EMID Fall 2024 - MIDI Air Hockey Project Final Report

Report by Karen Bei

Group: Drew Cohen, Andy Navarro, and Isaac Monheit

Introduction:

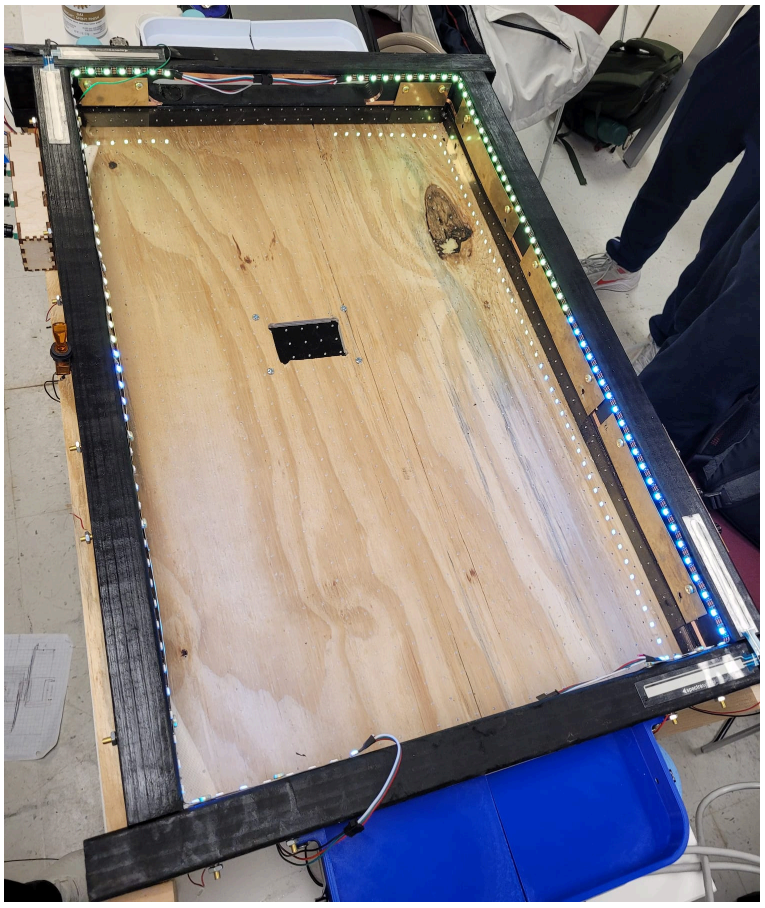

Figure 1: Completed MIDI Air Hockey project

The MIDI Air Hockey project (Figure 1) offers an innovative way to blend interactive gameplay with live music creation, turning a classic air hockey game into a dynamic musical experience. In contrast to traditional electronic music instruments, this project combines competition and creativity, enabling players to “play” music while they compete. Through the use of sensors, wireless communication, and real-time MIDI control, MIDI Air Hockey converts every collision, score, and puck movement into a musical event, allowing players to influence the game’s soundtrack through their performance. From the first iteration, we have ensured a more robust mechanical design, upgraded functionalities of the electrical components, along with elevating the capabilities of our musical design.

The goal of this project was to design and build an electronic instrument that could capture both the thrill of competition and the evolving nature of music. We sought to create an experience where two players not only competed in a physical game of skill but also worked together in real-time to craft a one-of-a-kind musical composition.

To accomplish this, our team designed and constructed a custom air hockey table equipped with sensors that track wall collisions and goals. Controls are placed along the sides of the table, allowing users to manipulate various aspects of the music. The sensors and the motion status of the puck are linked to an Arduino, which processes the data and maps it to specific musical elements through Max, with the audio being rendered in Reason.

Below are the key design elements integral to our project.

1. Complete Integration of Gameplay and Music

The main objective was to design a seamless experience where the mechanics of a traditional air hockey game directly impacted the musical output while the players were in action. Every event—such as puck motion, puck collisions, wall hits, or goals—was intended to trigger a corresponding musical and gameplay effect, turning the project into an intuitive interface for generating music.

By linking musical parameters to various in-game actions, we aimed to allow players to "perform" music effortlessly to complement the competitive nature of the game. The interaction design was key: players should feel as though they are engaging in a natural gameplay experience, not learning a complicated musical instrument. This proved to be quite complex, so we added control boxes on the sides to enhance the game's musicality. However, to keep the game accessible, we simplified these controls to just three knobs and two soft pots, avoiding unnecessary complexity. With the slider for goals scored removed, each user can alter a total of five controls during the entirety of the playing duration and they do not have to update anything to reflect the score change. We aimed to place the sensors in a way that felt natural and ergonomic for the player so that they did not have to exert extra strain in order to activate various musical features.

2. Player-Controlled Musical Expression Through Game States

Our primary goal was to give players the ability to indirectly influence the musical expressions and shape the sounds of the game. Actions like collisions or goals would naturally impact the rhythm, pitch, and instruments of the game, enabling both creative sequences and musical exploration. To enhance expressiveness, we added rotary and linear soft pots, allowing players to adjust tempo, scale, and other musical effects as they interacted with the instrument.

3. Real-Time MIDI Generation and Processing

Another important aspect of our design was ensuring that MIDI could be processed in real time from the game actions and sent directly to Max for optimal efficiency and responsiveness. We aimed to minimize latency and delays between data generation and processing to enhance the overall game experience. Furthermore, we ensured that the musical changes happen from the most recent player controls, as musical lags would negatively impact the ongoing and competitive nature of the game. With the users having the potential to constantly adjust their features, we wanted to process any current data before it was overwritten by newer data. By mapping sensor and gameplay data to specific MIDI events, we allowed players to interact with both the instrument and the game simultaneously.

4. Scalable and Modular Design for Future Iterations

During the planning phase of our project, we explored potential pathways that could enhance the game experience, with one key example being the use of Bluetooth for collision tracking and puck location. In this iteration, we incorporated Bluetooth inside the puck to measure whether the puck was in motion, and thus corresponding to the LED display around the table. We also utilized sturdier mechanical components that would render the table reusable in potential additional phases.

Not only was the physical design considered, we also looked into how we could upgrade the musical experience for increased compositions and musical elements. Each player is able to choose their “character”, which gives them a unique musical sound. The musical design is adaptable for different music genres and soundtracks based on player feedback and interaction.

Fabrication and Assembly

1. Mechanical Components

- Air Hockey Table Structure (see Figure 2)
    - Material: 2x4 wood panels and Acrylic Sheets
        - We chose wood once again for the base and the walls because we wanted it to be structurally sound throughout the high intensity of and impact from the game. We also knew that wood would be a suitable material to incorporate the sensors with and a reliable material to hold our gameplay pieces together. We cut and screwed the wood pieces together and layered all other components onto the wooden base.
        - We used acrylic again for the playing surface because we saw potential from the last iteration in allowing for smooth puck movement. We laser cut holes onto a large 3mm acrylic sheet, instead of requiring three separate sheets, to allow for airflow to reach the puck, while also maintaining the air tightness to emulate a traditional air hockey table. However, given the thinness of the sheet, we notice slight warping directly above where the fan is placed when there is airflow provided, meaning we should have

either provided supports under the center to keep the sheet level or chose a thicker material. The puck is able to glide, though, especially when it is around the center.

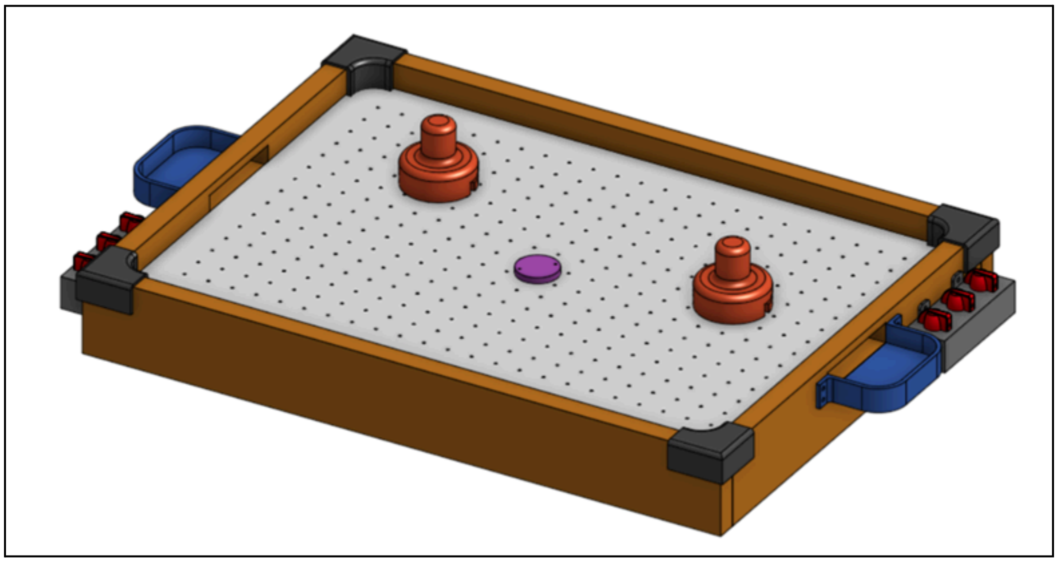

Figure 2: Air Hockey Table Structure

- Airflow System
  - Vacuum/Blower Unit: As mentioned above, we utilized a vacuum to supply pressurized air to the puck to replicate the gliding motion in a traditional air hockey table through a hole in the center of the table, attached directly to the bottom of the table (Figure 3). Compared to last time, we chose a stronger fan so that the air effects would be more noticeable.
  - Air Holes: Expanding off the above, we laser cut 2mm holes into the acrylic to allow the air to be evenly distributed during gameplay.

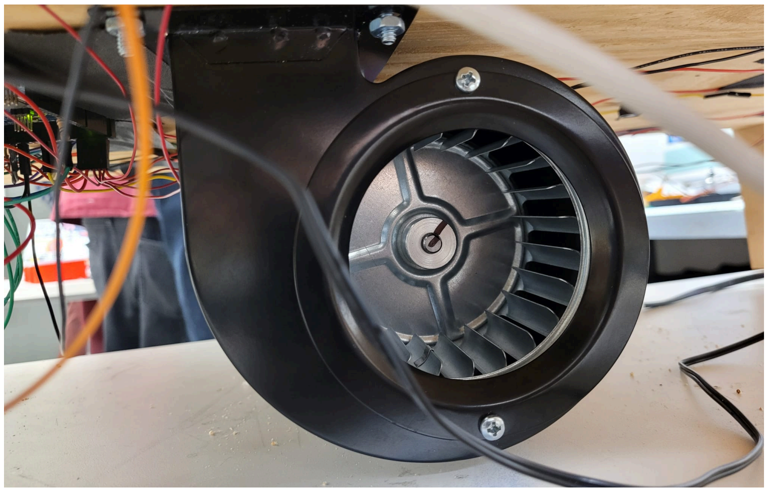

Figure 3: Fan attached to bottom of table

- Other components:
    - Puck: A 3D printed disc with a smooth surface, divided into a lid and base, was used to reduce frictional forces and maintain smooth motions across the table. Last time, we only used the “lid” of the puck, but to incorporate the puck velocity, we added Bluetooth inside the puck, thus leading to the decision to use the entire puck design.
    - Pushers: A 3D printed custom component to offer durability and precision to players - these are the same from the midterm project. We could have also incorporated accelerometers inside the pushers if we incorporated multiple peripherals communicating with each other.
    - Dials: New adafruit knobs were added around the rotary potentiometer knobs (see Figure 4) to make them easier to rotate, instead of using 3D printed ones. The black from the knobs helped the table to look more cohesive and give it a cleaner presentation.

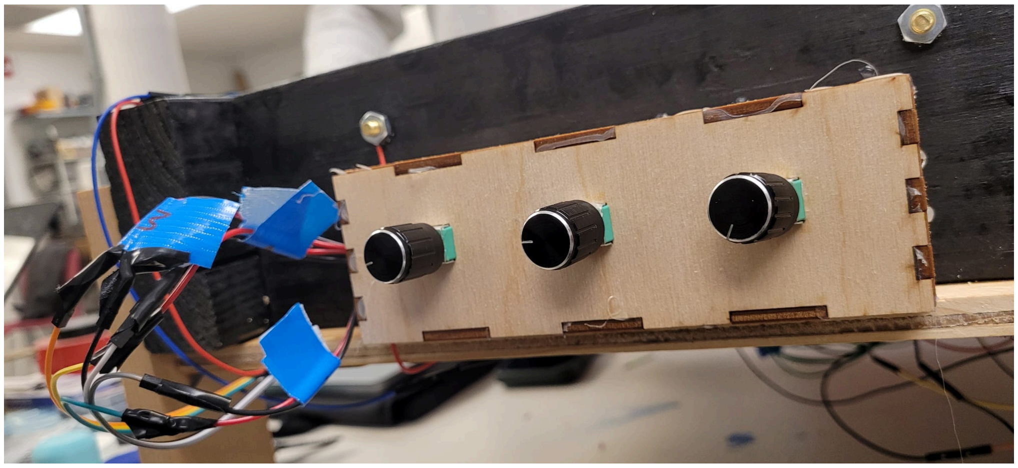

Figure 4: New knobs on rotary potentiometers

    - Knobs box: A custom laser cut box made of wood to hold the rotary potentiometers. The holes on the box are now square shaped to fit the knobs themselves instead of being circles for the knob handles to come out of. We found the new knobs to be bigger than the previously 3D printed knobs and they only stayed on the rotary potentiometer knobs if they were fully pushed onto the latter.
    - Goal region: Two 3D printed goal halves were screwed to each end of the table and taped together on the bottom, with a hole for the IR break beam sensors drilled into them to allow for goal detection. The goals for this iteration are roughly double that of the original iteration to prevent a player from camping directly in front of the goal and taking away from overall gameplay. However, a goal cannot be scored because the wood in front of the goal is slightly too high, and though we attempted to make a tape ramp for the puck to slide into the goal, the puck was getting stuck on the tape. We would have had to shave down some of the wood for this goal region to be fully functional.

2. Electronic Components and Input Devices

- Collision Detection
    - Combining our previous cardboard microswitch, see Figure 5, with the goal to make the table look sleeker, we settled on brass plates and copper tape for the collision detection. The copper tape behind the bottom of each plate provided a common ground to all the plates. Each plate was powered separately by twisting a wire around the right screw and running it from the outside of the table to send collision data to the Arduino. The plates were screwed onto the insides of the table with a washer between it and the table, allowing us to easily adjust the separation distance to optimize collision detection. The washer gives us a little space to work with between the wall and the plate, ensuring that there is not always contact detected. Given the stability of brass over cardboard, we had to make minimal corrections to the plates after we found a reliable placing for them, while the cardboard would lose its form after a couple tests, requiring us to constantly check for contact and reshape it.
    - This followed from our idea developed during the first iteration to have the switches automatically reset to a neutral position after the puck makes contact with it. We again utilized a total of 10 plates, where each plate tracked collisions independently of the other ones, and each one lit up a corresponding LED segment above it when it registered a collision. In addition to improved consistency, this setup enabled greater repeatability and effectiveness in collision detection.

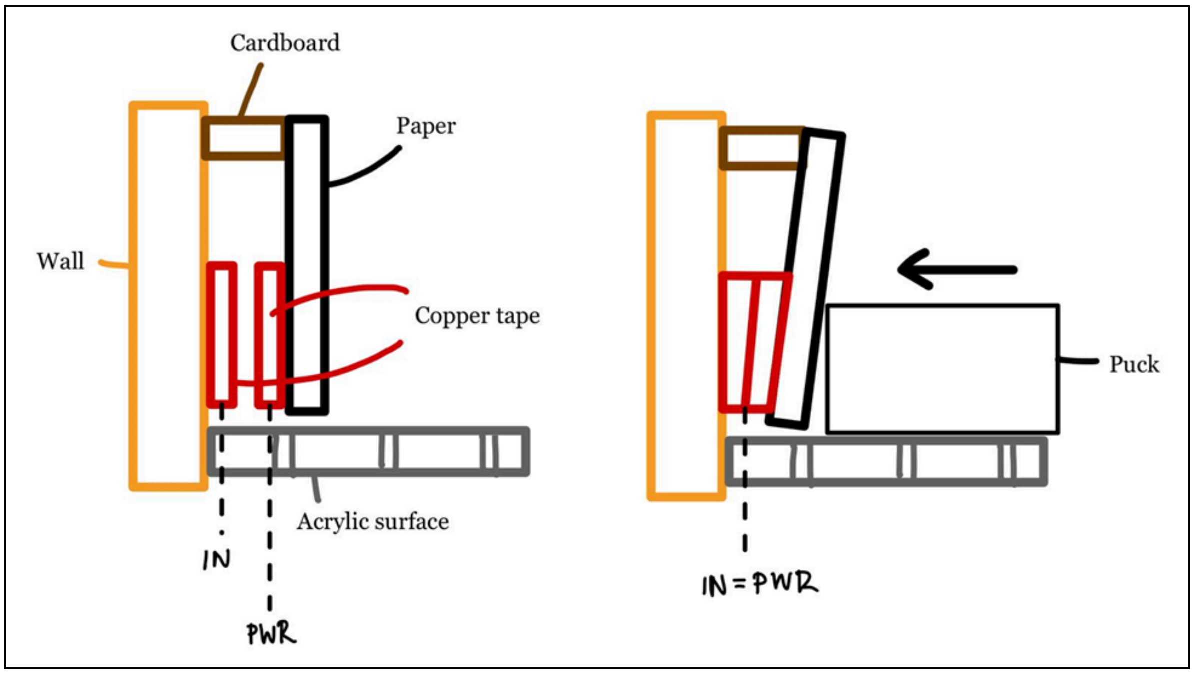

Figure 5: Our DIY sensor for collision detection where the cardboard is now replaced with brass

- Linear Soft Potentiometers (Soft Pots)
    - Each player had a set of two soft pots on the right corner of their side of the board to control musical parameters. These control the same parameters as the previous iteration. For both players, the vertical soft pot controlled the pitch bend. For player 1, the horizontal soft pot controlled LFO2 and for player 2, the horizontal soft pot controlled the oscillator envelope amount. However, the effects are a bit hard to notice given the backing tracks.
- Infrared (IR) Break Beam Sensors
    - Again, same as last time, we drilled holes for two break beam sensors close to the goal posts of each goal to detect when a puck passes through them. Each time the receiver detects a broken beam, it sends a 1 to the Arduino that a goal is scored and turns on the corresponding LED colors of the player that scored. These events triggered a 10 second delay in the Max patch to allow for score updates, player reset, and increased music tempo.
- Rotary Potentiometers
    - Also on the right side of the board were three rotary potentiometers for the player to adjust musical parameters as the game was in action. This feature merged the competitive spirit that air hockey brings with the artful expressivity of music. These knobs allowed for player control and music personalization, sending analog inputs to the Arduino as the values changed.
- LED Strip
    - To upgrade from our prior design, we incorporated visual feedback for wall detection through different segments of the LED strip lighting up when there is a collision. At the start of each round, one corner segment stays on a bit longer than the rest to indicate which player has control. The colors can be toggled at the start state, where each color is supposed to correspond to a unique sound. When a goal is scored, all the LEDs light up sequentially and slowly fade away. When the bluetooth and accelerometer was working, the LEDs would remain lit for as long as the puck was in motion, thus making it possible to light up all the segments during gameplay, then fade away when the motion was no longer in motion.
- Toggle Switch
    - Within the game start state, we incorporated a toggle switch to change between the different characters. When the toggle is reset to center the LEDs light up to meet at the middle with the colors for the different characters, then the game starts.

3. Microcontroller and MIDI Hardware

- Arduino Mega
    - The Arduino Mega converted all the data from the sensors to be processed and sent to MIDI in the designated format and range. We could process the data immediately and in real time, where we were able to use both the digital and analog pins for the various sensors. The data from the protoboard was sent to analog pins, where the Arduino could process input from the sensors

simultaneously. The ground and power pins connected to the respective rails on the protoboard, where we soldered header pins to allow for easier connections to these common resources.

- Seeeduino and Accelerometer
    - This was the bluetooth module we were using to transmit and receive accelerometer data. In a perfect world, the module inside of the puck (Figure 6) is powered by a coin cell battery and able to continuously send acceleration to the receiver module on the bottom of the table (Figure 7). However, the battery is not very strong, leading to the module to stop working. Also, the wires are very fragile so it was difficult to fully incorporate it with the puck, though we were able to demonstrate proof of concept by having it outside of the puck and determining a boolean of moving and not moving.
    - The original plan was to send different PWM, pulse width modulation, signals through a counter that tracked the amount of times the clock went high. This would allow us to scale down the read in acceleration value, which was calculated through the average of the change in motion in each direction. Unfortunately, we noticed that the serial print statements on the seeeduino were too slow, as the prints happen on the fastest frequency. This meant that slowing the clock down more would increase lag and not be a productive way of tracking the acceleration data, which is why we reverted to recording a binary value to indicate whether the puck was in motion.

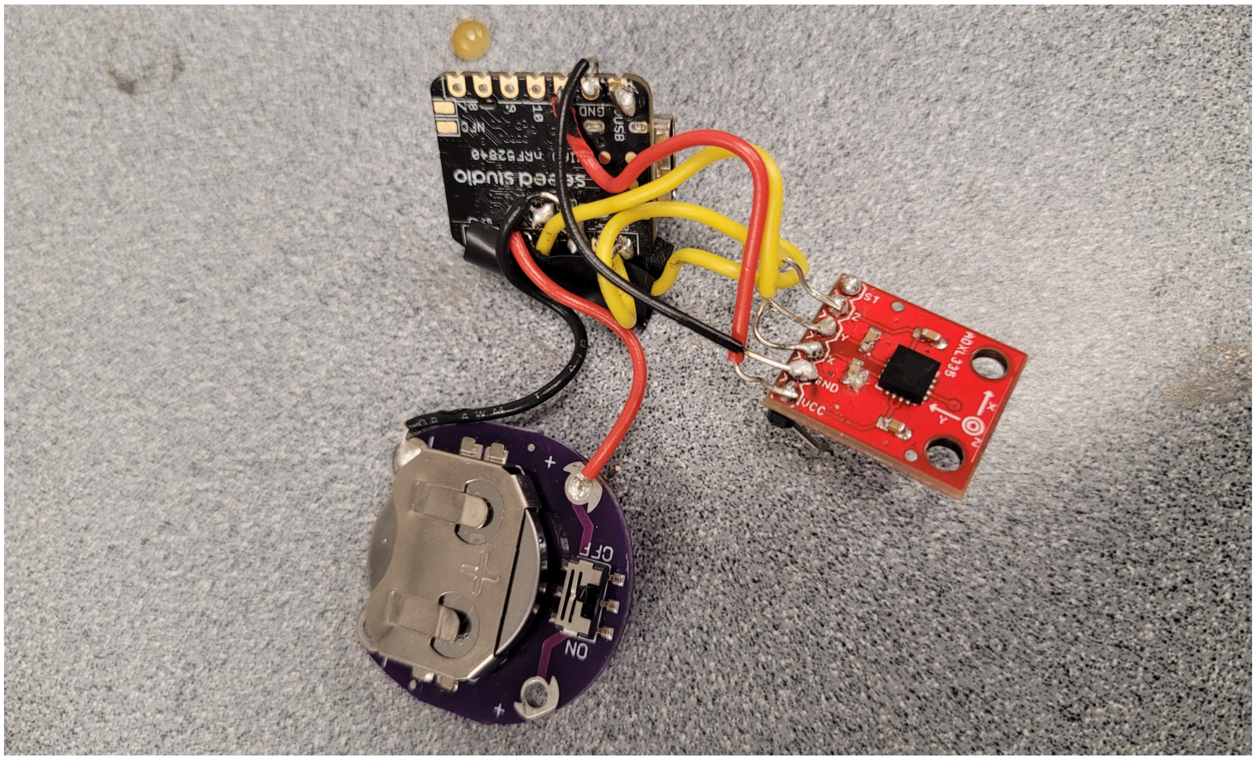

Figure 6: Bluetooth, accelerometer, and coin cell battery component meant to be inside the puck

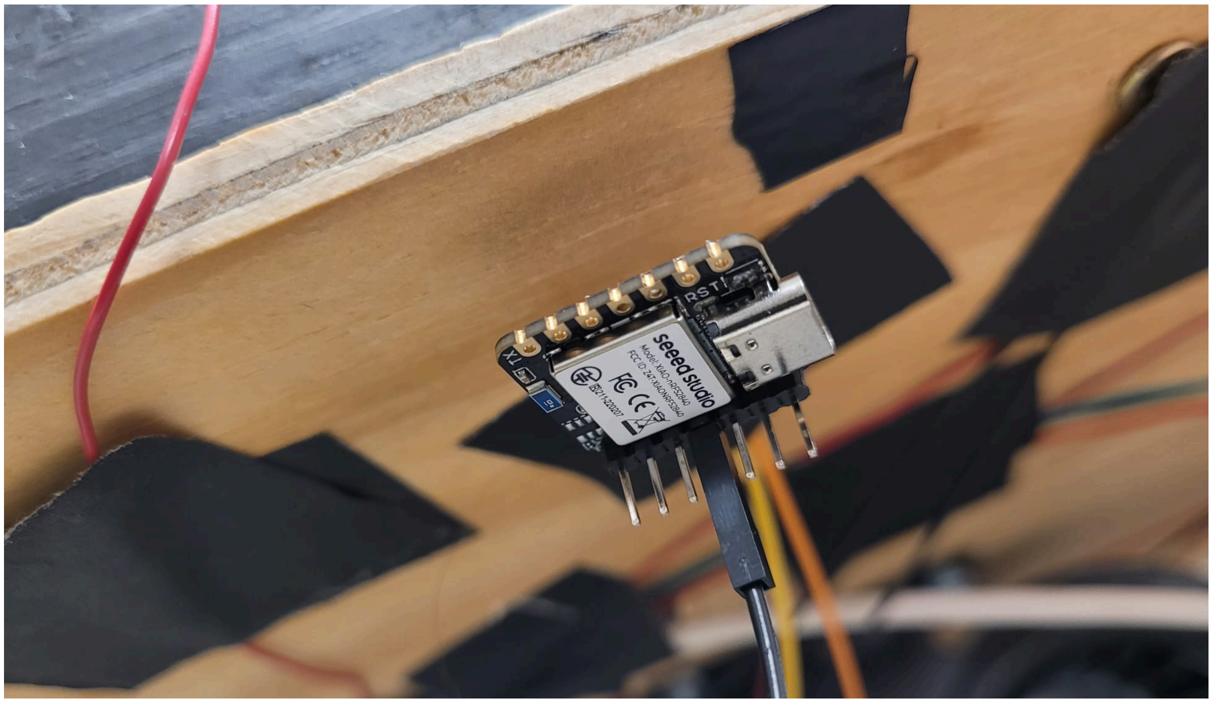

Figure 7: Receiver module at bottom of table

Circuitry

We reused the protoboard from our previous iteration (Figure 8) to hold data wires while the board was upside down. We did have an extra wire from the first time, so we cut those wires to avoid confusion and reduce the clutter. Unlike a breadboard, this protoboard offers better durability and reliability for our project, as the wires would be securely soldered to the board, rather than hoping they would be sturdy inside a breadboard. We would read the sensor values from the back of the resistor and send them to the Arduino.

The circuitry was straightforward but tedious and delicate. We provided power and ground to the board from the Arduino and connected the corresponding rails across the protoboard. One end of the resistor was connected to power, while the other was placed on an arbitrary row of the board. One colored wire was connected to ground, and the other to the sensor. Eventually, another wire was added between the resistor and the sensor input to send the data to the Arduino (see Figure 9 and Figure 10). In total, there were 10 sets of wires and resistors, one for each of the sensors that the players could control: 3 rotary potentiometers and 2 linear potentiometers. We crimped wires to length and grouped them together in sets of 5 to condense where the wires were beneath the board for these sensors, allowing us to track how they were receiving power and ground. We also labeled each wire from the sensor to track where the data came from and where the data was being sent to on the Arduino. This step would save hassle in debugging as we would be able to easily find the wire instead of having to search through everything each time we ran into an issue.

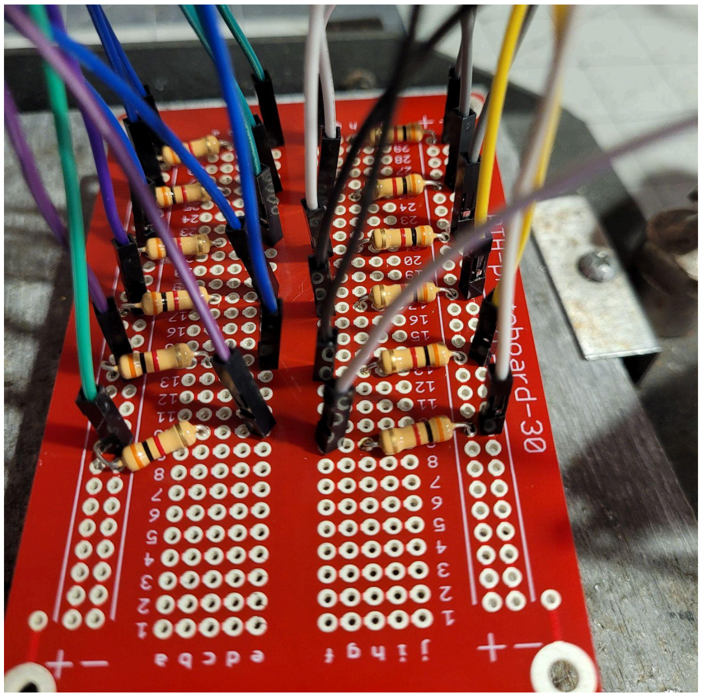

Figure 8: Reused protoboard from previous iterations with only sensor input wires

We used most of the analog pins on the Arduino to accommodate the 12 adjustable sensors, represented by the green wires in Figure 9. These wires correspond to the data being read by the sensors. Since we could not solder the Arduino pins, we used tape to secure the wires to the board or crimped header wires on customized wire lengths. Additionally, the yellow wires in the diagram are connected to digital pins to capture values from the break beam sensors. Most of the wires had to be extended to reach the protoboard from the edge of the table.

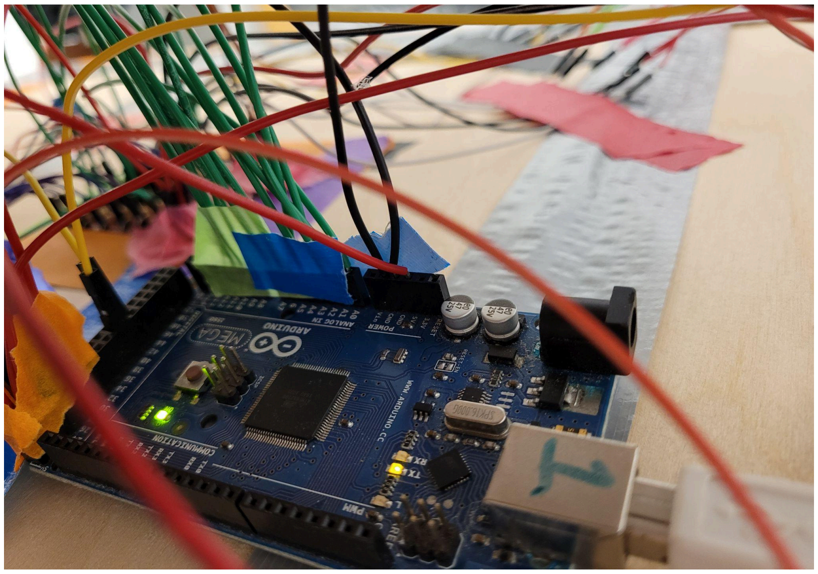

Figure 9: Arduino with sensors connected

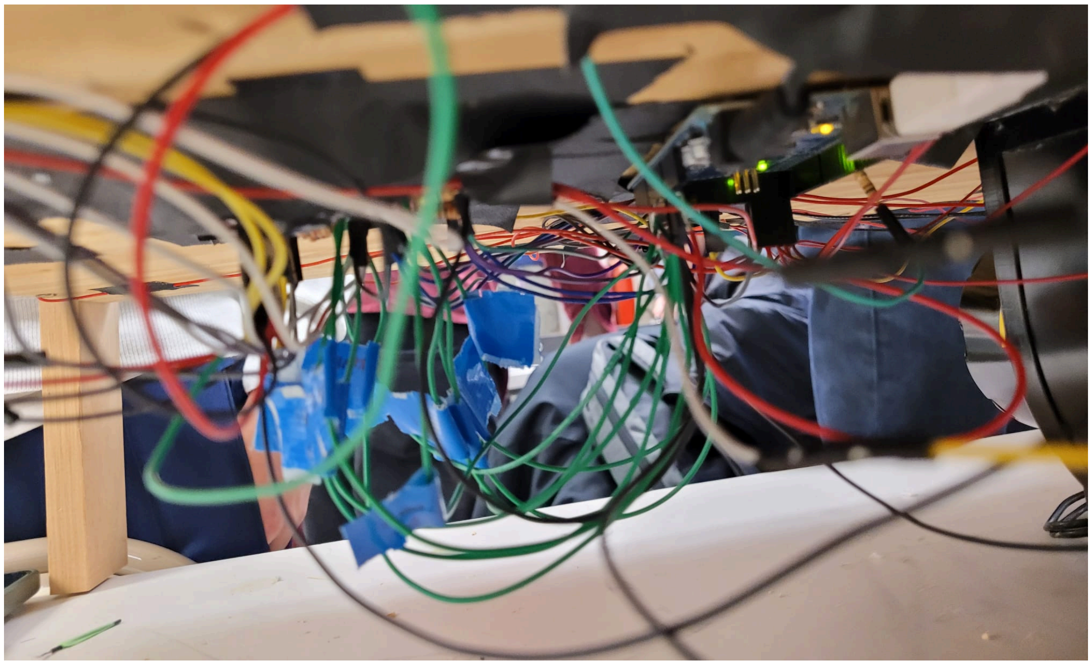

Figure 10: View from under the board of wires connecting from protoboard with Arduino

We did design a schematic (Figure 11) for what we thought the wiring would look like, but given how plans change during the building process, we did not fully reference it. The goal was to organize the wires so that debugging could be easier. We wanted the wires to be flush to the bottom of the table, too. We had to increase the space between the Arduino and the protoboard to allow for all the wires, despite the sketching showing the two being in close proximity. We found that when we crimped wires, it was better to give them a little room to bend, rather than forcing it into a sharp angle, as the headers were done delicately. We turned the Arduino for the analog pins to be closer to the protoboard, as those were where the sensor data was being sent.

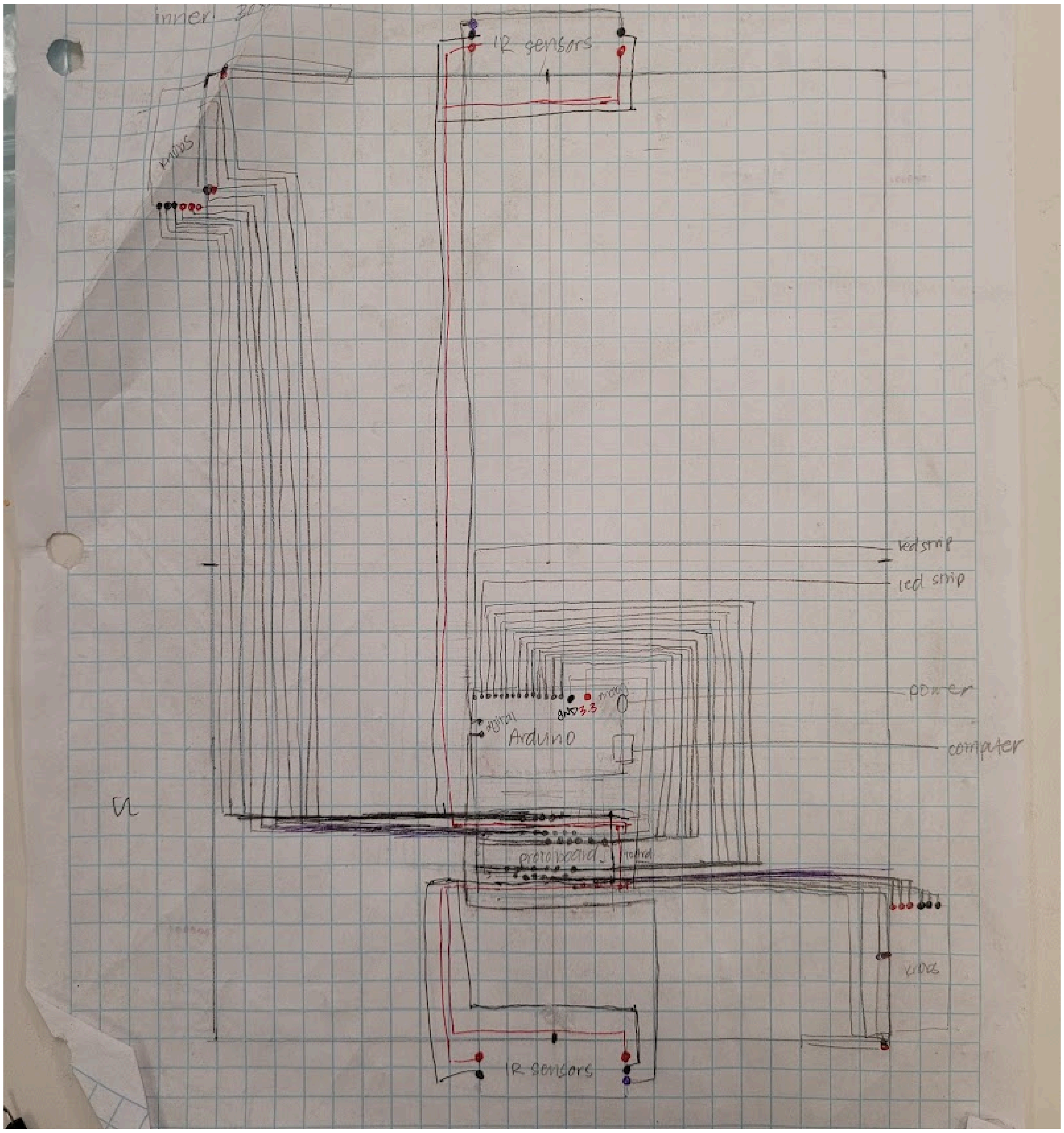

Figure 11: Initial circuit schematic under the table

From above the board, the circuitry appeared much cleaner than before since we intentionally taped them down on the black wood with black electrical tape. They are still wrapping around the edge and directed towards the bottom where the Arduino is. With our new laser cut box, we retained the method from before where there was a hole on one side for the wires to pass through.

Functionality of the Max Patch

- At its core, the Max Patch still has the same features as before. At the highest level, the Max Patch enables each player to control their soft pots and knobs, producing various musical effects and supporting our turn-based concept.
    - Player 1's sounds are outputted through the subtractor patch.
        - The vertical softpot controls the pitch bench using a range of 2 semitones
        - The horizontal softpot controls the LFO 2 amount, which is on Amp
        - The first knob, the one closest to the player, controls the mod envelope amount, which is mapped to FM
        - The second knob, in the middle of the box, is connected to the filter envelope amount.
        - The third knob, the one furthest from the player, controls the mod wheel, specifically parameters such as filter resonance, filter frequency, phase, and FM.
    - Player 2's sounds are outputted via the NN-19 patch using a string sound
        - The vertical softpot controls the pitch bench using a range of 2 semitones
        - The horizontal softpot controls the oscillator envelope amount, which provides a similar effect to the LFO but still presenting unique sounds for the player.
        - The first knob, the one closest to the player, controls the LFO amount, which is mapped to a square wave and the filter.
        - The second knob, in the middle of the box, is connected to the oscillator pitch.
        - The third knob, the one furthest from the player, controls the mod wheel, specifically parameters such as filter frequency, filter decay, and amplitude.
- After a goal, the Max patch plays the backing track faster, adding to a more intense environment. It is still maintaining the same sequence. Interestingly enough, there is a bass drop during the first transition from game start to game playing.

Reason

As mentioned earlier, we implemented a Subtractor patch for player 1 and an NN19 patch for player 2. To incorporate the player characters, there were four patches for each player from their respective modules. Given that NN19 provides a wide variety of sound samples, we were able to create two distinct sounds, allowing players to craft unique musical phrases and enhancing the game's expressiveness. The NN19 patch offers more detailed information about the samples, whereas Subtractor has a more fixed architecture.

To establish the overall atmosphere of the game, we utilized three Reason patches: Grain Sampler, Dr. Octo Rex, and Redrum. For Redrum, we selected one of eight preset patterns to set the tempo of the backing track, based on the sum of the players' scores. All the musical elements of our game were generated using these Reason patches. The Reason sequences provide the backing track.

What You Would Have Done Differently and Future Plans

From the current design, although we did a better job with the wiring, I would have still liked to make it more robust. We were able to use longer wires and crimp them as we saw fit so that they were more flush to the bottom of the board. There were some connections that I could have done a better job with since not all of them were secure, but we were able to tape them together for the presentation.

With more time, I would have liked to make the Bluetooth more robust since the entire component was fragile as well. I resoldered wires on the battery, seeeduino, and accelerometer multiple times since the wires kept snapping. We were unable to fully incorporate it into the puck because of how structurally weak everything was; however, we were able to get it to send data for a proof of concept that we could track the motion of the puck. We wanted the Bluetooth to map to note velocity. It would also be great to add accelerometers inside of the pushers, which could control a musical parameter such as distortion.

With the sensors on the right side of the table, the game is thus played with the left hand. It would be nice in the future to design an ambidextrous game where there is not a bias for a handedness of the player. Currently, it is just preference as to whether the dominant hand controls the pushers or the sensors.

I think a future plan would be to have a project where all the components work seamlessly with each other since we have working parts through the different sections. When we started to put everything together, we were seeing issues with the various electronics. Hopefully in the future, if we were still to work on this, we could create a game where all the mechanical, electrical, and musical aspects are fully integrated with each other to ensure constant gameflow.

Conclusion

The MIDI Air Hockey project successfully merged competitive gameplay with real-time music creation, giving players control over musical parameters. The system employed sensors and custom components to track player movements and collisions, generating dynamic MIDI data that evolved throughout the game. This MVP showcased the exciting possibilities of combining gaming and music, despite the project's inherent complexity. Future enhancements could involve better sensors, more developed wireless functionality, and a wider range of sound options.