ENGL 273
WAW.MIDI.Controller@gmail.com
WM, AG, WB

MEMORANDUM

To: Kim Lemieux

From: Wes Marsen, Andrew Goertzen, Walker Bradley

CC: Mel Dundas, James Van Oort

Date: Oct 15, 2025

Subject: Progress Report – WAW MIDI Controller

Summary

The WAW MIDI Controller is proceeding according to schedule we are currently under budget and have successfully addressed some initial key challenges: converting key inputs and button presses to MIDI signals, creating slider pad and XY trackpad designs, completing our first Test PCB, and creating a working peak detector circuit for the piezo velocity sensor.

We are still solving some other challenges, such as successfully getting the sensor chips programmed to use the touch surface data and designing the circuitry for our user feedback LEDs. We still must finish designing our final schematic layout for our full design and laying it out into a PCB. However, we are feeling very positive about bringing our design to being a fully functional device by December 12.

Background

Since the release of the first MIDI controllers back in 1983, there has been a standard design that has had very few changes over the years. There has been an increase in variety and popularity over the 42 years, but the most common design hasn't changed much. If we look at a model from the 80s and one from today, there hasn't been much diversity ***(Fig 1 and Fig 2)***. The design has stayed simple, 25-88 keys with a few knobs and sliders for sending control signals, called Continuous Control (CC) signals. Additionally, some units have a joystick, some have drum pads, and some have all the above. The connectors for

Figure 1: Roland's Jupiter 6 (1983). [1]

Figure 2: AKAI's MPK261 (2025). [2]

MIDI controllers have evolved from the original 5-pin DIN MIDI jack, to include TRS 3.5mm jacks, USB connections, and even Bluetooth. Most larger MIDI controllers require their own power source, as well as a connection to the computer or synthesizer. Some limitations that modern units have are the lack of variety and styles.

Our controller, the MTP-25K (MIDI Touch Piano - 25 Keys), will break the mold with a slim design, limiting mechanical inputs in favour of capacitive touch inputs. We will be eliminating the need for a separate power cord by powering our controller with the USB bus-power connection to the computer. Portability will be a key feature, as it will be compact enough to fit in a bag with a laptop, comfortably.

Hardware

A great deal of our progress has been in component selection and testing. We have decided to use integrated circuits (ICs) to control our touch sensors, sliders and XY pads, and LED indicators, which communicate with our main processing unit (MPU) through the Inter-Integrated Circuit (I2C) protocol, discussed in our Firmware section. We have chosen piezoelectric sensors in combination with peak detector circuits to determine note velocity, and low-profile push buttons to control musical features. We also ordered a PCB to test different electrode styles for our touchpads and sliders, which we have been working on.

Teensy 4.1

We have been very happy with the Teensy 4.1 MPU's performance in testing. Its processing speed has been more than adequate in testing our components, though we plan to implement interrupts in the final firmware. This is discussed in more detail in our Firmware section.

Touch "Keys"

Initially, we used prebuilt capacitive touch sensors to test sending MIDI note data to a computer ***(Fig 3)***. For our final product, we will be integrating these directly into our PCB.

Our piano “key bed” will be made up of these built-in touchpads, with piezoelectric sensors mounted to the bottom of the PCB.

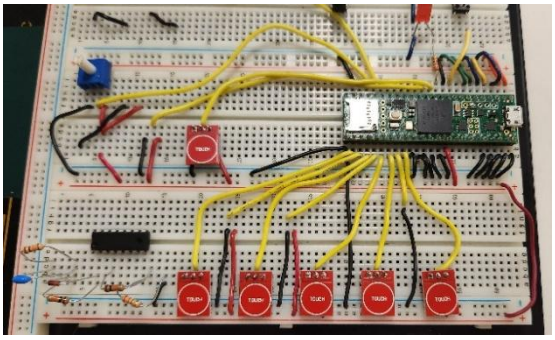

Figure 3: Breadboard with prebuilt touch sensors.

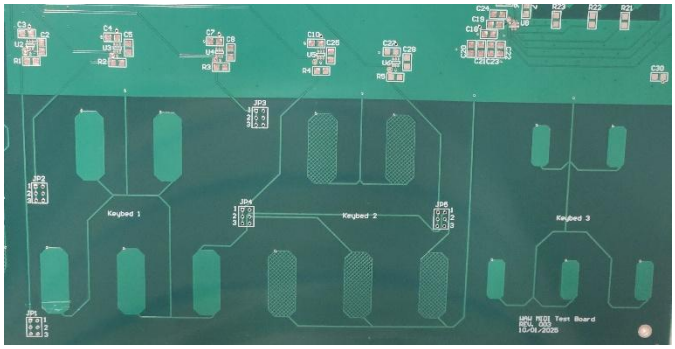

Figure 4: Test PCB with built in touchpads

Touchpads

The touchpads use electrodes, flat pieces of copper pour sitting on the PCB, to detect user input. When the user touches the surface of the touchpad, a small change in capacitance is introduced. These changes are detected by the AT42QT1010 sensor ICs, which generate on/off signals for our MPU to read and process.

Our test PCB arrived in week seven, and we have begun testing electrode styles ***(Fig 4)*** as well as different resistance and capacitance values for the AT42QT1010 circuits, as these affect the sensitivity. We expect to find an acceptable combination in time for the next PCB order in week nine.

GPIO Expanders

Rather than using dedicated general-purpose input/output (GPIO) pins for each of our touchpads, our final board will use two MCP23017 GPIO expanders to read the touchpad states. These will communicate with the MPU through the I2C protocol, reducing the pins used for reading touch pad states from 25 to four.

Piezoelectric Sensors

To detect how hard the user hits a key, we are using piezoelectric sensors attached to the bottom of our PCB. When the sensor vibrates, it generates a voltage waveform proportional to that vibration. We can read that voltage with our MPU’s internal analog to digital converter and use it to set note velocity, changing the volume and timbre of the MIDI note played.

Peak Detectors

Due to the varying frequency and amplitudes of the piezoelectric sensor waveforms, we had trouble consistently reading the maximum values reached in initial testing. To fix this problem, we added peak detecting circuitry. These circuits use capacitors to hold the maximum voltage level reached by the sensor, giving us much more reliable results. We used Digikey's article on peak detection [3] as a reference in designing this circuit. We tested basic functionality with LM324N op amps from our parts kit but will be implementing faster TL074HIPWR op amps on the final board.

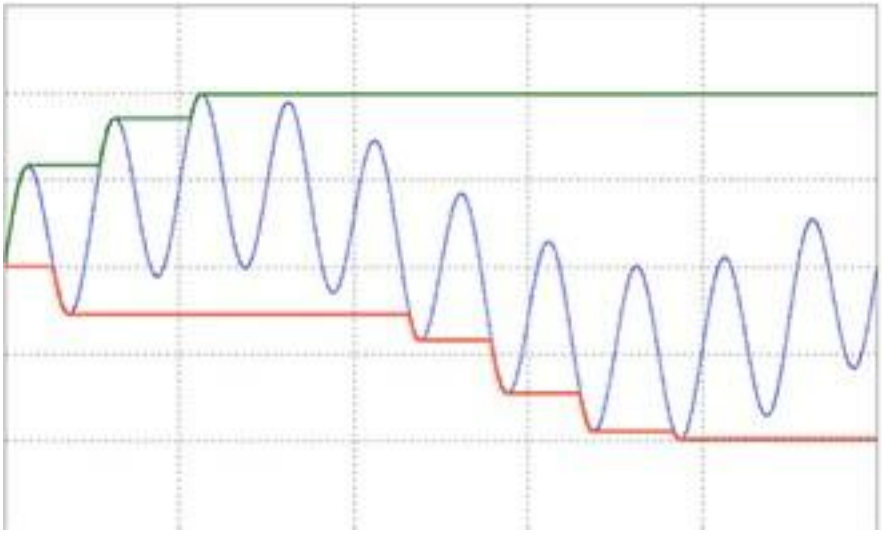

Figure 5: Example of a Peak Detector Waveform. Ours would be the green. [1]

Touch Modulators

Instead of mechanical knobs and sliders, the MTP-25K will use capacitive touch for sliders and an XY pad for modulation. We are using two IQS7211A ICs to control these: one set up to control the four sliders ***(Fig 6)***, and one set up for XY trackpad ***(Fig 7)***. These ICs track the user's finger as it moves across the modulation inputs and communicate with the MPU over I2C.

We have begun testing these on our test PCB using the CT-210A configuration tool. This tool pairs with software to show user input and fine tune sensitivity. We expect to finish fine tuning these by the end of week 9 to allow time to fully integrate them in software before the final PCB's arrival in week 11.

Figure 6: Two Slider Styles on our Test PCB.

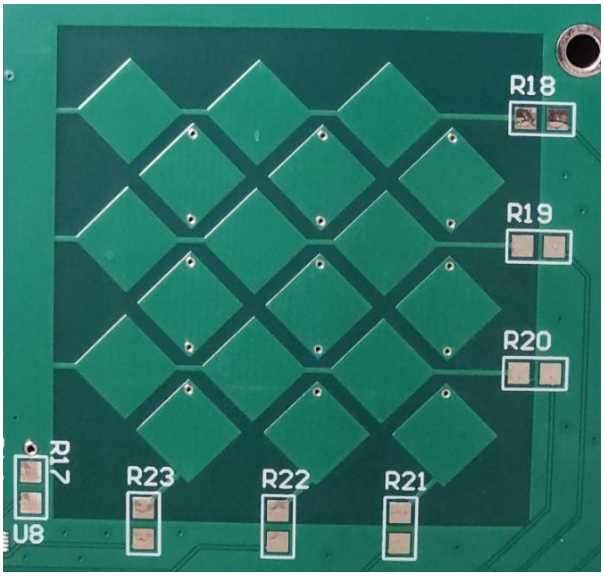

Figure 7: XY Trackpad on Test PCB.

Mechanical Push Buttons

The MTP-25K will be equipped with five push buttons. Four will control musical features (octave up, octave down, hold, and arpeggio) and the last will act as a Shift key. Holding the Shift key will change the behaviour of the buttons and sliders, allowing the user to change feature modes and arpeggio tempo. We have tested basic functionality on a breadboard and are working on implementing these on our final PCB.

PCB

Our test PCB arrived on time in week seven, which can be seen in **APPENDIX B – *Fig 9***. We initially ran into some trouble with the touchpad response; however, after some modifications, we are now seeing note signals as expected. We are currently finalizing our decision on the electrode style in preparation for our final PCB order in week nine. There is a diagram of the Schematic Drawing for our test PCB in **APPENDIX A – *Fig 8***.

LED Indicators

The MTP-25K will use LED indicator lights as visual feedback for the user. To drive these, we will be using an IS31FL3731 IC. This is compatible with up to 144 LEDs through Charlieplexing, an advanced form of multiplexing that allows for individual control of LEDs

within a large array with limited I/O pins. The large array can be seen in **APPENDIX C – Fig 10**. Instructables has a great entry-level description of Charlieplexing[4].

The IS31FL3731 is designed to drive LED arrays, we will be purchasing one to use for testing. In our final design, these LEDs will be more spread out, which will be handled in firmware. We are working on integrating these indicators into our final PCB and expect to complete this in time for the next PCB order in week nine.

Firmware

To test our hardware components, we have written small test programs for our touch sensors, feature buttons, and piezo sensors, sending commands to MIDI-OX, a free MIDI utility software on Windows. Now that we have our test PCB, we are working on communicating with our XY pad and sliders through I2C communication. Our next big steps are integrating interrupts, the arpeggio function and shift key, and the LED indicators.

Inter-Integrated Circuit (I2C)

The MTP-25K relies on the I2C communication protocol to manage all our input and output components. Rather than rely on the limited GPIO pins on the MPU, we can outsource much of this to dedicated ICs. SparkFun's article explaining I2C [5] is a great explanation, but it is useful to think of it as a video conference call, like Zoom.

On our first call (bus), the Teensy board is the host (master), and the IQS7211A and IS31FL3731 chips are participants (slaves), each with a unique name (address). Let's call them Bob and Charlie. Bob and Charlie have the microphones muted most of the time and only unmute when the host calls out their name. They can signal that they have something to say by raising their hand.

In our case, we have another participant we want to talk to, a second IQS7211A, whose name also happens to be Bob. Both Bobs are stubborn and refuse to respond to anything else. Fortunately, the Teensy board has three separate buses to communicate on, allowing a second dedicated call for the other Bob. We are using our third bus to talk to our touchpad GPIO expanders, the MCP23017s. Unlike the Bobs, these ICs have settable addresses, or nicknames. We'll call these ICs Tom and Tommy.

Interrupts

We built our test programs with a polling approach, checking each input every time we move through our main loop. This was effective at a small scale, but uses a lot of power, and is likely to result in additional input lag as we connect more components.

Interrupts allow us to trigger code based on those changes. When Tommy detects changes to his touchpads he raises his hand, and the Teensy board immediately reads the change and processes the data. This is more complicated to implement, but the improvements will be critical in ensuring we stay within our power budget and response time goals.

Feature Buttons

To date, we have implemented the octave and hold buttons in firmware. Next, we need to work on the arpeggio function and the shift key. We will start working on this in week nine in preparation for our final PCB's arrival in week 11.

LED Indicators

The IS31FL3731 Charlieplexing IC comes equipped with a powerful code library capable of adjusting each LED independently. We intend to purchase an LED array for testing purposes and will be working on this alongside the final push button functions in week nine.

Finances

We set a personal budget of $500 and to date we have spent $305.40. We still need to order the Charlieplexor, GPOI expanders, TRS Jack, LEDs, and additional resistors and capacitors. We are hopeful that we will finish the project and be under the $500 budget we have set.

Conclusion

Over the first eight weeks, we have made a lot of advancements for the MTP-25K. We have calibrated touch pad sensors, created a test PCB with each touch component on it, have MIDI signals being sent from touch pads and completed the octave change buttons and the hold feature. The tasks that remain are debugging the PCB which will allow us to finish the final PCB design, build the LED array and get the Charlieplexing working, getting the chips to control the sliders/ XY trackpad, the interrupts hierarchy, finishing the arpeggiator and the tempo indicator. We have been able to keep up with our self-made schedule, seen in

the Gantt Chart in **APPENDIX D – *Fig 11***, and are confident that we will maintain our pace and meet the Capstone symposium with a finished, consumer ready, product.

References

[1] SynthArk, "Roland Jupiter-6," [Online]. Available: https://www.synthark.org/Roland/Jupiter-6.html.

[2] AKAI, "MPK261," [Online]. Available: https://www.akaipro.com/mpk261.

[3] DigiKey, "Hardware vs. Software Tradeoffs – Part 2: Peak Detectors," 12 10 2017. [Online]. Available: https://www.digikey.ca/en/articles/hardware-vs-software-tradeoffs-part-2-peak-detectors.

[4] AUTODESK, "Charlieplexing Made Easy (and What It Even Means?!)," [Online]. Available: https://www.instructables.com/Charlieplexing-Made-Easy-and-What-It-Even-Means/.

[5] sparkfun, "I2C," [Online]. Available: https://learn.sparkfun.com/tutorials/i2c/all.

Appendix A: Schematic Drawing of test PCB

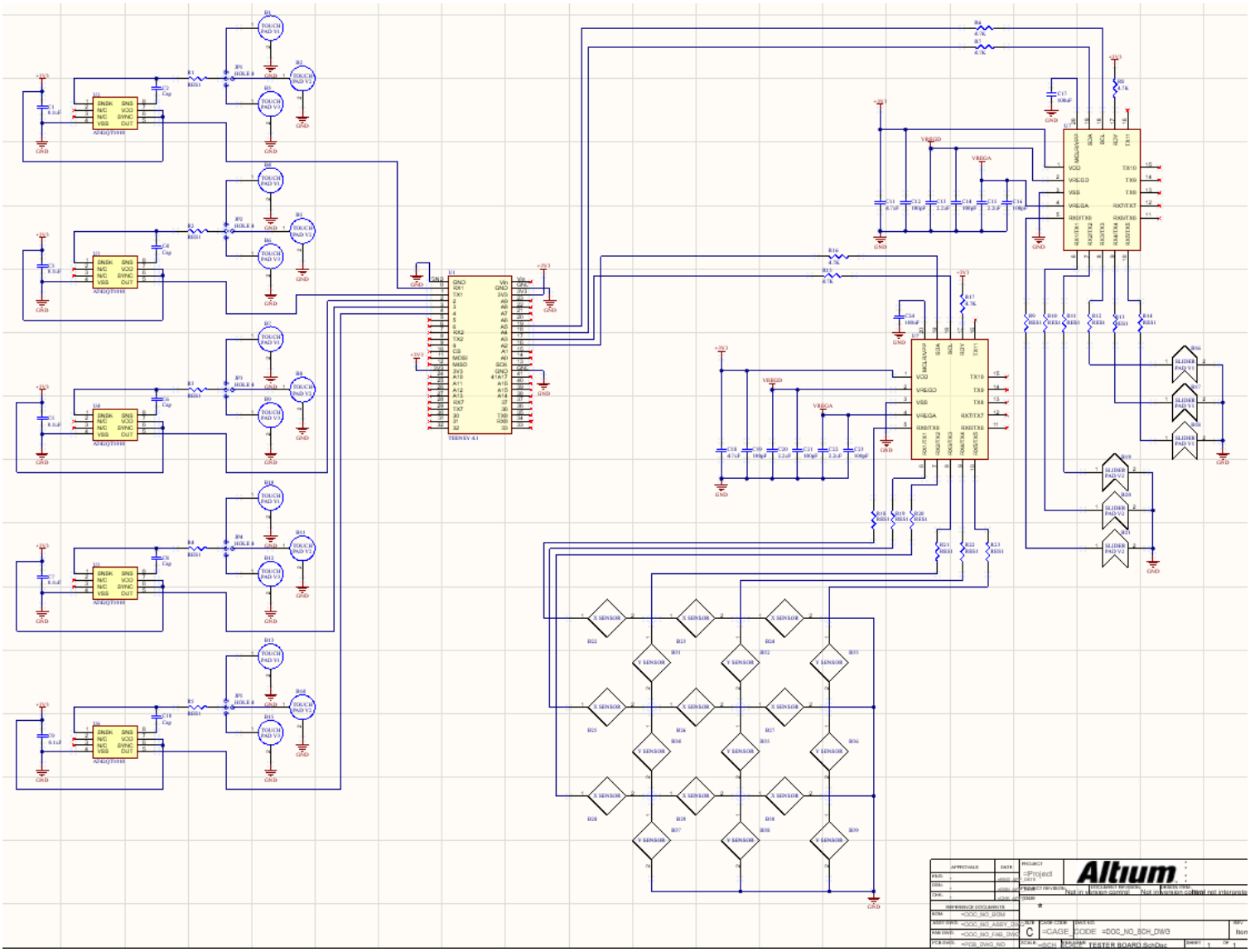

Figure 8: Schematic Diagram of the Test PCB.

Appendix B: Test PCB Board

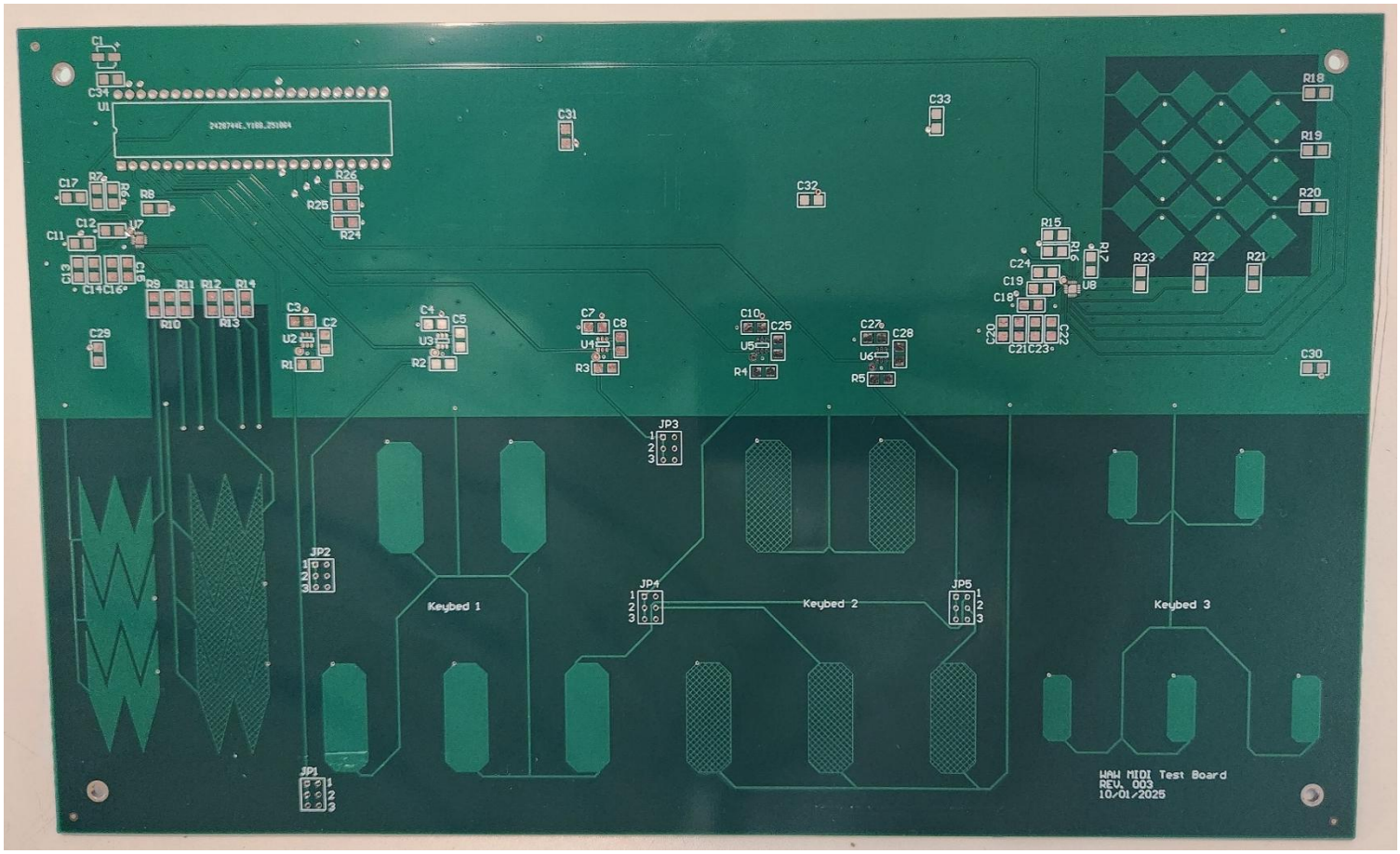

Figure 9: Test PCB with no parts.

Appendix C: Charlieplexing LED Array.

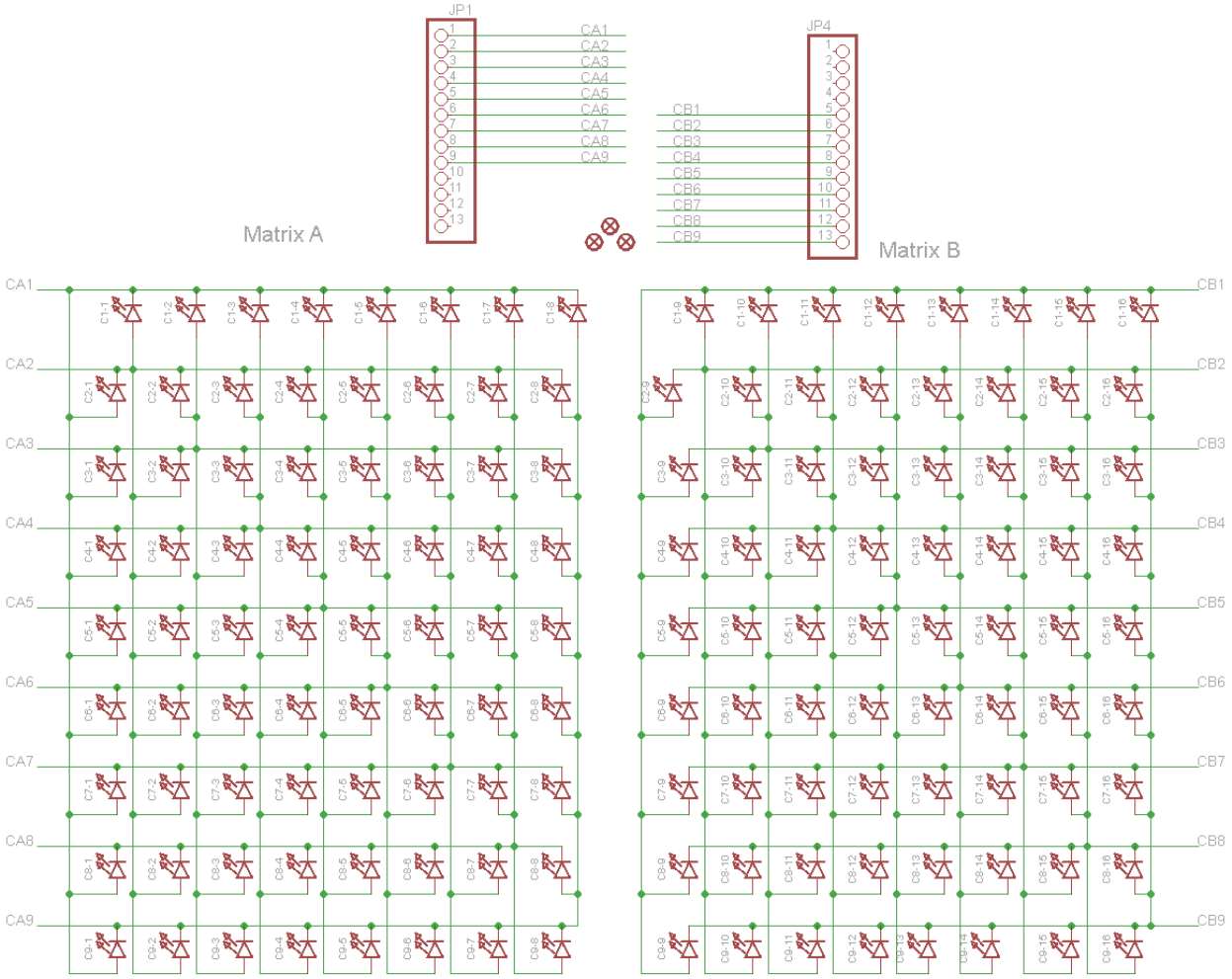

Figure 10: LED array for the Charlieplexing Module.

Example: If we want to turn on LED C1-1, we must send a high signal on CA2 and a low signal on CA1.

Appendix D: Gantt Chart Timeline

Task Name	Duration	Start	Finish	Predecessors
Concept Work	**31 days?**	**Tue 25-09-02**	**Tue 25-10-14**	
Assign Tasks and Roles	1 day	Tue 25-09-02	Tue 25-09-02	
Rough draft for the enclosure	20 days	Mon 25-08-18	Fri 25-09-12	
Coding to get button press as MIDI data	7 days	Mon 25-09-08	Tue 25-09-16	
create TESTER schematic & PCB in altium	7 days	Thu 25-09-18	Fri 25-09-26	3
Create Final PCB design	12 days	Mon 25-09-29	Tue 25-10-14	5
Correct TESTER PCB design	2 days	Mon 25-09-29	Tue 25-09-30	5
Getting the OK and sending off TESTER BOARD	1 day	Wed 25-10-01	Wed 25-10-01	7
Building Controller	**38 days**	**Wed 25-10-08**	**Fri 25-11-28**	**1**
Agree on Sensor/ Slider/ XY design	2 days	Wed 25-10-15	Thu 25-10-16	8
Finish Coding for the board	4 days	Fri 25-10-17	Wed 25-10-22	10
Finish Final PCB design	11 days	Fri 25-10-17	Fri 25-10-31	10
Correct FINAL PCB design	2 days	Mon 25-11-03	Tue 25-11-04	12
Order FINAL PCB	1 day	Wed 25-10-29	Wed 25-10-29	
get the Final PCB!	1 day	Mon 25-11-03	Mon 25-11-03	12
Run Tests to break code	2 days	Tue 25-11-25	Wed 25-11-26	15
Test to break hardware	1 day	Tue 25-11-04	Tue 25-11-04	15
Create enclosure	7 days	Mon 25-11-03	Tue 25-11-11	3,12
Finalize enclosure	4 days	Wed 25-11-12	Mon 25-11-17	15,18
Final design Built	1 day	Mon 25-11-24	Mon 25-11-24	19,15
Finishing Touches	**15 days?**	**Mon 25-12-01**	**Fri 25-12-19**	**9**
Celebrate with a dinner!	1 day	Fri 25-12-05	Fri 25-12-05	20
Capstone Symposium	**1 day?**	**Fri 25-12-12**	**Fri 25-12-12**	**20**
	1 day	Fri 25-12-12	Fri 25-12-12	
Documentation	**68 days?**	**Wed 25-09-03**	**Fri 25-12-05**	
Work on the User manual	19 days	Mon 25-09-15	Thu 25-10-09	
Finish User Manual	1 day	Thu 25-10-09	Thu 25-10-09	
Work on Website	45 days	Mon 25-10-06	Fri 25-12-05	
Finish Website	1 day	Fri 25-12-05	Fri 25-12-05	

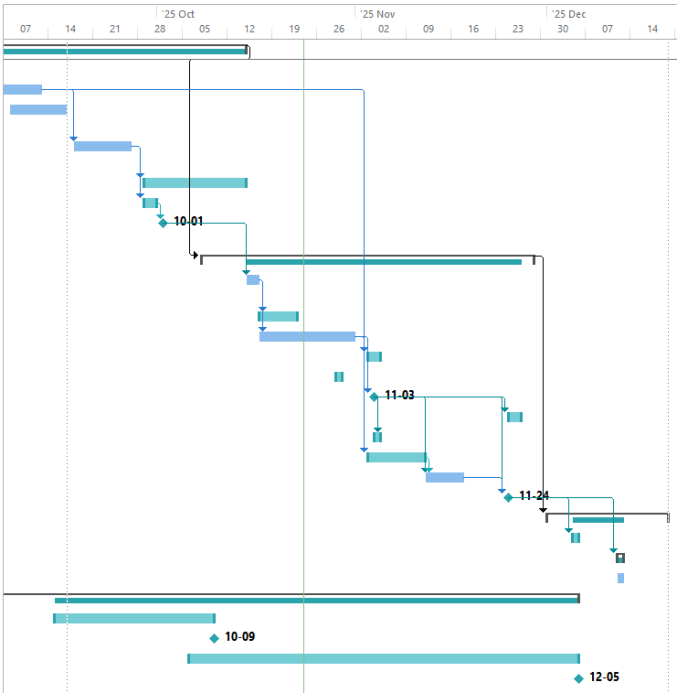

Figure 11: Gantt Chart.