Team Upsilon: Jonah Belback, John Darnall, Jay Suggs

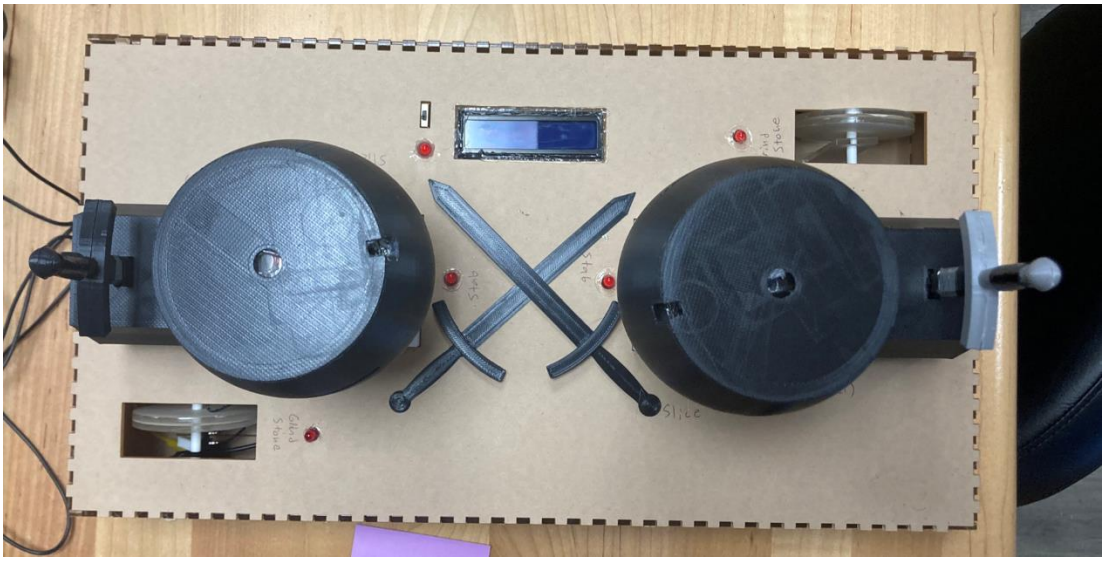
Bop-It Design Project Report

ECE 1895

Design Overview

Description and Purpose

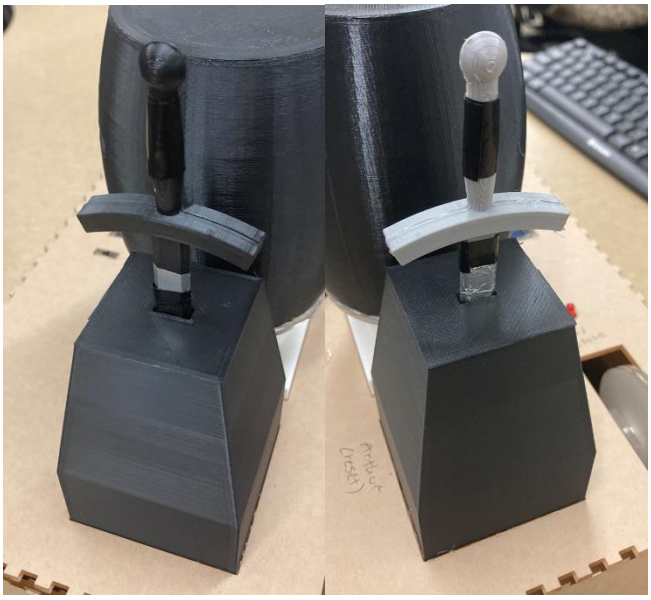
This is Duel-It!



We proposed a competitive game with two players where each player tries to complete their given action on the opponent's side of the board while simultaneously trying to prevent their opponent from completing their action. These actions and the way to complete them are done with tiny swords that have two plates of metal on either side that's connected to each other to either later complete a circuit or just use the sword to press a button.

One of the swords foiled-up

The game starts when the power switch is turned on and both swords are drawn out of their containers (rocks, akin to the King Arthur legend) where it is a slit with a layer of foil on both sides of it each with its own lead/wire. When both swords are in each slit, it completes the circuit constantly resetting the Arduino. This module is referred to as “Arthur” or “Arthur Stone”



Both ”Arthur” Module with Swords placed

Foil design, each sword is entered it completes circuit

Each “round” both players are simultaneously given the same random action that they must complete given to them, the specific one indicated by an LED located near the action item. A round is complete when the first player to complete their action is completed, giving that player 1 “point”, increasing one of the values displayed by an LCD screen (starting at zero), and going to the next round. Each player's value will be on the side of the screen they are on. Only one player can be given a point each round. Each round has a time limit starting at 5 seconds that decreases every round by 10%.

When a player gets to 99, the game ends.

When neither player can complete their task in time, the game ends.

When the game ends and either a player has 99 points or a player is 3 points ahead of the other, that player wins, flashing the LCD screen with “PLAYER # WINS”

In the result that the game ends and neither player wins, it's a draw, and flashes such on the LCD screen.

The game then waits for both players to return their swords to the starting rock they pulled it out of to start, then starting the game again when they are redrawn.

If the game is turned on from the power switch and the swords aren't registered in starting rocks, it will wait for them to be returned.

Actions that can be randomly selected every round:

On the other side of the board across from a player is a barrel they must complete the “Stab” and “Slice” actions that the other player must try to prevent them from completing

1) “Stab”
    a) On the top of the barrel is a slit that the player must put their sword into to score a point before the other player. When the sword is entered it competes a circuit between two sheets of Aluminum foil that's separated by plastic, same as the “Arthur” Module. When the circuit is complete, it gives a point to the correct player.

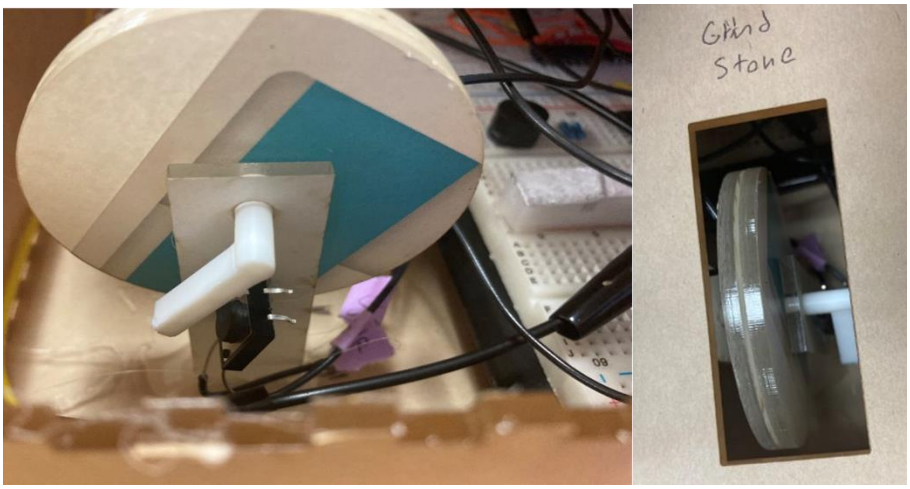
Sword performing “Stab” action

2) “Slice”
    a) On both sides of the barrel are two wide horizontal slits that have a button in each. The buttons are in parallel so a player may press either of them with

the length of their sword to be able to reach the button, and when pressed completes the circuit and gives a point to the correct player.

Close up of sword and button for “Slice” action

3) “Grindstone”
    a) The player must use their sword to spin a wheel in the shape of a grindstone by running their blade across. This wheel is connected to an axel that has a spoke sticking out that, when spun, contacts a button. When the button is pressed the attempt is marked complete for the correct player.



Close up of “Grindstone” mechanism

The game starts with a specific tune. Each round starts with a single note where the round starts when it stops, then an LED next to the specific module that's randomly

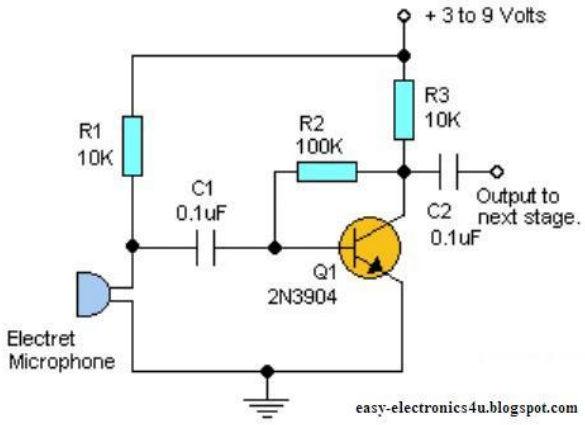
selected lights up to indicate to the players this is what they're fighting over. There's also an LED to indicate the system is on and the circuit is stable. There is also a power switch.

To reduce the number of pins used, each module to connected to a Demux that's selected and given power by two Arduino pins. Each module connects to 1 pin for each player that's used to determine the completion of a round and for what player. This way we use only 4 pins instead of 16.

Considered Concepts

Everything said above was designed and created, however a lot of was either removed or dumbed down to complete it time.

There was originally a 4th module "Warcry" that was scrapped. At the start of the project, we found a hobby circuit that just needed the mic to make with the rest of the components found in benedum. It ended up being fake, but it was too late in the project to order a premade "Sound Sensor", so we just removed it.



Fake Circuit

The "Grindstone" was originally thought to have multiple spokes that were flexible and was programmed for that module specifically to detect a change in what state the circuit was in. We scrapped this and just made it a solid button as we were unsure how to make that realistically.

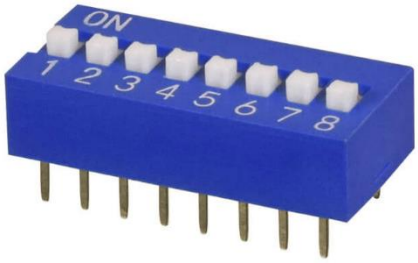
The "Slice" originally was three different boards that stick out of the barrel, each one when pressed hits a button and comes back out with a spring. When "Slice" was selected, only one of these boards was randomly selected and indicated with its own LED and the

module could only be completed if that board was pressed. This was trimmed down and reduced to two parallel buttons in a slit that can only be reached by using the sword.

Design Verification

Hardware and Software

Both hardware and software were tested by using the prototyping breadboard and seeing problems in our components and how they were connected by running code and seeing it progress using a DIP switch to complete the connection of a player's module.



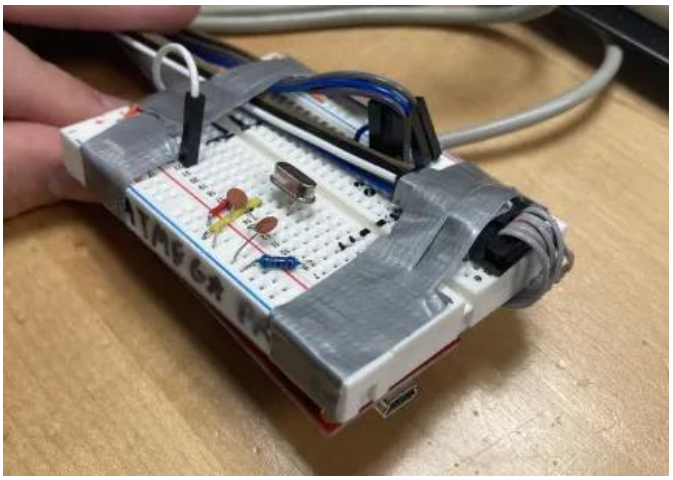
(DS01C-254-S-08BE) Dip Switch used

Checking for expected results when state of a button on the slide switch was complete:

- Only gave a point for the module given power by the DeMux selected at random
- Player points increased by 1 per round
- Player wins if a failure occurs but they're ahead by 3
- Player draw if not ahead
- Making lots of temporary LEDs to a GND connections at certain parts of the circuit to test signals

This let us test even components from the LCD screen, the AND gate, the Demux, the Invertor, and all the pinouts of the Arduino.

To aid in the programming of the system we made this breadboard with the Tiny USB setup premade on it so we could very easily take the Arduino off the board on to this and program it without any risk of breaking a programming computer from a connection of a 9v battery.



Video of testing the Lead connections soldered to buttons and foil contacts before stuffing them into Module Enclosures:
https://youtu.be/-gJ3eO2wnik

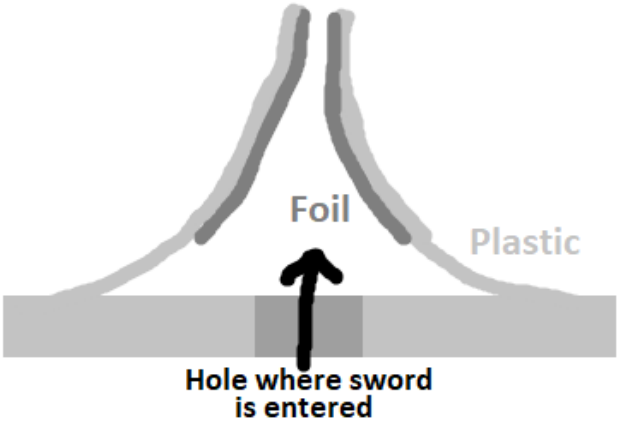
Enclosure

The enclosure of *Duel-It* was not tested until most of the hardware and software had been tested. There was a lot of consideration taken into designing the various components of the enclosure as well as sizing them properly, since we were pressed for time and would only be able to 3D-print the large components once. Before placing the wires and components inside the 3D-printed modules, we first connected them all to the breadboard and made sure they all functioned as intended. We then planned out how we would attach the components to the modules and fully assemble the enclosure. We slotted all the parts together before attaching them to each other to verify that they all fit together properly and did our best to make sure all actions were easily completable before full assembly.

When assembly came, we did some testing on different types of foil connections. The one we decided on, the two-sided foil between plastic, has a diagram in the section ‘Design Overview’. We tested these by entering tying the two different foil connections to power and the positive terminal of an LED connected to ground temporarily on our breadboard, then entering a sword wrapped in foil in and out several times. They were all very unreliable, especially after multiple uses, but the two-sided foil was found to work the best.

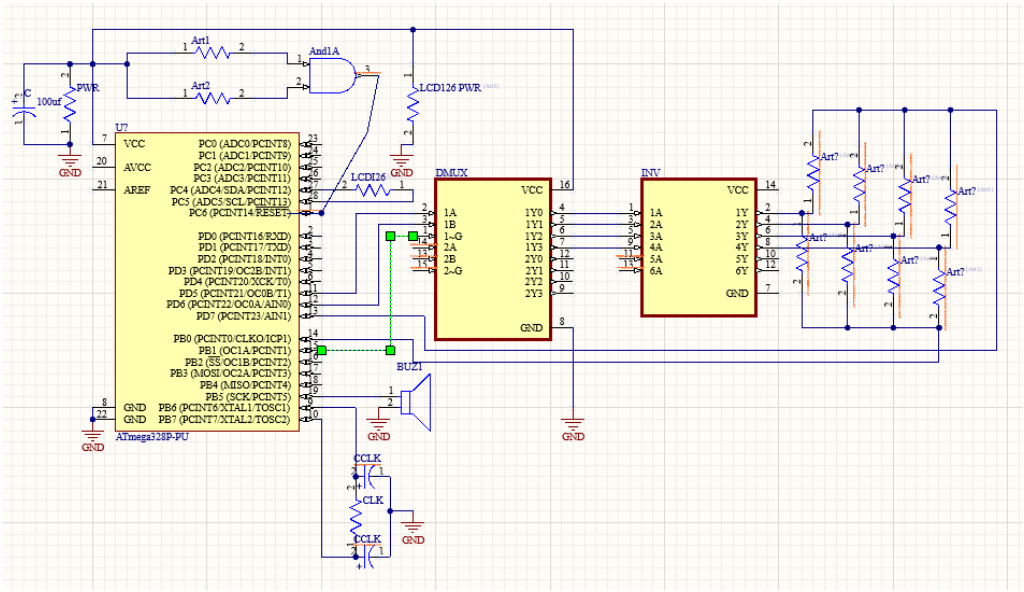
Two other designs that were thrown out:


Sodium
Protein
ORANGE
#6, COTTON SEED


Foil
Plastic
Hole where sword
is entered

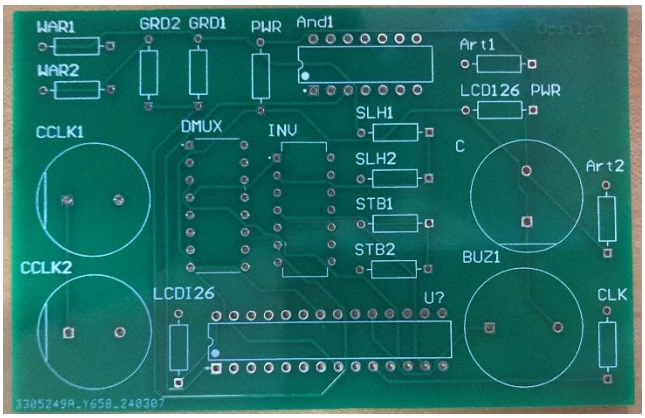
Electronic Design Implementation

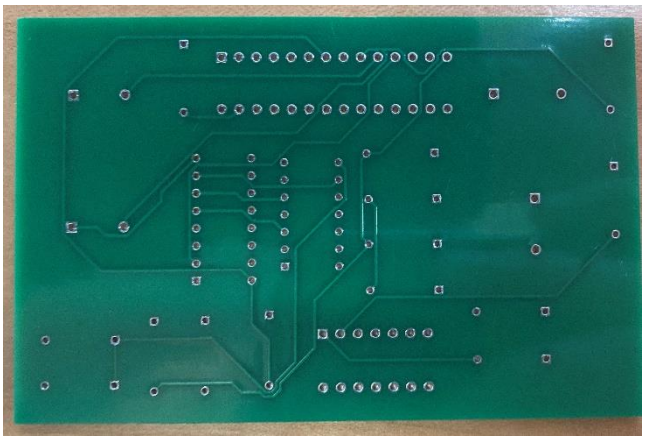
PCB Schematic

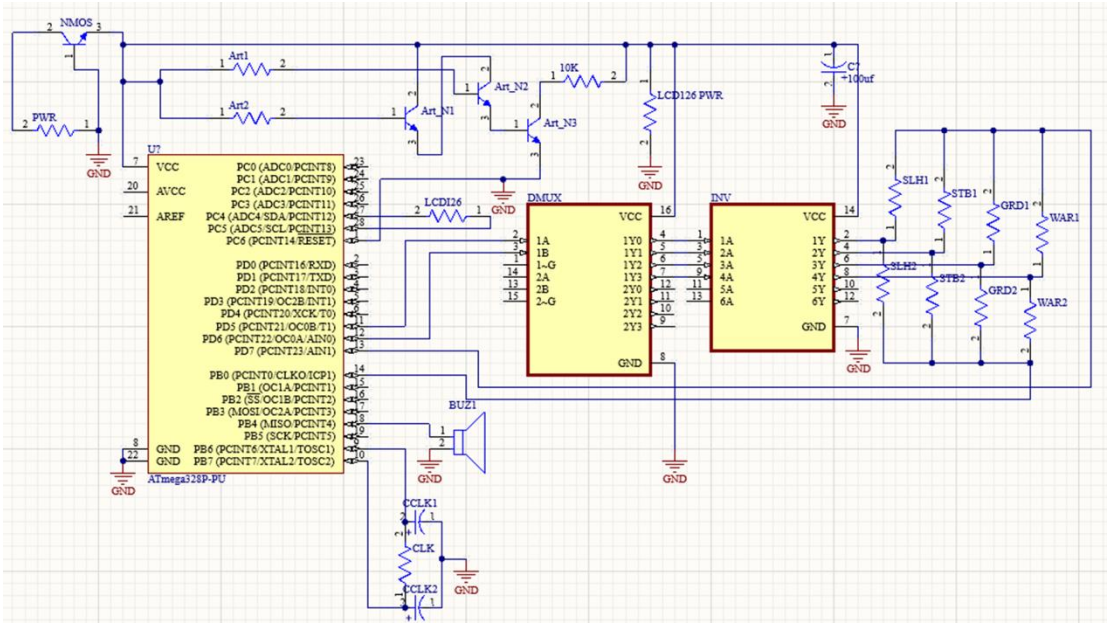


First Iteration

Submission for assignment 15, based on our original understanding of the components.





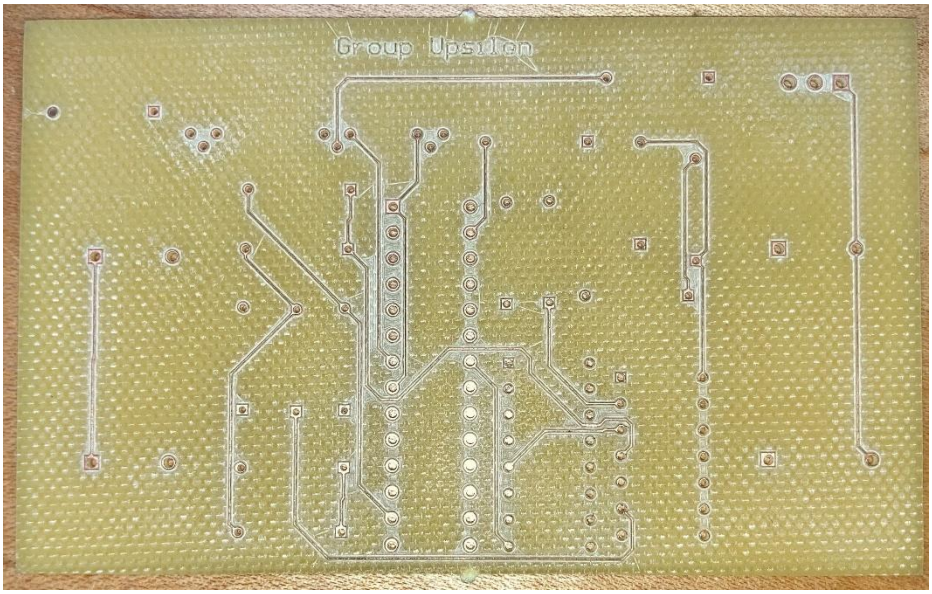


Second Iteration

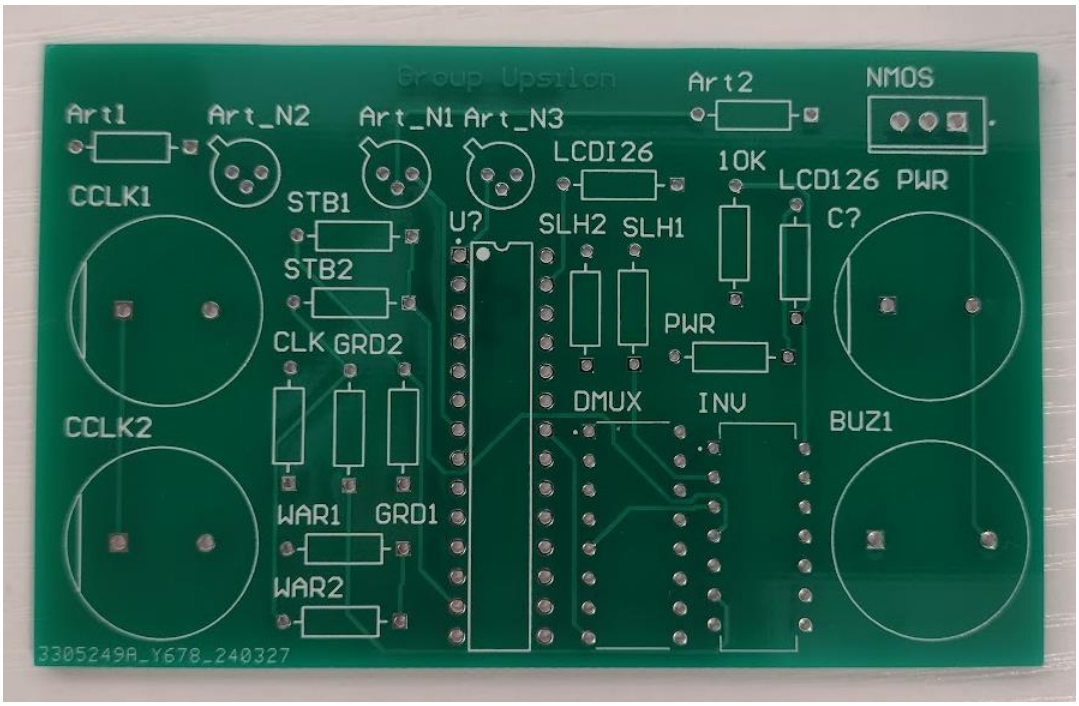
In the second generation we got around a problem with the “AND” chip we were using to just use make the circuit ourselves with NPN transistors. This was mainly influenced by a more completed prototype and the arrival of some components we didn't have in the first Iteration.

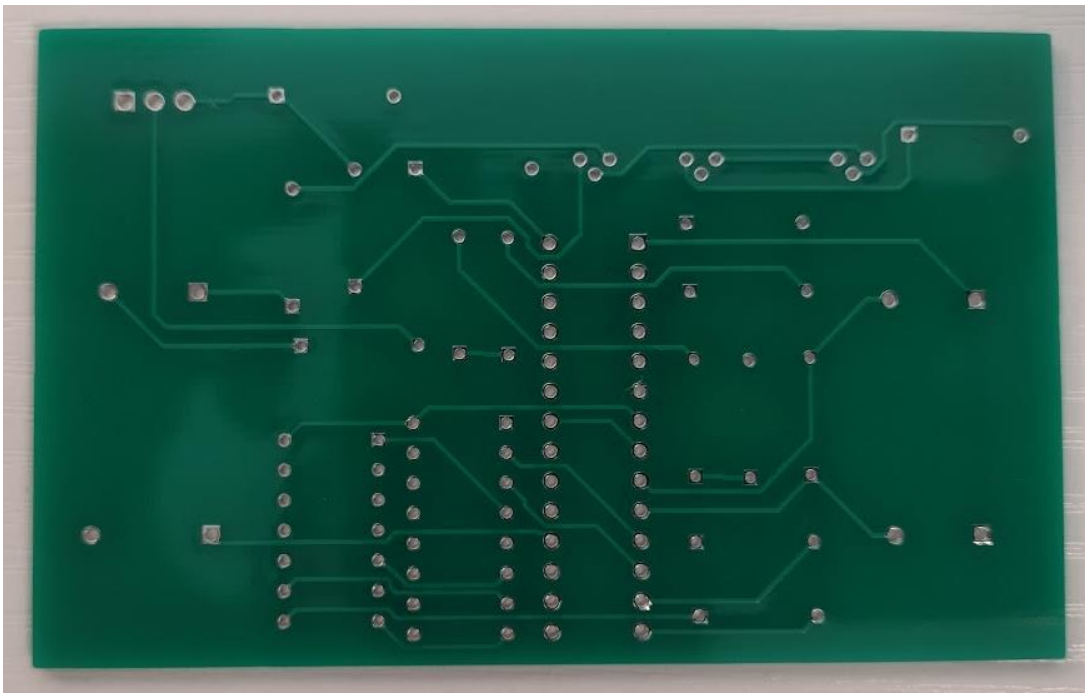
The modules are split off into just a positive/negative with the intention of them being their own disconnected circuit made on a Perf board that when provided power will output a signal to the Arduino noting a complete attempt.

We had to print this PCB via SERC; we made another order, but it wasn't made in time.

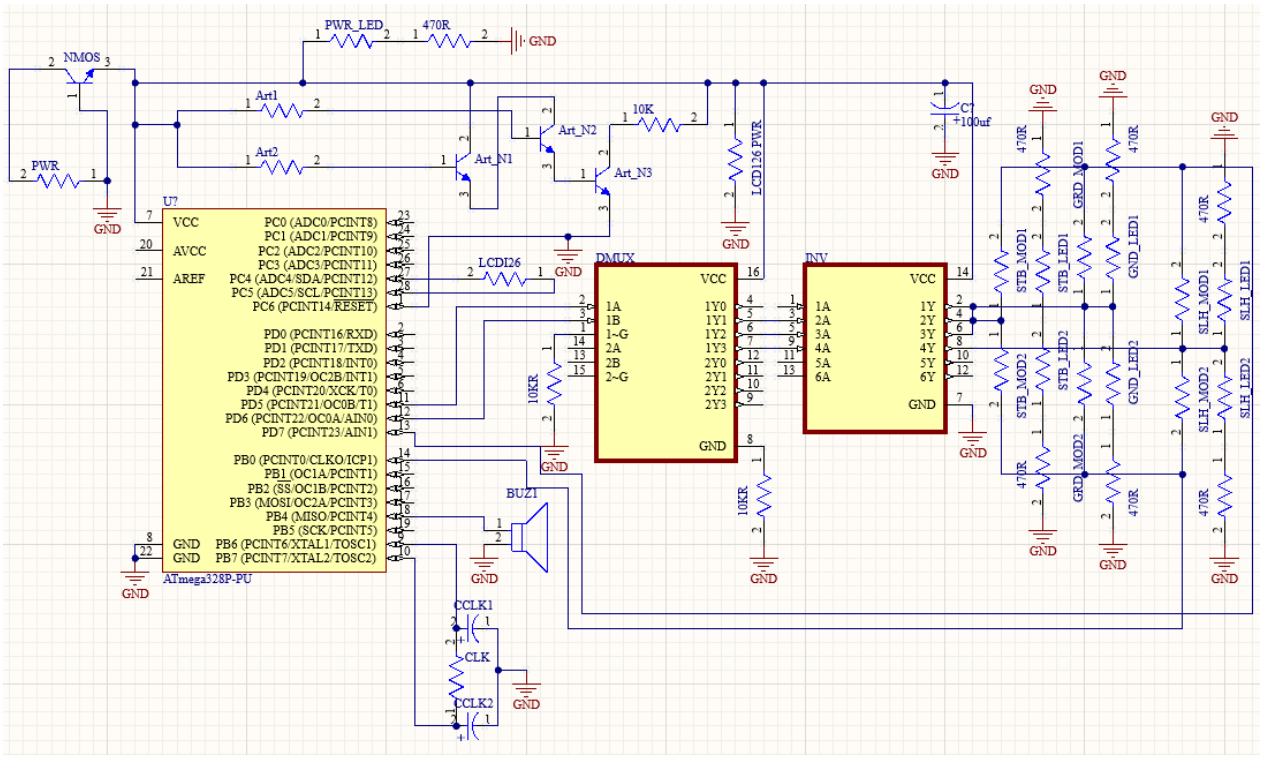


SERC PCB Iteration 2





JLC PCB Iteration 2

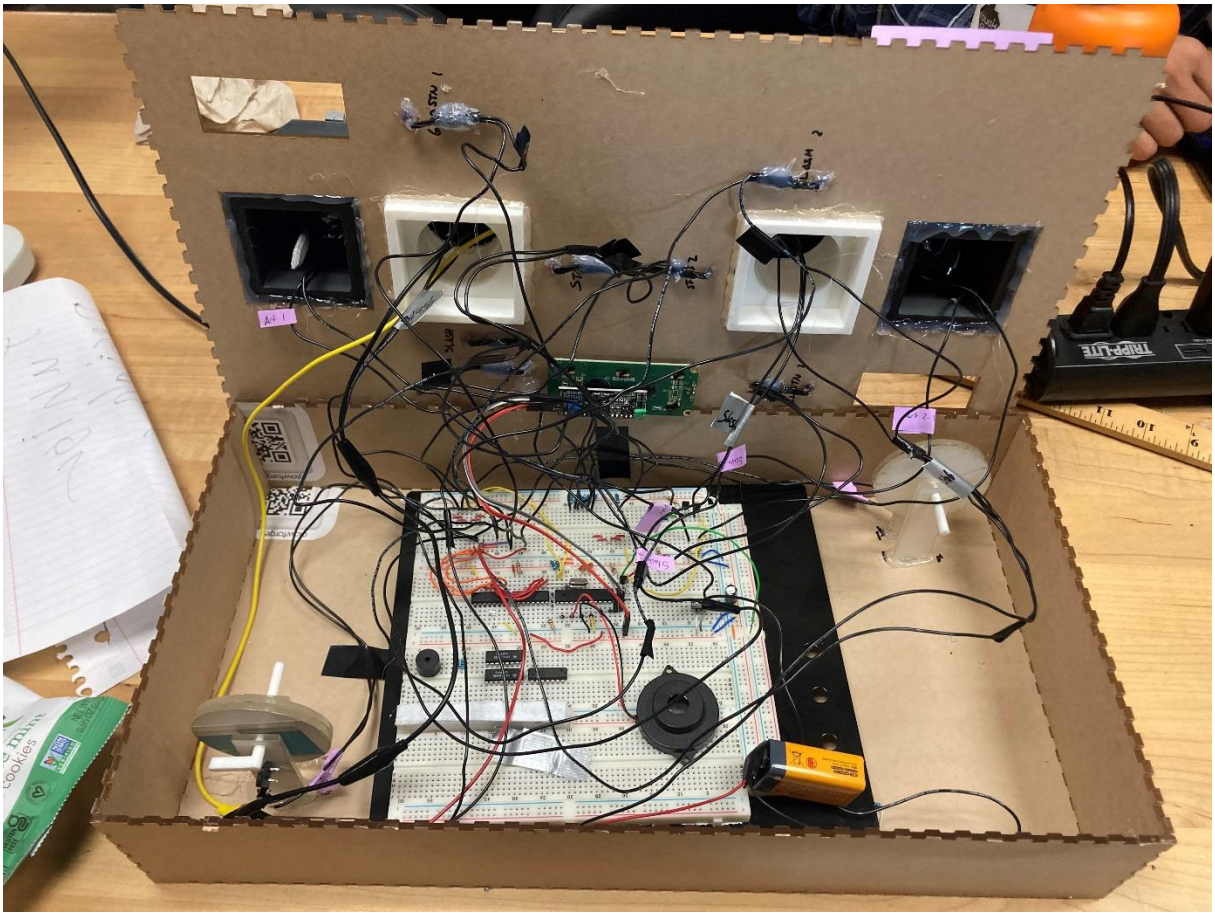


Third Iteration

The third generation was made after the showcase after all the changes we had to make on the breadboard during final assembly. We ran into a lot of issues we didn't have in prototyping where there were lots of connections from power or ground that needed a resistor to now work properly.

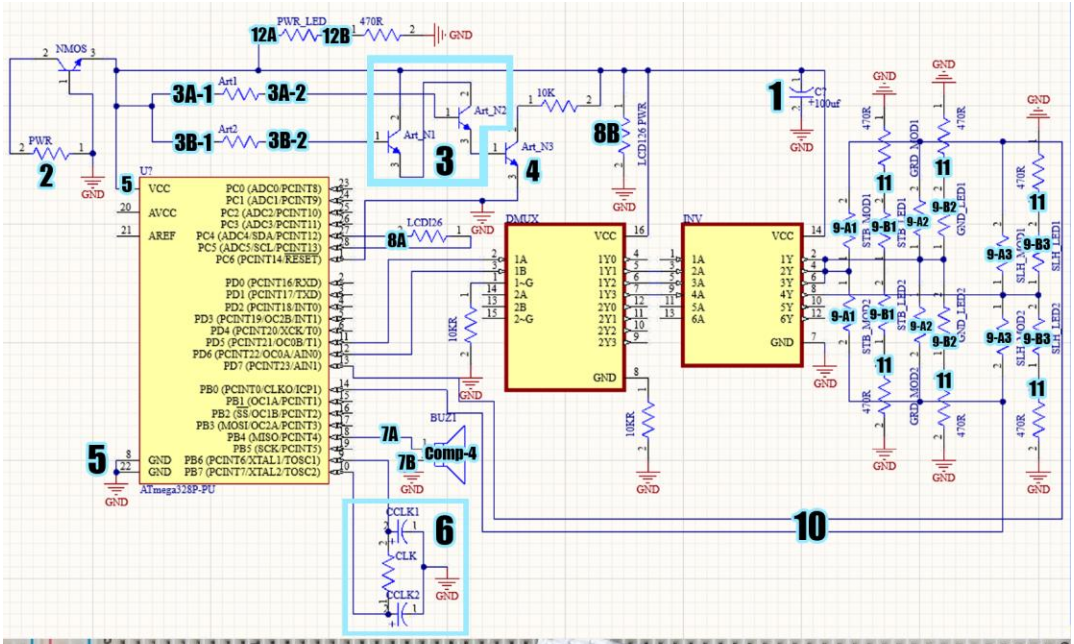
We also gave more outputs for each module, having the work being done on the PCB rather than a disconnected Perf board.

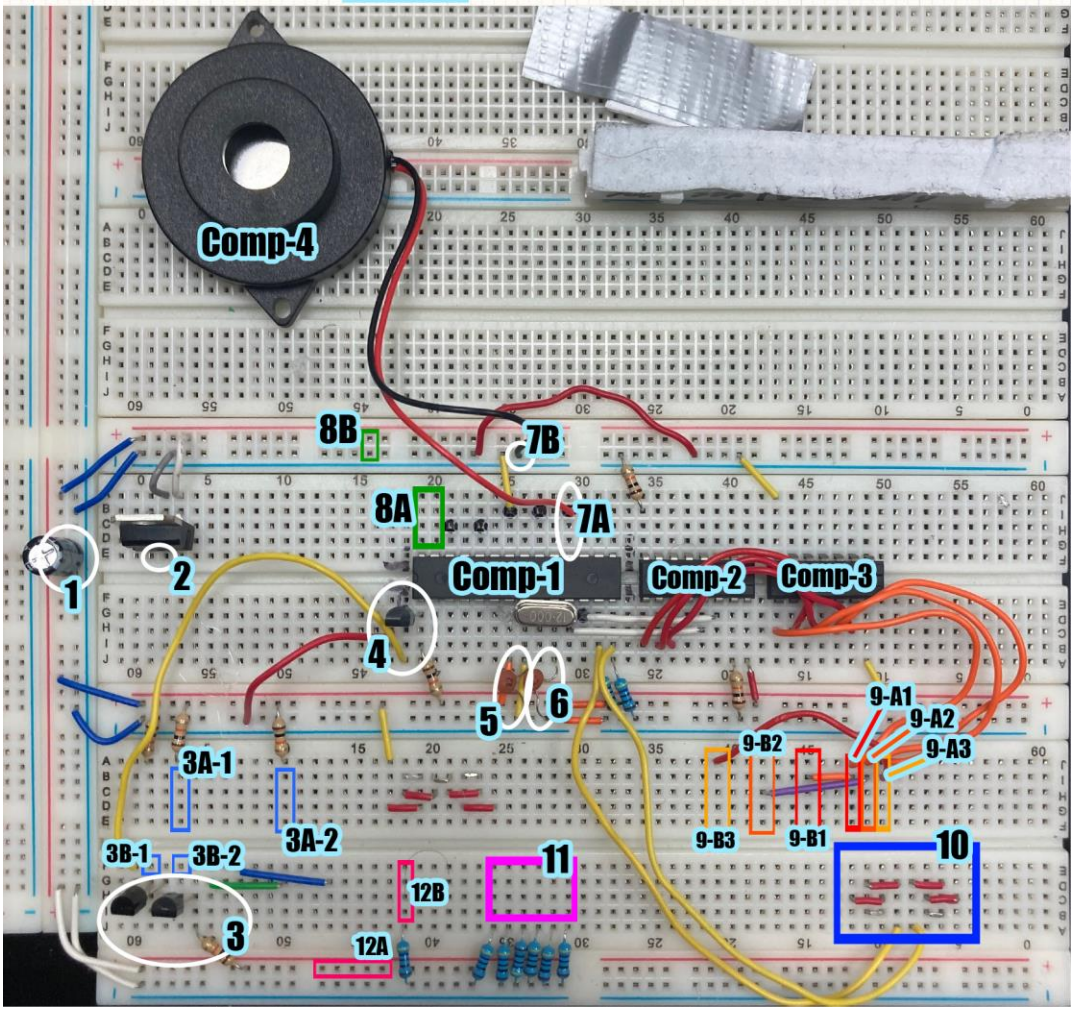
Because we couldn’t get another SERC board in time for the showcase we had to keep this design on a breadboard with a bunch of lead off wire connections to each module. We considered making it at SERC just for the report after the Showcase, but Professor Dickerson said not to bother.



Third Iteration on a breadboard with many lead connections (rat nest)

PCB Layout (clear Schematic two pages ago)





LEGEND: (“>>>”: note***)***

Comp-1: atmega328p

Comp-2: 2:4 Decoder

Comp-3: Inverter

Comp-4: Speaker

1) 100mF Capacitor over positive and negative of whole system
2) Battery Plug-In (leads to voltage regulator)
   >>> Positive: left slot, Negative: right
3) AND Gate: King Arthur Reset
   a) Positive Terminal of King Arthur Module (1: player 1, 2: player 2)
   b) Negative Terminal of King Arthur Module (1: player 1, 2: player 2)
4) NPN Transistor ruling over reset of Arduino
   >>> Gate controller by #3
5) Positive(left), Negative(right) of Arduino
6) Arduino Clock and Capacitor
7) Speaker Terminals
   a) Positive controlled by Arduino
   b) Negative
8) LCD Screen Terminals
   a) SCL(left) and SDA(right)
   b) LCD Positive(above) and Negative(below)
9) Power to Module [selected from demux and corrected by invertor]
   >>>Two leads off each row for both players
   a) Power to Module’s Button or Foil Mechanic
      i) Stab
      ii) Grindstone
      iii) Slash
   b) Power to LEDs indicating Module
      1-3 same order as 9A
10) State Feedback from Modules (circuit completion from) 9A
   >>> Left Connected to left yellow wire: Player 1 Modules
   >>> Right Connected to Right yellow wire: Player 2 Modules
11) GND for the Module Indicator LEDs from #9B
12) System Power Indicator LED
   a) Positive Terminal

b) Negative

Software Implementation

Code Architecture

The Code Architecture was in Arduino with a setup function and a looping function.

In the Setup function: the LCD was initialized and set to display the scored with was made to zero. The original time between rounds is set to 5 seconds, and the tune for the start of the game is played.

In the loop a function is called to do the setup for the start of a round including making a random selection, reducing the time by 10%, and playing the tone to indicate its start.

Then in the loop based on the selection made in the round start function it goes through an if statement branch that applies the right digitalwrite to the demux to select a module and give it power. Then it goes through a while loop where either player 1 gives the Arduino a signal they won and breaks out of the loop, player 2 does, or time runs out and calls a failure function that holds the system in it forever until the Arduino is reset.

In the case the round was completed, a tone is played indicating a win and the player that scored gains a point which is updated in the display.

GitHub: https://github.com/turtneck/JEB382_ece1895_Duel-It

Sub Routines

There are functions split up from the main loop for readability like starting a round in 'start_func()' that was described below

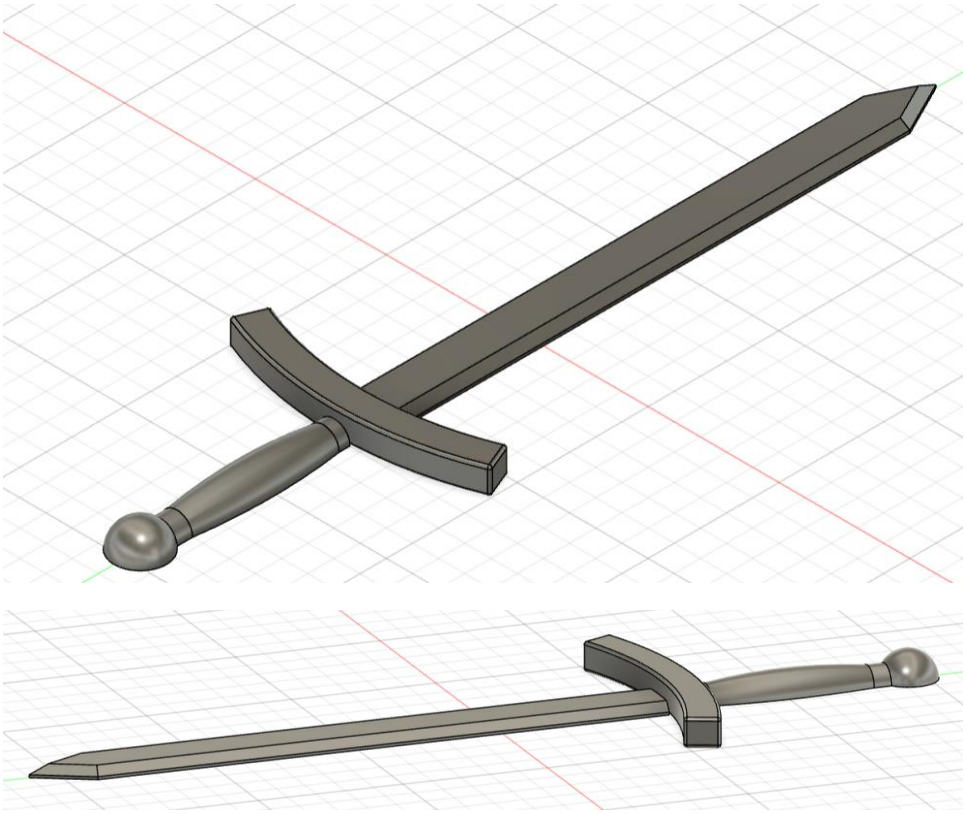
Other functions:

1. buzzer_start(): plays the tone for the start of a round over the buzzer using the 'tone' library
2. fail(): displays a mocking message for failing a round complete with a song that goes note by note with the partial display of the message.
    a. In the case of a draw, it will display such and make a unique song
    b. In the case a player wins, it will display such with the right player and a unique song.

3. playerwin(int player): it will display the exclamatory message of victory with the right player given by the ‘int’ argument ‘player’ and play a unique song.
4. score_disp(): update the display showing each players point

Enclosure Design

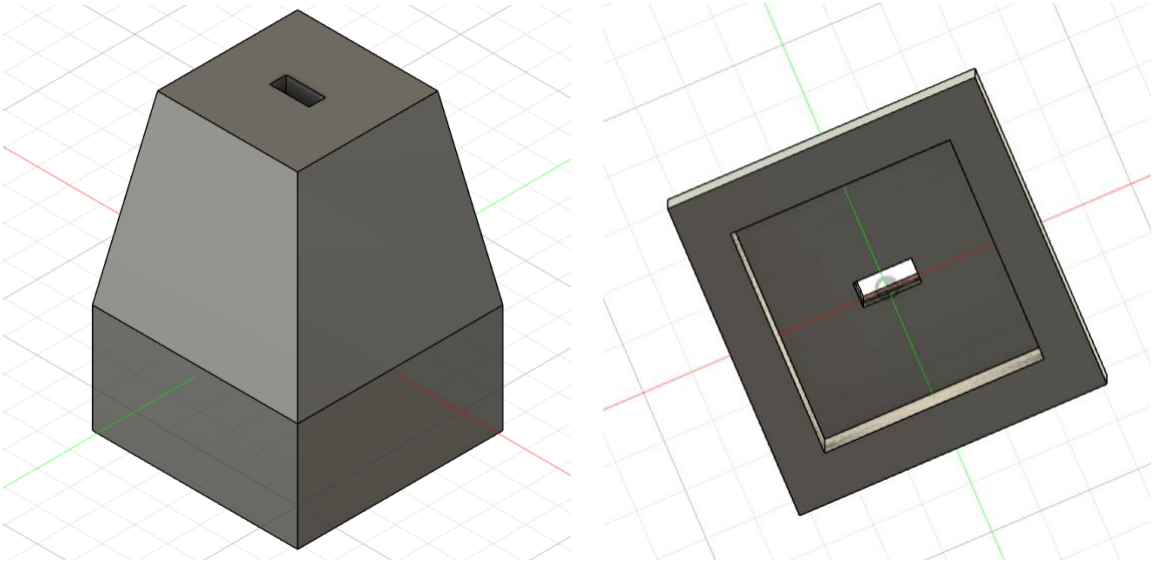
Since the theme of our design was a medieval duel, our enclosure closely resembles the training grounds of knights and squires. There were four main components that needed to be designed and printed. First, are the swords. Each sword was 3D printed in halves, so that aluminum foil could be placed in the middle of the sword, enabling the sword to complete circuit loops when slotted between two metal leads. The swords are an original design and are approximately 160mm (around 6”) in length, with the blade being 1cm (around 0.4”) wide. After fully assembling our enclosure and running into some problems using the sword. We resorted to wrapping the swords in a layer of aluminum foil rather than inserting a layer in between the swords. This provided us much more leeway when executing the “Stab” command and placing the swords back in their starting modules.



Sword Fusion 360 Design

3D-Printed Swords

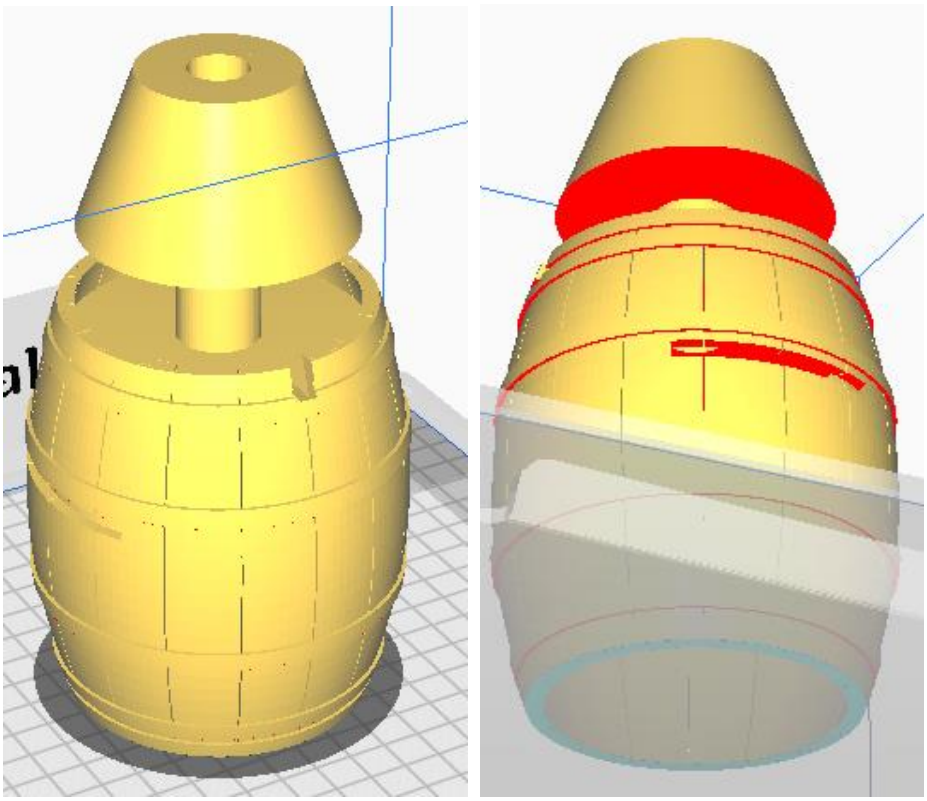
Speaking of starting modules, our second component was our “Sword in the Stone” module that we affectionately named “Arthur”, after the legendary king. The component is a simple pedestal hollowed out on the inside, enabling wires to be run through it. Arthur has a small slit in the top for the sword. Inside we have attached two aluminum sheets, separated by a piece of plastic, which when connected, complete a circuit. Thus, when the sword is placed in the slit, the aluminum touches both contacts and completes the circuit. This mechanism functions as our game reset, in our design when both swords are inserted into their stones, the game resets itself, and will not begin until both swords are removed. However, in practice our reset is only controlled by one sword.



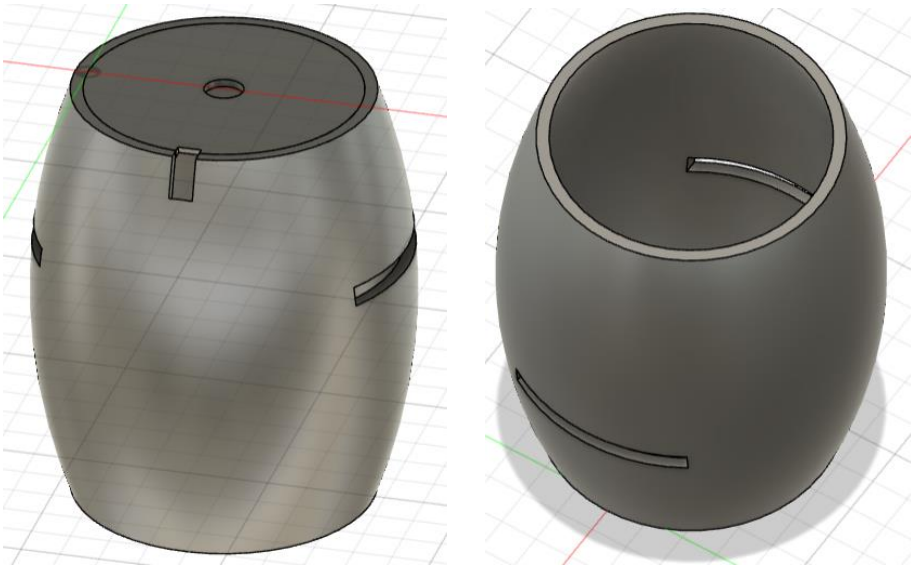
Arthur Fusion 360 Design

3D-Printed Arthur Module

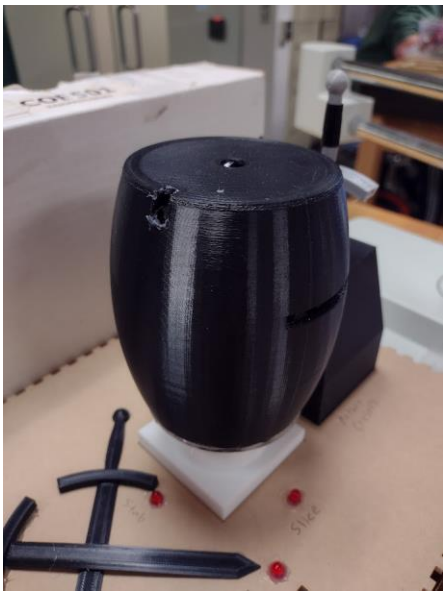
The second module is the barrel and stand, inspired by makeshift training dummies. Our original barrel design was a very elaborate design found on YouTube, but we were unable to 3D-print it, due to time constraints. Thus, we resulted to a simple barrel, with a hole in the bottom for wiring. There is a hole in the top of the barrel for the microphone, (not implemented), two slits on opposite sides for the buttons of the “Slice” mechanism, and a vertical slit near the top of the barrel for the “Stab” mechanism. We also 3D-printed a short stand for the barrel and laser cut a piece to attach the barrel to the stand.



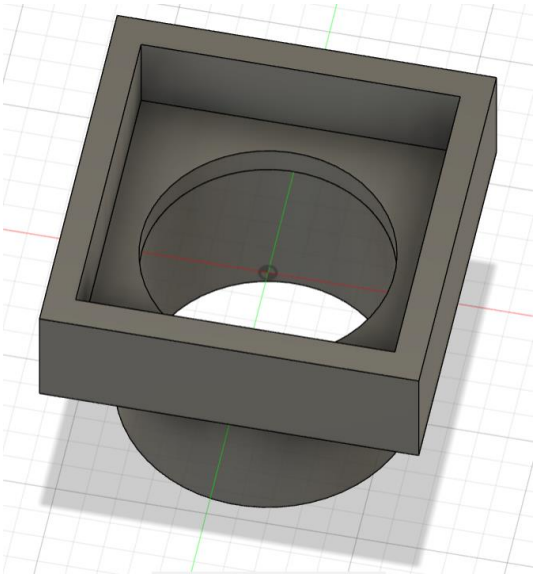
Original Barrel Fusion 360 Design (in Cura)



Barrel Fusion 360 Design



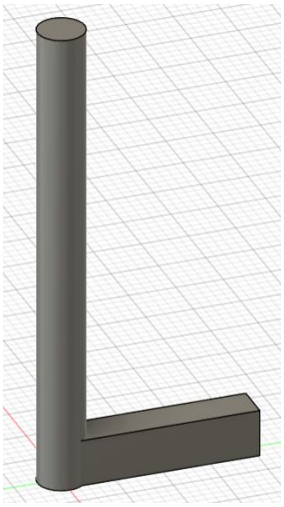
3D-Printed Barrel Module



Barrel Stand Fusion 360 Design

3D-Printed Barrel Stand

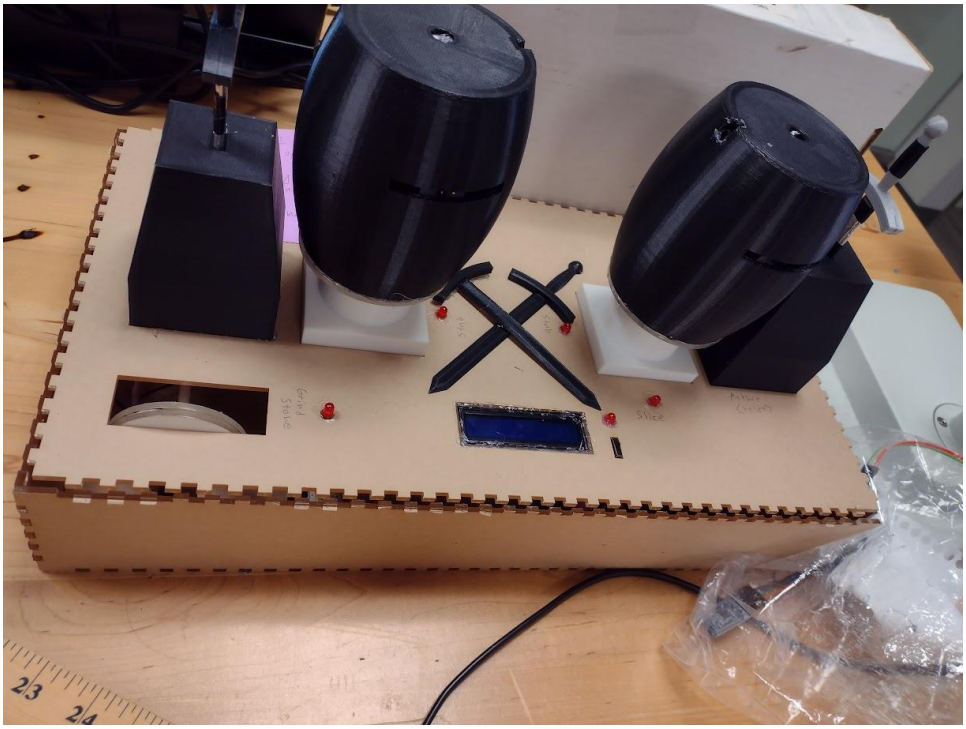
The last module we designed is the grindstone. The wheel is comprised of two laser-cut, acrylic circles glued together. The axel of the stone, which we glued to the wheel, was also designed in Fusion360. The axel is designed in an “L” shape, so when the wheel is spun, the lever of the axel hits the button, adhered to the stand of the grindstone.



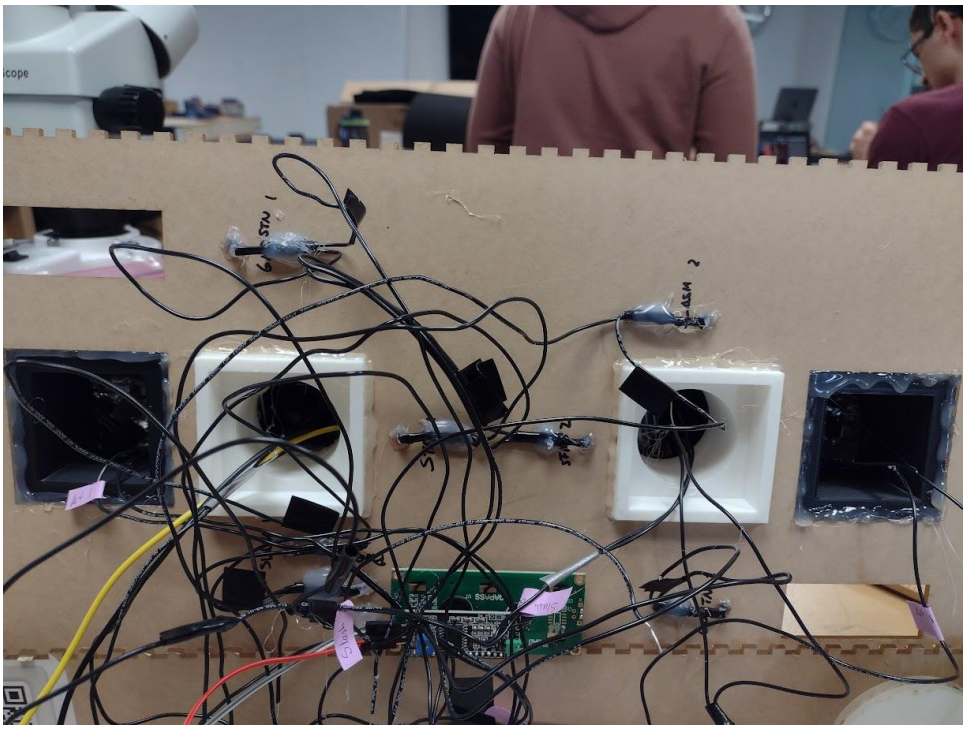
Axel Fusion 360 Design

Assembled Grindstone Module

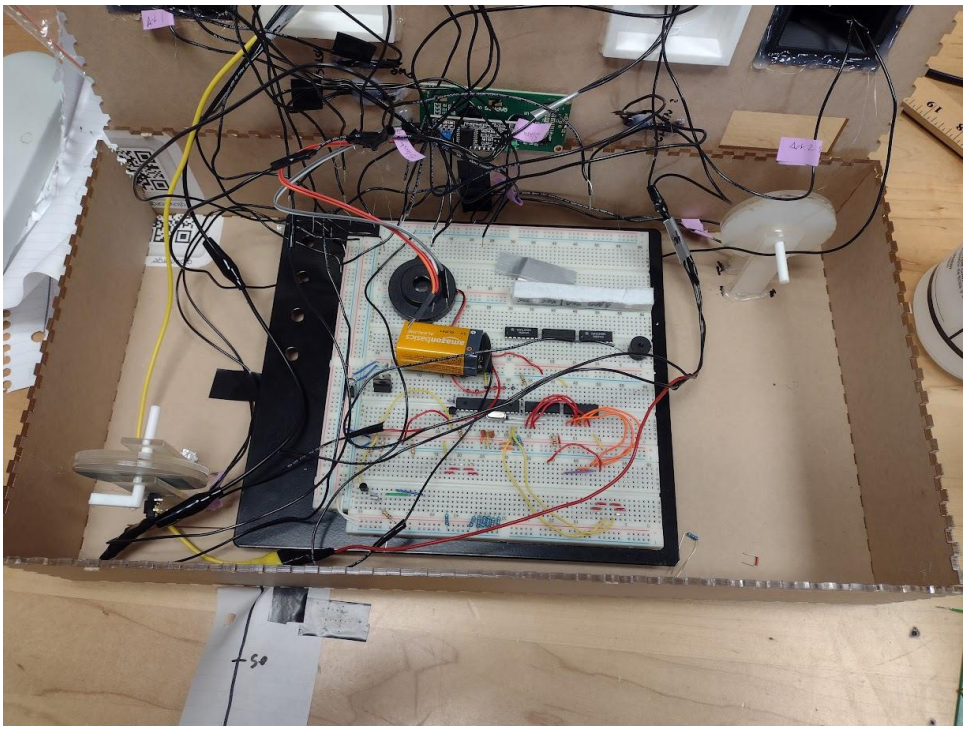
All our modules are attached to a 17” x 9” x 3” laser-cut box. The lid of the box has holes for the modules of both players as well as an LED for each module (except Arthur) as well as power. There is also a space for the LCD display. We also used two halves of the 3D-printed swords as a cool pattern across the top of the box. The Arthur and Barrel modules are glued to the lid of the box, while the Grindstone is glued to the bottom of the box.



Exterior of Enclosure

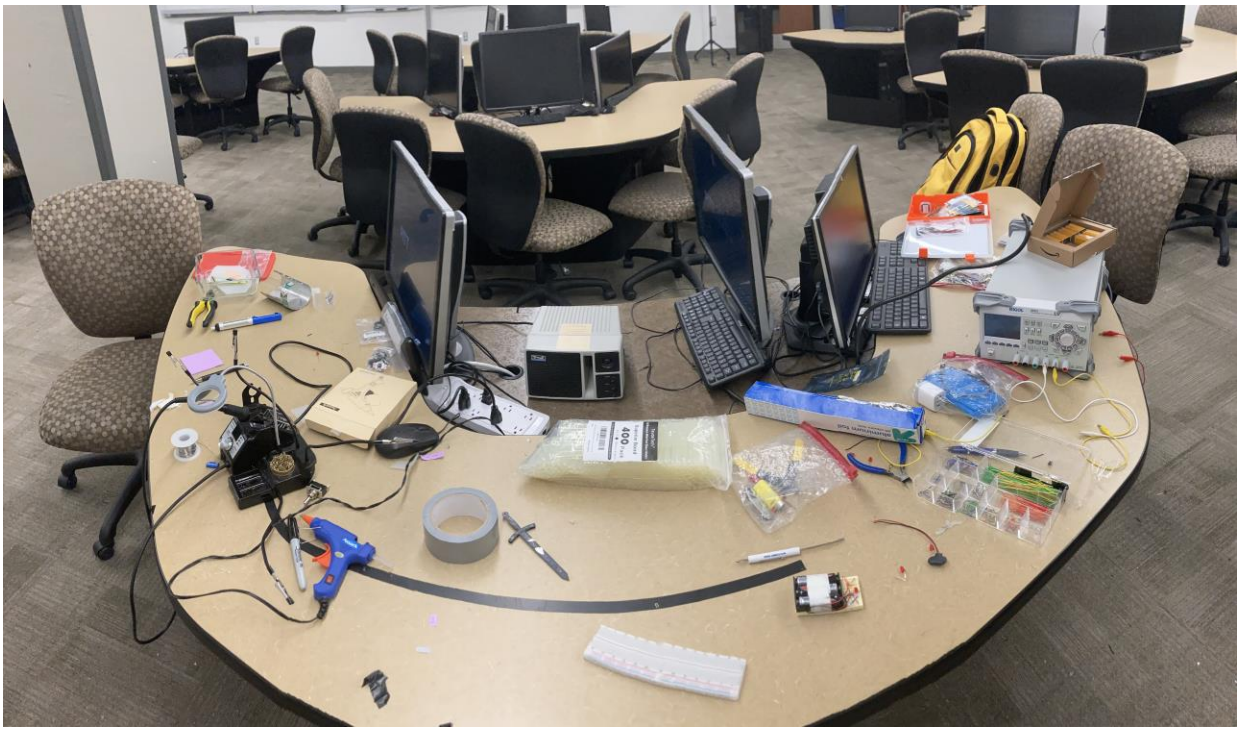


Interior Lid of Enclosure



Bottom of Enclosure

Assembly and System Integration



Aftermath

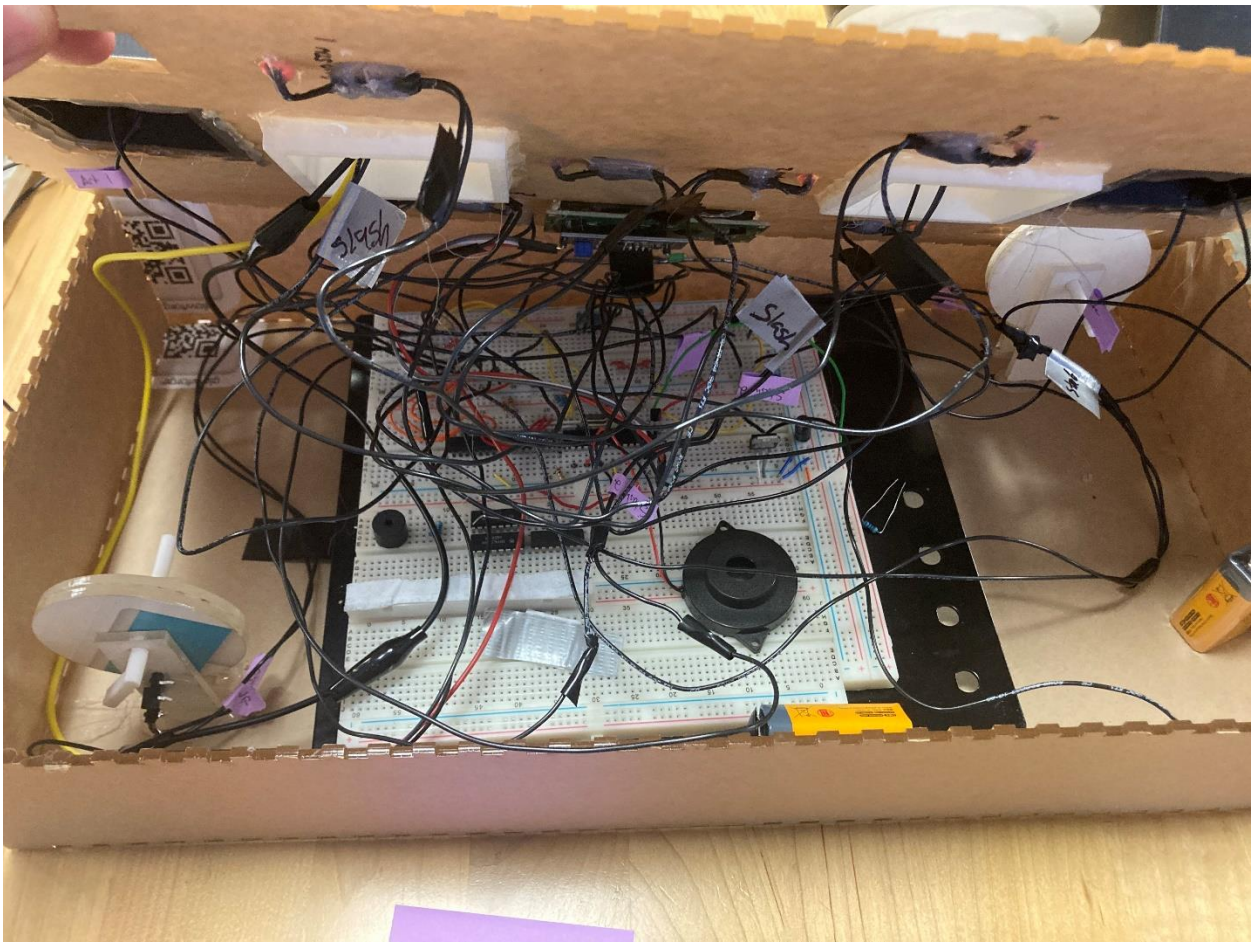
Methods Used

The main enclosure is laser cut, so it was able to be self-assembled. Most of our components were assembled using hot glue. All our LEDs and modules were hot glued to the main enclosure. A combination of hot glue and electrical tape was used to attach the leads to the metal contacts. We also had to improvise some mechanisms. When our original “Arthur” and “Stab” contacts were not functioning as intended, we developed a new strategy. Instead of lining metal along the sides of the slots, we attached two aluminum sheets inside the modules, separating them with scraps of plastic that Jonah recycled. This mechanism proved much more effective than our previous design.

Challenges

The biggest struggle during the assembly of our design resulted from our lack of a functioning PCB. Since we were running off a breadboard, simply getting all our wires attached was challenging (see photo below). Without a PCB, our wires quickly became a

mess, and sorting out the leads was time consuming. Many of our leads also disconnected from the modules, resulting in certain inputs not being registered and the game becoming unplayable. Another frustration from using a breadboard is that we had to use many resistors to regulate the voltage across the whole circuit to prevent certain components from failing. Aluminum foil proved to be an unreliable material. It did not adhere to solder well and would bend and tear. We often had to reinforce the aluminum by layering it or gluing it to plastic.



Consequences

Design Testing

Test Plan

Our test plan was mostly dependent on the breadboard prototyping to find issues with it and then redesigning our PCB to match changes that were made.

One road bump in the test plan was when we got to assembly and suddenly the exact same circuit was not acting the same when the buttons and foil plate connections had long leads coming off and we had to do a lot of troubleshooting and add a lot more Resistors. There was so much change we had to move off our planned Second-Generation PCB from SERC.

This was likely from the increased power draw and unstableness from the rat-nest of wires coming off the breadboards that were extremely sensitive to giving an improper connection. Talked about more in the 'Challenges' section.

More information on our testing in the section 'Design Verification'

Assembled Prototype

A photo of finished prototype is at the start of the report

Video of a Draw: https://youtu.be/OVM2WVpvJ4A

Video of Player2 Win: https://youtu.be/ccv0OVQIA6Y

Video of Player1 Win: https://youtu.be/BccRCO2uvBI
(For the last video: At this point of the project, the lead wires were so sensitive the board had to be upside down and sideways)

Unsuccess and Problem Solving

Despite our best efforts, we were unable to get *Duel-It* fully functioning properly for a live demonstration on Tuesday April 2. This was due to some loose wires in our power module, as well as in one of our barrels. After reconnecting the loose wires, we were able to get the game functioning at about 85% capacity. Everything worked except for a single input, (and our second reset). As mentioned above in our "Enclosure Design" Section, we were having some difficulty with our "Stab" and "Arthur" commands, and we created a new mechanism to solve the issue. The only other thing that doesn't function quite as intended

is that the game is supposed to start when both swords are removed from the “Arthur” modules using an AND gate. However, our AND gate does not work in the circuit, even though it works properly by itself, possibly because of a floating ground of some sort. As a result, the game’s start is triggered only by player 1’s sword.

What Doesn't Work and Why

Our only game input that currently does not function as intended is the “Stab” for player 1. This is due to loose wires in the barrel, which prevent the input from being registered. The foil in the barrel is sensitive to pressure, so it has moved and changed shape since it was originally placed in the barrel, dislodging the leads it was connected to. We already have a video of the entire game functioning as intended, so we did not want to risk further damaging the game by attempting to fix the loose wires.

Budget and Cost Analysis

The total amount of money we spent ordering parts in our BOMs was $34.04, and the total cost of our 5 PCBs is estimated to be about $69.50. This brings our total amount spent to $103.54. This is a reasonable amount of money, especially for developing a strong prototype for a toy. If team Upsilon was working for an actual company, this would certainly be in the development budget. We did end up spending some unnecessary money on microphones and toggle switches that we were unable to implement into our design, but everything else was incorporated to the best of our ability.

Team

Roles

Jonah: Software Development, Initial Circuit design, Circuit Prototyping, Arts and Crafts

John: PCB Design, Arts and Crafts

Jay: Enclosure Design, 3D-Printing, Arts and Crafts

Everyone helped with assembling the final product.

How We Worked Together

We kept in constant contact with each other over a text group chat. Very early we made the general schematic and idea and made plans with each other on splitting up work, when we need to meet in class, and when we need to all meetup outside of class and have a work session.

Timeline - Milestones

- Feb 12 – Project chosen, Duel-it selected as design
- Feb 13 to March 3 – Initial Prototype complete, Breadboard using buttons constructed
- March 5 – PCB Files submitted
- March 7 – Project update presentation
- March 4 to March 26 – Faults found in initial design, improved prototype constructed
- March 21 – Enclosure Design Started
- March 26 – Updated PCB ordered, put on rush order
- March 27 – Final bug found in design, prototype updated to fix problem
- March 28 – Enclosure Construction Started
- April 1 – Enclosure Construction Completed, Project fully constructed using fully functional breadboard prototype in place of PCB
- April 2 – Completed Design Presented in class, enclosure issues result in limited functionality
- April 7 – Design Report Submitted

Summary, Conclusions and Future Work

Despite the hiccups posed by material choice and bugs in our PCBs, we are extremely happy with how our project turned out. In the end we were able to deliver a functional project, though the small hiccups resulted in it occasionally failing. Our team worked and communicated well with each other and that is reflected in the scope and functionality of our final design. We knew that making a two-player game was farily ambitious, but Team Upsilon rose to the challenge. We all learned more about the design process through both our successes and failures of managing the timeline for a deliverable.

And we also learned how to make use of many of the resources that SERC and the makerspace have to offer, such as in-house PCB design, designing around Arduino I/Os, 3D-printing, and laser cutting. We also became immersed in new software such as Fusion 360 and Cura to manufacture our parts. We will carry these skills into Junior Design Project 3, our own personal projects, and beyond.

Changes We Would Make

The largest change we would make to this project is the material used in modules that required a sword completing a circuit. We predicted that aluminum foil would be a good conductor of electricity, and selected it due to its large availability. The foil did prove to conduct as well as we hoped, but was extremely delicate and would tear, bend, and crinkle when used anywhere in the design. This resulted in the 'Stab' and 'Arthur' inputs being extremely inconsistent.

Knowing what we know now, we would use a more rigid material, such as copper tape, in place of aluminum foil, which would likely resolve many of the issues we had upon enclosure construction.

The other issue with our PCB was from not getting to total assembly faster and finding the issues we found sooner. There was a small period between getting our ordered components in, and when we could've ordered another PCB. This was actually when we ordered our second iteration. However, due to other classes and commitments we could not get to a state that early in that point in time, forcing us to use the Breadboard and have unstable lead connections.

Other than that, our design worked almost as intended and we are proud of our achievement.