Electronic Battleship

School of Engineering and Applied Sciences
Harvard University

Cambridge MA.

Description: Making a fully functional electronic version of the board game Battleship with NeoTrellis inputs and LED display outputs.

By: Aida Baradari, Lana Wagner, Arseniy Zvyagintsev

Class: Engineering Sciences 50
Instructor: Christopher Lombardo

May 2022

ABSTRACT

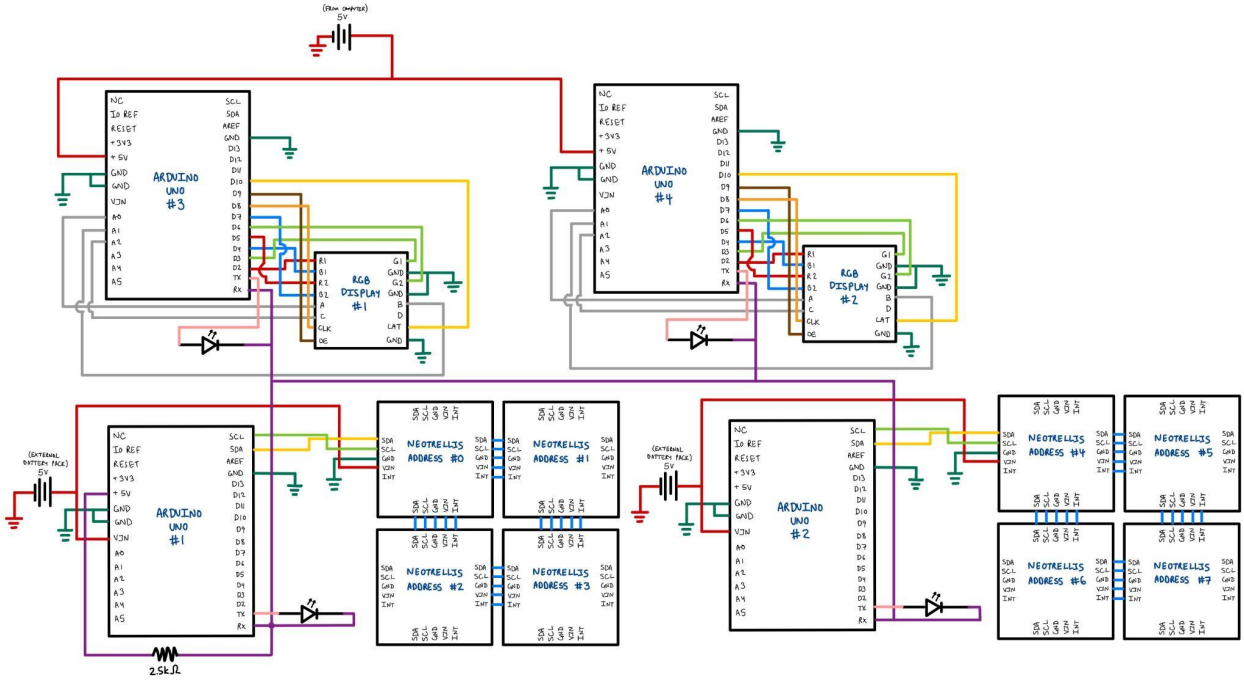
With our project, the goal was to make a fully playable version of the popular board game Battleship with electronic components. We had three main goals and stages for the game's functionality: having each player place their ships, having each player take turns firing to guess a location of their opponent's ships, and having that guessed location reacted to accordingly by the game with a "hit" or "miss" cue. There was also motivation to figure out a simple and user-friendly representation of the Battleship game given the limited functionalities of the electronic components we were given.

INTRODUCTION

Our team wanted to make a project that was interactive and playable, and that built upon the concepts we explored in both lecture and lab. While initially struggling with generating a concept, we eventually landed on making a version of the board game Battleship, which was one of our personal favorites. Our goal was to make a product that was simple and sleek and fun, and translated the defining physical aspects of Battleship into an engaging experience.

DESIGN

Going into the design process, we knew we had to find hardware representations of three main aspects of the game. Since Battleship is played on a coordinate grid, and each player puts plastic ships on the grid at the start of the game, we decided there had to be a way to represent this input of an array of potential locations through hardware. After placing ships, players then guess the location of their opponents' ships, so we determined that we needed a physical way of representing these guesses. Once a player guesses, their guess will either be a hit or miss, so we needed a way to represent and store the current and prior guesses in some physical manner that would be clear and visible to the player. Initially, we considered making physical, 3D printed custom Battleship pieces, and creating an array of photoresistors to detect where the pieces were placed. However, after looking into the implementation of this concept, we determined that it would be too logistically challenging to implement and parse information from the array, and buying enough photoresistors would take up most of the allotted budget. Before starting our final project, we had recently used NeoPixels in lab, and after some exploration on the Adafruit website, we found an acceptable hardware representation for the input: the NeoTrellis, a premade 4x4 board of buttons that have LEDs, and are able to be tiled together with I^2C. For our location-storing mechanism and user interface, we decided on a compatible 32x16 RGB LED Matrix from Adafruit. We needed one LED display and one 8x8 NeoTrellis array (4 individual NeoTrellises tiled and soldered together) per player. Once we decided on the components we needed, we designed housing for the components using Autodesk Fusion 360 (Figure 1, 2). The pieces of the housing were then laser cut and assembled with bolts and T-slot joints.



Above: Circuit schematic for the electronic Battleship game

PARTS LIST

Name	Part #	Link	Quantity	Item Price
32x16 RGB LED Matrix Panel	420	https://www.adafruit.com/product/420	2	$24.95
4x4 NeoTrellis RGB Driver PCB	3954	https://www.adafruit.com/product/3954	8	$12.50
4x4 Silicone Elastomer Button Keypad	1611	https://www.adafruit.com/product/1611	8	$4.95
Half Breadboard	64	https://www.adafruit.com/product/64	1	$5.00
2.2kΩ Resistor (25pcs)	2782	https://www.adafruit.com/product/2782	1	$0.75
Arduino Uno	A000066	http://store-usa.arduino.cc/products/arduino-uno-rev3	4	$24.20
3mm Red LED	2460-L314HD-ND	https://www.digikey.com/en/products/detail/american-opto-plus-led/L314HD/13165101	4	$0.22
Female to Male Jumper Wires (80pcs)		https://www.amazon.com/dp/B01L5UKAPI?ref_=cm_sw_r_cp_ud_dp_60FE36E37QVQCV0GCV6K	1	$6.49
24”x12” Sheet ⅛” Acrylic		Makerspace supplies	1	

24”x12” Sheet ¼” Acrylic		Makerspace supplies	1	
12”x12” Sheet 1/16” Acrylic		Makerspace supplies	1	
4-40 Bolts (Assorted)		Makerspace supplies	72	
4-40 Hex Nuts		Makerspace supplies	72	
			TOTAL:	**$299.42**

PROJECT IMPLEMENTATION

First, we began by figuring out how to work a singular NeoTrellis, and how to control it through code. Once we got one NeoTrellis to work, we tiled the NeoTrellises together by soldering the side finger pads of each board together. To make sure we could control each individual NeoTrellis in code, we gave each board a unique I^2C address by putting different amounts of solder in 5 unique pads that would give each board a binary address up to 32. To test whether each NeoTrellis was functional, we ran example Arduino code from the Adafruit library.

After getting the example code to successfully work, we started writing our pseudo code. The overarching plan was to give the players thirty seconds to input the positions of their ships, and store the input on the LED display afterwards. Once this initial stage was finished, the game needed to prompt the player to choose a cell to attack on the NeoTrellis board, and check whether that cell hit one of their opponent’s ships. Then, it would be the other player’s turn to choose a cell and the same logic would apply. Finally, if all ships are hit, the winner’s screen should turn green and the loser's screen would turn red.

To implement the game rules into our code, we split up these stages into multiple functions to make the code more efficient and logical. To store the information of the positions of the ships, we created an 8x8 array of rows and columns. If there was a ship at a given position, we would set that spot in the array to one, otherwise it would be zero.

When it came time to check whether a player’s guess was a hit, the Arduino of the input NeoTrellis pad would communicate with the Arduino of the opponent’s display to check whether the positions match. If they do, the cell on the opponent’s LED display and the cell on the NeoTrellis pad would both turn red. Otherwise, the cell on the opponent’s LED display would turn off and the corresponding cell on the NeoTrellis pad would turn white.

As we were working on assembling the electronics and writing code, we decided to make a housing for all the hardware. The main constraints we kept in mind while designing this housing were that the two players needed to be separated and unable to see each other’s NeoTrellis keypads, and that the pieces must be assembled with simple hardware and keep our electronic

components in place. The housing was designed with CAD using Autodesk Fusion 360. We decided on making a base box that had the necessary holes for external wires from a computer and power source to plug into the Arduinos and displays, and that had properly sized holes on its top face to hold the NeoTrellises. One issue we ran into while using the NeoTrellises was that the silicone keypads would come detached from the NeoTrellises during use, so we designed a piece to hold the keypads in place on top of the NeoTrellises. In the middle of the base, we made a box to hold the displays in place, and orient them so each player could see their own display. We used T-slot joints to join the faces of our housing together, and checked in Fusion whether the faces would fit together (Figure 1). Once we finished using CAD to design the housing for the electrical components, we created DXF files for each component we needed to cut, and uploaded these DXF files to the laser cutting software. We cut the base out of 1/4" acrylic, the display box out of 1/8" acrylic, and the keypad holder and NeoTrellis base plate out of 1/16” acrylic. After these components were cut, we transferred our electronics and hardware to the housing, and assembled it using nuts and bolts.

Along the way, we ran into three main problems. First, we figured out early on that the Arduino MKRZeros were incompatible with the electrical components we planned to use. However, the lab had a low stock of Arduino Unos, which were the backup option. For future projects, we would likely figure out in advance which microcontrollers are compatible with our given specifications and prepare accordingly. Since the displays had 16 output wires each, we needed an Arduino Uno to control each one, and needed separate Arduinos to control the NeoTrellises, which brought us to a total of four Arduino Unos.

Second, since we needed to control four total Arduino Unos, we had issues maintaining communication between the Arduinos. We used the RX and TX pins to communicate between Arduinos, so we had to write code to make sure information was being successfully passed and received between the Arduinos. All four Arduinos shared one channel of communication, and sometimes the channel was overloaded which forced us to send the same piece of information several times in a row.

Finally, the power supply for the NeoTrellis was interfering with the channel of communication and leading to an undefined behavior, and causing our Arduinos and code to run out of sync. To fix this issue, we had to add an additional power source to the system. After we implemented these solutions, our Battleship game was able to run smoothly and reliably.

TEAM MANAGEMENT

Due to their respective backgrounds and skills, Aida and Arseniy were responsible for the code and the hardware, while Lana worked on the housing. All team members worked on assembly, wiring, and soldering.

OUTLOOK AND POSSIBLE IMPROVEMENTS

We believe that given our budget and resources, we were able to successfully implement all our goals for the project, and get a reliably working product. If we had even more time and resources to work, we could have tried to implement other games on the hardware setup, such as Chess or Checkers. We could have also improved performance by upgrading our Arduino Unos to microcontrollers with improved processing power and communication skills, such as a Raspberry Pi, but this would not improve the general functionality of the game by much.

ACKNOWLEDGEMENTS

We would like to thank Professor Loncar and Lombardo for giving great advice and support about how to implement our project. We would also like to thank the Active Learning Lab staff for helping us get the electronics parts we needed. Finally, we would like to thank our project TF, Molly Bosworth, for helping us troubleshoot our electronics and code, and encouraging us along the way.

DISCLAIMER

We consent to having our report, code, photos, and videos shared.

REFERENCES

https://learn.adafruit.com/32x16-32x32-rgb-led-matrix/
https://learn.adafruit.com/adafruit-neotrellis
https://fablab.ruc.dk/how-to-connect-multiple-arduino-boards/

APPENDIX

Figure 1: Full CAD Assembly

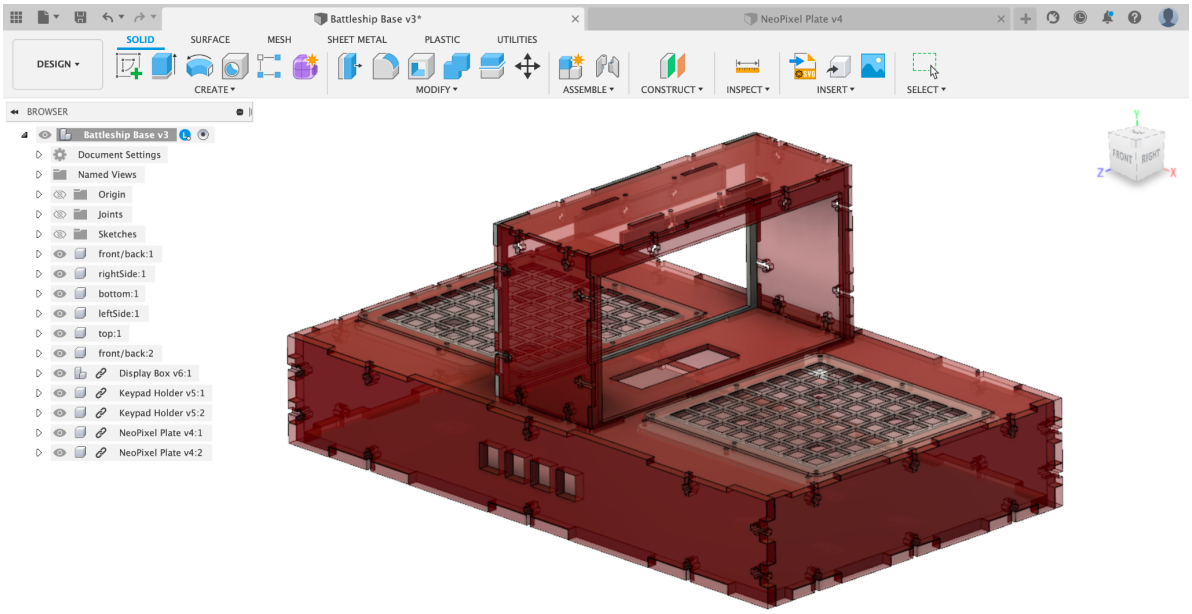
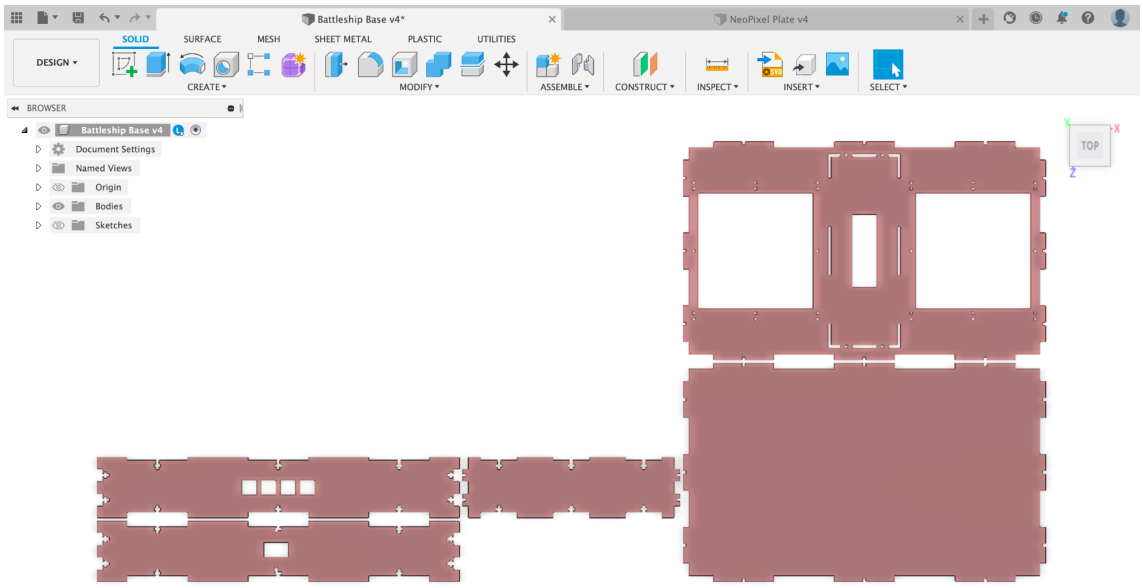


Figure 2: Top View of Battleship Base Plates



To view the project's code, please refer to the files in the .zip archive. Read the READ.md file for more specific information about the code.