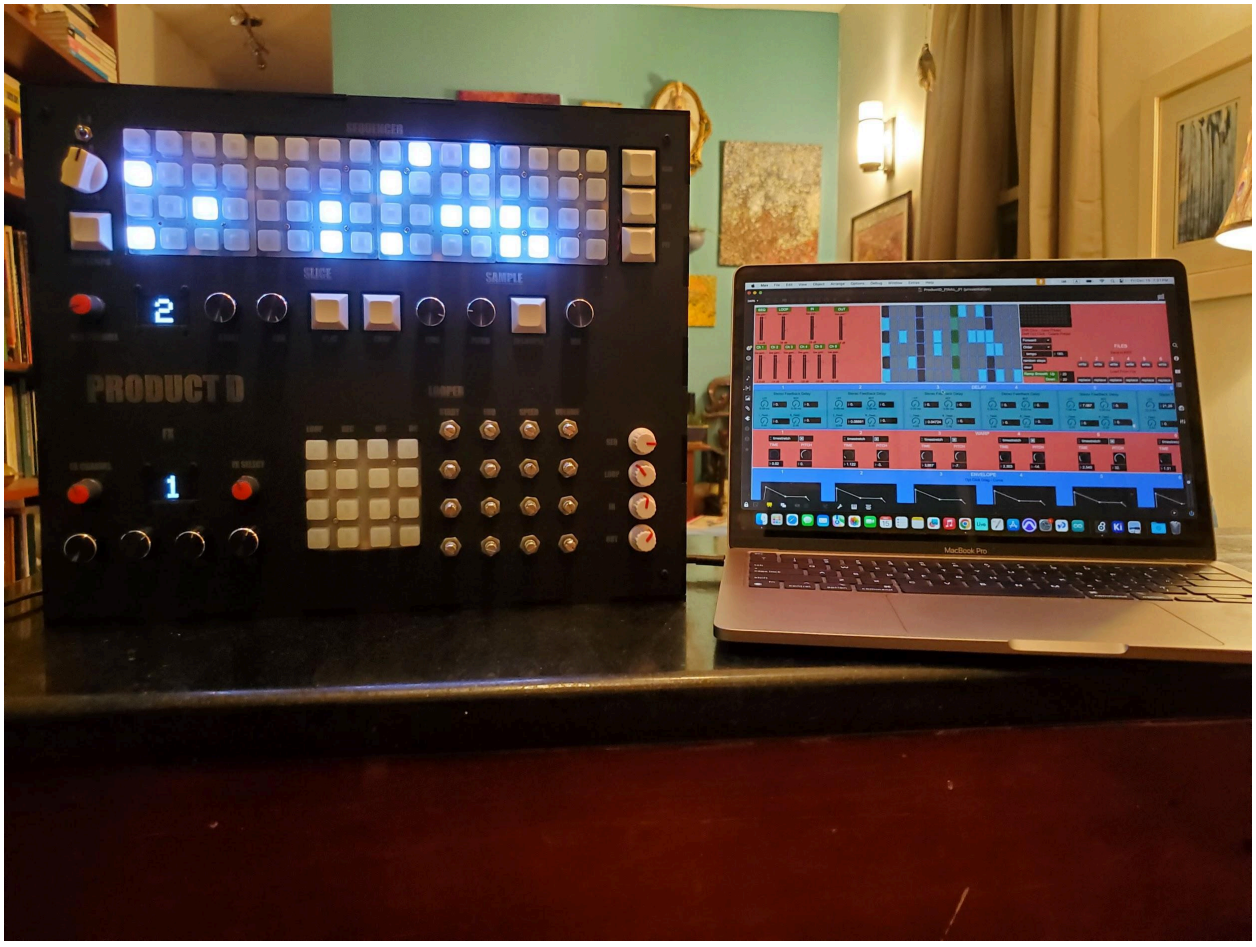
Product D

Paul Franklin

12/14/2023



Introduction

Welcome to the Product D README. This document provides an in-depth overview of the Product D, a custom hardware controller meticulously designed to complement and control a Max patch (Cycling 74/Max4Live). The Product D features a 6-channel live sampler and drum sequencer, alongside 4 channels of live looping, offering a robust suite of functionalities for a dynamic and intuitive musical experience.

The Product D was born from a vision to create a hardware interface that seamlessly integrates with its software counterpart, aiming to enhance the creative process for

musicians and producers. This README will guide you through the features, design choices, and technical aspects of the Product D, providing valuable insights into its capabilities and usage.

Background

The development of the Product D was anchored in extensive research, primarily focusing on technical and process-oriented aspects rather than adhering to industry prototype standards. This approach was driven by the unique needs and goals of the project, which often diverged from conventional industry practices.

Throughout the journey, I engaged in discussions with experienced engineers and leveraged resources from academic courses, notably receiving valuable insights from Steven Litt. My research encompassed a wide range of topics, including electrical engineering theory for PCB design, and I actively participated in code forums and idea exchange platforms.

A significant aspect of the development process involved utilizing AI tools. These tools were instrumental in generating troubleshooting guides and brainstorming innovative solutions, thus playing a crucial role in overcoming challenges and refining the design of the Product D.

Video Demonstration

Given the extensive features of both the Product D and the accompanying Max patch, I've prepared a comprehensive video demonstration. This video showcases key functionalities and highlights how these elements work together in a practical context.

Product D features explanation

Written Explanation:

GRID Interface

The GRID is an interface for sequencing drums in a Max patch. Key features include:

- Matrix Buttons: Utilizes a matrix of gummi buttons over Adafruit NeoTrellis grids, each with built-in LEDs. Pressing a button toggles its light and state.
- Grid Synchronization: Aligns with a 6x16 Max grid. Our hardware grid (4x16) can switch between the top 4 rows and bottom 2 rows using a Toggle Switch, retaining sequence memory.
- Tempo Control: Adjust sequence playback speed with a Tempo Knob and a Tap Tempo Button.
- Additional Controls:
    - Random Step Button: Activates a random number of steps (0-7) on each row.
    - Clear Button: Clears the entire grid.
    - Play Button: Starts/stops sequence playback from the last position.

CHANNEL Control Strip

The CHANNEL section serves as a master control for each grid row, featuring space-efficient controls common to all channels:

- Channel Selection Encoder: Selects the active channel. The starting position is non-critical, avoiding value jumps.
- Channel Number Display: An OLED display showing the currently selected channel.
- Start/End Encoders: Adjusts the start and end points of a sample. Nearby controls include:
    - Sample Play Button: Plays the selected sample segment.
    - Crop Button: Crops the sample to the selected segment.

- Sample Manipulation Encoders:
    - Time Encoder: Adjusts sample duration.
    - Pitch Encoder: Alters the pitch of the sample.
- Decouple Button: Separates or links the Time control's effect on pitch, emulating tape machine behavior.
- Volume Control: An encoder adjusting the sample's playback volume.

FX Section

The FX section is designed for effect management across sequencer and looper channels:

- Channel Selection Encoder: Switches between the 6 sequencer channels and, upon pressing, toggles through the 4 looper channels.

- Channel Number Display: An OLED display indicating the currently selected channel.
- FX Selection Encoder: Navigates between delay settings (applicable to all 6 sequencer and 4 looper channels) and ADSR (specific to the 6 sequencer channels).
- Parameter Control Encoders: A set of 4 encoders beneath the FX section, assignable to various parameters of the currently loaded effect.

LOOPER Section

The LOOPER section provides intuitive controls for managing loop channels:

- NeoTrellis Matrix Controls: Each of the 4 loop channels has a dedicated row with four buttons:
    - Loop Button: Engages looping for the channel.
    - Record Button: Starts/stops recording for the loop.
    - Off Button: Disables the loop channel.
    - On Button: Enables the loop channel.
- Channel-Specific Potentiometers: Four potentiometers per channel, each controlling a distinct parameter:
    - Start: Adjusts the loop's starting point.
    - End: Sets the loop's ending point.
    - Speed: Alters the playback speed of the loop.
    - Volume: Controls the loop's volume.

MIXER Section

The MIXER section is a straightforward interface for volume control:

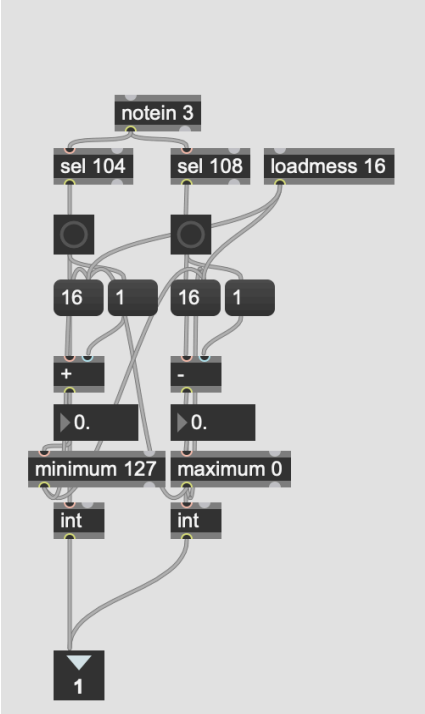
- Volume Control Potentiometers: An array of 4 potentiometers, each dedicated to a specific volume control:
    - Sequence: Adjusts the overall volume of the sequencer.
    - Looper: Controls the collective volume of all looper channels.
    - Input: Manages the volume of the external device used for recording to the Max patch.
    - Output: Functions as the master volume knob for the entire setup.

Problems and Solutions

This section highlights key challenges encountered during the project's development, particularly in areas of communication, and the innovative solutions that were implemented.

Communicating to Max

- MIDI Message Utilization:
    - *Note Messages:* Utilized for button interactions, sending "note on" messages to trigger actions in Max. Different MIDI channels are used to change channels without altering the note.
    - *Continuous Control (CC) Messages:* Employed for smoother incremental controls like potentiometers, with MIDI channel variation for cross-channel uniformity.
- Encoders Handling:
    - Implemented via the encoder library, where each click sends distinct MIDI notes. In Max, sequences increment or decrement values based on the received note.
    - *Example Configuration:* Usage of “notein” object for MIDI channel selection and “sel” for specific MIDI note detection (e.g., notes 104 or 108).



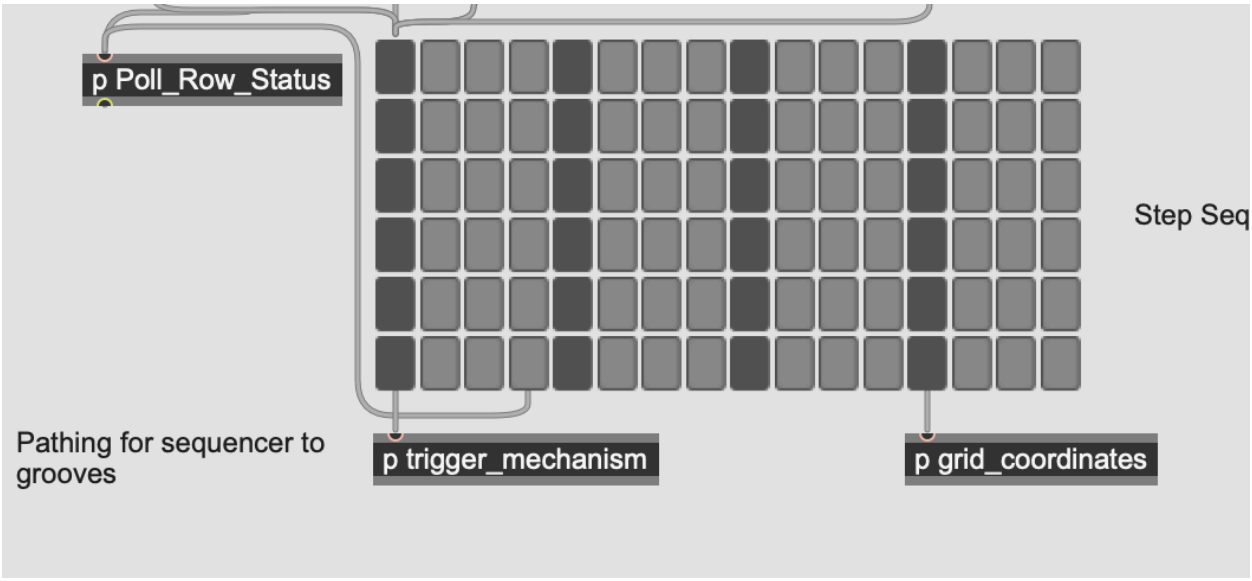
Communicating Between Teensy1 and Teensy2

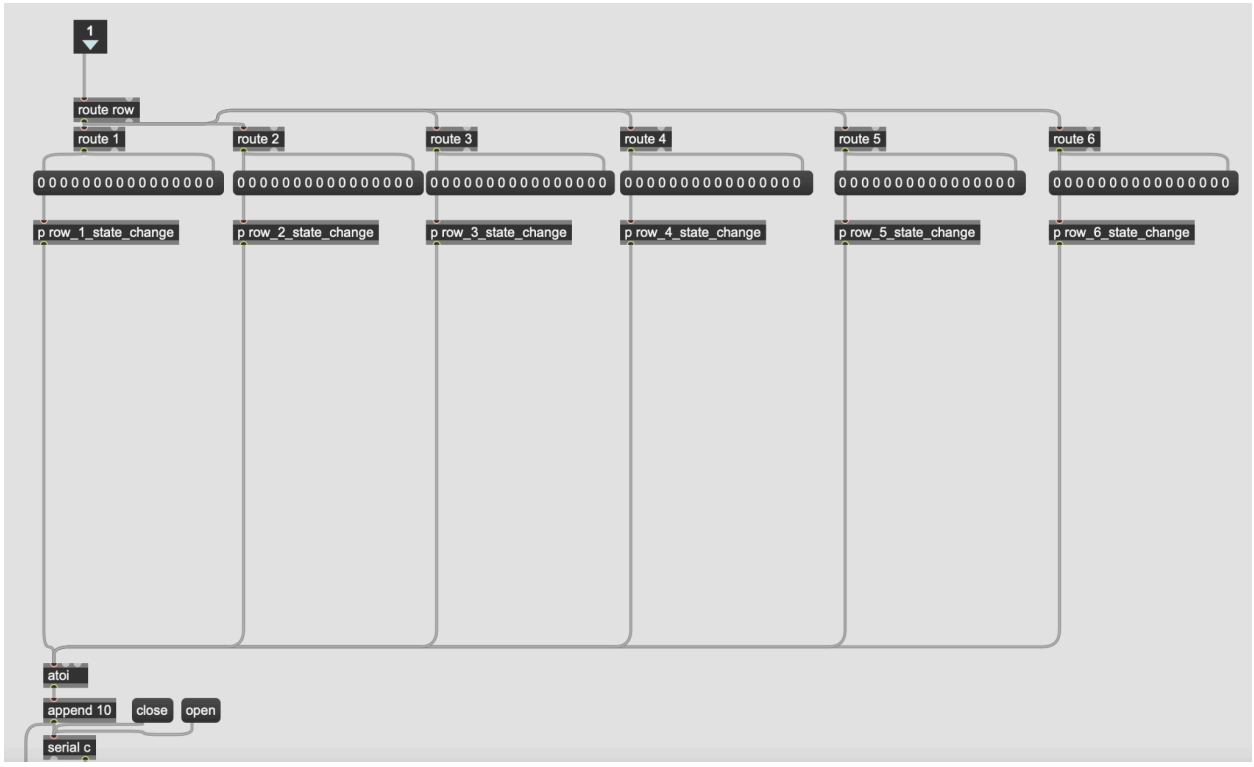
- Challenge: Insufficient analog pins on Teensy 4.1 for all required potentiometers.
- Solution: Utilized Serial communication between two Teensy microcontrollers.
    - *Implementation:* One Teensy reads potentiometer values and sends them to the other Teensy via Serial communication (TX to RX pins).
    - *Communication Protocol:* A specific number is sent first to signal incoming data, followed by the quantity of data points to expect.
    - *Efficiency Tip:* To reduce processing load, values are only transmitted when there is a significant change (e.g., absolute value change > 2), preventing Teensy bottlenecking.
- Outcome: Successfully expanded the number of usable potentiometers without additional menu complexity, with Teensy2 dedicated to potentiometer control.

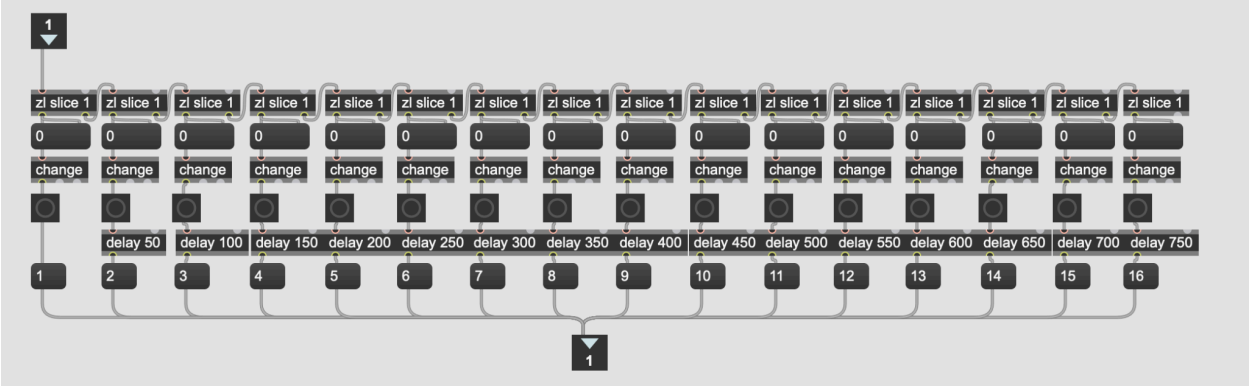
Communicating from Max

- Challenge: Establishing effective and efficient communication between Max and the hardware, with initial attempts leading to significant lag and system strain.
- Initial Approach:
    - Developed a communication protocol to send data from Max to the Teensy.
    - Utilized ASCII encoding for the transmitted data and added delays to prevent overloading the serial monitor.
    - Attempted optimization by converting ASCII to binary within Max, reducing Teensy's processing load.
- Issues Encountered:
    - Despite efforts, the latency and system lag rendered the solution non-functional.

      Included images illustrate the data handling process from grid variables to serial output.


p Poll_Row_Status
Step Seq
Pathing for sequencer to grooves
p trigger_mechanism
p grid_coordinates


1
route row
route 1
route 2
route 3
route 4
route 5
route 6
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
p row_1_state_change
p row_2_state_change
p row_3_state_change
p row_4_state_change
p row_5_state_change
p row_6_state_change
atoi
append 10
close
open
serial c



Final Solution:

- Reconsidered the necessity of direct Max-to-hardware communication for all functions.
- Implemented an alternative approach for the grid's random step function:
    - Generated random lists and sent MIDI values to the live grid for step updates.
    - Updated the LEDs directly within the Teensy, bypassing the need for extensive communication from Max.
- This approach preserved functionality while maintaining user-friendly operation and reducing system strain.

One Final Problem: Handling I2C Bus with NeoTrellis Matrices

- Challenge: Managing a single I2C bus for all 5 NeoTrellis matrices, leading to complications due to their linear addressing (0 to 79) and physical layout.
- Specific Issues:
    - Mismatch between the NeoTrellis matrix size and the Live.Grid GUI object in Max.
    - The need to accommodate for 6 rows in Max, with plans to expand to 8, despite the 4x4 grid limitation of NeoTrellis.
- Implemented Solution:
    - Created a boolean 1D array for channels 5-8, allowing for toggling between different channel arrays.
    - Integrated this toggle system into various functionalities:
        - The random list generator, which produces 6 rows of random numbers, now accounts for the channel toggle.
        - Modified the blink function (based on the NeoTrellis demo) to handle exclusions and toggle states, determining which array to update.
- Outcome:

- Successfully integrated the toggle system, though it increased the complexity and interdependence of the code, making problem isolation more challenging.
- The solution is functional and accommodates the unique layout and control needs of the project.

Final Thoughts

Reflections on the Project

- Simplicity Beyond Initial Challenges: Beyond the initial complex problems discussed, other aspects of the project, including coding the less intricate parts and building the enclosure, were relatively straightforward.
- Resources and Community Support:
  - Found value in review videos and guides for general guidance.
  - Leveraged online forums for addressing common concerns and complex issues, employing skills in research and reverse engineering.

Role of AI in the Project

- AI as a Problem-Solving Tool:
  - As the project's challenges became more unique and specific, AI tools played a significant role.
  - AI was especially useful for breaking down complex ideas, suggesting new approaches, and providing step-by-step guides for testing and debugging.
- Cautions on AI's Limitations:
  - It's advisable to use AI as a guide and for idea generation rather than for generating complex code directly, due to its limitations in handling highly specialized tasks.

Future Work and Applications

Planned Enhancements:

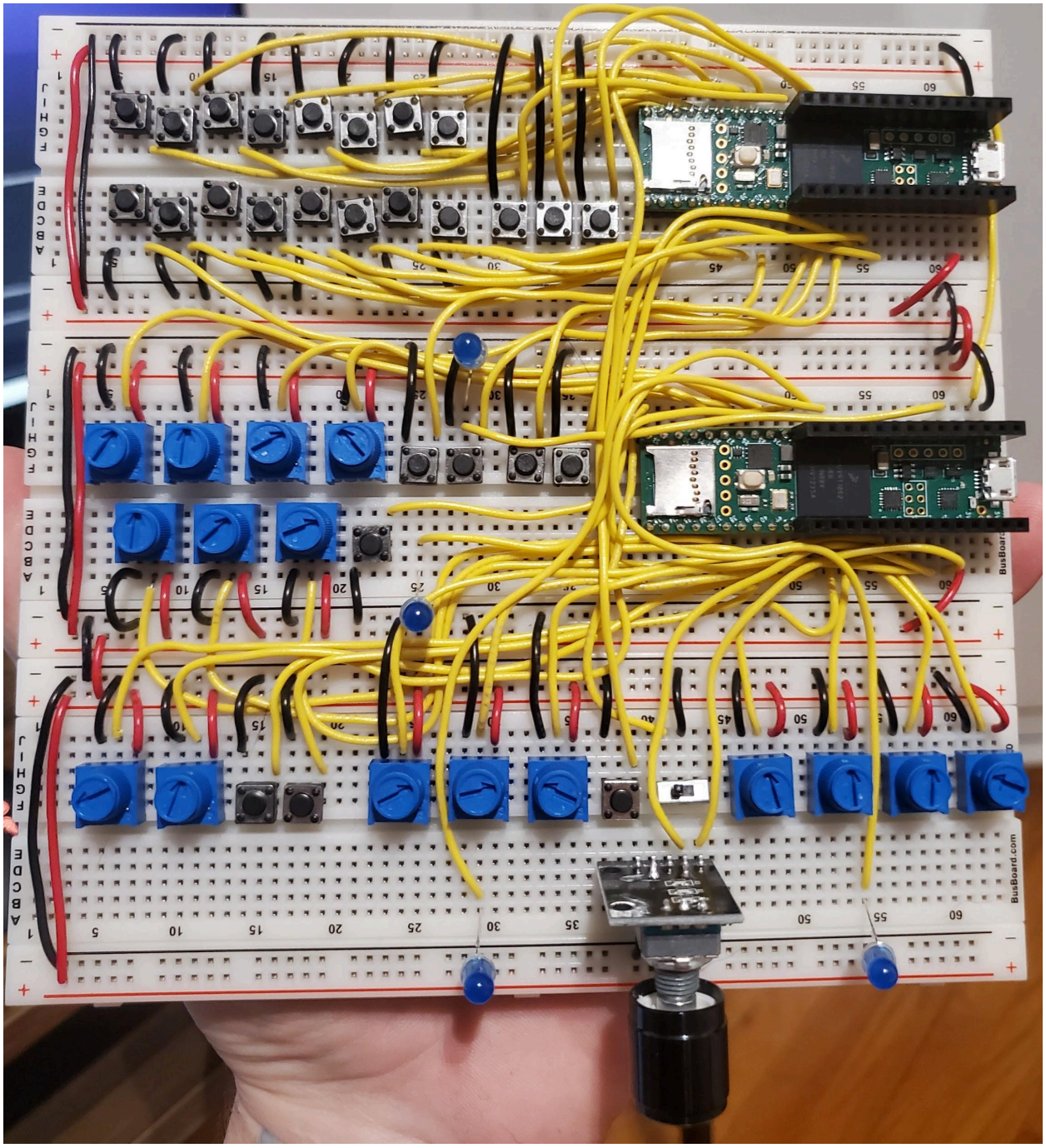
- Channel Expansion: Aiming to increase the channel capacity from 6 to 8, aligning better with the hardware's capabilities.

- Additional Effects: Considering the addition of more effects to enhance creative flexibility. The design allows for easy integration of new effects via software updates.
- Visual Sync Feature: Planning to implement a synchronized visual representation of the active step in the LED grid with the live grid in Max, enhancing the user experience with an engaging visual effect.

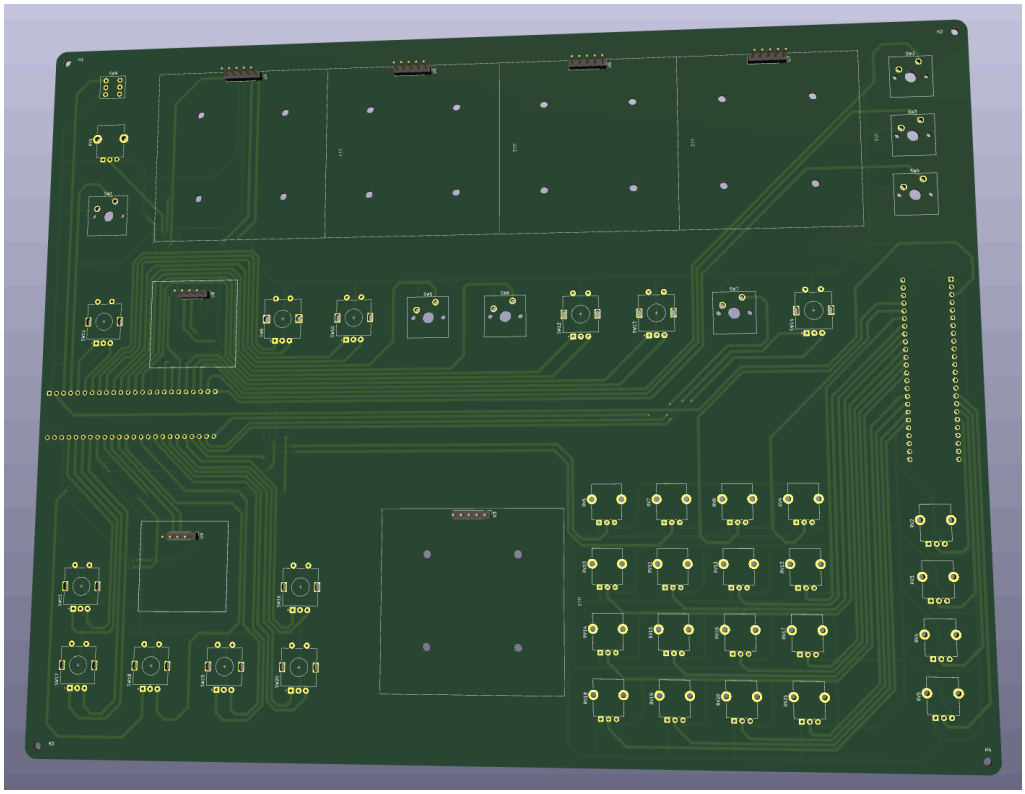
Applications:

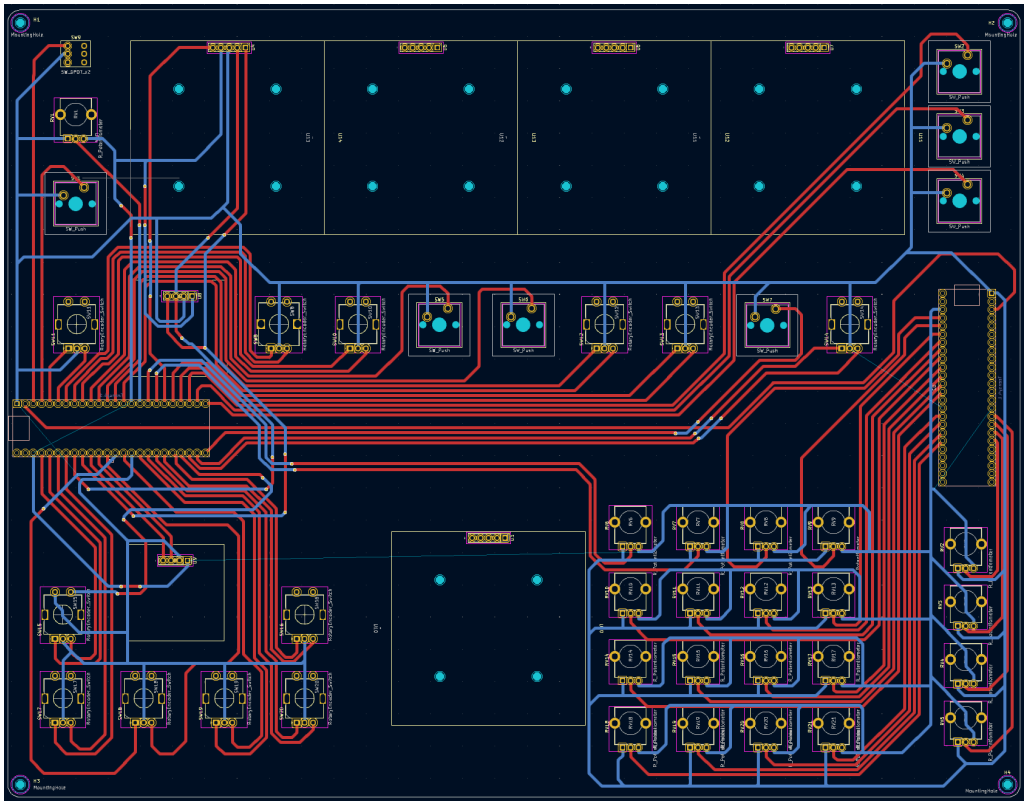
- Personal Use in Music: The device will be a key element in my personal music compositions and live performances.
- Portfolio Showcase: This project will be prominently featured in my portfolio to demonstrate my design process and problem-solving skills.

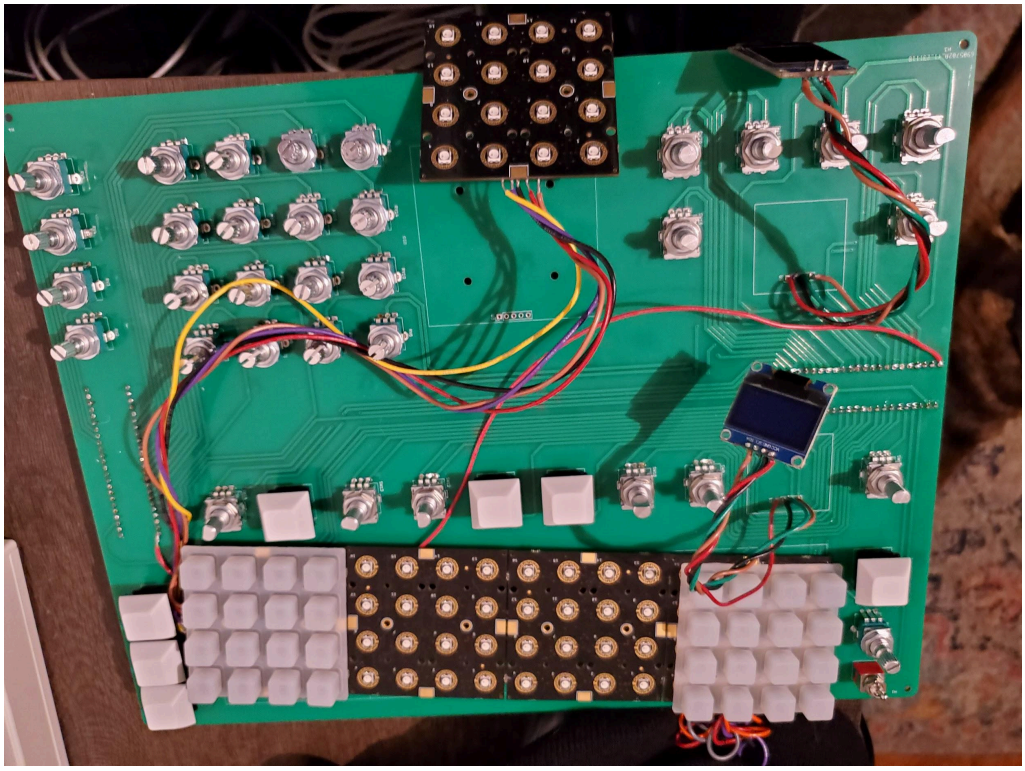
Gallery:



Breadboard Demo



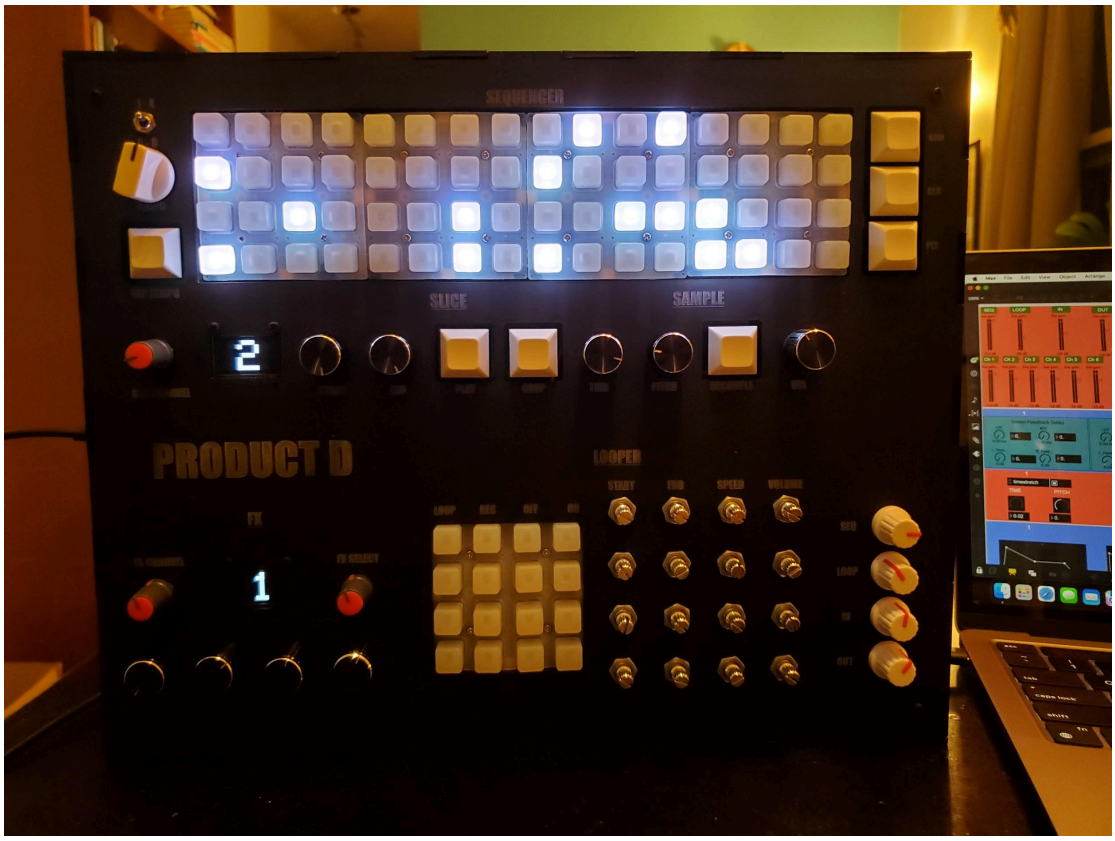


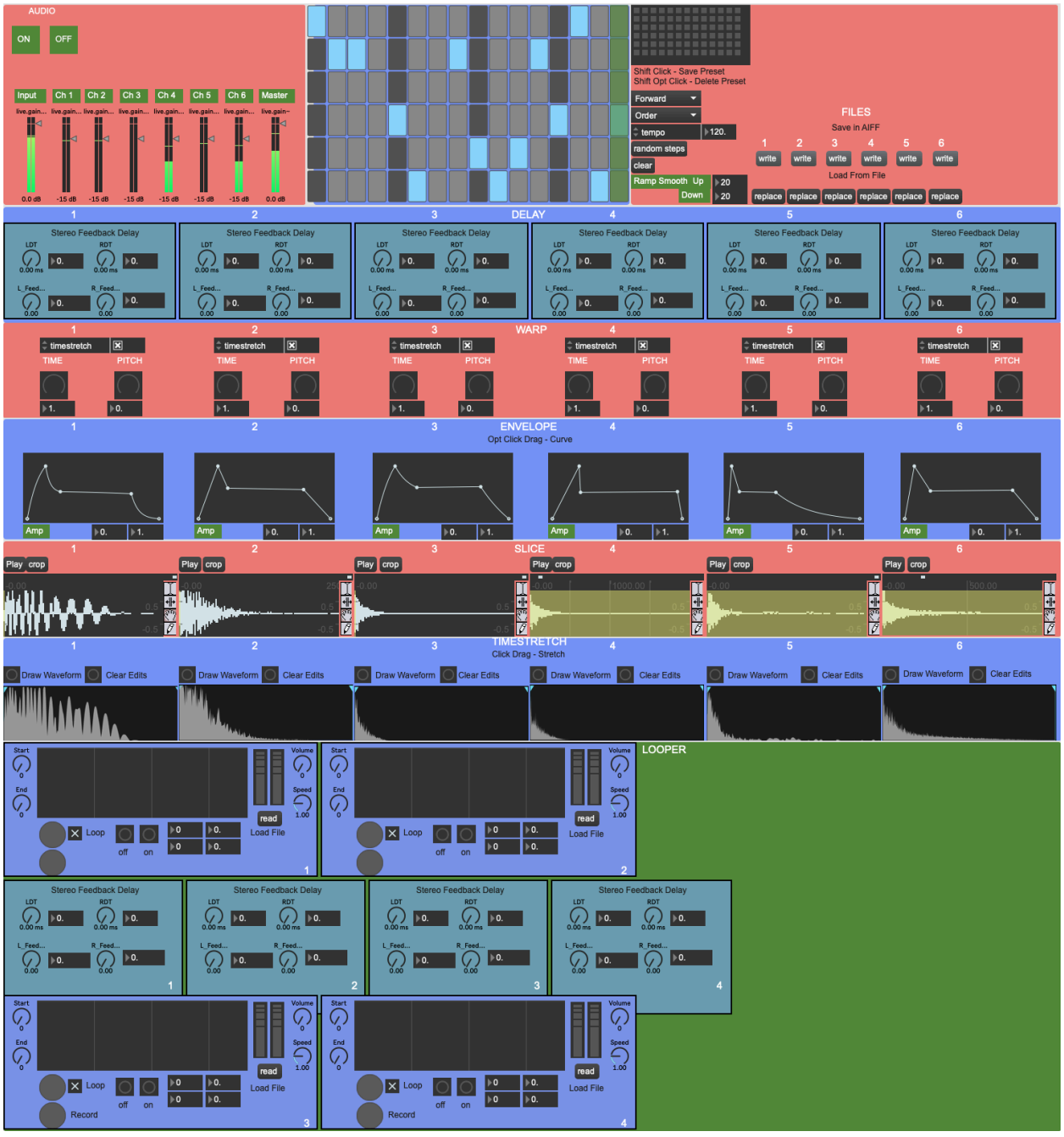


PCB

Enclosure


SEQUENCER
SLICE
SAMPLE
PRODUCT D
LOOPER
FX


SEQUENCER
SLICE
SAMPLE
PRODUCT D
LOOPER
FX


AUDIO
ON
OFF
Input
Ch 1
Ch 2
Ch 3
Ch 4
Ch 5
Ch 6
Master
Shift Click - Save Preset
Shift Opt Click - Delete Preset
Forward
Order
tempo
120.
random steps
clear
Ramp Smooth Up
Down
FILES
Save in AIFF
write
Load From File
replace
DELAY
Stereo Feedback Delay
WARP
timestretch
TIME
PITCH
ENVELOPE
Opt Click Drag - Curve
Amp
SLICE
Play
crop
TIMESTRETCH
Click Drag - Stretch
Draw Waveform
Clear Edits
LOOPER
Start
End
Volume
Speed
read
Load File
Loop
off
on
Record

Max Patch