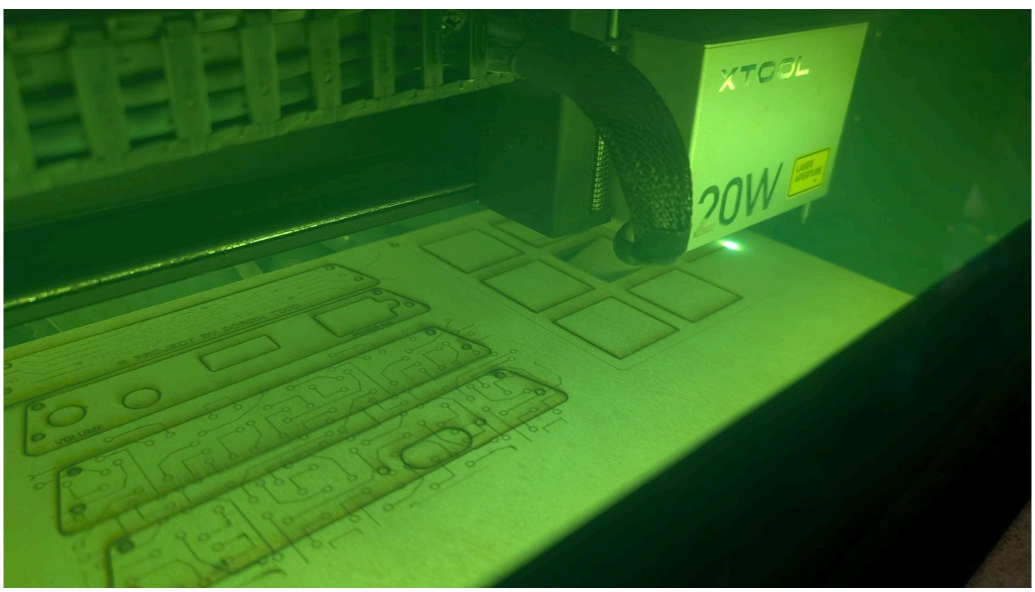
As you can see here, the matrix displays are all wired together. In retrospect this made it almost impossible to move around without breaking connections, but was necessary for testing. Additionally the wire size was likely too small and broke off the soldering pads on occasion, but a small wire size was necessary to keep the electronics small and lightweight.



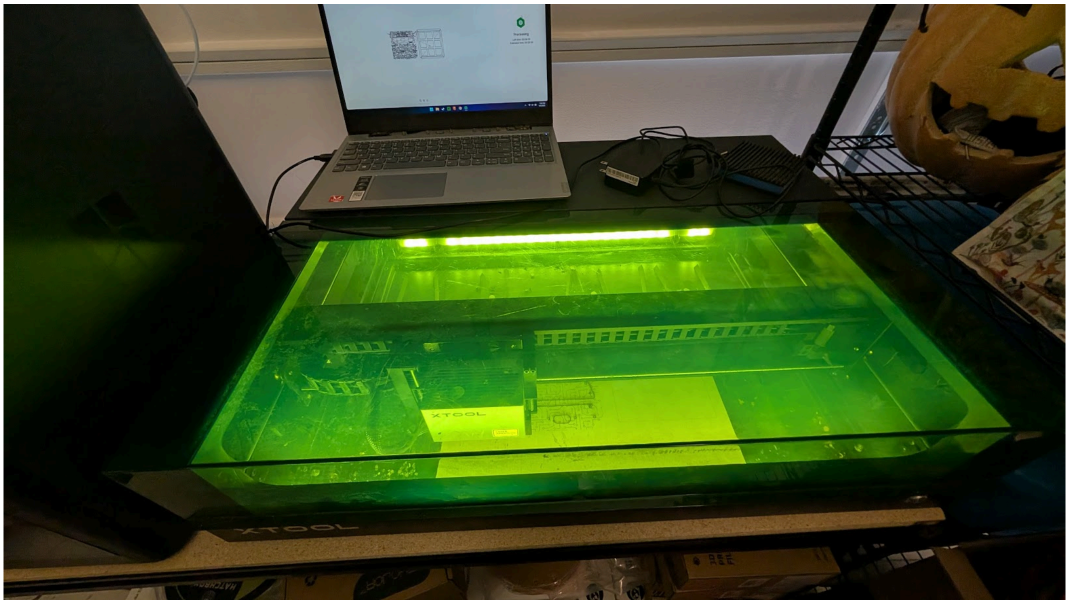
Not many wires need to route back to the Adafruit Feather, thanks to the multiplexers. This is great, but all of those connections still exist, just elsewhere on the wiring harness.



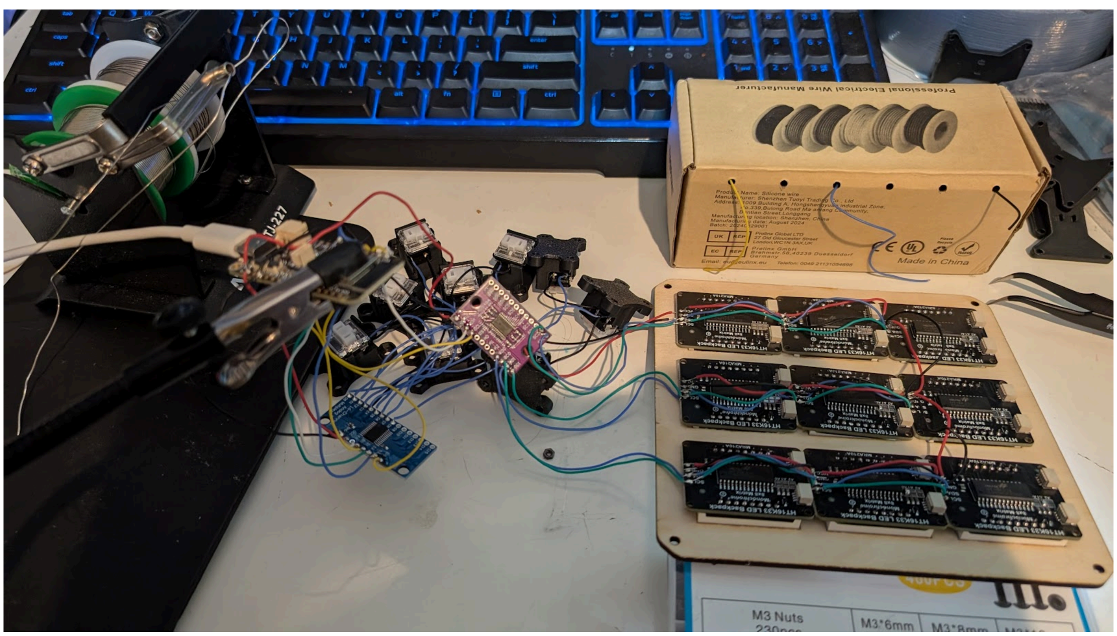
All nine switches connected to all nine displays. Putting this together demonstrated that the connection between the switches and the panels was somewhat weak, and would need to be supported in another way.



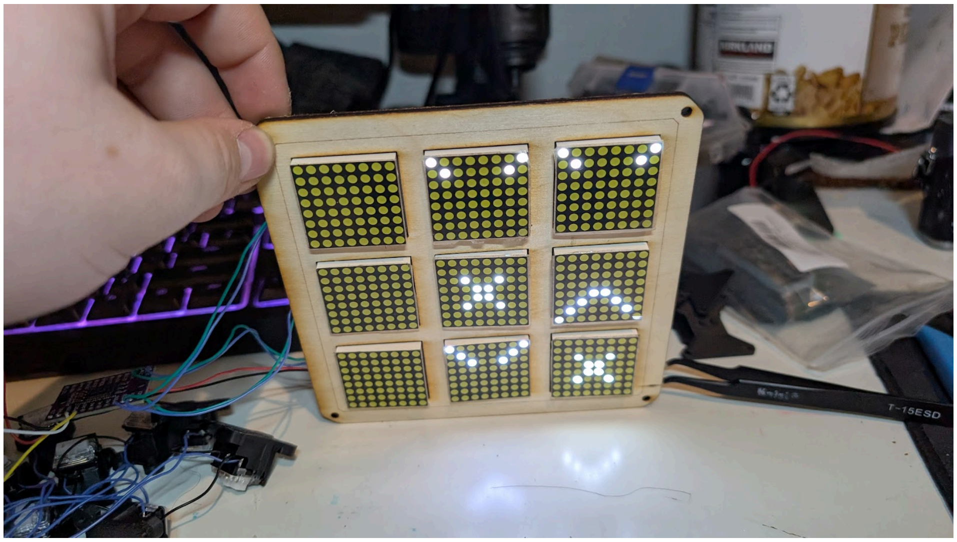
Taking a break from soldering and wiring, I focused on getting the side and top panel laser cut out of some scrap wood. This batch of laser cut parts was made to test the interference fit of the panels, and to hold the matrix displays in place during testing.



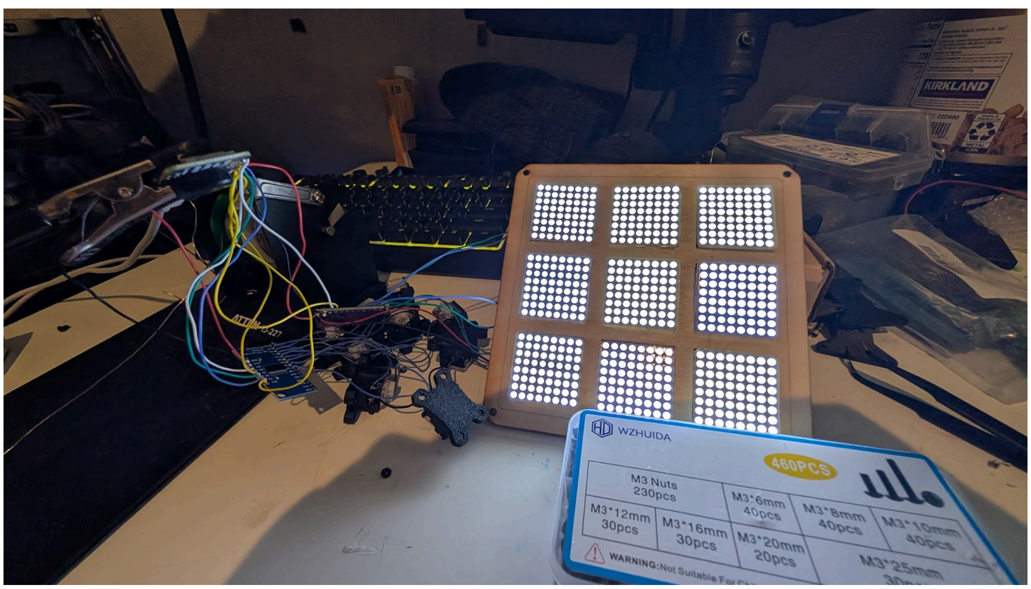
All of the parts for this project were cut on a XTool S1 20W laser cutter. It's quite effective at cutting and engraving thin hobby woods, but is underpowered with metals and other thick materials.



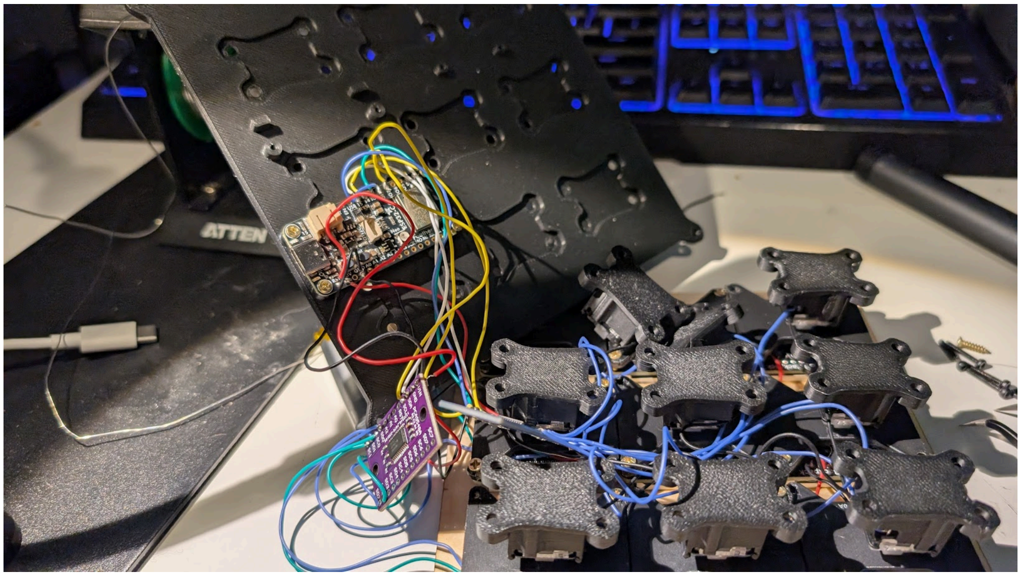
Using the newly laser cut top plate, I was able to test the panels in a more controlled format, without the risk of breaking connections. In this stage I began progress on writing simple Arduino Code that could help me diagnose issues.



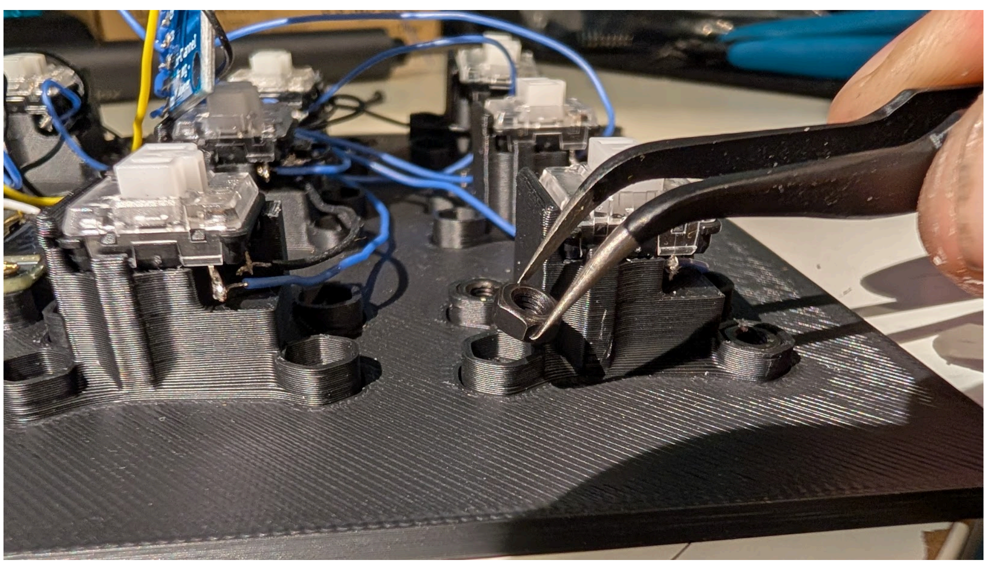
Testing was slow, and required a lot of patience. Utilizing code assistance tools I was able to make a functional demonstration with most of the displays showing pixels.



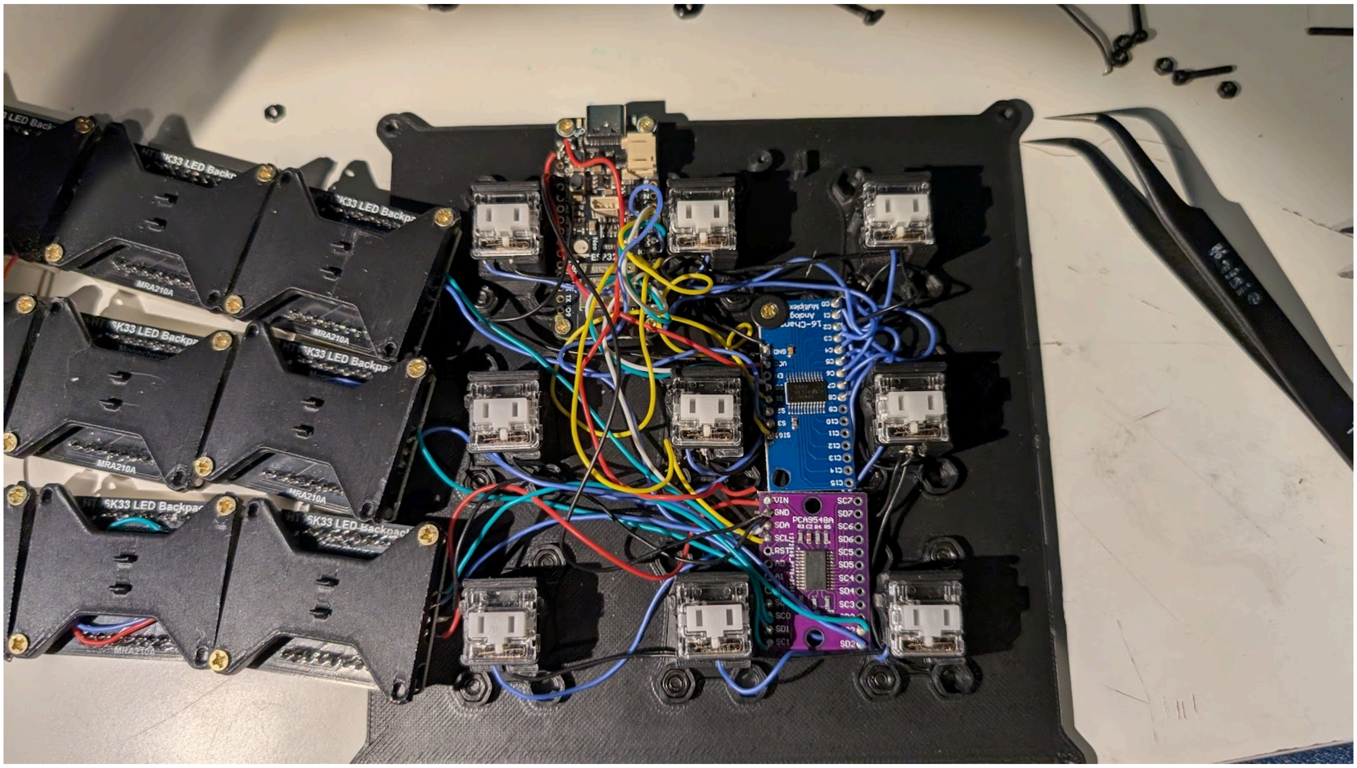
Eventually I was able to create a full illuminated grid of squares, which was great to see after hours of blank matrix displays. Additionally I was able to get the switches functioning separately, and now simply needed to combine the two working programs.



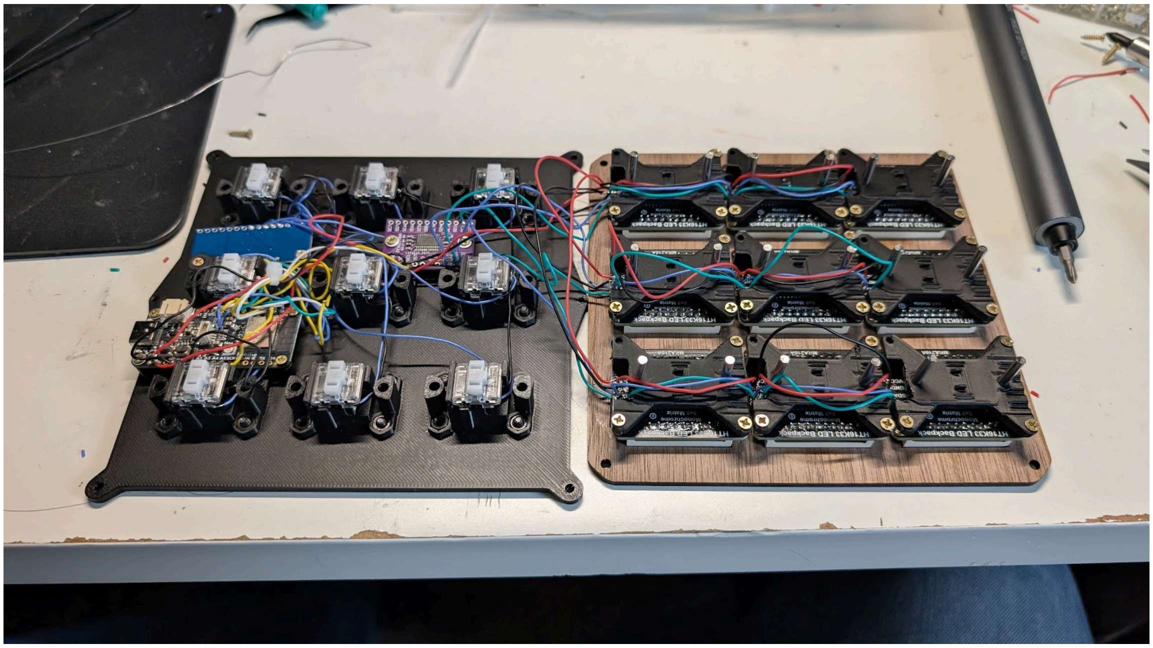
While the testing software proved challenging, it was also difficult to to assemble the project with the lengths of wiring tangling together, and many pieces falling out of place. In the photo above, it's clear how disorganized the assembly process was at this stage.



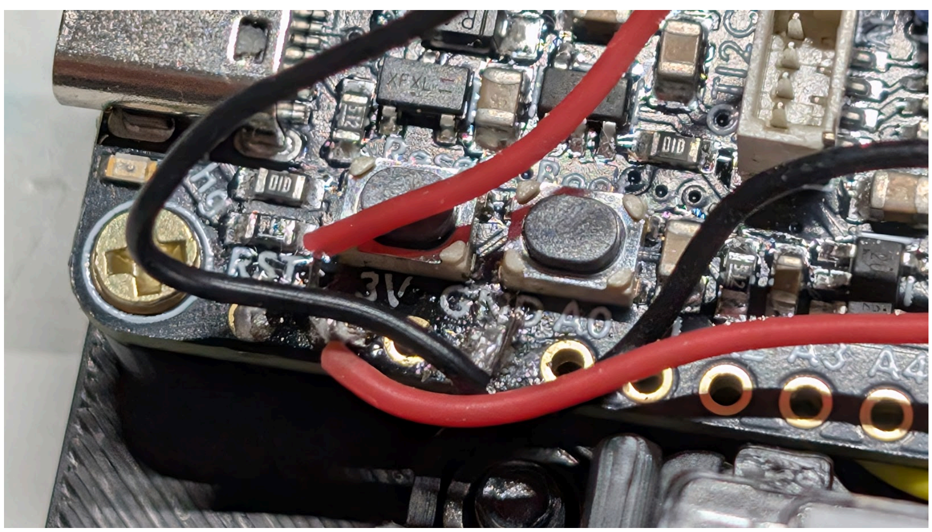
I backtracked to try a more methodical approach to assembly, and actually utilized the features I put into the design for easier assembly. These M3 hex nuts slot perfectly into the hex-shaped holes, and allow countersunk screws to be inserted through the bottom of the board.



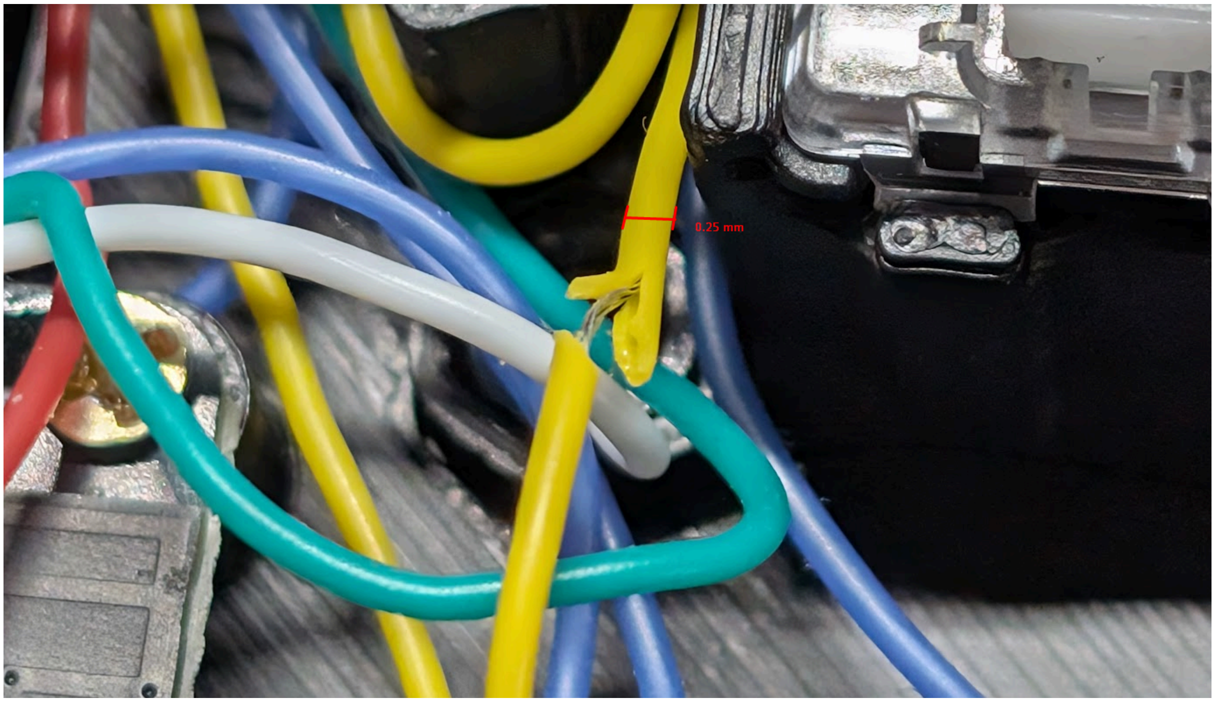
Now wires could be routed much more organized since the switches were mounted to the bottom plate. (this looks unorganized but it's much better than before.) While many of the issues were solved with previous small redesigns, after trying to press down on each display, it was clear that the display modules were placed too close to each other, leading to multiple displays being pressed down when one display is pressed down. Each display was moved away from the center by 0.05in to create clearance for all of the display modules.



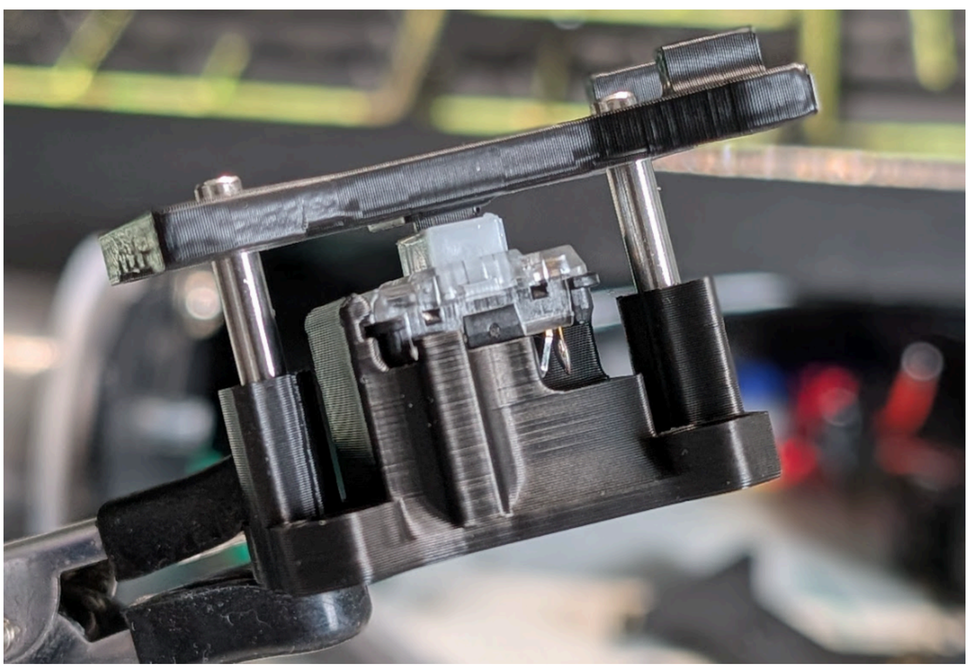
A few additional design changes were made in the days leading up to the SVMA event. Primarily, stabilizers were added to each switch, and the wires were all resoldered at a more optimal angle to avoid breaking connections.



Some of the small connections as seen in the photo above would fray, shorting the ground and voltage and causing many issues. Lots of time was spent fixing these weak or frayed connections.



An example of the small gauge wire being split due to rash movements inside the device. For a sense of scale a measurement has been placed on the photo.



Nearing the completion of the project, I was finally able to work on game logic for the actual tic tac toe portion of it. This was actually pretty straightforward, especially leveraging new AI models to assist with code debugging and optimization.

For the full code and more info on how it works, check out the GitHub Page!