E-thello

A Research Paper submitted to the Department of Electrical and Computer Engineering

Presented to the Faculty of the School of Engineering and Applied Science
University of Virginia • Charlottesville, Virginia

In Partial Fulfillment of the Requirements for the Degree
Bachelor of Science, School of Engineering

Ziang Jiao
Spring 2024

On my honor as a University Student, I have neither given nor received unauthorized aid on this assignment as defined by the Honor Guidelines for Thesis-Related Assignments

Advisor
Adam Barnes, Department of Electrical and Computer Engineering

Statement of Work

Brian contributed to the setup and testing of the microcontroller and LED strips. He wrote code to control individually addressable LEDs using existing libraries before creating a proof-of-concept circuit with an external power bank, Raspberry Pico WH, and 8x8 LED matrix to demonstrate this functionality. He also helped to make the switch to the Raspberry Pi, including installing the OS, porting over existing code, and adjusting pin connections with the power bank and LED panel. He then updated the push button microcontroller code and wired the push button control PCB board to the Raspberry Pi.

He also assisted with cutting and soldering the LED strips and updating the code controlling the LEDs. He then worked with everyone else to perform system testing, which verified the functionality of displaying the board state on the LED strips and detecting push button presses. Lastly, he also served as the reimbursing agent for parts ordering and kept track of the team's expenses and budget.

Dennis contributed to researching, writing, and optimizing the advanced AI Othello program that controlled the computer opponent's moves. In addition, he wrote another program that prepared and ran the actual Othello game and combined it with the AI player. He also set up and integrated the LCD into the rest of the system and wrote the GUI menu allowing players to configure the game. Furthermore, he worked with Lawrence on the designs and redesigns of the physical housing and ensured that everything was accounted for in the models before 3D printing.

He led the assembly effort, adjusting (sanding, gluing, etc.) the various components and fitting everything together; for example, he attached all 64 pieces to the LED strips, making sure that they were spaced such that they all fit through the grid of holes smoothly. He carefully routed all of the wires connecting to the Raspberry Pi such that they did not obstruct the buttons without damaging the LED strip solder joints, which was one of the main roadblocks we had to overcome. He also bent each of the wires connecting the LED strips together such that they did not obstruct the outer housing, which was very precarious because the solder joints were

small and easily broken. He resoldered and/or replaced all of the LED strips that broke or had broken solder joints.

Lawrence contributed to the design, assembly, and testing of the mechanical components and othello housing. He designed the outer shell and inner shelf to enclose all electronic components and removable panels to access the power supply and LCD. He worked in conjunction with Dennis and Ziang to design and integrate the Othello pieces with the LED strips. Then he printed out prototypes to verify that the pieces were correctly sized. Afterwards, he and Ziang fabricated all the designs and assembled the components to ensure a proper fit. He would redesign if changes were made and continuously prototype and test.

Ziang contributed to the design, assembly, and testing of the push button control PCB board. He designed the PCB on Multisim and managed the component placement on Ultiboard to align with the dimensions of the LED strip used. He customized the footprints on Ultiboard for the specific push button used in the project. He soldered all the components onto the PCB in the assembly stage and wrote the prototype for the microcontroller code to accurately locate the activated push button in Python. He tested the performance of the 8x8 push button arrays before and after assembly.

He also received 3D printing training and helped print the body of the Othello housing in the testing and assembly stages. He was responsible for laser cutting the translucent acrylic tops for the push buttons. He also assisted with the final assembly of the Othello board.

Table of Content

Table of Figures

Abstract

The E-thello board is an electronic Othello board that allows users to play against a built-in artificial intelligence program with multiple difficulty levels. What makes this stand out from current, standard Othello boards is that it combines the traditional physical feel of the game with online Othello features such as move highlighting, automatic board updating, and more, making it great for both beginners wanting to learn the mechanics of the game and for advanced players looking to hone their skills. This project involves a sophisticated program that has two purposes: to play as the opponent and to determine the user's valid moves. The software is integrated with the main PCB, which consists of an 8x8 grid of pushbuttons, as well as eight LED strips arranged to form an accompanying 8x8 grid. The LEDs are controlled through the Raspberry Pi, which uses pre-existing Neopixel libraries that simplify interfacing with the LEDs. In addition, the buttons are embedded onto the PCB board as a push-button matrix; a program is written into Raspberry Pi to locate the activated button by iterating through the rows and columns of the matrix. Lastly, all of these components are assembled and housed in a 3D-printed enclosure.

Background

Othello is a classic two-player board game played on an 8x8 grid typically using discs that are colored black on one side and white on the other. Players take turns placing one disc on the board; they must make a move that results in flipping at least one of their opponent's discs. To do this, the player must place a disc such that a row of the opponent's pieces is "sandwiched" between one of the player's previous discs and the newly-placed one. Then, the opponent's sandwiched discs will be flipped to the player's color. The game ends when the board is filled or no legal moves remain for either player, with the winner being decided by disc count.

Like many other popular board games, Othello can also be played online, which has advantages, such as legal move highlighting and automatic updating of the board state, making it much more friendly for beginners. In addition, many online versions of Othello also offer the option to play against a computer if there is not a second person to play against. However, people who prefer playing on a physical board or those without internet access miss out on these convenient features and cannot play at all if there is not a second player. These problems led us to incorporate all of these features into a self-contained physical board as described in the abstract.

While there have not been any past prototypes based on the concept of a physical light-up Othello board that have been documented online, there have been hobby projects for other board games like chess. A prototype for a light-up chessboard [1] was the main inspiration for this project, as it added legal move highlighting for any piece the user chose. Another example was an automatic chessboard with an AI opponent that would mechanically move a piece on its turn using a 2-axis movement system, much like the one in mechanical printers [2]. Aside from these projects being only for chess, they also do not encapsulate all of the convenient features mentioned earlier into a single, self-contained board. Furthermore, they track the board state with physical pieces, which is not necessary in our implementation of Othello.

There are numerous commercially available LED chessboards; however, most of these lack the aforementioned functionalities (legal move highlighting, AI opponent, etc.). One exception to this is the ChessUp board, which is a physical board that offers move highlighting and the ability to play against real players anywhere online by indicating their moves on the board [3].

To our knowledge, the idea we present is still novel, as there are not any other physical LED Othello boards with all of the functionalities that ours offers. This, along with the fact that the board is entirely self-contained, enables it to be portable and independent from the Internet, making it suitable in areas with no cellular service or power outlets. It also requires minimal user setup, utilizing a rechargeable power bank as the power source. Additionally, the board does not require physical pieces, as the board state is represented entirely by the state (on/off) and color of the LEDs. This makes setting up an Othello game quick and hassle-free and removes the possibility of pieces being lost. Lastly, since the difficulty of the AI opponent is adjustable, this product is great for players in any skill range.

This project utilized knowledge from several classes in the engineering curriculum: Intro to Engineering; Software Development; ECE fundamentals I, II, and III; Embedded Systems; and Artificial Intelligence. We decided to delegate the task of developing the AI Othello program to Dennis, who had the strongest background in developing these game models. In addition, all group members were comfortable with embedded programming in C/C++ with the MSP430 launchpad. Although this specific microcontroller was not used for the project, it provided a good foundation to learn how to program with the Raspberry Pi. All group members also had previous experience with 3D-printing and Solidworks from Intro to Engineering, which proved helpful for designing the board housing. In addition, everyone had experience with ECE concepts and skills, such as PCB design, power supplies, and circuit design, which were necessary skills for the core of the project. These skills and experiences

helped the team successfully implement the Othello AI Algorithm and push-button PCB in addition to the general system requirement.

Societal Impact Constraints

Safety - Unlike the traditional Othello board game, our product eliminates the use of small physical pieces, making it a safer choice, especially for young children who may face choking hazards with traditional game pieces. However, being an electronic device powered by a 5V/2A supply, it does introduce a potential electric shock risk. To mitigate this concern, our design incorporates key safety features. Specifically, we ensured that no electronic components are exposed to users, and we incorporated a protective, insulating outer housing and chose a power bank that aligns with commercial safety standards, significantly reducing the risk of electric shock.

Environment - The major environmental impact of our board game primarily pertains to waste generation, particularly during the production of the 3D-printed board housing, where any deviations from design specifications might necessitate disposal. To minimize this environmental footprint, we prioritized recycling, with a focus on repurposing materials for new 3D-printing filament or industrial use. However, handling electronic components introduces complexities, as unused ones can be stored for future students' use, but malfunctioning ones are challenging to recycle due to their intricate composition. To mitigate this issue, we emphasized careful design and testing to prevent component damage or burning, reducing waste and our overall environmental impact.

Culture - Through the incorporation of technology and AI into a timeless board game, our board represents a pioneering approach to the traditional setup, likely sparking engaging discussions within the community of devoted game enthusiasts deeply rooted in the cultural significance of this classic. These aficionados often seek to uphold the game's fundamental essence while embracing innovations that enhance the overall experience. An example of such innovation is our automated piece-flipping feature, driven by sophisticated algorithms, which notably streamlines gameplay, alleviating the players' manual workload. We anticipate that this progressive addition will garner appreciation and recognition from the Othello community, uniting tradition and modernity in a harmonious cultural context.

Welfare - Our electronic game board offers significant advantages, serving a dual purpose for both education and solo gameplay, catering to Othello enthusiasts who may find themselves without a partner. Utilizing advanced algorithms, it boasts the ability to

meticulously analyze the board's configuration, illuminating valid move options for players. This feature proves invaluable for teaching children or newcomers the intricacies of the game, simplifying it compared to the traditional setup. Notably, it eliminates the need for players to manually reset and rearrange the board after each game, streamlining the experience. Additionally, it automates piece flipping, further enhancing user convenience. With AI-driven support for single-player mode, the need for a second player becomes obsolete, addressing the concerns of elderly individuals living alone and dedicated Othello enthusiasts who seek continuous practice to hone their skills. This innovation extends a helping hand to numerous user groups who were previously marginalized by the classic game due to various constraints.

Economics - Our product primarily targets entertainment and educational purposes, necessitating affordability for users, which is crucial for the sustained competitiveness of our electronic Othello in the market. During our Capstone phase, our emphasis was on prototyping rather than production control. However, the transition to mass production demands a shift towards the most cost-effective and labor-efficient manufacturing methods. For instance, while 3D printing sufficed during prototyping, mass production would benefit from the adoption of more scalable techniques such as plastic injection molding. Additionally, optimizing the mechanical design of the product becomes imperative to streamline the assembly process and minimize production costs.

Physical Constraints

Usability - The E-thello board is an educational device designed to cater to individuals of diverse ages and skill levels. Its exterior design emphasizes simplicity, ensuring that instructions and moves are easily comprehensible and user-friendly. Furthermore, the integrated AI is engineered to be adjustable but very tough at the highest difficulty, thereby enhancing the educational value and enjoyability of the product. The system is also crafted to uphold high standards of quality and durability, guaranteeing a prolonged lifespan for users.

Because of this, we had to take all of these traits into account when designing our E-thello board to be user-friendly. We directly addressed this by incorporating an LCD that allows users to change the difficulty, undo moves, and start the game via scroll and select buttons. This constrained our design as we had to adjust our CAD models, Othello program, and wiring to accommodate the display.

Part Availability - The product consists of several key components, including a custom-designed outer housing for the 8-inch x 8-inch x 3-inch (L x W x H) Othello board,

which we produced using 3D-printing technology. Additionally, it features a 6.95-inch x 6.95-inch PCB; an LCD; LED strips; push-buttons; a powerbank; a Raspberry Pi; and specially-designed, 3D-printed pieces. The availability of these components was constrained by the electronic suppliers' offerings and the maximum printing dimensions of the University's 3D printers. As such, we had to account for both the availability and cost of these components for the weekly parts orders.

Manufacturability - The production of the Othello board was straightforward, barring some constraints, as all necessary PCB, LED, and electronic components were readily available from established electronics suppliers. The 3D-printed outer housing and buttons were manufactured at the 3D-printing lab located on the third floor of Clemons Library. Additionally, the Raspberry Pico WH we initially used was easily obtainable from DigiKey, and one of our group members already had a spare Raspberry Pi 3b+ that we pivoted to, which cut down on our budget.

The size of our push-button control PCB was limited to 60 square inches, which also affected the overall size of our Othello board. This was a limitation that we had to account for when designing the outer shell and when arranging all the internal components during the final assembly.

In addition, a major roadblock we encountered halfway through the project was that the Raspberry Pico WH lacked the processing power, RAM, and storage needed to run our Othello AI algorithm. As a result, we ended up switching to a Raspberry Pi 3b+, which required porting all of our existing code to Python 3 and adjusting the wiring to account for the different pin connections.

Another constraint imposed was the waiting period between submitting parts orders and actually receiving the parts, which usually took around 5 to 6 days. As such, we had to designate a team member to order more time-sensitive parts separately through Amazon Prime and get reimbursed later.

There were several constraints for 3D printing, including dimension constraints, machine availability, print time, and filament color. A majority of the 3D-printing machines were accessible to the public and had to be shared. The machine that was used to print the larger components (i.e. outer shell, inner shelf) was a special request printer, so we had to get the design approved by management. The printing lab was only open for a limited period, so our printing durations had to be under six hours. This required us to break up the larger components and design the resulting pieces such that they would fit together easily and securely

during assembly. Lastly, we could not choose which colors the components would be printed in unless the current filament ran out.

Cost - The project was allocated a budget of $500, with the majority of expenses incurred in procuring essential components for the final design, including the PCB, RGB LED strips, and push-buttons. In addition, some expenses also went into parts used for initial prototypes, such as the 8x8 LED panel, as well as tools and materials like super glue and black spray paint. Notably, there was minimal expenditure for the physical board housing, which was provided as a free University amenity – we coordinated with the 3D-printing lab, so the printing services and materials were provided at no cost after we completed the required formal training. This arrangement left us with a substantial surplus in our budget, which was allocated toward extra supplies such as another roll of LED strips.

Tools Utilized- A comprehensive software toolkit is used throughout the project:

The physical design of the E-thello board was completed in ***Fusion*** and then imported into an .stl file for use with 3D-printing software.

Thonny was the IDE we used to directly run programs in Python 3 on the Raspberry Pi. Once we integrated the AI Othello program with the Raspberry Pi, it was also used for program adjustments as well as running smaller scripts to verify the functionality of the LED strips after soldering them.

Makerbot Desktop and ***Ultimaker Cura*** served as the software platforms for 3D printing, importing .stl files, and slicing them for the printer's use.

In the development process of the push-button PCB design prototype, ***Multisim*** was utilized for construction, and subsequently, the design was transferred to ***Ultiboard*** for physical configurations aligning with specific dimensions.

Python 3 was the programming language employed for both the AI and control of the PCB, LCD, and RGB LED strips. These programs were executed within the ***Raspberry Pi OS*** to undergo thorough testing on the microprocessor.

Rhino was the software we utilized for precision laser cutting of the acrylic board, which was used for the translucent tops of the pieces.

External Standards

This product clearly included electronic components and electrical connections, and these needed to be regulated to prevent hazards. IEC 60950 is an international safety standard regarding information technology, electrical, and electronic equipment. Some notable

regulations we followed were measures to protect against electrical shock and other hazards; requirements for insulation, wiring, and grounding; and protection against overcurrent and fault conditions. The materials used in this project were also subject to the same standards, which include mechanical and electrical construction, requirements for accessible parts, and safety markings and labels information.

The PCB for the Othello board was housed in a 3D-printed enclosure, which was subject to regulations pertaining to the electronics within the enclosure. NEMA standards provide a rating that defines the environments for electrical enclosures, outlining the enclosure's ability to withstand certain conditions. Our housing fell under the NEMA 1 rating, which states that enclosures are constructed for indoor use and provide protection against human contact with live electrical parts. The housing also protected the equipment from falling debris such as dirt [10][11].

A major requirement for this project was the PCB that integrated the LEDs and buttons. The IPC-2221 standard provided guidelines and best practices for the design of printed circuit boards. Some standards that applied to this project include design guidelines such as layout, component placement, routing of traces, and grounding techniques. It also covered mechanical design considerations, including board thickness, mounting holes, and mechanical fasteners, to support the physical integrity of the PCB. [12]

An important aspect of the board was the multicolored LEDs used to indicate possible moves and player pieces. UL 8750 outlines safety requirements for LED equipment used in lighting products. Some considerations were electrical safety to prevent electrical shock and hazards. It includes provisions for insulation, wiring, grounding, and protection against overcurrent and fault conditions.

The power system PCB utilized batteries to drive the LEDs and buttons. The IEC EN60086-1 and IEC EN60086-2 standards are intended to standardize primary batteries with respect to dimensions, nomenclature, terminal configurations, markings, test methods, performance, safety and environmental aspects.

Intellectual Property Issues

We identified three patented products that involve materials similar to those in our project. The first patent describes a Reversi game device (US20030027615A1), which introduces an Othello game device capable of displaying board states using a screen and detecting whether a piece is trapped or not. The second patent covers a multi-touch pad

controller (US9076419B2) that integrates translucent push buttons with embedded RGB LED functionality. Pressing the button grants control to the system, and the embedded LED offers interactive feedback to user input. The third patent pertains to a keyboard (US5575576A) that shares a comparable mechanism with our product, relying on the states of push buttons for user input.

Our project introduces innovative features that set it apart from these patented products. In comparison to the Reversi game device, which primarily focuses on converting physical board setups into electronic games, our board goes beyond mere conversion. We intricately integrated software, hardware, and mechanical elements to create an electronic Othello board. This integration facilitated the incorporation of an AI opponent, automated board state updates, and interactive LED push buttons. The Reversi game device patent does not delve into this comprehensive approach to implementation and integration.

Regarding the multi-touch pad controller, although our project may share a similar appearance, its internal mechanism is entirely distinct from the patented technology. The patent employs pressure and location-sensitive sensors, whereas our design relies on a mechanical pushbutton matrix and an ongoing scanning process by the microcontroller to locate these buttons. This fundamental difference in technology distinguishes our project from the patented multi-touch pad controller.

Similarly, in comparison to the keyboard patent, which focuses on input based solely on push buttons, our project is not solely centered around this concept. While both involve push buttons for input, our electronic Othello board encompasses a broader scope by integrating sophisticated software, an AI opponent, automated board updates, and interactive LED buttons. Thus, the fundamental essence and purpose of our project differ significantly from that of the patented keyboard.

In summary, while there may be certain superficial resemblances or shared elements with existing patents, our project's holistic approach, unique integration of software and hardware, and distinct functionality set it apart, making it substantially different from the patented technologies in question.

Project Description

Objective and Specification

The E-thello Board aims to innovate the classic game by seamlessly integrating AI software as the user’s virtual opponent while still providing them with an authentic on the board

Othello experience. This innovative LED board features illuminated buttons, each representing black and white pieces, all connected to a computer algorithm that autonomously flips pieces after the player or AI moves. It also highlights legal moves in distinct colors, making gameplay dynamic and engaging. Players can signal their moves with a push of a button and pieces are flipped accordingly through the LED control algorithm.

Resources and equipment

The fabrication of the Othello board was carried out through 3D printing, which initially posed a budgetary challenge in terms of acquiring a 3D printer. However, we successfully addressed this by establishing a collaborative partnership with the faculty at the 3D-printing lab, located in Clemons Library. They graciously offered access to their 3D printer and materials at no cost after completing the provided training. Furthermore, to assemble the PCB and power system, we relied on a soldering iron, which was readily available within the NI lab, streamlining our production and testing processes. We also resort to our network with the School of Architecture to use their laser cutter to cut out the translucent circular acrylic lid on top of the push buttons.

Object Block Diagram

The block diagram of the Othello board is shown below in Figure 1:

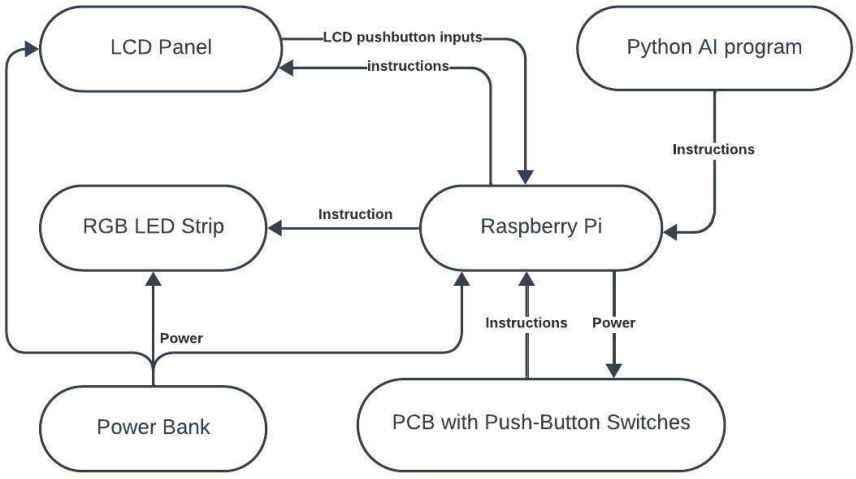


Figure 1. Othello Board Design Block Diagram

In the illustrated block diagram, the system's power source derives from a portable power bank. Its primary purpose is to provide power to several components, including the Raspberry Pi, PCB, and LED strips.

The push buttons serve as input devices, enabling users to communicate their actions to the microprocessor. The microprocessor interprets these signals, capturing the current state of the board. Subsequently, an AI algorithm takes charge, determining the subsequent move and dictating the illumination color of specific LEDs in response to the game's dynamics.

Othello AI Algorithm

As mentioned previously, one core aspect of this project is the AI program that users are playing against. This program was written from scratch; we did not use any open-source code. The main algorithm that was implemented is principal variation search, or NegaScout [15]. This is a more advanced version of the minimax algorithm, which aims to minimize the maximum gain of the opponent, resulting in a more optimal move [4].

Perhaps the most challenging part of this algorithm is determining a way to calculate this gain; that is, developing and tuning a set of heuristics that is able to evaluate the board state (assign a score) as accurately as possible. In the context of the game of Othello, several preliminary heuristics [14] [16], which we investigated the feasibility of implementing and efficacy in move generation, are mobility (the number of legal moves a given player has), stability (the number of pieces a player has hat cannot be flipped over again), and the number of corners a player has.

The latter two heuristics are relatively obvious as to why they are desired - stable pieces, which include corners, directly contribute to a player’s final piece count. Mobility is a more nuanced heuristic that seems counterintuitive at first: programs that prioritize mobility tend to capture *fewer* tiles during the early-to-mid game. The idea behind this strategy is based on the fact that pieces during all but the last stages of the game are extremely volatile, being able to be flipped back and forth numerous times. However, if priority is placed on maximizing the number of moves, which naturally minimizes the opponent’s moves, it will be able to control where the opponent plays. This provides a significant advantage, as the opponent can be forced to play on undesirable squares, which, for example, may give up a corner. In the end, though, some of these heuristics may be too inefficient to calculate (notably stability), so they might not be able to be used.

The algorithm's iterative deepening nature simplifies difficulty selection. The program can be limited in the amount of time it has or the number of moves ahead (plies) it can search. In addition, because it is agnostic to move ordering in the game, the user will be able to select who goes first.

Microcontroller and LED Strips

Although we initially chose to use a Raspberry Pi Pico WH, it lacked the processing power and RAM needed to run our Othello AI algorithm. As a result we ended up switching to a Raspberry Pi 3b+ to address this issue. This miniature computer runs on the Raspberry Pi OS, and thus supports development using IDEs such as Thonny. This is responsible for running the Othello AI program, which also controls the LEDs for each tile, adjusts the board state based on a button press on the push button control PCB, and determines the possible moves for a given board state.

The Raspberry Pi 3b+ can run Python programs through the Thonny IDE [5]. We also used PyCharm as our IDE of choice for development on a separate computer, and transferred files over through email when needed. In addition, the Raspberry Pi has tutorials and libraries on setting up the board to control the WS2812B LED strip, which greatly simplified the programming process [6]. Implementation-wise, we used the Neopixel Python library to control the LEDs by initializing a strip variable before passing in the number of LED nodes, brightness, and GPIO pin. Each LED can then be individually addressed much like indexing through an array, all of which provides the high level code needed for us to easily program the strips to display our Othello board state.

In terms of the wiring, the WS2812B LED strip has a 5 V, GND, and Data In (DIN) pin [8]. Each of these were wired to pins 2 (5V power), 14 (GND), and 16 (GPIO 18) as shown in Figure 2. In Figure 3, each individual component is laid out.

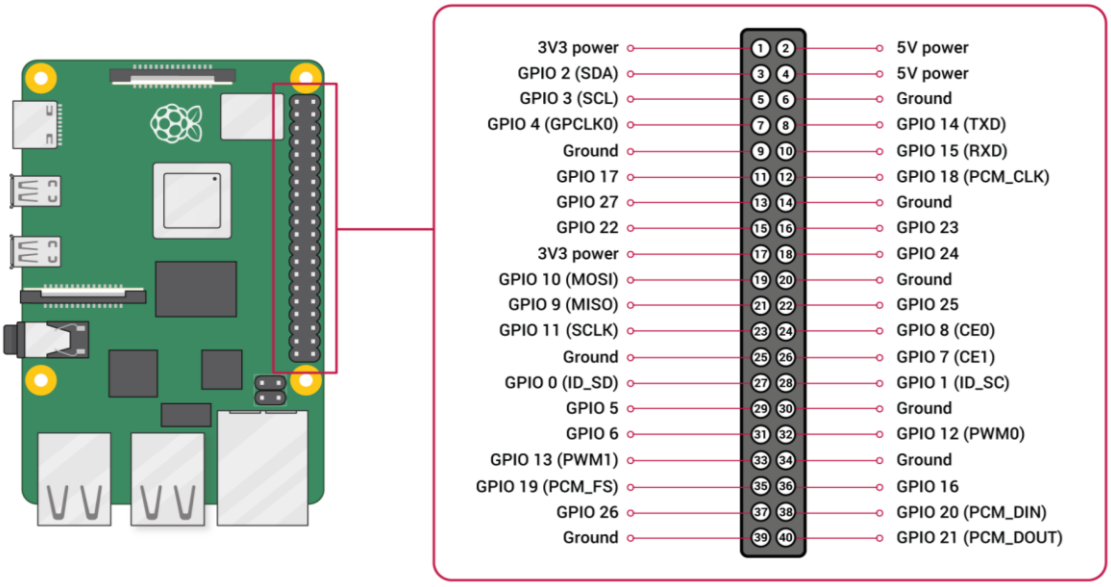


Figure 2. Pinout Diagram for Raspberry Pi

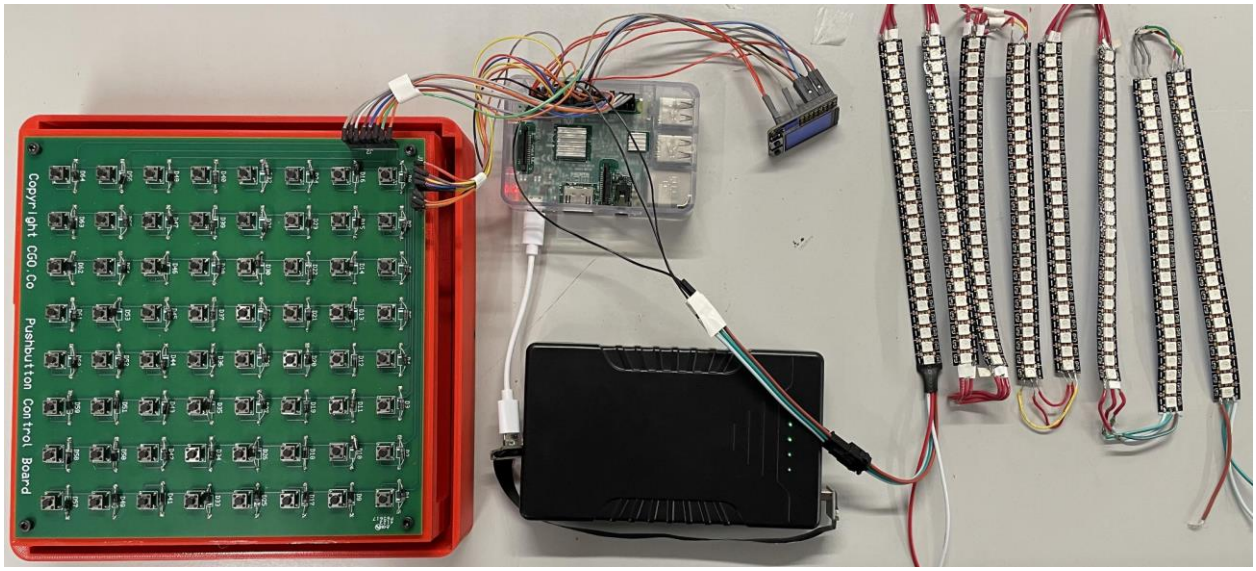


Figure 3. Layout of Raspberry Pi with LED Strip, LCD, power bank, and PCB

Power System

This project has a greatly simplified approach to the power system, which supplies 5.0V and 2.5 A to the Raspberry Pi through a Micro USB plug [7]. Then, the rails on the Raspberry Pi can be used to supply ~5V to the WS2812B LED strip [8] and a negligible amount of power to the push button control PCB. However, the power bank used to deliver the necessary voltage only had a 5V, 2A, 12000 mAh USB output, but through testing this didn't negatively impact the performance or stability of the Raspberry Pi. Theoretically, assuming the Raspberry Pi has a power consumption of 950 mA, this means that the bank should supply enough power for 12.63 hours. In addition to LEDs, there were also 64 buttons, each corresponding to an LED.

Lastly, a 12.6V, 1A external DC power adapter is bundled along with the power bank. The power bank can be charged while playing the game so there is no risk of the system running out of power. Overall, this feature eliminates the need to stress test our product to figure out how long this device can be powered.

Push button PCB controller and Polling Algorithm

To allow the player to input their moves, a push button PCB board was designed to house 64 buttons, with each corresponding to a tile on the 8x8 Othello board. Traditionally, this would require wiring 1 GPIO pin and 1 ground pin to the Raspberry Pi for each button, meaning 128 IO pins would be needed for all 64 buttons. Due to the impracticality of this, we went with a multiplexing approach based off of Charlieplexing, which is an efficient method for scanning and decoding a large number of switches using few I/O lines [18]. This method involves using N I/O lines to scan a matrix of $(N/2)^2$ keypad switches [18], with half the lines being used for input and half for output. In our case, we set up 8 row scan lines as input, with 8 column lines driven as outputs for a matrix of 64 buttons. The push button PCB layout is shown below in Figure 4.

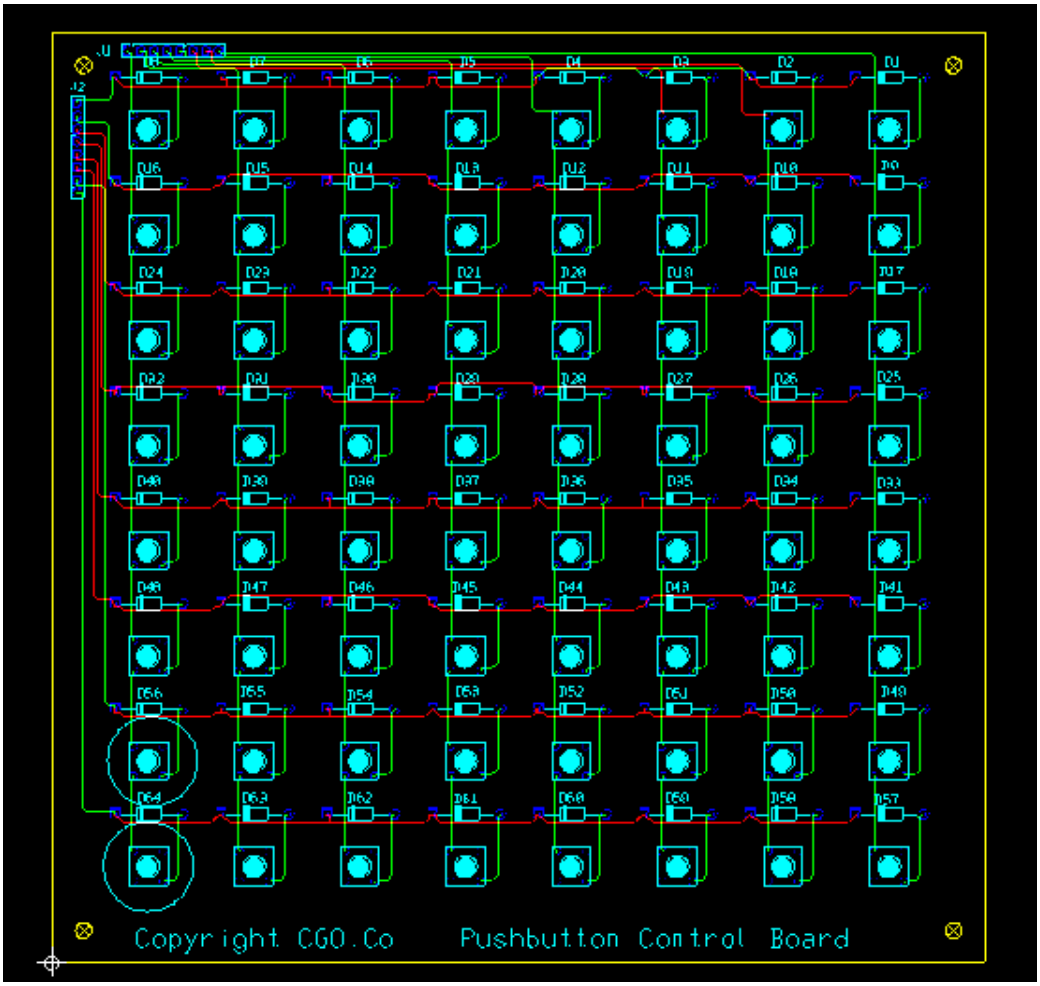


Figure 4. PCB Design for Pushbutton Control Board

To poll for a button press, our Python script contains a loop that repeats for as long as a button hasn't been pressed. Within it, we iterate through each output pin, forcing it to high for each column line. For each column pin set to high, we check if any of the row pins have been set to high. If so, then this indicates that a button at the corresponding row and column has been pressed, because this completes a circuit between a certain output pin and input pin. This is represented as a coordinate, where the letters A-F indicate columns 1-8, while rows 1-8 are represented by their numbers. For instance, B3 corresponds to the button at column 2, row 3. This coordinate is then sent to the Othello program to be parsed as the player's move.

To wire this PCB to the Raspberry Pi, GPIO 2, 3, 4, 17, 27, 9, 0, and 5 were all connected to column pins A-F, respectively. Meanwhile, GPIO 14, 15, 7, 1, 12, 16, 20, and 21 were all connected to row pins 1-8, respectively. For reference, Figure 2 above shows the pin header on the Raspberry Pi.

LCD

One of the primary ways the user interacts with the Othello board is through the integrated LCD. The LCD is connected to the first 2x12 set of pins on the Raspberry Pi in the configuration shown above in Figure 2. Our Python program also controls the user interface menu displayed on the LCD. To draw text on the display, we used the adafruit-rgb-display Python library, which conveniently handled the communication between the LCD and Raspberry Pi. Finally, the user input is performed through the two push-buttons that come as part of the LCD.

The menu itself was written from scratch, consisting of a main menu, two options menus, and a game menu. The main menu allowed users to toggle between three options: difficulty selection, turn order selection, and game start. Difficulty selection was the first options menu; users could choose between easy, medium, and hard AI opponent difficulty. The turn order menu enabled users to choose whether they wanted to go first or second. After the user selects the game start option, the LEDs are illuminated and the game begins. The LCD displays the current score of the game in a visually-pleasing format and allows the user to undo their moves and exit the game if they desire.

Major Challenges

With this level of complexity for the project, several major challenges were encountered. The first of these was designing a PCB capable of communicating button presses from 64 different tiles to the microcontroller. Given the Raspberry Pi's small number of pins, this was overcome by using a technique known as Charlieplexing, which is described in an earlier section. In addition, we ran into issues with attempting to run our AI algorithm on a Raspberry Pico WH due to its lack of RAM and computing power. As a result, we ported all of our code over to a Raspberry Pi 3B+, which has 1 GB of RAM and a 1.4 GHz processor [20]. Despite this change, many optimizations and changes to the AI algorithm were still made to ensure that the program can run in this environment while still performing well.

We also faced several challenges regarding the mechanical design. First, we had to figure out a design for the Othello pieces such that it could interact with the push buttons on the PCB, while also allowing the light from the LEDs to shine through. We came up with a design where the Othello pieces would have a small extrusion on the bottom to push the push buttons and holes on the sides to thread the LEDs through. Then we ran into an issue with the dimensions of LED strips and the PCB. The PCB was constrained to 60 square inches, so

we had to find feasible LED strips such that they would be centered in each of the push buttons. We decided on 48/m LED strips as that would allow us to fit 8 LEDs in one row while also staying under the 60 square inches mark. Unfortunately, the supplier that we purchased the LEDs from, delivered the wrong LEDs so we were forced to pivot to 144/m LED strips. Instead of having every LED light up like we initially planned out, we had to light every third LED. Lastly, we ran into many issues with printing the Othello pieces and outer shell. The 3D printers had many inconsistencies while printing the components. The shell holes were often rough and had little extrusions, making it difficult to smoothly slide the pieces through. The pieces also were rough, adding additional friction between the holes and pieces. We ended up sanding down both the pieces and holes to make the pressing action smoother.

The biggest challenge came with assembling the entire Othello board. First, we had to thread the Othello pieces through the LEDs. This was one of many concerns for us because the soldered joints were fragile and the wire we had used previously, namely the threaded wires, were difficult to work with and unmalleable. Once we threaded all sixty-four pieces through, we had to secure the Othello pieces from moving. The LED strips had adhesive tape on the underside so we fixed the Othello pieces to the LED strips. When we began to slot the pieces through the holes we had trouble fitting all the wires within the outer shell. We improved and created more space by sanding out a chunk of the outer shell. Lastly, was aligning the pieces above the push buttons. This was arguably the hardest challenge we faced because all the wires had to be set in a certain position for the pieces to depress.

Test Plan

The testing plan consists of three major sections: the AI algorithm, PCB, and the mechanical design. Most of the testing for the PCB was done on a breadboard, hence why it is in its own category. Once the design is finalized, the board was fabricated and went through the same test to ensure that the PCB is fully functional. The algorithm tests are straightforward and shown in Figure 5. The mechanical design test plan is split into three major components: the outer shell, inner shelf, and the Othello pieces. All three of these designs follow the engineering method. The components are designed, prototyped, and tested. Once the final prototypes are fabricated, they undergent integration testing. The mechanical testing plan is shown in Figure 6.

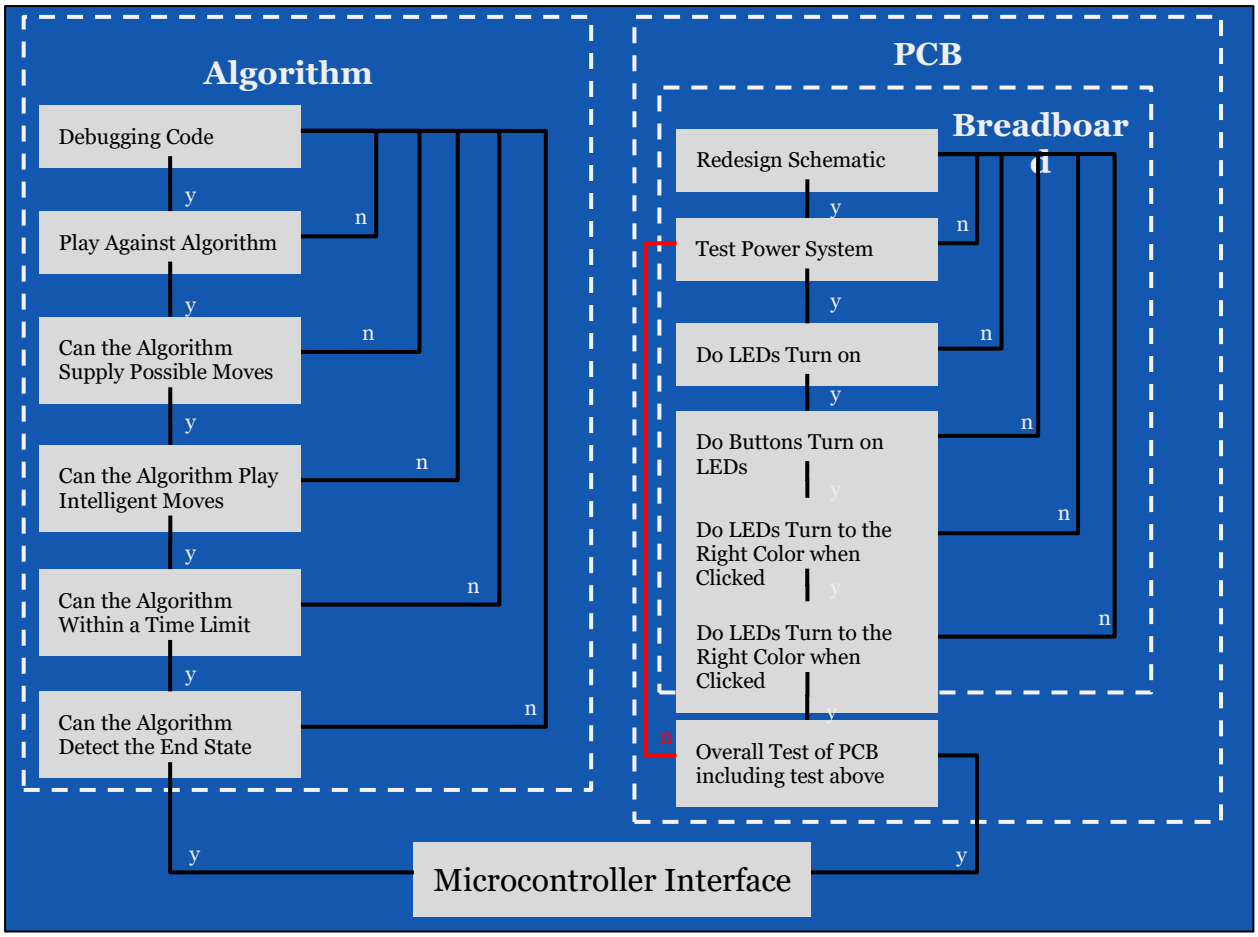


Figure 5. Test Plan for AI Algorithm and PCB Design

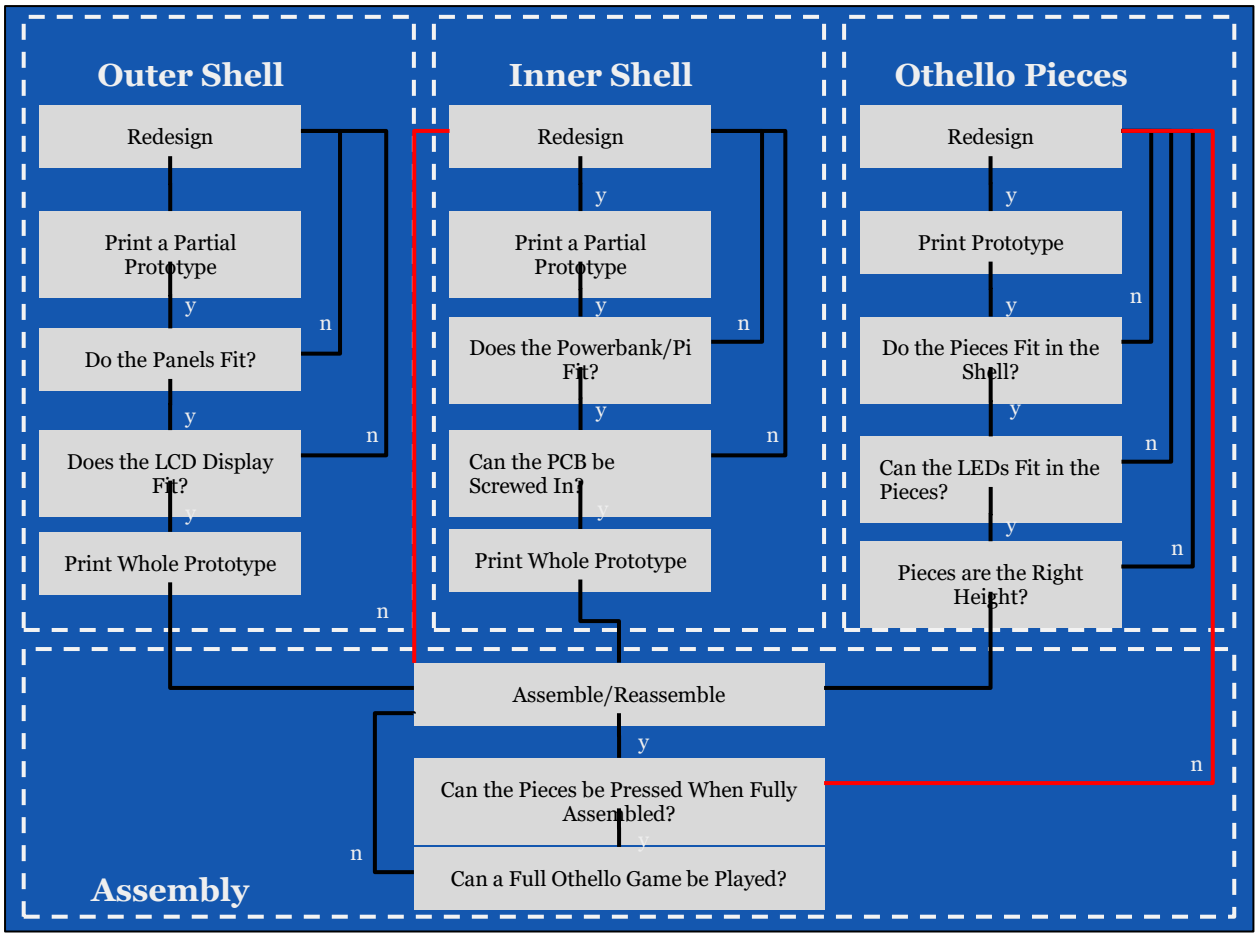


Figure 6. Test Plan for Mechanical Design

Deliverables

The physical structure of the Othello Board is shown below in Figure 7:

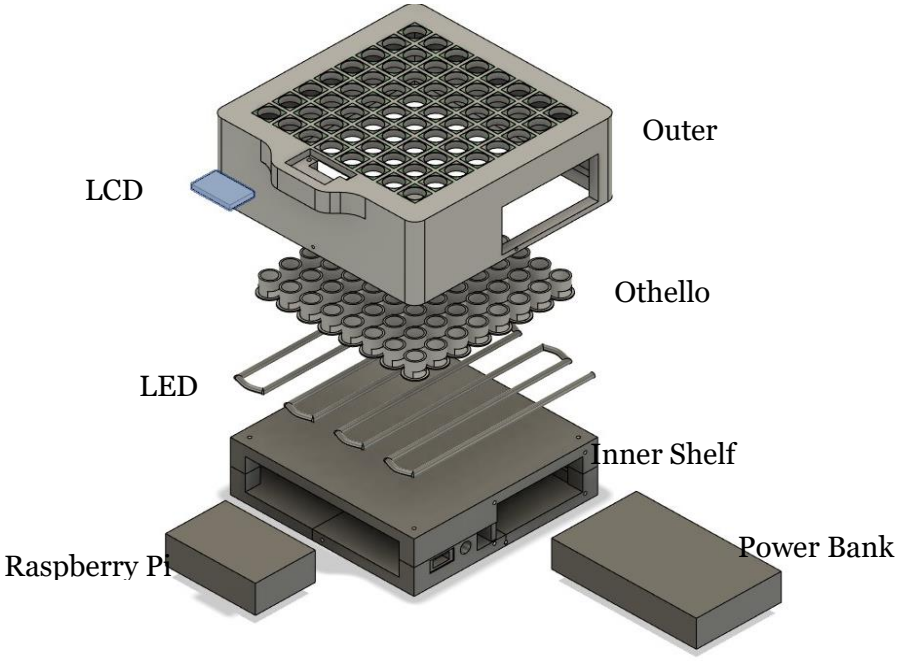
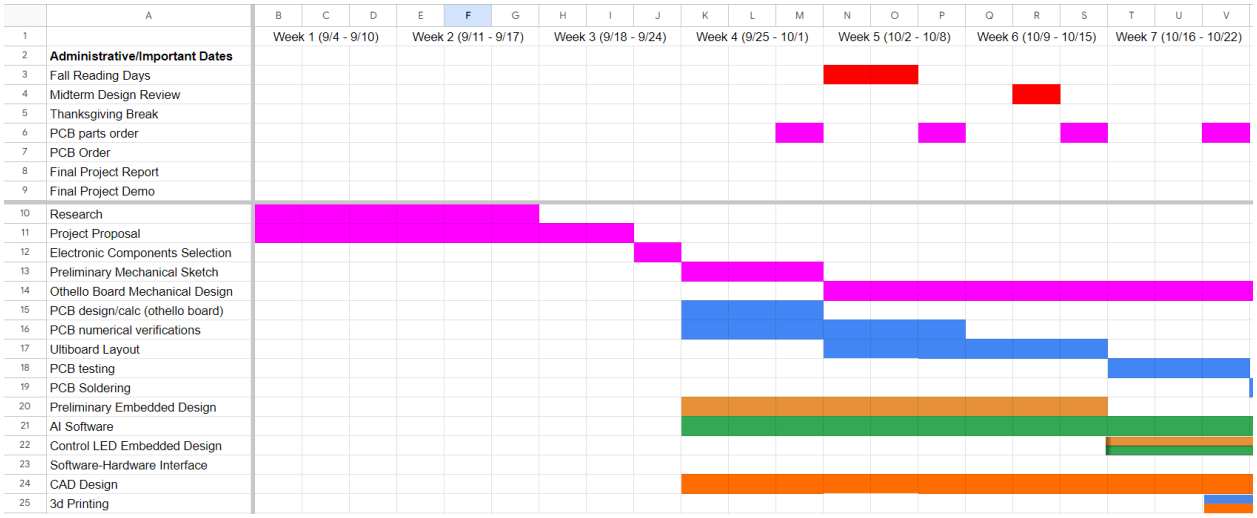


Figure 7. Othello Board Physical Configuration

The culmination of our efforts resulted in a final product that closely resembles a traditional Othello board but with the distinctive feature of its pieces being replaced by 3D-printed press buttons. Concealed beneath the board's surface are essential components, including the microprocessor, power system, and PCB, all seamlessly tucked away within the board's casing, ensuring a professional and aesthetically-pleasing design.

Timeline



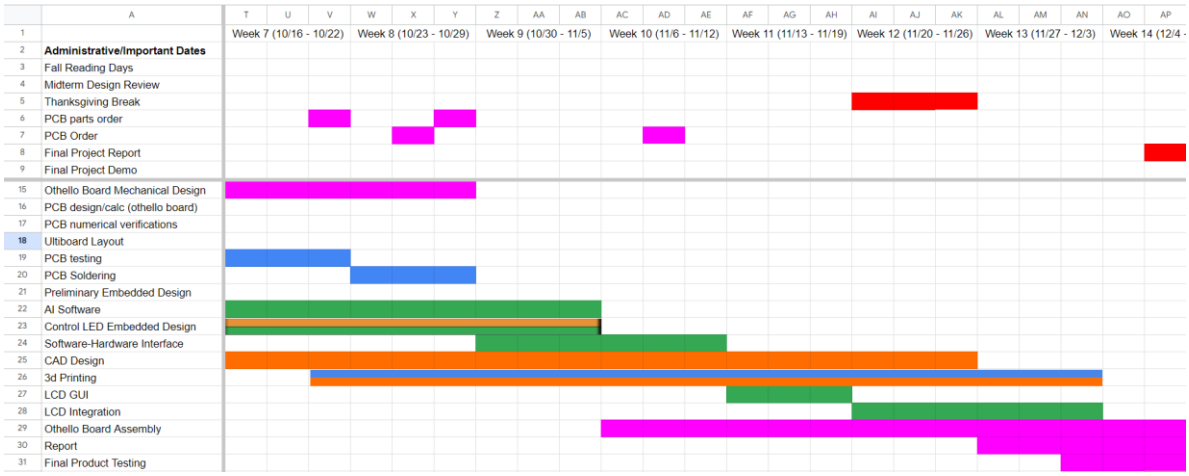


Figure 8. Gantt Chart

The Gantt chart, presented above in Figure 8, highlights the timeline for the development of the electronic Othello board, including administrative dates, deliverables, and holidays. The chart is separated with different colors, indicating tasks assigned to each member of the team. Some tasks have a mix of two colors, meaning that the task involved a collaborative effort between two members of the team. Pink squares represent tasks that are shared between all team members, and red shows the administrative dates and due dates.

The four key components in this project are the PCB, AI software, microcontroller interface, and housing for the physical components. The AI software can be developed independently of the other four tasks. Numerical verification can be done in tandem with the design of the PCB. PCB testing naturally came before ordering the PCB. The 3D-printed housing can be worked on in parallel with the design of PCB. The assembly and testing of the final product was done last.

The responsibilities were divided up as follows. Ziang led the PCB design. Lawrence was primarily responsible for mechanical and CAD design with the assistance of Ziang and Dennis. Dennis was primarily responsible for developing firmware for the microcontroller with support from Brian. Dennis was also primarily responsible for software development for the AI program. Brian was primarily responsible for the embedded software and LEDs. These roles were subject to change depending on the needs of the project, and we contributed to every part of the project in varying amounts.

The electronic Othello board should meet the following requirements:

- The Othello board shows legal moves on the player's turn
- The AI opponent makes a move without error or intervention
- The AI updates the game board with its move

- The AI successfully detects the end of the game
- Pressing a button on a tile changes the selected piece to the player's color
- Pressing the button updates the state of the game (flip all necessary pieces)

Letter Grade	Criteria
A+	6 requirements are met
A	5 requirements are met
B+	4 requirements are met
B	3 requirements are met
C	2 requirements are met
D	1 or fewer requirements are met

Table 1. Grading Scale

Cost

Given our project budget of $500, most of the major expenses went into parts used directly in the final design, or for aiding in creating the final design. This second category consists of parts used for prototyping core features of the E-thello board, such as the 8x8 LED panel and Raspberry Pi Pico WH microcontroller. Although parts like these were left out in the final design of the E-thello board, these greatly aided during the design process and testing. In addition, some expenses also went into tools and materials like super glue and black spray paint, some of which were and weren't used for the final assembly. Outside of this, some expenses also went into labor from 3rd parties, such as having Advanced Circuits print our push button control PCB. Lastly, there were a few expenses that were unnecessary in the grand scheme of the project, such as a WS2812B 48 LEDs/m LED strip that was unusable in the final design due to having the incorrect spacing between LEDs.

With this latter point in mind, there are many cost reductions that can be made per unit if we were to manufacture our E-thello board in 10000 unit quantities. This includes bulk pricing discounts for components ordered from Digikey, with many parts being close to 50% off per unit when ordered in quantities greater than 1000. In addition, all components outside of those used for the final assembly of our E-thello board can be omitted from the cost to

manufacture each unit. Lastly, we could potentially secure large discounts for 3rd party services if we decide to manufacture in 10000 unit quantities. For instance, Advanced Circuits would likely offer special pricing plans and discounts should we decide to manufacture our board in larger quantities.

See Table 2 in Appendix A for the full budget spreadsheet.

Final Result

Our final prototype has all of the intended features outlined in the abstract: the ability to run a full Othello game, legal move highlighting, an AI opponent with adjustable difficulty, an LCD to show and control various aspects of the game, a LED strip that displays the board state, and an outer shell to house all of the necessary components. In terms of the success criteria defined in the proposal, we have met all 6 requirements, and added some new features on top of them. First, during the player's turn, the LED strips light certain tiles green to display the corresponding legal moves for the current board state. Second, the AI opponent will automatically play a move if the user has already made their move, or if the user chose for the AI opponent to play the first move after turning on the board. Third, through extensive testing, the opponent has also been shown to never make illegal moves or incorrectly update the board state, which has been ensured by the algorithm implementation. We chose to represent white pieces as red LEDs and black pieces as blue LEDs due to the strip being unable to display black as a color. Fourth, we've also shown through testing that the AI can properly detect the end of a game, which is when there are no possible remaining moves for either player. Fifth, pressing a button on a tile always changes the selected piece to the player's color, assuming that it was a legal move. Sixth, the proper Othello game logic is all incorporated in the program, including game state updates, which are done whenever a legal move is inputted through a button press.

In addition to these criteria, we've also incorporated additional features. The LCD panel as mentioned in the project description adds a level of simplicity, flexibility, and user-friendliness to our Othello board. Users can not only adjust the AI difficulty level, but can also undo their moves, see the current score, and choose which player starts first. These quality of life changes allow both beginners and experienced players to customize their game settings to fit their needs. In addition, these features also make this board an even better training tool for learning and improving at Othello. For instance, adding in move undoing allows the user to analyze their previous moves and reconsider how alternative moves could affect the game.

Future Work

The Raspberry Pi 3b+ has wireless capabilities, which introduces the possibility of playing against online opponents rather than an AI opponent on the board. Due to time constraints, this wasn't a feature we were able to implement, but it is feasible for a future group to incorporate this into a similar project. This would also require them to design a web or mobile application that communicates with the Raspberry Pi, enabling any online player to input their own moves on a virtual Othello board.

In addition, the ability to play against another human player could be another quality of life feature to add in the future. This would be a good addition to our device because if there are two players present, this feature would allow players to play against one another. This would be an easier feature to implement, but we were met with time constraints and unfortunately unable to add this feature.

There were a myriad of unforeseen issues that could have been avoided during the design process. The biggest of these was soldering the LED strips, which was an extremely tedious process; the tiny copper connectors were not only hard to solder, but were also prone to falling off even with very little force, which led to us having to constantly resolder the connections. In addition, a lack of insulation on the soldered ends led to certain LEDs short circuiting and burning. Since the WS2812B LEDs are connected in series in the strip, one burnt out LED required us to replace an entire 24 LED strip. To avoid this pitfall, we could've redesigned our PCB to directly include pinholes to solder individual WS2812B 5050 chips onto, rather than having to cut separate WS2812B LED strips. This would be a cheaper and easier way to incorporate LEDs into our project that also maintains the benefits of using existing WS2812B Python libraries.

Another unforeseen issue was heavily relying on 3D printing. As mentioned in the physical constraints section, dimension limitations, machine availability, print time, and color were all restrictions imposed on our project. Despite the flexibility of 3D printing, having to remove, sand-down, and reprint recurring components such as the buttons was an unnecessarily tedious process. In addition, the machine that was used to print the larger components (i.e. outer shell, inner shelf) was a special request printer so we had to get the design approved by management. The printing lab was also only open for a limited period, so our printing durations had to be under six hours. Lastly, we could not choose which colors the components would be printed in, unless the current filament ran out.

Reference

[1] A Chess Board Which Shows You The Possible Moves of Each Piece, (Sep. 19, 2021). Accessed: Sep. 17, 2023. [Online Video]. Available: https://www.youtube.com/watch?v=uSMOKTrHZw0

[2] *Automatic Chess Board*, (Apr. 25, 2018). Accessed: Sep. 17, 2023. [Online Video]. Available: https://www.youtube.com/watch?v=JUX-hgx_V8Y

[3] "ChessUp | Level up your Chess game | As seen on Shark Tank," *Bryght Labs*. https://playchessup.com/ (accessed Sep. 17, 2023).

[4] A. Reinefeld, "An Improvement to the Scout Tree Search Algorithm," *ICG*, vol. 6, no. 4, pp. 4–14, Dec. 1983, doi: 10.3233/ICG-1983-6402.

[5] "Install Thonny | Getting started with Raspberry Pi Pico | Micropython | Coding projects for kids and teens." Accessed: Nov. 28, 2023. [Online]. Available: https://projects.raspberrypi.org/en/projects/getting-started-with-the-pico/2

[6] "Control Multiple Fully-Addressable WS2812B RGB LED Strips with a Raspberry Pi Single Board Computer - Tutorial Australia," Core Electronics. Accessed: Nov. 28, 2023. [Online]. Available: https://core-electronics.com.au/guides/raspberry-pi/fully-addressable-rgb-raspberry-pi/

[7] "Raspberry Pi Documentation - Getting Started." https://www.raspberrypi.com/documentation/computers/getting-started.html#getting-started-with-your-raspberry-pi (accessed Nov. 28, 2023).

[8] Utmel, "WS2812B Addressable RGB LED: Datasheet, Pinout and Applications." https://www.utmel.com/components/ws2812b-addressable-rgb-led-datasheet-pinout-and-applications?id=534 (accessed Sep. 18, 2023).

[9] "Drive a LED grid with a Raspberry Pi Pico and Web Serial - Part 1." https://bandarra.me/2022/02/23/Driving-a-ledgrid-with-a-Raspberry-Pi-Pico-and-WebSerial-Part-1/ (accessed Sep. 18, 2023).

[10] "What is Nema." https://www.ascopower.com/us/en/resources/articles/what-is-nema.jsp (accessed Sep. 18, 2023).

[11] "What Are NEMA Standards? | Complete Guide To NEMA Ratings," Mar. 17, 2017. https://www.psicontrolsolutions.com/blog/nema-standards/ (accessed Sep. 18, 2023).

[12] "IPC-2221A | Generic Standard on Printed Board Design," May, 2003. https://www.ipc.org/TOC/IPC-2221A.pdf (accessed on Sep. 18, 2023)

[13] "PLA Recycling: Can PLA 3D Printer Filament be Recycled?," *Wevolver*, Feb. 24, 2022. https://www.wevolver.com/article/pla-recycling-can-pla-3d-printer-filament-be-recycled (accessed on Sep. 19, 2023)

[14] C. Frankland and N. Pillay, "Evolving game playing strategies for Othello," in *2015 IEEE Congress on Evolutionary Computation (CEC)*, May 2015, pp. 1498–1504. doi: 10.1109/CEC.2015.7257065.

[15] P. Gangwar *et al.*, "Hardware/Software Co-Design of a High-Speed Othello Solver," in *2019 IEEE 62nd International Midwest Symposium on Circuits and Systems (MWSCAS)*, Aug. 2019, pp. 1223–1226. doi: 10.1109/MWSCAS.2019.8885136.

[16] T. P. Runarsson and E. O. Jonsson, "Effect of look-ahead search depth in learning position evaluation functions for Othello using -greedy exploration," in *2007 IEEE Symposium on Computational Intelligence and Games*, Apr. 2007, pp. 210–215. doi: 10.1109/CIG.2007.368100.

[17] A. Vijayakumar, "Developing an artificial intelligence bot for Othello," in *2015 IEEE Integrated STEM Education Conference*, Mar. 2015, pp. 216–220. doi: 10.1109/ISECon.2015.7119927.

[18] "Keypad Scan Uses Few I/O Lines, Decoding Keypad Buttons, Scan to Decode Matrix of Keypad Switches." Accessed: Nov. 29, 2023. [Online]. Available: http://www.mosaic-industries.com/embedded-systems/microcontroller-projects/electronic-circuits/matrix-keypad-scan-decode

[19] "Raspberry Pi Documentation - Raspberry Pi hardware." Accessed: Nov. 29, 2023. [Online]. Available: https://www.raspberrypi.com/documentation/computers/raspberry-pi.html

[20] "Raspberry Pi 3 Model B+." Available: https://datasheets.raspberrypi.com/rpi3/raspberry-pi-3-b-plus-product-brief.pdf

[21] "Adafruit Mini PiTFT - Color TFT Add-ons for Raspberry Pi." Available: https://learn.adafruit.com/adafruit-mini-pitft-135x240-color-tft-add-on-for-raspberry-pi/

Appendix A

Budget Outline

The following spreadsheet is the complete budget for the entire project, including parts and 3rd party services. Note that the total price includes taxes.

Item	Unit Price	Quantity	Total Price
Raspberry Pi Pico WH (Pre-Soldered)	$14.90	1	$14.90
PCB	$35.00	1	$35.00
PCB Assembly ($5 flat + $0.40 per part)	$33.80	1	$33.80
WS2812B 48 LEDs/m Strip	$30.17	1	$30.17
WS2812B 8x8 LED Matrix	$10.00	1	$10.00
Buttons (120 pack)	$5.99	2	$11.98
3D Printing	$0.00	1	$0.00
Power Supply Adapter	$8.19	1	$8.19
Power Bank	$39.99	1	$39.99
USB to DC Adapter	$7.99	1	$7.99
DC Power Jack	$5.99	1	$5.99
USB to Micro-USB Cable	$1.15	1	$1.15
M3 x 0.5 mm Screws 15mm (50 pack)	$10.00	1	$10.00
Male to Male DC Extension Cord	$6.99	1	$6.99
90 Degree USB to Micro USB Cable	$14.50	1	$14.50
3 State Sliding Switch	$4.04	2	$8.08
Red Pushbutton	$1.24	1	$1.24
M3 x 0.5 mm Screws 20mm (50 pack)	$10.00	1	$10.00
Black Spray Paint	$6.98	1	$6.98
Gorilla Super Glue	$7.98	1	$7.98
WS2812B 144 LEDs/m Strip	$12.99	3	$38.97
Schottky Diodes (125 pack)	$5.99	1	$5.99
Micro USB Female to Micro USB Male	$6.99	1	$6.99
Mini PiTFT IPS Display (LCD)	$15.26	1	$15.26
Raspberry Pi 3B+ (already obtained)	$0.00	1	$0.00

Total	$332.14
Remaining Fund	$167.86

Table 2. Overall Budget