SENIOR DESIGN II

Spring 2024

Dr. Lei Wei

Final Report

OPTICAL INTERACTIVE CHESS BOARD FOR BEGINNERS

Group 6

Alec Barno	Photonics Engineer
Alejandro Felix	Photonics Engineer
Cassidy Phillips	Electrical Engineer
Vinicius Resende	Electrical Engineer
Nikolai Coletta	Computer Engineer

Advisors

Patrick LiKamWa	patrick@creol.ucf.edu
Justin Phelps	justin.phelps@ucf.edu
Hatem Abou-Senna	habousenna@ucf.edu

Table of Contents

List of Figures

List of Tables

1 Executive Summary

There is no denying the conveniences that the rise of online chess brought to players around the world. Aside from the clear benefit of online play of providing players with the flexibility to engage in matches from the comfort of their homes, there are many features of online chess we take for granted today. Online chess platforms offer various features specifically designed to assist players throughout their chess career and enhance their learning experience, many of which are not available in traditional physical play settings. For example, these platforms often provide guidance to players during matches through means of movement guides, displaying the possible moves for each piece to assist the players with their game vision and decision making, analysis tools are also a common feature that allows players to review completed games, pointing out mistakes and suggesting better moves, enabling beginners to learn from their gameplay.

Even then, many chess players will agree that the feeling of playing on a physical board is unmatched, impossible to replicate within an online setting. However, the absence of such assist tools while playing can often prove daunting to beginners due to the reputation the game has, with a deep knowledge gap between advanced players and newbies. Thus, the motivation for this project. Our goal is to bring the conveniences of online play to a physical chessboard to stimulate learning and transform beginner players' first experience into less of a daunting task and make it a more engaging and enjoyable process.

This project aims to revamp the chess-playing experience by introducing some interactive features to traditional chessboards inspired by online-play platforms. The centerpiece of this project is an interactive chessboard equipped with detection and identification sensor system and LED technology. Upon a piece being lifted from the board, the corresponding tiles light up, displaying possible moves for the selected chess piece, allowing players to easily visualize the many possibilities in each of their turns. The board utilizes RGB LEDs to maximize the number possible information we can convey using the board itself without relying on external systems. For when we do need additional information, complementing this feature is a display system capable of offering real-time tips and guidance during gameplay, as well as assist in other matters such as time-keeping, displaying the time each player has left on the clock. These features help foster strategic insights into the players' gameplay and reinforce the ability of quick decision-making for players at every skill level, aiming on aiding the growth of these skills.

The method we use for piece identification is somewhat rare among other existing designs. The approach we settle on makes use of photodetectors and their exposure to green light that is reflected for underneath each square onto the bottom of each piece to detect, identify, and determine the position of the pieces on the board. This system allows for a quick response and accurate response, able to provide a smooth experience to the players. We do this by utilizing fiber optics

to illuminate the bottom of each piece and having different filters on the bottom of each different type of piece so that we can correctly identify and differentiate them. Afterall, it is important to us that there are no rooks being mislabeled as pawns, or any other mistake that could happen by incorrectly setting up on the user's side.

To finish it off, these features are contained within our specially designed housing, made to complement the design with both aesthetic and functionality in mind. The materials chosen not only ensure durability but also contribute to the overall sleek design. The housing seamlessly integrates and contains each of the systems of the board and enhances the overall user experience, allowing for portability and offering a comfortable and stable platform while also providing the feel of playing in a traditional board with a modern twist.

With this product, our vision is to not only make the gaming experience friendlier to all players but also to inspire new chess enthusiasts and learners. Users will be able to gain a thorough understanding of the basics of chess and how it is played as well as some understanding of basic chess theory and tactics. The interaction between player and board will provide players with the opportunity to learn this timeless game in a modern setting, serving as a tool to facilitate their growth.

2 Project Description

This chapter section serves as a fundamental guide for understanding the base aspects of our project, offering a panoramic view of its purpose, goals, and guiding principles that we have followed throughout this paper. Additionally, the section outlines the rules and parameters that govern our project, establishing a robust framework that ensures coherence and consistency in our endeavor.

2.1 Project Background

Board games are a great way to spend time with friends and family. Like all new games, beginners learn how to play from a manual, or they might be lucky enough to have a player present that has played the game before and can coach them through a game. Some games rely more on luck while others depend on strategy. Chess is one of those board games that relies heavily on strategy. Professional chess players can often see 10-15 moves ahead and plan their actions accordingly when playing a match. For beginners though, they do not have that much experience to predict their opponent's moves, especially when they do not know the basics of the game yet. For chess beginners, having that extra person present to help guide them would be ideal, but that is not always possible. Our group wants to create a physical board that helps teach beginners the basic rules of chess and possibly build on it to further teach them basic chess theory.

2.2 Goals

The overall goal of this project is to create a physical chess board that will help teach beginners how to play chess. To teach players the basics of chess, they need to first be shown how to initially set up the board and how each piece functions on the board. In order to accomplish this goal, we first have to create smaller goals that need to be accomplished first. We divided these smaller goals into basic goals, advanced goals, and stretch goals.

There are two basic goals for this project. The first is to detect and identify the different chess pieces. The second is to light the LED array underneath the chess board properly so that it shows the player where their selected chess piece can move to and to show a player where their piece cannot move to. These two basic goals are the most important goals we need to accomplish so that our chess board works as we intend it to.

For the advanced goals, we want to convey additional information about a player’s selected piece and to be able to save the game state for players to pick up and resume at a later point in time. By using a small built-in display, when a player selects a piece to move, the screen will display a small paragraph of text that describes how the piece moves. By using both the visuals of the board and the text on the screen, players should be able to gain a solid understanding of the basic mechanics of the game. This screen will also serve additional purposes based on the mode the board is setup in.

Our stretch goals for this project include making further use of the chess engine that will be programmed into the microcontroller to have an AI opponent for singular players to challenge against. This will allow singular players to still have an opponent to challenge even when a second player is not present. A second stretch goal we considered is implementing a small speaker into the board that will play sound effects when certain cases arise. We can play sounds that can both encourage and discourage players in their decisions on selecting the best moves.

2.3 Objectives

In order to detect and identify the different chess pieces, we must first create a system that will accomplish that. We used cylindrical bases under each piece to house IR LEDs. The light emitted by the IR LED will then be reduced by different filters at the bottom of the cylindrical base. These different intensities will then be detected by the photodiodes enabling use to determine the different chess piece type. By comparing different characteristics of IR LEDs such as intensity and turn on voltage, we were able to select the best IR LEDs possible to supply light to each square's chess pieces are on the chess board. The illumination system which has the IR LED has to maximum amount of light is provided making it easier to have a bigger voltage range for our chess piece. The light from the illumination system will then be reduced by filters depending on the chess piece type. Each chess piece will have a different filter to get them into the voltage range associated with their piece type. We used many types of IR filters to get each piece into its corresponding voltage ranges. To detect this change in intensity, each square will contain a photodiode. The photodiodes will have a change in current depending on how much light it is detecting. The more current it has, the more light it receives. The current will then be converted to voltage because of the photodiode circuit. By reading the voltage outputs of these photodiodes, we can determine the type of chess piece on that square.

In order to implement enough LEDs below the chess board so that each square has an LED, we determined the best LEDs to accomplish our goals. Ideally, we want to be able to implement more than one color into the final project so that players can associate a certain color with a specific message. We then have to determine the best way to place these LEDs so that we can easily control which LEDs turn on and which ones we want to keep off. A driver must then be selected to control the LEDs so that consistent power and current are supplied to them without burning out any of the individual LEDs.

2.4 Engineering Specifications/Requirements

TABLE 1 ENGINEERING SPECIFICATIONS/REQUIREMENTS FOR THE COMPONENTS THAT WILL BE USED TO BUILD THE SYSTEM

Component	Parameter	Specification
Shift Register	Clock speed	24Mhz
Photodiodes	Viewing angle	50°
Fiber optic cables	Core diameter	0.75 mm (0.03 in.)
Battery	Watts	72.15 w
Display	Screen length size and Resolution	4’’ 480 × 320 pixels
Microcontrollers	CPU speed	600 Mhz

TABLE 2 ENGINEERING SPECIFICATIONS/ REQUIREMENTS FOR THE PERFORMANCE AND PHYSICAL CHARACTERISTICS OF THE SYSTEM

Requirements	Units
Battery Life of the entire chess box	≥ 4 hours
Chess Box Dimensions* (LxWxH)	24 in. × 18 in. × 6 in.
Chess Board Dimensions* (LxW)	18 in. × 18 in.
Weight of the entire chess box	≤ 15 Ibs
Delay/Activation from when piece is picked up to when move is shown on the chess board	≤ 3 seconds
Voltages to differentiate the types of chess pieces	7 different voltages
Piece detection accuracy	≥ 95%

Asterisk (*): These dimensions are the max dimensions that our project will go to. We aimed to build the board as small and compact as we can.

Note: These specs in the table above are only estimations and guidelines for our project. We aimed to improve upon all the specs listed.

2.5 Hardware Block Diagram

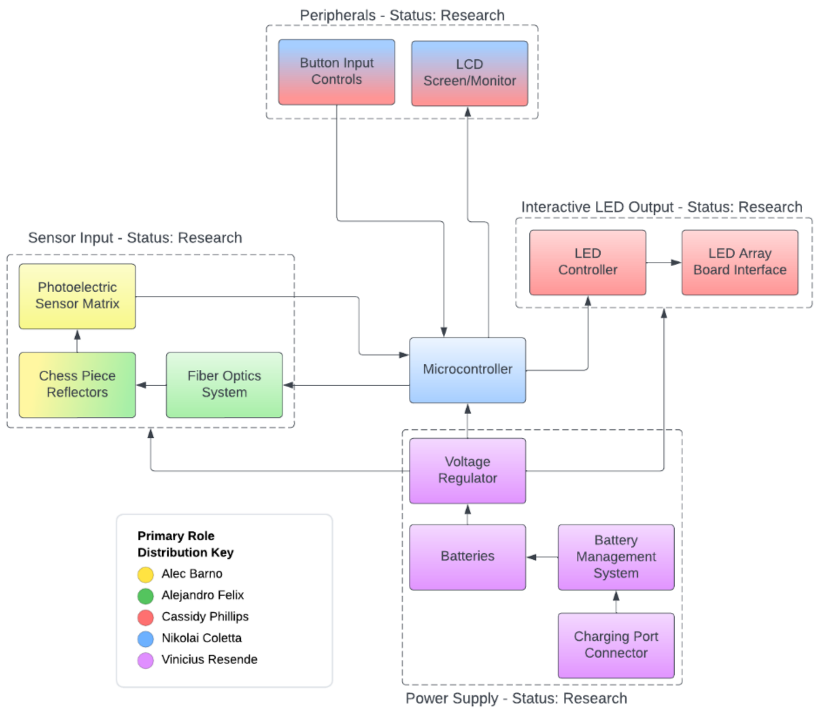

Peripherals - Status: Research
Button Input Controls
LCD Screen/Monitor
Interactive LED Output - Status: Research
LED Controller
LED Array Board Interface
Sensor Input - Status: Research
Photoelectric Sensor Matrix
Chess Piece Reflectors
Fiber Optics System
Microcontroller
Voltage Regulator
Batteries
Battery Management System
Charging Port Connector
Power Supply - Status: Research
Primary Role Distribution Key
Alec Barno
Alejandro Felix
Cassidy Phillips
Nikolai Coletta
Vinicius Resende

FIGURE 1 SIMPLIFIED HARDWARE BLOCK DIAGRAM WITH WORK DISTRIBBUTION

2.5.1 Hardware Description Summary

This section will provide a description and overview of the project's hardware components focusing on the core infrastructure of our design in a manner to simplify understanding of the device’s working principles and streamline the design process.

With the convenience of the user(s) in mind, a rechargeable battery would be the ideal power source for this project as it allows the device to operate without being tethered to a fixed power source, such as a power outlet. Batteries provide flexibility and portability, ensuring uninterrupted operation when it would otherwise be incapable of running. Along with it, a voltage regulator will be required to keep voltage consistent and stable during the device’s operation regardless of system load or battery voltage output. This will prevent damage to sensitive electrical components due to overvoltage, prevent undervoltage conditions that can lead to improper operation or data loss, and increase overall power efficiency and battery life. This component pair will work to provide power to the microcontroller, sensors, LEDs, and any other components within the circuit.

To provide the system with the information it requires to operate, a matrix of photoelectric sensors will be in place to read the wavelength of the light from the fiber optic system underneath the board that is reflected by the bottom of the chess pieces and, from there, be able to verify the position of each individual piece in play. This group, labeled “Sensor & User Input” will act as the main source of data collection for our processing functions along with any other manual input from the user interface, such as signals to start or pause the game. Meanwhile, the components listed in the “Interactive LED Output” block will serve as the main form of interaction between the user and the board. Each square in the board will be able to individually light up to communicate different things to the player(s) during the game to convey helpful information to beginners.

A secondary method of communication and interactivity between user and device will be present in the form of an attached LCD Screen that will be used to present more detailed information that would otherwise be hard to convey using only the LEDs on the board. This information may include things such as game time, strategy, current game evaluation, etc.

To connect all these together, the microcontroller serves as the central processing unit for the design. It will receive and process data from all the sensors and inputs, as well as handle any logic and decision-making in a centralized manner to optimize system performance. It will also interface with any peripherals or additional components that should be integrated in the project at a future date, that are not essential for the device’s operation but provide additional functionality.

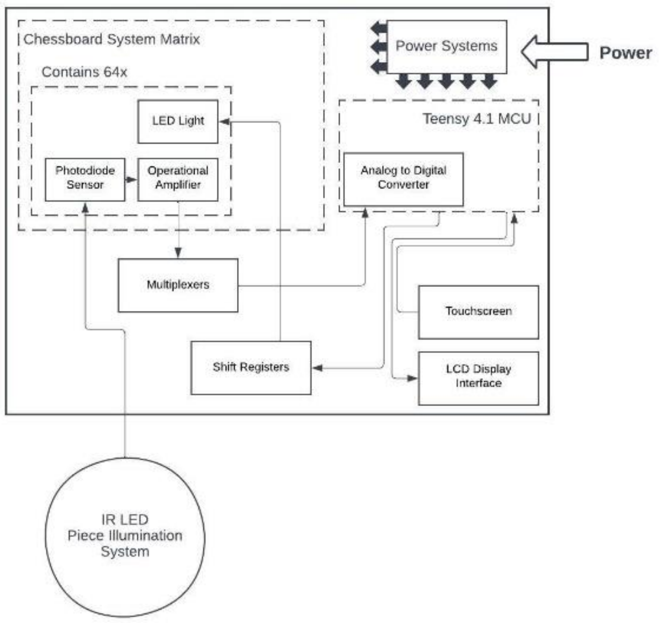


FIGURE 2 FINAL HARDWARE BLOCK DIAGRAM

2.6 Software Flowchart

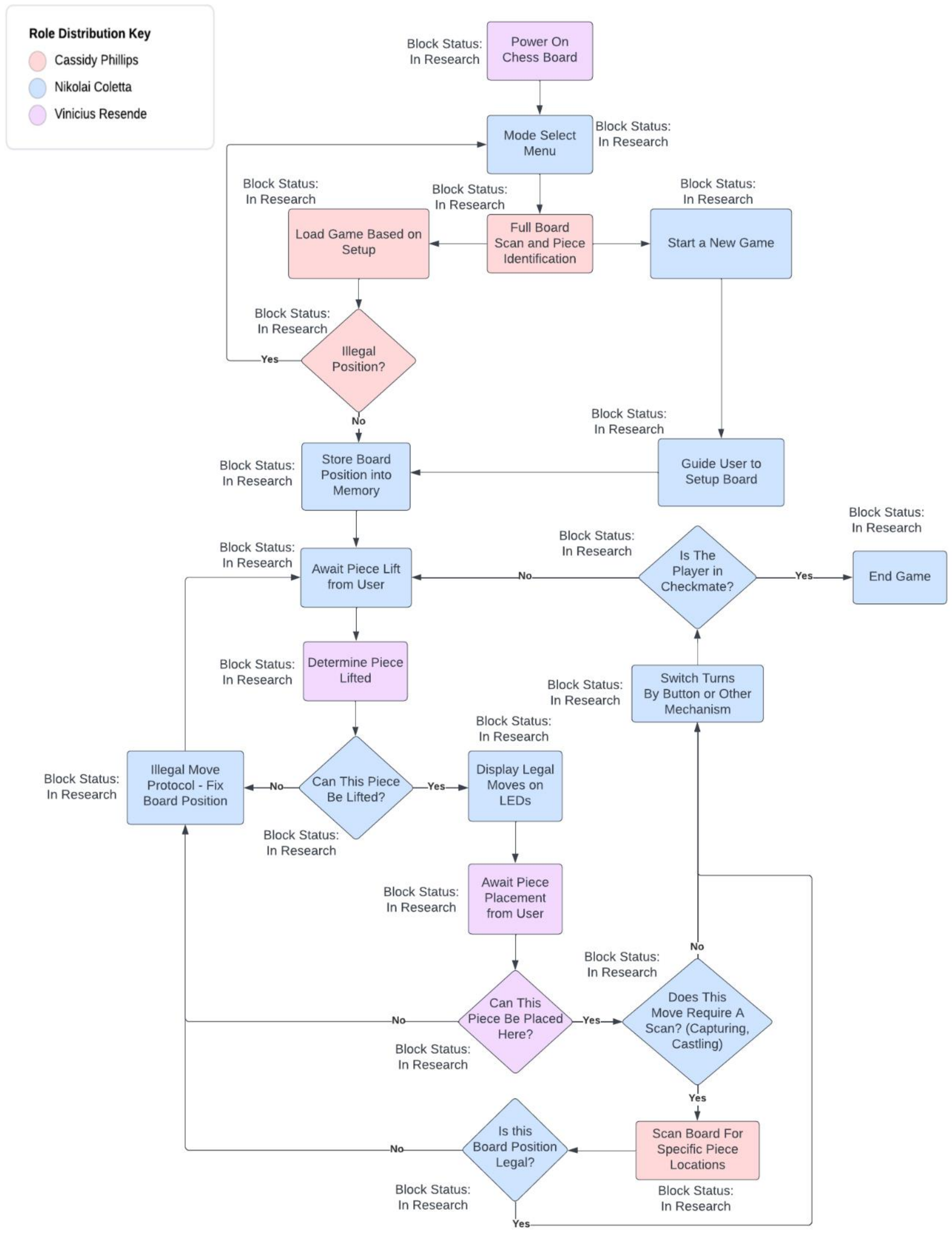
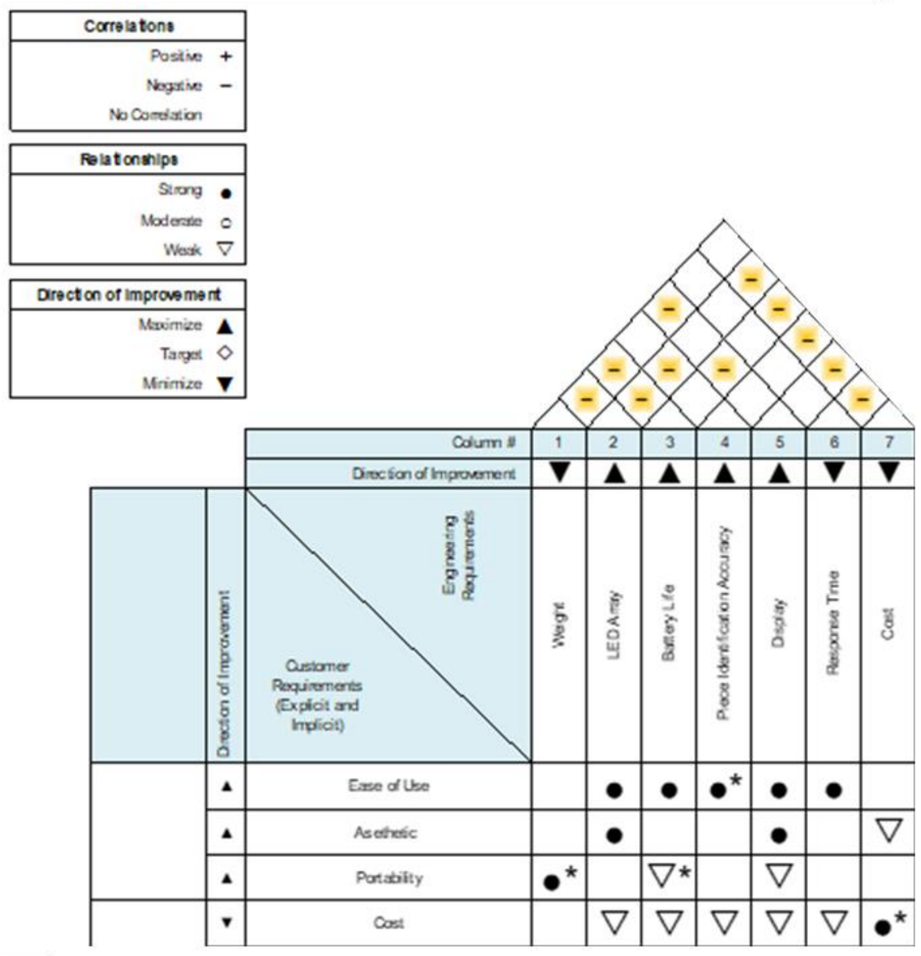

Role Distribution Key
Cassidy Phillips
Nikolai Coletta
Vinicius Resende
Power On Chess Board
Mode Select Menu
Full Board Scan and Piece Identification
Load Game Based on Setup
Start a New Game
Illegal Position?
Yes
No
Store Board Position into Memory
Guide User to Setup Board
Await Piece Lift from User
Is The Player in Checkmate?
End Game
Determine Piece Lifted
Switch Turns By Button or Other Mechanism
Can This Piece Be Lifted?
Illegal Move Protocol - Fix Board Position
Display Legal Moves on LEDs
Await Piece Placement from User
Can This Piece Be Placed Here?
Does This Move Require A Scan? (Capturing, Castling)
Scan Board For Specific Piece Locations
Is this Board Position Legal?
Block Status: In Research

FIGURE 3 SOFTWARE BLOCK DIAGRAM

2.7 House of Quality



	Column #	1	2	3	4	5	6	7
	Direction of Improvement	▼	▲	▲	▲	▲	▼	▼
Direction of Improvement		Weight	LED Array	Battery Life	Piece Identification Accuracy	Display	Response Time	Cost
▲	Ease of Use		●	●	●*	●	●	
▲	Asethetic		●			●		▽
▲	Portability	●*		▽*		▽		
▼	Cost		▽	▽	▽	▽	▽	●*

FIGURE 4 HOUSE OF QUALITY

2.8 Chess Gameplay Restrictions for Project

The rules of chess are the foundation for many of the interactive capabilities for this project. A variety of these rules pose restrictions for the design, and this project must be able to cater to each scenario on the board that can occur during a match. All rules of chess for this project, which includes concepts such as piece movement, piece capturing, castling, promotion, en passant, check, and checkmate, follow the standard rules of chess [1].

Currently, there are only three options available for the LED colors per square- red, blue, and purple. These options may expand, or change to different colors, but the system can easily be designed around these three colors. As explained more specifically in the following sections, these colors will be used to communicate certain information on the board, with the general rule of blue being used to show the user where to put a piece, red for where to remove a piece, and purple as replacing a piece. To keep this board accessible to audiences that are colorblind, there will be additional information displayed as needed on the screen display on the side.

2.8.1 Illegal Moves

Because this chess board is designed to help teach beginners on how to teach beginners, this board needs to be able to detect and fix all illegal moves. An illegal move in chess is a broad term that refers to any move, or lack thereof, that breaks the rules of the game. A simple example of an illegal move is moving the bishop directly left from where it as it. The bishop can exclusively move diagonally, so this type of move would be illegal, and the game cannot continue until the user undo's this move and makes a valid move instead. This covers the first category of illegal moves, which is simply done when a user moves a piece to an invalid square it can move to.

Whenever the user picks up a piece, the square it was picked up from will blink blue on and off, to show the user where to return the piece if either they change their mind or if they do an illegal move. All squares that the player can place this piece will be lit up blue, or any squares they can capture a piece will be lit up purple instead. For the scenario where the user places their piece in an invalid square, that square they placed it on will light up red, and the display will show an 'X' symbol with text below saying that that was an illegal move. The game cannot continue until a valid move is played, so the red light will stay there until the mistake is corrected. That is how the chess board will display these types of illegal moves.

It is important to note that for making invalid moves, more must be considered than just the move-set of as given piece. In chess, there is a concept known as "pinning a piece", where a piece cannot move without sacrificing more valuable pieces behind it. In some cases, a piece may be pinned to the King, which can be legally sacrificed, so that pinned piece moving out of the way and exposing the king will be an illegal move. The protocol will be identical to the previous move correction- it will light up red on the incorrect square and continue to display the valid locations. Certain locations will also be removed on the board for pieces that are pinned to show a reduced list of available moves for these pinned pieces, to reduce further confusion.

Other illegal moves to consider for a beginner to do includes picking up too many or too few pieces. If the player makes a valid move, but then instead of passing their turn to the next player decides to make another 'valid move', there needs to be a system in place to both identify where the illegally moved piece went and where it needs to return to. This system will need be extensively designed to make it as easy as possible for the user to return to the initial position of the board, while only using three colors and a display.

2.8.2 Recovering Disrupted Board States

Illegal moves can display helpful information for what went wrong when doing only single moves, but in the situation where multiple pieces are bumped over or switched by accident, the board could reach a disrupted state where multiple corrections are needed. The reason this board needs a guidance system to return the board state is really because it is designed for beginners, and a complete beginner might accidentally move a bunch of pieces at once, or even knock over a couple pieces to the ground.

There are two different routes to take when correcting disrupted board states from the players. First, every piece that is either illegally placed or missing can have LED indicators underneath, blinking red where they need to go and having solid red for underneath the squares that they are currently placed that needs to be removed (in the case that they aren't on the board, there will be no indicator besides the return point). This can get confusing though, as if multiple pieces are missing then there will be lots of flashing squares that the user can't distinguish between until trial and error.

The second solution could be just to have the first additional move displayed and corrected, and wait until the user fixes that one before displaying the next error, until it is finally returned to a legal state. This method may get tedious if multiple pieces are taken off but will likely work the best in streamlining the corrections and making it as easy for the user as possible, so this method will be the one that is implemented. All LEDs will turn off instead of just the flashing blue of where a piece needs to go, and a solid red LED turned on underneath the piece that needs to be moved there. This is only done for cases where many moves are made instead of just one.

2.8.3 Special Moves

In chess, there are three special moves that can be made that will have to alter the way the LEDs and display interact with the user. These special moves must be detected for and break the regular pattern done by just the simple capturing and moving of pieces.

Castling is the first of the three special moves in chess, which differs from all other moves in those two pieces of the same color move in that one move. Ignoring this rule would put the board in a disrupted board state as described in the section above. To detect and allow this, when picking up the king the additional move of castling will be shown in its highlighted available moves. After moving the king to its desired square for castling, the light under the rook will flash red and the one square that the rook needs to move for castling will be lit up blue. Note that these moves for castling should only be allowed if castling is legal at that point of the game. If the player has moved their king or rook already, castling on that side is no longer allowed (Or on either side if the king is moved in particular). In addition, if the king is in check, or passed through a square that would put him in check, castling isn't allowed either. All of these would have to be considered when lighting up the square or not as a valid move for the king. For the sake of the user experience being smooth as well, moving the rook to a castling square like it would castle won't display castling as a forced next part of the move, since that is a legal move of moving your rook by itself, but it won't throw an illegal move if you move your king after that rook movement, if it is in the same turn. This way the user doesn't have to undo the rook move, do the king move, then the rook moves if they want to castle but just moved the rook first.

En passant is another special chess rule in the sense that it is the only move to capture a piece that isn't on the square you move to. What this would mean for the display is that instead of highlighting purple where the piece is directly on top of it, it will highlight purple where you move the pawn to do en passant, and then it can highlight red underneath the pawn you capture during en passant after the previous move was selected. By detecting this move, en passant is only possible after the opponent on their most recent move moved their pawn two squares forward, and your pawn is in the right position. This opportunity passes though after your move, so if en passant isn't taken at that moment it can't happen again with that specific pawn. When determining if en passant is a legal move to make, it will have to consider the previous move made.

Lastly, promotion is a special rule in that as a pawn is pushed to the end of the board, it needs to be replaced with another piece, which is the only circumstance you would replace one of your own pieces. The way this would have to be implemented in this project is by having the pawn move to the last row result in the square being lit up red, and then blinking red until replaced with the promoted piece. Another thing the player may do it though is simply place down the piece being promoted to, so that needs to be a valid option for the player, by lifting their pawn for promotion and placing a different piece at that square.

3 Research

In this research section, we delve deeper into the intricacies and nuances of our project's system and design, seeking to establish a comprehensive understanding that transcends surface-level knowledge. Here will be explained the theoretical foundations of the project and share our personal approaches to the challenges we faced when designing our chessboard. Our aim is to cultivate a rich reservoir of insights, drawing from established principles in the relevant fields to create the structure that will serve as the intellectual backbone of our design.

3.1 Existing Products, Past Projects, Similar Work

Currently there are multiple versions of this idea in the market. Some of the companies of these projects include Square Off, ChessUp, and Chess House. The prices overall range from $139 to $349.

The least costly of the three products we looked into is the Chess Genius PRO from ChessHouse. This board currently costs $139. Compared to what we want to implement into our project, this board does not have a lot of the aesthetic characteristics we are aiming for. The ChessHouse board does not make use of lights to provide information to the players, instead it makes use of a small LCD screen that is embedded into the board to display the piece movements to the player. This board is suited more towards single players because of its built-in chess engine (ChessGenius by Richard Lang). This board detects the piece movements due to pressure sensors built into the chess board squares.

The Square Off Pro chess board from Square Off is the best portability wise. Currently priced at $239, this board is flat and can simply be rolled up when it is done being used. This board only makes use of small white lights to convey which squares a player can move to. These lights do not light up the entire square though. The lights are on the line between squares so when a square turns on, the lights on the sides of it will turn on. The use of lights is simple, but it is effective enough to still relay information to the player. The chess engine this board uses is Stockfish. With this built in chess engine, players have the opportunity to play against an AI that can be adapted to different difficulties so beginners will not be overwhelmed. Two in-person players can still face off against each other on this board and the statistics about a game can actually be saved so that they can be analyzed later.

The ChessUp board is the closest to what we want our final project to accomplish. This board is currently priced at $349. The ChessUp board makes use of 3 different colored lights to relay information to the player as a game is played. These colors are used so that all possible squares a selected piece can move to are lit up, but the color used determines how strong the move to that square is. A companion app is also available for this board so the amount of assistance a side can receive from the board can be programmed in to even the playing field between two players with a wide gap in experience. This board ensures players play by the book and will not allow any illegal moves to be made. The board detects selected pieces as soon as a player touch any of the pieces on the board.

There was a senior design project done in the past at UCF from summer 2018 to fall 2018 that heavily inspired our project. The method this team used to detect pieces is the biggest difference between the two projects. This past team used a combination of reed switches and resistors to determine the pieces. Each chess piece contained a unique resistor value so that 12 different values of resistors were used in total, with each side having 6 unique resistors. When the pieces were moved, the equivalent resistance of the board would change and that was the value that was monitored to see how the chess pieces were moved. The microcontroller monitored these changes by using an ACDC converter. Even though they were able to accomplish the task of identifying the different pieces, their system was still not perfect. Some of the main issues they dealt with lied in their piece identification and detection system. The reed switches that were used were not consistent and they faced challenges of getting consistent readings of which piece was moved due to noise or poor contact.

Our goal with this project is to improve upon the previous projects' issues and limitations. We believe that incorporating optics into the project will enable us to create a chess board that is more consistent with detecting and identifying pieces. Using optics will also enable faster detection of the pieces and a quicker response from the programming and software to provide the users with a smoother gameplay experience.

3.2 Chess Piece Identification System

One of the most important systems for our project is the chess piece identification system. Our project's main goal is to teach new players how to play the game of chess by showing them where each piece can move to on the board. We needed to develop a chess piece identification system that would enable players to learn how each of the different pieces moves around the board, no matter the state of the game and where the pieces are on the board. Our chess piece identification system needs to be accurate, consistent, and reliable or else the players won't be able to learn how to play the game of chess properly. Getting this chess piece identification system to properly work is of utmost importance because failure to do so will result in an unusable chess board and teaching tool.

For our chess piece identification system, we believe that using optics and photonics as the main principle for the system would create the best results possible. After doing extensive research on other "smart" chess boards or chess trainers, we believe that optics and photonics can fix and improve on some of the shortcomings and weaknesses in other systems that rely on magnets or pressure or switches. In our chess piece identification system, there are two main components involved: 1) the optical sensor and 2) the key to differentiate each piece. Understanding both components and the relationship they have was important in order to choose the best parts for this project. As outlined in upcoming sections, we explain the different options we had and explain why we chose the specific parts that we did.

3.2.1 Optical Sensors

3.2.1.1 IR Sensors/Sensing

One of the first ideas we thought of for our chess piece identification system was to use infrared (IR) sensors. IR sensors are an optoelectronic device that can measure and detect infrared radiation. IR radiation is below visible light on the electromagnetic spectrum and therefore cannot be seen by humans. IR radiation ranges from the end of the red wavelengths from visible light all the way to microwaves, typically from 0.75 µm to 1mm. IR sensors are typically used as motion detectors or proximity sensors. We first thought that using IR sensors and putting one under each square for our piece identification system would have worked, but we encountered a few issues with that idea. The main issue with using the IR sensors as the optics for our piece identification system is that being able to differentiate each different type of chess piece would have been difficult to achieve using the IR sensors. We would have had to adjust each different piece's IR emittance, whether that was through different LEDs or other type of source. That would have become too complex and tricky to properly figure out since each piece would have required a power source and circuit design to fit inside the pieces. IR sensors are relatively cheap, going from anywhere between $1 - $10 on average, but the components that would have been needed to go inside each piece to differentiate the types of pieces would have been expensive and much more difficult. For those reasons, we decided not to use IR sensors as the sensing mechanic for our project.

3.2.1.2 Color Sensors/Spectrometers

Another idea that we had for our chess piece identification system was to use color sensors. Color sensors are optoelectronic devices that can determine the color of an object. The color sensor emits light, typically by an LED, and then uses photodiodes to collect the reflected light from an object. The light received by the photodiodes is converted into an intensity value that can be split into a ratio of colors, typically on an RGB scale and can accurately determine the color of the object. For example, if the object is yellow, then the reflected light received by the color sensor would break down into red and green intensity values, but not blue. Using color sensors would make differentiating the different kinds of chess pieces very simple and straightforward as we would just have to color code the pieces accordingly and provide a key for reference. The main issue with using color sensors as the optics for the chess piece identification system is the cost. Color sensors typically range in price from $5-$20 per sensor. Since we need 64 sensors, one for each square of the board, that would become very expensive to do. The cost of the sensors needed for the project could range from $320-$1280. Since our project is not sponsored by anyone and all of us are in college and don't have large bank accounts, we decided that we couldn't use color sensors for this project.

We did however try to come up with a different approach to still use color in our chess piece identification system as color coding each of the different pieces would be the most simple and straightforward way to help teach brand new players how to play the game and learn how the pieces move. Another idea we had for the optical sensing of our chess piece identification system was to create a spectrometer system. A spectrometer is a scientific device that is used for detecting and separating wavelengths of light on the electromagnetic spectrum. Spectrometers are very sophisticated machines and therefore are very expensive. Spectrometers typically cost in the range of hundreds to thousands of dollars, sometimes even tens of thousands of dollars. We obviously knew we couldn't buy and use a spectrometer in our project, but we tried to develop a way to make a "homemade" spectrometer that would still allow us to differentiate each of the pieces using color coded bottoms. We thought of an optical sensing system that would use photodetectors and color filters to achieve the same result as using a spectrometer. We thought that we could use a white light source aimed at the bottom of each square, underneath the board, and point towards the bottom of a piece that would be on the square. Then the white light would be reflected from the bottom of the chess piece and go into the photodetector. The photodetector would convert the reflected light into an electrical signal that would then be related to a specific color.

We encountered a few problems with this system design. The first problem we encountered with this design is that a photodetector cannot distinguish between various wavelengths of light when it receives the reflected light and converts it to an electrical signal. We would have needed to use color filters on the bottom of the chess pieces to differentiate them and also needed to put the color filters on top of the photodetector to only allow that specific color and its associated wavelengths through into the photodetector. That wouldn't be a major problem if there was some way to have all of the different colors related to the different chess pieces that could be filtered through one photodetector. But we are not able to do that as stacking all of the color filters onto the photodetector would just allow every wavelength of visible light to come through and it would just read the reflected light like it was just the white light. Since there are 6 different pieces in chess, there would be 6 different colors to differentiate through the photodetectors. That means we would need 6 photodetectors under each and every square on the board. Our project would need 384 photodetectors for the entire chess board. Each of the photodetectors would have a color filter on it so that it would only receive that particular range of wavelengths of light coming from the bottom of a certain kind of chess piece. Photodetectors are relatively cheap, ranging in price from \$0.50-\$5, but since we need 384 photodetectors for the project, the price for our piece identification system would drastically increase. The cost for all of those photodetectors could have ranged from anywhere between \$192 all the way up to nearly \$2000. We decided not to go with this design of the chess piece identification system as the price and complexity was just too much when considering the other options available.

3.2.1.3 Photodiodes

The last chess piece identification system design that we came up with is using photodiodes. Photodiodes are a semiconductor device that takes in light and converts it into an electrical current. Photodiodes are very similar to photodetectors and sometimes they get confused and mistaken for one another. Photodetectors are a more general term for optical sensors whereas photodiodes are a specific type of photodetector because of how it works. Photodiodes typically use a P-N junction in order to produce a photoelectric current. The photons (light) get absorbed into the P-N junction and they create an electron-hole pair that moves through an electric field, which creates an electrical current. The design of this system using photodiodes is very similar to the design of the system that would have used photodetectors and color filters. In this version of our chess piece identification system, we have one photodiode under each square of the chess board and it will measure the intensity of the reflected light from the bottom of the piece on top of the board. This intensity value of the reflected light will then be converted into an electrical current and we determine what piece it is based on the current value we read.

There are a few advantages in this system design over our ideas about using photodetectors. For example, when using photodiodes, we are only concerned about intensity of light. We don’t have to worry about relating the different kinds of chess pieces to a specific color. This gives us some flexibility when determining how we can differentiate the six different kinds of chess pieces, which will be further explained later on. We also only need one photodiode under each square of the board instead of six photodetectors with the previous system. This will reduce the cost of the piece identification system overall and also give us more room to be able to adjust the size of our chessboard and the dimensions of each square, piece, etc . Photodiodes are very similar in price to photodetectors, ranging anywhere from \$0.50-\$5. The realistic price for all sixty-four photodiodes will be in the range of \$30-\$50 and shouldn’t be any more than \$64 (\$1 per photodiode).

Another distinct advantage for using photodiodes is the simplicity of the entire system overall. Since photodiodes will convert the reflected light into an electrical signal (current), we just need to design a simple circuit to be able to properly determine the six different kinds of pieces. We can control the resistance, input voltage and overall design of the circuit, which will enable our group to just focus on measuring the current produced by the photodiodes. Since the current produced by the photodiodes is classified as analog (continuous-time varying), we need to convert that into a digital signal so that the microcontroller can read the value and determine if/if not there is a piece on a specific square. To convert the signal from analog to digital, we need to buy and incorporate into our system an analog-to-digital converter. An analog-to-digital converter (ADC) is a device that converts an analog signal measurement into a digital value in the form of binary code. Since an analog signal is continuously changing, the ADC has to first “take a snapshot” and then quantify that data into binary digits, or bits. Once the signal is quantified and given a resolution in bits, then the data can be read in binary values and sent to the microcontroller as a digital signal.

TABLE 3 OPTICAL SENSORS TECHNOLOGY COMPARISON

	IR Sensor	Color Sensor/Spectrometer	Photodiodes
Function	Measures IR radiation changes	Measures and determines specific wavelength ranges related to color	Measures intensity of light
Wavelength Range	0.75 µm to 1 mm	≅ 380 nm to 750 nm (Visible Light)	Any wavelength range (specific to diode specs)
Extras/Other Components	Light source and circuit inside each chess piece along with some filter or other method to differentiate them	Color filters for each different kind of chess piece	Analog-to-Digital Converter (ADC)
Implementation	Medium to Hard	Hard	Easy
Number of sensors	64	64 - 384	64
Cost (1 sensor)	≅ $1 to $10	≅ $5 to $20 (color sensor) ≅ $100s to $1000s (spectrometer) ≅ $0.50 to $5 (photodetectors)	≅ $0.50 to $5

After considering all options and determining the pros and cons of each system design, we decided to create our chess piece identification system using photodiodes as our optical sensors. Using photodiodes enables us to have great flexibility with how we determine the different kinds of chess pieces; it is the most cost-effective option, and it is the simplest option for data collection and implementation into our chess board.

3.2.1.4 – Part Selection

TABLE 4 PHOTODIODES PART SELECTION

Name	Photosensitive Diode	PD204-6B	SFH 203 P
Brand/Manufacturer	Uxcell	Everlight Electronics	ams-OSRAM USA INC.
Seller	Amazon	DigiKey	DigiKey
Peak Wavelength	940 nm	940 nm	850 nm
Spectral Range	400 nm - 1100 nm	840 nm - 1100 nm	400 nm -1100 nm
Receiving/Viewing Angle	40°	N/A	150°
Active Area	N/A	N/A	1 mm^2
Price	\$0.65 (per diode) \$6.49 (10 pcs)	\$0.43 (per diode) \$15.17 (100 pcs)	\$1.00 (per diode) \$6.50 (10 pcs) \$42.60 (100 pcs)

After deciding to use photodiodes as the optical sensors for the chess piece identification system, we needed to determine the best photodiode to use for our project that fits our requirements and needs. The most important qualities about photodiodes that affect our project specifically are the spectral range that the photodiode can read, the peak response wavelength for the photodiode and the receiving/viewing angle that the photodiode can detect light. After doing extensive research, we determined that the photodiode we used for the chess board is model PD204-6B from Everlight Electronics. We chose this specific photodiode because it best fits our parameters for the project. Since we used light in the IR wavelength spectrum to go into the hole in each square, we needed a photodiode that can only read in the IR spectrum as ambient light from the surroundings of our project could affect the readings of the pieces and cause inaccuracy. The photodiode we chose has a spectral range of 840 nm to 1100 nm and also has a peak response wavelength of 940 nm.

3.2.2 Differentiating each Piece

3.2.2.1 Colors/Color Filters

The first idea we thought of when trying to figure out how to differentiate each of the six different kinds of chess pieces was to use color. We thought that color coding the pieces would be the most simple and straightforward way to classify the piece types. We believed that color coding the pieces would help the players learn each piece faster as there would be a visual stimulus to relate each piece to and it would also make setting up the board before a game easier to understand and memorize. The only way we could color code the bottom of the pieces is by using color filters. Color filters are a type of filter that can affect the color of light either through absorption, reflection, conversion and/or balancing of any range of wavelengths in the visible spectrum. We would have needed to use color filters on the bottom of each piece and on top of all of the photodetectors. These color filters would only allow specific ranges of wavelengths of light through to the photodetector and classify each piece type based on each specified wavelength. Since we would have used a white light source under each square to illuminate the bottom of the chess board, the reflected light would change from the broad spectrum that white light has to a very small range of wavelengths corresponding to a specific color. This would have enabled the microcontroller to easily identify the different piece types and allow us to have a high accuracy for determining pieces during any game.

Color filters are relatively cheap and usually come in a pack with a bunch of different colors. These filters are typically sold as big sheets that can be cut to specific shapes and sizes, depending on the needs of the customer. The filters can cost anywhere from $10 to $30, but that usually includes most of the basic colors of the rainbow and multiple sheets for each color. If we decided to use color filters as the way to differentiate the different chess pieces, we would have only needed 1 pack of filters since the bottom of our chess pieces will be small in size when compared to the dimensions of the filter sheet. The reason we won’t be using color filters is not because of the color filters themselves, but because of the cost and setup of the optical sensors for the piece identification system. Color filters are our backup plan to use in case the other kind of filters we use to differentiate piece types doesn’t go according to plan.

3.2.2.2 Materials

Another way we determined how we could identify and differentiate each of the chess pieces was to use different materials. We thought that using various materials that vary in reflectivity could work since we are using photodiodes which just measure intensity of light. Choosing materials based on how reflective they are to identify piece types would work with our choice of photodiodes and optical sensor setup. We could have used any sort of material in the world, and they wouldn’t have had to relate to each other. We thought of using a mirror, shiny reflective tape that is typically used in construction, aluminum foil, a penny, and many more random objects could be used as they all are reflective at various levels.

The main issues with using materials and relating their reflectivity values to differentiate piece types is that there is no source or large amount of data about different reflectivity values for various materials and sources. There is not much information gathered about the exact reflectance values of random objects in the world, which means we wouldn't be able to provide an exact and precise reflectance value without lots and lots of testing. We would need to have tested many different materials and objects dozens if not hundreds of times so that we could get an accurate average value for the reflectivity. That wouldn't have been an issue if we only needed a couple of different values, but since there are six different kinds of chess pieces, we would need at a bare minimum six distinct reflectivity values of materials that wouldn't interfere with each other. For our project specifically, we would like to get as many differentiating values as possible to make it easier for the programming and coding of our board to be as smooth as possible. To have done that would have required dozens and dozens of tests with many various materials such as tapes, metals, mirrors, plastics, etc. and the values we would come up with would only be estimates and averages as there is not truly a source that we could fact check our numbers with. It could still be done, but because of the complexity and extensive amount of testing needed to be done, we decided to not differentiate our chess piece types based on the reflectivity of various materials.

3.2.2.3 Light Filters

The last concept we thought of to differentiate the piece types is to use filters. There are many kinds of light filters that can change light in many various ways. Light filters are designed to narrow the range of wavelengths of light to achieve a specific goal or objective. There are some light filters that specifically absorb certain ranges of wavelengths or reflect certain wavelength ranges. There are also filters that can refract or diffract certain wavelengths as well. For our project, we would be focusing on neutral density (ND) filters. ND filters are light filters that are designed to reduce light intensity evenly across a specified wavelength spectrum. ND filters are typically neutral gray in color and act as a mask to control the intensity of any illumination source. ND filters are characterized by the value at which they reduce the intensity of light. The value that each ND filter has is tied to a percentage that denotes the intensity of light that is transmitted through.

TABLE 5 DIFFERENTIATING EACH PIECE COMPARISON

	Colors/Color Filters	Materials	Light Filters
Function	To differentiate the chess piece types based on color and wavelength associated to colors	To differentiate the chess piece types based on intensity of reflected light due to the reflectivity values of various materials and objects	To differentiate the chess piece types based on overall intensity of light

Wavelength Range	≅ 380 nm to 750 nm (Visible Light)	Any wavelength range (depends on wavelength of light source)	Any wavelength range (specific to wavelength of light source)
Implementation	Medium	Hard	Easy
Cost	≅ $10 to $30	≅ Free to the price of whatever material is used	≅ $10 to $35

The main reason why we decided to use ND filters for our project to differentiate each of the 6 kinds of chess pieces is because of the simplicity of ND filters. Using ND filters allows us to associate a specific number to a certain kind of piece and will know how much light is transmitted through. ND filters are just as simple as color filters, but they don’t require the same level of setup with the optical sensors. We only need to use one photodiode under each square of the board to measure the intensity of the bottom of a piece. ND filters are also relatively cheap and equivalent to color filters, ranging anywhere from $10 to $35. When buying ND filters, they usually come in a pack with three to four different values of filters in there so there are different options depending on the purpose for the filters. Overall, we decided to use ND filters because they are the most simple and easy to use when compared to the other options for how we could have differentiated the chess piece types.

3.2.2.4 – Part Selection

TABLE 6 FILTER COMPARISON

Name	Square Filter Kit	Gel Filters, CTO Transparent Light Sheets	Lighting Neutral Density Gels Filter Sheet
Brand/Manufacturer	SIOTI	Meking	RENIAN
Seller	Amazon	Amazon	Amazon
# of filters	4	4	6 (2 per each kind)
Value of filters	ND2 (½ transmission) ND4 (¼ trans.) ND8 (⅛ trans.) ND16 (1/16 trans.)	1 ½ ¼ ⅛	ND3 (≅48% trans.) ND6 (≅24% trans.) ND9 (≅12% trans.)
Type of filters	Neutral Density (ND)	Color Correcting/Enhancing	Neutral Density (ND)
Material	Pmma	Polyester	Polyester
Price	$22.99	$18.99	$16.98

Once deciding that ND filters would be the best way to properly differentiate the chess piece types, we needed to figure out exactly how we would implement that into our project. The main concern for differentiating each chess piece type is determining how many different values we need the photodiodes to measure so a proper game of chess can be played. There are 6 different chess piece types. Ideally, we would want to have 12 different values to be associated with each of the 6 different types of pieces for both colors, white and black. Unfortunately, getting 12 distinct values measured from the reflected light from the bottom of the chess board will be extremely difficult and has increased risk of overlapping that will cause the microcontroller to read the wrong values and therefore light up the wrong moves that a player could make. Currently the plan is to get 7 distinct values to differentiate the pieces on the board: 2 different values for each set of pawns (white and black) and 5 values related to the other chess piece types. We want to differentiate between each set of pawns as pawns are the only piece that cannot move backwards. If we differentiate between them, then that will allow the microcontroller to more easily track and determine which pawn is what color and what direction they should move in. We don't need to worry about what the color is for the 5 other types of pieces as they can all move backwards. Their direction of movement isn't strict like it is for pawns.

To achieve 7 different values that will differentiate the chess pieces on the board, we chose 2 different sets of ND filters. The first set of ND filters has 4 different filters: ND 2,4,8, and 16. Each different filter reduces the intensity of the reflected by a certain factor. The second set of ND filters we chose has 3 different filters: ND 3,6, and 9. By using these 2 sets of filters, this provides us with 7 possible options to differentiate the pieces without even counting the base layer that each piece will have of just the mirror. The base layer that doesn't have any filter and only the mirror will act as 1 value as it should show exactly how much light is being reflected from the bottom of the pieces on the chess board. These 2 sets of filters provide us with options and flexibility when determining what values, we want to control each piece with. It gives us the chance to test different combinations and precisely choose 7 distinct values for our project.

3.3 Illumination System

The purpose of the illumination system is to generate light so that it can be directed toward the photodiode which will then distinguish what piece type is on the square above it. In order to accomplish this, we needed a light source that will generate enough optical power to make it easier to distinguish between each piece type. The light source will be placed inside a cylindrical base which will sit under the chess pieces. The illuminations source has to be placed on top of the square in a consistent area since moving the light source could change the intensity of the light received by the photodiode.

3.3.1 Illumination Sources

Illumination sources are the devices that provide light for the photodiode to read the intensity of the filtered light. We needed 32 illumination sources since there will be an illumination source under each of the 32 pieces in the cylindrical base. There are many illumination sources that can provide light to the photodiode such as lamps, LEDs, lasers, laser diodes, quartz halogen lamps, xenon metal halide lamps, and other high and low powered sources.

Xenon metal halide lamps stand out for their high-power output and ability to generate high-quality light. Notably, approximately 24 percent of the energy input is transformed into light, marking a significant efficiency improvement compared to incandescent lightbulbs, which typically operate at an efficiency range of 2 to 4 percent. However, Xenon metal halide lamps come with certain trade-offs, including relatively short lifespans compared to some other light sources. Another consideration with Xenon metal halide lamps is the warm-up time required to reach full brightness, taking a few minutes. Additionally, these lamps need a high startup voltage to ionize the gas within, generating a considerable amount of heat. These characteristics should be taken to though for our optical system, considering factors such as instant brightness requirements and the impact of heat generation on the overall system. To optimize the performance of lamps within our optical system, considerations such as the incorporation of reflectors or lenses may be essential. Reflectors, if not already integrated, can enhance the directionality of light, focusing it on the target, while lenses play a crucial role in channeling the light efficiently towards the photodiodes. Moreover, the versatility of lamps extends to their potential use in combination with optical filters.

Quartz halogen light bulbs, belonging to the incandescent light family, are recognized for their ability to generate light by heating materials until they emit visible light. However, this distinctive process comes with basic inefficiencies, as the majority of the energy is lost as heat, making incandescent lights generally inefficient. Despite these drawbacks, quartz halogen light bulbs offer certain advantages, particularly in the context of compact optical systems. The compact and cost-effective nature of quartz halogen light bulbs makes them an attractive option for optical setups with limited space, aligning with the spatial considerations of our system. Their small size and affordability contribute to their appeal, allowing for versatile integration within small or tight spaces. However, there are significant limitations associated with quartz halogen light bulbs. Notably, these light sources are notorious for producing high temperatures during operation. Consuming a considerable amount of energy, ranging from 55 to 100 watts, quartz halogen bulbs are among the less energy-efficient lighting options available. One of the primary concerns with quartz halogen bulbs is their potential to generate excessive heat, posing a risk of melting the plastic housing and the cylindrical base holding the illumination source within our system. This thermal issue not only threatens the integrity of the optical setup but also has the potential to compromise the entire system's functionality. Long exposure times to high temperatures could result in irreversible damage, leading to the degradation of both the housing and the cylindrical base.

Another suitable option for our project to consider was the Helium-Neon (He-Ne) laser. These laser devices exhibit several advantageous features, making them noteworthy contenders for our optical system. One key strength lies in their stable wavelength, providing consistency in the emitted light. Additionally, He-Ne lasers are renowned for producing good Gaussian beam profiles, ensuring precision in light distribution. The high quality of the beam and the simplicity of the alignment process further add to the appeal of He-Ne lasers for our application. However, the Helium-Neon laser does have certain limitations that influenced our decision-making process. One notable drawback is the relatively short lifespan of He-Ne lasers, ranging from 1000 to 10000 hours. This limited lifetime poses practical challenges, especially in scenarios where long usage times or minimal maintenance is needed. Furthermore, despite their advantageous qualities, He-Ne lasers exhibit a relatively low output power, typically ranging from 35 to 50 milliwatts even for the most powerful variants. This may be a critical factor depending on the specific illumination requirements of our system. Another consideration is the warm-up time required for some He-Ne lasers to achieve population inversion. This delay in activation could impact the responsiveness of the system, particularly in applications that demand swift and immediate uses. One substantial drawback that weighed prominently in our decision-making is the bulkiness of He-Ne lasers. These devices tend to be over 1 foot long, exceeding the parameters we've established for our system. The compactness of our optical setup is pivotal for integration, and the size of He-Ne lasers was deemed impractical within the defined spatial constraints.

Another option in our consideration for a light-emitting device is the Laser Diode. Laser diodes present many advantages that make them appealing for our optical system. Notably, their compact size sets them apart, being smaller than many other types of lasers. This characteristic aligns perfectly with the spatial constraints and design requirements of our system, enabling efficient integration without compromising on performance. Affordability is another notable aspect of laser diodes. They stand out as one of the most cost-effective devices for producing laser output in the desired wavelengths. This economic advantage positions laser diodes as an attractive choice for our project without compromising the quality of the emitted light. The high efficiency of laser diodes further enhances their appeal. Their efficiency in converting electrical power into laser light aligns with our goal of optimizing energy consumption while ensuring reliable and consistent performance. A notable feature of laser diodes is their capability to be manufactured into arrays effortlessly, owing to their small size and output capabilities. This versatility opens up possibilities for customizing and scaling our optical system based on specific project requirements. However, it's important to acknowledge certain drawbacks associated with laser diodes. They typically produce smaller optical powers, ranging from 5 milliwatts to 10 milliwatts, when compared to other lasers. Laser diodes produce a considerable amount of heat creating heating issues. Additionally, laser diodes generate more divergent laser beams, a factor that needs careful consideration depending on the specific application and system configuration. Furthermore, laser diodes require sufficient current to surpass the threshold current and enter their lasing mode. Managing this current threshold is critical for maintaining stable and consistent laser output. While lasers and laser diodes concentrate higher optical power into a small area, producing a focused beam, it's essential to note that these characteristics limits their ability to illuminate multiple objects simultaneously. The light emitted is not dispersed, posing a constraint in scenarios where broad illumination is required.

An additional contender as a light source is the light-emitting diode, commonly known as LED. LEDs distinguish themselves with a high level of energy efficiency, efficiently converting approximately 80 percent of the energy input into light and only about 20 percent into heat. This efficiency not only minimizes energy wastage but also results in significantly less heat production compared to many traditional illumination sources. While LEDs may have limited power compared to some other light sources, ongoing advancements have led to the development of newer LEDs that exhibit enhanced efficiency and higher intensity outputs. A key advantage of LEDs is their instantaneous response to power, eliminating the need for a warm-up period. They light up almost instantly, providing immediate illumination. Furthermore, LEDs boast impressive longevity, contributing to their appeal for applications where a long lifetime is needed. Directionality is another noteworthy feature of LEDs. Unlike some light sources that emit light in all directions, LEDs are inherently directional. This characteristic is advantageous as it minimizes wasted light, directing it precisely where illumination is required. This targeted approach not only enhances energy efficiency but also allows for more precise control over what areas are illuminated. Additionally, the directional nature of LEDs makes them particularly good to use since they would not need reflectors to direct the light toward the photodiode. In other traditional light sources reflectors would be needed since light would be emitted in all directions. This leads to wasting light since not all of the light will be directed towards the photodiode if reflectors are not used. Having LEDs be directional makes the building of the cylinders easier since reflectors will not be needed.

Electromagnetic spectrum

The electromagnetic spectrum includes the entire range of all possible frequencies of electromagnetic radiation. This spectrum includes a wide scope of different waves, each characterized by its specific wavelength and frequency. Starting with the longest wavelengths and lowest frequencies are Radio Waves, Microwaves, Infrared Radiation, Visible light, Ultraviolet (UV) Radiation, X-Rays, and Gamma Rays which have the highest frequencies and shortest wavelengths. Radio waves are commonly employed for communication and broadcasting purposes. Moving towards shorter wavelengths are microwaves which are used in technologies like microwave ovens and certain communication systems. Infrared radiation follows, with wavelengths longer than visible light and are perceived as heat and used in applications such as thermal imaging. Visible light, the narrow segment perceivable by the human eye and includes different colors, is between infrared and ultraviolet radiation. Ultraviolet radiation has higher frequencies and is responsible for effects like sunburn and has application in sterilization. Beyond this, we encounter X-rays which are used in medical imaging and security screening. Gamma rays are the highest frequencies and the shortest wavelength and are emitted during nuclear reactions and utilized in medical treatments and certain scientific applications. The electromagnetic spectrum is a fundamental concept in physics, crucial for understanding the various types of electromagnetic radiation and their applications across a wide range of fields, from communication and medicine to scientific research.

Choosing the right wavelength to emit from the illumination source is another important consideration. Light has a wide spectrum of wavelengths and the light which we can see is called visible light. Visible light has many colors which have corresponding wavelength values. For instance, blue has a wavelength range of about 450 to 495 nanometers while red has a wavelength range of about 620 to 750 nm. Wavelength is an important consideration because we had the light from the piece and from the LED under the square all near the photodiode. The light from the illumination source cannot be the same wavelength as the light from the LEDs under the square since it could mess with how we measured the intensity of the light due to interference effects. The light from the LEDs would give false values that would jeopardize our design. To counteract this, we used photodiodes that are sensitive to the wavelength that we chose to have under each piece. If we are using photodiodes sensitive to a specific wavelength, it would mean that white light could not be used for the illumination system or the LEDs under the square. This is due to white being a combination of multiple wavelengths of different colors. What could be done is to use a single wavelength for the illumination source and not have the LEDs under the squares emit that wavelength or vice versa. For example, the LEDs under each square will show possible squares to move for piece or if someone is in check. Each of these will have a specific color so if we choose green to show potential moves of pieces, our illumination source cannot emit any green wavelengths since it will mess with our readings and blue or red wavelengths will have to be chosen. Another option is not using visible light for our light source which will enable us to use all of the colors the LEDs under the squares can make.

IR vs visible light LEDs

LEDs are versatile sources of illumination, capable of emitting a broad spectrum of wavelengths encompassing visible light and extending into the range of infrared radiation. LEDs that emit visible light and infrared light have some key differences such as their visibility, wavelengths, applications, energy levels and heating. Visible light LEDs operate within the visible spectrum for the human eye, producing an array of colors like red, green, blue, and more. This emitted light falls within the visible spectrum, which is the range of wavelengths in the spectrum that enables its detection by the human eye. On the other hand, infrared LEDs emit light beyond the visible spectrum, emitting light in the infrared spectrum, where wavelengths surpass the range of human vision. Notably, infrared light has longer wavelengths compared to visible light. Visible light LEDs find widespread use in applications such as for lighting applications, display technologies such as LED screens, indicators, and various decorative purposes. In contrast, infrared LEDs carve a niche in scenarios where the light needs to be invisible to the human eye. Some examples of these applications are in remote controls, security systems, and night-vision devices. The differences extend to energy levels, with visible light LEDs typically boasting higher energy levels than their infrared counterparts, which exhibit lower energy levels and longer wavelengths. Additionally, the heating aspect sets them both apart since visible light LEDs generate less heat, while infrared LEDs, particularly in applications demanding intense infrared radiation, tend to produce more heat.

3.3.2 Technology Selection: Light Source

The best illumination source would have the most optical power while still being as cheap as possible. This is very important because it would generate a lot of light making finding the intensities of the reflected light to be easier. There are many expensive sources that are bright enough, but those don’t fit into the budget constraints of this project. Another requirement is that the illumination source has to be small enough and weigh as little as possible to fit inside the cylindrical base under each of the pieces or not make the dimensions of the bottom of the chess board to be too big which would risk the portability of the board in addition to making the board too heavy to move around. Heat from the illumination system must be kept to a minimum since we are using plastic materials and having lots of heat can ruin them. Ideally, we would want to have no warmup time since it would be inconvenient for the player to wait until the illumination source is at full to play. The illumination source also shouldn't consume a lot of power since it would cut down on the battery life of the whole chess board system. A light source with a long lifetime would also be ideal since it wouldn't have to be replaced as often.

TABLE 7 LIGHT SOURCE TECHNOLOGIES MARKET

	LED	HeNe Laser	Laser diode	Quartz halogen lamps	Xenon metal halide lamp
Temperature (Heat)	Low	Medium	Medium	High	High
Warmup time	None	10 min	None	None	15 min
Size	0.98 x 0.2 x 0.2 in	1.74 x 1.74 x 10.70 in	0.23 x 0.55 x 3.54 in	3.07 x 0.32 x 3.07 in	2.1 x 5.3 x 2.1 in
Quantity in package	100	1	30	5	1
Minimum quantity needed	1	64	64	1	1
Operating current	<20 mA	6.5 mA	<20 mA	0.83 A	1.25 A
Weight	1.27 oz	0.92 lbs	1.55 oz	1.48 oz	2.08 oz
Output power	2 mW	2 mW	2 mW	100 W	150 W
Cost	$ 6.75	$ 1732.50	$ 12.99	$ 8.99	$ 20.71

After gathering all this information, the illumination source we decided on is the LED. Any type of laser with a small beam size would not make since the pieces would have to be perfectly aligned on top of the photodiode. If the piece with a laser light source was slightly off it could miss the photodiode completely or hit it partially giving us false values. LEDs disperse the light more than lasers however they are still directional. Having the LED light be more dispersed enable for the pieces to not be as precise giving the player some more wiggle room. LEDs are the cheapest among the sources listed above with the Quartz halogen lamps being the next closest. LEDs are the smallest weight with again the Quartz halogen lamps being the next closest. LEDs are also the smallest in size easily allowing them to fit under the chess board. The LEDS, laser diode, and Quartz halogen lamps have no warmup time which is great since no time will be wasted waiting for them to reach full power. They also generated the lowest amount of heat which would not affect our plastic components or housing. Laser diodes and LEDs have the second smallest operating current, only behind the HeNe laser although if we use less than the 20 mA and for example used 10 mA the LEDs will have a longer lifetime than if they used 20 mA. The drawback from this is that this would cause the LEDs to produce less light which wouldn't be the best. However, if more light is needed, we could add more LEDs since they are so small and energy efficient. The small operating current also means that they would consume less power allowing for longer battery life, which is ideal for the longevity of the system.

3.3.3 Part selection: LED illumination source

Now that we selected LEDs as our illumination source, we chose between which LED is best to use. Since our illumination source is required to be only one color, we only need LEDs that emit at a narrow wavelength range. LEDs of one color should be cheaper than that of multiple color due to semiconductor properties. However, if the price is close enough, we could use multicolor LEDs and just use filters to keep it centered on one wavelength. We would want LEDs without frosting on the outside because the light wouldn't be as dispersed and would be concentrated in one direction. This would be useful to maximize the amount of incident light directed towards the photodiodes. We would again want the cheapest LEDs while still being as bright as possible. Being as bright as possible is very important as it's the main reason for these LEDs because this will allow for a lot of light to hit the chess piece and be reflected to determine the piece type. Being bright as possible might contradict trying to have longer battery life since the brighter LEDs shine the more energy they would consume. We could use multiple LEDs that are dimmer to conserve more energy or spend more on more energy efficient LEDs to combat this issue.

TABLE 8 LED ILLUMINATION SOURCE MARKET PARTS

Brand	CO-RODE	Chanzon	CO-RODE	Chanzon	Ams-OSRAM
Size	0.197 x 0.197 x 0.689 in	0.228 x 0.228 x 0.728 in	0.197 x 0.197 x 0.689 in	0.152 x 0.152 x 1.161 in	0.197 x 0.197 x 1.043 in
Weight	0.79 oz	0.634 oz	0.79 oz	0.352 oz	0.634 oz
Max Luminous Intensity	15,000 – 20000 mcd	15,000 - 18,000 mcd	4,000 - 5,000 mcd	N/A	N/A
Wavelength	525 nm	515 nm	590 nm	850 nm	940 nm
Volts	3 V	3V	1.8 V	1.4 V	1.2 V
Quantity	100	100	100	100	50
Cost	$6.59	$5.99	$6.49	$8.99	$28.87

Based on the chart above the best LED to choose would be the Ams-OSRAM LED. This is because even though they cost the most, we get the highest current from the Ams-OSRAM LED on the photodiode, while also having the lowest forward voltage. The ALLECIN also has a max luminous intensity of 15,000 - 20,000 mcd but has only 10 LEDs that can produce this intensity since it’s evenly divided between 24 different values. This would mean we would pay more for less LEDs that would work for our project. Even though the Chanzon LEDs are the cheapest they produce the least amount of luminous intensity and ranges from 15000 to 18000 mcd. The viewing angle was 20 degrees meaning that the light is directional within our desired parameters. The forward voltage is 1.2 volts which is a low voltage and would mean low power consumption. The wavelength for the IR light emitted is 940 nm enabling us to easily separate the wavelengths of the RGB LEDs and the IR LEDs.

3.4 LED Array

A significant component of this project is the use of an LED matrix to convey information to the player. By using different colors, we want to convey valid moves and invalid moves by using a designated color. To confirm these good moves versus the bad ones, we would ideally need two colors. We do have stretch goals that make further use of other colors, so to have the flexibility to incorporate that in without much added difficulty would be most ideal.

Since a chess board is essentially made up of a grid eight squares wide and eight squares long, an LED has to be under each individual square. The total number of LEDs needed to place one under each square of the chess board will be 64 individual LEDs total. The challenge here would be to find the best way to connect these LEDs and control them to relay information to the players as they play through their game.

3.4.1 Selecting LEDs

Knowing that we wanted to at least communicate valid and invalid moves to the players, at least two colors were needed. We do have stretch goals to implement other colors into our project to serve other purposes, so we wanted to be able to include that option without too much added difficulty. RGB LEDs have built in diodes for red, green, and blue so each can be activated individually to achieve that specific color, but we can turn on combinations of them to achieve other colors. With all the possible combinations, we would have access to seven different colors total.

When researching RGB LEDs, there were decisions that had to be made about the LEDs themselves. The first one was deciding the casing the LEDs would come in. There were only two options: frosted or clear. With the clear casing, the light that would be emitted would shoot straight out from the diodes, so it resembled a laser. Knowing we wanted our squares to be evenly lit as possible, the clear casing was not the best option to accomplish this. With the frosted casing, the light would be diffused by the casing so it would give more of a glow effect. We can use this glow to our advantage when lighting up our board. Another significant detail about the two casings is their ability to mix colors. With the clear casing, when more than one diode is turned on, the colors would not be able to blend. With the frosted casing, the light from multiple diodes would have a chance to mix with the diffusion so we would have a more notable difference with the colors.

Another decision that had to be made about the LEDs were the configuration of the diodes themselves. There are also two ways the diodes could be configured: common anode or common cathode. As the names suggests, the portion that is common would all share a common node. After looking into different LED array configurations, we found it to be easier to implement common cathode RGB LEDs. To turn on common cathode RGB LEDs, the anode of the desired color would have to be connected to a voltage source while the cathode would be connected to a lower voltage source or a ground so that current flow through the diode.

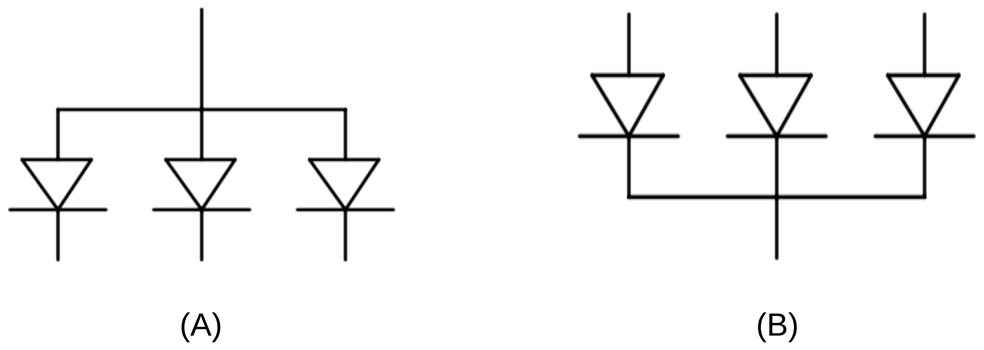


Figure 5 (A) Common Anode COnfiguration (B) Common Cathode Configuration

The table below shows the specifications of the LEDs that were selected. Each specific color diode has a different forward voltage so when selecting the resistor value to limit the current, we needed to consider both the forward current and the forward voltage for each color. To determine the limiting resistor value, we take into consideration the source voltage to the LED, the forward voltage of the LED, and the forward current of the LED. This formula shows how the limiting resistor is calculated:

$$R = \frac{Source\ Voltage - Forward\ Voltage}{Forward\ Current}$$

TABLE 9 LED CHIP TYPICAL ELECTRIC AND OPTICAL CHARACTERISTICS: (TA=25°C)

Items	Color	Symbol	Condition	Min.	Typ.	Max.	Unit
Forward Voltage	Red	V_F	I_F=20mA	1.8	2.0	2.2	V
	Green			3.0	3.2	3.4	
	Blue			3.0	3.2	3.4	
Luminous Intensity	Red	I_V	I_F=20mA	---	---	800	mcd
	Green			---	---	4000	
	Blue			---	---	900	
Wavelength	Red	$\Delta\lambda$	I_F=20mA	620	623	625	nm
	Green			515	517.5	520	
	Blue			465	466	467.5	
Light Degradation after 1000 hours	Red	-4.68% ~ -8.27%					
	Green	-11.37%~ -15.30%					
	Blue	-8.23%~ -16.81%					

3.4.2 LED Controller

One of the most difficult parts of the LED array is finding a way to turn on and off desired LEDs. In order to simplify the design of the matrix, the LEDs will be connected by row and column in a grid system. Each column would correspond to the anodes of each RGB LED so there will be 24 columns total across the whole matrix. Each row would correspond to the cathodes of the LEDs in that row. Below shows a 2 x 2 matrix showing these connections.

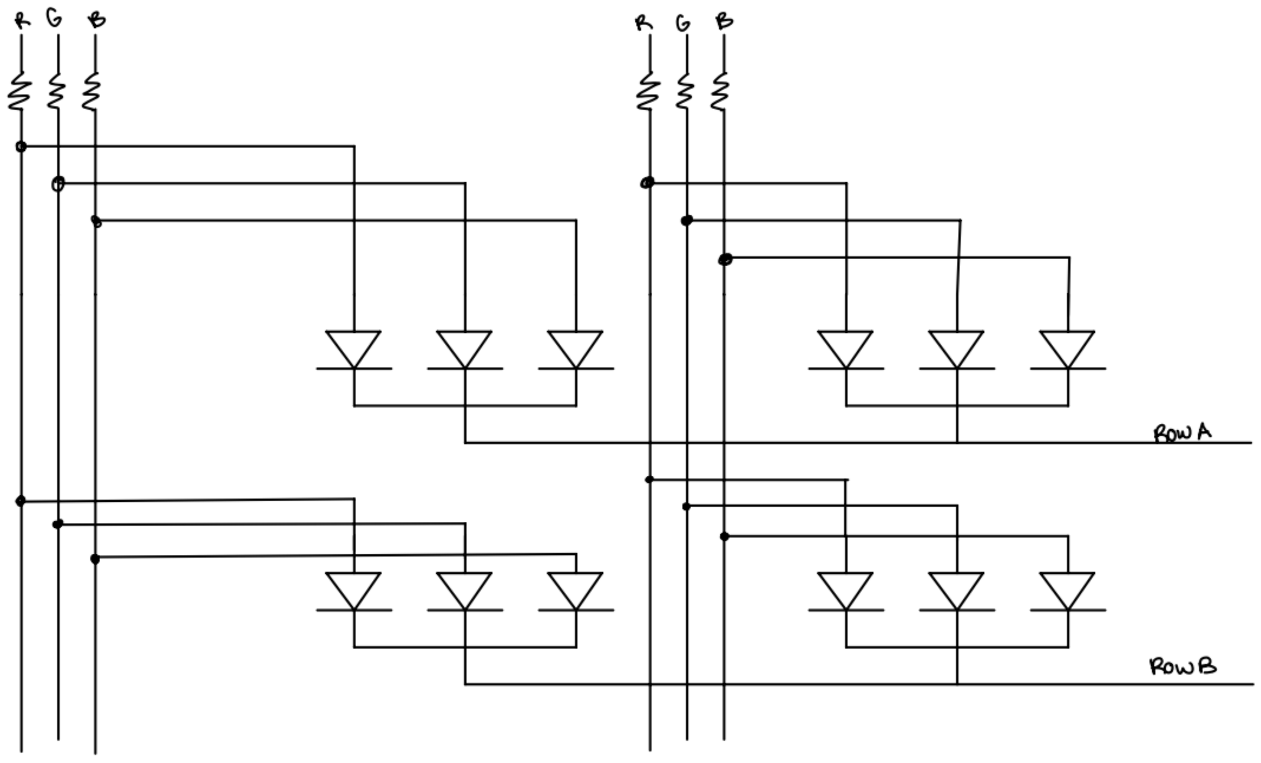


FIGURE 6 2 X 2 LED MATRIX

Knowing the method to turn on the diodes, we can then make assumptions about the connections that have to be made to this matrix. Each row must be connected to a ground or a voltage source lower than what is provided to the anodes while each column must be connected to a positive voltage source. With these connections, current can then flow through the diode, and it will turn on. There is an issue with this configuration though, LEDs that are diagonal from each other cannot be turned on at the same time. If this is attempted, the two opposite LEDs on the opposite diagonal will light on instead. To fix this issue, we must implement a way to turn on and off voltages to the desired rows and columns. For our rows, we can use NPN transistors to switch connect a ground to the desired rows. But with this limitation on being able to only turn on one LED at a time, we need to create a subsystem quick enough to trick the human eye. What appears to the human eye as all the LEDs in the matrix being on is actually singular LEDs turning on and off at a speed that is faster than what the human eye can detect. The flicker fusion threshold is the frequency at which the human eye sees a flickering light as a steady light. This threshold frequency varies among humans, but it can vary from 50 to 500 Hz. So, we need to find a controller that has a clock frequency that is at least greater than 500 Hz.

When researching LED matrixes, we found out there are multiple ways to control these arrays. Two of them in particular stood out: LED drivers and shift registers.

One of the factors that needs to be considered when selecting our driver is the packing of the integrated circuit itself. We want to be able to test the integrated circuit with a simple version of the matrix to ensure the correct component is selected. When looking into the different kinds packaging for an integrated circuit, there were three main forms our selected components came in: through-hole, surface mount, and chip carrier.

Chip carrier packaging is designed so that they can be plugged into a PCB or soldered whilst protecting the components inside the chip. When researching how to test this kind of packaging, it would have to mounted to a PCB that would then connect to a bread board for testing. It would take more time to order a PCB and solder the select the specific component and for it to not work in the end.

Surface mount components are designed with shorter leads so that they can placed directly onto a PCB without the need for through-holes. The leads on these components are significantly shorter compared to their through-hole counterpart. The leads on these components are also not at a length long enough to insert directly into a bread board for testing. Surface mount components would not be the most suitable option for testing either.

Through-hole components are pretty self-explanatory. These components are designed so that they can be inserted into a PCB and soldered into place. The leads on these components are significantly longer compared to surface mount components and the angle at which the leads end at are 180 degrees. The spacing between the leads on these components are also spaced enough to work with the spacing present on the breadboards. This kind of component housing would the most ideal for us since it will allow us to test the component before soldering it directly onto a PCB. The ability to test is crucial to ensure that we are selecting the best driver for our matrix and microcontroller.

3.4.2.1 LED Driver

LED drivers are designed so that constant current and voltage would be supplied to the LEDs so that none of them will overheat, flicker, change color unexpectedly, or not perform as expected. To accomplish this, LED drivers convert alternating current to direct current and some drivers even have the ability to shut off all the LEDs if the temperature reaches above a certain point.

Factors that need to be considered is the number of bits the LED driver can control at a time. Ideally, we want a driver that can control the number of columns of the LED matrix. None of the drivers we researched were capable of controlling a matrix of LEDs so we had to find a way to selected which row of LEDs will be turned on. We can do this by connecting the desired row to the ground or a low voltage source. The TPS929240-Q1 is an LED driver that is capable of controlling all 24 columns with one chip. It is also capable of controlling the brightness of the LEDs through PWM. For this project, we do not need to worry about the brightness of the LEDs because we want them to be able to shine through a clear material. The TPS92120-Q1 is the 12-channel version of the TPS929240-Q1. It has the same features as the TPS929240-Q1 such as the current sink, the PWM control, and the protections built in for the LEDs. The TLC59210 is different compared to the two previous drivers. The TLC59210 is an 8-channel driver that makes use of flip-flops. So, it has 8 input pins and 8 output pins. We only have a limited number of pins on the microcontroller that will be used so using this driver would not be ideal. To implement it into our design and still conserve microcontroller pins, we would have to pair the TLC59210 with a shift register.

Another factor that needs to be considered is the output characteristics of the output pins. In order to control the columns, the drivers must be able to output enough current which is at least 20 mA. The LEDs that were selected have a forward output current of 20 mA. With the TPS929240-Q1 and the TPS92120-Q1, they are both capable of meeting this demand of 20 mA. They can both output current about four times greater than the 20 mA required for the LEDs. Both of these drivers would be suitable to control the columns of the matrix. The TLC59210 is different compared to the two other drivers though. The TLC59210 is designed as a current sink circuit. Current sink components are meant to intake current from the output pins. This current sink feature would not be ideal to control the columns of the matrix but it could still be implemented to control the rows of the matrix. Another method would have to be paired with the TLC59210 to control the columns.

A factor that is not essential to consider but would make calculations easier is the amount of voltage output can be provided by each pin. By knowing this source voltage to the diode, the forward voltage of the LED, and the forward current of the LED, we can calculate the resistor value that would be necessary to provide the highest illumination possible. By using the formula below, we can calculate the resistor value necessary for each color of LED. V_s is the supply voltage, V_f is the forward voltage, I is the forward current of the LED.

$$R = \frac{V_s - V_f}{I}$$

TABLE 10 LED DRIVER COMPARISON

	TPS929240-Q1	TPS92120-Q1	TLC59210
Number of channels	24	12	8
LED current per channel (mA)	100	75	200
Vin min-max (V)	4.5-40	4.5-40	3.3-5.5
Vout min-max (V)	0-40		
Maximum Frequency	1 MHz	1MHz	
Extra Features	Current source Enable/shutdown FlexWire control interface	Current sink PWM control Thermal shutdown	Current sink Thermal shutdown PWM control
Package Type	HTDDOP (surface mount)	HTSSOP (surface mount)	PDIP (through-hole) TSSOP (surface mount)
Price	$3.280	$1.742	$0.697

3.4.2.2 Shift Registers

Shift registers are a type of sequential logic circuit that can be used to store or transform binary data. Each shift register is made up of multiple flip flops designed to hold multiple bits of data. The way a bit of data is sent to the corresponding register is by making use of clock pulses to push each bit individually until it arrives at that register.

One of the features of these shift registers is the ability to daisy chain them together to create a larger shift register that is capable of controlling more bits. Daisy chaining is where the output of one shift register is directly fed into the input of the next one. So, if two shift registers are daisy chained together and are sent 16 bits of data, the first shift register will take the first 8 bits and process them and push the 8 remaining bits to the next shift register. This process would be the same with 3 shift registers and even larger amounts of daisy-chained shift registers. With the ability to daisy chain, if we cannot find a decent shift register large enough for our project, we can make one that meets the specifications we need. To control the entire array, 24 pins would be needed to control the columns and 8 pins would be needed to control the rows of the array. In total, we would need at least 32 pins to control our LED array.

Ideally, we would be using the shift registers to source current from each of the output pins.

When selecting shift registers, a key feature is for it to draw enough current for the diodes. Each diode for the LEDs needs about 20 mA to be turned on. Most shift registers out there cannot supply enough current to completely turn on a row of LEDs so we can use these shift registers to turn on switches to power the diodes. There is also another issue with how the shift registers would power the rows and columns. Due to this current power limitation, we can only turn on one LED at a time. When we need to be able to light up multiple squares, this would be a problem. There is a trick to resolve this problem. The human eye can only see about 30 to 60 frames per second. We need to find a way to quickly alternate turning on single LEDs so that it appears to our eyes that all of them are on at once. So, we would need to use a shift register that is fast when processing data.

When researching shift registers, one of the major specifications needed for our project is a serial-in to parallel-out (SIPO) configuration. This configuration takes in a single line of input, let us say 16 bits of data for now, and transforms it so the designated bit goes to the corresponding register. With this configuration, we are able to control all of the rows and columns at once. Ideally, we want a shift register to control both our rows and columns, so 32 bits total would be needed. When looking into 16-bit shift registers we found one that is in the SIPO configuration. The one issue is that a single unit would be $31.592 so for two of them it would be about $63. Compared to the price of getting enough 8-bit shift registers to control 32 bits of information, the price for 4 8-bit shift registers is significantly lower.

With the 8-bit shift registers that we found, they also varied in their specs. We selected two to compare against: the SN74HC595 and the TPIC6A595. The TPIC6A595 is capable of drawing more current, and this value actually goes higher with less outputs on. The SN74HC595 meets the current requirement for our LEDs because it can draw a maximum of 35 mA for a singular output. One of the deciding factors here though is the speed of the clock. We need a high clock speed when controlling the matrix and the SN74HC595 has the best clock speed for its price.

TABLE 11 SHIFT REGISTER COMPARISON

	SN54LS673	SN74HC595	TPIC6A595
Number of Channels	16	8	8
Max Current (mA)	12	35	350
Vin min-max (V)	4.75-5.25	2-6	4.5-5.5
Clock Speed	20 MHz	24 MHz	10 MHz
Price	$31.592	$0.073	$1.150

3.4.2.3 LED Driver vs. Shift Registers

One of the final decisions that needs to be made is whether to go with the LED driver or the shift registers when controlling our LED matrix. Ideally, we need the driver to be quick enough to light up the correct squares for our players almost immediately after a piece is identified. The problem with the matrix is making sure the incorrect LED is lit due to latency with the selected driver. Depending on what driver we select, we might be able to only power one LED at a time. We can create the illusion that multiple are lit at once by taking advantage of the speed of the driver to alternate quick enough. This timing restriction limit is what will help us decide between these two options. We couldn’t be able to have a final decision until we get time to test all of our components together and see which driver will give us the most consistent results.

3.5 Display Screen

There are a lot of factors to consider as we start looking for displays to use for our project to select the best option. In addition to showing game time and providing useful information to the players, the display will also act as a general graphical user interface for navigating menus and configuring the game.

3.5.1 Display Requirements

This section will go over the display requirements and characteristics we put into consideration when choosing a display to implement in our design and put forth some available products for comparison.

3.5.1.1 Size and Resolution

The size of our display is high on our list of priorities. Naturally, given the limitations on the device's bulk and dimensions, we must select a size that fits the form factor of our project without sacrificing the display's quality or readability. This also considers the resolution of the display, since displays with higher resolutions offer images that are clearer and more detailed.

3.5.1.2 Refresh Rate and Response Time

The display's response time is still a point of contention, despite being less significant for our design. Since our application won't be displaying any fast-moving images or videos, we don't need exceptionally high refresh rates on our screen. Nevertheless, it's crucial that the display remains responsive to our program's requirements to give the user a positive experience.

3.5.1.3 Power Consumption

The display will probably be one of the most power-intensive parts of the design, so knowing how much power it uses is crucial considering that our project will run on a battery. Most of a display's power consumption goes toward lighting the backlight, so it's critical to select a display with a reasonable power consumption without sacrificing brightness or image clarity.

3.5.1.4 Support

This step serves to ensure that the display is compatible with the microcontroller or processor used in our project. If the display also includes software libraries, development kits, or drivers that make integration with our system easier, we could save a significant amount of development time.

3.5.1.5 Cost

The cost of the display is a significant factor, especially on the budget constraints for our project. Careful cost management contributes to the success of the project, making it important to find a display that meets your requirements at a reasonable cost.

3.5.1.6 Touchscreen Capabilities

Finally, one characteristic that is not essential but is also of note is the possibility of using the display as a form of user input. We planned to use buttons on the design beside the display for controlling user input, however, if the matching display contained a good touchscreen, it would make for much sleeker design.

3.5.2 Display Choice

Given all the requirements described above, we need to analyze the different types of displays available and decide on which one best fits our needs.

3.5.2.1 Seven-Segment Displays & Dot Matrices

There is little reason to consider these two display types, as they are far below our desired specifications. Seven-segment displays are far too limiting on the type and quantity of information we can display, and dot matrices have incredibly low resolution due to their nature, as they are essentially a group of individual LEDs much akin to what we are trying to replicate on a larger scale with our own LED matrix for the chessboard lighting.

3.5.2.2 OLED & AMOLED Displays

OLED stands for Organic Light-Emitting Diode, these use an organic, carbon-based material for emitting light, meaning they lack the necessity of backlighting. This category of displays allows for superior image quality but consumes more power and usually provides worse visibility in bright light. “Hobby-Oriented” OLED and AMOLED displays for microcontroller implementation are also very expensive when compared to other display types and are usually found in smaller-sized packages, which are not exactly ideal for the type or amount of content we plan to display.

3.5.2.3 LCD Displays

Probably one of the most common displays used for general applications, the traditional liquid crystal displays are simple, demand low power, and are considerably cheaper than the OLED displays discussed above. LCD displays vary wildly in their design depending on its intended use, however, displays used for microcontroller integration tend to have limitations, such as only being able to display a small number of characters at a time, and being unable to display full images. For those reasons, LCD will be unfit for this design.

3.5.2.4 TFT LCD Displays

TFT LCDs use a thin film of transistors rather than a single layer of transistors like traditional LCDs. This results in improved image quality, faster response time, and lower power consumption. TFT LCDs are also thinner and lighter than conventional LCDs, making them ideal for use in mobile devices, like our chessboard.

TFT displays are capable of reproducing images and are often cheaper and have a larger screen size over the other contenders. Not only that, but some manufacturers even include touchscreen capability, which, as discussed in the requirements section above, although not something that is obligatory, it is appreciated and would expand the possibilities of our design and overall contribute to the product’s presentation.

All these qualities make the TFT displays a great choice for our project.

3.5.3 LCD Market Research

Recognizing the dynamic nature of the market, after choosing TFT displays as the ones to be used for the project we deemed it essential to conduct a comprehensive market search specifically focused on TFT (Thin-Film Transistor) displays.

TABLE 13: LCD PART COMPARISON

	HiLetgo 2.8" SPI TFT LCD Display	HiLetgo 3.5" TFT LCD Display	Hosyond 4.0” TFT LCD Display
Screen Size & Resolution	2.8" 240 x 320 pixels	3.5” 480 x 320 pixels	4” 480 x 320 pixels
Operational Voltage	3.3V~5V	3.3V~5V	3.3V~5V
Expected Current Consumption	120mAh	150mAh	180mAh
Driver IC	ILI9341	ILI9488	ILI9486
Communication Protocols	SPI	SPI Capability (Not Implemented)	SPI
Touchscreen	Yes (Resistive)	No	Yes (Resistive)
Cost	$16.39	$18.49	$19.99

All three displays seen on the table were all relatively inexpensive in both their initial hardware and power consumption costs, which are estimated at only a 1Wh power draw for the largest display, less than the expected consumption of the LED grid we used to light up the chessboard. For their drivers, all three displays use relatively well-known IC chips with decent library support and are capable of SPI communication, except for the 3.5” display, which has the capability to perform SPI but does not have it integrated in the module by default.

The clear choice among our options was the Hosyond 4.0” Display, however, we did find ourselves a bit limited on the size of the display. A large majority of displays with 5” or bigger screens use HDMI channels and are overall harder to introduce into the design without significant upgrades to additional components, which would drastically impact on our budget. To make up for the smaller size, it is possible to build the display in an angle, as opposed to flat on the chessboard to facilitate reading.

3.6 Power Management

When deciding on a type of battery for our design, we are faced with two options that are the most popular in the market, lithium polymer pouch cells (LiPo) and lithium-ion cylindrical cells (Li-ion). Other types do exist, such as nickel metal hydride batteries, which do not meet our energy needs, and lead acid batteries, which are unnecessarily heavy, bulky, and expensive for use in this application.

3.6.1 Lithium-Ion vs Lithium Polymer Batteries

Since we already determined that the other kinds of batteries do not attend our project's requirements, in this section we analyzed the different characteristics of Lithium-Ion and Lithium Polymer batteries to determine which will be the best fit for our design.

3.6.1.1 Physical Properties

Lithium polymer batteries are a winner in this category. Contrary to the typical steel hard casings of their conventional lithium-ion counterparts, LiPo batteries are usually manufactured in a flexible casing, which are much lighter and have a much smaller footprint.

3.6.1.2 Power Characteristics

Even though, as mentioned above, lithium polymer batteries are usually much lighter than lithium-ion cells, when accounting for their energy density (Wh/Kg), lithium-ion batteries pull out ahead, able to store larger amounts of energy for less added weight, allowing for longer runtimes in comparison to LiPo cells.

3.6.1.3 Safety

Safety was a major concern over the choice of batteries, both for the safety of the designers and the users. Lithium-ion batteries tend to be safer than lithium polymer pouches due to their hard metal casing exterior, making them more resistant to impact and puncture, which is going to be great project to make it safer during transport and usage, since we are aiming for portability. [41]

3.6.1.4 Cost

Li-ion wins again. Although comparing individual cells show that lithium polymer is cheaper, we must consider that, to achieve the same capacity we would have with li-ion, we would need a larger amount of battery cells, which would impact on our budget.

The table below summarizes our discoveries discussed above in a clear and concise manner that will be useful to reference to later.

TABLE 12 BATTERY COMPARISON

	Lead Acid	Ni-MH	Li-Ion	Li-PO
Cycle Life (Cycles)	200 - 2000	500 - 1000	500 – 2000	> 1200
Efficiency (%)	70 - 90	70	75 - 90	70
Energy Density (Wh/Kg)	30 - 40	30 - 80	100 - 250	130 - 200
Weight	Heavy	Medium	Light	Lightest
Total Cost	Low	Medium	High	Medium
Toxicity	Very High	High	Low	Low

3.6.2 Battery Pack Design

Probably the most crucial part in our power design. Due to the need for convenience and portability of the design, it is important that the board doesn’t rely on wired connections to an external power supply to maintain operations. The solution to this issue is simple and effective: to implement a battery pack comprised of 18650 lithium-ion battery cells into the design.

There are several benefits to this approach, but in this case, these cells are particularly advantageous due to their high energy density, that is, their ability to store a large amount of energy in a relatively small package. Not only that, but by grouping together individual cells into a custom power bank makes it simple to alter the design in later stages of development.

Once the voltage (Volts) and capacity (mAh) requirements of the system are established, it is possible to assemble a battery pack that meets the specifications by connecting these cells in parallel to reach the desired amperage and afterwards, connect such parallel groups, which we called modules, in series to achieve an appropriate voltage.

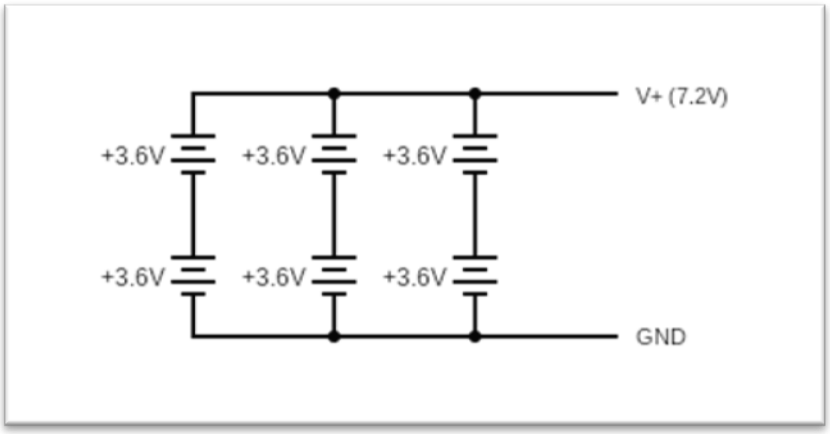


FIGURE 7 BATTERY PACK CONFIGURATION

$$\text{(A)}\quad \#\, \boldsymbol{of\ Parallel\ Cells} = \frac{\boldsymbol{Desired\ Pack\ Capacity}}{\boldsymbol{Cell\ Capacity}}$$

$$\text{(B)}\quad \#\, \boldsymbol{of\ Serialized\ Modules} = \frac{\boldsymbol{Pack\ Nominal\ Voltage}}{\boldsymbol{Cell\ Nominal\ Voltage}}$$

FIGURE 8 (A) BATTERY EQUATION FOR NUMBER OF PARALLEL CELLS
(B) BATTERY EQUATION FOR NUMBER OF SERIALIZED MODULES

3.6.3 Power Estimations

The LED matrix used on the board is, by a large margin, the most demanding component power-wise. Assuming an average current draw of 20mAh per LED on a voltage output of 5V, we can estimate, based on average range of 12 to 17 LEDs concurrently lit, that the total power consumption of the LED array will range between 1.2 and 1.7 Wh.

The LCD display comes in second place for power consumption, drawing an expected current of 180mAh at 5V, resulting in an expected power rating of 0.9W.

The remaining components are mostly used in low-power applications, so it's expected that they contribute little to the overall operational demand. However, to account for these other components and give the project some leeway in the upcoming phase of prototyping and development, we considered the total power consumption of the project to range between 3.5 and 4.5 Wh, although since a large part of our project ended up working at 3.3V, the consumption may be even lower than initially thought.

To have enough energy on our battery to supply to the project, we need to set expectations on the length of time we want the battery to last. Assuming a minimum duration time of 6 hours, we can multiply that number by the hourly power draw of the circuit to determine that our battery will need a total of 27 Watts. Easily achievable with our battery design.

The initial plan for our battery will be a 2S3P design, meaning it will hold two groups of three 18650 cells each. Considering each cell has a nominal voltage of 3.6 V and a capacity of 2500 mAh, this design will make a 6-cell battery pack with a nominal voltage of 7.2 V and a total capacity of 7500 mAh, which totals for a 54 Watts battery, which should easily be able to maintain the demand of the system for much longer than our 6-hour minimum.

3.6.4 Battery Management System (BMS)

A Battery Management System (BMS) serves to ensure safe and efficient operation while running our product on battery power. This system is essential for the type of battery we are using, as it is designed to monitor, control, and safeguard our battery.

In this case, our management system will oversee the balance of charge across all cells, making sure they charge evenly throughout. Without such a system in place, some cells could possibly charge up faster, as not every cell is chemically identical, which would lead misalignment in the voltages of the cells, and ultimately to the destruction of the battery due to overvoltage.

Additionally, other protection mechanisms are included in systems like this one, such as short circuit protection, as well as protection from overcharge, overdischarge, and overcurrent. In essence, a BMS is crucial for this design and will prove itself to be very useful for dealing with any safety concerns regarding our device’s power source.

Today, we can find several boards in the market with many of these protection circuits already pre-installed. Displayed in the table below are a few examples of model protection systems made specifically for 2S battery packs like the ones we used. All the listed models below possess the same protection circuitry listed above as requirements, however, we decided to go with the JH20 due to its inbuilt charge balancing system, which the others lack.

TABLE 13 BMS MODELS COMPARISON

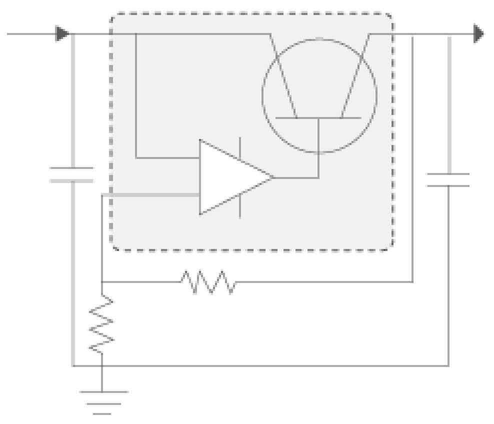
	HX-2S-A10	HX-2S-JH20	HX-2S-D01
Charging Voltage	8.4V-9V	8.4V-9V	8.4V-9V
Upper Limit Operating Current	8A	10A	8A
Dimensions	41 x 8 x 2.2 (L x W x H in mm)	46.7 x 23 x 3.15 (L x W x H in mm)	40 x 17 x 3.5 (L x W x H in mm)
Charge Balancing	NO	YES	NO
Cost	$9.49 / 5 ($1.898 ea)	$11.99 / 5 ($2.398 ea)	$6.99 / 2 ($3.495 ea)

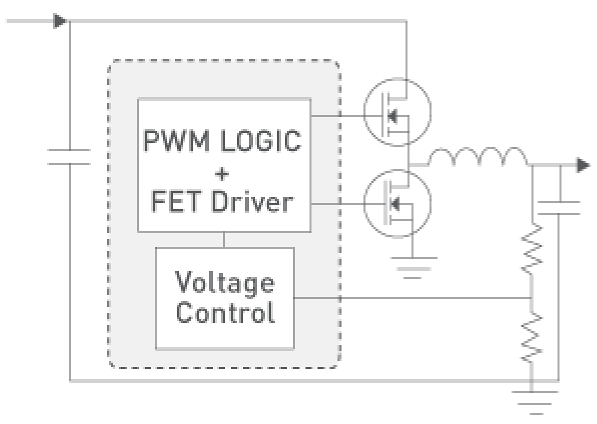
3.6.5 Voltage Regulators

Voltage regulators are extremely useful devices. They are a type of component in a power supply unit that ensures a constant voltage supply through all operational conditions. It regulates the device's voltage during cases of power fluctuations and load variations. This project will require consistent and stable voltage levels during the device's operation. This will prevent damage to sensitive electrical components due to overvoltage, prevent undervoltage conditions that can lead to improper operation or data loss, as well as increase overall power efficiency and battery life.

Although the general use for voltage regulators is to maintain the output voltage at a specified amount, there are multiple approaches to achieving this. Thus, there exists two types of voltage regulators, called switching and linear regulators respectively.

Linear regulators are an excellent choice for powering low-power devices. However, despite its simplicity, and low cost, linear regulators are typically inefficient, as they dissipate a great amount of power as heat during operation, making them inadequate for use on a battery-powered device. Meanwhile, switching regulators, are highly efficient and are available in the market in compact and reliable modules that will aid us in the implementation process. [40]


(A)



(B)

FIGURE 9 (A) LINEAR VOLTAGE REGULATOR CIRCUIT AND (B) SWITCHING VOLTAGE REGULATOR CIRCUIT

The table below outlines some important distinctions between Linear and Switching regulators. It summarizes the important characteristics we did need to consider when acquiring the voltage regulators.

TABLE 14 LINEAR AND SWITCHING VOLTAGE REGULATOR COMPARISON

	Linear Regulators	Switching Regulators
Design Availability	Buck	Buck Boost Hybrid (Buck-Boost)
Efficiency	Low	High
Complexity	Low	Medium-High
Total Cost	Low	Medium to high due to cost of external components
Input Voltage Range	Small Range	Wide Range

In our design we used a buck converter, which is utilized when the DC output voltage required is lower than the DC input voltage. The ones we used are a very commonly used type of switching circuits for voltage regulation.

The circuit of a Buck converter and its operation are both very simple. The circuit consists of a switching transistor, together with what is called a “flywheel circuit” a closed loop involving an inductor, a capacitor, and a diode. While the transistor is on, current is flowing to the load through the inductor. As any with any inductor, this component will oppose changes in current flow and act as a small energy storage. The inductor stores energy when the transistor is on and when it turns off, the diode connected in parallel with the load allows the energy stored in the inductor to flow to the output, preventing a voltage spike and ensuring a continuous current. This system is further enhanced with a parallel-connected capacitor connected at the output to smooth out any remaining ripples in the output voltage and provide a stable DC voltage to the load. [42]

We are using these, as opposed to boost converters because our battery will have a larger voltage than the 5V we used in the system. The basic topology of a buck converter can be seen in the image below, do note that the finished model contains control circuitry to maintain the proper operation of these circuits.

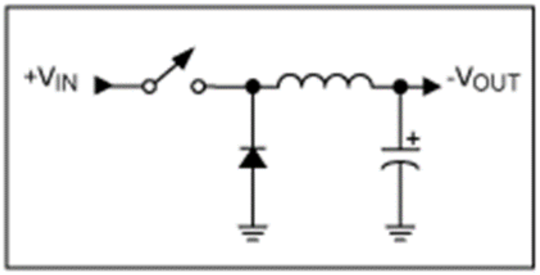


FIGURE 10 BUCK CONVERTER CIRCUIT TOPOLOGY

The buck converters we used in this project are found readily available in the market in the form of small self-contained and ready-to-use modules. They utilize LM2596 series chips, a collection of integrated circuits that provide all the active functions necessary for a step-down switching regulator. These chips can drive a 3 Amp load, which should be well over our load requirements for this chessboard.

These regulators are easy to use, requiring only a minimum number of external components and are available in both fixed and adjustable output version. We used the latter for this product even though our design runs exclusively on 5V, we'd like the flexibility to add components that utilize different voltage levels in the future should the need arise.

The main concern over the use of this buck converter was that it would need a higher voltage battery to be able to properly stabilize the voltage because, when completely discharged, our battery will have approximately 5.8V which makes the difference between output and input voltages in the buck converter relatively small. However, upon testing, the voltage was properly regulated even with voltage differentials between input and output were as low as 0.7V, which means we are safe to use a 2S battery pack that will fluctuate between 5.8V and 8.4V when fully discharged and fully charged respectively.

3.6.6 Battery Recharging

While first envisioning the process for recharging our batteries, we thought of many ways to possibly do it before settling in with our current solution. We had first considered utilizing replaceable batteries, which the user would take out and individually recharge each cell after a certain period of use but considering the number of individual cells and the overall construction of our project, that solution would be a big hassle to maintain through the entire period of testing as well as during operational use. That’s when we realized we needed a friendlier and more efficient way to keep our project’s energy requirements in check, so we kept looking for a more ideal solution. Thankfully, we came upon an easy alternative for charging our batteries, making for an exciting discovery.

USB-C is a connector format compact enough to fit in small of devices like smartphones, while still having enough power transfer capability to charge bigger devices like laptops. This implementation would be ideal, simply for the convenience that would come with not having to use a specific charger to power our board but be able to use an USB-C connector, which can be found mostly anywhere as of today. And thus, we turn to USB-C PD, or USB-C Power Delivery, an extremely handy tool. It is a popular technology that has only grown larger in the last few years, completely changing the way on how portable and handheld electronics are charged since its introduction.

Power Delivery allows for devices to request from the charger the exact rate of charge they need before transmitting and receiving power, that ensures that the same 100-watt PD charger that can deliver enough power to charge a large device such as a high-end laptop, can charge a 20W cellphone battery without destroying its systems. To allow the use of this technology in our project, we had to first figure out a way to first be able to have a USB-C input. Therefore, we purchased a USB-C PD “trigger board”, which is responsible for providing an USB-C compatible input from the charger. These PCBs have a minuscule footprint, and are extremely cheap, bought for only 98¢ each. They are responsible for enabling the PD functions of the USB-C adapter and allow our project to be charged through it. The trigger board we got is capable of selectively go through a varied range of different voltage levels, but we used 9V for our battery charging purposes.

Unfortunately, these boards did not survive to the final design. We had problems in the implementation of USB-C due to its bidirectional nature, forcing us to adapt the charging circuit to make sure the board was only used as input, because otherwise our battery would be drained instead of charge when plugged in. To make it worse, the chip in the trigger boards ended proving itself unable to handle the final design of our battery. It could manage the smaller prototype battery just fine, however, when attempted to charge the bigger battery for the final design, the trigger board would completely shut down. Thus, we moved on from the idea of using USB-C for charging and ended up utilizing a common 9VDC adapter for charging. This adapter can provide us with 2A at 9VDC, which is just enough to maintain a good rate of charge through all our cells.

3.7 Chess Engine

When beginning to research what Microcontroller to select, a few different decisions could be made regarding functionality of the product at the end. A fully implemented chess engine will add features such as best move selection, and an evaluation of the current position, but it may ultimately affect the microcontroller’s selection as specifications have to be selected based around this software component, which will be the most CPU intensive. This portion of research was approached with an open mind to see if it was possible to easily implement a chess engine, and not give too many restrictions to the rest of the project if these features are implemented.

3.7.1 Chess Engine ELO

ELO is one term used often to compare the strength of chess engines. ELO is a rating system meant to compare the relative strengths of players in game, and chess often uses this rating to compare the best players. Your ELO rating increases or decreases based upon wins or losses to opponents, with each increase / decrease based on the difference in ELO before the match. Typically, a player with an ELO 100 points above their opponent would win about 64% of their games, while a player with an ELO 200 points above their opponent would win about 75% of their games. Magnus Carlsen achieved the highest ELO ranking ever for classical chess with a rating of 2882 in 2014.

Chess engines also can ELOs associated with them, determined by play with other chess engines or against human players. Ideally for this system, the higher the ELO of the chess engine the better the engine would be at showing the player the best moves. A chess engine with a very low ELO could provide moves that don’t help the player and hurt the functionality of it. Of course, with players of a high enough ELO using this board, the ELO of the chess engine would have to be still a few hundred points higher than them to guarantee best move selection, and only select chess engines have that high of an ELO. I chess engine with a lower ELO would still likely be better than a vast majority of its users, but not have as high of an accuracy in best move selection.

3.7.2 Chess Engine Selections

3.7.2.1 StockFish

StockFish is the most powerful chess engine to date, winning the “Top Chess Engine Championship” 14-times. It is open source and written in C++, compatible with Windows, Mac, and Linux, which means for this to be placed on a microcontroller, it must be a microcontroller that can have an operating system on it. StockFish exists in 16 versions, where the latest versions generally have greater strength but larger file size and memory usage. Earliest versions of this are under 400 KB, and the most modern versions are about 40 MB. On top of that, I found data sheets with different computer’s performance, and it is safe to say that a much less powerful processor, like one that would be in a microcontroller, would struggle to run StockFish at any level. StockFish’s latest version’s ELO is approximately 3200, and in some cases scoring up to 3400+ in rating, which would be stronger than every human chess player to date.

3.7.2.2 Micro-Max

In opposition to StockFish, Micro-Max is an extremely lightweight chess engine that runs on C, known as the smallest C Chess program in existence. This option would function well on a microcontroller, with only having 1433 characters total. However, the greatest weakness of this chess engine is it's ELO, with recorded ELO of about 1950. This would place it at about the 99.7 percentile out of all USCF (United States Chess Federation) members, meaning that a good majority of all chess players would lose to this engine, but there will still be exceptions. The code provided doesn't initially support being able to load a game at any position other than the starting one, but with some modification that aspect shouldn't be much more difficult to emulate.

3.7.2.3 Tom Kerrigan's Simple Chess Program (TSCP)

Like Micro-Max, TSCP is a lightweight chess engine that runs on C, with a very low ELO of about 1700. It comes in at 2,248 lines of code, but unlike Micro-Max, it is written in a way designed to be very easy to read and teach people how chess engines work. Not only that, but it has an opening book available to emulate an actual player towards the start of the game. Being able to load a position on the board as well as evaluation of any position is also already there and would just have to be slightly modified to be able to use that data ourselves. With its low ELO, it would place at about the 99.3 percentile out of all USCF members.

Through experimentation with this engine itself, its features extend beyond just analysis of a game or a computer opponent, but it can also organize and hold all the information for both sides to play the game. Rather than having to program the entire logic for each move, create systems to store the positions, etc. this program could do that all for us, and easily be modified to work with the rest of the micro controller. This is a huge strength as it will speed up implementation immensely.

One potential issue with TSCP though is that TSCP is copyrighted and needs explicit permission to use the work. However, this issue was quickly resolved by reaching out to the owner of TSCP, Tom Kerrigan, and we got approval to use TSCP if we make it clear that we did not write TSCP. We also cannot be profiting off or redistribute TSCP, but neither of which will happen in the scope of this project.

3.7.2.4 GNU Chess

GNU Chess is a somewhat of an average of the previously found chess engines. It has a very high ELO of 2660, which places it in the 100.00 percentile out of all USCF, and the 99.97 percentile of all non-scholastic USCF. This means that it is extremely unlikely anyone would beat this bot, except for a handful of super grandmasters out of the entire world. This alone will make it extremely accurate for best moves detection and evaluation. In addition to that, the file size isn't too unreasonable with only taking 3.2 MB of space, which shouldn't be too big of an issue on most microcontrollers.

However, it has a few weaknesses, one of which being that it must use a Linux operating system to run. Requiring an operating system will cause many more constraints on the project, as it will become more difficult interacting with hardware and keeping high battery life, so this a major downside to this powerful engine. This engine appears to be one of the best options though for a device with an operating system though, as it is more light-weight than StockFish, and having it run on Linux, the most light-weight operating system, would likely make it the simplest to implement on that type of system.

TABLE 15 CHESS ENGINES

Qualities:	StockFish	Micro-Max	TSCP	GNU Chess
Approximate ELO	3200	1950	1700	2660
File Size	40 MB	6 KB	160 KB	3.2 MB
Requires OS?	YES	NO	NO	YES
*Load Position Feature?	YES	NO	YES	YES

**This quality is true if the given chess engine can load up a chess position rather than only being able to begin from the starting position.*

3.7.3 Chess Engine Conclusions

Based on the research completed on the previous engines, the best choice for a chess engine to be ported on for this project would be Tom Kerrigan's Chess Program. While it has the lowest ELO out of the selections listed, the primary focus of this project is to help beginners learn and play chess, so any ELO as "low as" 1700 is still very high compared to the average beginner. Its readability, and light weight design will make it very easy to implement into this project and become a tool to help implement the many other features outside of just the chess engine aspect of it. Stockfish may have similar features but would be far too demanding on the hardware to be able to run at all. Micro-Max is stronger ELO wise and could fit easily on a microcontroller, but its "minimalist" code style makes it far too difficult to modify for our usage.

I initially began research into this area just to see if this feature was possible to implement, and what restrictions that would impose for the microcontroller selection, but with this engine I see how this would help implement many of the core aspects of this project, and therefore is the best choice and should be implemented regardless of if the analysis features should be added or not. In addition, the specifications of the microcontroller should not be affected much by the selection of this component, if it has storage capable of holding the 160KB file and it can run C code, but the stronger the processing power is the faster and better the engine will run.

3.8 Microcontrollers

3.8.1 Microcontrollers or Single Board Computers?

When beginning to research different microcontrollers to implement for this project, I noticed a large difference in some of the specifications of the devices. Some devices claiming to be at the top end of the industry only had CPU speeds less than a GHz, while I would find other devices from other companies with CPU clock speeds in the multiple GHz. This difference led to the discovery that some of these devices weren't really microcontrollers, but something different known as Single Board Computers (SBC).

SBCs are mainly different from microcontrollers in the sense that they are intended to run Operating Systems, usually Linux. As a result, the specifications for those devices do need to be much higher to be able to support many more libraries of code, and as a result, extended functionality. This would make running more complex code easy, as you can program in a variety of languages that all compile down, with a simpler interface, however this comes with the cost of making access to the hardware components on the device much more difficult interact with. Microcontrollers have very easy access to this without an operating system.

Given the complexity of the hardware stack for this project, that difference alone is enough to make the Microcontroller the clear option for this project. Furthermore, power constraints will also make the microcontroller the winner. Since SBC's have OS running always, the microcontroller can have far more efficient low-power modes, and therefore will allow for longer battery life for our device. Lastly, as researched in the previous section, an operating system can be avoided by selecting a chess engine that runs on C, which was found with TSCP. This project will use a microcontroller.

3.8.2 Microcontroller Specifications

When selecting a microcontroller to use for this project, there are many different specifications to consider that will ultimately cause major issues for the implementation of this project if not selected correctly. Taking into consideration the hardware stack needed, specs such as storage, RAM, CPU speed, GPIO pin count, and other specific functionalities will have certain minimum requirements to make everything work. In addition to that, minimizing cost of this component will also remain a priority, but it must be done so without sacrificing functionality.

Flash storage on the microcontroller is the first benchmark worth considering. This one should not be very restrictive for the project, as the chess engine itself selected from the previous section only will take up a few hundred Kilobytes. With having a display and menus added, there will need to be more data stored, but it is safe to say that everything should be able to fit on the microcontroller with 2MB of flash storage. This is an estimate that will mostly determine on the display graphics and drivers for what needs to be stored. The chess engine will only be a 160 KB file, so it is possible for the program to be stored under 500 KB, but a 2MB max is good realistic upper bound.

RAM size and CPU speed are both important specifications that the stronger they are, the quicker the program can become (and potentially more powerful the chess engine can become). CPU clock rates upon initial research typically range from about 100 up to 600 MHz, and RAM sizes typically range from 200 KB up to 4 MB. The software will likely still be able to run under those minimums provided, but for the sake of minimizing response time, they should be as high as possible.

Low-Power mode should be a feature on this microcontroller to help with battery life on the board. Since there will be many points where the board is just waiting for the user to lift or place a piece, a low-power mode can be implemented to reduce its battery usage. In addition to that feature, the number of GPIO pins is important to have. Depending on the hardware stack, upwards of 30 GPIO pins may needed. Lastly the communication protocols of SPI, I2C, and UART may be used for communicating to these hardware devices, so missing any of these interfaces will restrict what hardware can be selected.

Something worth noting I found after researching these microcontrollers as that there are manufacturers of the processors, and manufacturers of the microcontroller which uses those processors. I found it like buying graphics cards- where a RTX 3070 isn't only sold by Nvidia but buy other manufacturers too who have the hardware components of the Nvidia but with slight modifications. Some of these microcontrollers researched have the same processors in them, but because of changes in the way they are produced on the microcontroller manufacturer specs may differ such, besides CPU clock rate.

3.8.3 Microcontroller Selections

3.8.3.1Teensy 4.0

The Teensy 4.0 microcontroller has high strength with a small size. With an ARM Cortex-M7 CPU, with clock rates of 600MHz and 1MB of RAM, it has some of the best processing power out of any modern microcontroller. It has 2 MB of Flash, which is the bare minimum required for the project, but this may prove to be punishing if my calculated estimation is incorrect. There are 40 digital input/output pins, which should be plenty for this project. For those communication ports, there are 3 I2C and 3 SPI.

One feature of this chip that is amazing is its variety of additional features it hosts, a majority of which aren't initially in the scope of this project but could be expanded for stretch goals. One of which includes its two I2S ports for digital audio. Other features included are Cryptographic Acceleration, Random Number Generator, RTC for date/time, and more. One feature that will be used for this project if this chip is selected is its Pixel Processing Pipeline- this will allow for an accelerated image processing for the display. Depending on what type of display is used, this addition could vastly speed up what and how things are displayed. To top things off, the price and availability of this device is amazing- only $25 and the ability to be delivered in less than two weeks.

Overall, this seems like a solid choice for the project. Its speed will greatly contribute towards our goal of having a short response time and give the chess engine the ability to be very responsive. My primary concerns come with the low flash memory, as this device may run into a situation where the entire program cannot be flashed on, resulting in major changes to the software. In addition, there is no UART communication ports which may make some peripherals unable to be connected at all.

3.8.3.2 ESP32

After finding different microcontrollers, the ESP32 appeared to be one of the most popular microcontrollers currently. It is known for its very cheap price with some strong specs. I manage to find one for $10, with fast delivery of under a week, making it super accessible for starting the project. In addition to that, it has 4 MB of RAM, and 4 MB of Flash, surpassing all the minimum requirements. Its number of GPIO pins isn't too extraordinary at 28 and has communication support for SPI, I2C, and UART. Other additional features this microcontroller has includes Bluetooth connectivity, a 2 MHz – 60 MHz oscillator, Ethernet MAC interface, 5 power modes including an Ultra-Low-Power mode, and a Random Number Generator. Most of those features likely won't be used, but the clock and power mode settings are very helpful for implementing this project efficiently.

Its greatest weakness comes from the CPU speed, at 240MHz. This is not necessarily that slow, but the Teensy 4.0 runs at over double the speed. And this 240MHz is only its *maximum* computing speed, which means realistically it will be running slower than that most of the time. This will be the bottleneck for the system, as the chess engine will directly scale its strength and response time with the CPU speed. However, its many strengths may cause this microcontroller to be the best choice for the project despite its relative slowness.

3.8.3.3 STM32

STM32 does not have nearly as strong specifications as the other options previously found. With a clock rate of 168 MHz, it is about 2/3 slower than the ESP32 and almost 1/4 as slow as the Teensy 4.0. In addition, the RAM comes in at about 200 KB, and the Flash at 512 KB. This is below what I believe should be used for this project, although it is possible it could fit on this small amount of space. Lastly, its price is the highest so far coming in at $28.

However, it does have some strengths. Like the ESP32 board, it has lots of communication with ports for SPI, I2C, and UART. What is greater than that is the fact the microcontroller has about 50 GPIO pins, if not more when configured correctly. This makes it highly adaptable to devices that have many inputs, where the other devices would run out of input space. It is also available very quickly, where it will ship in in less than a week. In addition, it also has the capability to connect to the internet, use an SD card slot, and be easily programmed with the JTAG / SWD debug interface.

The circumstances I could see this project using the microcontroller option is in a dire situation where we need to have a lot of GPIO pins, more than the other options can handle. However, its slow speed and very small amount of RAM and Flash do make it an unlikely contender for the becoming the microcontroller for this project. These specs theoretically could all work for the project, but it will make implementation more difficult, and the final product may have issues with response time.

3.8.3.4 Teensy 4.1

After researching for more similar products to the Teensy 4.0, I found the PJRC Teensy 4.1, which is more expensive, yet more powerful option. Using the same processor as the Teensy 4.0 (an ARM Cortex-M7), it reaches a clock rate of 600 MHz and has installed RAM of 1 MB. However, this version of Teensy improves on its predecessor, with a flash memory of 8 MB (4x greater than version 4.0). This eliminates one of the main concerns with using the Teensy 4.0 as now the flash memory much exceeds my minimum requirements, and it should be comfortably programmed on. In addition to all of this, the are 55 GPIO pins available, which is 15 more than the original version.

One additional special feature this version of Teensy offers is the ability to install additional RAM or Flash Memory. These PSRAM chips can add 8 MB of RAM each, and with two slots available to the Teensy 4.1, you can add up to an additional 16 MB of RAM. The cost of adding these features would total out to be an additional $9.00, where it would be $1.60 per added RAM, but an additional ~$5.00 in shipping. But, if shipping is purchased from the site I provided and not from Amazon, and I purchased the rest of the Teensy 4.1 microcontroller, it will only be about $3.00 more expensive to add these chips as the Amazon option is more expensive.

Other features this microcontroller has are the same as its predecessor, with RTC for date and time, a Pixel Processing Pipeline, and more. These features, particularly the Pixel Processing Pipeline, will help with lowering the response time and allow for the display to be high quality.

Overall, this microcontroller is an amazing contender for this project. It has the highest CPU speed, Flash, GPIOs, and potentially RAM with the additions out of any of the other options listed. A few downsides this microcontroller has though is its price, coming in at $35 or $38 with added RAM, making it the most expensive option. In addition, with manually installing memory to this microcontroller, there is risk of damaging it and needing to replace parts. It also potentially can take the longest out of any other option as well to show up, but that is still within a reasonable range.

3.8.3.5 Arduino Nano

The Arduino Nano has good mix of strengths and weaknesses. This device takes advantage of the ESP32 chip, but the microcontroller itself is manufactured by Arduino. Hence, the CPU speed is at 240 MHz, identical to the other ESP32 option.

It has a small amount of RAM with only 512 KB, and very few GPIO pins with only 14. In addition, there are only 4 total communication ports, 1 SPI, 1 I2C, and 2 UART. This may be a major issue when trying to connect to the other peripherals, as communication may be limited.

Its greatest strength however lies in its Flash Memory. Its internal Flash Memory comes in at 8 MB, already equal to the Teensy 4.1, but it also comes with an additional 16 MB of external Flash Memory. This 16 MB external Flash Memory is much slower than the internal one, but with those two combined it can hold 24 MB of Flash Memory. This gives massive flexibility in the software development process, as it is very unlikely that while programming this, we reach the cap of memory. Not only that, but this device is also one of the most affordable, coming in at the cost of $19. Lastly, this device will ship quickly arriving in less than a week.

Overall, this device has some solid strengths, but isn’t likely to become to the selected microcontroller for this project based on its low number of GPIO pins and communication ports. With a complex hardware stack, the four communication ports likely won’t be enough. It is possible to communicate to multiple devices using one port, but the added latency from doing so will hurt the overall response time of the project.

TABLE 16 MICROCONTROLLERS

Specifications:	Teensy 4.0	ESP32	STM32	Teensy 4.1*	Arduino Nano**
CPU Speed	600 MHz	240 MHz	168 MHz	600 MHz	240 MHz
RAM	1 MB	4 MB	200 KB	1 MB / 17 MB	512 KB
Flash	2 MB	4 MB	512 KB	8 MB	8 + 16 MB
GPIOs	40	28	50	55	14
Communication Protocols	3 SPI, 3 I2C	4 SPI, 2 I2C, 3 UART	3 SPI, 3 I2C, 3 UART	3 SPI, 3 I2C	1 SPI, 1 I2C, 2 UART
Price	$25	$10	$28	$35 / $38	$19
Delivery Time	< 2 Weeks	< 1 Week	< 1 Week	< 1 Week / < 2.5 Weeks	< 1 Week

**Two values are shown for some of these specifications: values on the left are without the added RAM, values on the right are with the added RAM.*

***The two values listed under Flash are for 8 MB of internal flash and 16 MB of external flash. External flash is much slower than internal flash.*

3.8.4 Microcontroller Conclusions

After examining many different microcontrollers, the conclusion was met that the best choice of microcontroller would be the Teensy 4.1. While the Teensy 4.1 is the most expensive option, it remains in the price range of $20 - $40 for this component. Due to a chess engine needing the most computation power available to be the most successful it can be, using the powerful ARM Cortex-M7 chip running at 600 MHz was desired the most. In addition, the ability to reach up to 17 MB of RAM will give this device a high upper bound for performance.

When designing the software, the entire program should realistically fit in under 2 MB, but with Flash Memory of 8 MB, there is plenty of room for adjustment if this estimation was incorrect. The 55 GPIO pins also will permit for very easy and clean hardware integration.

As further hardware components are researched, the communication protocols used needs to be carefully monitored. There is now the added restriction that UART cannot be used to communicate as this chip does not support that communication system. In the circumstance that a hardware component cannot use UART, then a different microcontroller must be selected. In that scenario, the ESP32 would be the best choice.

3.9 Converting Analog to Digital Signal

One of the biggest components of our project is finding a way to accurately read values from the photodiodes. This will require us to find some way to take the analog signal from the photodiodes and convert it to a digital signal for our microcontroller to read. The microcontroller that was selected is only capable of ADC conversions on 18 of its pins. Each square on the chess board will contain a photodiode so 64 different photodiodes need to be read in total. To help simplify the number of analog inputs to the microcontroller, we needed to use a multiplexer. In the off chance that the built in ADC on the Teensy 4.1 is not enough, we looked into other ADCs that we could use.

3.9.1 Multiplexer

A multiplexer is a type of combinational logic circuit designed to take in several different inputs from multiple pins and combine them all to a single output line. Essentially a multiplexer completes the opposite task that a shift register is designed to do. To accomplish its task, a multiplexer will alternate reading each input and combine each input to the next. This can be compared to a recording taking a sentence from 8 different people and combining it all into one file. Multiplexers will help trim down on the number of inputs to be read at once.

There are mainly two different kinds of multiplexers: analog and digital. As the names suggest, multiplexers are made to handle certain types of signals. Digital multiplexers are only capable of handling digital signals. Digital signals can be described as being high/on or low/off. Analog multiplexers are capable of digital signals as well as analog signals. Analog multiplexers can transfer the analog signal through its output.

Since we want to send the analog signal directly to the microcontroller to be converted, an analog multiplexer would be the best option. Like with the LEDs, we need to be able to consider the switching speed of the multiplexer. We also need the multiplexer to read the full range of voltage values that the photodiode will give off. We plan to measure the output voltage of the photodiode so the multiplexer must meet the demand of these voltage values.

As we did for the LED drivers, we want to be certain that we can test the multiplexer before soldering it onto a PCB. This means that we need to find a multiplexer that is in through-hole packaging.

To simplify the circuit for the photodiodes, they will most likely be connected in a way similar to the LEDs. We used a matrix system to connect all 64 photodiodes together. One way that we are thinking to collect values is to cycle through each of the 8 rows constantly and record the states of the photodiodes each time. This way, we can easily discover when a photodiode changes states based on if a piece is placed or removed. The multiplexer would be connected to the bottom of the columns for the matrix while a shift register will be used to supply power to each row. This method will make it easier for us to detect which square state changes based on the grid built into the matrix. We can also use multiple of them at once to constantly monitor all 64 photodiodes.

The most important factor that needs to be considered when selecting a multiplexer for our project is how much current it can intake. With our photodiodes, we need to be able to read at least 7 distinct values, so a high resolution is necessary. At this time, we do not know the exact range of current values the photodiodes will provide.

Below is a table showing off 4 8-to-1 analog multiplexers. The TMUX8108 is an analog multiplexer that is capable of reading current of up to 100 mA. This analog multiplexer is does not come in a through-hole package so we would not be able to fully test it in a breadboard. The TMUX6208 is an analog multiplexer that can handle the most amount of current out of the three multiplexers that were researched. Like with the TMUX8108, the available packagings of the TMUX6208 would not make it possible to test in a breadboard environment either. The CD74HC4051 is the one out of the three multiplexers that can be tested in a breadboard environment. The issue with this multiplexer is that it can only take in current values up to 25 mA.

A factor to consider is the switching speed of the multiplexer itself. The multiplexer must be quick enough to keep up with switching between the 8 inputs consistently. Out of the three multiplexers, the TMUX8108 is the quickest on its switching time with a time of 12 µs.

For this project, we selected the TMUX8108 for its switching time. We needed at least 8 of them to be connected to each of the analog pins on the selected ADC to feed all 64 inputs into our microcontroller.

TABLE 17 MULTIPLEXER COMPARISON

	TMUX8108	TMUX6208	CD74HC4051	CD405X
Vin (single) (V)	12, 16, 20, 36, 44, 72, 100	5, 12, 16, 20, 36	1.8, 2.5, 3.3, 5	Up to 20 V
Input/output continuous current (max) (mA)	100	300	25	10
Packaging	TSSOP (surface mount) WQFN (chip carrier)	TSSOP (surface mount) WQFN (chip carrier)	PDIP (through hole) SOIC (surface mount) SOP (surface mount) TSSOP (surface mount)	PDIP (through hole) TSSOP (surface mount)
Switching time	12 µs	140 ns	10 ns	400 ns
Price	$3.850	$2.432	$0.097	$0.61

The goal of this multiplexer is to save on the number of pins for our microcontroller. There is still a possibility that we may not need to use the multiplexer in the first place. We didn't have a definite answer until we select all of our components and map all the pins on our microcontroller accordingly.

3.9.2 Analog to Digital Converter

This section will cover our research into analog to digital converters (ADC). ADCs are an integrated circuit designed to take analog input and convert it to a digital signal. This digital signal will then be sent to our microcontroller to interpret. We used the ADC to read the voltages of our photodiodes in order to determine which piece is above the specific photodiode.

The role of this ADC is to read the voltages off of the photodiodes. The voltages of the photodiodes will change depending on how much light it is receiving. With an increase in the intensity of light that it reads, the more voltage it will provide. We looked into arrays to simplify the reading process of these photodiodes and we decided on an array that works almost like our LED array. We needed at least 8 analog inputs on our ADC to read all 8 photodiodes in a row.

ADCs have multiple specifications to consider when selecting one. The number of bits an ADC is the equivalent to how many levels of values it can detect. So, for a 3-bit ADC, 8 levels of output codes would be available. The more bits would mean more values that we can access.

If we select an external ADC, we also need to consider the communication protocol it uses. The Teensy 4.1 is not able to use UART and it has a limited number of pins for SPI and I2C. We are already using all three SPI ports for the LCD screen and the switch registers for the LED array. We can add on the ADC converter to the shift registers, but it would take more work programming wise to implement it. Ideally, an ADC with I2C would be our best option for this project.

The sampling rate refers to the number of samples per second. By using the sampling bits, these samples are then converted to a digital signal. Since we want to ideally be able to detect 7 different values, but we can still work with 6, we want the samples that are to be taken to be precise. The number of bits and the sampling rate will play a significant role in our ability to read the voltages precisely.

The microcontroller we selected has built in ADC pins, but they are limited to how much voltage they can read. The maximum voltage one of these pins can read is 3.3 V, anything higher is not recommended because it would burn out the pin. We do not yet know the exact range of voltages our photodiode will provide us, but we want to have another ADC ready in case our microcontroller cannot meet the demands we require. There is also not a lot of information available about the exact specifics of the built-in ADC on the Teensy 4.1 so we had to test it out.

The ADS7138-Q1 is a 12-bit ADC converter that has 8 analog inputs. With 12 bits of resolution, we had access to 4096 different values, with one of these values being 0. The sampling rate is also reasonably high at 140 ksps. Each input pin is also capable of reading input voltages up to 5.5 V. The communication protocol of this ADC is I2C.

The ADS1015 is also another 12-bit ADC. The issue with this ADC is that it can only take in 4 analog inputs at a time, so we would have to use 2 ADS1015 chips to satisfy the requirement of reading 8 analog input values. The ADS1015 can read a higher voltage compared to both the Teensy 4.1 and the ADS7138-Q1. The sampling rate is also lower than the sampling rate of the ADS7138-Q1 because it can only takes samples at a rate of 3.3 ksps.

Both of the ADC ICs that were looked into have better specifications compared to the built in ADC on our microcontroller. We didn’t know for sure until we start to do testing to know for sure which ADC will work best for our project.

TABLE 18 ADC COMPARISON

	Teensy 4.1 (built in ADC)	ADS7138-Q1	ADS1015
Maximum voltage for input pin	3.3 V	5.5 V	~6 V
Input pins	18	8	4
Sample rate	N/A	140 ksps	3.3 ksps
Input voltage	N/A	5.5 V	0-5.5 V
Communication protocol	N/A	I2C	I2C
Resolution (bits)	10	12	12
Price	N/A	$3.57	$1.1

3.9.3 Operational Amplifiers

This section will cover the selection of the operational amplifiers (op amps) that will be used in the transimpedance amplifiers for this project. When selecting op amps, we wanted to select an op amp that ran on low voltages between 3.3 V to 5 V. The reason for this voltage range is because of the voltage regulator that is on our selected microcontroller and the fact that a majority of the systems in this project will need to run on 5 V provided by our batteries.

One consideration that was made was the fact that the ADC pins on the Teensy 4.1 can only withstand voltages up to 3.3 V, otherwise we would lose access to that pin. One property of operational amplifiers is called rail-to-rail. Rail-to-rail is the ability of an op amp to swing the output voltage as close as possible to the two supply voltages. So, we decided to use an op amp that could run with a supply voltage span of 0 V to 3.3 V.

To also save on the amount of soldering to place an op amp on all 64 squares, we also wanted to select an op amp that had a quad packaging. This means that there are four op amps in one IC and all four can be supplied the same supply voltage.

The operational amplifier that we ended up selecting for this project is the OPA4990IDR. This op amp runs at low supply voltages and is also capable of rail-to-rail output. The cost for a singular op amp is $1.765 per IC.

4 Project Standards and Constraints

In the dynamic landscape of modern industries, adherence to established standards is paramount. These standards, whether industry-specific or regulatory, embody a collective wisdom that has evolved over time, encapsulating best practices, safety protocols, and efficiency benchmarks. This section will go over relevant standards for our project and their importance to its success.

4.1 Standards about I2C Bus Specification

4.1.1 UM10204 – I2C-Bus Specification and User Manual

In UM10204, the specifications and the uses of the I2C-bus are discussed. I2C is a bus that is both popular and powerful due to how devices communicate on this interface. One device is designated as a master device while all other connected devices are referred to as slaves. This document goes into detail about how the I2C-bus works and specifies the limits that all I2C devices must operate at.

One of the major features is how devices communicate with I2C. Only two bus lines are needed between the master and slave devices: a serial data line (SDA) and a serial clock line (SCL). The master has to always initiate communication with the slave whether it be to send data or to receive it. When sending data, the master sends a start message to the slave, sends the data, and then terminates the connection used to communicate. More steps are necessary when receiving data though. The master still sends a start message, then sends what register of the slave it wants data from, the slave transfers the data, and the master will then terminate the communication line. When there is more than one slave device connected to these two bus lines, the master will then use a unique address to refer to each slave device. The ability to easily add and remove slave devices is one of the major benefits of using I2C because no drastic changes have to be made to the circuit housing these devices.

4.2 Standard About Soldering Standards

Both J-STD-001 and IPC-A-610 are standards that involve the soldering process and includes industry standards for PCB assembly. IPC-A-610 is the overall standard used for soldering because it covers the electronic assembly acceptance while the J-STD-001 is more specific because it covers the materials and processes used for soldering. The LED array in this project require each individual LED to be soldered into the PCB that will contain the wiring for the matrix. The photodiodes that were selected will also need to be mounted to the PCB by soldering as well. These standards will help ensure proper care is taken when soldering these components to the PCB.

4.2.1 IPC-A-610 – Acceptability of Electronic Assemblies

IPC-A-610 is the most widely used electronics assembly acceptance standard in electronics history. This standard deals with the visual aspects of PCBs and ensures that all electronic assemblies meet a certain level of quality. Specifically, this standard deals with component placement, soldering, cleaning, and marking. With this approach in electronic assembly, manufacturers can ensure their products meet acceptable levels of functionality and appearance. This standard covers through-hole, surface-mount components, and mixed technologies.

With component placement, there are certain requirements that need to be met. Since a majority of our project will make use of through-hole components, a lot of detail will be going into designing the PCB. Proper spacing, alignment, and orientation of the components must be considered when designing the PCB. Components must be spaced and aligned properly so that interference between two components does not occur. Proper spacing also allows for an easier time when assembling and testing components on the board. Orientation refers to how polarized components such as diodes and capacitors will be placed onto the PCB so that it follows the design. If the orientation is not correct, the component would not function as intended and will cause further complications. The components have to mounted properly so that they are both secure and the leads of the component are long enough to have it properly soldered to the board. This standard also goes into depth on the soldering requirements. These requirements cover the formation, the fillet, the shape, the cleanliness, and the strength of the solder joint. By covering these specific components of the solder joint, soldered components will be securely attached. Cleaning is required after soldering is completed so that any residing flux is removed so that it does not cause any interference.

4.2.2 J-STD-001 – Requirements for Soldered Electrical and Electronic Assemblies

This standard sets the requirements for the manufacture of electronic assemblies and is used by electronic assembly manufacturers, suppliers, and users. Specifically, this standard covers the necessary materials, methods, and criteria for creating a high-quality electronic assembly. This standard is important to consider because none of the team members have experience in soldering and having basic information about the skill will be very useful.

The standard materials for soldering are covered in J-STD-001. In order to ensure the correct operation and dependability of electronic assemblies, the general composition and qualities of solders are covered. Cleaning agents are covered so that flux residues are properly removed as well as ensuring that the electronic assembly is not damage in the same process.

4.3 Standards for Rechargeable Lithium-Ion Batteries

4.3.1 IEC 62133 - Safety requirements for portable sealed secondary cells, and for batteries made from them, for use in portable applications.

Manufacturers, governmental authorities, and stakeholders involved in the lithium-ion battery industry commonly acknowledge and use IEC 62133 as the benchmark for battery safety. This international standard, developed by the International Electrotechnical Commission (IEC) and last updated in 2021, outlines specifications and tests for the performance and safety of lithium-ion batteries used in a variety of portable electronic devices, such as tablets, laptops, cell phones, and other gadgets. The standard addresses various aspects of battery safety, including electrical, mechanical, and chemical safety.

Adhering to the standard contributes to guaranteeing the safety and dependability of our lithium-ion batteries. IEC 62133 specifically addresses problems like thermal runaway, short circuiting, overcharging, and overdischarging, all of which have the potential to pose a risk to the safety of our users if left unchecked. The standard also specifies testing protocols to confirm standard compliance and labeling and documentation requirements.

IEC 62133 outlines various testing procedures, including checks for electrical abuse such as overcharging and short-circuit conditions, to guarantee that the device is safe for use in its intended applications. One of these tests includes charging a cell utilizing a charging method applicable to cells and batteries that are subjected to external short circuit, thermal abuse, crush, and forced internal short circuit tests before short-circuiting the cell, connecting its positive and negative terminals with a total external resistance of less than 100mΩ. In such abusive electrical scenarios, the battery needs to show a certain level of resilience to avoid thermal runaway or other dangerous reactions, otherwise a failure in the device's operations could pose a great hazard to the users' safety. [22]

However, one of the most pertinent sections for our needs is the one assembly of lithium-ion cells into lithium-ion batteries. It specifies that if several lithium-ion cells are used to build a single battery, then each battery must have precisely matched capacities and be made by the same manufacturer, chemist, and design of cells. The importance of ensuring that each lithium-ion battery, if there are several, has its own independent control and protection is also mentioned in this section. It also emphasizes how important it is for the battery designer to make sure that the design and assembly follow the guidelines provided by the cell manufacturers regarding the maximum limits for voltage, current, and temperature.

IEC 62133 is essential to maintaining the dependability and safety of lithium-ion batteries, which in turn contributes to the general security of the equipment and systems that use them. Manufacturers must adhere to this standard to satisfy regulatory requirements, secure market acceptance, and foster customer trust.

4.3.2 IEC 62115 – Electric Toy Safety

Outside from IEC 62133, another standard that we may want to look at for our project that can impact on our battery design is IEC 62115, which specifies safety requirements for electric toys, that is, any toy that has at least one function dependent on electricity. Any product designed or intended for use in play by children under 14 years of age are considered toys, whether designed exclusively for that demographic or not. [21]

This standard covers the whole range of electric toys from small button battery operated lights to large ride-on electric toys powered by rechargeable batteries. This results in different requirements and tests according to the type of electric toy. Nonetheless, the aim of this standard is to reduce risks when playing with electric toys, especially those risks that are not evident to users, in the case of its electrical function.

4.4 Software Standards

4.4.1 ISO 9899 – C Programming Language

One of the most important standards to be followed for the implementation of our design is the ISO 9899, which standardizes the C programming language. Throughout this document, there are definitions on the syntax, semantic, input specification, output specification, and limits to C. Without this standard, every single device might have its own variation on the language of C, making reusing code virtually impossible as every device would have its own set of commands that differ from the next- meaning either re-writing all the code, translated for that variant, or finding code that someone developed specifically for that device, both of which would cause major restrictions in development.

Some other parts of what this design standard covers includes how it deals with floating-point numbers. In TS 18661, there are five sections listed out describing Binary and Decimal floating-point arithmetic, interchange and extended types, and supplementary functions and attributes. Floating point arithmetic is often one of the most CPU intensive function, so unifying the way the C language uses these will minimize how intensive it will be. Another important section is in TR 18037, which discusses Embedded C programming, and how the C language will interact with all the variable hardware components. This standard is particularly important because programming on any device will use the same command terminology.

4.5 Standard for Chess Board Layout

A standard game of chess is played on a square board that is arranged in an eight by eight grid, with a total of 64 squares. The chessboard alternates the color of the squares between light and dark colors, as related to the color of the pieces. There will be 32 light squares and 32 dark squares. The rows of a chessboard are always referred to as “Files” and typically labeled by letters in the alphabet, ranging from A to H going from left to right. The columns of a chessboard are always referred to as “Ranks” and typically labeled by numbers, ranging from one to eight going from bottom to top. Chess pieces are split into two separate sets, a lighter color (usually white or beige) and a darker color (usually black or brown). Each player will control a set of pieces that contains 16 total pieces. Each side has 6 different kinds of pieces that includes eight pawns, two knights, two rooks, two bishops, one queen and one king.

The setup of the board is the exact same for both players and never changes. The orientation of the board will always be the same for every game. The board will always be oriented so that there is a light colored square in the bottom right corner and a dark colored square in the bottom left corner. A game of chess can never be played if the board is not positioned like that. The setup of the pieces for the game is the exact same for each player as it will always mirror one another. In the back row, the rooks are placed in the corners, followed by the knights placed in the next inside squares, followed by the bishops next to the knights. In the two middle squares is where the queen and king are placed. The queen is always placed on the left middle square while the king is placed on the right middle square. The queen should always be on a square of the same color (the white queen on a light square and the black queen on a dark square). The king will always be placed on an opposite colored square (the white king on a dark square and the black king on a light square). In the row in front of the back row is where all eight pawns will be placed.

The dimensions of the board will vary from country to country, but there are a few standards that are followed, and enforced for professional chess. The dimensions of each square on the board are the exact same as well as the entire board itself. Square size on a chessboard is always proportional to the size of the king’s base diameter. The base diameter of the king should be roughly 75%-80% of the size of the square. This guideline ensures that there will be proper piece spacing on the board during a game. The official World Chess Federation (FIDE) Championship Chess set has a square size of 2 inches and a height for the king at 3.75 inches. Meanwhile, the United States Chess Federation (USCF) has a standard tournament chess board that has 2.25 inch squares and the king’s height is 3.75 inches. Another guideline related to the size of the king piece relates to the king’s base diameter compared to its height. Both FIDE and the USCF follow the guideline that the king’s base diameter should be between 40%-50% of the king’s height.

Once the board is fully set up with 32 pieces altogether, then the game begins. White will always go first followed by black. The objective of the game is for one player to checkmate the opposing king. Once a king is put in checkmate, the game is over.

4.6 Project Constraints

4.6.1 Design Constraints

Design constraints refer to certain physical restrictions that are present when designing a project. The main goal of this project is to create a smart chess board that is portable which causes some constraints to arise when designing our project.

One of the biggest factors that deal with portability is the material the chess board and the box under it will be made out of. We want to use a material that is sturdy enough to protect the hardware it encases but at the same time not be too heavy for a person to carry. There is also a limit to what materials we can use due to economic constraints. None of the hardware components should be disturbed if something were to happen to this shell. Inside of this shell, we want to be able to house two extra layers that the players cannot see: the matrix that contains both the LEDs and the photodiodes, and a PCB that contains the microcontroller and other hardware components. The dimensions of this box should not be too large where it makes it awkward to carry either. We want to aim for a standard chess board size with only a few inches of added depth.

The top layer of our project has some design constraints as well to make it as appeasing as possible to the players. We do not want to give away what is occurring underneath each square so we need to use a material that can allow the LED to emit light but at the same time prevent players from seeing into the chess board. We also want to embed an LCD screen into the side of the project so that it can provide an explanation to players about their selected pieces. We want an LCD screen that is large enough to provide a few sentences but nothing that will add extra bulk to this project.

Another big component is the layer that will be underneath the board the players will play on top of. This layer will contain the LED array along with the optical system that will detect the different pieces. One of the biggest issues with this is to find a way so that the LED does not cause any disturbance to the photodiode and give false values. There are two possible ways to solve this problem. The first is to seal off the optical system under each square so that it can only detect the light that is reflected off the chess piece. This method would then cause the placement of the LED to be shifted so it would be off-centered. The new placement of the LED could cause uneven lighting to occur under a square, so we need to find a way to make it appear even. The second method we can use to solve this problem is using a specific color of light in the optical system. We can use a filter on the photodiode so that it only picks up a certain color of light and make sure the LED array does not light up to that color. In both of these cases, we still have to ensure that the light from each LED does not bleed into neighboring cells and give false information to the players.

There is also a constraint between the two bottom layers of this project. We plan to use fiber optic cables to direct light to each square. The issue with this is the placement of the LED these cables will run to and how these cables will be connected to each square. We want to place the LED on the bottom layer so that it can be in the center and run the cables up through the second layer to attach to the bottom of the top layer. The issue with this is how to implement this in a way where we can still adjust the project without too much disruption to these cables. We can make small holes in the PCB that will hold the LEDs and photodiodes so that these cables can run through to the top layer. The biggest challenge is making all three layers connect.

4.6.2 Economic Constraints

Due to the nature of this project, there is a major economic constraint on the design. Regardless of the scale of what team is designing a project, there is always a tradeoff between minimizing the cost and maximizing the quality of the final product. If there was no economic constraint, then every component would be the highest quality available, and the quality of the final product would be very high- with the cost being equally high, of course. On the other hand, if there was only an economic constraint with no care for the quality of the final product, the result would be very cheap, yet barely function and run at incredibly undesirable speeds.

Because this team is comprised of five undergraduate university students, the tradeoff between cost and quality will have to favor more towards low costs with some reduction to functionality. Each component will have to be selected with this in mind, but ultimately depend on exactly how much value does each one offer for the additional dollar. There are few different groups of components to consider where to distribute cost to. Each component falls into one of the following groups: Microcontroller (MCU), Display (LED array and Display Screen), Piece Identification (Photosensors, Fiber Optic, and Chess Pieces), Power (Battery and Voltage Regulator), or Case + PCB (PCB, Chess Board, Buttons).

The Microcontroller is one component that has certainly high value for every additional dollar spent. Because one of the main challenges over past similar projects was the response time, having a powerful microcontroller would help solve those issues. However, the constraint exists, and this component can get very expensive when reaching very high specs. Ultimately the constraint of < $40 for this seems reasonable, and there are plenty of powerful microcontrollers that would help address the issue of response time while still being affordable.

The Display components has its tradeoffs. The LED array can be restricted to less colors making it cheaper, which ultimately won't have too big of an effect on the final product, as two colors per square is the least amount that would be needed. In addition, the display screen is a component that can get very expensive with increased size, but ultimately the design can be built around a small screen used to communicate smaller pieces of information, rather than a lot of information at once that would need a larger display for usability. Generally, these components can still function well with cheaper options selected, meaning that a large economic constraint should be placed on these products.

The Piece Identification System (PID) also needs to have economic constraints but may experience some flexibility like the MCU. One specification for this project is to have a piece identification of > 95%. In general, the cheaper the component, the less reliable it will at detecting the piece. When selecting the fiber optic cable, photosensors, and material for the bottom of the chess pieces, the difference between less than 95% and greater than 95% accuracy of detection will be worth the extra dollars. However, there is still constraint, as with the rest of the project, and at a certain point keeping the costs lower and taking a lower accuracy is appropriate. The difference between a less than 99.9% accuracy and a greater than 99.9% accuracy is less worth the additional dollars than the previous specification.

The power supply system in particular needs strong economic constraint to avoid raising the cost of the entire project significantly. One specification for this project related to this component is the battery life, which should be greater than 4 hours. Beyond that minimum though, there should be strong constraint to avoid higher prices- the battery can get very expensive for greater life. The other components related, such as the voltage regulator, may have less of a tight constraint on it as the battery, as if that component fails then many other components may have to be replaced, significantly raising the cost of the project.

The remaining components with the Case and PCB should be following a strong economic constraint. Apart from existing and providing its basic functionality, increased spending on these components will not improve the achievement of any of the other specifications. It can help with the overall aesthetic design, but for functionality's sake, it is not worth the additional costs. These components should be at a minimum in price.

Overall, the economic constraint exists in every portion of this project. Breaking the budget set for any of the components can lead to troubles implementing, as this project does not have any sponsors and will have to directly come out of the pockets of the students designing it. Most of the components do have some impact upon how successful the final design will be at achieving its specifications, so cost should be appropriately weighed in against the quality of its functionality, but those components without that direct correlation should minimize its cost as much as possible.

4.6.3 Manufacturing Constraints

When creating a product, it's important to consider the many manufacturing constraints so that your product can be as efficient and as cheap to produce as possible. The selection of the material has a very large impact on the manufacturing process and can drive up costs, for example using LEDs to light up the squares and as our illumination source. We had gotten 100 LEDs in one order due to the high availability of this component. To show this further, the 100 LEDs also only cost $6.75, making the cost per unit to be 7 cents each, making them very cheap. LEDs have a very long lifetime meaning that they are very durable and don't require high powers that will wear them out. The availability of material, the cost of each component, and durability of the parts are all considered in the material selection. There were many different types of light sources we could have used, however most of them would have run into one of these problems and not been as efficient.

Additionally, the manufacturing process of how parts are made should be considered. Selecting the right process to make your product and making sure it’s capable of creating the right quality and quantity is important to make sure the product lasts a long time and is reliable during that time. For our project, we decided to 3D print the chess pieces because we would need the inside of each piece to be hollow. This is because we put filters inside each piece to integrate them into the optical system effectively. This would not affect the quality of the pieces since it would still be one color and would not affect the shape of each piece. 3D printing might take a long time for each type of piece, but it would allow us to customize the sizing if needed and make it hollow inside to fit our purposes. We could try buying plastic pieces to shorten this design process but since we would want them to be hollow it might cost more to customize them as needed. Another method we could have tried is to use wood and to carve the pieces out but that would likely take longer to make than 3D printing and if we made it would likely affect the quality because it would depend on the carving skill of the carver. In addition, wood carving tools are expensive and would be considered an unnecessary expense for this project.

Another manufacturing constraint would be the specifications of your product in terms of its size and the components within it. If there are small specifications that could lead to more precise and costly manufacturing due to tolerances and need for more advanced equipment. If we choose a small LED, we could make the cylindrical base smaller. However, it would be harder to manufacture a smaller LED due to the already small size of an LED and the equipment need to make an LED that small. This would lead to the cost of the development of this project increasing a lot since we would need a lot of LEDs to achieve our goal.

Furthermore, manufacturability, which shows how easy something is to create in a factory setting, is something that needs to be considered. Making designs that would be easy to manufacture by minimizing complex shapes and having hard to reach places would be helpful because they can cause production costs to increase, and the chances of defects would be higher because of the need for precision. An example of this would be our electrical circuit design. By making the electrical setup smaller it would decrease the complexity and the cost of the project but would also make it harder to build. We tried many different designs to make the electrical setup smaller like putting our components into boxes to have less wiring to reduce cost and reduce material waste.

Moreover, scalability is an important manufacturing constraint because some manufacturing techniques are better for high volume productions while some are better for low volume customized products. In our project we could have made filters for specific wavelengths, but it would have cost more instead of general filters that filter all wavelengths out evenly. LEDS are also very scalable since we could get 100 of them in one order. Scalability as a whole is important because any product has to comply with supply and demand, if our project cannot keep up with these changes in the market it will fail as a whole.

In addition to cost constraints, we need to better understand the actual costs of manufacturing. Things to consider that were factored into our product design are material cost, labor costs, and overhead costs. In our project this would involve our optical and electrical design since making a more complex one would cost more to make because of additional time requirements. Adding multiple layers would increase the difficulty of manufacturing, labor, and money by adding extra steps and more work. This would mean having to spend more money because workers need to be paid, machines need maintenance and repair, and more advanced machines cost more money.

Another manufacturing constraint would be supply chains. We would need to analyze our supply chain to spot potential bottlenecks or risks. Having an alternate source for critical material and components is important. There are many components in our project from different companies with different specifications. While we are choosing what is best for us now, later on we needed to consider alternative sources in case some businesses go bankrupt, products are discontinued, or other circumstances happen without our knowing. We also needed to understand what's happening in the world due to material shortages that could happen or shipping problems that could be caused by terrible world events such as COVID 19. In summary, keeping close tabs on the different factors that would affect us getting the materials we need to make this product a success is important to the product as a whole.

Finally, the transportation and logistics of our manufactured material needs to get back to us to use it. We also considered the cost and delivery time. If our materials are made in another country, they could be shipped to us in a cargo ship, which would be the cheapest option. However, would take a longer time. If we wanted priority, it could be flown to us but will cost more. Another thing to think about is gas prices, as those have been on the rise and have affected ground shipping costs as well as time to ship using this method. Different materials have different lead times for this reason. If there are deadlines to be met, such as what's required of the product right now, purchasing will have to be done very ahead of time to make sure we get everything on time and can make the product as needed.

By having these manufacturing constraints in mind during product development, we can reduce the chance of costly development changes and production delays looking forward. Organizing a feasibility study can help make a smoother and cost-effective production process. Having documentation that will outline these problems and how to best handle them will also be very helpful.

4.6.4 Environmental Constraints

Considering environmental constraints when developing a product is becoming very necessary in the world today as environmental sustainability is becoming a bigger worry as climate change continues to take hold. Failing to take this into account can lead to long-term negative environmental impacts and reputational damage within the company that produces the product.

A major environmental constraint would be energy efficiency. Designing our chess board to be energy efficient would reduce energy consumption and avoiding greenhouse gas emissions used to power it would be ideal. An example of energy efficiency in our project would be the use of LEDs on our chess board. LEDs are one of the most efficient light sources since most of the electrical energy is converted into optical energy or light instead of being lost to heat. Incandescent light sources produce light by heating up materials, but this is very inefficient since most of the electrical energy is turned into heat. In fact, less than 5 percent of the energy is turned into visible light while the rest of the energy is loss as heat. When choosing our illumination source, we looked into incandescent light sources, and we quickly realized that it would be too inefficient and would require more energy to use. The lumens efficacy for a 120-volt incandescent bulbs is 16 lumens per watt while an LED is 100 lumens per watt. Incandescent lights are so inefficient that some governments are starting to phase them out to decrease energy consumption. Dealing with energy constraints in product development decreases environmental impacts, increases cost savings, generates a longer lifespan for the product, and increases competitiveness due to energy efficiency is important in the market, as customers are becoming more mindful of environmental needs and considerations. It is important to run simulations and energy inspections to make decisions relating to energy consumption in product developments to ensure that all standards are being met to the best of the company's ability.

4.6.5 Energy Constraints

Energy constraints are important to consider when developing a product because energy usage has economic, environmental, and operational implications. We already talked about how we are using LEDs which are very energy efficient and minimize energy consumption.

Managing battery life is a critical energy constraint. The target is to achieve a minimum of 4 hours of battery life, requiring a thoughtful approach to energy consumption optimization. The key lies in selecting energy-efficient components and utilizing effective power management strategies. One example within our system is the LED array. While there is a straightforward method of turning on all three red, blue, and green LEDs simultaneously, this approach demands a considerable amount of energy, potentially compromising the desired battery life. To address this challenge, we've opted for a more complex solution that involves rapidly cycling each LED on and off at a speed not visible to the human eye. This technique, known as flickering, not only creates the illusion of a consistent illumination but also significantly conserves energy compared to simultaneous activation of all LEDs. Flickering is a well-established practice in various optical applications, capitalizing on the phenomenon that our eyes cannot perceive rapid changes in light intensity. By strategically cycling the LEDs at a frequency beyond our brain’s comprehension, we achieve the desired visual effect while also minimizing energy consumption. This approach aligns with our commitment to maximizing battery life without compromising the functionality and aesthetics of the chessboard. Furthermore, the selection of energy-efficient components is a pivotal aspect of our strategy. Each component, from LEDs to other essential elements, is carefully chosen based on its energy efficiency, ensuring that the overall system operates optimally within the defined power constraints.

Another crucial aspect to consider within illumination systems is the generation of heat. Heat generation not only signifies a potential inefficiency in the energy conversion process but can also have implications for the overall performance and lifetime of the system. As highlighted earlier, incandescent lights are a perfect example of this concern, as they are known for their inefficiency, losing a substantial portion of their energy as heat. In stark contrast, LEDs emerge as a promising solution to combat heat-related challenges. LEDs showcase an impressive level of energy efficiency, converting a significant majority of approximately 80 percent of the electrical energy into optical light power. This characteristic not only maximizes the use of the energy input but also minimizes the production of heat, addressing one of the key limitations associated with traditional incandescent lights. The minimal heat output of LEDs is a result of their unique semiconductor-based operation producing light without the substantial heat generation characteristic of incandescent sources. This property is especially valuable in applications where heat dissipation is a critical consideration, such as in small/ tight optical systems or environments sensitive to temperature changes. Furthermore, the reduced heat generation contributes to the overall efficiency and safety of the illumination system. By minimizing wasted energy in the form of heat, LEDs not only enhance energy conservation but also promote a cooler operating environment, reducing the risk of overheating and potential damage to surrounding components.

In addition, an equally significant aspect involves educating the user on optimizing energy usage in our chessboard project overall. By teaching information about our project to users, we can enable them to interact with the chessboard in a manner that not only aligns with energy-efficient practices but also ensures the longer lifetime and effectiveness of the system. One approach to educating users on energy efficiency involves conveying the importance of interactions with the chess pieces. For instance, users could be informed that holding a chess piece for an extended period may contribute to energy wastage, as the system might continue to illuminate that specific area unnecessarily. Conversely, emphasizing the energy-saving benefits of leaving chess pieces on the board when not in use can promote more resourceful usage. Moreover, maintenance of the project plays a crucial role in sustaining energy savings over time. Providing users with information on routine maintenance tasks enables them to contribute to the system's energy efficiency. For instance, users could be encouraged to periodically change the LEDs, ensuring that the LEDs operate at their optimal efficiency. Regular inspection, replacement, or cleaning of filters are additional maintenance steps that users can undertake to maximize energy savings. By incorporating a user education component, we not only promoting a sense of responsibility among users but also enhance their understanding of how their interactions and maintenance practices directly impact the energy efficiency of the chessboard system. This collaborative approach between users and the technology underscores a commitment to sustainable practices and long-term energy conservation.

5 ChatGPT and AI

In this section we discuss the topic of AI and how its current popularity surge is affecting many different people and industries. Here we focus on explaining the phenomenon that is AI generation and how it can relate to our project in both positive and negative ways.

5.1 Generative AI

Recent years have witnessed a profound surge in the popularity of AI technology, especially in the field of generative AI. Millions of companies across the globe are utilizing artificial intelligence tools to improve their businesses. According to Reuters, ChatGPT alone has reached over 180 million users as of September 2023. [17]

Originally, traditional AI was designed to complete specific tasks given a predefined set of rules. This type of AI operates in a deterministic manner, following explicit rules and instructions. Contrary to traditional AI, generative AI can create a wide variety of new content by analyzing patterns in existing data and utilizing those to create a unique new output. It can produce complex content that mimics human creativity relatively realistically, making it a valuable tool for many industries such as gaming, entertainment, and product design. In one of their articles, Harvard Business Review states, “AI can not only boost our analytic and decision-making abilities but also heighten creativity.” [18]

Right now, generative AI is being applied and quickly improving in various domains, including natural language processing, image synthesis, text completion, code generation, and more. These advancements have opened new possibilities for the utilization of these tools and will fundamentally change many lines of work.

5.2 Ethical and Bias Concerns

As generative models become more powerful, increasing ethical concerns are being raised due to potential misuse of the technology, as well as raising reasonable fears that AI will ultimately replace human workers throughout the economy. This conflict brings to the surface questions over AI ethics as well as copyright and labor laws.

Many of these concerns stem from the field of creative work, such as that of visual artists. These generative AI models are trained on large datasets, often including copyrighted material scraped from the internet, causing them to directly copy the style and content from the publications of real artists without any compensation or credit. To make matters worse, deep neural networks are often considered "black box" systems because it can be challenging to understand how they arrive at specific decisions or generate certain outputs. The lack of explainability raises concerns over accountability, transparency, and the ability to address errors or biases within the model.

Privacy issues also became a public concern in the field of AI. Its capability to generate synthetic images that closely resemble real people can potentially lead to the unauthorized use of personal likenesses or even identity theft. In the same vein, there are concerns over the spreading of misinformation and the potential for malicious use of AI-generated content for deceptive purposes, which could harm individuals or manipulate public opinion.

These are all valid concerns, and addressing these issues requires a multidisciplinary approach involving technology experts, ethicists, policymakers, and society at large. It is essential to strike a balance to ensure that AI technology contributes positively and serves to augment human work, not replace it.

5.3 AI in Design Process

The multi-step process of designing entails investigating, developing, and perfecting an idea to produce a marketable product. With so many opportunities to improve productivity, creativity, and the design process, artificial intelligence can be a very useful tool in product design.

Every product design starts with a problem or challenge that requires resolution. Since it establishes the tone for the whole design process, identifying this issue and establishing the project's scope and goals are critical steps. With AI, designers now have access to new tools for trend and data analysis, that might assist launch projects much quicker.

Because AI can swiftly analyze large amounts of data, it also provides streamlined techniques for collecting information in research, increasing the efficiency of this process. AI can also be a fantastic source of inspiration by utilizing it to study existing designs and come up with original ones. Artificial Intelligence has revolutionized the creative design process, especially in terms of ideation and problem-solving. Based on human prompts, generative models can create new ideas and innovative solutions for designers. This quickens the ideation process and creates new opportunities for design exploration and innovation.

And lastly, the prototyping stage of a product can be greatly aided by AI. Advanced algorithms enable designers to replicate real-world scenarios, eliminating the need for physical models and facilitating the testing and improvement of product prototypes. This lowers the price of physical prototyping while also quickening the design iteration process. AI-driven simulations offer insightful information that helps designers decide on product performance and design changes. Through the application of AI in user feedback analysis, prototype generation, and design iteration, designers can guarantee that the product fulfills the requirements and anticipations of its intended market.

In summary, AI technologies have become invaluable assets in the design industry, easing jobs and enabling designers to push the boundaries of their imagination. They are used for everything from concept formulation and prototyping to data analysis and design optimization, and designers must adapt the way their work to accommodate the incorporation of AI into the creative design process, as we usher in a new era of efficiency and innovation.

5.4 Research Methods with AI

As generative AI is shown to be very valuable in the design process, its worth going more in depth into how research can be optimized using these contemporary tools. While the information generated by AI isn't always true, when it is interpreted instead as just a reputable opinion on something then the floodgates open with potential uses; It can be used to create recommendations for what to research, used to review what research you have done already, and used to solve any barriers that you may be currently stuck with.

Using AI to create recommendations for what to research is more efficient and plan internet searches. By taking advantage of an AI's ability to generate output with considering the context provided, you can create a list of devices worth researching into that matches that context. For example, in the microcontroller selection research portion of this paper, generative AI was used to help point in the right direction the initial devices to research. When asking it what devices it would recommend using given some of the project specifications, it gave a list of 6 or so devices. From there, to ensure the accuracy of the information found, the rest of research was completed manually by looking up those devices specifically. Rather than reference some random opinionated blog posts with generic lists of microcontrollers, using an AI both removed some of the bias and increased the relevance of those results. Out of those selected by the AI, all of them were great suggestions and considered as candidates for implementation.

Another strategy that could be used when conducting research is to use it as a tool to review what choices you have already selected. By using its ability to derive from context, you can prompt a generative AI with all the research options you have done so far and ask it if you are missing any major areas of research. For example, if I only have done research for a device from one manufacturer, it will be able to cater to my projects needs and recommend me some other manufacturers that can also be considered.

Lastly, using generative AI with research can help providing solutions for barriers you are currently struggling with. By once again taking advantage of the large context provided, it can take in many considerations and do its best to reason out a solution. For instance, let's say that while researching, you can't find any LED arrays that would be compatible with the selected microcontroller. Rather than having to get rid of the microcontroller, using AI to prompt it with the exact issue mentioned might shed insight into some components that would work, or check if the problem does need to come down to changing out the MCU. AI serves as another opinion to help aid in making project changing decisions.

5.5 Programming Assist

With generative AI being trained across billions upon billions of data entries into the internet, some AIs have the ability to create or modify code. Even at the very peak of generative AI, most cases it can’t simply just write the entire program for a user, but almost every generative AI can be used to assist the writing of it and be used as a catalyst for faster programming. Using AI as an assistant programmer is a technique that will only grow in use as time passes, until AI is strong enough to do all programming by itself. Until then, this team will be able to develop code much faster than one individual without AI.

When designing a section of code, using AI as programming assistant can go as follows:

- In detail, describe to the AI what a certain section of code should look like.
- Take generated code and modify it to fit the rest of the program.
- Copy and paste errors to the AI and give it the code back that you modified.
- Consider solutions and potential implement them.

This approach to programming allows for every piece of code you write to always be just a modification from a larger piece of code generated for you. Most of the code can be generated instantly for you, but because of limitations in the current strength of the AI, it can’t correctly process all the context of the rest of the program, including library restrictions, variable names, etc.

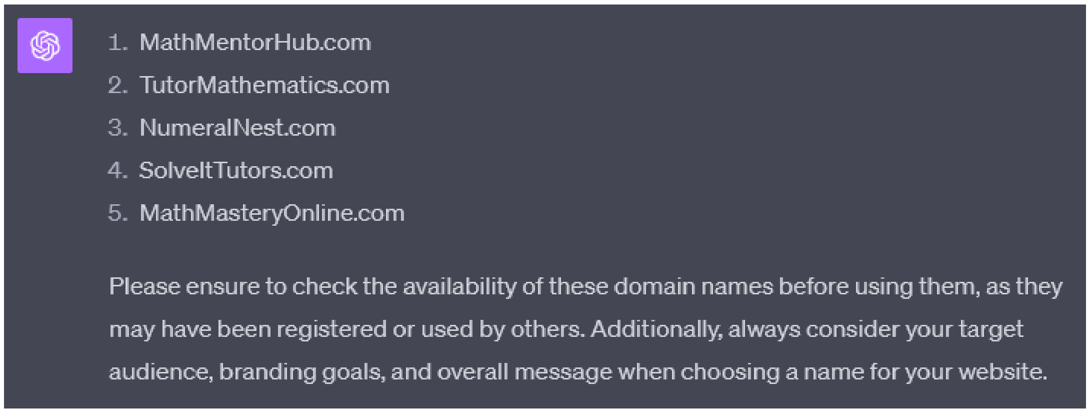
The approach described for debugging is also very helpful in the development process. By simply copy and pasting the errors given, as well as the code that is giving the error, the generative AI will do its best to solve the error on its own- often getting it right. This process can be repeated a few times if not solved immediately, but in some cases the error may not be able to be solved completely from the AI. That’s when you ask for suggestions for what to look for or what specifically is likely causing the issue, and it can help the developer look in the right areas first that may be causing the problem. This method will save lots of time programming by having all basic errors and bugs solved by the AI- which is trained on data from millions of issues of a similar nature- and only resorting to using the developer to debug more complicated areas of code. This optimizes what each party is best it- the AI is fast but not necessarily the smartest, while the human developer is the opposite, slow but can theoretically solve any issue.

5.6 Prompt Engineering

While generative AI has proven to be a powerful tool for many industries, this tool is greatly weakened without proper use of it. Since all the input you provide for the generative AI is its prompt, the better prompt you create, the better the output you can receive. The collection of strategies used to maximize the output from generative AI is known as prompt engineering. By manipulating the only communication between the user and the machine, result will likely be far more relevant and accurate then what a generic prompt could offer.

For example, let’s say you are trying to create a name for a new website you created that offers virtual private math lessons. You would want the best name possible as it will be the first impression it makes on users, which may turn into paying clients. A generic prompt you may yield:

"*List me 5 names for a math tutor website*"



1. MathMentorHub.com
2. TutorMathematics.com
3. NumeralNest.com
4. SolveItTutors.com
5. MathMasteryOnline.com

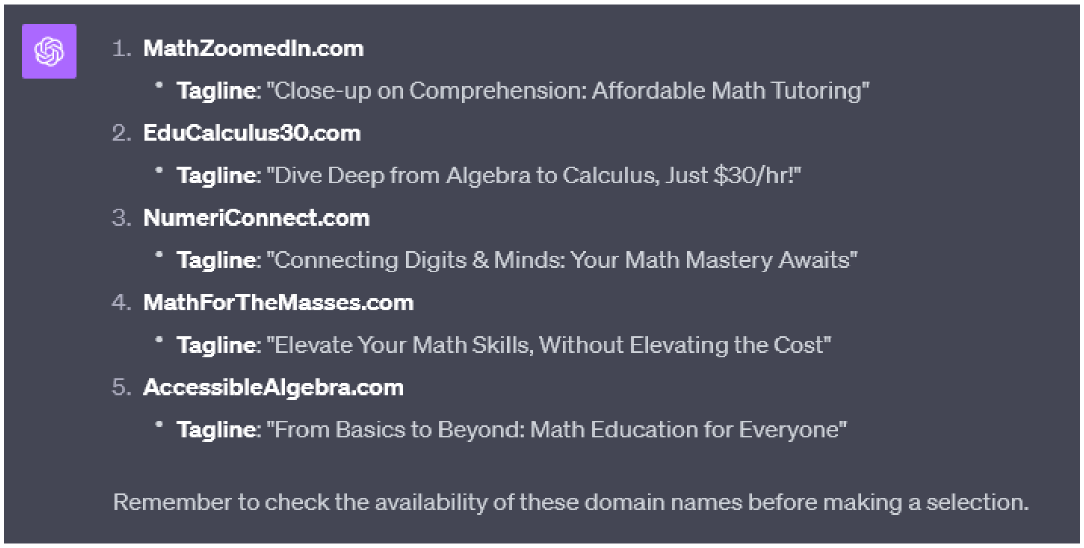
Please ensure to check the availability of these domain names before using them, as they may have been registered or used by others. Additionally, always consider your target audience, branding goals, and overall message when choosing a name for your website.

FIGURE 11 OPENAI'S GPT-4 PROMPT 1

Now these five website names provided aren't too bad! There are some catchy names that generally explain what the site does. However, the names are a bit too generic for the website and won't really make it obvious what is special about this site.

To improve this, we used prompt engineering. One basic technique with prompt engineering is to feed in many more details for what names you would want. The generative AI will do its best to consider all these details to craft a suitable list of website names. Here are the results when using a more precise prompt:

"*Generate five creative and engaging website names for a math tutor platform focused on providing affordable and accessible math lessons through Zoom, targeting high school students and those at earlier levels. The lessons cover topics from Algebra 1 to Calculus 2, offered at a rate of $30 per hour. The website's mission is to make math education easily accessible to all. Please ensure the names are catchy and inspire confidence in potential students*."



1. **MathZoomedIn.com**
   - **Tagline**: "Close-up on Comprehension: Affordable Math Tutoring"
2. **EduCalculus30.com**
   - **Tagline**: "Dive Deep from Algebra to Calculus, Just $30/hr!"
3. **NumeriConnect.com**
   - **Tagline**: "Connecting Digits & Minds: Your Math Mastery Awaits"
4. **MathForTheMasses.com**
   - **Tagline**: "Elevate Your Math Skills, Without Elevating the Cost"
5. **AccessibleAlgebra.com**
   - **Tagline**: "From Basics to Beyond: Math Education for Everyone"

Remember to check the availability of these domain names before making a selection.

FIGURE 12 OPENAI'S GPT-4 PROMPT 2

Not only did the website names get more relevant, but it one came with its own tagline! Out of all the options, I would have ended up selecting the MathZoomedIn.com one, which wouldn't have been found if I didn't prompt it with something more relevant to the output I wanted.

Another technique used to create very precise prompts, is to turn the generative AI into the prompt engineer for itself. By asking the AI to create a prompt that would give the best possible results for an AI, it can restructure your ideas in a way that makes the AI more responsive to your specifications. For example, in the last example, that prompt was generated by ChatGPT! The prompt used to generate the prompt was:

"*Create me a prompt that would generate me the best possible results for a generative AI to create a list of 5 website names for a math tutor website. The website will be done by setting up private lessons through zoom, running at the rate of $30 per hour. Its goal is to make math accessible for anyone at high school level or earlier, with math lessons ranging from Algebra 1 to Calculus 2*."

The feedback loop from ChatGPT allowed for my requirements to be clearly organized into an effective result. Using my own written prompt might have given similar results, but in the sake of optimizing the output from the AI, this is a highly effective method.

One more strategy that is very helpful when creating prompts is that generative AI works well when given context. When debugging code, including all the code for that file, and explaining the error that is happening gives the AI all the tools it needs to attempt to solve the problem. Missing the additional context means that it will have to predict what context you didn't include, which may lead to further inaccuracy. Including context doesn't have to be in a well formatted text file either, because of the way it reacts to any prompt given, dropping in dozens of messy error lines can still be translated and used by generative AI.

For this project, any usage of generative AI should certainly utilize these prompt engineering strategies. Doing so will save time from the user’s side, save money as some generative AI’s charge per prompt, and increase the accuracy of the task given to it. All the information provided by generative AI will not be correct, as it is only trained off the internet and isn’t the internet itself, but using techniques to modify the prompt will maximize the correctness possible.

6 Hardware Design

6.1 Illumination System

6.1.1 Cylindrical base design

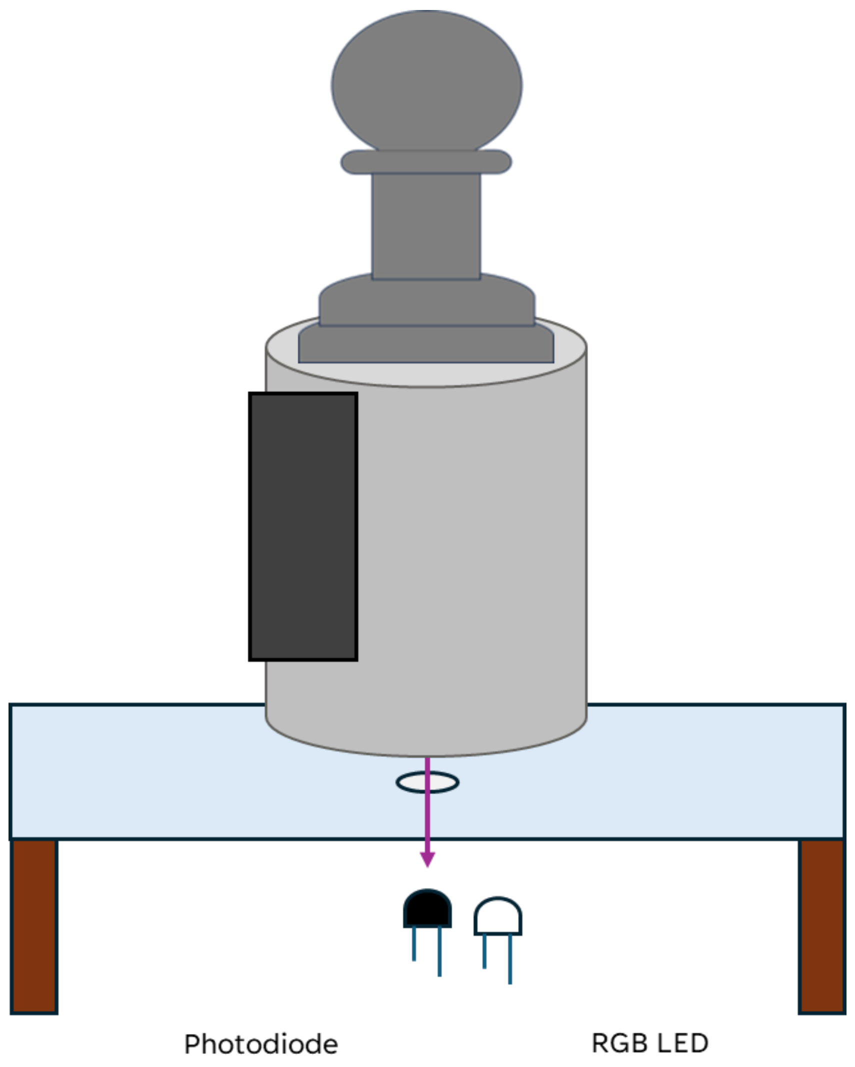


FIGURE 13 ZOOMED OUT CYLINDRICAL BASE ON CHESSBOARD

Figure 13 shows the chess piece on top of the cylindrical base. The chess piece will be attached to the top of this cylindrical base. This base will sit inside a circular ring that is on top squares. At the center of the ring will be the hole in the square where the light will travel through to get to the photodiode. The cylinder will be placed inside the ring to ensure consistent reading of the LED light by the photodiode under the square. On the side of the cylinder will be where the battery will be to power the LED. The battery is on the side so that it is easy to replace and takes up less space inside the base. The battery will have a battery life of a little under 50 hours. The cylindrical bases will all be the same size, so it is easier to make. There will be 32 cylindrical bases, one for each chess piece.

A) $P = V * I$

B) $Energy\ Capacity\ (Wh) = Battery\ Voltage\ (V) \times Battery\ Capacity\ (Ah)$

C) $\frac{Energy\ Capacity\ (Wh)}{Power(W)} = Battery\ life\ (Hours)$

FIGURE 14 A) EQUATION FOR POWER B) EQUATION FOR ENERGY CAPACITY C) EQUATION FOR BATTERY LIFE

Our LED consumes a current of 47 milliamps at a forward voltage of 1.28 v. If we plug this into the power equation, we get a power consumption of 60.16 mW. Now we must determine the battery for each piece. We had the same battery type for each piece to keep it consistent. This will enable for one design for all the pieces and its easier for the player to replace the battery since all of the pieces would have the same type of battery. The batteries must be small and lightweight enough to be on the side of the cylinders. However, we also wanted to have the highest battery capacity to have a longer battery life. These two preferences, however, contradict each other since batteries with higher battery capacities are usually bigger in size. We initially chose to have a smaller battery which gave us a smaller battery capacity, but it didn’t take up much space on the side of the cylindrical base. However, after some testing we noticed that the intensity of the light produced by the LED went down over time. When we first connected the LED to the battery, we could see a strong intense light on the IR card and after around 30 minutes we could hardly see the light on our IR card. This is because over time the voltage of the battery decreases. Since we were using a battery with a smaller battery capacity the voltage dropped fast because the battery life at the rate, we were using it was a few hours. Having batteries with high battery capacities causes this voltage drop to take longer spreading it over more time.

Our battery will have a voltage of 3 V since it is more than the 1.28 v needed for the LEDs. We would not need a battery of 5 V or 9 V since it will be much more than what we required. A capacity of 1000 mAh is usually found in larger batteries like AA or AAA rechargeable batteries, or specific types of lithium- ion or lithium-polymer batteries. Coin cell batteries are commonly found in small electronic devices, but their capacities are usually in the range of 220 mAh to 300 mAh. If we get a 3-volt battery of 220 mAh and plug this into equation B we get an energy capacity of 0.66 Wh. If then use this energy capacity for equation C we have a battery life of 10.97 hours. This will be the minimum battery life of a coin cell battery and considering that the average chess game lasts 10 minutes to an hour we have battery life for multiple games.

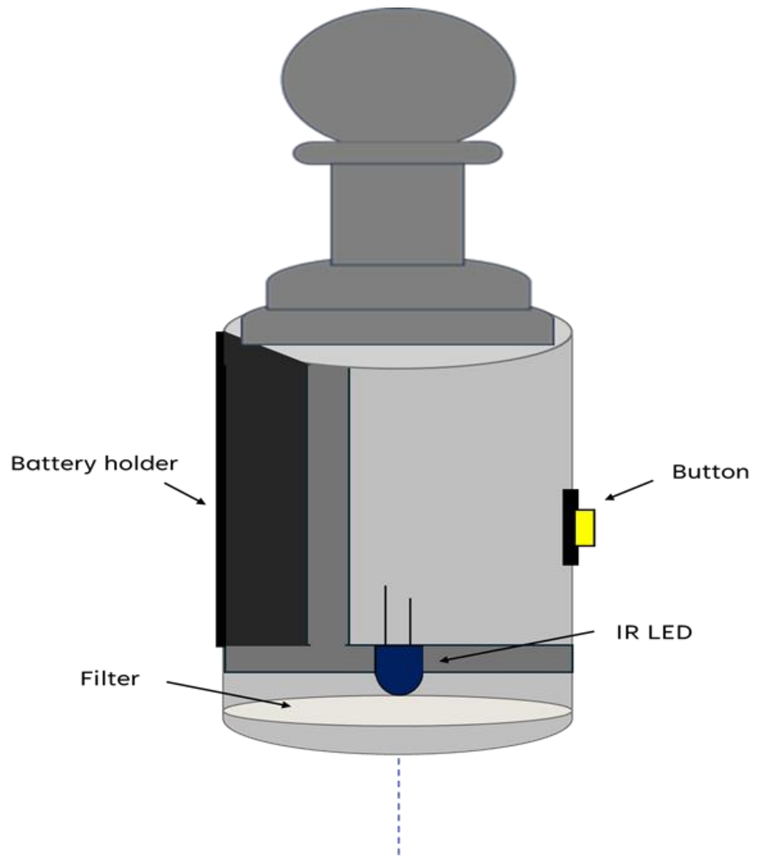


FIGURE 15 ZOOMED IN VIEW OF CYLINDRICAL BASE

At first we looked into coin cell batteries because of their small size and weight that would enable them to be placed on the side of the cylindrical base. We decide between many coin cell batteries taking into account their cost, size, and battery capacity. We also want the batteries to be 3 volts. Having the highest battery capacity is ideal so we started off by looking for this requirement. The coin cell battery with the highest battery capacity is the CR2477. This battery has a battery capacity of between 850 mAh to 1000mAh which is great for longer battery life. However, CR2477 batteries are larger than typical coin cell batteries due to this increase in battery capacity. The dimensions of a typical CR2477 battery is about 24.5 mm diameter and a thickness of 7.7 mm. The cheapest CR2477 batteries cost about 2 dollars each. This price isn't great considering that we need one for each of the 32 chess pieces, meaning that the minimum amount of money needed to power all the cylindrical bases would be 64 dollars. An argument can be made that even though the CR2477 batteries cost more than the normal coin cell batteries upfront the longer battery capacity would save you more money in the long run. Another battery that was considered was the CR2032 battery. It has a diameter of 20mm and a thickness of 3.2mm making it a smaller battery. However, the CR2032 battery has a battery capacity of around 200 to 240 milliampere-hours. This is much smaller than the CR2477 batteries, lowering the battery life by a lot. The counter argument to this is how cheap the CR2032 batteries are. They can be as cheap as about 0.20 cents per battery meaning a pack of 40 batteries would cost 8 dollars. The last battery we had in consideration was the CR2450 battery. CR2450 battery has a battery capacity that ranges from around 550 to 620 milliampere-hours. This battery has a significantly higher battery capacity than the CR2032 battery but less than the CR2477 battery. A standard CR2450 battery has a dimension of 24.5 mm diameters and a thickness of 5.0 mm. This is also bigger than the CR2032 battery but less than the CR2477 battery. The cheapest CR2450 batteries cost about 0.50 cents per battery. This is significantly cheaper than the CR2477 batteries but not as cheap as the CR2032 batteries. The CR2450 batteries seem to be a middle ground between the better performing but more expensive CR2477 batteries and the worse performing but cheaper CR2032 batteries.

The battery capacity of double A batteries depends on the type of material used to build the battery. The battery capacity of Alkaline double A batteries ranges from 1500 to 3000 mAh while Lithium double A batteries range from 2000 to 3500 mAh. Nickel-metal hydride rechargeable double A batteries range from 1000 to 3000 mAh. Double A batteries have an average diameter of 14.5 mm and height of 50.5 mm. Disposable double A batteries such as ones made with alkaline, and zinc have a voltage of 1.5 volts while rechargeable double A batteries like Nickel-metal hydride only provide 1.2 volts. Using the disposable double A batteries would make sense in this case since they have a higher voltage than what we need.

Triple A batteries are similiter to double A batteries where their battery capacity is affected by the material they're made of. Alkaline triple A batteries have a battery capacity of 600 to 1200 mAh, Lithium triple A batteries are from 800 to 1200 mAh, while Nickel-metal hydride rechargeable triple A batteries are from 600 to 1100 mAh. Triple A batteries have smaller battery capacities compared to double A batteries however triple A batteries are smaller than double A batteries. Triple A batteries are typically 10.5 mm in diameter and 44.5 mm in height. Triple A batteries are like double A batteries where the disposable batteries have a higher voltage of 1.5 volts and the rechargeable batteries have a voltage of 1.2 volts. We looked at lithium batteries for both double A and triple A batteries since they seem to perform better than alkaline batteries and the Nickel-metal hydride rechargeable batteries. For table *** we calculated the battery life of the cylindrical base using 2 double A and 2 triple A batteries.

TABLE 19 BATTERY SELECTION

Batteries	CR2477	CR2032	CR2450	Double A	Triple A
Battery capacity (mAh)	850 - 1000	200 - 240	550 - 620	2000 - 3500	800 -1200
Battery voltage	3V	3V	3V	1.5 V	1.5 V
Battery life (hours)	42 - 50	10 - 12	27 - 31	100 - 175	40 - 60
Diameter (mm)	24.5	20	24.5	14.5	10.5
Thickness / height (mm)	7.7	3.2	5.0	50.5	44.5
Weight (g)	10.5	2.9	6.6	23	11.5
Cost per battery	$2.00	$0.20	$0.50	$0.50	$0.25

We originally went with the CR2032 batteries but since they have a short battery life the voltage would decrease drastically. We ended up choosing the triple A batteries since they have a higher battery life and are cheaper compared to the other batteries with similar battery capacities. The triple A batteries have a smaller battery capacity than the double A batteries, but they are cheaper and smaller. This is important because we are restricted in size and since we must power 32 LEDs in the cylinders, having cheaper batteries enables the least amount of money to actually play chess. Using triple A batteries compared to double A batteries enables the cylinder to be smaller and lighter since there is less mass. The other coin cell batteries don't have much battery capacity except the CR2477 battery. However, the CR2477 battery is the most expensive battery that we compared. A drawback of using triple A batteries is that we would need to properly power the LED because we want the voltage of the batteries to be higher than that of the forward voltage of the LED.

Considering that we chose triple A batteries, we need to find a way to hold these batteries in place. The batteries will be outside of the cylinder for ease of access to take the battery on and off due to due to replacing the battery and to give us space inside the cylindrical base to for our components to power the LED. A way to hold the batteries outside the cylindrical base is to use a battery holder. Since we are using triple A batteries we need a battery holder that can hold these types of batteries. An advantage of using a battery holders is that it is easy to connect to the circuit to power the LED since there will be metal contacts at the end of each battery side. The top of the battery holder will then be connected to wire which enables an easier and stable build. If we just had a battery by itself the user must connect the right wire to the corresponding end of the battery and find a way to keep the wire in place. Using the battery holder we don't have to worry about the connection to the battery being moved since the wires will be soldered to the battery holders. We originally planned to also 3D print the outline of the battery holders to save on money however we would have to find metal contacts for this 3D printed battery holder. These metal contacts must then be placed in the 3D print and be secured to the battery holder to ensure that they don't move around. Once they are in place wire must be connected to the metal contacts to allow the battery holders to be connected to the circuit. We, however, did not decide to go with this idea because it required more time and more assembling. We instead went with the idea of buying the battery holders and then leaving a slot for the battery holders to slide into the cylindrical base.

Our original design had optical fibers in our Illumination system. The optical fibers would take the light that was coupled from the LEDs and would then transmit them on the other end at the bottom of the chess pieces. The bottom of each chess piece had a mirror at the bottom and filters to distinguish what type of piece it was. The mirror would then reflect the light back towards the photodiode. We would have used green LEDs in this design and had them on to one side of the chest board and have the light be carried by the optical fibers to the square. There would be one LED for each row meaning that 8 fibers would share one LED light source. We had many problems with this design, mainly with the optical fibers. Firstly, one of the problems with the fibers was that we had to get enough optical power to each piece so that the light could get reflected and read by the photodiode. Another concern was the even distribution of light into each optical fiber. All the fibers had to couple the same amount of light to get consistent readings. The easiest solution to this was to have an LED for each fiber, however we didn't have enough space for this because this would compromise the portability of the chess board. The biggest problem with the optical fibers was the positioning of the fibers. We had to have the optical fibers at a certain angle so that the maximum amount of light hits the photodiode. Having the fibers not move and hold them in place was a very big concern because even the smallest movement on the fibers could change the readings on the photodiode. We had to find a way cheap way to fix them in place that was easy to make. Another concern was about the divergence of the light coming out of the fiber was too big that most of the light wouldn't hit the photodiode. A solution to this would be to use a lens, however we would have to get lenses for all 64 squares. The lenses had to be small enough to fit under the squares and cheap since we would have needed so many. The lenes would have most likely needed to be angled and mounted into position. The lenses added so many factors into the design that was constraint by space. This design would have been complicated to build and hard to maintain consistent readings with.

The next design that we had for the illumination system was to use an IR LED under the square that would modulate and hit the bottom of the chess piece which would have a mirror and filters based on the type of piece. The light would then reflect off the bottom of the piece to the photodiode. We would have had to build a circuit to modulate the IR LED which would have added more to our PCB. The IR LED was chosen because it wouldn't interfere with the LEDs that light up the square enabling us to use green blue and red to light up the squares. The IR LED would have modulated to measure only the light coming from the LED and not from ambient light. This would have been a safety precaution from ambient light if we through ambient light would have been a big concern. However, since the chess piece would be onto the square most of the ambient light would have been blocked anyways. We also would have needed a lens for this design for each square to focus the light. Again, we would have to worry about the cost, size, how to position the lens and angle the lens. Another big reason why this design was not chosen was because it would have been harder to read since we would have to do signal analysis. Since we would have modulated the IR LED at 1 kilohertz there would have been a lot of data in a short amount of time.

A design change we made from our previous designs is an incorporation of a circular ring on top the center of the square. The purpose of this ring is to have a place where we can place the chess piece cylindrical base can sit in. The reason why we need this is to have consistent readings of the light intensity on the photodiode created by the IR LED. The ring on top of the board is supposed to be an area where we can get accurate readings from. The most intense light is directly under the IR LED. We would want the most intense light for our reading because it will make it easier to differentiate the different piece types. If we didn't have this ring the player would just place the chess piece anywhere on the square they want. However, this can create all a lot of variability on the light intensity that will go into the chess board. For example, if a piece is placed on the corner of a square the most intense light might be blocked while less intense light will be read. The ring will be a circle at the center of the square with a hole in the center of it for the light inside the cylinder to go into the square to the photodiode. The ring will have a diameter of 1.55 inches and a height of 3 mm from the top of the chess board. The chess piece should easily be able to fit into the divot without having to make many adjustments to try to fit it in. The divot will have a diameter of 1.55 inches and the cylinder will have a diameter of 1.35 inches. This will allow the cylinder to have some room to sit inside the divot.

We put a button on the other side of the cylinder from the battery holder to turn off and on the LED inside the cylindrical base. This way we wouldn't have to take the battery off and on to turn off and on the IR LED. Having a button would also be easier than taking off and on a battery. We originally decided to not use button or switches since the dimensions of the cylindrical base are already small holding it and not touching the battery holder or the switch/ button would be tricky. We also originally were using CR2032 batteries which are small and would have been easy to slide off. After switching to 2 triple A batteries, we realized that taking off and putting on the batteries would require a lot of effort just to turn on and off the LED and that is why we decided to incorporate a button. A concern about using switches or buttons is if it accidentally turned off or on you won't be able to see the infrared light emitted by the LED causing trouble playing the game or draining the battery. To counteract this, we could incorporate some type of mode to make sure that the LED is on and would be able to measure the IR light. We can also check that the LED is on by placing it on the board and seeing that the LEDs under the board light up and show possible moves. The problem with this is that it wouldn't show the measured intensity of the IR light. After thinking about these two methods, we realized either way we turn off or on the LED in these methods we would have to interact with each of the 32 cylindrical bases at least once at the beginning and end of the game. If we decide to take out each of the batteries, we would have to interact with the chess piece base twice and would require more effort than pushing a button. After realizing this we decided that using a button would be the easiest method to turn off or on the LED.

Other method of turning off and on the LED, we thought of was to somehow have the LED turn on every time it would touch the chess board and off when it was picked up. We thought of using reed switches on the bottom of the chess pieces to turn on once on the board. However, the switch would have to be a the bottom of the chess piece and that's where we would have filter to change the intensity of the light. Another idea was to have a metal contact on the chess board. Once the piece was placed on the board the metal contact would enter the piece through a hole in the chess piece. Once the metal contact was inside the chess piece the circuit would complete, and the LED would turn on. The problem with this is that the Chess pieces have to be place properly so that the metal contact would go into the piece. This means the player would have to move around the piece to have the contact go into the piece. Another idea was to have the power be on the top of the board and once the piece was placed on the square the power would turn on the LED inside the piece. To do this we would have to find a way to get power to the top of the chess board. This would have required wires that could have mees up the lighting of the square. This also would take up space under the square which is already limited. Another idea was to use magnets strips on the top of the board and once the piece was placed down the circuit inside the chess piece would close turning on the LED. We, however, thought there wasn't enough space in the chess piece to do this since we would also have filters near the bottom.

Another major consideration we had to turn on the LED once the chess piece was on the board was to use induction. Induction is the generation of voltage across a conductor when it is exposed to a changing magnetic field. This can be done when a conductor, like a wire coil, is placed in a changing magnetic field, the magnetic flux through the coil changes, inducing a voltage across the coil. This induced voltage can then drive an electric current through the conductor if a complete circuit is present. What we would have done is have a circular coil the size of the chess board under the board. This would require us to make another PCB to properly have induction. The benefits of this would have been that the user wouldn't have to do turn on or of the LED. This would have made the project more player friendly and easier to use. However, induction had some drawbacks. First of all, the LEDs would not produce the same intensity depending on where on the board the chess piece is. Chess pieces on the center of the board would have their LEDs produce more intense light compared to those chess pieces who are on the edges or corners producing less intense light. This would greatly impact the readings on the photodiode causing the microcontroller to misidentify the piece type. Secondly, this design, while making it more player friendly, would be harder for us to build. We would need to have another PCB for this to work while there are simpler and cheaper solutions to turning off and on the LED. We realized that with every design we made we had to decide between what is easier for the player to use and what is less challenging for us to build and be cheaper.

At first, we wanted all the circuits for the LED, the battery and the filters to be all inside the chess pieces. To do this we realized we would have to 3D print a hollow version of each of the chess pieces. Some pieces would have been harder to build in since they are smaller compared to others, for example a pawn and a queen. To compensate for this, we would have to alter the size of each piece to allow more room for everything that would be needed to be inside the piece. There are two problems with this idea though. The first problem is that the chess pieces wouldn't look like normal chess pieces. They might have to be bigger in some areas compared to the normal pieces. This would make our project less aesthetically pleasing. The second problem and biggest problem is that we would have to redesign each chess piece type so that we could 3D print them. This would consume a lot of time and would still have many problems that would need to be solved such as where the battery will be and how we would access it. We thought of having the battery be inside the top of the chess piece and have the top be removable by twisting it off. This would also require us to design this so that we can 3D print it. At this point designing all of this to be 3D printed was looking to be a lot of time and would probably still have problems with space inside the chess piece. That is when we thought of the design to put the circuit, LED, and battery under the chess piece in the cylindrical base.

The cylindrical bases under the chess pieces will be made by 3D printing them. We chose to use a grey filament because it is between black and white so it would go with each piece color. We decided to use one color for all the cylinders so that we can use the same cylinder for each piece type no matter the chess piece color. We 3D printed a hollow cylinder so that the components can be placed inside. We created an indent for the battery holder can go slide on the outside of the cylinder. This will also help the battery holder be attached to the cylinder. Inside the cylinder we also have a horizontal plane with a hole in the center for the IR LED to be placed in. We call this the bridge because it connects the two wall of the cylinder. The top of the IR LED would go inside the hole and the anode and cathode will be sticking out on the other side. The bridge should prevent the LED from moving around however me must make sure that all the LEDs are place straight down and not an angle since that would mess with the light intensity readings. The top of the cylinder will be partially enclosed with a big hole in the center to allow us to access the inside of the cylinder. The top is not completely open because we need to attach the cylinder to the chess piece. From the opening on the top, we can connect the batter holder with the anode and cathode of the LED with wires. At the bottom of the cylinder will be the filters. The filters will not be at the actual bottom since they will be inside the piece a bit. This is to protect the filter from getting dirty from contaminants on the chess board such as dust. We want the filters to be clean as possible since it being dirty can cause the light intensity to drop.

There are many methods to connect the bottom of the chess piece and the top of the cylinder. One method is to use glue. This is a simple and cheap option, however gluing these two parts together will make it hard to access the inside of the cylinder if needed. If the inside needs to be accessed the glue between the two parts has to be removed to be able to get to the top of the cylinder. After being separated and the user wants to reconnect the pieces together, they must reglue it together using new glue. Another cheap method is to use double sided tape to attach the chess piece and the cylinder together. Using double sided type is easier than glueing and if the user can reattach the pieces easier. The problem with double sided tape is that it might not be held together as well as glue and might have a hard time handling the weight of the cylinder. Another concern is that the chess piece might not sit evenly on the tape making it look like it is tiling on way. A more expensive method compared to the two past methods is to use magnets to connect these parts together. A magnet can be glued on the bottom of the chess piece. The other magnet would be on the chess piece. There are many different ways to design having the magnet on the cylinder. One way is to have two small magnets on the top of the cylinder. Another way is to have a magnet with the shape of a ring that has the hole size the same as the hole on the top of the cylinder.

When building the illumination system, the concern of how much light was received by the photodiode was a top concern. If not, much light was received by the photodiodes. There were 2 ways to solve this problem. The first way was to get a more powerful LED which would increase the intensity received by the photodiode. However, usually more powerful LEDs require a higher forward volage and current. This is a problem because since we are using triple A batteries for the bases the battery life of these batteries will be shortened due to more current and voltage being used. We could fix this by getting better batteries but then this would cause the bases to be too big.

The other way to make sure that the photodiode receives the proper amount of light is to use lenes. Lenses would be used at the bottom of the hole on the chessboard to focus the light onto the photodiode. The lens would prevent the light from diverging and if the photodiode is placed properly will be at its focal length. If the photodiode is placed at the lenes focal length the light intensity would greatly increase because all of the light would focused on to the photodiode. The one concern with this is that the photodiode can be oversaturated. However, the photodiode does not have to be at the focal length it just has to be within the light path to receive more light because the light would be converging due to the lens instead of diverging. The problem with using lenses is that the lenses have to be small enough to fit under a square and be cheap enough to have 64 of them for each square. The lenses must be mounted in place and held there for all 64 squares. The lenses could be placed at the bottom of the cylindrical base however this would make the bases taller than they are already. Luckly, we don’t have to use lenes because our IR LED provides enough optical power for the photodiode to be able to differentiate 7 different pieces.

Chess piece base sizing

We designed our 3D printed bases using Tinkercad. This was chosen since it was the simplest to learn in the fastest amount of time. We also choose to make the cylindrical bases out of gray filament since it is between black and white. Now will be explained how we designed our bases.

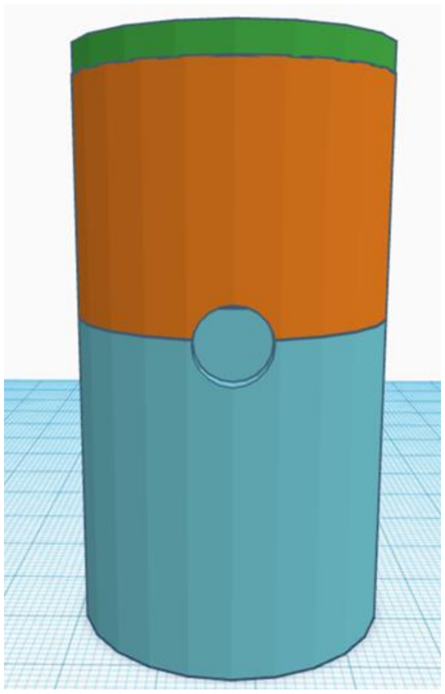


FIGURE 16 FRONT OF CYLINDRICAL BASE

Figure 16 shows the front of the cylindrical base design. We can see that the base is divided into three. The top half being orange, the bottom half being blue, and the green being where the chess piece will be sitting on. We divided the cylindrical bases into three to make it fast and easier to print since doing it requires less support. Another thing that can be noticed is the hole in between the top and bottom halves. This is where the button will be placed and glued to turn on and off the IR LED. The hole for the button has a diameter of 8 mm or 0.314 in to enable the button to fit nicely. The diameter of the entire cylindrical base is 34.29 mm or 1.35 in. This was chosen according to the size of the square which is 1.75 inches. The height of the cylindrical base is 2 and a half inches or 63.50 mm.

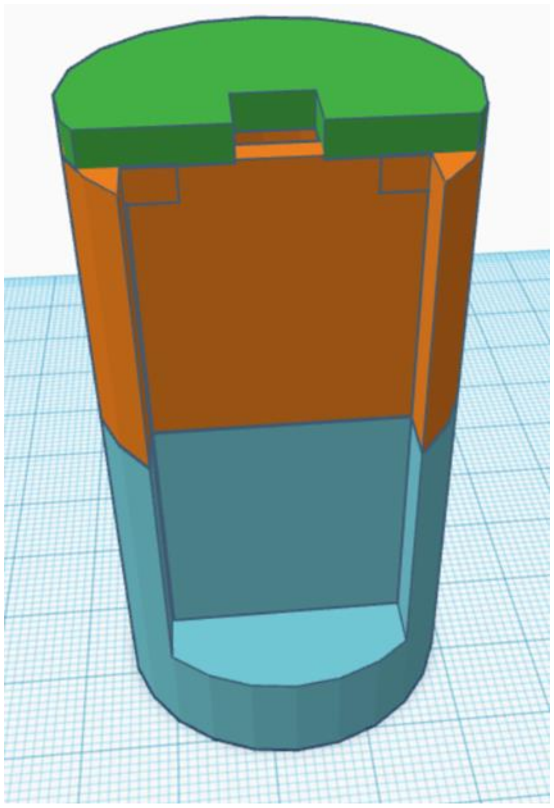


FIGURE 17 BACK OF CYLINDRICAL BASE

Figure 17 shows the back of the cylindrical base design. Here we can see the indent created in the back for the battery holders that will hold the two triple A batteries. This indent goes about 9.5 mm into the cylinder and has a height of 50.50 mm or 1.98 in. This indent has a length of 24.08 mm or 0.948 in enabling the battery holder to fit inside the indent. We can also see the green top which will hold the bases will have a cut out for the wires of the battery holders to go inside the cylinder to complete the circuit. This cutout will go 6 mm into the green semicircle shape and 7 mm in length to give enough room to run wire into the base.

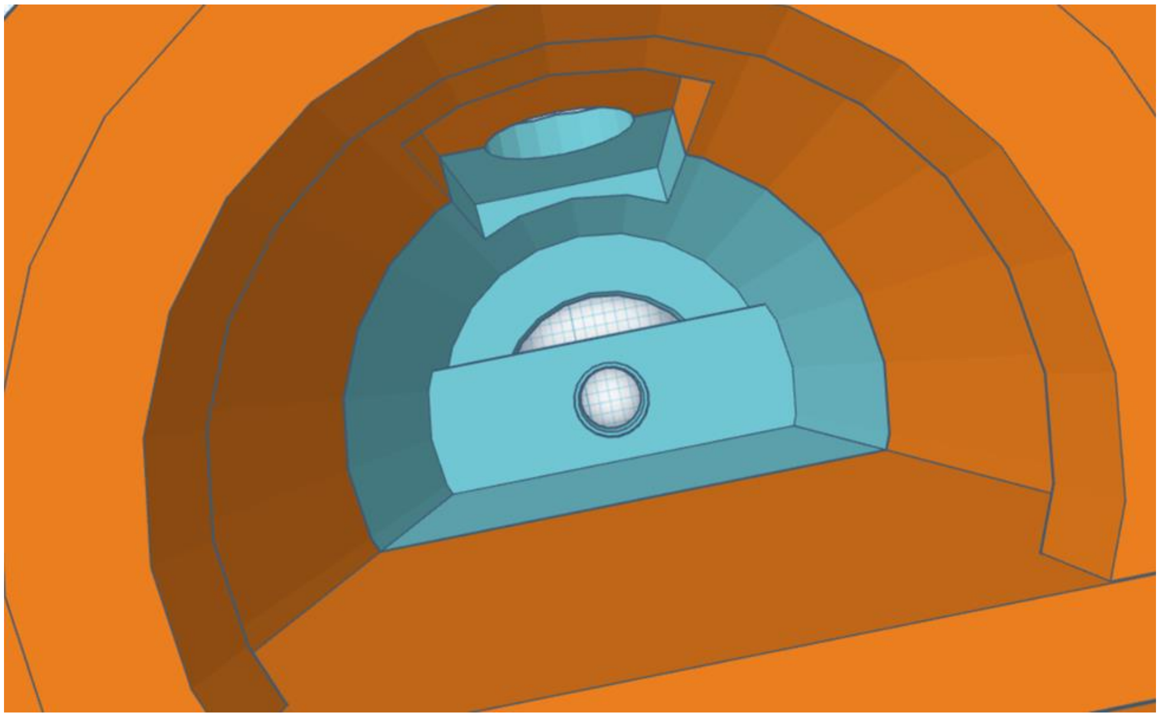


FIGURE 18 INSIDE CYLINDRICAL BASE

Figure 18 shows inside the cylindrical base. In between the two top and bottom half we can see a rectangle cutout for the button. This rectangular cutout has a height of 23.63 mm and a length of 12.7 mm. On the bottom half of the base is a bridge between the walls. On the bridge there is a hole for the IR LED to sit in. There is a smaller hole that goes all the way down that has a diameter of 5.20 mm, and the bigger hole has a diameter of 5.8 mm and height of 1.40 mm from the top of the bridge.

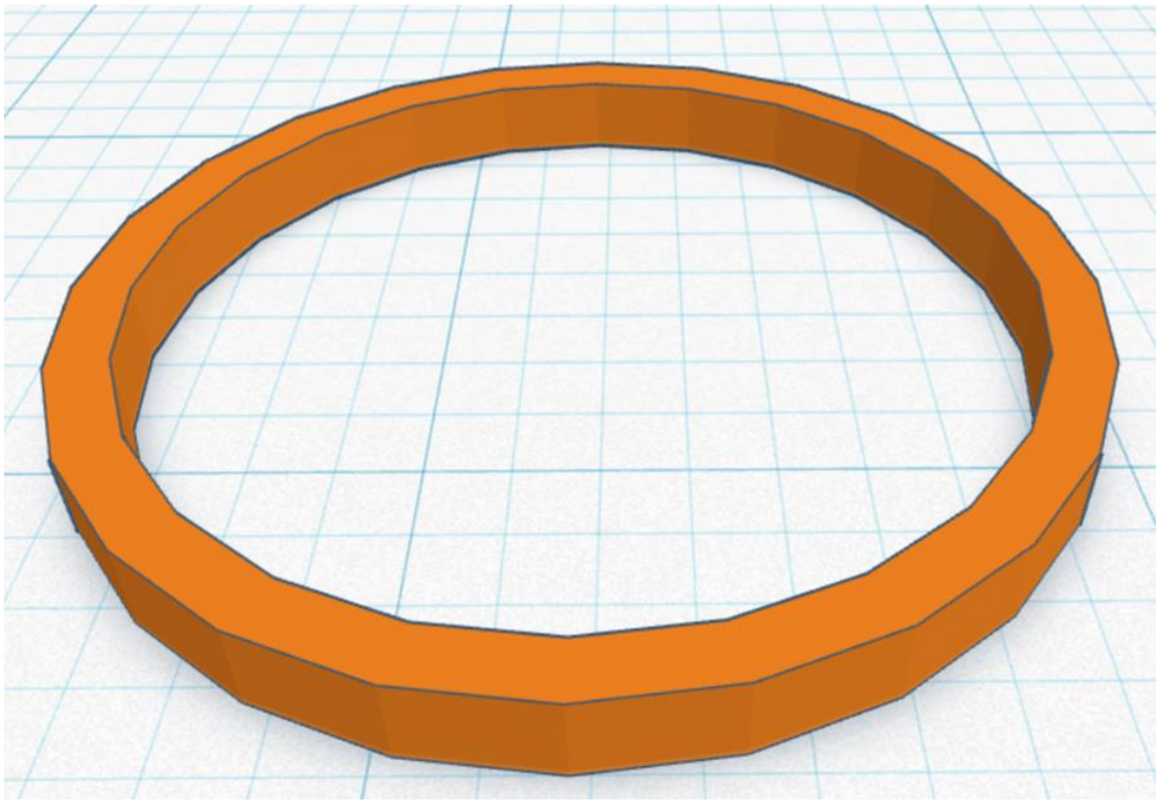


FIGURE 19 CIRCULAR RING ON TOP OF THE SQUARES

Figure 19 shows the rings on top of the squares to keep the cylindrical base aligned with the photodiode. The ring has a height of 3 mm and has a diameter of 1.55 inches or 39.37 mm. The ring has a thickness of 0.05 in meaning that there is 1.45 in of space inside of the ring for the cylindrical base. Since the cylindrical base has a diameter of 1.35 in and the ring has a diameter of empty space of 1.45 in there will be .10 in of wiggle room for the cylindrical bases.

6.1.2 Honeycomb structure

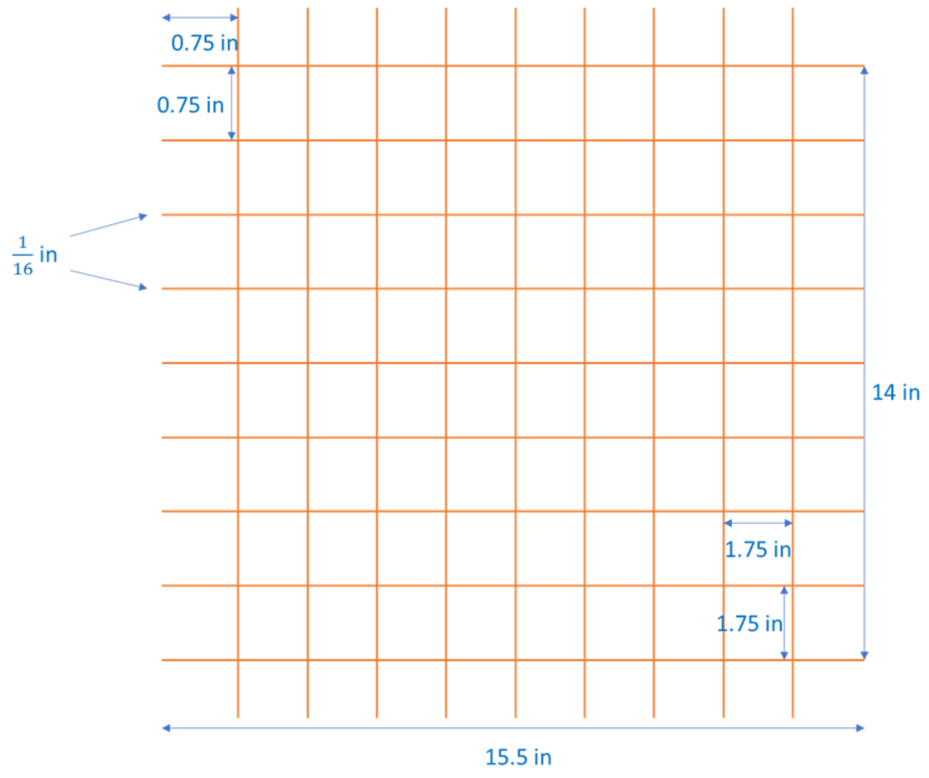


FIGURE 20 TOP VIEW OF HONEYCOMB STRUCTURE

The honeycomb structure serves as a pivotal component in our design, made to fulfill its primary role of preventing light leakage between squares. The intention is to confine the light emitted from each square's LED within its designated space. To achieve this objective, we opted for a lightweight and easily removable material, namely cardboard, known for its practicality, cost-effectiveness, and accessibility for potential repairs.

The dimensions of the honeycomb structure are planned to ensure optimal functionality. The walls of the honeycomb, designed to block light and create defined boundaries for each square, have a width of 1/16 inches. Each individual square within the honeycomb measures 1.75 inches by 1.75 inches (length x width), providing a well-contained space for the LEDs and photodiode. These components are positioned inside each square to illuminate the square above and capture the intensity of light from the bottom of the cylinder on top of the square.

The playing area, consisting of 64 squares, is fully enclosed on all sides, forming a 14 inch by 14 inches grid. Notably, the honeycomb structures on the outer edges are partially enclosed, featuring only three walls. These outside cells, positioned under the outer border of the chessboard, are slightly smaller at 0.75 inches by 0.75 inches (length x width).

To stabilize the honeycomb structure, particularly in the four outer corner cells, blocks of wood are placed to prevent any movement. Meanwhile, the remaining outside cells, devoid of any internal components, serve as potential extra space if needed. The total footprint of the entire honeycomb structure measures 15.5 inches by 15.5 inches (length x width), with a height of 1 inch.

The construction of the honeycomb involves precision notching, with columns and rows featuring specifically placed notches to create the walls. The notches in the columns and rows are positioned to form a grid. The honeycomb structure would be built by having notches from the top of the cardboard to about halfway down for the walls that will be the columns (from top to bottom of chess board). The walls that will be the rows (from left to right) will have notches from the bottom of the cardboard up to halfway. The first notch in both column and row walls will be 0.75 inches in and then the rest of the 8 notches will be at 1.75 inches apart from each other. From the last notch the carboard will extend another 0.75 inches until the column or row ends. These honeycomb structures are designed to sit atop the Printed Circuit Board (PCB) and beneath the chessboard, forming an integral part of the intricate chessboard system.

6.2 Chessboard and Chess Pieces

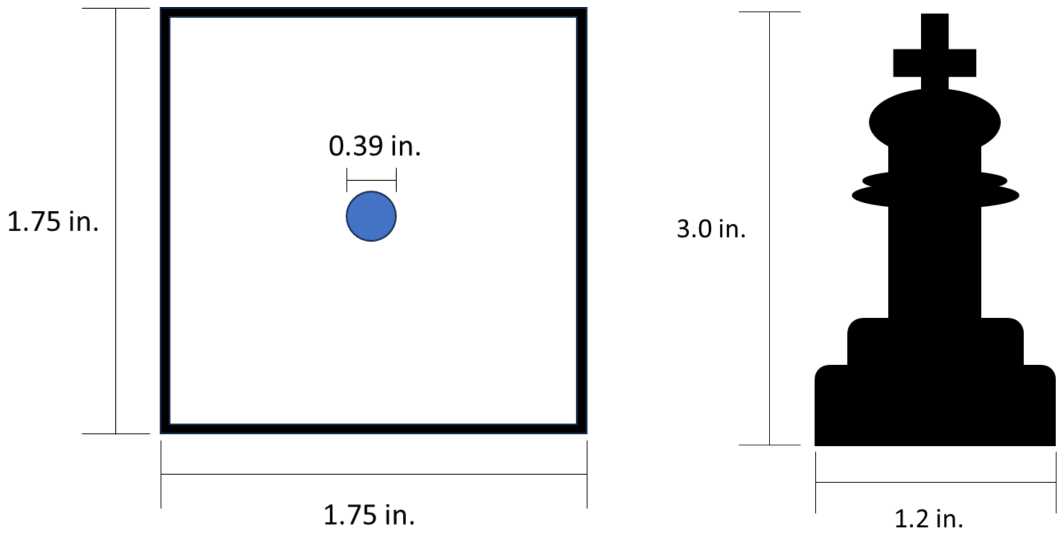


FIGURE 21 SCHEMATIC OF EACH SQUARE ON THE CHESSBOARD ALONG WITH THE DIMENSIONS OF THE KING.

For our chessboard, the sizing of the squares is an important aspect to determine and get right. We must figure out the right dimensions of the square and the opening in each square that not only enables our project to work through the use of optics, but also that will allow for the construction of an overall project that is compact and portable while being aesthetically pleasing. The dimensions that we decided to go with are a square size of 1.75 inches long by 1.75 inches wide. We chose these dimensions because a chessboard is an 8 by 8 grid, so it would create a chess board that is 14 inches long by 14 inches wide. When looking at other chess boards and their dimensions in person, we felt that these dimensions were an appropriate size considering our parameters for this project. Most chess boards we compared to that were meant for casual use and could be bought at any store were typically smaller in size than our dimensions. When comparing to a typical professional chessboard that is used in tournaments and such, we noticed that those boards are much bigger in size. We believe that choosing dimensions that are somewhere in the middle of the chess boards that we were comparing with was the best choice since our project is aimed for beginners to learn how to play the game while still heavily incorporating electronics and optics to create a unique product.

Once we determined our square size, we had to figure out how big the opening in each square would be so that enough light from the IR LED in the base could come through the square so that the photodiodes can detect and determine which piece it is. We discussed many different options for the shape of the opening as well since that would have a big factor in how much light could come through and how the positioning of the components under each square would be. We decided to make the opening in each square a circle with a diameter of 10mm, which is roughly 0.39in. We chose this shape to start with, as we believe this would be the easiest to make and incorporate into each square of our chessboard. We decided on the diameter of the circle to be 10mm as we needed room to insert a lens at the bottom of each square if we needed the light coming from the pieces above to be enhanced and focused to the photodiode. The diameter of the circle, aesthetically, doesn’t look too big or small in the square.

For our project, determining the square size and the size of the opening in each square was important since it has such a big impact on the components and parts underneath the chessboard, but the size of each piece also is very important for our project to work as intended. Chess piece size and square size are closely related to one another. Typically, the king is both the tallest piece on the board and also has the largest base diameter. The king’s base diameter is usually around 75%-80% the size of the square. After determining we would need to attach a ring to the top of each square that would be used to center and contain the piece on the square, we figured out that base diameter wasn’t going to be related to square size as precise as what normal chessboards are determined by. We determined the ring diameter that was put on each square to be 1.55 inches on the outside of the ring and 1.45 inches on the inside of the ring. We set the base diameter to be 1.45 inches as that would allow each piece to fit snuggly inside the ring without any potential movement but not be too snug and tight to cause problems with picking and placing the piece you want to play. Since we couldn’t 3d print the pieces, we ended up buying a chess set with pieces that would fit with our bases.

The king piece was 3 inches tall and had a base diameter of 1.2 inches. The smallest piece, the pawn, was 1.48 inches tall and had a base diameter of 0.94 inches. We need to make each base for each piece have the same base diameter to make sure that no matter where any piece is placed on a square, we always got a reading from the photodiode and the board can then show the movements of any piece that is about to be played. We can't have different size bases for the different piece types because it might cause the board to not accurately and properly detect the piece and be able to work as intended.

The height of the chess pieces isn't as important as its base diameter when relating to square size. As shown in Figure 13, the height of our king piece will be 3.0 inches. The king piece height is usually around double the size compared to its base diameter. The heights for all of the chess pieces will vary depending on the piece type and it will look normal and appropriate while also aesthetically appealing as we trie to put our own unique designs and influences into it.

6.2.1 – Differentiating Chess Pieces

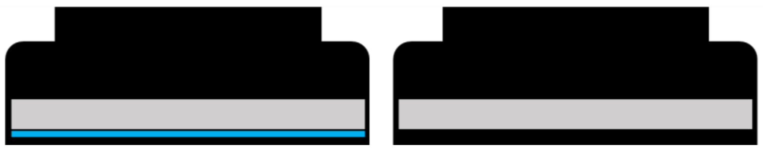


FIGURE 22 BASE OF CHESS PIECES

To differentiate each piece, we plan on putting a set of ND filters in every single base to serve as the way to differentiate each piece type. As shown in Figure 14 above, all of our chess pieces will have at least 1 ND filter, if not a combination of ND filters to help us get the most accurate and consistent outputs. After determining that the "initial value" would be a base without any filters, we then put filters on that would slowly decrease the intensity of light for the rest of the piece types so we can differentiate them based on how much light is transmitted towards the photodiodes. There are different values of filters that we used to differentiate the pieces. Hopefully we only need one kind of filter for each piece type, but we can adjust to incorporate as many filters as we need to properly differentiate each piece. The filters won't be placed on the bottom of the pieces but instead slightly inside them so that they don't get damaged or worn down due to usage during a game. They will be positioned slightly inside but not too far inside to where the values of the reflected light will be altered or affected.

6.3 ADC for Piece Identification

This section will cover the process of designing the ADC subsystem that will be used to send the analog inputs to the microcontroller. With the transimpedance amplifier circuit implemented, a source voltage does not need to be supplied to the photo diodes and they cannot exactly be connected in any way to simplify the overall matrix. This means we had to find some way to constantly read 64 analog inputs from each of these amplifier circuits.

The Teensy 4.1 only has 18 analog inputs so the first problem that needs to be solved is how to cut down on the number of analog inputs. Part of our research was looking into multiplexers, specifically analog multiplexers. By using these multiplexers, we can use an eight 8-to-1 multiplexers to read all 64 analog inputs. This would mean that only 8 ADC pins on our microcontroller would need to be used.

When deciding how to connect the multiplexers, we found out that we can connect each address line pin of all of the multiplexers together. By connecting them this way, only three pins on the microcontroller would need to be used to control all 8 multiplexers. Each multiplexer will be responsible for a specific row of photodiodes. By changing the address input, we can then select which column would be sent over to the microcontroller. By using this method, we can constantly monitor all 64 photodiodes to find any changes about the state of the board. The CD74HC4051E has an active low enable pin so it must be connected to a ground in order to read from its inputs.

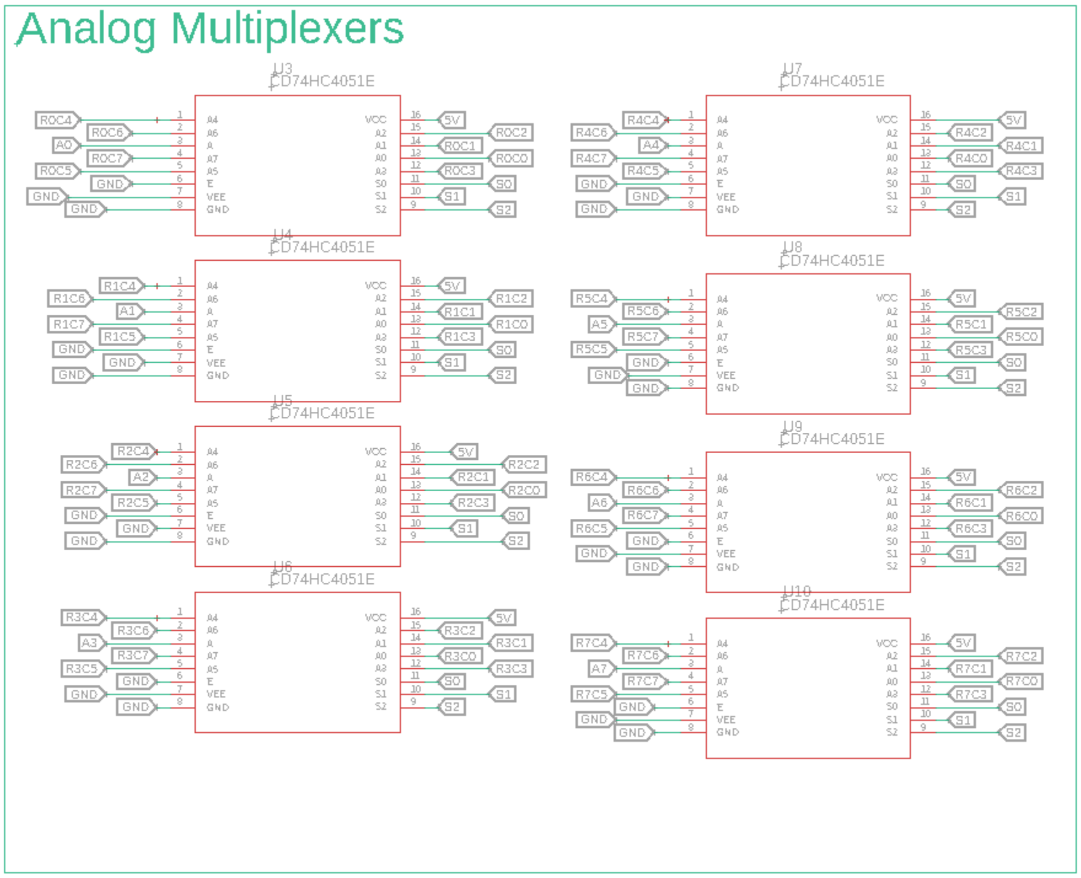


FIGURE 23 ANALOG MULTIPLEXERS FOR ADC

6.4 Battery & Power Systems

Responsible for providing reliable and consistent power, this set of systems ensure the seamless functionality of the remaining electronic components within the design. The careful integration of battery and power management solutions is imperative to support the needs of the project as its success hinges on the reliability and adaptability of its power supply systems, which make them indispensable components in the pursuit of optimal functionality and operational longevity.

6.4.1 Battery Management & Safety Systems

On our project, we entrusted most of the battery safety features to a single board called a Battery Management System (BMS), as discussed in a previous section, we chose for this task the HX-2S-JH20 protection board manufactured by Jessinie. This PCB offers many forms of protection to the battery, and by extent the rest of the system as well as its users. It also possesses capabilities to aid on the proper process of recharging the batteries, which will discussed in depth in this section.

Unfortunately for us, the documentation on this specific battery management module was scarce and hard to find. However, this component is an essential part of our system and therefore, to properly understand its role in the project we need to analyze the circuit board and identify its functions. Most importantly, its charging balancing feature, which will be extremely important for the life and operation of our battery pack.

The HX-2S-JH20 board contains a HY2212-BB3A chip designed by HYCON Technology Corp, which is used in Li-Ion and LiPo multi-cell battery packs for single-cell lithium battery charge balance control. Its purpose is to oversee the recharging process of the battery and ensure all cells receive proper charge. The following circuit diagram displays the board’s charge balancing system that is spearheaded by the BB3A IC and was recreated by analyzing its datasheet and relating it to the remainder of the management board’s circuitry.

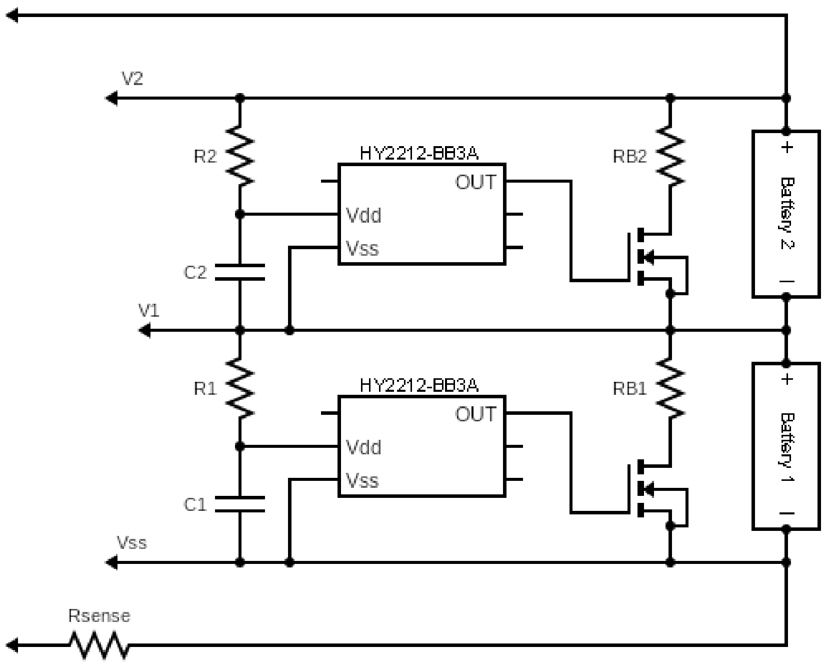


FIGURE 24 CELL CHARGE BALANCING CIRCUIT

We can analyze this diagram to understand how this simple circuit makes a whole lot of difference in our battery's efficacy. Once each cell reaches its maximum voltage, 4.2V for our 18650 cells, the MOSFET will enable current to flow through the resistors labeled RB, which will discharge their respective cells allowing for the remaining cells to charge without stepping over the voltage limit for that cell. This way, all cells get charged to 4.2V, so we can fully take advantage of the entire capacity for each cell.

The reason why we need such a system is because during charge and discharge of our battery, battery cells may not be completely in sync, meaning that some cells may lose or gain charge in different speeds. That is important because when charging our battery, some cells could reach maximum voltage prior to others and if we stopped charging all cells once the first cell reached its maximum voltage level, it is likely that much of battery's total capacity would be going to waste as other cells still need to finish charging and thus our battery life would be considerably diminished. Conversely, we do not want to keep charging the cells that have already reached maximum capacity to avoid overcharging and damaging them, possibly even causing them to fail.

Besides the balancing function, the board also includes several functions which utilize a HY2120-20CB IC to oversee the protection systems for the battery. This chip is connected to larger 80N03 MOSFETs and monitors the voltage of the battery to control charging and discharging, specifically to protect the cells in situations where voltage overcharge and overdischarge as well as charge and discharge overcurrent conditions would occur. This works by disabling the MOSFETs when these conditions are triggered, and thus cutting off the remainder of the circuit from the battery.

From the information above we can devise a simple diagram, such as the one seen below, to help visualize how this board will fit into the larger circuit and observe the overall functionality of our project's power systems.

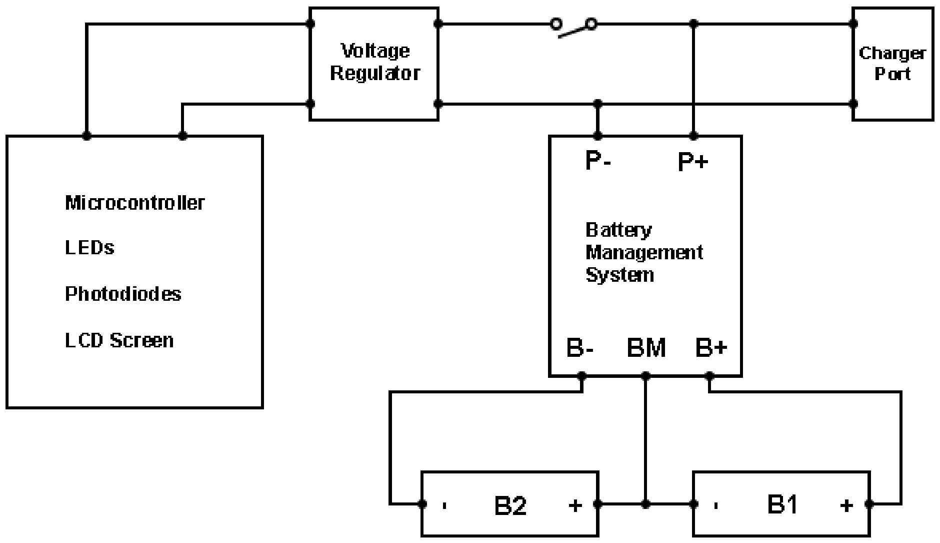


FIGURE 25 POWER DISTRIBUTION DIAGRAM

As evident in the diagram, we heavily emphasize on the need for safety while working with these batteries. Although usually safe, when improperly handled, lithium batteries can bring serious risks to the integrity of the project as well as to the client's well-being. Following this design, we make sure no cell is connected directly to the remainder of the circuit, where it would be most vulnerable to power spikes and other anomalies that might be harmful to it. Instead, we use our management system board as an interface to the batteries, equipped with several protection systems, it will drastically improve the reliability of our pack.

6.4.2 Reviewed Power Consumption & Battery Capacity Estimates

We had to make slight adjustments to the size of the battery due to the availability of parts. We are still using a design including two serialized groups of cells, each of which contain three 18650 batteries, for a total of six cells. The difference is, instead of 3250mAh, each cell has a nominal capacity of 3000mAh, which caused us to have to redo our power estimations. Since we are already decided on the converters we used, we can also include them into the equation, as their efficiency at stepping down our voltage will make for important data to consider.

Considering the setup described above, we have a battery with a total nominal voltage of 7.4V and a capacity of 9000mAh, making for a total wattage of 66.6

Watt-hours. Studying the LM2596 datasheet, it is revealed that we can expect a power efficiency rating of around 80% from our converters. That shoves off a good chunk of our total battery capacity, taking our effective battery to only 53.28Wh.

Still, we believe that this battery will be more than enough to meet our quota. We believe that the expected power consumption may fall well between the values predicted during our preliminary research stage, but even extrapolating our predictions and expecting a power consumption as high as 5Wh, we are well within the range of a 10-hour battery life cycle, blowing away our initial expectations of having only four hours of operation between recharge cycles.

6.6 – Photodiode System in Chess Piece Identification System

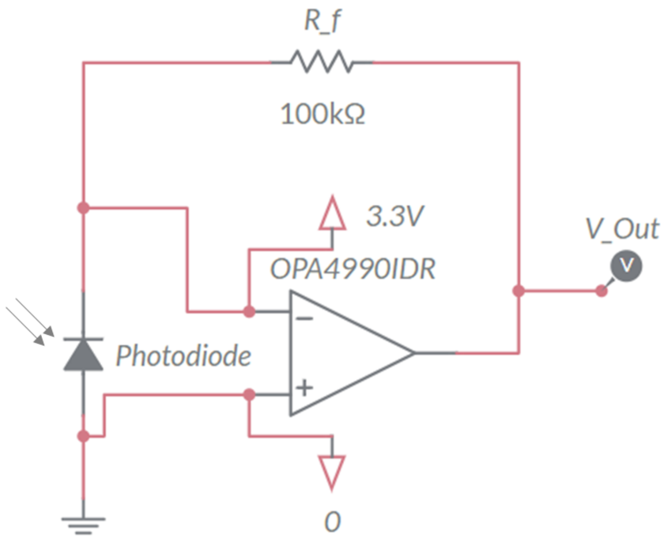


FIGURE 26 THE CIRCUIT USED TO MAKE OUR PHOTODIODE WORK.

For the chess piece identification system, we need to make a circuit that will allow our photodiode to work properly and be able to provide an output signal that can be read by the microcontroller so it can detect and identify a piece on a square. To do this, we had to build a transimpedance amplifier circuit for our photodiode. A transimpedance amplifier circuit is a current to voltage converter circuit. A transimpedance amplifier circuit will take the low-level photocurrent produced by the photodiode when light shines on it and convert that into an increased voltage output. The circuit is designed to enhance the output signal that the photodiode produces so that we can have a larger range of values to use to differentiate the different chess piece types. The transimpedance amplifier circuit we built for our project is shown in Figure 18.

The reason why we needed to build a transimpedance amplifier circuit is because trying to measure the photocurrent produced by the photodiode would have been too difficult to do for our project and its specific goals. The photocurrent produced in the circuit is very small given that the input power of the reflected light from the bottom of a piece is typically either milliwatts or microwatts. The photocurrent that is generated with that kind of input power is not even one microamp. We would not have been able to differentiate between the different chess piece types measuring the photocurrent on the scale of less than one microamp. That measurement scale does not provide our project with enough wiggle room and large enough ranges to be able to differentiate the pieces while also maintaining excellent accuracy of our piece identification system.

By converting the output signal to a voltage measurement that reads in volts or millivolts provides our project with a greatly extended range for values to differentiate each piece with. If we could get enough input power coming from the reflected light, we could potentially have our ranges for the different pieces be as large as 0.4V. Being able to differentiate the values associated with the different pieces by as large as possible would be incredibly helpful for us to hit our identification accuracy goals.

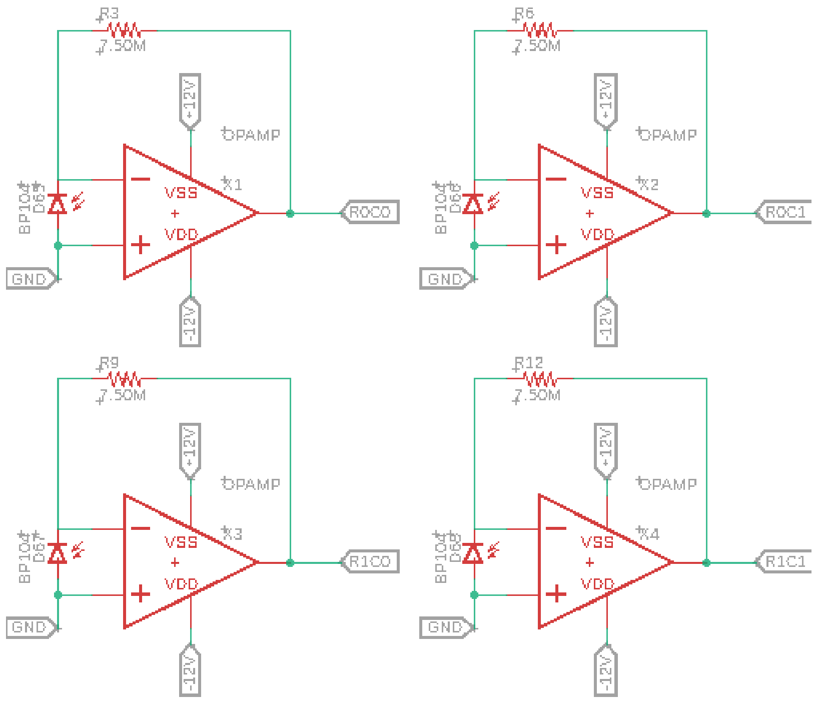


FIGURE 27 2X2 ARRAY OF TRANSIMPEDANCE AMPLIFIERS

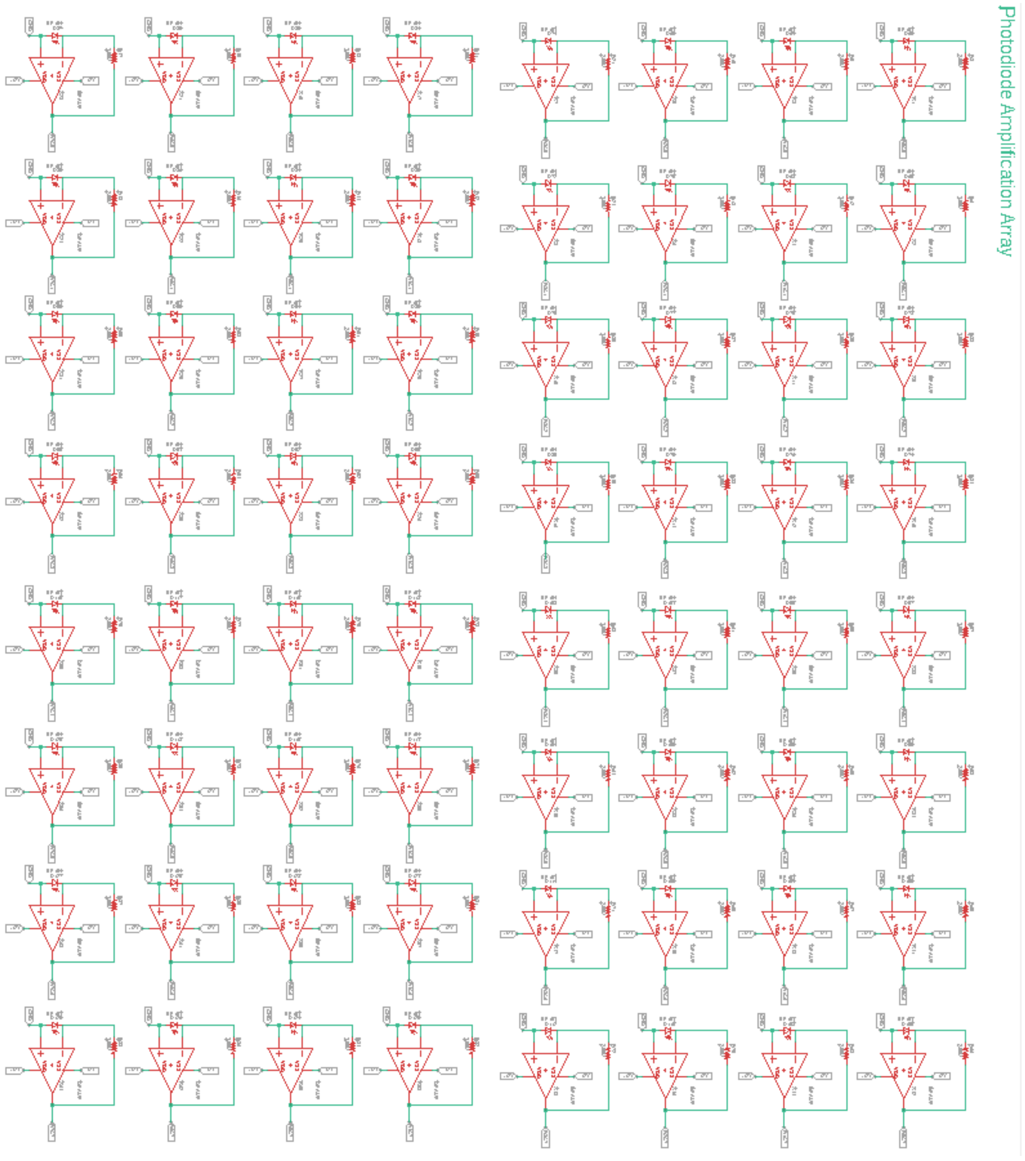


FIGURE 28 FULL ARRAY OF 64 PHOTODIODES

6.7 LED Matrix Subsystem

This section covers the design and implementation of the LED matrix. The LED matrix is one of the most significant subsystems of this project. As described previously, this matrix will cover the entire bottom of the playable chessboard so that each square contains an LED. By using this LED matrix, we can then relay information to the players by using these lights.

We want to keep the design of this subsystem as simple as possible to ensure the highest level of success when implementing it into our final design. As discussed previously, the LEDs will be connected so that LEDs in a row will share a common node for their cathodes while LEDs in a column will share a common node for their anodes. This design reduces us controlling 64 RGB LEDs individually (192 different diodes!) to only needing to control 32 different variables. These 32 variables correspond to the total number of rows and columns of the matrix. 24 different columns will need to be controlled and only 8 different rows need to be controlled as well.

After figuring out that the LED color that will be used in the piece identification subsystem would be green, we were able to simplify our array even further by removing the columns for the green anode. Any use of the color green in our array could cause incorrect values for our photodiodes and we want to avoid any inaccuracies. This cuts down the total number of columns we need to control now. From 24 columns that need to be controlled, we are now down to 16 columns (8 for red and 8 for blue). In total, we would only be able to use 3 different colors to convey information to the players: red, blue, and purple. To control these columns without using up a significant percentage of the available pins on the microcontroller, we used shift registers to cut down on the number of pins that need to be used. We are lucky enough to have enough GPIO pins on the microcontroller to control the rows for our matrix, so we only need to worry about having enough shift registers to control the 16 columns. With 2 8-bit shift registers, we would have access to enough bits to control these columns.

6.7.1 LED Controller

The controller we decided upon for our design is the SN74HC595N. This 8-bit shift register can both output current and take in current so if the need arises, we can use these shift registers for a common anode RGB LED matrix. These shift registers can also be daisy-chained to other shift registers to create a larger shift register. By daisy-chaining we can use multiple shift registers with the same number of pins it would take to control a single shift register. Since we only need to control two different colors, each shift register will be responsible for a specific color: red or blue. These shift registers will be responsible for providing a positive voltage source to the column of the LED we want to turn on. The figure below shows the two shift registers along with the connections. The first shift register will control all of the columns that are connected to the anodes of the red diode. The second shift register will be responsible for controlling all of the columns that are connected to the anodes of the blue diodes. Daisy-chaining can also be seen with the connection from pin 9 of the first shift register to the input at pin 14 of the second shift register.

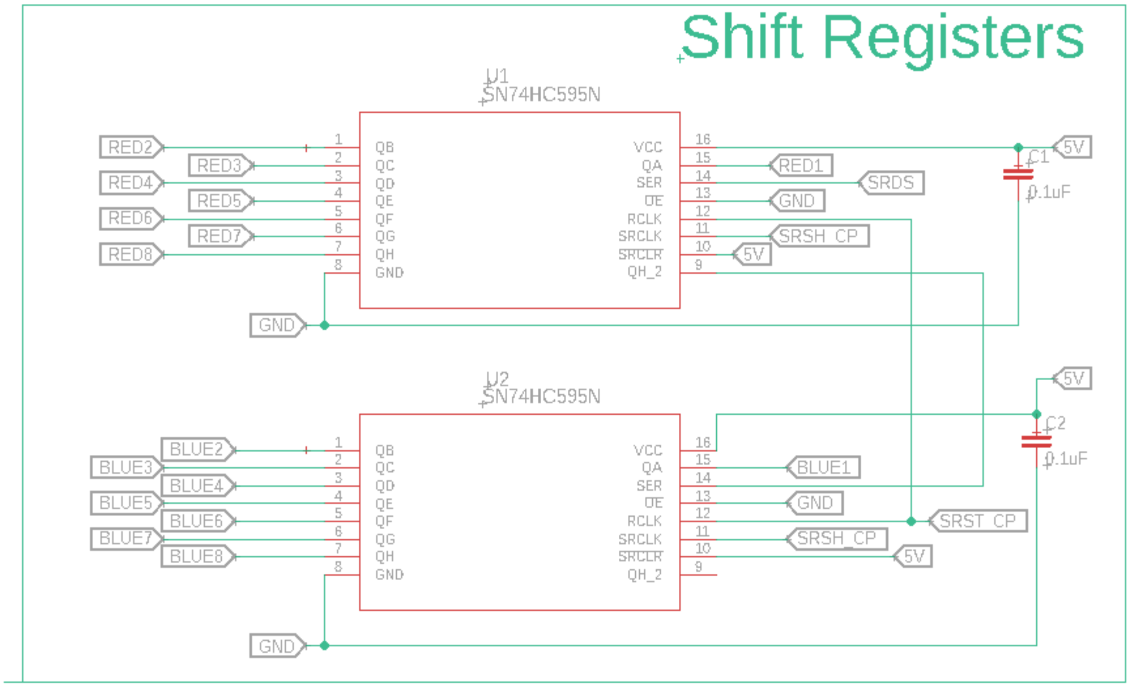


FIGURE 29 SHIFT REGISTERS

5.2.2 LED Source

In order to turn on a select LED color, the anode of the LED must be connected to a "high" voltage source while the cathode is connected to a voltage source that is less in value or simply to a ground. We used 4 shift registers to control all of the columns and rows in our project. The shift register we selected in able to both source and sink current. To turn "on" a row, the shift register pin out would have to change from high to low. Figure 30 shows a 4x4 version of the LED array. Each column and row will be connected to another 4x4 PCB or to the other PCB that will contain the current limiting resistors for the LEDs and the shift registers. We organized each shift register by color/row.

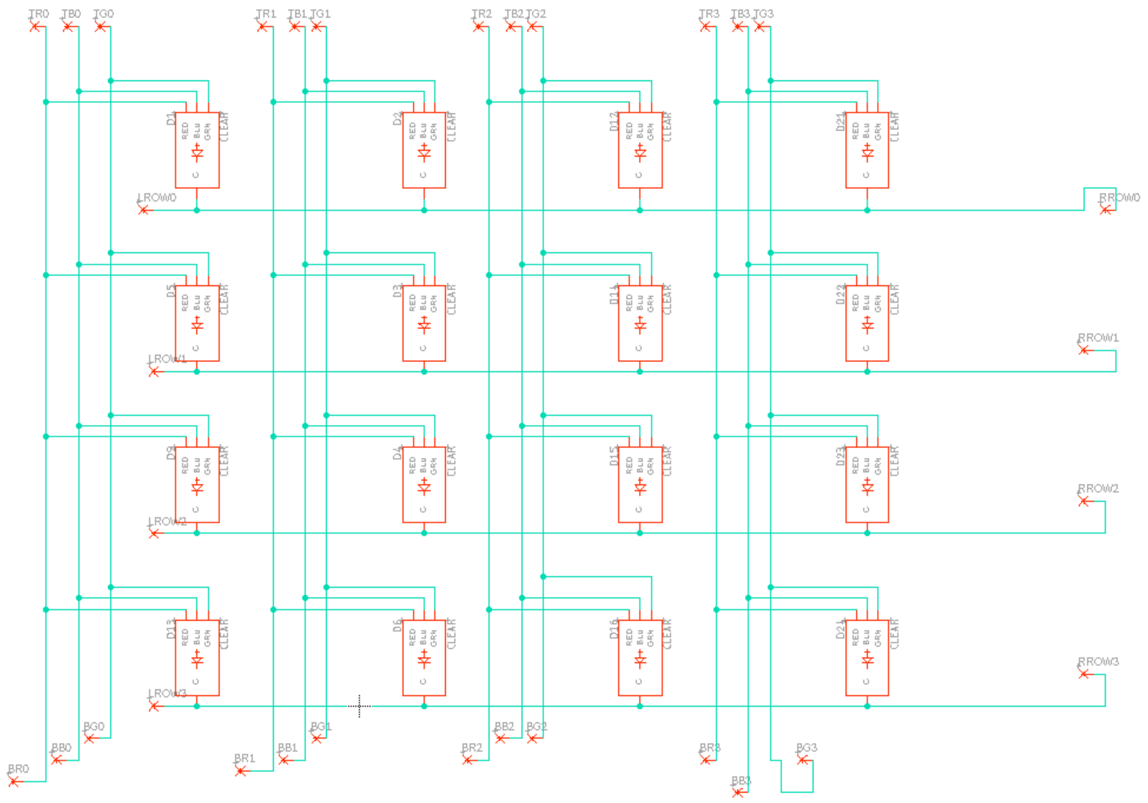


FIGURE 30 4X4 LED ARRAY

6.8 Microcontroller Pin Layout

Below shows the schematic for the Teensy 4.1. Part of the schematic shows the intended purpose of the used pins. The pins labeled in this schematic will be included when designing the PCB for our project. We do have a LCD screen that will be a part of the project but due to the location of where the LCD screen will be, we used wires to connect the microcontroller and the LCD. Pins that have the “SR” in front of the term are for the shift registers while terms with just a “D” in front of it will be for the display. Labels A0 through A7 are for the ADC converter that is built into the microcontroller. Pin 55 on this schematic is the Vin pin for this microcontroller. This pin will be directly connected to the voltage regulator that will be responsible for powering the entire project. We were unable to find an exact schematic, but we at least wanted to label and cover this information.

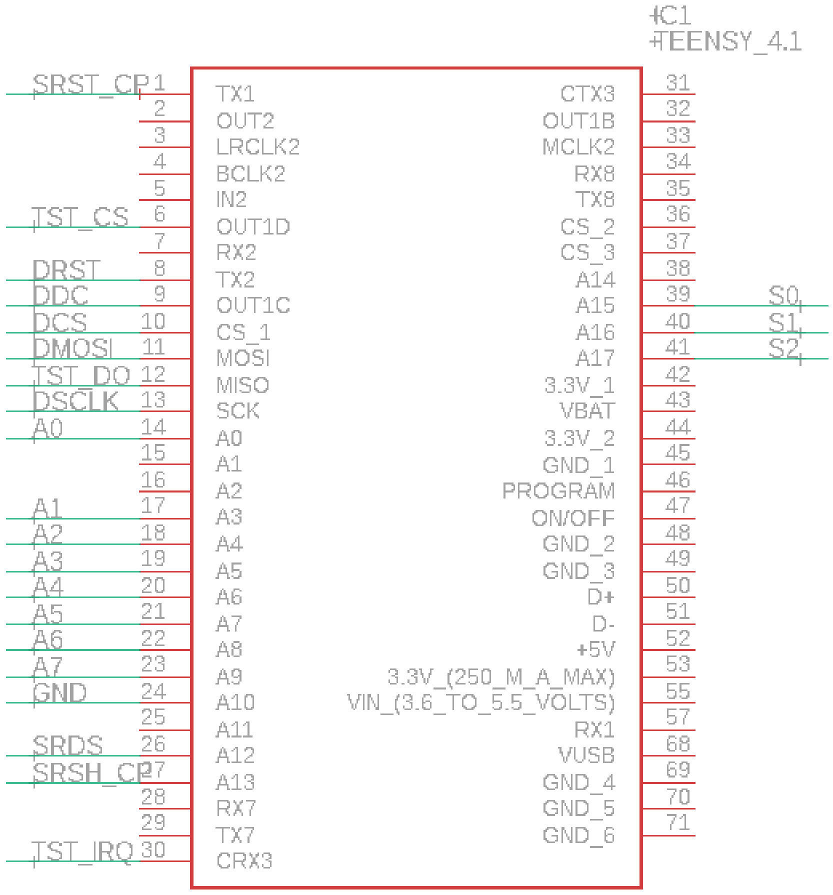


FIGURE 31 TEENSY 4.1 PIN LAYOUT SCHEMATIC

6.9 Overall Schematic of Subsystems

Below shows all of the schematics for all of our subsystems. The subsystems have been outlined below with their corresponding name. Due to the size of the schematic, we split the image into two parts for it to be visible in the report.

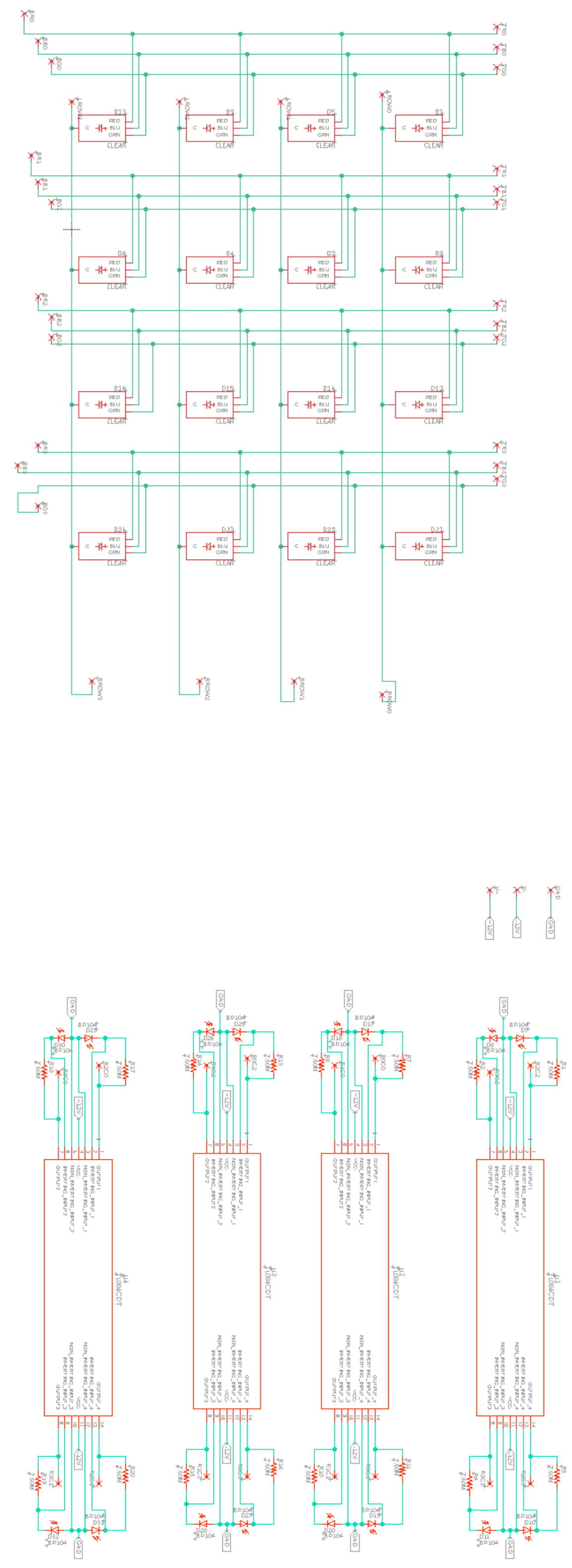


FIGURE 32 OVERALL PROJECT SCHEMATIC (1/2)

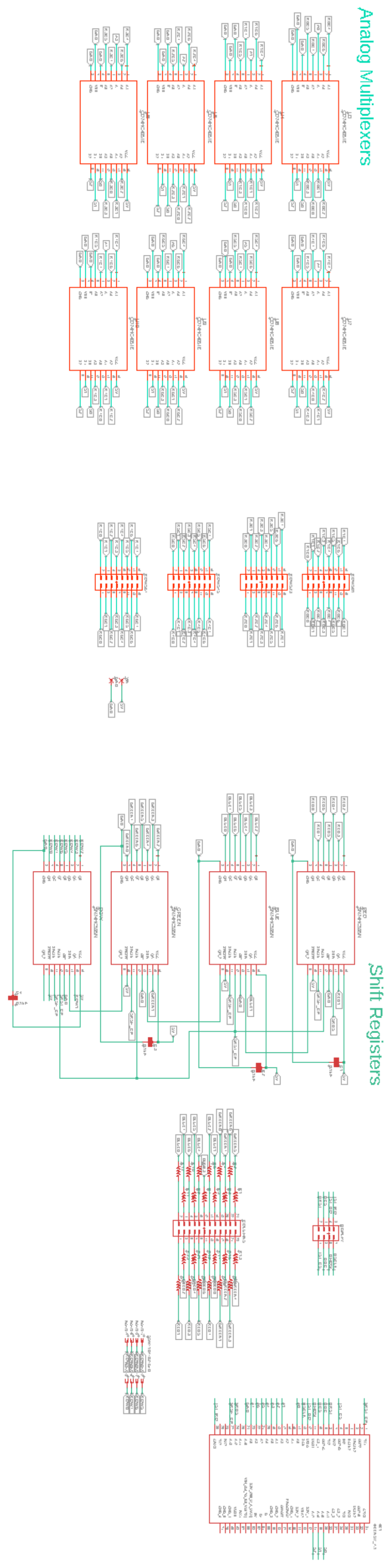


FIGURE 33 OVERALL PROJECT SCHEMATIC (2/2)

7 Software Design

7.1 Hardware Integration

With the large variety of hardware being used from the project, every driver that could help make this project easier on the software side will be used. However, since each system is quite different in the way they communicate/send information, realistically the only device that could need a driver would be the Display. The LED array will take advantage of shift registers and can be implemented with a few simple lines of code for communication. The Photo Diodes will be set in a system of activation and reading an analog signal, so using the ADC built into the microcontroller will be done to process and store values from all the photo diodes in real time. Buttons and potentially a touch screen implementation will allow for the user to have easy interaction with the display. Lastly, the SD card slot provides the ability for non-volatile memory and allow for additional functionalities.

7.1.1 Display Driver

To configure communication with the display, the best way to do so is through a pre-existing library. The challenge with implementing such a library though is to find one specifically that can be configured to work for both the Microcontroller and the Display itself. Through tedious testing and configuring until its functionality was achieved, the library that worked was the “TFT_eSPI” library.

To configure this library to work with the given display, all that had to be done was to go into a file in the library called “User_Setup.h” and comment out the lines that specifically correlated to the ST7796 Driver- which is the driver that was built into the 4.0” display selected. Doing so already configures the library itself to work specifically for that display.

The next, and more complicated part of the setup was to make this library work for the Teensy 4.1. Because Teensy 4.1 is not a common microcontroller, finding the correct configuration took hours of searching through similar configurations and data sheets. The configuration that worked was to set the CS, DC, and RST pins to 10, 9, 8 respectively, and for the MOSI and SCLK pins to be set to 11 and 13 respectively. Selecting these pins correlate to the Teensy’s default SPI I/O, so if these pins end up needing to be used for another device, it can potentially be reconfigured to fit on pins 26, 27, and 38, along with theoretically any pin for the CS, DC, and RST pins.

The library itself allows for a large amount of standard screen drawing, including different shapes, text, and even sprites. Each text can be further configured to even select different fonts and allow for different languages to be displayed as well (one of the demos provided by the library shows the Hiragana Japanese alphabet being displayed). Sprites, depending on their size, can be displayed quickly. However, if the size is too large it may show a flicker as it fills in the sprite line by line (only in cases with screen-sized sprites though). This could affect the design if there was wanting to be some sort of animation through many sprites, it may be limited to a low frame rate due to the slight delay in fully displaying the image. This effect also occurs when using the “fillscreen” command to fill in the entire screen as on color. Rather than it appearing as if the screen instantly fills with one color, it will instead have a diagonal line that fills in behind it quickly drop through the screen- once again mostly negligible but in cases where the background needs to quickly alternate screen fills it may get distorted.

7.1.2 Utilizing the Shift Registers for LED Array

To communicate the LED Array, we take advantage of shift registers to transmit which color of each LED to turn on. It will be divided into a way where there are certain shift registers for certain colors, as then it is easy to omit an entire color, as mentioned previously for our design. Because we don’t want to interfere with the green light used by the fiber optic to scan the bottom of the piece, the two colors for the LED that will be used is blue and red. We can also turn both on simultaneously to produce purple.

For setting up shift register, values will need to be set in the code for the Data Pin, Latch Pin, and Clock Pin. These shift registers can all share the same clock pin though since there is no reason not to have them use one clock and waste the microcontrollers other clocks for the same functionality. There are also ways to connect multiple shift registers together, but for the demo I only set up one shift register. I set these pins to be 11, 12, and 13, respectively, but because of conflicts with the display these pins will have to be changed, likely to 26, 27, and 38, respectively.

For the program to be able to organize and know which squares are which color, we set up a two-dimensional array that is 8x8 and store the respective state per each square. It will best be set up as a char array, so each value in the array can then be set as: “X”, “R”, “B”, or “P” to abbreviate each led value as either not lit (“X”), red (“R”), blue (“B”), or purple (“P”). This array can then be passed to a specialized function that converts the array into data to the shift registers, and hence turning on the specified LEDs.

One feature that will be needed to be implemented as well is the ability to have the LEDs either be blinking – by oscillating between turning it on and off a color – or just on. This can be done a few different ways and can be selected depending on how the rest of the software is designed. One way is just to modify the previously mentioned array to make lowercase values blink and uppercase value stay solid for that LED. For example, if the square e3 was set to “b”, then the LEDs under that board will blink blue until the array is changed. Another way of implementing this is to have the array only display the LEDs at the exact moment, and just have a function that would switch the blinking ones out with the “X” value and back to the solid light after some time. This method would make it very easy to know the LED board state at any given time but require some additional values to be set to remember which ones need to return to a certain light. Ultimately, the first method seems more straightforward to implement, but may depend on the rest of the structure developed.

7.1.3 ADC for Photo Diodes

To be able to process the amount of green light going into the photo diode, the data will need to be communicated as an analog input. The selected microcontroller, Teensy 4.1, has lots of support for this, by both allowing up to 18 pins being used for analog inputs, and having an ADC (Analog to Digital Converter) built into the microcontroller already. Computers are not able to really use any analog signal for any calculations or math, so that signal needs to be converted to a digital signal for the computer to be able to do anything.

The way this ADC will be setup for the software is by taking advantage of the ADC library created specifically for Teensy. This library technically might not be needed, as Teensy only recommends using it for “More advanced use”, but that will be determined once the prototype is developed, and the exact ADC commands are determined. Based upon the current system designed, only 8 analog signals would need to be monitored concurrently, and that can be processed by switching through each one through the ADC to read certain values.

The default resolution that the ADC provides is 10 bits, allowing for the digital output to range from 0 to 1023. It describes that the hardware does allow for a larger range up to 12 bits of resolution, but by default it ignores those two bits because they are not usually usable due to noise. When the piece identification system is developed and tested, ranges within the range of 0 to 1023 will correlate to each piece and be hard coded into an array that given an ADC value will tell the piece that it is. This portion will compare by iterating through the array until it finds the range that suits the given ADC input. When calculating this though, built-in timers on the microcontroller may be used to ensure that the piece detection is reading stable value. If a piece is being lifted off, theoretically it can hit multiple other piece values before reaching the value range for no piece being there, so if the machine just immediately calculates the respective digital value being in a new range, the machine will think a piece was swapped out and given a misread. Using built in clocks can avoid this feature and intentionally add a small delay (less than 1 second) after lifting.

7.1.4 Buttons and Touch Screen

For this project, all user input that isn’t scanned by the piece identification system will be delivered through two buttons or through the touch screen. The buttons will server a variety of purposes, both as the way to navigate the menus and user interfaces, as well as the way to indicate the end of your turn in the game, with the clock switching sides. For the menu navigation, one of the buttons will serve as the “menu” button, and the other will serve as the “select” button. They will be programmed to either switch between menu options (menu button) or select the option that is highlighted (select button). This style of buttons minimizes the number of buttons needed total and follows a simple design that many arcade cabinets / old game consoles would use. During the game itself, the buttons will switch functionality to be the way to switch who’s turn it is. Pressing the button when it is your turn ends it if a valid move was made. This gives the user the ability to make sure the move they want to make is truly the move they wish to make. Having it just respond after placing down a piece may lead to unintentional moves, making a frustrating experience for the player.

The button will be able to simply communicate digitally to any of the microcontroller's pins. When configuring the button, different options may be selected like whether the button reads +Vcc when pushed, or GND when pushed, and the opposite for when it's not. Any of these options only affect the way that it will be coded, so it is easily modifiable.

Another way to communicate user input is through the touch screen feature built on the display. Through some initial testing, the touch screen is responsive and accurate, meaning it can carry some roles for selecting options. However, the biggest issue with using the touch screen is that it would take away another SPI port. The microcontroller does have enough ports to support this device, the visual display portion of the display, and the shift registers, but with conflicts such as the analog ports, it is unknown if these ports will be fully available to the use. Based on that, all menu navigation and product usage can be completed with the buttons, but if the configuration for the microcontroller is set up and the touch screen is confirmed to be a part of the board, then certain features could be picked up by the touch screen. For example, any menu input during a match would have to be done by touch screen, since the buttons are being used to switch who's turn it is. This can be avoided by adding some additional functionality with the buttons like if both are pressed simultaneously, but that may cause unwanted delays in response and effect how interactive the chess board feels.

7.1.5 SD Card

Attached to one side of the microcontroller is an SD Card, which will allow for this device to have non-volatile memory. Out of the list of current features intended to being implemented to this project, all memory can be non-volatile, however if the stretch goal of saving and loading a chess game ends up happening, this will be the method to store and retrieve that data.

This SD card will allow also for large data storage and could even be permitted to store a history of games played. Teensy recommends using the Arduino SD library to gain access to the SD card. Access to this data will be slower than the other memory there, but still accessible enough to store and retrieve any data that would be needed to stay after turning the device off and back on. A very small SD card, with only a few Megabytes per say, would be able to implement these additional features for a low cost. The intent of showing this aspect of the software design is to allow for easy addition to the project if that ends up becoming a main feature- it does not require redesigning any existing components, or buying any new devices besides a cheap SD Card.

7.2 IDE for Teensy

Developing for the microcontroller selected, Teensy 4.1, required a selection if which IDE (Integrated Development Environment) to use. With the provided documentation for Teensy, there are 5 recommendations they provide for an IDE.

Visual Micro is one recommendation they provide, which works to combine with Microsoft Visual Studio code to be able to program any Arduino compatible boards. Arduino’s development environments are very similar to those for Teensy, so this would work for this microcontroller as well. The biggest restriction for this IDE is that it is not cross platform, and only is available on Windows. This will suffice for most of the development being done, however some of the development would be done easier if cross-compatibility was allowed.

Another recommendation provided by the Teensy 4.1 documentation is PlatformIO. This IDE is a cross platform one, that supports Windows, Macintosh, and even Linux. This will allow development to be done by many different devices, making it simpler to work on this project. Something that makes this IDE unique is that it hosts many features, making a versatile tool for microcontroller development. However, with that in mind, the IDE is more challenging to use, and has a large learning curve to it as it’s a bulkier program.

CircuitPython is one more option that is provided by the microcontroller. This allows for python to be integrated into the development of the project. Python will not work well with what the rest of the project has been setup around, so this option simply doesn’t make sense. In addition, CircuitPython does not support all Teensy 4.1’s hardware, meaning they may be conflicts later that will not be resolvable on this IDE.

One method for developing on the software is to just use a Makefile on CLI- essentially making the IDE just whatever text editor you use to modify that file. This option is the simplest to use by far, but has no additional features that IDEs would provide, and would make adding libraries in particular a tedious process. Perhaps if a user has high expertise on programming for microcontroller this option would be easy to quickly start developing, but it may be restrictive to use for this large project.

Lastly, the IDE which will end up being used for this project is the main recommendation of programming environment for Teensy, which is the Arduino IDE with Teensyduino. As mentioned before, programming for Arduino is like Teensy, so just modifying Arduino’s native IDE makes for a great development environment that is easy to use, well documented and maintained, and capable of supporting this project. It also is cross-platform, for Windows, Mac, and Linux, making development for this project very accessible. In addition, the Arduino library manager is especially effective for managing all the libraries being used and flashed for the project and given the complexity of the hardware stack those libraries will be necessary. This is the IDE that will be used for development.

7.3 Implementation of the Chess Engine

7.3.1 Importing TSCP

This project will take advantage of a chess engine to simplify the software portion of this project. Rather than painstakingly programming all the rules and logic of chess, using TSCP's user friendly code will give a good framework for all the chess code in this project. This strategy should greatly reduce the amount of code being written by us and allow for more time to be spent on integrating the other hardware and potentially achieving some of the software related stretch features.

When implementing this engine, there are two ways to go around with user input. Currently, in its unmodified state, the code uses "scanf" statements just to await user input and process it, via command line. Since the user will be making actions on the board that translate to these moves, there is no reason why the user should have to type out their move every single time they want to make one. The first solution to this is to manually go through the code and change out those "scanf" statements to some other input function that takes in that processed input. This method wouldn't take too long to achieve but will result in heavily modifying the source material. The other method could be to maintain the original code how it is and use some sort of interface/multi-threading to communicate by those "scanf" statements to the engine. The latter would be desired, as ultimately can result in not modifying the original code at all in this case, but the difficulties with that would be both that setting up the environment may be very challenging and may restrict getting/modifying the data needed from the chess engine. Ultimately the first method will likely be the one used, but research will continue to see if option two is possible.

7.3.2 Communicating and Displaying from TSCP

Because TSCP is already interfaced to display text-based graphics for seeing the board, converting that data to the LED array flashing is one of the greatest challenges when implementing this engine. Fortunately, TSCP is well documented and organized, so those modifications are mostly straightforward. One of the included C files in the TSCP project is data.c source file. This contains all the information on how each piece is stored for both color and type of piece. Reading from those arrays will be critical to how the current board state is remembered by the microcontroller.

In addition to the data.c source file, there is the provided board.c source file that contains lots of helpful functions to help communicate from the rest of the engine. One function detects if a square is being attacked by a piece. This can modify so that it becomes the vision when a piece is lifted, and it shows the valid moves available. Using some of the other functions available as well we can implement detection if it's a capture (turning the square purple instead), and if moves are legal (moving the board state into an illegal move recovery). Overall, with all these specialized c files, all the functionality of the chess engine is neatly divided up and well documented, so accessing any specific display or general data will be straightforward.

7.3.3 Best Move Calculation from TSCP

Chess engines are traditionally only used to find the best move, and to create powerful computer opponents. However, this project is using a chess engine in a more creative way to prioritize the speed and efficiency of implementing the rules of chess by using a well-documented chess engine like TSCP. One of the first extensions of this project that may be achieved after implementing the primary features will be the best move calculation this engine provides.

Best move calculation has three main uses. First, it allows for the user to understand and receive hints on which move is the best in each position. This could help beginners greatly, as they may get in situations where they simply don't know what to do and could use an expert's opinion. This will make the game very forgiving for beginners, as it shows a potential way out of every bad position (unless all hope is lost…). The second main use this provides is the ability to evaluate the board, because best move calculation requires the board position to be evaluated and to maximize score for the user with the next move. This evaluation can allow for the game to have a running score between the players and give them back feedback depending on the size of the errors. A small error would only affect the evaluation minimally, while a massive blunder may result in a greater shift. This feature can even be enabled on or off, to give users the option to see who is really winning the game so far. The last of the three main uses is for best move calculation to create powerful computer opponents. To allow for beginners to go into a single player mode to simply be able to play the game more, a match against a computer that always makes what it thinks is the best move can be set up. A lot of additional coding would have to come along to make that feature work, since there would have to be special messages on the display to guide the user to pick up and move certain pieces- the machine can't move it itself. This feature has the greatest reward though, since it reduces the amount of people required to play a game by half.

7.4 Menus on the Display

7.4.1 Menus and Their Navigation

Utilizing the display on the side of the board, there will be a series of menus designed to assist the user in beginning a game, configuring options for that game, and informing the player during the game. The menus will be navigated using the two buttons, as specified earlier, and provide all the input for the graphical display. Utilizing the touch screen may end up affecting the design to take advantage of that feature, but currently this implementation works with only two buttons.

The first thing that will appear on the display when turning on the entire project will be a title page. This title page will display the name of the project, as well as the details about which senior design group this is and for which semesters. This will stay on the screen for about 3 to 5 seconds, and then switch over to the main menu. The main menu will take advantage of the two-button input to navigate and begin the game. Using the menu button, the little triangle cursor will switch between the different options available per the screen. When the desired option is being pointed at by the triangle, the select button will be used select that option. Selecting the new game option will begin the game into the set-up board phase, and selecting options will open an options menu.

The options menu will provide a few different options that affect the gameplay. The timer option indicates how much time each player will have when they play the game. In the rules of the game, if a player runs out of time on their clock, the other player is deemed the winner of the match. The timer will have the option to select time controls for each player from 1 min, 3 min, 5 min, 10 min, 15 min, 30 min, 60 min, and no time, which will turn off the clock and allow each player to play with any duration of moves. The next option on the list is the tool-tips option, which when enabled to be ON, allows for information to be displayed about each piece when they are lifted. Because this board is designed to teach beginners, this option will be selected ON by default and provide the beginner with helpful tips that will ease them into the game better. The next option is an option that by default will allow for time on the clock to still pass when an illegal move is made. Because this board pops up another menu when an illegal move is made, the option is allowed for users to have their game paused while the board state is being recovered. The last option is to highlight moves when lifting a piece. This will be enabled by default, as turning it off would make much less LEDs turn on, only turning on for illegal move correction. This may be distracting for players, so this is an option that will be implemented to accommodate. The last thing that can be selected is just the “back” option, that will return you back to the main menu.

When beginning a game, the setup board screen will display, unless the board happens to already be setup correctly. The way it will instruct the players how to setup the board is by displaying an image of one of the pieces and then highlighting all the squares those specific pieces go. When all those pieces are correctly setup on the board, then it will switch to the next piece in the order and repeat the process until every piece is set. The order this will follow is Pawns, Rooks, Knights, Bishops, Queens, and then Kings. It is important that if one of the pieces that is set ends up being taken off after the fact, it needs to detect that and go back in the order to make sure the entire board is set up perfectly.

Once the board is fully set up, the display will begin displaying information for one player at a time, or for both at the same time by displaying the graphics vertically instead. The display will only display the time for each player (If there is any time), an arrow indicating who’s turn it is, and a pause button. When the game begins (after the white player’s first move), then the time for the next player will begin decreasing. Turns will switched by the respective player pressing their button (formally used as the Menu and Select buttons). The arrow will also switch to indicate who’s turn it is. The text may also highlight to emphasize who needs to be making a move. This process can be paused at any point though, by pressing both buttons at the same time, pausing both timers. The game can be un-paused by the same mechanism. At any point, if an illegal move is made, the board will switch the illegal move menu, with text being able to be read by the player who made the illegal move. It will contain instructions on how to fix the illegal move, as well as the LED array lighting up what square a piece needs to be. The timer may or may not be paused, depending on the option selected in the options menu, while in this menu. Once the board state is restored to a legal position, it will return to clock screen.

Finally, when the game ends, a game over screen appears with the results announced. It will either say “White Wins!”, “Black Wins!”, or “Stalemate!”. At this point, the board will wait until the user presses any button to then go back to the title screen, restarting the program. This minimal menu program will provide the user with lots of control and interaction with the chess board.

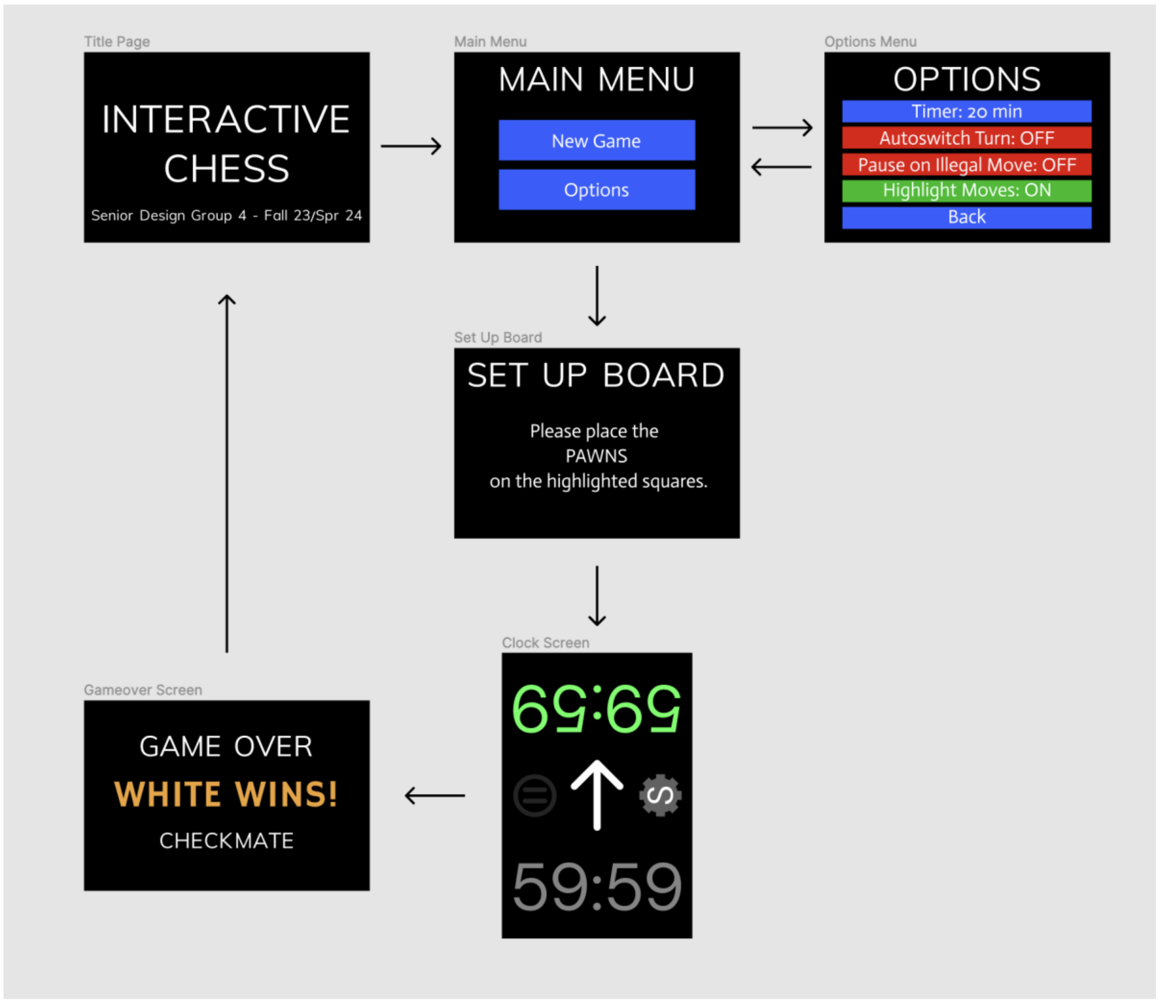


FIGURE 34 FIGMA MODEL OF ALL MENU SCREENS

7.4.2 Displaying Graphics for the Menu

Based on the Figma model of the menu screens provided above, there are only a handful of different elements that make those screens. Using the TFT_eSPI library for graphics allows to use specific commands to draw those to the screen.

Text is the most repeated element across all menus. In the Figma model, I used fonts “Awesome 5 Brands”, “Actor”, “DM Sans”, and “Hammersmith One” at various sizes to display the information a varied and aesthetically pleasing way. Most of these fonts are exclusive to the Figma website, but finding downloads of these fonts would allow me to upload them into the TFT_eSPI library and use them on the project. Either that will be done to achieve this look, or similar fonts will be selected based on what is already available on that library. The display supports the feature of being able to display any color, so perhaps this draft of the menus could be altered to take more advantage of that and have more colorful text. The only non-white text that will need to be programmed is the red text associated with an illegal move, and the gold test for when game over screen is displayed. In addition to this, the text will have to face different ways at different times. For the menus at the start before the game begins and after the game is over, all text will be horizontal with the display. However, when the game begins with the clock screen and illegal move screen, that text will be aligning vertically instead. This can be achieved with the command “setRotation”, which will just change the orientation of how things are drawn to screen.

Besides text, a few shapes exist that can be drawn on by the display library. The first instance of this is the selection icon on both the main menu and the options menu. This is simply just a triangle, with one point pointing towards the text. This can easily be done be commands set for drawing triangles. The next shape would be the illegal move ‘X’. This shape can just be drawn in as well by taking two lines and having them intercept. This shape would also be red, so the color attribute of these shapes would have to be modified depending on the circumstance.

Lastly, the remaining graphics on the screen will be sprites. As seen in both the setup board and illegal move screens, a picture of a specific chess piece will need to be loaded. The graphics driver supports loading sprites, but sprites do take up a large amount of space in memory so they will need to be minimized in usage. Each of the 6 chess pieces will need an image like that in the Figma, so be able to describe information for each of those pieces. The other sprites that will be included is the pause sign and turn-indicating arrow on the clock screen. Both could be drawn in as a collection of shapes by the graphics library, but it would be much more extensive than a simple triangle or ‘X’ symbol, so it is far simpler just to save those sprites. This project doesn’t have many sprites for the display to have in the first place, so adding a few more sprites to simplify the code will not much cause harm to the amount of storage available.

8 System Fabrication/ Prototype Construction

This section will cover the physical design of our system. These parts do not necessarily fall into the other categories. This chapter will cover the PCB design, and the dimensions of the overall project once assembled.

8.1 PCB Design

So, we used two different PCB designs for this project: one for the materials that go under the chess squares and the other for the controllers.

To cut down on costs, we split the chess PCB into four quadrants. By placing through holes for the rows and columns, we are able to connect these smaller boards together to assemble the full 8x8. Figure 34 shows the PCB layout for this board. The placement of the photodiode footprint was also considered so each photodiode sits in the dead center of the 1.75” chess square. We also had to place through holes for the output of each amplifier so we could run wires from the output to a pin on the multiplexers.

Another thing to note is that the footprint we found for our selected LEDs, was glitched in the CAD software we were using. The library itself was working correctly but with the DRC rules, overlaps of the pads was present. So, we had to alter the sizing of the pads to stop the overlap. If we did a second iteration of this board, we would have changed the DRC even more to give more spacing between pads because it made soldering the LEDs difficult.

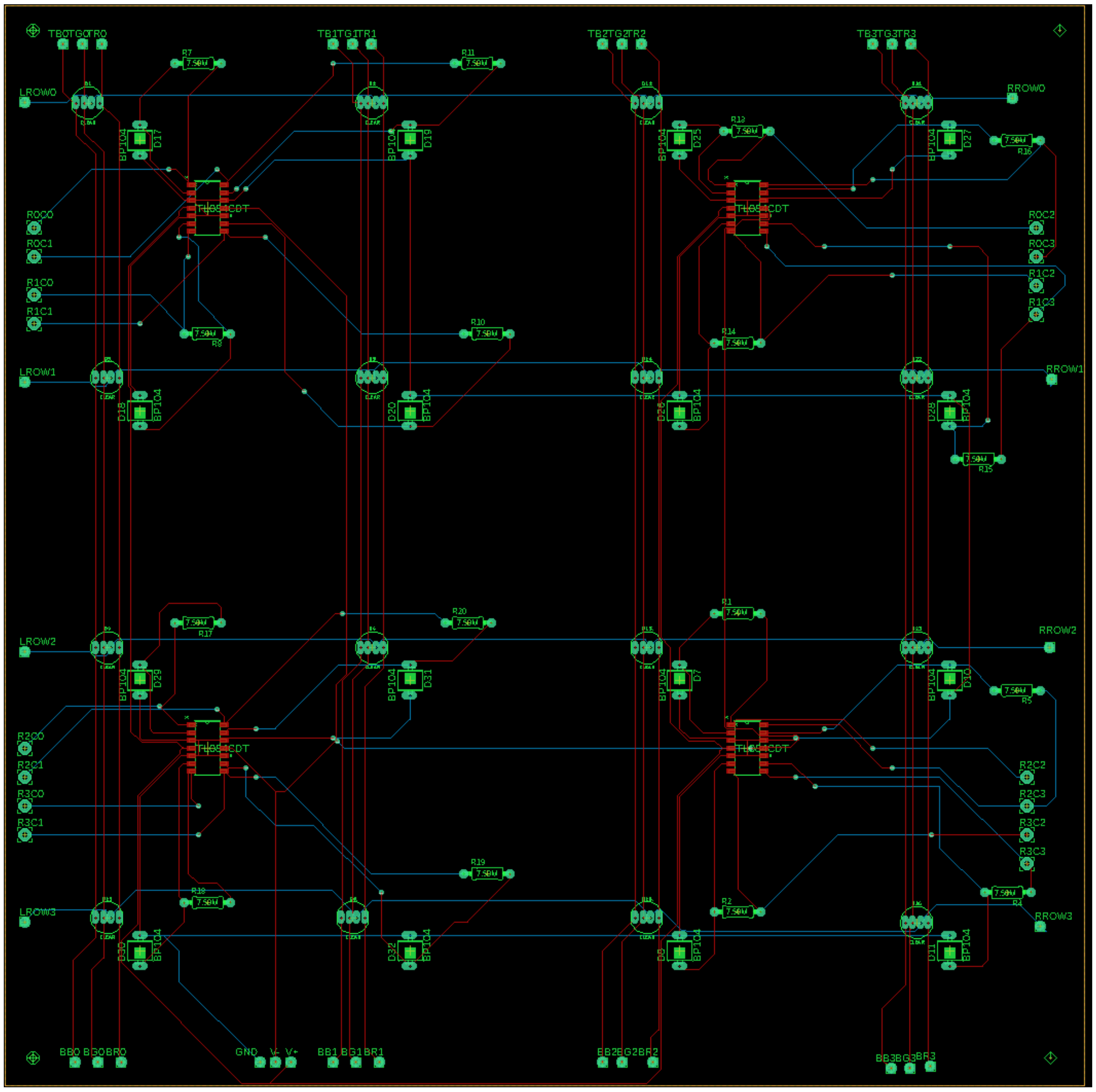


FIGURE 35 CHESSBOARD PCB

Figure 35 shows one fully assembled quadrant of the 4x4 PCB.

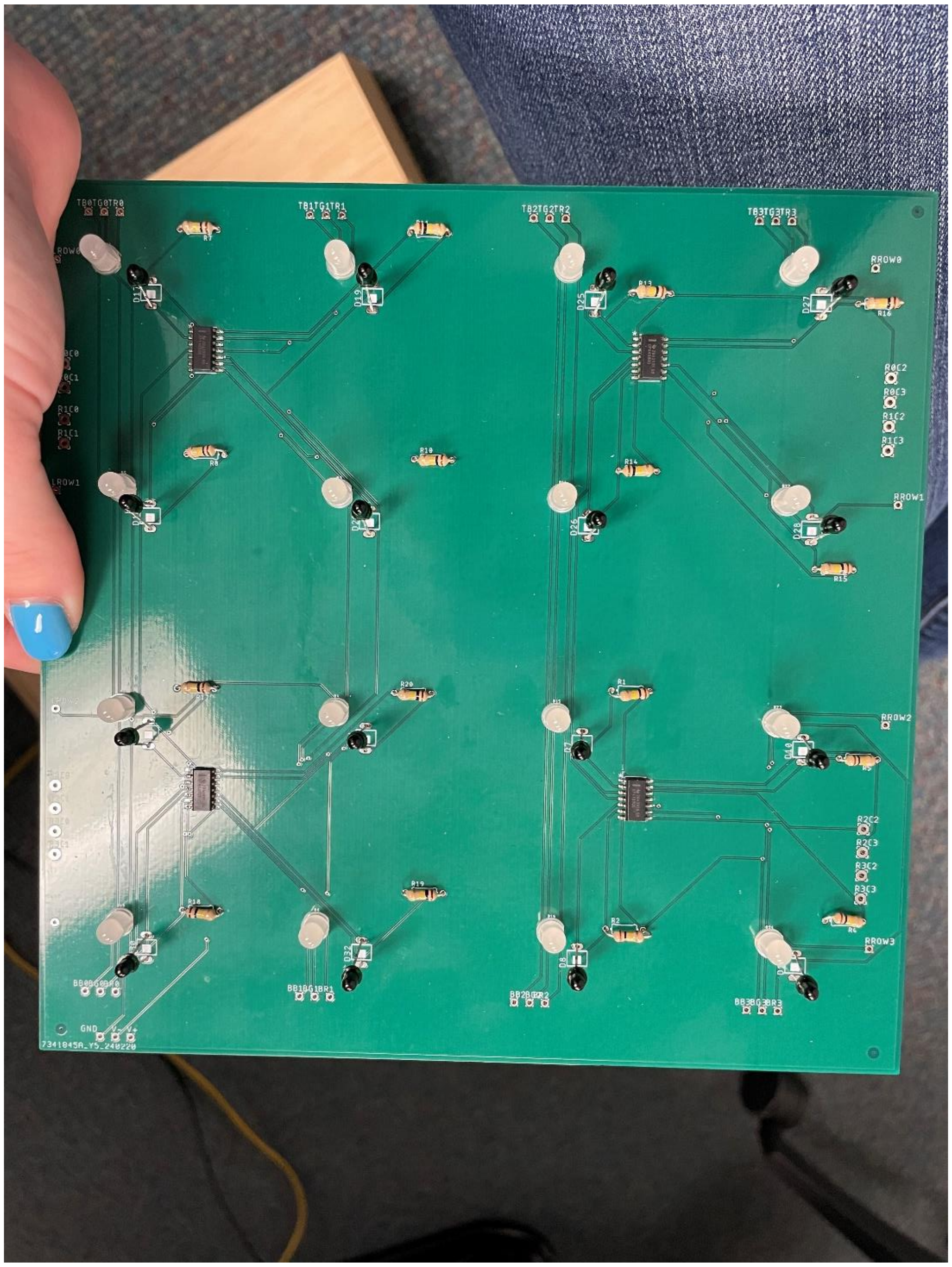


FIGURE 36 FULLY ASSEMBLED CHESSBOARD PCB

The other PCB in our project contained the resistors for our LEDs, the shift registers, the multiplexers, and our MCU. Figure 37 shows the layout of this second PCB. We used headers to hold all of the wires for this project. Counting all of the wires from the transimpedance amplifiers, LED array, and the display screen, over 100 wires had to be managed for this project. Figure 38 shows the fully assembled PCB for our project minus the MCU.

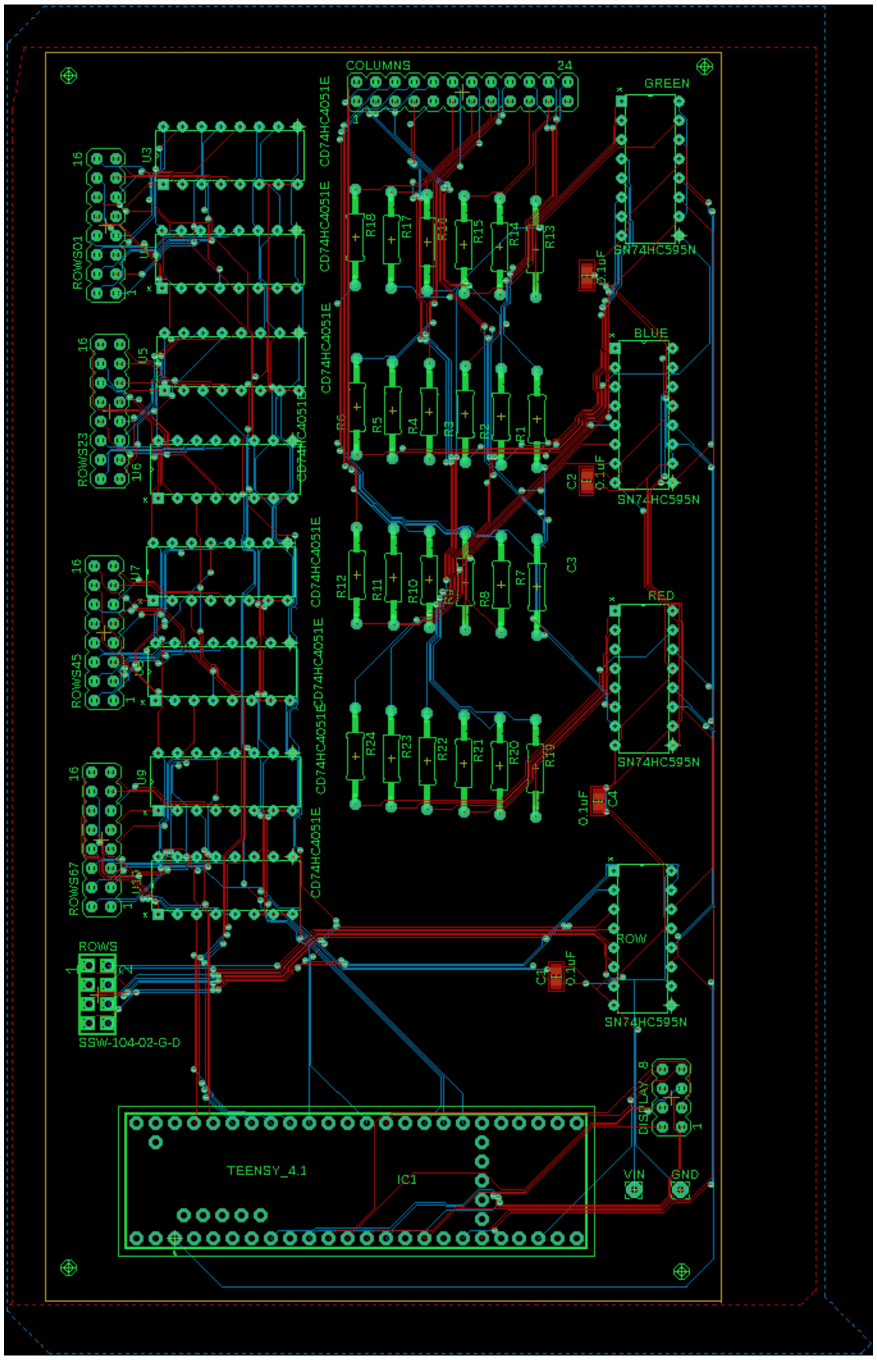


FIGURE 37 SECOND PCB LAYOUT

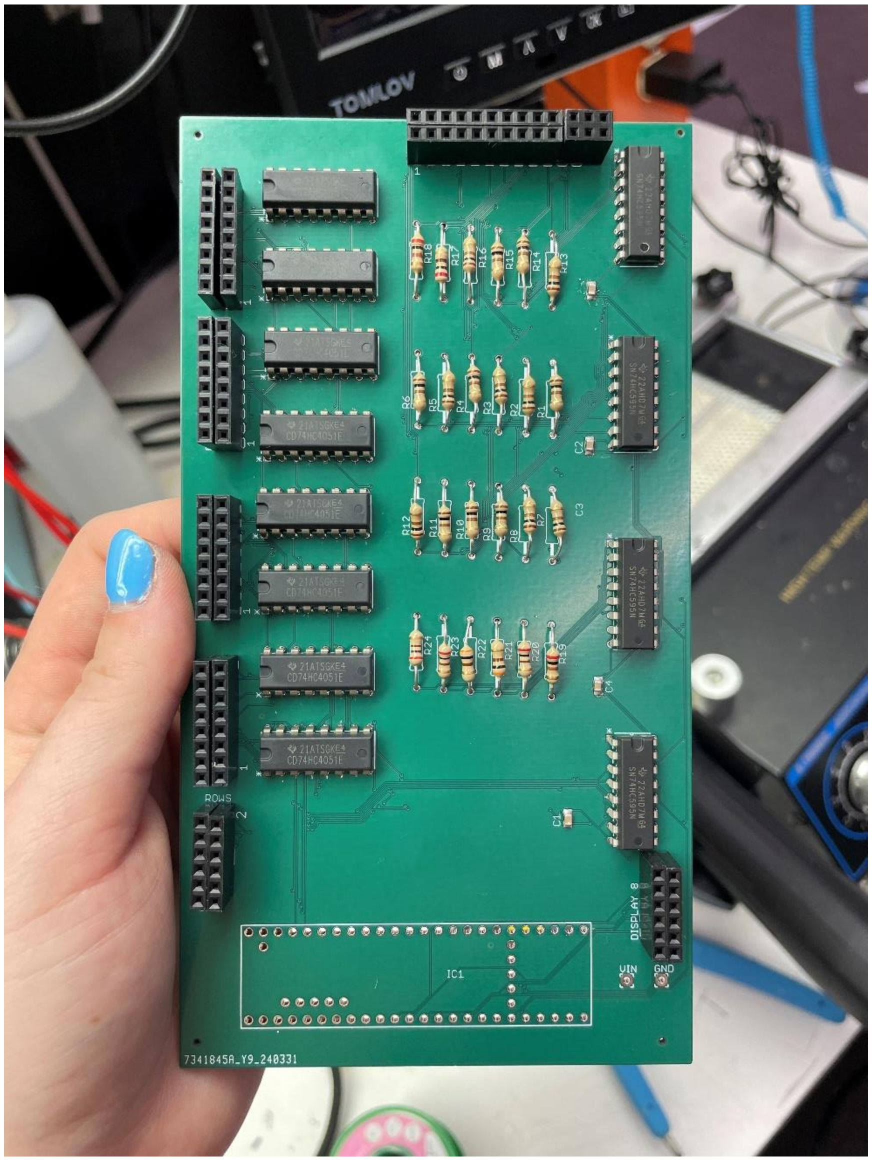


FIGURE 38 FULLY ASSEMBLED SECOND PCB

Then to show off all of the wires needed for this project, figure 39 shows the backside of the fully assembled 8x8 PCB. All of the blue wires are for the transimpedance amplifiers, the red wires are for the columns of the LED array, and the black wires are for the rows of the LED array. Some wires were also used to connect all of the 4 boards together.

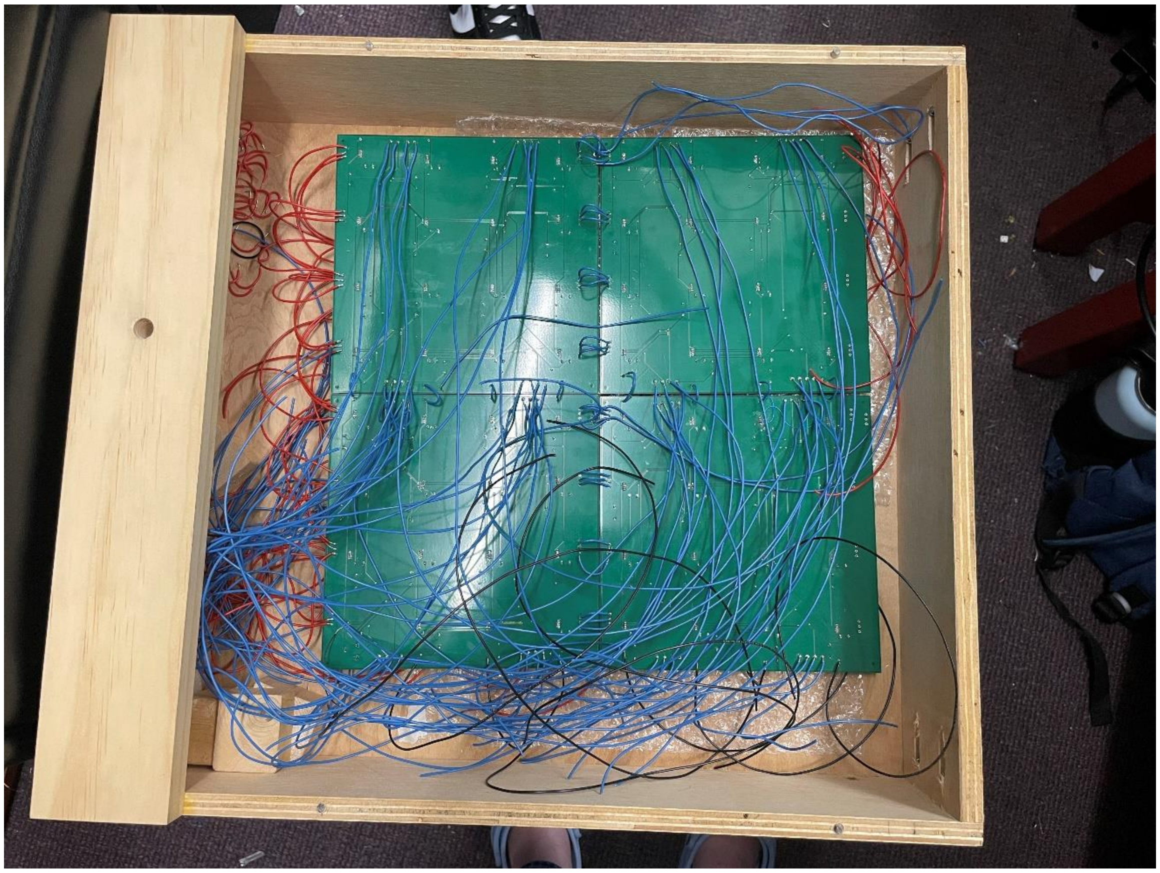


FIGURE 39 BACKSIDE OF 8X8 CHESSBOARD PCB

8.2 Chessboard Box Schematic

The project's housing is a very important piece of the design. It is responsible for containing and protecting each of our components as well as giving it an actual shape and allow for it to be aesthetically pleasing and user-friendly experience. The main body of the outside of the box will be made from wood, it provides decent hardness and resilience to the casing and is easy to work with, all the while giving the product an elegant and professional look.

The overall measurements for the exterior hull of the project are listed in the table below, we are expecting to be able to fit all the components into a relatively small frame when compared to what we had initially envisioned, which is great for transportation and use convenience. We are also hoping that the reduced size will help us not only achieve, but outdo the weight limits we had first set when we first imagined the project.

TABLE 20 OVERALL DIMENSIONS

	Measurements
Total Dimensions	20” L x 18” W x 4” H
Chessboard Playable Area	14” L x 14” W
Chessboard Tile Area	1.75” L x 1.75” W

The following schematic further details the dimensions of the box’s exterior and helps illustrate and visualize the look of the finished product.

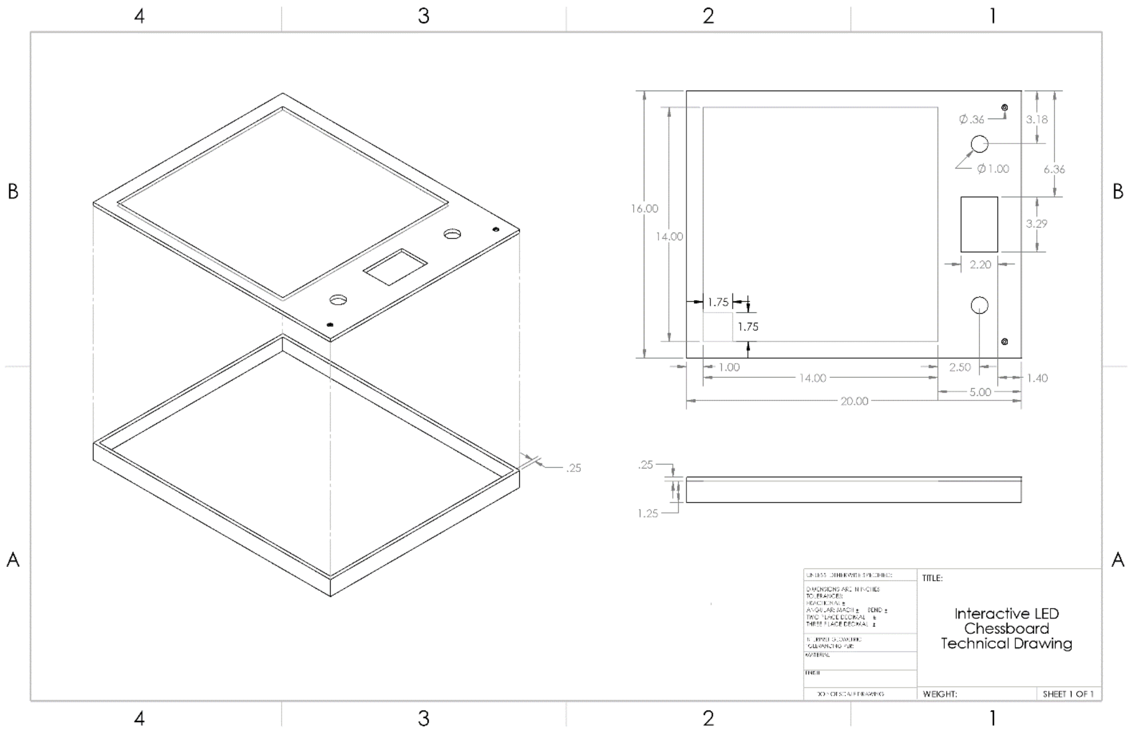


FIGURE 40 BOX EXTERIOR

We wanted to have a streamlined design aiming to facilitate both assembly and usage of the product. The top of the chessboard will be able to be flipped open and removed from the way to allow for easy maintenance and troubleshooting of the systems within. The top is comprised of a single piece of wood, which will be affixed to the remainder of the box through screws, and it contains holes for the buttons and LCD screen, as well as the space we needed for our actual chessboard. Additionally, the board will also contain one other opening on its side for a charging port to allow for an easier way to recharge our battery.

The playable section of this chessboard will be comprised of 64 smaller squares made from epoxy resin with small wooden separators in between them that will prevent the light from each square to bleed into the adjacent squares. This grid will be then laminated with a thin sheet of clear resin to be made into one solid square piece that will be able to slot into place and affix to the top surface of our box.

Wishing to maintain a close resemblance to traditional chessboards, we used different pigments when creating our epoxy mix to represent both colors of the board. This is so that, even when not illuminated, the light and dark squares will be clearly distinguishable to the players, providing a better playing experience.

8.2.1 – Chess Board Internal Housing

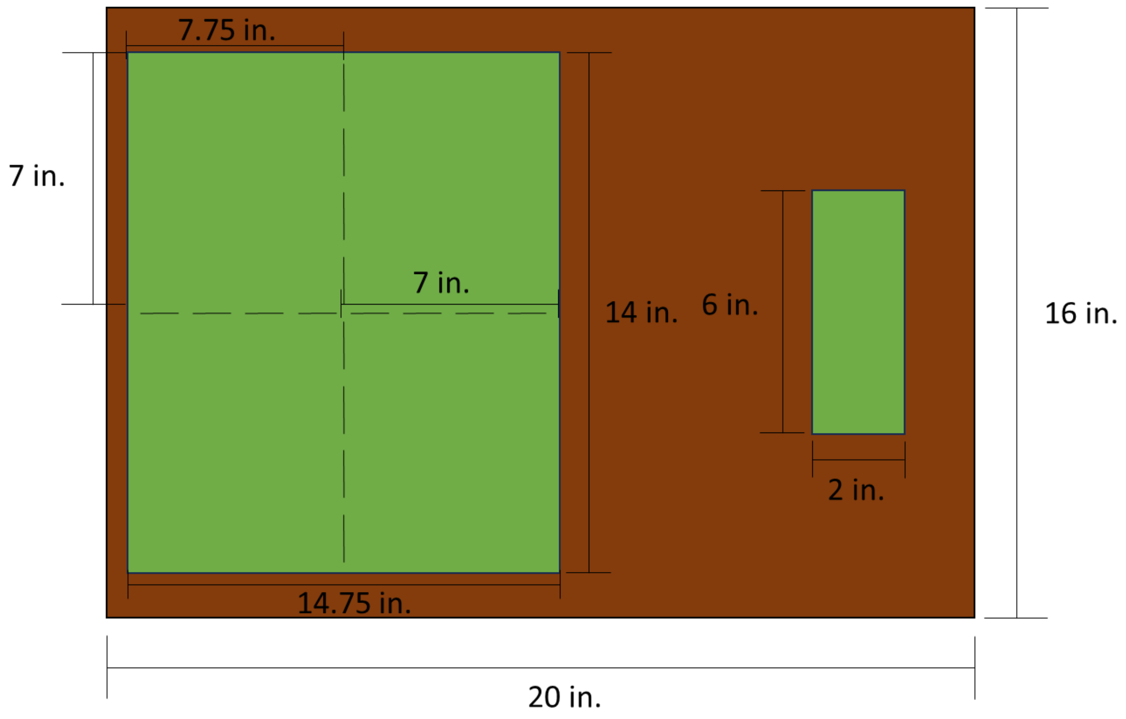


FIGURE 41 SCHEMATIC OF THE PCB LAYER IN OUR PROJECT

For Figure 34 as shown above, right underneath the honeycomb structure will be the PCBs used in this project. Our project requires the use of multiple PCBs that each have their own specific purpose and function. We plan on building a PCB layer that will sit right under our chessboard that connects all the LEDs used to light up the squares on the board and also connect the photodiodes so they can communicate with our microcontroller to properly and accurately determine and identify each piece on the board. Since our chessboard will be 14 inches long by 14 inches wide, we need to combine multiple PCBs together to become one giant PCB. We couldn’t just make one giant PCB as that would have been very expensive to buy along with providing less room for mistakes or changes we may need to make later on. To accomplish the look of one giant PCB, we needed to split the chessboard into four quadrants and have four PCBs put together. The two PCBs on the left half of the chessboard will be slightly different in dimension than the right side of the chessboard as we need to make them bigger to account for the green LEDs that will provide the light going into the fiber optic cables. The PCBs on the left side will be 7.75 inches wide by 7 inches long. The two PCBs that make up the right half of the chessboard will be 7 inches wide by 7 inches long. We then combined all four PCBs together to create one giant PCB layer that will be lined up and positioned with our chessboard. We also needed one small PCB on the right side of our project to connect the microcontroller, voltage regulator, and display. This PCB on the right side of our project will provide communication from the main PCB layer underneath the chessboard to the microcontroller and act as the central hub from which signals will go into and transmit from to operate the chessboard. Our current estimations for the dimensions of this PCB are that it will be 2 inches wide by 6 inches long. This is only a current estimation as we aren't quite sure yet how big of a PCB we need to make it. Once we start assembling our project, then we were able to determine the exact dimensions.

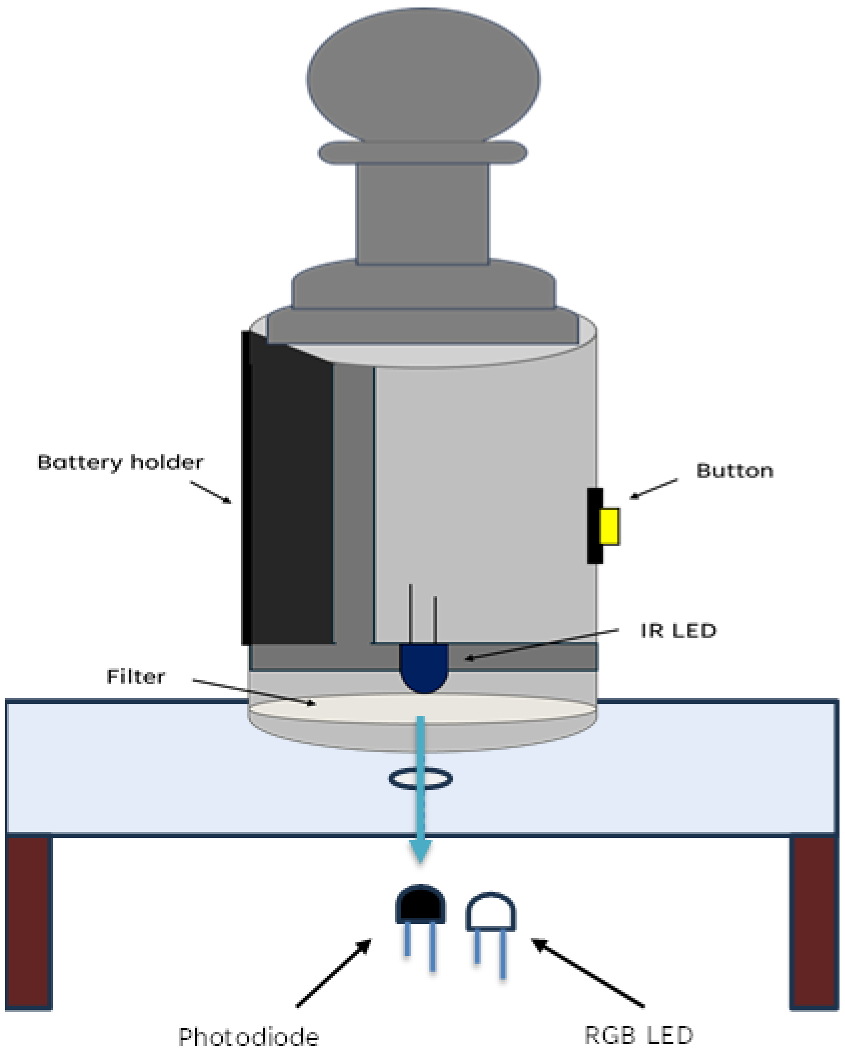


FIGURE 42 SCHEMATIC REPRESENTING A SIDE VIEW OF A SQUARE IN THE CHESSBOARD

A side view of our project can help visualize how exactly our project will work underneath the chessboard. As shown in Figure 35 above, our design is the same for each square on the board. The light from the IR LED will be positioned directly above the opening in the square so that the light coming through down towards the photodiode would be at max intensity and as centered as possible. Then once the light is received by the photodiode, then the RGB LED next to it would then light up depending on what piece was detected and show the players where a piece can move to.

Currently, the entire project be 4 inches tall. Our top layer of the project will be 0.25 inches thick. For each square, we used an acrylic sheet for the top layer in which the game would be played on. We frosted acrylic top slightly to signify white and black squares normally on a chessboard. There will be a very small amount of space from the top of the honeycomb structure to the top layer. We preferably didn't want the top layer of the project to be resting upon the honeycomb structure. The max height of the photodiodes and LEDs in each square will be 1 inch. This height for these components varies depending on the soldering to the PCB. We tried to keep the height of each component the same for each square, but there was some variance from square to square. The photodiode will be slightly taller than the LED as we don't want a large distance for the light to travel from the bottom of the piece to the top of the photodiode. We want to minimize the distance the light has to travel as the greater the distance, the intensity of the light will decrease. The thickness of the PCB will be very minimal and is practically negligible. We are currently leaving 2.5 inches of space underneath the PCB layer to give us space to adjust things and position any other components or wires or anything else not in the honeycomb structure.

9 System Testing

This section is going to deal with the overall integration of our individual work into one unique product. Here we focus on the planning and design of the finished chessboard, everything from its internal systems to its external appearance is outlined here. Measurements and data in this section are prone to changing as further testing is realized, however, we consider that it is important to establish a baseline to work towards during Senior Design II to create a good starting point. It allows us to be able to refer back to the information in this section when there are any doubts on the next steps to take.

9.1 Hardware and Software testing

In this chapter, we describe the results of our preliminary testing. The theoretical foundation laid out in the chapters before this one is put to practice so we can learn and improve upon previously determined concepts. Prototyping serves as a dynamic tool, creating a tangible representation of the envisioned system, and providing valuable insights into functionality, usability, and performance. These insights will be listed in the section that follows.

9.1.1 – Optical testing over time

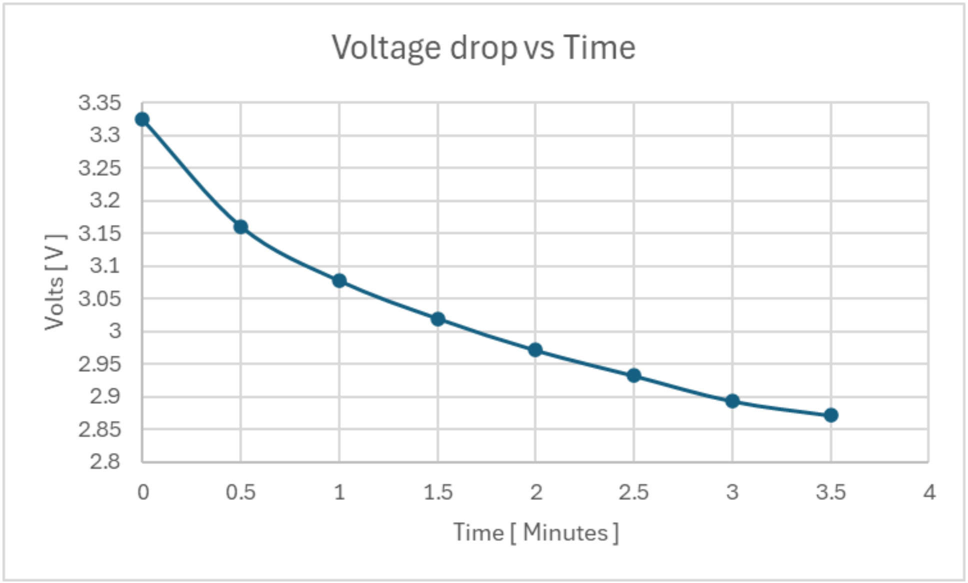


FIGURE 43 VOLTAGE DROP VS TIME

For this test we tested the voltage drop experienced by the photodiode over a period of 3.5 hours. We achieved this by building the photodiode circuit that will be under each square. We build this circuit using a breadboard and not the actual PCB. In this circuit we used a resistor value of 20k ohms because at this resistance value we could see the voltage dropping also since it was close to 3.2 volts. It is ideal to test around 3.2 volts since the microcontroller can't handle anything over 3.3 volts. The next thing we did was to place a small 2 by 2 of our chess bord that only has 4 squares over the photodiode. The hole of the square we would do our testing on must be direct over the photodiode with the photodiode being centered under the hole. We then built the bottom half of our cylindrical base with new batteries in the battery pack, the IR LED on the hole on the bridge, the 56-ohm resistor, and the button all in a circuit. With this all inside the bottom half of the base we placed the half base over the center of the hole in the square. We left the IR LED on the whole entire time and didn't touch anything for the whole testing period. We measured for 3 and a half hours and took recordings every 30 minutes. After taking all this data and graphing it we can see that the rate at which the voltage drops starts to slow down over time. The biggest drop of voltage is from when we started to 30 mins in with a drop in voltage of 0.164 volts. The smallest drop in voltage is from 3 hours to 3.5 hours with a drop of 0.022 volts. As we can see this matches up with can be seen that the rate at which voltage drops decreases over time. The most dramatic changes in voltage will be at the beginning when the LED is turned on. The longer the LED is on the less voltage drop there will be.

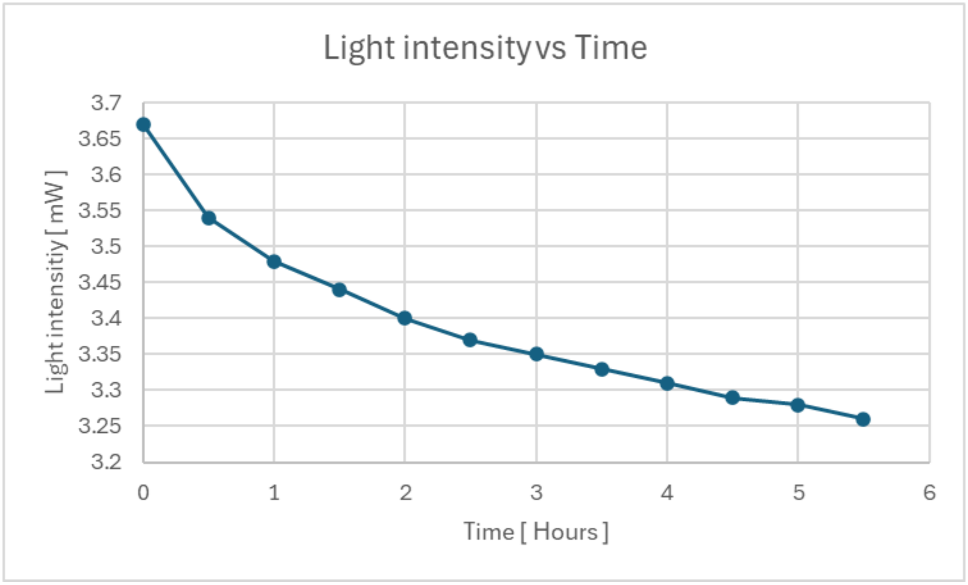


FIGURE 44 LIGHT INTENSITY VS TIME

In this test we measured the light intensity over a period of 5 and half hours. We did this by building the LED circuit with the batteries in the battery holder, the button, and using two 100-ohm resistors in parallel. We built this on a breadboard and held the breadboard sideways using two blocks of wood to keep the breadboard from moving which would disrupt our testing. We then used a power meter set to measure light at 940 nanometers. We left this set up like this for the whole 5 and half hours without touching it and recorded the power measured by the power meter every 30 minutes. From the chart we can see that the light intensity drops over time. We can also see that that the rate at which the intensity drops slows down over time. At the beginning of the test the power meter measured a light intensity of 3.67 mW and at the end of the 5 and a half hours the intensity measured was 3.26 mW. This is a drop of 0.41 mw over the 5.5 hours. The biggest drop is from the beginning of the test to 30 minutes where the light intensity drops from 3.67 mW to 3.54 mW which is a drop of 0.13 mW. The smallest drop is from 4 and half hours to 5 hours were the light intensity drops from 3.29 mW to 3.28 mW which is a drop of 0.01 mW.

9.1.2 – Photodiode Testing

TABLE 21 RESISTOR TESTING FOR THE TRANSIMPEDANCE AMPLIFIER CIRCUIT USED WITH THE PHOTODIODE

Resistor Value	Type of Measurement	Measured Value
3MΩ	Covered	≅20mV
	Ambient Light	≅300mV
	Light from fiber	Up to 1.7V
6.2MΩ	Covered	≅60mV
	Ambient Light	≅0.66V
	Light from fiber	1.5V - 3V
6.8MΩ	Covered	≅60mV
	Ambient Light	≅0.73V
	Light from fiber	1.6V - 3V
7.5MΩ	Covered	≅60mV
	Ambient Light	≅0.82V
	Light from fiber	2V - 6V
10MΩ	Covered	≅60mV
	Ambient Light	≅1.06V
	Light from fiber	1.75V - 4.5V

Once we determined that we needed to use a transimpedance amplifier circuit with our photodiode, we then had to figure out the value of the resistor. To do this, we had to test different resistor values under certain conditions so we could get a good idea of how that specific resistor value affected the output voltages that we would be using as the values to differentiate the chess piece types.

We conducted our testing in the senior design lab in the engineering building as they had the best assortment of resistor values to test with. The ambient light in the senior design lab is normal for any room, and we wouldn't consider it as too dark or too bright to where it would have negatively affected our results. When first constructing the transimpedance amplifier circuit, we were only using resistors such as 1kΩ and 10kΩ. After doing some research, we realized that we would need to increase the value of the resistor greatly so that our voltage outputs wouldn't be so low in the millivolt range.

As shown in table 22, we tested many different resistors in the mega ohm range. We determined that the mega ohm range was what provided us with the best results. We tested different resistors in this range while the photodiode was under various conditions. We tested when the photodiode was covered so that we could see what a "base" value would be. We tested when the photodiode was only receiving ambient light as our photodiodes in our chessboard would be sensing that whenever a piece wasn't on top of a square. We then tested how much voltage was produced from the circuit when the fiber would guide the green LED light towards the photodiode. We did not account for the light reflecting off the bottom of a piece as we wanted to get a good understanding of the amount of light that would be coming out of the fiber. We needed to know this to better determine during what "stage" of the process do we lose the most amount of light and why our final output values would be the way they are. Not worrying about the light reflecting off the bottom of a piece eliminates a lot of variables we eventually had to be concerned about such as proper placement and alignment of all parts.

As shown in table 22, we determined through testing that the 7.5MΩ resistor would be the best fit for our circuit. The 7.5MΩ resistor provides us with the greatest range of output voltage values while also not having too large of an ambient light value. Something we noticed during our testing was that there was a certain "sweet spot" for the resistor values that would increase and "maximize" our voltage range values while keeping the ambient light measurement low enough to provide us with the best possible difference between the two values. The sweet spot for the resistor values could be compared to a bell curve, where the peak of the bell curve would represent the best resistor value that would maximize our output voltages while every other value decreases in "efficiency" and will not be able to provide the largest range of values between ambient light and the measured intensity of light from the fiber. 7.5MΩ resistor value was the closest to that "sweet spot" that we noticed as once we went up to 10MΩ, then the ambient light measurement significantly increased when compared to the range of voltage outputs that we received. By increasing the ambient light measurement, it would decrease the difference in output when there is light provided to the photodiode to when there is just ambient light. Decreasing the difference between the two measurements would not be good for our project as it lessens the amount of wiggle room that the values differentiating each piece would need. This wiggle room would account for variability that could occur from square to square. Once there is overlap with the values that differentiate every piece, then our entire project doesn't work properly, and our piece identification accuracy would be nearly zero.

Once we determined the right resistor value for our circuit, we then did some calculations to determine what kind of photocurrent was being produced by the photodiode under those certain conditions. As shown below, we used Ohm's law to determine the photocurrent value. The photocurrent we calculated that was being produced ranged from 8nA to sub microamps. These photocurrents are expected and make sense for the photodiode we chose to use.

Calculations to find photocurrent produced in the circuit:

$$V_{out} = R_f * I_d$$

(Covered)

$$60\,mV = 7.5M\,\Omega * I_d$$

$$I_d = 8 * 10^{-9}A = 8nA$$

(Ambient Light)

$$0.82V = 7.5M\,\Omega * I_d$$

$$I_d = 1.093 * 10^{-7}A = .1093\,\mu A$$

(Light from Fiber Output)

$$3V = 7.5M\,\Omega * I_d$$

$$I_d = 4 * 10^{-7}A = .4\,\mu A$$

FIGURE 45 CALCULATIONS USING OHM'S LAW TO FIND THE PHOTOCURRENT PRODUCED BY THE PHOTODIODE.

The calculations done for the photocurrent also help validate calculations done to show that the photodiode we used has an accurate responsivity. As shown in figure 33 below, we used a simple ratio to prove that the responsivity of the photodiode is accurate and will lead to similar photocurrents we got when testing based on the intensity of light that we measured earlier. The responsivity of the photodiode we chose for this project is stated as 0.034 A/W or 34 mA/W. This responsivity is not the best available for photodiodes, but we believe it will be enough for this specific project and its goals. When using the power measured of the green LED before the light goes into the fiber optic cable, we calculate that the photocurrent produced by the photodiode would be 34 microamps. When recalculating the photocurrent using the power of the light coming out of the fiber, we get 0.4896 microamps. This photocurrent calculated is similar to the photocurrent calculated when the voltage produced by the circuit is 3V.

Calculations to prove accurate responsivity value of photodiode:

Responsivity of photodiode = 0.034 A/W = 34 mA/W

(Using power from green LED): $\frac{34\ mA}{1\ W} = \frac{x}{1\ mW} \quad x = 34\ \mu A$

(Using power from fiber output): $\frac{34\ mA}{1\ W} = \frac{x}{14.4\ \mu W} \quad x = 4.896 * 10^{-7} A = .4896\ \mu A$

FIGURE 46 CALCULATIONS TO PROVE OUR PHOTODIODES HAVE AN ACCURATE RESPONSIVITY

These calculations justify that our measurement of the intensity of light before and after the fiber optic cable is approximately correct and similar to the actual intensity of light that will be used in our project.

9.1.3 LCD Display Testing

The display we used has an ILI9481 driver chip, a 4” screen and a resolution of 480x320 pixels. It is a versatile and compact display and, most importantly, one of the factors that most contributed to the choice of this component, it is built on Thin-Film Transistor (TFT) technology, which should ensure swift response times during screen transitions, which is very important since it will be the only graphical interface the user can interactive with, and the easiest way for us to convey detailed information to the user that would otherwise not be possible considering the other forms of user-product interaction, which are very limited on that regard.

Another notable feature of this display is its integrated touch screen functionality, which initially gave us the impression that it would add to and further improve the interactive elements of the project, enabling users to input data or interact directly with the display without the need for additional mechanisms like buttons and switches. This idea had us excited at the thought of enhancing the overall user experience to even greater heights, however, after getting our hands in one such display, we decided against the use of the touchscreen and decided to fall back to the original idea of utilizing buttons to operate the system. The small size of the display and the cheap nature of the tech was enough to convince us that the touchscreen might not be as reliable as we would need it to be to have it as the only method of navigating through all the menus and scenes without turning the experience frustrating to the user.

The process of wiring our display is relatively simple. It has a total of 14 pins, only 9 of which we are interested in since we are not using the display’s touchscreen capabilities. Although simple, it is still important to take note of the documentation to be capable of properly wiring the module to our microcontroller, which will be commanding display operations. The pins and their descriptions can be found on the table below, we then touch briefly on the most important pins and their functions within our design.

TABLE 22 LCD MODULE PIN INFORMATION

Number	Module Pin	Description
1	MISO	SPI bus read data pin
2	LED	Backlight control pin
3	SCK	LCD SPI bus clock pin
4	MOSI	LCD SPI bus data pin
5	DC/RS	LCD data / command selection control pin
6	Reset	LCD reset control pin
7	CS	Chip select control pin
8	GND	Power ground pin
9	VCC	Power positive pin

As evident from the table, we can easily wire the display thanks to its use of SPI communication, which means we would only need four wires to send the data to be displayed. Pins 1, 3, 4, and 7 together make the 4-wire SPI configuration and even then, we can afford to maintain pin 1 as a non-connection pin since the MISO (Master In Slave Out) line is not strictly necessary for SPI communication, which primarily involves the MOSI (Master Out Slave In), SCK (Serial Clock), and CS (Chip Select) lines. The MISO line is primarily used for full-duplex communication, allowing the slave to send data back to the master, which is not the case for this application. The remaining pins are relatively straight forward and serve more basic functions, such as providing power to the module or determining the display’s brightness level.

The ILI9xxx controllers are a widely recognized controller line for driving TFT LCD displays, these drivers enhance the module's performance and compatibility with different microcontrollers. With support for RGB-565 color, this controller can display a range of 65 thousand different colors, ensuring that the module can render vibrant and colorful images, contributing to the display’s (and by extend the project’s) overall visual appeal. The display does suffer slightly from its size and resolution however, and we expect that positioning it flush with the board might cause some visibility problems from the perspective of the players. To deal with this issue, we want to implement, if possible, a mechanism that will allow users to adjust the angle of the display so that it can face the players instead of remaining on a horizontal position on the top of the board. Please refer to the section on stretch goals for further detail on the topic of display visibility.

The last characteristic important to mention as we finalize this first round of tests on the display, the sheer number of online resources regarding this and similar models of TFT displays. As mentioned above, this controller series boasts a large popularity and consequently benefits from an active online community that provides access to valuable support for troubleshooting, allowing us to feel free to explore innovative solutions to any design problems that may surface. Not only that, but the manufacture’s website contains libraries and many different example codes for the module, making it friendly towards us, developers, as we are able to easily find official reference material to base our work on. For our testing, we utilized the TFT_eSPI library, which is available online on github.

9.1.4 Voltage Regulation Testing

The buck converters we are using in this project are found readily available in the market in the form of small self-contained and ready-to-use modules. They utilize LM2596 series chips, a collection of integrated circuits that provide all the active functions necessary for a step-down switching regulator. These chips can drive a 3 Amp load, which should be well over our load requirements for this chessboard.

These regulators are easy to use, requiring only a minimum number of external components and are available in both fixed and adjustable output version. We used the latter for this product even though our design runs exclusively on 5V, we’d like the flexibility to add components that utilize different voltage levels in the future should the need arise.

The main concern over the use of this buck converter was that it would need a higher voltage battery to be able to properly stabilize the voltage because, when completely discharged, our battery will have approximately 5.8V which makes the difference between output and input voltages in the buck converter relatively small. However, during testing, the voltage was properly regulated even with voltage differentials between input and output were as low as 0.7V, which means we are safe to use a 2S battery pack that will fluctuate between 5.8V and 8.4V when fully discharged and fully charged respectively.

9.2 - Plan for Senior Design 2

Going into senior design 2 next semester, we feel pretty good with the progress we made so far. We figured out most of the important dimensions and positioning of the components under each square and we managed to order and receive most of the parts already before winter break.

Our main objective to start on in senior design 2 is to build a prototype of our board so that we can test the chess piece identification system and associated subsystems. We need to build a prototype to test the entire process of our project from beginning to end. We need to test that enough light can go into the fiber optic cables from the green LED and travel to each square and provide enough light intensity to reflect off the bottom of a piece that would be on top of the square so that the photodiode can receive a signal and produce a voltage output that would be transmitted to the microcontroller so it can detect and identify that particular piece. To do this we need to build a 3x3 grid representing our chessboard as shown in figure XXX and also a couple of different pieces to test with.

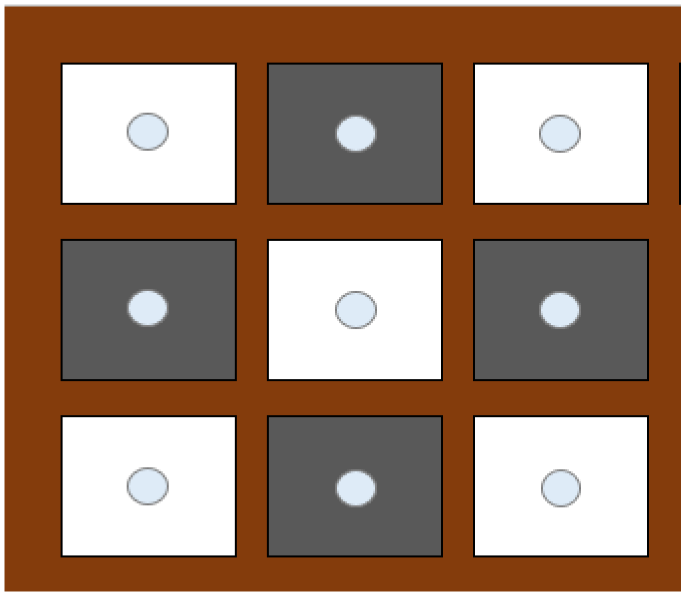


FIGURE 47 DIAGRAM OF A 3X3 SECTION OF OUR CHESSBOARD

The reason why we need a 3x3 grid of our chessboard to test with is because that is the optimal size that would enable us to be able to test various chess piece types and certain chess gameplay situations/interactions while also not being too big to build and sink a lot of time and resources in as it is just a partial prototype of the entire chessboard. A 3x3 prototype of our chessboard would enable us to test scenarios with certain pieces such as pawn movement and capturing, en passant, a knight moving past pieces in front of it, etc. Through testing on the prototype chessboard, it should help us figure out any issues and complications with our parts/components and any design flaws that would negatively affect the entire project.

9.3 Project Stretch Goals & Secondary Design

In this subsection, we discuss additional features that we would like to have added to the design. These mostly consist of low-priority features that are not essential but would make for great additions to the project. Unfortunately, due to time and design constraints, there are features that may not be implemented into the final design. We believe these features are still worth noting, as they give some further objectives to work towards in the future.

9.3.1 Display Visibility Improvements

For us, user experience and accessibility were top concerns when visualizing our design and the features we wished to include. During one of our meetings, a concern was raised over the visibility of the display included in one of the edges of the board. Visibility significantly impacts the functionality of the display and effectiveness of the information displayed. Whether the LCD screen conveys important data, such as our menus and tooltips, or real-time updates displaying or clock timers, ensuring that users can easily read and comprehend the information is pivotal to us.

The problem in question was of concern due to the small size of the display and the position it would be installed in, and its likelihood to cause the image to not appear to the players as clear as we would like it to be during the use of the device and therefore, spoil the overall experience and possibly render the display pointless to some degree.

After ordering our parts and getting our hands on the display, this issue resurfaced, and we decided to make one of our stretch goals to develop a mechanism that will allow us to support the display, so it stands at an angle above the chessboard top to increase screen visibility during use. This idea came to be because we wanted to keep the board flush for storage and transport purposes while also having the increased display capabilities that a protruding would make possible. We did, however, push it to a stretch goal since it isn’t a requirement for the proper functionality of the board but would add a nice touch to the design if implemented.

Still, the implementation of an adjustable mechanism for the LCD display requires careful consideration of design elements, there are certain stipulations and design restraints that we need to abide by if we wish to properly implement this feature in the future. First, the mechanism should be seamless in operation, sturdy in construction, and should not compromise the overall aesthetics of the project, otherwise, the implementation of this piece would be detrimental to the overall project instead of helpful in any manner. Additionally, the success of the adjustable LCD display also hinges on its user-friendliness. The adjustment mechanism should be intuitive, requiring minimal effort from the user and providing a range of angles to accommodate diverse user preferences.

Lastly, and probably most importantly, while designing the adjustable mechanism, it's crucial to account for the power supply and connectivity of the LCD display. Afterall, we won’t be able to keep the moveable screen soldered to a PCB, which means we would need to route cables to connect the display module to the remainder of the circuit, all the while ensuring that these additional elements can be seamlessly integrated with the display mount, avoiding any disruption when changing the screen angle. This integration should not only be functional but also maintain a tidy and organized appearance, preferably contained within the inside of the box away from the user's view.

Much is still to be discussed regarding this issue, but some initial ideas can start to be explored when it comes to the implementation of this device. The simplest approach to which involves incorporating a kickstand-like mechanism, much like one would use on a bicycle to keep it from falling over. This mechanism would run over a rail containing multiple notches that would prevent the stand from easily falling and allowing users to prop the display up at their preferred angle. This idea was inspired by simplistic tablet and display stand designs that can be found in the market. The figure below helps to illustrate the concept, however, note that, as a low-priority feature, the schematic below is but a crude sketch and hardly represents the finished product, as the design will have to go through extensive testing to discern the best method for its implementation.

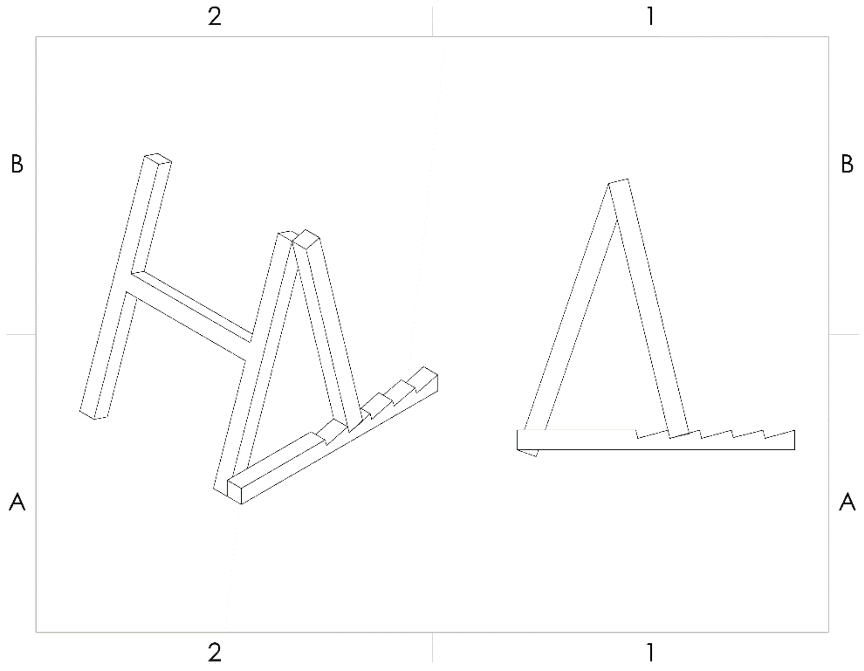


FIGURE 48 ROUGH SKETCH ILLUSTRATING A POSSIBILITY FOR THE DISPLAY MOUNT MECHANISM

9.3.2 Sound System

Incorporating a sound system into our chessboard design could significantly enhance the overall user experience and engagement. It not only serves as a practical tool for conveying information but also contributes to the overall enjoyment and immersion of the gaming experience.

Integrating sound effects would provide us with yet another route of interaction with the player during gameplay. For instance, a distinctive sound could play when a player makes a move, captures an opponent's piece, or successfully executes a checkmate. This auditory feedback adds a dynamic and immersive layer to the chess-playing experience, making each move more satisfying and memorable. Not only that, but there is often information that is easier to communicate in audio form rather than through visual means.

For example, it would be possible to utilize the sound system for alerts and notifications such as a gentle tune to remind a player of their turn when they start taking long to make a move, or a louder alert sound for when a player realizes an illegal move or to otherwise indicate danger when they have their king in check.

10 Administrative Content

10.1 Budget Estimates and Actual Budget

TABLE 23 BUDGET ESTIMATES

	Parts	Price per part ($)	Quantity	Cost ($)
1	LED’s	$0.07-3	65	$4.55-195
2	Sensors	$0.10-3	64	$6.4-192
3	Power supply	$30-110	1	$30-110
4	PCB	$40-120	2	$80-240
5	Microcontroller	$20-40	1	$20-40
6	Chess board material	$7-25	1	$7-25
7	Chess pieces set	$2.50-60	1	$2.50-60
8	Fiber optic cable	$0.30-2	64	$19.2-128
9	Display screen	$10-$90	1	$10-90
10	Charging controller	$20-80	1	$20-80
	Total			**$199.65-1160**

TABLE 24 ACTUAL BUDGET FOR EVERYTHING USED

Person	Total	# of items	Item Name/Description
Nikolai Coletta	$33.82	1	Teensy 4.1 Development Board
Cassidy Phillips	$33.45	1	PJRC Teensy 4.1 ARM Cortex-M7Processor at 600 MHz with a NXP iMXRT1062
Cassidy Phillips	$0.59	2	Header 24*1 For Teensy 4.1 PCB
Cassidy Phillips	$1.18	6	SN74HC595N (Shift registers)
Cassidy Phillips	$0.57	12	CD74HC4051E (Multiplexers)
Cassidy Phillips	$1.77	20	OPA4990IDR (Op amps)
Cassidy Phillips	$1.38	5	SN74HC595N (Shift registers)
Cassidy Phillips	$38.34	1	PCB Order 3 (Backup Boards)
Cassidy Phillips	$54.19	1	PCB Order 1 (LED PCB plus other board V1)
Cassidy Phillips	$17.76	1	Micro USB adaptor
Cassidy Phillips	$13.58	1	0805 Inductors
Vinicius Resende	$13.60	10 ft.	Heat Shrink Wrap for Batteries
Vinicius Resende	$6.99	5	ON/OFF Switches
Vinicius Resende	$4.99	1	Thermal/Insulating Tape
Vinicius Resende	$12.31	1	9V Power Adapter & Jack
Vinicius Resende	$15.68	1 (6 Pack)	Wire
Vinicius Resende	$29.52	6	Batteries (2nd Order)
Vinicius Resende	$10.21	5	Battery Protection Board
Vinicius Resende	$18.99	1	4" Display
Vinicius Resende	$7.99	5	LM2596 DC to DC Buck Converter
Alec Barno	$24.39	1	SIOTI square filter kit with ND2, ND4, ND8, and ND16 filters

Alec Barno	$18.38	1	Lighting Neutral Density Gels Filter Sheet with ND3, ND6, and ND9 filters
Alec Barno	$20.32	1	Cloudray 900-1600nm IR Detection & Alignment Visulizer Card
Alec Barno	$42.12	1	Clear Acrylic Sheet 1/4" Thick 18"x36"
Alec Barno	$16.06	2	FolkArt Top Coat Frost Effect, 8oz
Alejandro Felix	$42.17	4	12 pack LAMPVPATH AAA Battery Holder, 2 x 1.5V AAA
Alejandro Felix	$19.42	2	mxuteuk 30pcs Micro Push Button Switch
Alec Barno	$8.55	1	FLANCCI LED Light Blocking Stickers (Blackout stickers)
Alejandro Felix	$5.85	100	EDGELEC 100pcs 56ohm Resistor
Alec Barno	$19.16	1	ELEGOO PLA 3D Printer Filament 1.75mm Space Grey 1KG
Alec Barno	$16.97	1	AMEROUS Upgraded Weighted Chess Pieces with 3 inch King
Alec Barno	$69.91	3	48 pack of Energizer AAA Batteries
Alec Barno	$12.64	20	PD204-6B Photodiode 940nm from Digikey
Alec Barno	$19.94	100	PD204-6B Photodiode 940nm from Digikey
Alejandro Felix	$28.87	50	SFH 4544 IR LED Emitter 940nm from Digikey
Alec Barno	$23.76	6	Loctite super glue
Alec Barno	$2.68	1	5 colored rolls of electrical tape
Alec Barno	$4.99	1	Plaskolite plastic cutter
Alec Barno	$19.99	2	3ft 1.5"x1.5" poplar hobby board and multicolored pushpins

The final total cost of the entire project for everything that was used and put inside the final housing came out to be $733.07. This total cost was much higher than expected but was still inside our the estimated budget from Senior Design One.

TABLE 25 ACUTAL BUDGET FOR NON-USED ITEMS

Person	Total	# of items	Item Name/Description
Nikolai Coletta	$6.39	1	USB C to Micro USB Cable
Cassidy Phillips	$4.99	1	MCIGICM 10pcs Male Header strip (2.54mm) for Arduino Connector
Cassidy Phillips	$12.98	1	Glarks 112pcs 2.54mm Male and Female Pin Header Connector Assortment Kit
Cassidy Phillips	$7.99	1	10 pcs CD4051 CMOS Analog Multiplexers/Demultiplexers
Cassidy Phillips	$6.99	1	BOJACK LM324N Quadruple Operational Amplifier LM324
Cassidy Phillips	$6.25	5	MKL02Z32_t4_QFN16

Cassidy Phillips	$1.20	5	W25Q64JVXGIM_DFN8
Cassidy Phillips	$1.56	6	TLC59210IN
Cassidy Phillips	$203.91	1	PCB Order 2
Vinicius Resende	$13.63	10	Batteries (1st Order)
Vinicius Resende	$5.78	2	USB-C Connector
Vinicius Resende	$114.02		Various Tools for Soldering, Welding, Etc.
Alec Barno	$6.95	10	uxcell 10pcs photodiode, 3mm clear round head
Alec Barno	$7.45	3	uxcell photodiode module with digital and analog output, 3pcs
Alec Barno	$8.46	5	uxcell photodiode, 5mm round head
Alejandro Felix	$10.57	1	AKEPO PMMA END Glow Fiber Optic Cable Kit with 100pcs
Alejandro Felix	$7.17	100	Ultra Bright 5mm Round Transparent Green LED
Alec Barno	$13.77	10	10mm LED Optical Lens, Smooth Convex
Alejandro Felix	$8.51	100	3mm IR 940nm LED, Clear Transparent Round Head
Alejandro Felix	$22.32	3	12 pack LampVPath CR2032 Coin Cell Battery Holder
Alejandro Felix	$10.69	40	PGSONIC CR2032 3V Lithium Battery
Alec Barno	$7.19	5	VTP9812FH Photodiode 580nm from Digikey
Alec Barno	$10.34	10	SFH 203 P Photodiode 850nm from Digikey
Alejandro Felix	$9.69	10	5mm IR LED 940nm 30 Degree Viewing Angle from LEDSUPPLY
Alec Barno	$32.00	1	VIS Light Diffusing Film 100x100mm
Alec Barno	$6.41	1	Electrial tape

The total cost of all the items we bought, used, tested but never put inside the final project was $547.21. Since we had to redesign our project multiple times, we ended up having many leftover items that we used for prior versions of our project. We unfortunately didn't expect this part of the senior design project to cost this much, but all of the items listed in Table 26 were critical for us to figure out our final designs. The total cost of Table 25 and Table 26, which counts everything that was ever bought for this project came out to be $1,280.28. This final cost was higher than our initial estimate, but our initial estimate also didn't consider a lot of the new parts our designs required.

10.2 Milestones

TABLE 26 FALL 2023 SENIOR DESIGN 1 MILESTONES

Fall 2023		
Description	Time	Dates
Project ideas	3 weeks	August 21- September 8
Project Division	1 week	September 9 – September 15
Initial project documentation		September 15
Research on past projects	3 weeks	September 16 – October 7
Individual writing	4 weeks	October 8 - November 2
60-page draft		November 3
Design and Development prototyping	4 weeks	November 4 - November 26
Documentation Review and Purchasing Components	1 week	November 27- December 4
Final Documentation		December 5

TABLE 27 SPRING 2024 SENIOR DESIGN 2 MILESTONES

Spring 2024		
Description	Time	Dates
Testing of component	1 week	January 8 - January 14
Building prototype	8 weeks	January 15 - March 10
Testing prototype	3 weeks	March 11 – March 31
Finalizing the project	2 weeks	April 1 – April 14
Final Documentation and Presentation	2 weeks	April 15 – April 28

10.3 Work Distribution Table

TABLE 28 FALL 2023 SENIOR DESIGN WORK DISTRIBUTION

	Alec	Alex	Cassidy	Nikolai	Vinny
Piece Identification	Primary	Secondary		Secondary	
Fiber Optic System	Secondary	Primary			
Software			Secondary	Primary	Secondary
Microcontroller			Secondary	Primary	Secondary
Power	Secondary	Secondary			Primary
PCB Design			Primary	Secondary	Primary
LED Array	Secondary	Secondary	Primary		
System Fabrication	Primary	Secondary	Secondary	Secondary	Secondary

When deciding on work distribution for our project, one of the main goals was to aim for a majority of the group (minimum of 3 out of 5 people) to have a role in each specific task so that there are at least 2 “backup” people to help the person with the primary role in a specific task. We believe this is the best way to ensure that every task in the project will be created in the best way possible through a “majority rules” idea. Another point of clarification about our work distribution table is that for system fabrication, everyone in the group will have either a primary or secondary role for that task because everyone in the group will have input and make decisions about how our project should be built, how the design/aesthetic of the board will be and how putting together all the components inside the board will occur. Everyone in the group will be responsible for getting their specific tasks parts and components and we came together as a group to discuss how to best implement them and create a finished product.

11 Conclusion

This paper described the extensive research and development journey that our group undertook to conceptualize and refine our project. Delving into the intricacies of our project's many systems, we not only gained valuable insight into its theoretical innerworkings but have also come across and were able to overcome a handful of practical challenges that arose throughout our testing.

As we transition into the next semester of Senior Design II, this comprehensive understanding serves as a solid foundation for the upcoming stages of prototyping and building the final product. The knowledge acquired during this research phase will undoubtedly guide us in making informed decisions during the prototyping process. With the preliminary stages of testing out of the way, we are confident on our ability to put together a working prototype for the next semester.

This paper will provide a clear roadmap for the future of this project and prove its usefulness as a tool for aiding us on tackling the complexities that lie ahead. This synthesis of theoretical knowledge and practical insights not only enhances the robustness of our project but also prepares us for the challenges that characterize the final stages of senior design.

The biggest challenge coming forward that will stand in our way during the construction of our chessboard prototype will be its piece detection and identification system. Our current approach utilizes delicate and precise optical components, making our system's readings susceptible to be easily disturbed through many different factors in case we fail to approach the problem in the correct way. The group has had many different ideas regarding this issue and have thought of many ways to minimize this problem, some of which we outlined in this paper. However, it is only through the extensive prototyping testing during the upcoming semester that we were able to discern the most beneficial method of implementation.

The collaborative effort and dedication invested in this research paper set the stage for a successful implementation phase. Each team member played a pivotal role in contributing their skills, experiences, and perspectives to the project. Through open communication and shared commitment, we forged a cohesive team dynamic that fueled creativity and innovation. Regular collaborative sessions facilitated the exchange of ideas, allowing us to navigate challenges collectively and make informed decisions. This collaborative spirit not only enriched the quality of our work, but also fostered a supportive environment where everyone's contributions were valued and considered. As we transition into the next semester, these bonds will certainly ensure a seamless and effective execution of the implementation phase.

In essence, Senior Design I laid the groundwork for the future of our project in this initial research and development stage. We are moving towards Senior Design II confident that our work through this past semester will aid us in the process of integrating our different systems together and creating the initial version for our prototype, which will mark a critical milestone in the realization of our senior design project. The careful orchestration of the plans set in this document, coupled with the lessons learned from our research and development, positions us for success as we move towards the final stages of bringing our vision to life. However, it is important to note that we won’t stop with what is laid out here, after all, we are always looking to improve our work and elevate it to the highest degree. This paper are guidelines for us to follow and we recognize that, should we develop a better understanding of a superior solution or come to imagine a feature we would like to add, we shouldn’t hold ourselves back from pursuing it, after careful consideration of course.

All in all, this semester in Senior Design has been an incredibly positive and transformative experience for our team. Working on our project provided us with the opportunity to apply the theoretical knowledge gained throughout our academic journey. As we conclude this first leg of Senior Design, the sense of accomplishment is immeasurable as we witnessed our project taking shape through this paper. The challenges we’ve navigated through and the skills we’ve learned in this semester will serve us well through Senior Design II.

Appendices

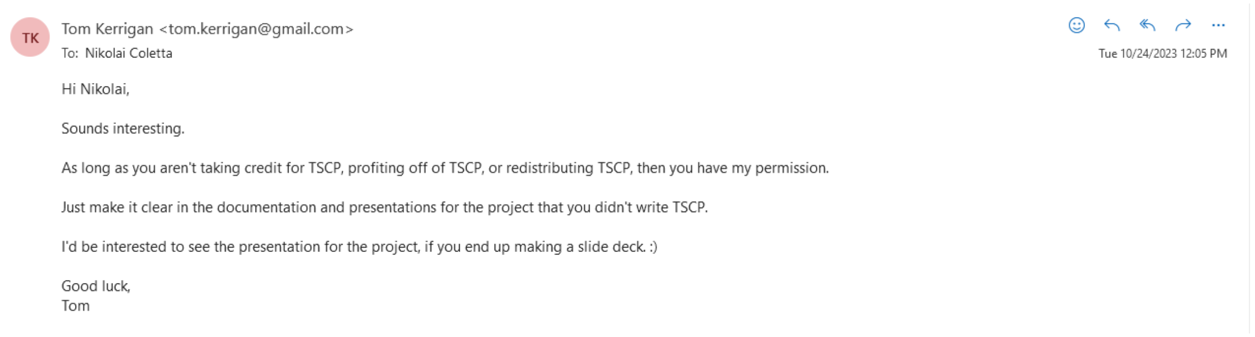
Copyright Permissions

Authorization to use Lichess: https://lichess.org/contact#help-authorize

Authorization to use ChatGPT: https://openai.com/policies/terms-of-use

Authorization to use Figma: https://help.figma.com/hc/en-us/articles/360042296374-Figma-Community-copyright-and-licensing

Approval for usage of Tom Kerrigan's Simple Chess Program (TSCP):



TK

Tom Kerrigan <tom.kerrigan@gmail.com>

To: Nikolai Coletta

Tue 10/24/2023 12:05 PM

Hi Nikolai,

Sounds interesting.

As long as you aren't taking credit for TSCP, profiting off of TSCP, or redistributing TSCP, then you have my permission.

Just make it clear in the documentation and presentations for the project that you didn't write TSCP.

I'd be interested to see the presentation for the project, if you end up making a slide deck. :)

Good luck,
Tom

References

1. "Learn How To Play Chess." Chess.com, Chess.com, www.chess.com, www.chess.com/learn-how-to-play-chess.

2. "Stockfish - Open Source Chess Engine." Stockfish, stockfishchess.org, stockfishchess.org/.

3. "AMD & Intel Chess Benchmark – Stockfish." IP Man Chess, ipmanchess.yolasite.com, ipmanchess.yolasite.com/amd--intel-chess-bench-stockfish.php.

4. Muller, H.G. "MAX, a MAXimum strength chess program." Home HCCNet, home.hccnet.nl, home.hccnet.nl/h.g.muller/max-src2.html.

5. "CCRL 40/40." Computer Chess Rating Lists, computerchess.org.uk, www.computerchess.org.uk/ccrl/4040/index.html.

6. "Elo Rating System." Chess.com, Chess.com, www.chess.com/terms/elo-rating-chess.

7. "US Chess Rating Distribution." US Chess, uschess.org, www.uschess.org/archive/ratings/ratedist.php.

8. Kerrigan, T.C. "TSCP: Tom's Simple Chess Program." TCKerrigan.com, www.tckerrigan.com, www.tckerrigan.com/Chess/TSCP/.

9. "GNU Chess." Wikipedia, en.wikipedia.org, en.wikipedia.org/wiki/GNU_Chess.

10. "Picking the Right Tool for the Job: MCU, SBC, or FPGA." Mouser Electronics, Mouser, www.mouser.com, www.mouser.com/blog/picking-the-right-tool-for-the-job-mcu-sbc-or-fpga.

11. "SparkFun Pro nRF52840 Mini - Bluetooth Development Board." SparkFun, www.sparkfun.com, www.sparkfun.com/products/15583.

12. "Teensy 4.0 Pins." PJRC, www.pjrc.com, www.pjrc.com/store/teensy40_pins.html.

13. "Pixel Pipelines Definition." Techopedia, www.techopedia.com, www.techopedia.com/definition/89/pixel-pipelines.

14. "ESP32 Datasheet." Espressif Systems, espressif.com, www.espressif.com/sites/default/files/documentation/esp32_datasheet_en.pdf.

15. "HiLetGo STM32F407VET6 STM32 Cortex-M4 MCU Core Board Development Board." Ubuy Indonesia, www.ubuy.co.id, www.ubuy.co.id/en/product/23G74LD8-

hiletgo-stm32f407vet6-stm32-cortex-m4-mcu-core-board-development-board-nrf2410-fmsg-sd-card.

16. “PSRAM for Teensy 4.1.” PJRC, www.pjrc.com, www.pjrc.com/store/psram.html.

17. “ChatGPT traffic slips for third month in row.” Reuters, www.reuters.com, www.reuters.com/technology/chatgpt-traffic-slips-again-third-month-row-2023-09-07/.

18. “Collaborative Intelligence: Humans and AI Are Joining Forces.” Harvard Business Review, hbr.org, www.hbr.org/2018/07/collaborative-intelligence-humans-and-ai-are-joining-forces.

19. “What is prompt engineering?” McKinsey & Company, www.mckinsey.com, www.mckinsey.com/featured-insights/mckinsey-explainers/what-is-prompt-engineering.

20. OpenAI. “Chat with OpenAI.” Chat OpenAI, chat.openai.com, chat.openai.com/.

21. “Safety of toy electric appliances for children - IEC 62115.” International Electrotechnical Commission, webstore.iec.ch, webstore.iec.ch/publication/28054.

22. “Secondary cells and batteries containing alkaline or other non-acid electrolytes - IEC 62133.” International Electrotechnical Commission, webstore.iec.ch, webstore.iec.ch/publication/32662.

23. “ISO/IEC 9899: Programming languages — C.” ISO, www.open-std.org, www.open-std.org/jtc1/sc22/wg14/

24. “ECSTUFF4U for Electronics Engineer.” *Advantages and Disadvantages of LASER Diode*, www.ecstuff4u.com/2018/02/advantages-and-disadadvantages-of-laser.html. Accessed 3 Nov. 2023.

25. Meyer, Michael. “HeNe Lasers vs Diode Lasers: Hene Laser Pros and Cons.” *Solid State Lasers and Laser Diodes from RPMC Lasers Inc*, 4 Aug. 2023, www.rpmclasers.com/blog/hene-lasers-vs-diode-lasers-hene-laser-pros-and-cons/.

26.“Fiber Optic Lighting.” *FOA*, www.thefoa.org/tech/lighting/lighting.html. Accessed 3 Nov. 2023.

27.ryno419. “The Pros and Cons of LED Lights.” *Brennan Electric*, 26 June 2023, brennan-electric.com/the-pros-and-cons-of-led-lights/.

28.You, Semiconductor For. “What Are the Advantages and Disadvantages of Laser Diode?” *Semiconductor for You*, 27 May 2021, www.semiconductorforu.com/advantages-disadvantages-laser-diode/.

29.Conditioning, Berkeys Air. “Advantage & Disadvantage of LED Lights.” *Berkeys Plumbing, Air Conditioning & Electrical*, 7 Apr. 2020, www.berkeys.com/2016/11/16/advantage-disadvantage-led-lights/.

30.Paschotta, Dr. R&uuml;diger. “Multimode Fibers.” *Multimode Fibers, Explained by RP; Optical Fiber, Large-Core Fibers, Fiber-Coupled, Laser Diodes*, RP Photonics AG, 2 Nov. 2023, www.rp-photonics.com/multimode_fibers.html.

31.Paschotta, Dr. R&uuml;diger. “Single-Mode Fibers.” *Single-Mode Fibers, Explained by RP; Launching Light, Monomode Fiber, Cut-off Wavelength*, RP Photonics AG, 2 Nov. 2023, www.rp-photonics.com/single_mode_fibers.html.

32.“What Do You Mean by Mode in an Optical Fiber?” *Quora*, www.quora.com/What-do-you-mean-by-mode-in-an-optical-fiber. Accessed 3 Nov. 2023.

33. Shaik, Asif. “Light Emitting Diode (LED).” *Light Emitting Diode (LED) - Working, Construction and Symbol - Diode*, www.physics-and-radio-electronics.com/electronic-devices-and-circuits/semiconductor-diodes/lightemittingdiodeledconstructionworking.html. Accessed 3 Nov. 2023.

34.“Thorlabs - HNL020RB HeNe Laser, 632.8 Nm, 2 MW, Random, 100 - 240 VAC Power Supply Included.” *Thorlabs, Inc. - Your Source for Fiber Optics, Laser Diodes, Optical Instrumentation and Polarization Measurement & Control*, www.thorlabs.com/thorproduct.cfm?partnumber=HNL020RB. Accessed 3 Nov. 2023.

35. *I2C-Bus Specification and User Manual - NXP Semiconductors*, www.nxp.com/docs/en/user-guide/UM10204.pdf. Accessed 3 Nov. 2023.

36. “IPC-A-610: The Standard for Acceptability of Electronic Assemblies.” *NextPCB*, www.nextpcb.com/blog/ipc-a-610. Accessed 3 Nov. 2023.

37. Kamprath, and Instructables. “RGB Led Matrix.” *Instructables*, Instructables, 30 Dec. 2017, www.instructables.com/RGB-LED-Matrix-1/.

38. “L6: RGB Leds.” *Physical Computing*, makeabilitylab.github.io/physcomp/arduino/rgb-led.html#:~:text=With%20the%20Common%20Anode%2C%20the,levels%20(e.g.%2C%20ground). Accessed 3 Nov. 2023.

39. Team, The Sierra. “IPC J-STD-001 Standard for Soldering.” *Sierra Circuits*, 28 Sept. 2023, www.protoexpress.com/blog/ipc-j-std-001-standard-soldering-requirements/.

40. *Linear vs. switching regulators: Unlocking the Best Power Solution for Your Design*. Renesas. (n.d.). https://www.renesas.com/us/en/products/power-power-management/linear-vs-switching-regulators

41. Battery University. (2022, March 3). *BU-304A: Safety concerns with Li-ion*. https://batteryuniversity.com/article/bu-304a-safety-concerns-with-li-ion

42. *Learnabout Electronics*. Buck Converters. (n.d.). https://learnabout-electronics.org/PSU/psu31.php