Knight Light LED Chess

Senior Design Documentation

Group 16

Nick DeSantis

Alexander Haas

Bryan Salicco

University of Central Florida

December 2012

Table of Contents

- 1 Executive Summary
- 2 Project Description
  - 2.1 Motivation and Goals
    - 2.1.1 Chess Piece Sensing and Tracking
    - 2.1.2 Display
    - 2.1.3 Gameboard Housing
    - 2.1.4 Human vs. Human Gameplay
    - 2.1.5 Human vs. Computer Gameplay
    - 2.1.6 Ease of Use
    - 2.1.7 Chess Computer Player
  - 2.2 Objectives
    - 2.2.1 LED Display
    - 2.2.2 LCD Display
    - 2.2.3 Human vs. Human Gameplay
    - 2.2.4 Human vs. Computer Gameplay
    - 2.2.5 Chess Computer
- 3 Requirements
  - 3.1 Software Requirements
  - 3.2 Hardware Requirements
- 4 Research
  - 4.1 Similar Existing Products
    - 4.1.1 Mephisto Chess Trainer
    - 4.1.2 Excalibur Grandmaster Chess Computer
    - 4.1.3 Product Comparison Summary
  - 4.2 Hardware
    - 4.2.1 Microcontrollers
      - 4.2.1.1 Microcontroller Specifications
      - 4.2.1.2 Programming Languages
        - 4.2.1.2.1 Assembly
        - 4.2.1.2.2 BASIC
        - 4.2.1.2.3 Pascal
        - 4.2.1.2.4 The C Programming Language
      - 4.2.1.3 Microcontroller Product Research
        - 4.2.1.3.1 CPU and Architectural Design
        - 4.2.1.3.2 Flash Memory and RAM
        - 4.2.1.3.3 Timers
        - 4.2.1.3.4 SPI Serial I/O Controller

4.2.1.3.5 I2C(Inter-Integrated Circuit)-bus serial I/O Controller
4.2.2 Sensors
4.2.2.1 Hall-Effect Sensors
4.2.2.2 Reed Switches
4.2.2.3 Photocells
4.2.2.4 Optical Detectors and Phototransistors
4.2.3 LEDs and LCDs
4.2.3.1 Liquid Crystal Displays (LCDs)
4.2.3.1.1 Blue Mode
4.2.3.1.2 Film Mode
4.2.3.1.3 Color and Double
4.2.3.1.4 Segment vs. Graphical
4.2.3.1.5 Reflective vs. Transmissive
4.2.3.1.6 Additional LCD Specifications
4.2.3.2 LEDs
4.2.3.2.1 Displaying Information
4.2.3.2.2 Bi-Color and Tri-Color
4.2.4 LED Drivers
4.2.4.1 Types of LED Power Converters
4.2.5 Power Supply
4.2.6 Shift Registers and I/O Expanders
4.2.6.1 Shift Registers
4.2.6.2 I/O Expanders
4.2.7 Development Boards
4.3 Software
4.3.1 General Chess Gameplay
4.3.2 Game Piece Properties
4.3.2.1 Pawns
4.3.2.2 Knights
4.3.2.3 Rooks
4.3.2.4 Bishops
4.3.2.5 Queens
4.3.2.6 Kings
4.3.3 Special Moves
4.3.3.1 Castling
4.3.3.2 En Passant
4.3.3.3 Pawn Promotion
4.3.4 Victory Conditions and Endgame Scenarios
4.3.4.1 Check
4.3.4.2 Checkmate
4.3.4.3 Stalemate

5 Design
- 5.1 Game Board Housing
- 5.2 Visualization
- 5.3 Piece Placement Sensing
- 5.4 General Game Logic
  - 5.4.1 Algebraic Notation
    - 5.4.1.1 Ambiguity
  - 5.4.2 Piece Movement Algorithms
  - 5.4.3 Gameplay Pseudo-Code
- 5.5 Artificially Intelligent Chess Computer
  - 5.5.1 Chess Piece Relative Value
- 5.6 Power Source
- 5.7 LCD Display and Buttons
- 5.8 Microcontroller Layout
- 5.9 PCB Plan

6 Design Summary
- 6.1 Software Design Summary
- 6.2 Hardware Design Summary

7 Project Prototype Testing
- 7.1 Hardware Test Environment
- 7.2 Hardware Specific Testing
  - 7.2.1 Chess Piece Sensing
    - 7.2.1.1 Optical Detectors and Phototransistors Sensing
  - 7.2.2 LED and LCD Testing
    - 7.2.2.1 LED Testing
    - 7.2.2.2 LCD Testing
  - 7.2.3 Power Source Testing
- 7.3 Software Test Environment
- 7.4 Software Specific Testing
  - 7.4.1 Piece Tracking
  - 7.4.2 Artificial Intelligence Tracking

8 Administrative
- 8.1 Senior Design 1 Milestones
- 8.2 Senior Design 2 Milestones
- 8.3 Budget and Finances
- 8.3 Project Summary
- 8.4 Conclusion

9 Appendix
- 9.1 Content Authorization

Table of Figures

Figure 4.1.1: Saitek's Mephisto Chess Trainer
Figure 4.1.2: Excalibur Grandmaster Chess Computer
Figure 4.2.2.1: Hall Effect Sensor
Figure 4.2.2.2: Reed Switch
Figure 4.2.2.3: Photocell
Figure 4.2.2.4: QRD1114 Phototransistor
Figure 4.2.4.1-1: TLC59116
Figure 4.2.4.1-2: TLC59116 Available Package Views
Figure 4.2.5-1: Fixed-output Generator
Figure 4.2.5-2: Generic Wall Wart
Figure 4.2.5-3: Generic Wall Wart with a USB Port
Figure 4.2.5-4: Barrel Jack Input Connector
Figure 4.2.5-5: USB-B Input Connector
Figure 4.2.6.2-1: MCP23017 Block Diagram
Figure 4.2.6.2-2: MCP23017 Available Packages
Figure 4.2.7-1: LPC2148 Header Board Schematic
Figure 4.2.7-2: LPC2148 Development Board
Figure 4.3.2: Chess Board Layout, Starting Positions
Figure 4.3.2.1: Pawn Movement
Figure 4.3.2.2: Knight Movement
Figure 4.3.2.3: Rook Movement
Figure 4.3.2.4: Bishop Movement
Figure 4.3.2.5: Queen Movement
Figure 4.3.2.6: King Movement
Figure 4.3.3.1: Castling
Figure 4.3.3.2: En Passant
Figure 5.1-1: Chessboard Layout with Holes for Sensors and LEDs
Figure 5.1-2: Chessboard Profile
Figure 5.1-3: Chessboard Perspective
Figure 5.2-1: Available moves shown by green LEDs
Figure 5.2-2: TLC59116EVM-390 Evaluation Module
Figure 5.2-3: TI TLC59116 Evaluation Board
Figure 5.2-4: TLC59116
Figure 5.2-5: Single 3-Driver LED Arrangement Cluster
Figure 5.2-6: Entire Conceptual LED Arrangement
Figure 5.3-1: I/O Expander Configuration
Figure 5.3-2: I/O Expander Subsystem Quarter
Figure 5.3-3: Full I/O Expander Subsystem Configuration

Figure 5.6-1: Barrel Jack to ON/OFF Switch
Figure 5.6-2: 5V Regulator Circuit
Figure 5.6-3: 3.3V Regulator Circuit
Figure 5.7-1: LCD Screen Wiring
Figure 5.7-2: Mode Buttons
Figure 5.8-1: JTAG Wiring
Figure 5.8-2: RESET Circuit
Figure 5.8-3: Crystal Oscillator for CLCK
Figure 5.8-4: Theoretical Entire Schematic
Figure 5.8-5: Main Micro Circuit Schematic
Figure 5.8-6: I/O Expander and Buttons Schematic
Figure 5.8-7: LED Driver Schematic
Figure 5.9-1: Main PCB Circuit Layout
Figure 5.9-2: I/O Expander PCB Circuit Layout

Table of Tables

Table 4.2.1.3-1: Microcontroller Comparison
Table 4.2.1.3-2: Microcontroller Acronym List
Table 4.3.1: Chess Piece Allotment
Table 5.1: Chess Board Component Dimensions
Table 5.4.2-1: Pawn Movement Algorithms
Table 5.4.2-2: Knight Movement Algorithms
Table 5.4.2-3: Bishop Movement Algorithms
Table 5.4.2-4: Rook Movement Algorithms
Table 5.4.2-5: Queen Movement Algorithms
Table 5.4.2-6: King Movement Algorithms
Table 5.5.1: Standard Valuations
Table 8.1: Senior Design 1 Project Milestones
Table 8.2: Senior Design 2 Project Milestones
Table 8.3: Budget Summary Table

1 Executive Summary

The Knight Light LED Chess Set is an interactive chessboard that can be used as a tool to learn the classical game of chess. This will be accomplished by tracking the movement of chess pieces during the game and by showing a player the possible moves they can make whenever they lift a piece off the board by lighting up all of the squares on the game board that this piece can move into. The chess set will have the functionality to allow two human players to play against one another and will include an artificially intelligent computer player for a single-player mode. When playing against the computer, the user will be shown where to place its pieces by illuminating the appropriate spot on the game board. This chessboard will include an eight-by-eight matrix of LEDs controlled by LED drivers, an eight-by-eight matrix of sensors to be detailed upon, LCDs to display text as appropriate, buttons to turn the game on and off and to control the game mode, and a microcontroller containing all appropriate chess-related rules, logic, and algorithms.

The primary motivation for the design and implementation of the Knight Light LED Chess Set is to provide a more interesting version of the classic game of chess, as well as provide a learning tool for new players. A learning tool must be user friendly, so the goal is to create something that will show a user what moves are available and take into account players changing their minds when they pick up a piece. Another motivation for this project includes taking a game that many people are fairly familiar with and put a new spin on it that has not been seen before. Chess games are readily available on computers or on mobile devices and the standard chess set is easily found in stores. Combinations of the two are not as prevalent or common. Bridging the gap between this classic board game and its digital renditions will create an intriguing product that will appeal to multiple demographics.

Research for this project comes from numerous sources. Research on the hardware side of the project includes topics such as PCB design, how to sense pieces, how to be able to control a large array of sensors or LEDs, what type of microcontroller to use, and how to power it all. Much time has also been spent looking into creating an artificially intelligent computer player and how to determine what moves to make and when. Significant contributors of information for research include technical forums, data sheets, user manuals and schematics, and documentation of similar existing projects.

2 Project Description

To compile a thorough description of this product, numerous aspects of the device's functionality and motivations behind each feature must be detailed upon.

2.1 Motivation and Goals

The motivation behind the Knight Light LED Chess Set is to create a fun learning tool for the game of chess. Chess is a complicated game and can be very intimidating for new players. However, as a classic board game, there is still a desire by many to learn how to play the game and creating something that will ease young players into the complicated moves and strategies involved in chess will be a rewarding project. LEDs are the preferred method of communicating to a user where they are allowed to move because it allows the game to avoid using chess jargon that new players may not understand. Instead of verbally telling the player or giving a text command the game board will show them visually where they can move with the piece that they pick up. With these thoughts in mind, the motivation and goals for the different features of our project are as follows.

2.1.1 Chess Piece Sensing and Tracking

The Knight Light LED Chess Set shall sense whenever a piece is lifted from the game board; this can be done through the use of hall effect sensors. The board will then show the user the moves that are available for that piece. When the player puts the piece down on the board it will recognize the new placement and determine if it is valid. A valid placement will include the original space of the piece as well as the legal moves. If the activated chess piece is moved to a valid spot, this player's turn will ends. If the board finds that the piece returned to the original position, it will remain this player's turn. If a player makes an invalid move, the board will have to realize this and tell the player to return the piece to the original location and make a valid move.

2.1.2 Display

The representation of possible moves for players will be accomplished by illuminating LEDs, these will be easily noticeable and should be a nice modernization of the classic chess game. Preferably the LEDs will be of the RGB variety so different events can be signified by the use of different colors.

Implementation could include something along the lines of red for an invalid move, green for a valid move, and blue could be used for a capturing move or a move that would put the opponent in check. Either a single LCD or two LCDs will also be used to teach players the actual chess terms that relate to the moves they are making in the game by displaying a text representation of the move they just made. An LCD or LED could also be used to show the players whose turn it is at any given time.

2.1.3 Gameboard Housing

Ideally, the housing for the hardware and actual game board should be aesthetically pleasing. The housing should keep all the components neatly contained and our goal will be to keep it from becoming too bulky. Preferably it will only be slightly larger than the regular dimensions of a chess board because of the components that will be included like the LCD display.

2.1.4 Human vs. Human Gameplay

The board will tell the players whose turn it is first and they will move. Move validity will be checked and the next player will be notified that it is their turn. While turns are going on the possible moves for a lifted piece will be illuminated and after a move is made the chess terminology for that move will be displayed.

2.1.5 Human vs. Computer Gameplay

The device will act in an almost identical manner as the human vs. human gameplay mode. The difference being that the computer player will determine what move it wants to make and will light up the piece it wants to move and where it would like to move to on the board. The human player will then move this piece and, as long as they move it to the right spot, the game will change to the human player’s turn. A possible goal for a future iteration of this project would be to automate the movement of pieces for the computer player. This would necessitate the use of motors that our group is not familiar with so we decided to avoid attempting this in the interest of being able to have a working prototype. With that in mind, for this iteration of the project, it will just be assumed that the human player can move the computer player’s pieces on their own.

2.1.6 Ease of Use

Use of the Knight Light LED Chess Set should be fairly simple. Controls and the way in which the game communicates with the players should be intuitive, so a new user can just turn the board on and understand how it works without a long, drawn out explanation. It should be as simple as turning on the board and selecting a single-player or multiplayer game, from there the board would show the player how to set up their pieces. Gameplay commences and the board walks the players through the game by showing the computer's moves or showing where the player can move any piece that they pick up.

2.1.7 Chess Computer Player

The backbone of human vs computer gameplay will, of course, be that an artificially intelligent computer program will have to be created. This implementation of the feature will ultimately be dependant on how long our programming process for the general gameplay is. The amount of time left once the computer player's logic begins development will determine the intricacy level of the design. Basic plans for different iteration designs will be created during the first semester of the project and once development has been started, which iteration is implemented will be selected.

2.2 Objectives

In the process of designing the Knight Light LED Chess Set a set of objectives that the board will have to accomplish has been created. The objectives are more specific than project goals and will be key stepping stones in the actual development of the chess set. The objectives are as follows.

Piece Sensing and Tracking- the movement of game pieces must be sensed by a matrix of reflective object sensors. The board must recognize when a piece is lifted and where it is placed. There will have to be a constant periodic scan of the board to locate all of the pieces. When a piece is lifted, the previous location must be remembered until the piece is placed in a valid spot, otherwise the integrity of the game will be lost.

2.2.1 LED Display

The chess board shall contain an eight-by-eight LED matrix that will illuminate the spaces players can move their pieces into. Whenever the active player lifts a piece, the LEDs beneath the tiles they can move into will light up green. The LEDs will also be used to show players how to set up the chessboard, this can be done by lighting them in a pattern: pawns, rooks, knights, bishops, queen, and then the king. The display will also be used to show a player when they have made an invalid move and where to move the piece back to by using a red LED to signify a wrong move.

2.2.2 LCD Display

An LCD or multiple LCDs will be used to tell the player what move they just made in chess terminology. The LCD will display text when a user makes a valid move that tells them the chess terms for the move they made. It will also display who won the game whenever the end of game situation is reached. This will act as a learning tool for new players to gain a mastery of the chess language. If time permits another LCD can be added to track the time it takes for a player to make a move.

2.2.3 Human vs. Human Gameplay

When two players initiate a game the board will show them how to line up their pieces and then tell them which player's turn it is. When a player picks up their piece it will then tell them what moves they can make and, if they make an appropriate move, will display the associated text with the move and signal for the next player's turn. If a piece is returned to it's original spot then the board must know that the turn has not ended yet. An invalid placement will be addressed by notifying the player and instructing them to return the piece to the previous spot and try again. When a piece is placed on a valid spot the appropriate description for the move will be displayed on the LCD screen. Gameplay will continue until the board finds that one of the victory conditions has been satisfied and will notify the appropriate player that they have won.

2.2.4 Human vs. Computer Gameplay

When a player initiates the single-player mode on the device, game the board will show them how to line up the pieces and then display if it is the player or computer player's turn. If it is the computer player's turn the computer will decide what move to make and signal to the board to light up the appropriate piece and destination for it. The player must move the piece for the computer and the placement will then be evaluated. Just like a player move, a valid placement will result in a turn change and an invalid placement will be addressed by returning the piece and showing the correct placement again. As in human vs. human gameplay, when a player lifts their piece they will see the available moves and then decide where to place to piece. Valid and invalid placements will be handled the same way and gameplay will continue like this until the board finds that win conditions have been satisfied. Whenever a piece is placed on a valid spot the appropriate description for the move will be displayed on the LCD screen. Once end of game conditions have been met the LCD will display who won the game.

2.2.5 Chess Computer

With the group's main focus being the general gameplay between two human players, the computer player's logic will initially be on the back-burner when development is started. However, research will be done and multiple possible approaches to the computer player will be planned. One can be selected to fit the amount of time available once regular gameplay is finished. If the due date is coming near, a simple brute force method of move selection can be implemented. This could include location ratings on the board to determine what spaces are most valuable. The more ideal situation would be to have enough time to attempt to implement a computer player built around a minimax algorithm that will minimize any possible losses and maximize the minimum gains made for any given move. The obvious preference is to implement a minimax algorithm, but ultimately this will come down to how much time is left for development.

3 Requirements

Once objectives and goals were established, hardware and software requirements had to be introduced as project research progressed. These requirements help to facilitate the research and organize accumulated thoughts and ideas for the design process.

3.1 Software Requirements

Software requirements were built around the needs of the project as the vision of a final product was discovered. These software requirements are meant to be comprehensive, but not overly extensive. These requirements describe what features must be included in the software application. These requirements also describe how the software will interact with the hardware to form a seamless bond between the two to create a finished product.

- The software must be able to interact with the hardware.
    - The software must take input from sensors on the chessboard to detect piece movement.
    - The software must provide output through the hardware to light up LEDs on the chessboard.
- The software must follow a chess game as it is played by keeping track of chess piece positions, movements, and statuses at all times.

- The software must know all the rules of chess.
- The software must provide an artificially intelligent program that contains functions and methods to simulate a real human opponent during a one-player mode.
- The software must suggest available chess moves to both players at the start of each turn.
- The software must recognize unique endgame situations such as a “check,” “checkmate,” and a “stalemate.”
- The software must recognize when special moves are available to players such as, “castling,” “en passant,” and “pawn promotion.”

3.2 Hardware Requirements

Hardware requirements are for reference as the project is developed and designed, and for guidance toward the end goal of this project. The following requirements and specifications have been decided upon. The requirements are meant to be comprehensive, but not overly extensive. Ideally, all of the requirements ahead will be achieved throughout the development process. However, as the project progresses alterations may have to be made.

- The microcontroller must have a sufficient amount of memory to handle what could be somewhat extensive software.
- The microcontroller must have SPI and I2C connections available to handle a grid of sixty-four LEDs and sixty-four Sensors.
- The microcontroller must be programmable via JTAG.
- The LEDs must be controlled by LED Drivers.
- I/O expanders must be used to interface the microcontroller with sensors
- The microcontroller must control an eight-by-eight grid of sixty-four LEDs to indicate available chess moves, among other things.
- The microcontroller must control two small LCD displays to convey game information to each player.
- The microcontroller must receive input from push buttons that will be used to start the game and switch between one-player and two-player modes.

4 Research

In order to begin moving forward with the concept of the Knight Light LED Chess set and start designing the actual product, multiple dimensions of research must be done.

4.1 Similar Existing Products

Chess is a classic game and one that has seen numerous different variations since its initial creation. In recent years, there have been multiple different electronic versions of chess that offer different features. Two well-known products in the digital chess community were identified. These products offer some interesting features and have good reviews from chess players worldwide.

4.1.1 Mephisto Chess Trainer

The first product that was researched was the Mephisto Chess Trainer, by Saitek. Shown in Figure 4.1.1-1 below, it is a complete chess set and a chess computer with a modest Elo rating of 1050. The board offers features like extended teach modes that help beginners learn the basics of the game by concentrating on the tactics for individual pieces. It also offers a hint feature, the ability to save your game, and an LCD display that shows animated face, chess moves, times, and a chess clock.

The Mephisto Chess Trainer also offers sound effects and the difficulty is adjustable with 768 level setting combinations, allowing difficulties suitable for players ranging from beginners to club players. While this product has some intriguing features and sounds like it truly can be a boon for new players, it seems that it may be a bit difficult for a new chess player to get a hang of quickly. While it will give the player hints, the board does not show a player exactly what moves they can make when they select a piece and it doesn't have the added fun of LEDs throughout the board.

Figure 4.1.1
Saitek's Mephisto Chess Trainer
Reprinted with permission from Saitek.

4.1.2 Excalibur Grandmaster Chess Computer

Another well known chess computer is the Excalibur Grandmaster Chess Computer, shown in Figure 4.1.2-1 below. Made by Chess Baron, Excalibur offers a strong chess program that can defeat over 95% of chess players with a master-strength Elo of nearly 2200. The board offers two LCD displays with clocks, countdown timers, hints, and warnings. It also will score one hundred different levels of play, has five teach modes, and contains thirty-two different strategic openings. Excalibur also offers one-player and two-player modes and has a save game feature. Similar to the Mephisto Chess Trainer, Chess Baron's Excalibur offers some nice learning features, but they assume the player knows the basics of chess. The different levels of difficulty are also a nice feature to have in a chess computer, so you can keep bumping it up as you get better.

Figure 4.1.2
Excalibur Grandmaster Chess Computer
Reprinted with permission from ChessBaron.

4.1.3 Product Comparison Summary

There are certainly some very nice electronic chess boards available on the market, including boards that have what seem to be some very solid learning tools built into them.

The Knight Light LED Chess Set's niche comes more along the lines of catering to players that are largely new to the game of chess by walking them through the game step-by-step while also catering to experienced chess players by showing a visual representation of available chess movements on the board.

Both the Mephisto Chess Trainer and the Excalibur Grandmaster Chess Computer offer similar LCD displays and a project goal is to have a similar feature built into the Knight Light LED Chess Set. The use of LEDs to illuminate spaces on the chessboard during gameplay also differentiates the Knight Light LED Chess Set from the Mephisto Chess Trainer and Excalibur Grandmaster Chess Computer.

4.2 Hardware

As a vision of the Knight Light LED Chess Set develops, hardware becomes the backbone of the project. Hardware must be be able to interact with software appropriately to provide the user with a game of chess.

4.2.1 Microcontrollers

Devices that interact with a user contain a microcontroller which controls the functionality of that particular product, such is the case with an LED / LCD chess board. All microcontrollers are computers. Computers contain certain characteristics that are common with microcontrollers such as containing a CPU (Central Processing Unit), RAM (Random Access Memory), ROM (Read Only Memory) and input / output devices. However, there are distinct characteristics that differentiate a microcontroller from a basic desktop computer and need to be considered when choosing the right microcontroller for the electronic chess board.

Microcontrollers are specialized units that often do only one task. They handle only one program, such as running a chess game, unlike many computers that run hundreds of programs stored in ROM. The electronic chess board will typically run only one task at a time. Another unique trait that microcontrollers possess is that they are often times embedded inside of a product, which is why they are commonly referred to as “embedded controllers.” The microcontroller used for this chess board will be embedded inside of the chess board housing.

Microcontrollers typically consume less power than full computers, which can sometimes use up to fifty watts of electricity. A microcontroller may only consume one one-hundredth of that power, which allows them to be both cost efficient and energy efficient, both of which are vital to this project. One of the goals of the electronic chess board is to maximize efficiency and reduce cost.

Microcontrollers require a designated input signal that will produce a desired output pertaining to the particular device. With the chess board system, it will incorporate an input signal from the players playing chess and moving pieces around the board. The microcontroller will interpret this information using custom-designed software and will produce an output signal to both LED lights and an LCD screen to display available chess moves and messages to the players. Signals will also be used to relay information to the software and allow the software to execute the appropriate code during game play.

4.2.1.1 Microcontroller Specifications

Microcontrollers today come in many sizes, different functionalities, and varying specifications. They can range in properties such as clock speed, core size, operating voltage, memory size, and many others. For basic microcontrollers, 8-bit, 16-bit, and 32-bit cores are common with up to a 200 MHz clock speed. The operating voltages can also range from 1.8V to 5.5V with operating temperatures ranging from -40C to 125C degrees.

An important characteristic to consider will be the memory size. It is absolutely critical that there is enough memory to handle the specific chess-related tasks. In addition to the other specifications, the ability to control LCDs, integrate LED controllers / drivers to show output for possible moves, and provide messages will be paramount to the successful development of this product.

Each of these factors will play a role in the computation of final cost. Getting the correct specifications is important to ensure the final design is affordable and maintains high levels of performance.

4.2.1.2 Programming Languages

When developing a product, the different possible programming languages will need to be considered, as it is the backbone to the logic of the entire system. There are many different options for languages, each with multiple strengths and weaknesses to be considered for project development.

The first choice for programming a microcontroller would be an object-oriented language such as C++ or Java. However, object oriented languages like Java aren’t supported by microcontrollers that are commonly used, so research must be done in order to determine which language will best suit the needs of this product.

Popular microcontroller languages include Assembly, BASIC, C, and Pascal. Evaluating each language is vital for the efficient programming of the microcontroller. A large amount of the project development will be programming the microcontroller along with creating chess algorithms, so it is crucial that the right decision is made.

4.2.1.2.1 Assembly

Assembly, which will likely be provided by the manufacturer is a very obvious choice for small programs but will become an issue when writing larger programs that include chess move algorithms. Although response time with assembly is usually very fast, it might not be the correct fit when creating large programs such as the more intricate software of the Knight Light LED Chess Set.

4.2.1.2.2 BASIC

BASIC is an easy language to learn and use. It contains many built-in functions and a BASIC compiler can produce code that will run as fast as a C compiler. BASIC is also known to have some disadvantages. Disadvantages include the fact that BASIC is not a standardized language and there aren’t many offered compilers to code for microcontrollers.

4.2.1.2.3 Pascal

Pascal, much like BASIC, is an easy language to learn if you have some programming experience and it also contains many built-in functions. In comparison to C, it tends to be a bit lengthy and will take up more memory than C. It is also not as flexible of a language as C, which leads to what may be the best and most obvious choice.

4.2.1.2.4 The C Programming Language

C, a standardized language, which runs fast, has many compilers available, and has a vast amount of built-in functions, is very popular in the software development world. It is the foundation of countless modern programming languages. It does not have many disadvantages. C typically isn’t the easiest language to learn, so there can be a rather steep learning curve. This is easily addressed since all group members have experience with C prior to beginning this project. C is also very picky about the correctness of its code, which can be seen as an advantage by forcing developers to focus and produce better code. C seems to be the best choice, however experience may prove that another language could be more beneficial.

4.2.1.3 Microcontroller Product Research

After initial research and addressing some important characteristics, it is crucial to compare microcontrollers out on the market and discuss the differences. Microcontrollers that are in comparison are the NXP LPC2148, and the Atmel Atmega128. Both are found in the same range of pricing with different levels of functionality and varying specifications to take into consideration. The most pertinent information is displayed below in a table comparing the two microcontrollers.

By comparing each microcontroller's specifications side-by-side there is a clear view of which will be the best fit for the given budget and functionality requirements necessary for a successful chess board microcontroller. Below, is the comparison of the two microcontrollers and an in-depth look at how each of them would be beneficial to the Knight Light LED Chess Set design.

Table 4.2.1.3-1
Microcontroller Comparison

Specification	**LPC2148**	**Atmega128**
Flash	512kB	128kB
Max. Operating Frequency	60MHz	12Mhz
CPU	32bit	8bit
USB Interface	Yes	No
USB Speed	2.0	N/A
Self Programmable	Yes	Yes
Timers	2	4
GPIO	45	53
PWM Channels	Yes	Yes
Temperature Range	-40 to 85 C	-40 to 85 C
I2C Serial Interfaces	2	1
ADC Channels	14	8
Watchdog Timer	Yes	No
Real Time Clock	Yes	No
Oscillator	Yes	Yes
UARTs	2	2
Power	3.3V - 5.5V	2.7V - 5.5V

RAM	32 kB + 8 kB USB	4 kB
SPI	2	1
Debug Software	Yes	Yes
DAC Channels	Yes	No
BOD	Yes	No
Architecture	RISC	RISC

Table 4.2.1.3-2
Microcontroller Acronym List

Acronym	**Description**
ADC	Analog to Digital Converter
BOD	Brown Out Detection
CISC	Complex Instruction Set Computer
CPU	Central Processing Unit
DAC	Digital to Analog Converter
GPIO	General Purpose I/O
I/O	Input/Output
I2C	Inter-Integrated Circuit
LCD	Liquid Crystal Display
LED	Light-Emitting Diode
PWM	Pulse Width Modulator
RAM	Random Access Memory
RISC	Reduced Instruction Set Computer
SPI	Serial Peripheral Device
UART	Universal Asynchronous Receiver/Transmitter
USB	Universal Serial Bus

When deciding which microcontroller is the most suitable for the project, each specification should be analyzed. It is important to know the functionality of each feature and which ones are more beneficial for the purpose of creating a LED / LCD electronic chess board. The microcontroller that stands out to be most beneficial for this project is the LPC2148. It will provide maximum performance while still staying affordable and within the budget.

4.2.1.3.1 CPU and Architectural Design

When considering a microcontroller, one of the first features to consider is the size of the CPU. Comparing different embedded controllers, the ideal microcontroller will be relatively fast so that the computing doesn't take a large amount of time and is also cost efficient. A 32-bit CPU will offer high speeds and fast calculations. In addition, larger amounts of memory will be readily available for the program to use in comparison to an 8-bit CPU. Typically a 32-bit CPU will be more expensive than an 8-bit, however the difference is not large enough to compromise the efficiency and fast computing. It is important that the chess board responds quickly to the players moving pieces. It won't be beneficial if the players are standing around waiting for the board to react to their moves. Secondly, the architecture of the system will play an important role when choosing the best microcontroller. There are two choices, either RISC or CISC, when considering the architecture of the system. The RISC instruction set and related decode mechanism are much simpler than that of CISC. The simplicity of RISC will result in a high instruction throughput and unmatched real-time interrupt response. Thus, a RISC structure will be more beneficial to the final product, the chess board.

4.2.1.3.2 Flash Memory and RAM

Flash memory is used for the storage of code as well as data from the system. Not knowing the exact size of the necessary chess algorithms, it will be important to obtain as much possible memory when programming the embedded controller. Flash memory, offers the ability to erase any existing errors that may occur when developing the program. Many microcontrollers enable the ability to alter code hundreds of thousands of time in order to ensure successful development of the algorithm as well as deleting any unwanted saved information that might be on the system's memory. 512 kB is the largest on-chip available size for microcontrollers, however there are many options when considering the correct part starting as low as 32 kB. In addition to Flash, microcontrollers also contain RAM in which code and data storage is available. Additional RAM may be provided when parts offer USB drives. Like Flash, RAM can be found in varying sizes varying from 8 bit up to 32 bit. The larger the RAM, the more room there will be available for both code and gaming information storage. In the case of the

LPC2148 the additional USB port will allow an extra 8 bit space for storage.

4.2.1.3.3 Timers

Timers will allow a simple way to show relevant information to the game of chess. When a user would like to time each move whether by them or the other user, the timer on the microcontroller will allow for this functionality. It could also play a role if the user wants to time the length of the entire game. Both are functionalities that many serious chess players utilize. Timing could be an option to be implemented of the final design of the chess board.

4.2.1.3.4 SPI Serial I/O Controller

The SPI is a full duplex serial interface, designed to handle multiple masters and slaves connected to a given bus. The SPI plays a significant role in the world of embedded controllers. They have various applications in which the chessboard can benefit. SPI's are used to talk to various peripherals such as sensors including temperature, pressure and touch sensors, camera lenses, ethernet communications and even LCD displays. With the intention of providing the user with both the ability to plan out moves with the illumination of LED lights provided by detection of sensors or displaying information to the user such as time and strategy, SPI is an important feature that the selected microcontroller must have for successful development. The SPI interface will be a good option for LCD control and gathering input data from sensors.

4.2.1.3.5 I2C(Inter-Integrated Circuit)-bus serial I/O Controller

The I2C will allow the ease of bi-directional communication between components that exist on the same board. Applications include the altering of volume of in speakers, reading diagnostic sensors and controlling LED displays. I2C will provide the ability to add sound to the chess board if decided upon and be a major factor in controlling the LED display. I2C can also be used in conjunction with I/O Expanders to read in data from sensors like the SPI feature. This is an extremely important feature to have in the selected microcontroller and having two I2C ports could be very beneficial to any project.

4.2.2 Sensors

An array of sixty-four sensors will be placed directly beneath the gameboard in order to sense chess piece positions and moves.

4.2.2.1 Hall-Effect Sensors

A hall effect sensor is a device based on the hall effect, which is when a current is passed through a conductor where a magnetic field is present, perpendicular to current flow and a voltage, called Hall Voltage, is generated perpendicular to both the magnetic field and the flowing current. Hall effect sensors are devices which can detect when there is a magnetic field present. When there is no magnetic field present, the hall voltage is zero since the is no magnetic field applied to the current on the semiconductor. When there is a magnetic field applied to the current, the device creates a hall voltage perpendicular to both the magnetic field and the current. The hall voltage is typically small and needs a source of amplification in order to produce the desired functionality.

Figure 4.2.2.1, below, is an image of a standard hall effect sensor from Sparkfun.

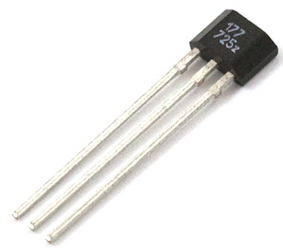

Figure 4.2.2.1
Hall Effect Sensor
Reprinted with permission from Sparkfun.

When considering hall effect sensors it is important to understand common applications that they are used for and determine if they would be beneficial for sensing the movement of chess pieces. Applications include current sensing, power sensing, speed detection and most importantly is the application of proximity detection. Proximity detection could be useful when playing chess for the detection of the pieces. The hall sensors could help locate where individual pieces are located in addition to where they have moved.

Advantages of hall effect sensors include a long life cycle since there is no wear and tear due to friction between the semiconductor and the magnetic field. Hall effect sensors also work under a wide range of temperatures as well as being immune to common external deteriorating factors such as dust, air and water. A notable feature is the hall effect sensor’s ability to operate at high speeds. Operating at high speeds such as over 100 kHz in other sensors such as capacitive sensors tend to distort the output. They are cost effective and generally small in size.

Disadvantages of hall effect sensors include interference from external magnetic fields, a large temperature drift and a large offset voltage. Unfortunately, latching can cause issues with hall effect sensors. This would make it necessary for both north and south poles of magnets to be used throughout the game; just moving a magnet away would not reset the output value of the sensor. While hall effect sensors have some great advantages, the latching issue may be something that disqualifies this component from being used.

4.2.2.2 Reed Switches

Reed switches are electromechanical devices having two magnetic reeds that are hermetically sealed in a glass tube. The introduction of a magnetic field will close the reed switch creating a function of switching. There are two states of a reed switch open and close. When the reed switch is open, there is zero current flowing in the device which makes them energy efficient. Reed switches can be used for various applications including reed relay and as a reed sensor. When a reed switch is used as a reed sensor, it can detect movement when there is a permanent magnet being used. With an applied magnetic field, there should be a separation between the magnet and the reed switch. Reed sensors can also detect counting, liquids and many other measurements. Applications for the Knight Light LED Chess Set concern the detection of movement. This detection could allow the program to track the movement of the pieces and current location due to the state of the reed switch under the presence of having a magnet or the absence of one.

Figure 4.2.2.2, below, is a simple diagram of what a reed switch looks like and where the different components of it are.

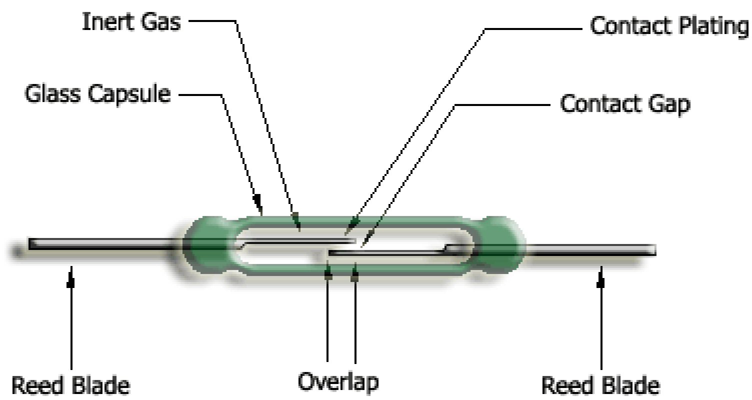

Figure 4.2.2.2
Reed Switch

Advantages of using a reed switch include that they are sealed within a glass envelope which eliminates contamination from outside factors and are safe to use in varying environments. They are immune to electrostatic discharge and do not need external amplification to produce desired output unlike hall effect sensors which need external amplification. They are also cost effective and small in size.

Disadvantages of using a reed switch include interference from the presence of an external magnetic field, similar to that of hall effect sensors which throw off the device. Since they are small in size they are very delicate and will not stand for large currents or voltages which can cause the device to spark. The glass is easy to break when soldering and incorporating reed switches will take a steady hand due to their small size but are still an effective option for the detection of chess piece movement.

4.2.2.3 Photocells

A third option to detect the movement of pieces would be light detecting photocells. Foregoing to process of purchasing magnets and attaching them to pieces, mini photocells could be placed in each square on the game board. Photocells are small, quick, and reliable. Depending on the amount of light being received, the resistance of the cell changes. They are also fairly cheap and not overly intricate.

Figure 4.2.2.3-1, below, is an image of a simple photocell.

Figure 4.2.2.3
Photocell
Reprinted with permission from Sparkfun.

The main disadvantage of photocells is that they do not output a specific digital or analog value, but alter their actual resistance. To identify movement of pieces the microcontroller would have to identify changing voltage/current coming from the wired photocells. While inexpensive, there are some cheaper options than photocells.

4.2.2.4 Optical Detectors and Phototransistors

The QRD1114 is an optical detector/phototransistor that emits an infrared signal via a diode and then detects the reflected signal through an infrared phototransistor. It features a daylight filter on the sensor to help keep daylight from resulting in false positives. Digital output signals vary depending on the amount of light sensed by the QRD1114. The range is generally 0-3 cm, with an optimal range of .5 to 1 cm. This could be extremely useful by finding a general output reading from indoor light or sunlight and setting that as being the upper range of values signalling no piece covering the sensor.

Figure 4.2.2.4, below, is a photo of the QRD1114, where it can be seen that the two major components (detector and emitter) are packaged together in one compact piece.

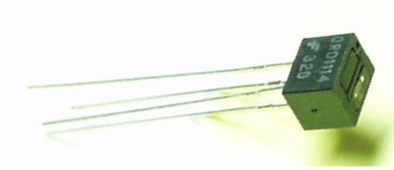

Figure 4.2.2.4
QRD1114 Phototransistor
Reprinted with permission from Sparkfun.

The price range for this component falls between a photocell and a hall effect sensor. The component is also rather small, making it easy to fit into the spaces that it needs to fit in. With an easy to read variable output, small size, and a fairly low price, the QRD1114 seems to be a great option for the Knight Light LED Chess Set.

4.2.3 LEDs and LCDs

The Knight Light LED Chess Set will communicate to the users through a series of LED lights and LCD displays. The manner in which these components work together makes the Knight Light LED Chess Set a unique product.

4.2.3.1 Liquid Crystal Displays (LCDs)

LCDs consists of extremely small molecules that are present between a piece of glass and an opaque substrate that produce a physical output visible to the chess user based on the electric charge. When electric charge and light is present, there will be an image displayed to the screen and when there is no electric charge the molecules will be transparent thus producing a blank portion of the screen to the individual user. There are many different types of LCDs to consider when creating a chess board. The main goal is to choose the correct part that will be power efficient, relatively cheap compared to other parts and within the realm of producing the desired results of the chess game such as time and possible moves.

4.2.3.1.1 Blue Mode

The most basic LCD screen is the Blue Mode STN. Although cheap and effective, a disadvantage would be the contrast ratio making it harder for the user to see from varying angles. For the chess board, it is ideal to have a system that isn't hard to read and visible from all angles.

4.2.3.1.2 Film Mode

Basic LCD screen that is relatively inexpensive and effective for our purposes of displaying text and numbers. It is improved from the Blue Mode simply by adding additional film to increase the ratio of contrast. This will allow the chess player to see the LCD from more angles and increase the appearance of the LCD images. However if we want color for the display this type of LCD will not produce colors. A Film STN LCD will keep the budget of the project relatively low while being power efficient but will only be in black and white.

4.2.3.1.3 Color and Double

The last two options are very similar products since the both produce colorful display with up to 65,000 different colors. The difference between Color and Double is that Color STN's will not provide as much contrast as the Double STN. It will be possible to consider on of these products for a chess set LCD however finding one for a low price will be a difficult task. For the chess set, the choice that needs to be made is whether the project will contain a colorful display or a black and white display.

4.2.3.1.4 Segment vs. Graphical

Another property that LCDs contain is the type of data that will be displayed to the screen. When choosing an LCD whether Blue, Film or Color, there are different types of data displays that are taken into consideration. The Segment LCD is typically described as old-fashioned in which only a set of characters can be displayed due to the fixed position of the molecules in the LCD. A Graphical LCD is controlled by segments of pixels that are found in rows and columns. They have the ability to display any character by energizing those pixels from the electric current. Determining which product to use will depend on pricing, ease of coding and availability of product since Segment LCDs are out-dated it might be easier to find a Graphical LCD for the consideration of the chess board.

4.2.3.1.5 Reflective vs. Transmissive

There are two main viewing modes for an LCD screen that correspond to the visibility of either outdoors or indoors or both. Reflective mode uses ambient lighting which makes this product much more effective for viewing outdoors. this is great for users that play outdoors but chess is typically a game played indoors, it is best that other modes are considered before choosing Reflective mode as the best fit. Transmissive LCD screens will often depend on an internal light source and is usually better for indoor games than outdoors due to the light source. As stated before, chess is typically an indoor game however there is a third choice that combines the two modes for use of the chess board in both indoor and outdoor scenarios. With a flip of a switch, the user will have control over the chessboard LCD and viewing can be made for both the indoor user and outdoor user. The idea that the chess user can use this product both indoors and outdoors is beneficial to all parties.

4.2.3.1.6 Additional LCD Specifications

Other properties that need to be taken into consideration are the resolution, dot pitch, size of screen, time of response, active temperature ranges, brightness and refresh rate when choosing the correct chess board LCD.

4.2.3.2 LEDs

LEDs today come in many different shapes, sizes and types and can function in many systems such as a chess board. When considering LEDs it is important to understand the purpose at hand. The main use of the LEDs on the chessboard will be to illuminate the chess board with possible moves to consider when playing depending on the specific piece. However, since LEDs do come in alphanumeric form, they can also be considered for display of numbers and text.

4.2.3.2.1 Displaying Information

Much like an LCD screen, LEDs can also be used to display text and numbers. When considering the alphanumeric display there are four subtypes that can be used which are the 7, 14, 16 and matrix segment. 7-segment display is used for only numbers, so that is not an option since it will important to have text as well. The 14- and 16-segment displays offer both text and number display along with the matrix display however only the matrix display offers both upper and lower case letters. The matrix display is the best fit out of these four choices but when compared to LCD, LEDs fall short. Much of the LEDs popularity has dropped due to the high sophistication of the LCD screen in comparison. Even though LCDs need backlight to be seen indoors, LCDs consume less power and generate less heat than LEDs. They also save on space, weigh less and do not flicker thus making LCD display a much better choice.

4.2.3.2.2 Bi-Color and Tri-Color

LEDs are made of light-emitting dies with a common anode or cathode all in a single casting that display the respective number of colors to the individual part. Taking into consideration the functionality and drawbacks of each of these parts will help to decide which is better for the purpose of illuminating a chess board.

Bi-Color allow only one color to illuminate at one given time due to the wiring of the part and only produce two colors. Unlike the bi-color, the tri-color will allow two colors to illuminate at once, thus producing the third color. The tri-color seems to make the most sense when developing a chess board that illuminates the moves of each piece. The three colors: red, blue and green will each signify their own purpose on the chessboard. Blue and green will be used for each player to differentiate the moves for player one and player two. The red LED light will be used for a move that either takes a piece from the opposing team or puts the king into check mate. There might be more functionality of the lights but for now the RGB, tri-color LED seems to be the best fit for the purpose of illuminating a chess board.

4.2.4 LED Drivers

LED drivers are an essential part to control the amount of current and voltage supplied to the LED which will be used for our chess board. Many different types of LED drivers are in consideration when developing a chess board which can include white LED drivers, PWM LED drivers, RGB LED drivers, constant current LED drivers and segment LED drivers. Since we are considering using RGB LEDs it only makes sense to use RGB LED drivers. RGB LED drivers are controlled by programming the SPI interface and internal registers to control the functionality. Sometimes it is common for RGB LED driver to contain a PWM LED driver which will allow the lights to be dimmed.

4.2.4.1 Types of LED Power Converters

Type 1: AC Voltage to DC Current
Advantage: Maximized energy efficiency
Disadvantage: Limited configurations of LEDs
Application: Good choice for fixed/constant number of LEDs
Output: Constant Current

Type 2: AC Voltage to DC Voltage
Advantage: Maximized LED configurations
Disadvantage: Extra components are required.
Application: Good choice for varying number of LEDs
Output: Constant Voltage

Since there will be a fixed number of LEDs, it is best to find a part that utilizes the Type 1 configuration which is a power source of AC Voltage to DC Current. TI offers the TLC59116, a 16-Channel I2C-Bus Constant Current LED driver.The

TLC59116 can take advantage of one of the high speed I2C buses and with four address pins, up to 14 can be connected on a single I2C bus. This could maximize the amount of LEDs that can be driven with a single output. It can be run at either 3.3v or 5v, making it flexible and easily adaptable to nearly any project using LEDs. With a constant output current that is simply controlled by an external resistor, the TLC59116 makes it easy to ensure the appropriate amount of current is making it to the LEDs.

Figure 4.2.4.1-1, below, shows a block diagram of the TLC59116. Below that, Figure 4.2.4.1-2 shows the two available package views.

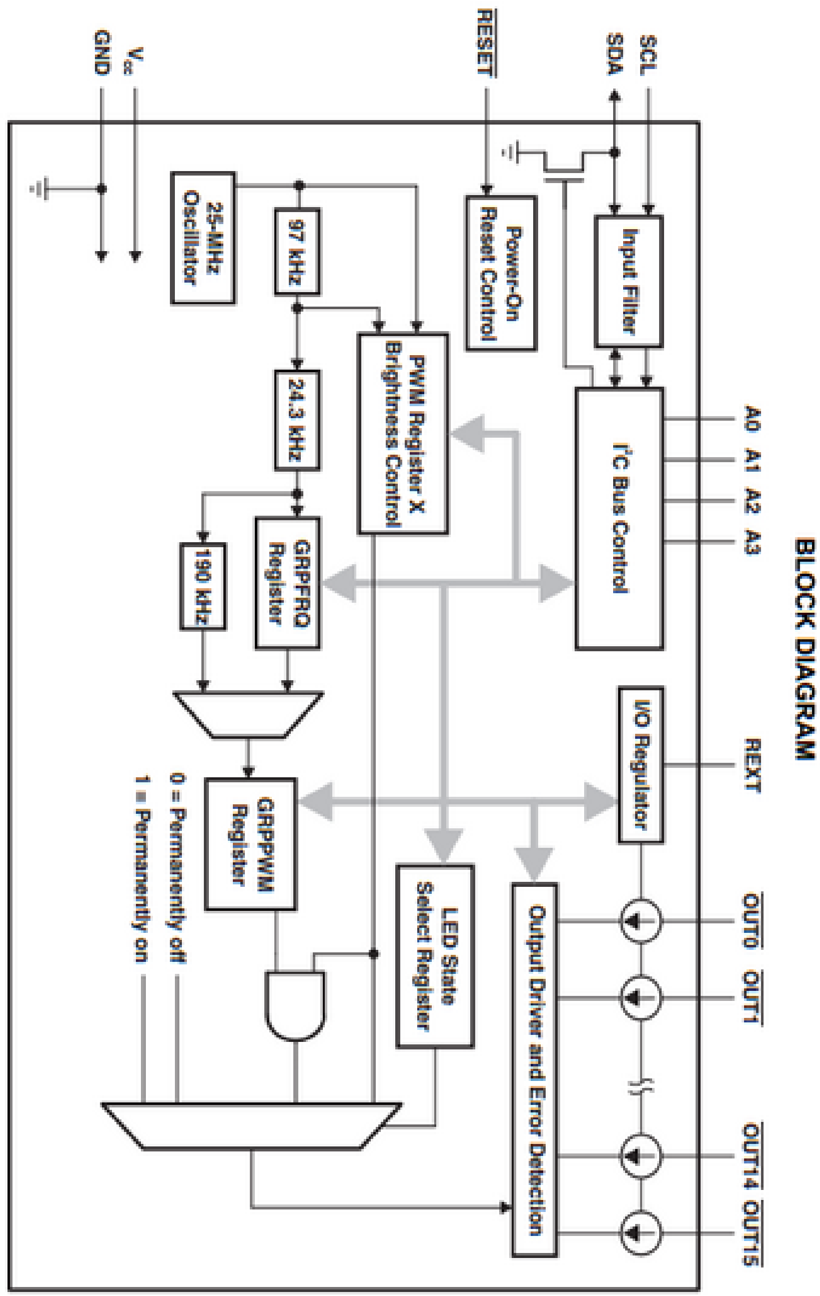

Figure 4.2.4.1-1
TLC59116
Reprinted with permission from Texas Instruments.

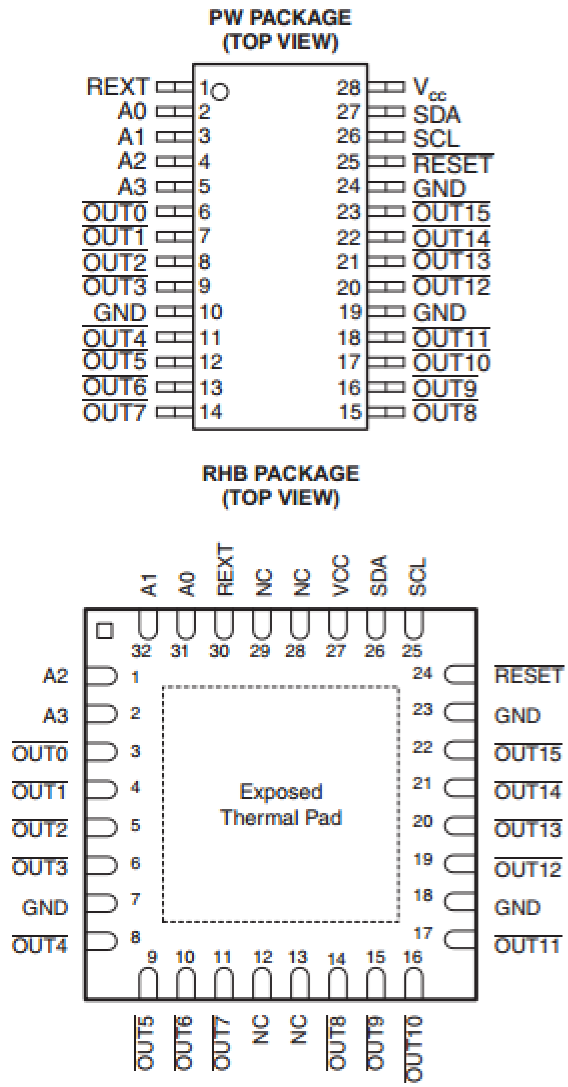

Figure 4.2.4.1-2
TLC59116 Available Package Views
Reprinted with permission from Texas Instruments.

The RHB package is the surface mounted package that would be used with a PCB. However it would be useful to purchase a few of the PW packages to test out before the PCB is received and all the components are added.

4.2.5 Power Supply

How components are going to be powered is a factor that must be carefully considered. Different components have different operating voltages and necessary input current. Most of the components that have been researched so far have operating voltages that include 3.3V and 5V, both of which are standard measurements for these types of components. Regardless of how power is being obtained, it needs to be regulated so there is a constant voltage and current coming from the power source. If the power source is not regulated there can be spikes in the amount of voltage coming from it that can damage circuit components. CUI INC offers a line of Switching Regulators that output 2.5, 3.3, 5, or 6.5 volts. Sparkfun also offers simple voltage regulators with varying output voltages. These can be used to provide a steady voltage and current to all of the components of the Knight Light LED Chess Set and should be fairly easy to utilize with only Vin, Vout, and GND pins.

Figure 4.2.5-1, below, is a simple example application circuit using a voltage regulator.

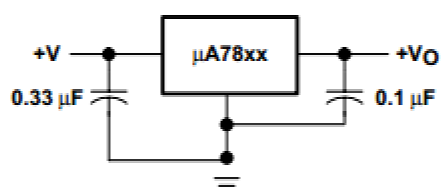

Figure 4.2.5-1
Fixed-output Generator
Reprinted with permission from Sparkfun.

When it comes to the power source, there are multiple different options that can be explored. Solar power is always enticing because it is free and easily available during the day, however the chess set would be better suited to be used inside, eliminating this option. Batteries are the next option and are very reliable. Batteries are easily obtainable and many companies offer convenient cases that will hold 2, 4, 8, or more batteries and allow them to be wired to a project. Batteries can also rechargeable, which is convenient. However, chess can be a lengthy game, and it would be wise to avoid a power supply that could run out mid game, leaving the users to have to get a new set of batteries.

Wall warts are cheap, plentiful, and can be found in many different varieties. Wall warts are helpful because they are removed from the equipment getting the electrical current helping to prevent overheating. Standard wall warts work with a barrel power jack that can then be wired to voltage regulator circuits. They can also be found offering various different voltages and currents, offering many different options when looking for how to get power to a project. USB wall warts offer multiple different voltages and currents just like the standard ones, but instead of using a barrel power jack they use a USB input. This can be very handy when a project is already using a USB input for programming.

Shown below in Figures 4.2.5-2 and 4.2.5-3, are the different types of wall warts and the connectors necessary for their use in a project.

Figure 4.2.5-2
Generic Wall Wart
Reprinted with permission from Sparkfun.

Figure 4.2.5-3
Generic Wall Wart with a USB Port
Reprinted with permission from Sparkfun.

Below, Figures 4.2.5-4 and 4.2.5-5 show both input connectors that would be used with the power and USB inputs from above.

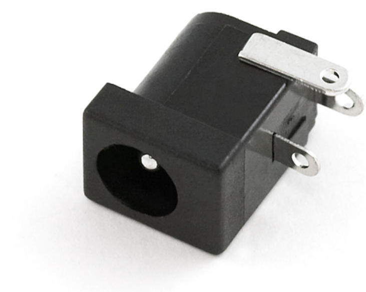

Figure 4.2.5-4
Barrel Jack Input Connector
Reprinted with permission from Sparkfun.

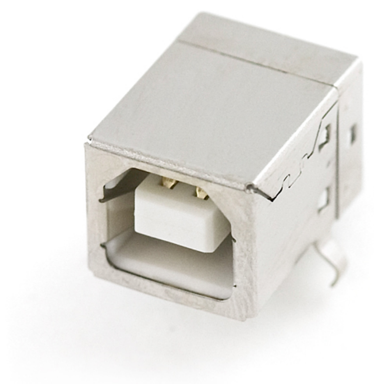

Figure 4.2.5-5
USB-B Input Connector
Reprinted with permission from Sparkfun.

Either connector can be simply wired up to any board to provide a power source. The source should then be regulated to ensure the proper constant voltage is received.

4.2.6 Shift Registers and I/O Expanders

The Knight Light LED Chess Set will be using an array of sixty-four RGB LEDs. In order to control and drive each of these individual lights, a total of 192 lights, shift registers and I/O expanders can be used to feed all of this data to the microcontroller.

4.2.6.1 Shift Registers

Shift registers can be a great way to increase the amount of inputs of a microcontroller. A Parallel-In / Serial-Out (PISO) shift register works by reading in multiple values and then packaging them together in a serial format. A simple 8-bit Parallel-In/Serial-Out shift register can take in 8 parallel lines and output a single serial output, this can then be easily sent to a gpio pin of a microcontroller to use the data for whatever necessary purpose. 8-bit shift registers can easily be found in the $1.00 - $2.00 range on numerous websites, making them affordable and easily accessible.

The SN74HC165N is available for $1.45 and offers the following features: a wide voltage range of 2V to 6V, Parallel-to-Serial data conversion, low power consumption, as well as others. This component looks like it could be a good fit for the needs of the Knight Light LED Chess Set. Without much in the way of disadvantages, it wouldn’t hurt to use a shift register to read in the values coming from sensors.

4.2.6.2 I/O Expanders

I/O expanders, just like shift registers, are useful in expanding the inputs available for a microcontroller. However, I/O expanders appear to be a bit more useful in the context of design for the Knight Light LED Chess Set. Microchip offers a line of I/O expanders including the MCP23016, MCP23017, and MCP23018. The biggest advantage in selecting I/O expanders over shift registers is that instead of only 8 bits, the I/O expanders can take in 16 input bits, making it necessary to have half as many components. This series of expanders is also compatible with the high speed I2C interface, making it even more eye opening than the shift registers. The MCP23017 has 3 hardware address pins that allow up to 8 of the devices to be used on the same I2C bus. Additionally, the MCP23017 has the ability to directly drive LEDs, this could be very beneficial toward development. The sheer amount of sensors and LEDs that will be necessary for the chess board makes the use of I/O expanders an extremely appealing prospect.

Below, Figure 4.2.6.2-1, is the block diagram for the MCP23017 I/O expander and Figure 4.2.6.2-2 is a diagram of the different available packages.

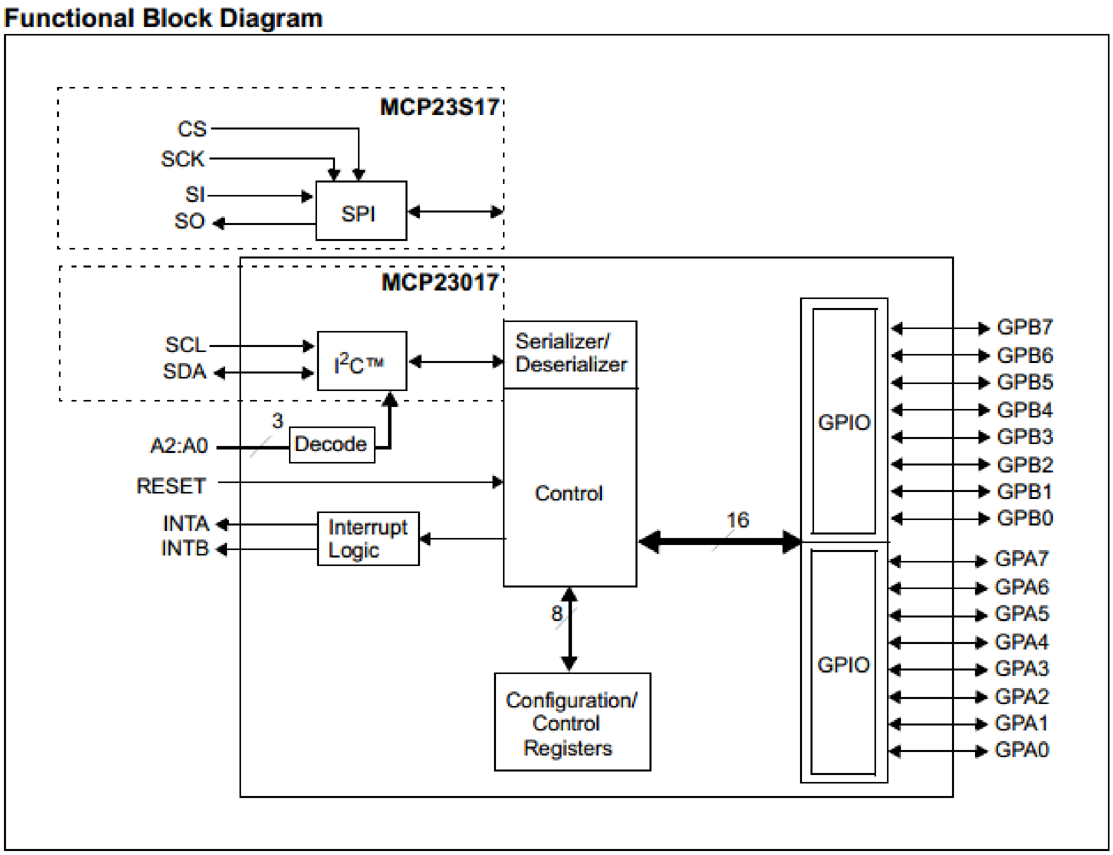

Figure 4.2.6.2-1
MCP23017 Block Diagram
Reprinted with permission from Microchip.

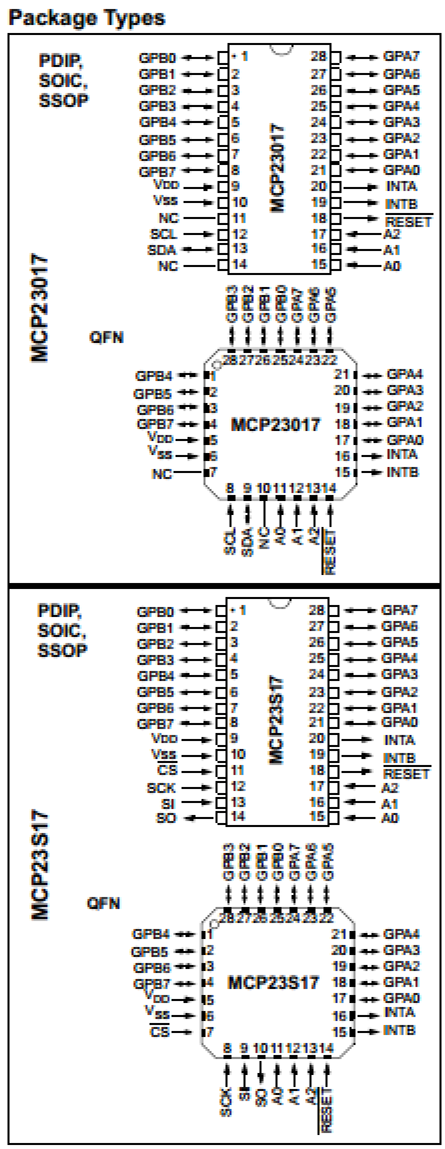

Figure 4.2.6.2-2
MCP23017 Available Packages
Reprinted with permission from Microchip.

It seems as though the clear cut choice for the interface between sensors and the microcontroller is the MCP23017 I/O Expander. The amount of inputs it can take, as well as the ability to use the I2C bus make this the apparent better option. Having to buy half as many components will save money and performance should be better than that of shift registers. With the reduced amount of components needed and the MCP23017 in the $1.50 - $1.75 range, they will be affordable and could help bring the overall project budget down a few dollars.

4.2.7 Development Boards

When working on a microcontroller project it is important to use an appropriate development board. A development board like the LPC2148 Header Board by SparkFun is handy because it provides header extensions for all the ports on the microcontroller. It allows for extensive testing before getting the PCB designed for a project. Using a board like this would allow LED and sensor configurations to be tested for the Knight Light LED Chess Set. It also provides the opportunity to practice programming for the LPC2148 before a prototype PCB can be designed, ordered, and received. This specific header board offers a USB connection that also provides power to the board, as well as a 12 mHz crystal, JTAG connection, power and status LEDs, voltage regulators, and extension headers for all ports of the microcontroller. The schematic of this header board is show below in Figure 4.2.7-1.

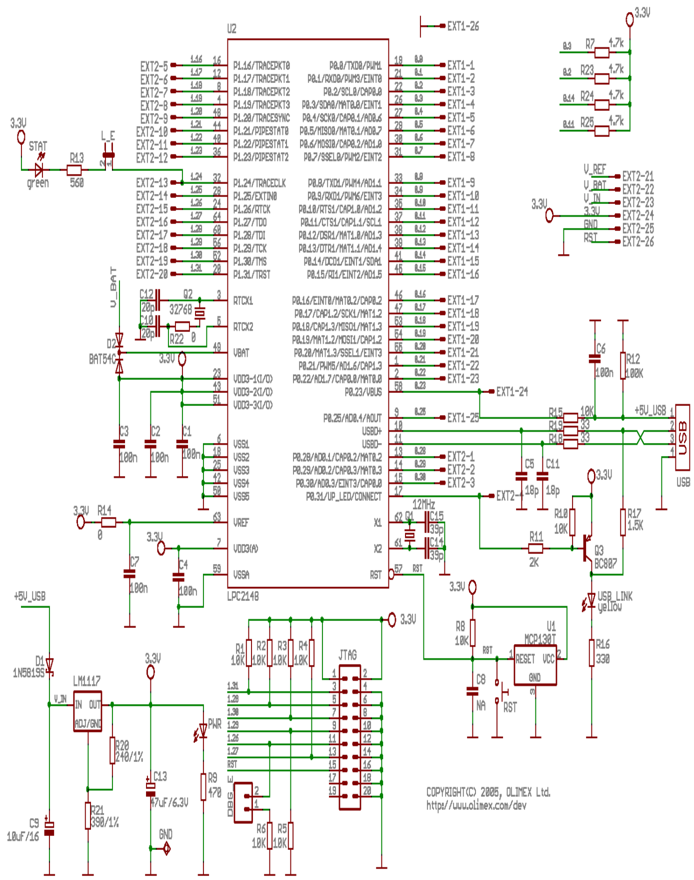

Figure 4.2.7-1
LPC2148 Header Board Schematic
Reprinted with permission from Sparkfun.

Another feature of this board is that it does not necessarily require the use of a JTAG programmer and can instead be programmed via USB. Tutorials on how to develop via USB instead of JTAG are easily accessible through websites like Sparkfun, but are a bit tricky to deal with. Below is what the header board looks like, the simple headers to access the different pins can be seen, as well as the easy to use USB connector. Example circuits from this datasheet can also be helpful in the power design for the Knight Light LED Chess Set.

Figure 4.2.7-2, shown below, is a photo of the LPC2148 development board.

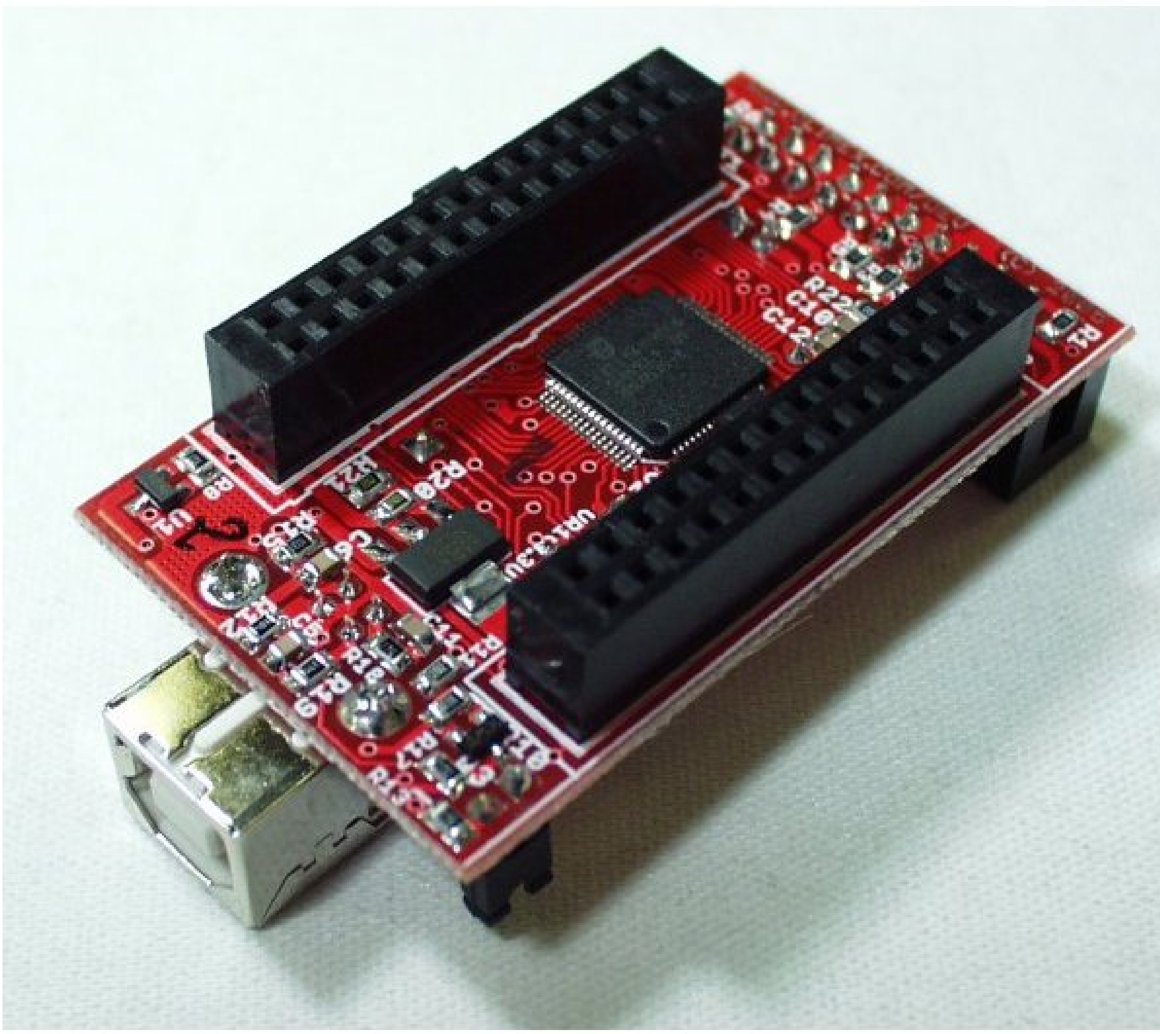

Figure 4.2.7-2
LPC2148 Development Board
Reprinted with permission from Sparkfun.

4.3 Software

As important as the hardware is in development the Knight Light LED Chess Set, the ability to program this hardware and make it useful is a job for the software.

4.3.1 General Chess Gameplay

The classical game of Chess is played with two players on a square, eight-by-eight, grid of sixty-four alternating colored squares. Half of the tiles are "light squares," usually colored white or beige, and half of the tiles are "dark squares," usually colored black or dark brown. The two players face each other, set the game board between them, and orient it in a way such that a light square is at the right-most corner of the board for both players.

Each player starts the game with a matching set of sixteen pieces and are assigned a color, chosen by the two players, either black or white. During the game, the two players are referred to by their piece colors, "Black" and "White." Each player starts with one king, one queen, two rooks, two bishops, two knights, and eight pawns.

The objective of the game is to corner the opponent's king into a "checkmate." Endgame conditions are discussed in detail in 4.3.4 Victory Conditions and Endgame Scenarios.

From each player's perspective, the eight-by-eight game grid is organized into eight rows, called "ranks," that run horizontally across the board and eight columns, called "files," that run vertically, up and down the board. The rows and columns are labeled using White's perspective of the board. The eight ranks are labeled with the numbers "1" through "8," starting at the rank closest to White and ending with the rank that is farthest from White. The eight files are labeled with the lowercase letters "a" through "h," starting at White's left and ending at White's right.

The eight pawns take up the entire rank that is second-closest to each player. All the other pieces fill in each player's closest rank. Each of the two rooks are positioned at the left-most and right-most corners of this closest rank, moving inward and toward the center of the board, the two knights are placed next to each rook. Next are the two bishops, one next to each knight. Only the two center tiles on the closest rank remain open at this point; this is where the king and queen will be placed. White places its queen to the left of the king and Black places its queen to the right of the king. The king and queen are arranged in this special configuration so that both pieces reside on the same file (column) as their opponent's and to prevent early endgame scenarios.

Shown below, in Table 4.3.1, are the amounts and types of chess pieces each player begins the game with.

Table 4.3.1
Chess Piece Allotment

Piece Name	Piece Shape	Number Allotted
Pawn	♙ | ♟	8
Knight	♘ | ♞	2
Bishop	♗ | ♝	2
Rook	♖ | ♜	2
Queen	♕ | ♛	1
King	♔ | ♚	1

4.3.2 Game Piece Properties

In this section, chess piece properties and movements will be described at the same level of detail that will be required for the software to be programmed.

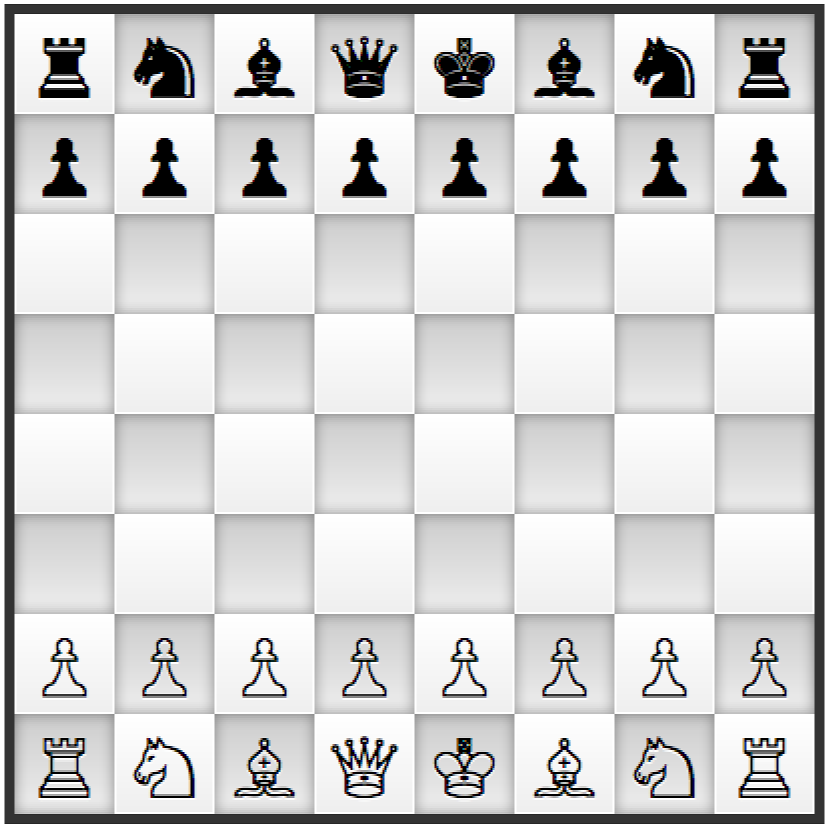

Figure 4.3.2
Chess Board Layout, Starting Positions

Chess is a turn-based game, in which, one player starts, completes their move, and then the other player moves; this cycle is repeated until the game ends. White always moves first.

Each type of piece has its own set of movement behaviors, with some pieces' behaviors acting as subsets of other pieces'. Some pieces are a great deal more powerful and valuable than others.

4.3.2.1 Pawns

Pawns are widely considered to be the weakest and most expendable piece in the game. At the same time, they can be used to achieve an advantage through influencing and controlling an opponent's moves. Strong pawn arrangements early in the game can allow for a great offense by creating a great defense.

Pawns are able to move forward one tile if that tile is unoccupied (shown by the black arrow in Figure 4.3.2.1 below), but may move forward two tiles if it is that pawn's first move of the game and both tiles are unoccupied (green arrows). Pawns are able to capture pieces if an enemy piece resides in a tile which is forward-diagonally on either side of the pawn (red arrows). Pawns may not leap over other pieces to either move or to attack and pawns may not move backwards.

Since pawns can attack diagonally only, setting up pawns in formations that protect other pawns, a player can form a diagonal wall of pawns and achieve a great strategic advantage.

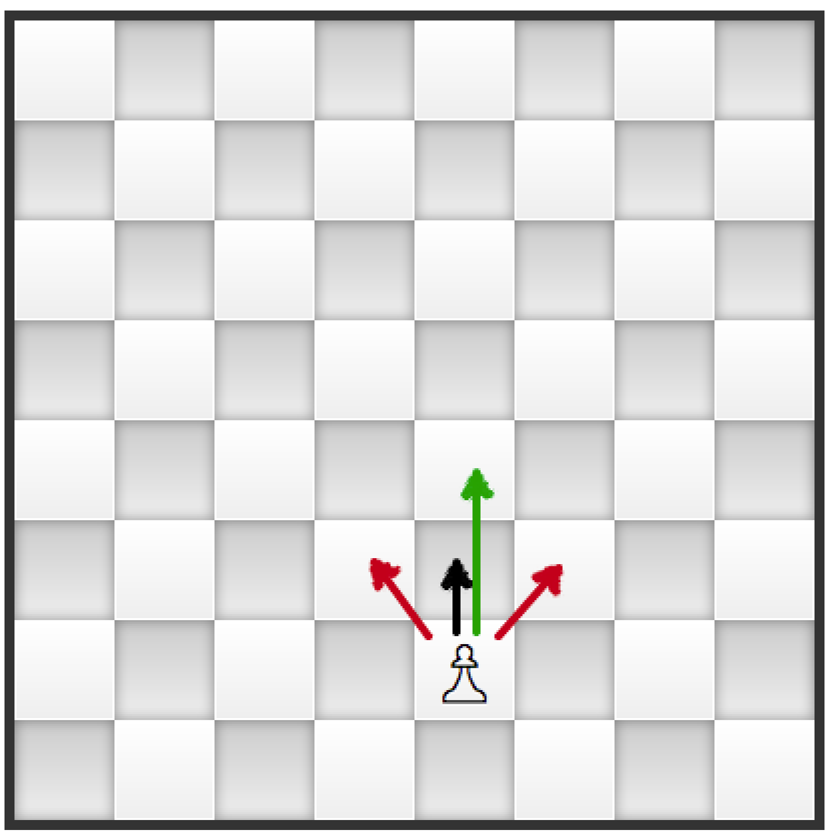

Figure 4.3.2.1
Pawn Movement

4.3.2.2 Knights

Knights are the only piece-type in the game that may leap over other pieces on the board. As shown by the arrows in Figure 4.3.2.2 below, knights can move in "L" shaped movements only; or more specifically, two tiles horizontally and one tile vertically or two tiles vertically and one tile horizontally.

Knights are considered to be quite useful in "closed positions" where other pieces may be unable to move due to cramped surroundings. Pieces like pawns, bishops, and rooks can become trapped in these situations, whereas a knight may have the opportunity to leap over a piece and break free.

A knight is able to capture an enemy piece by moving into its square.

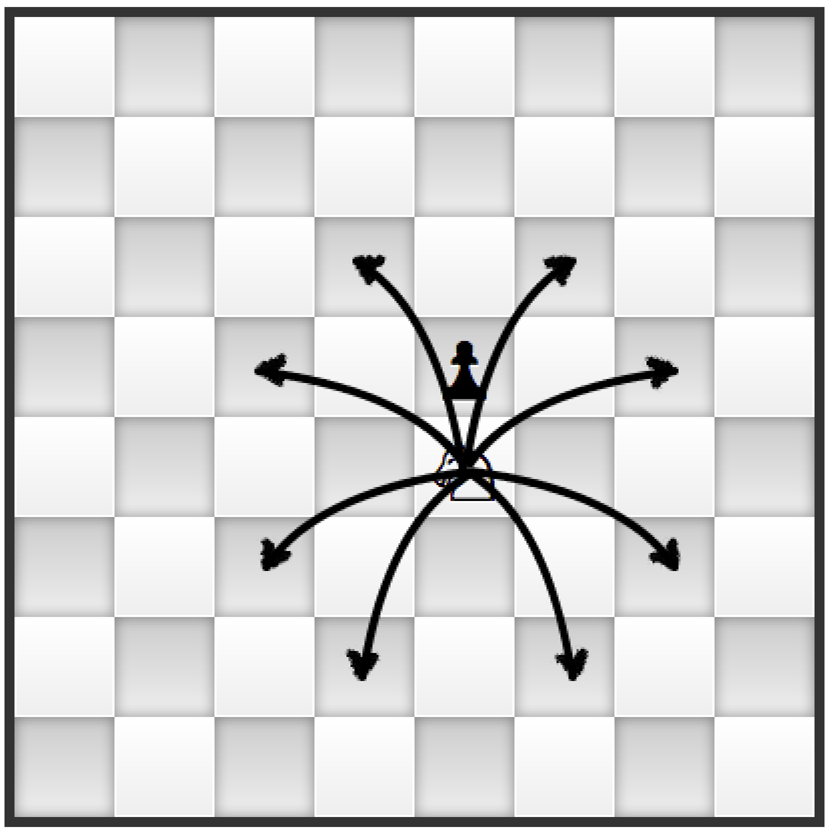

Figure 4.3.2.2
Knight Movement

4.3.2.3 Rooks

Rooks are one of the strongest piece-types in the game for reasons both offensive and defensive. They are most useful towards the end of the game because of a cramped starting position. After the game develops and other pieces move out of the way, rooks offer a huge protective advantage along the back row and sides of the board.

As shown in Figure 4.3.2.3 below, Rooks can move horizontally or vertically on the gameboard and can move any number of tiles along these horizontal and vertical lines if no other pieces are in the way.

A rook is able to capture an enemy piece by moving into its square. Rooks cannot leap over other pieces.

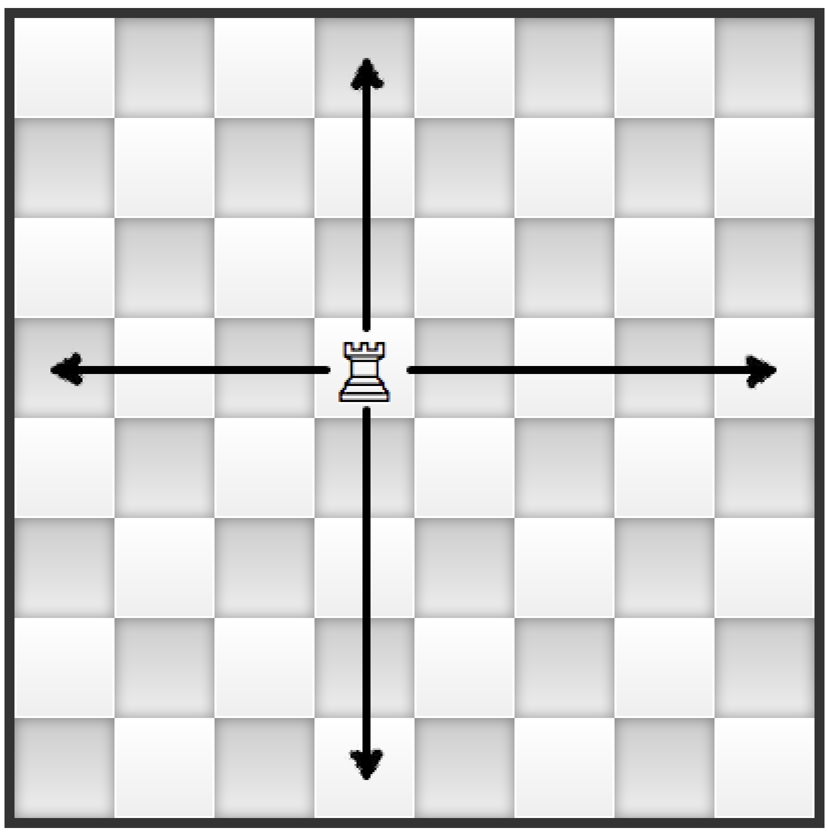

Figure 4.3.2.3
Rook Movement

4.3.2.4 Bishops

Bishops are great tools early in the game. Bishops have easy access to the middle of the board at a very early time because of their central location.

As shown in Figure 4.3.2.4 below, bishops move in a similar fashion to rooks, in that they have no distance restrictions, but unlike rooks with their strictly horizontal and vertical movements, bishops can make only move diagonally.

A bishop is able to capture an enemy piece by moving into its square. Bishops, like all other pieces except for the knight, cannot leap over other pieces.

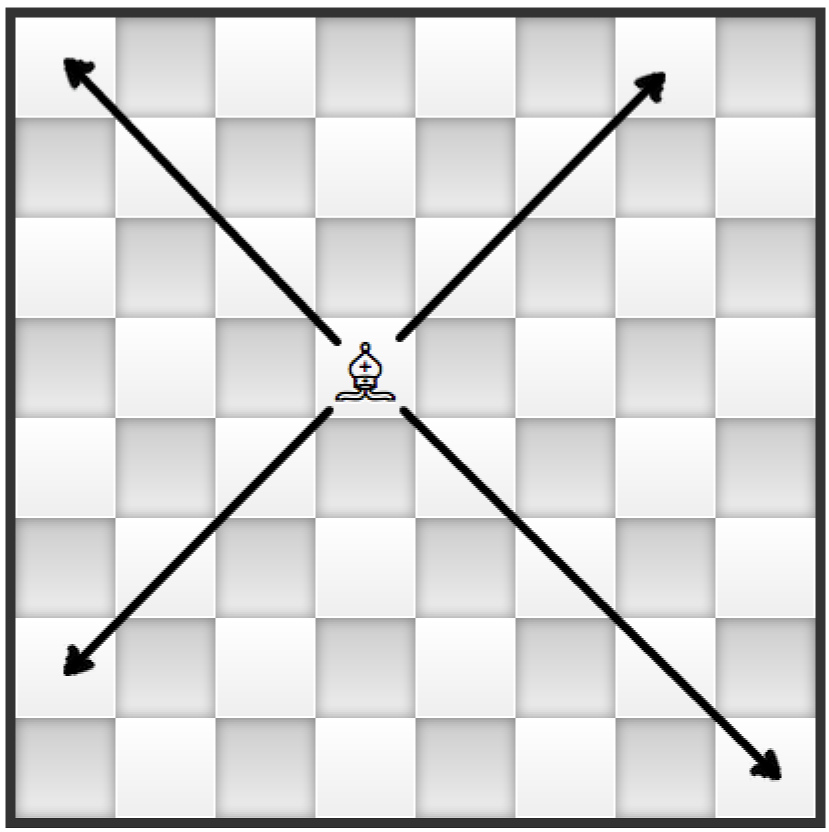

Figure 4.3.2.4
Bishop Movement

4.3.2.5 Queens

Queens are the most powerful pieces in the game. Most chess gameplay strategies revolve around using the queen in an attempt to gain an offensive or defensive advantage. Many gameplay strategies differ when it comes to deciding when to use the queen, either early or late in the game.

Bringing out the queen early is seen as an aggressive start. It can end the game early for the other player, but it's extremely dangerous for the owner of the queen. When there are many pieces left on the board, an unprotected queen is more likely to become trapped and captured. This would give the opponent a huge advantage later in the game.

As shown in Figure 4.3.2.5 below, queens can move horizontally, vertically, or diagonally in any direction and across any number of tiles, so long as those tiles are unoccupied. Like all other pieces, queens can capture an enemy piece by moving into its tile. Queens cannot leap over other pieces.

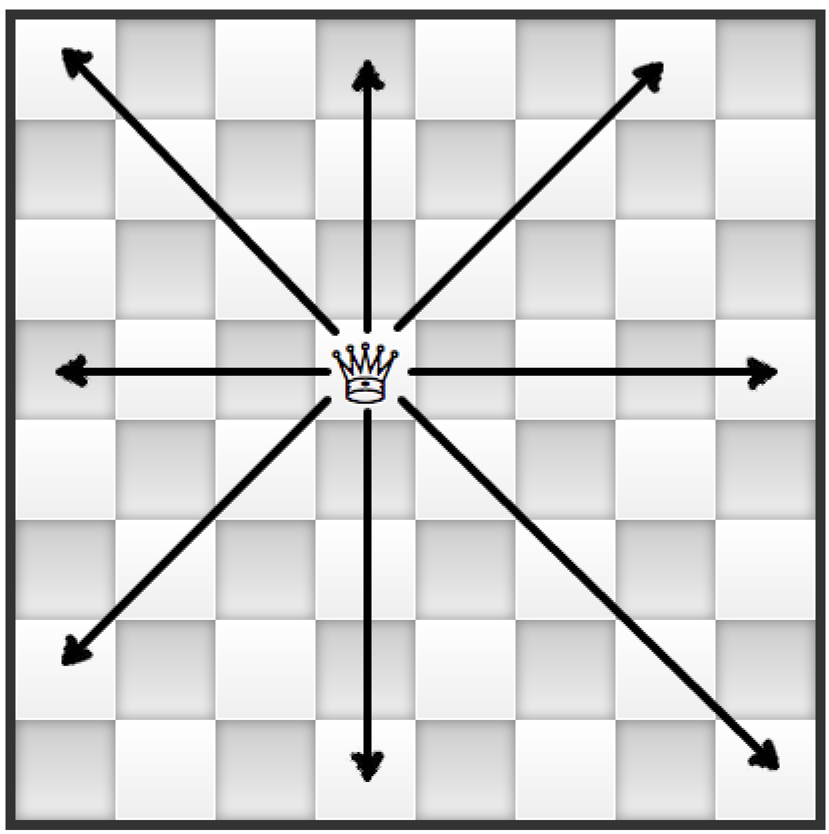

Figure 4.3.2.5: Queen Movement

4.3.2.6 Kings

Kings are the most important piece in the game, but they are also one of the weakest. When a king is captured, the game is over. Common chess gameplay strategy (and common sense) would have kings protected throughout the game, leaving them relatively useless.

As shown in Figure 4.3.2.6 below, Kings can move into an adjacent tile in any direction.

Kings may not move into tiles that are being attacked by enemy pieces. This is called “moving into check.”

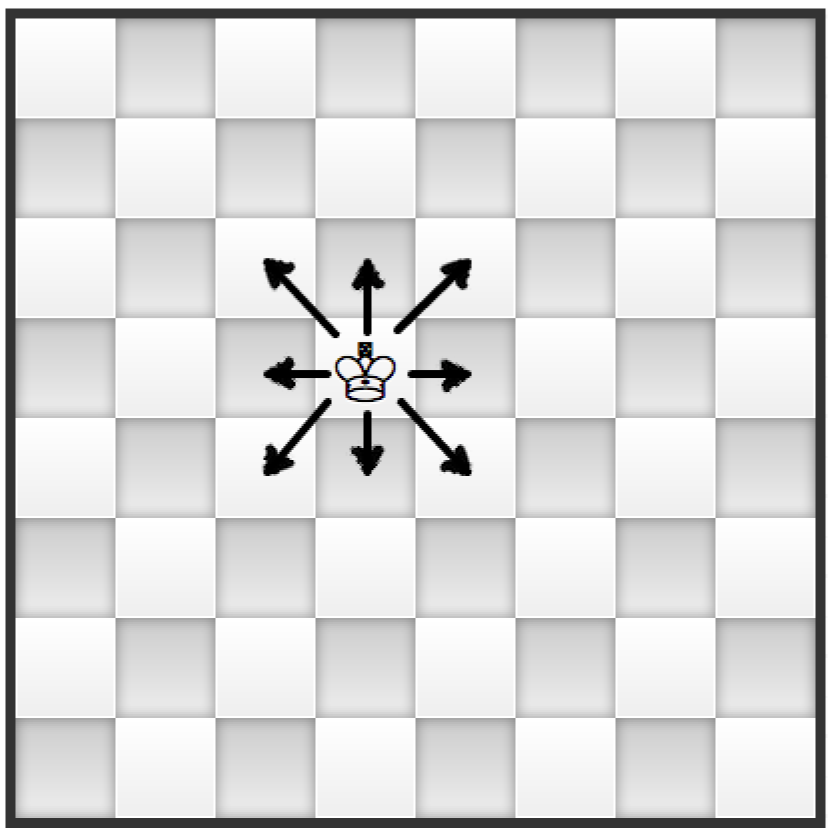

Figure 4.3.2.6
King Movement

4.3.3 Special Moves

Aside from the traditional movements pieces make in chess, as described in series of diagrams above, there are three special moves that certain pieces may perform under certain circumstances.

4.3.3.1 Castling

Castling, the first of the three special moves, is a maneuver that involves a player's king and either of the same player's rooks. Castling is a huge offensive and defensive move; it offers many benefits. It's the only move in chess where a player has the opportunity to move two of their own pieces during the same turn. By castling, a player's king moves to one side of the board, away from its central starting location, while at the same time, bringing out a rook, once hidden in a corner, into the center of the board, poised for combat. Figure 4.3.3.1 demonstrates this below.

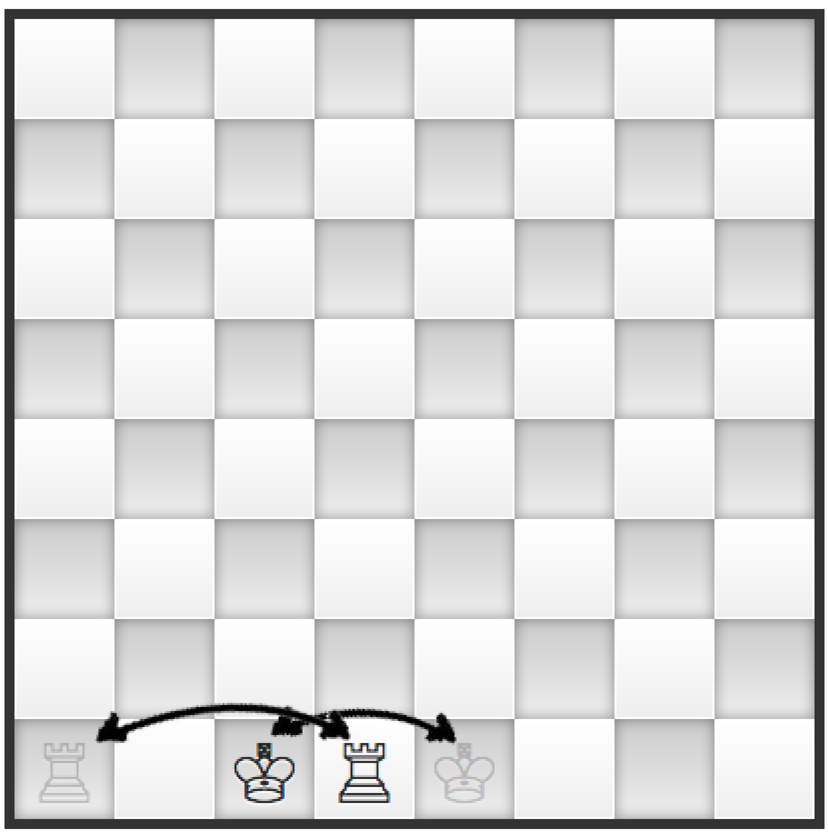

Figure 4.3.3.1
Castling

Castling can occur only if all of the following requirements are met:

- The king is not is check.
- The king has not previously moved.
- The selected rook has not previously moved.
- No pieces reside between the king and the selected rook.
- The king (through this maneuver) does not pass through a tile under attack.

4.3.3.2 En Passant

En passant is the second special move in the game of chess. “En passant” is French for “in passing.” It is a special type of pawn capture that a player can perform immediately after an enemy pawn moves two tiles from its starting position. Figure 4.3.3.2 demonstrates this below.

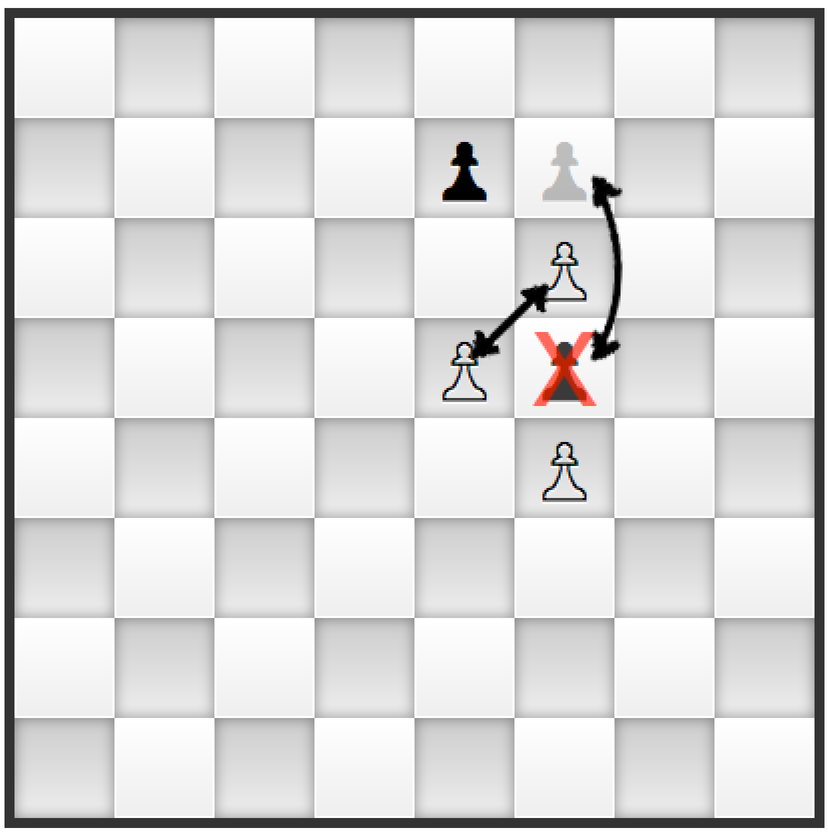

Figure 4.3.3.2
En Passant

En passant can occur only if all of the following requirements are met:

- The attacking pawn must be on the fifth rank.
- The victim pawn must be on an adjacent file and move forward two tiles (first movement for this pawn).

After the victim pawn makes it's first move, the attacking pawn can move directly behind it, executing the attack.

4.3.3.3 Pawn Promotion

Pawn promotion is the third and last special move in chess. When a pawn traverses the entire board and makes it to the opponent's home rank, the pawn must be promoted into the player's choice of a queen, knight, rook, or bishop.

Obviously, this move offers incredible advantages. Experienced chess players will usually resign once they realize they cannot stop an opponent's pawn from becoming promoted.

4.3.4 Victory Conditions and Endgame Scenarios

The object of chess is to arrange their own pieces in a way that forces their opponent's king into a "checkmate," which ends the game. There are three different types of endgame / near-endgame in chess: check, checkmate, and stalemate.

4.3.4.1 Check

A player is "in check" when any number of an opponent's pieces are attacking a tile in which the player's king resides in. Usually, the attacking player will make a verbal declaration when this move occurs and say, "check," as the player moves their piece into the attacking position.

This is a near-endgame scenario. If the checked player cannot escape or prevent the attack during the next turn, the game is over.

4.3.4.2 Checkmate

Once a player is "in check," they are presented with three options. These three moves are the only moves a player in check is permitted to make. The player must either: (1) move the king into a tile that is not under attack, (2) move another piece between the king and the attacking piece, thus blocking the attack (This will not work if the attacking piece is a knight; they can leap over other pieces.), or (3) capture the attacking piece by moving into its tile using either the king or another piece.

When the checked player cannot make any one of these three moves, the “check” turns a “checkmate” and this player loses the game.

4.3.4.3 Stalemate

Certain events may trigger a tie in chess. This usually occurs very late in the game when each player only has a few pieces left and at least one player resorts to using their king as a weapon.

Kings tend repeal each other like two magnetic poles. While a king may move one tile and any direction, it cannot move within one tile of the other king; this would be moving into check. If a player moves their king too close to the other king and it has no way of escaping, but isn’t in check or checkmate, a “stalemate” can occur. Experienced players tend to avoid stalemates at all costs.

5 Design

5.1 Game Board Housing

The main purpose of the product housing is to contain and protect all of the vulnerable components of the chess board, microcontroller, sensors, LEDs, and wires. Wood will be used to create the housing because of its familiarity of use within the group. Wood is durable, sturdy, and fairly easy to work with. Any custom paint work is simple to accomplish with wood, allowing the ability to create an aesthetically pleasing design.

A glass covering will be placed over the wood frame, still allowing the infrared sensors to work with the game pieces and make for a sleeker appearance. The goal is to make the product’s surface look as similar as possible to a regular glass chess board.

Following figures 5.1-1, 5.1-2, and 5.1-3 are basic design plans, however once actual production begins it may be decided that a component should be repositioned.

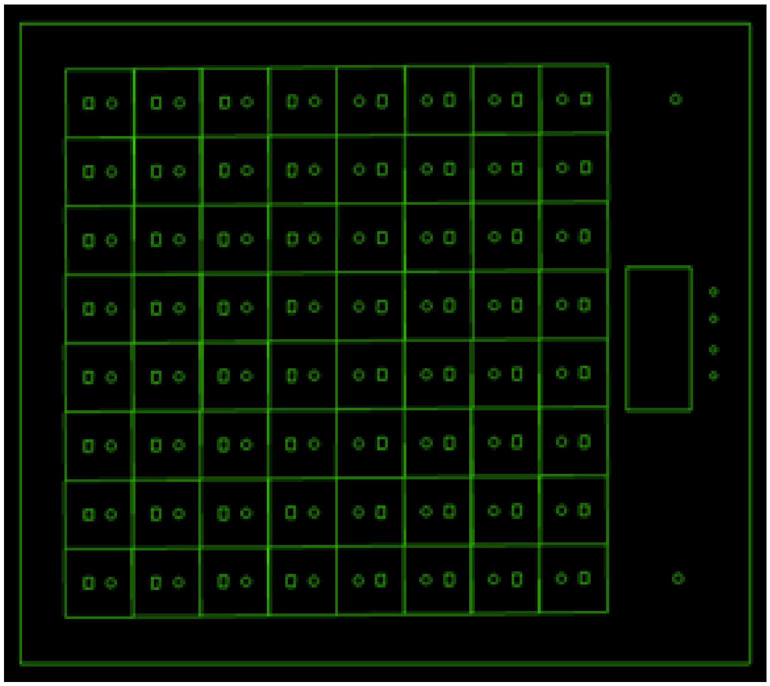

Figure 5.1-1
Chessboard Layout with Holes for Sensors and LEDs

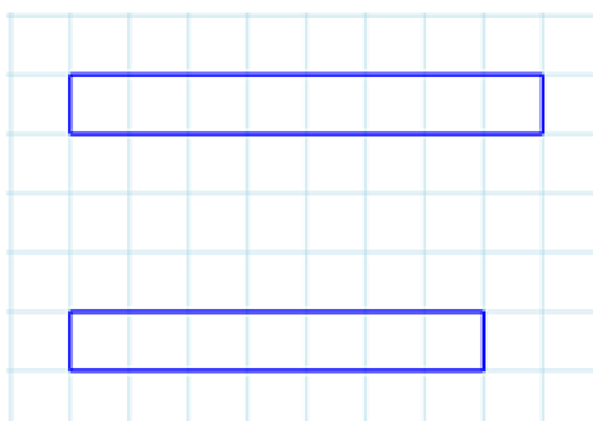

Figure 5.1-2
Chessboard Profile

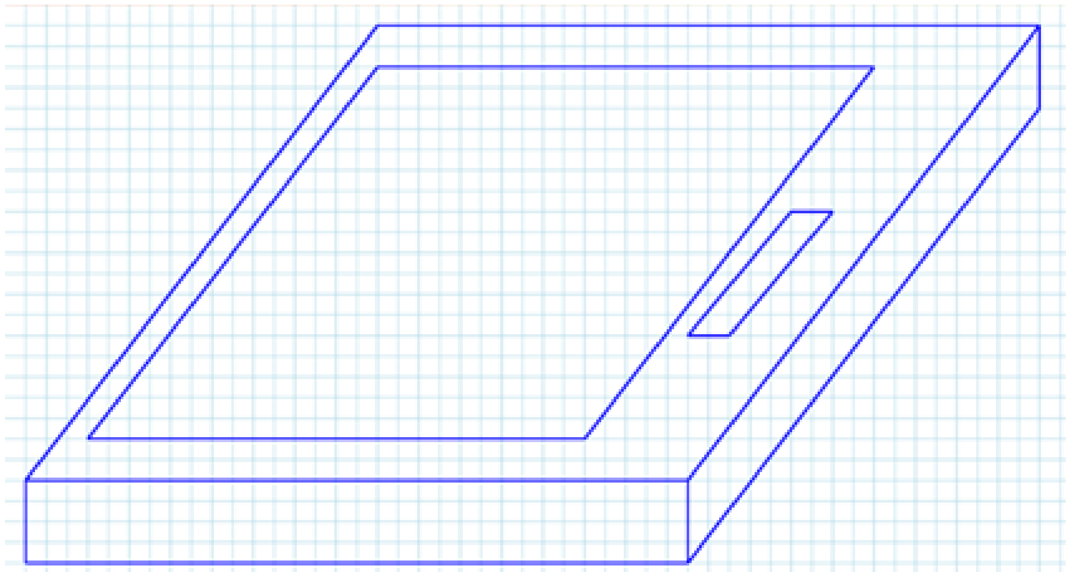

Figure 5.1-3
Chessboard Perspective

The first diagram shows the top down view of the chess board. There is a standard sized chess grid on the left and on the right there are LEDs at the top and bottom of the board that can be used to show which player's turn it is. On the right there is also a space for an LED screen and four push buttons. The buttons could be Start, Stop, Single Player, Multiplayer. The second diagram shows the side views of the Knight Light LED Chess Set; there is not anything of importance on the sides as it is all contained inside or on the coming through the top of the board. The third diagram shows a simple three dimensional rendering of the general design of the board. As the design process has gone forward the sketches and drawings have become more intricate.

The general dimensions of the housing were put together and are elaborated upon in Table 5.1 below.

Table 5.1
Chess Board Component Dimensions

Component	Dimension
Chess Square	1.5” X 1.5”
Chess Board	12” X 12”
Sensor	6.1mm X 4.39mm
LED	5 mm diameter
LCD Display	3.15” X 1.425”
Push Buttons	3.5mm diameter
Overall Board	16” X 14” X 2”

The 12” X 12” chess board decision came from following the standard sizes of chess boards. The standard sizes for chess boards generally include squares with sides of 12”, 14”, 16”, 18”, 20”, with 18” being the preferred size for tournaments. With the additional components alongside the actual board it is important to go with the smaller size for the board itself so the entire housing is not too large and clunky. A one inch border around the actual board should be appropriate and the extra two inches on the side for other components should suffice. If build time comes and it is found that 16” isn’t quite large enough, the board will be expanded to ensure there is an appropriate amount of space for all necessary components. The same goes with the 2” height of the board, it should be an appropriate dimension, but is subject to change if any issues occur.

5.2 Visualization

The backbone behind the visual display element of the Knight Light LED Chess Set is right in the name. This is the core concept of the device. Shown by green LEDs, a chess player can see available moves for a piece by simply lifting it off the game board.

LEDs are the driving force behind showing players what moves are available to them. The device can also communicate to the user if they have made an incorrect move by illuminating the invalid tile red. Figure 5.2-1 shows an example of the lighting features.

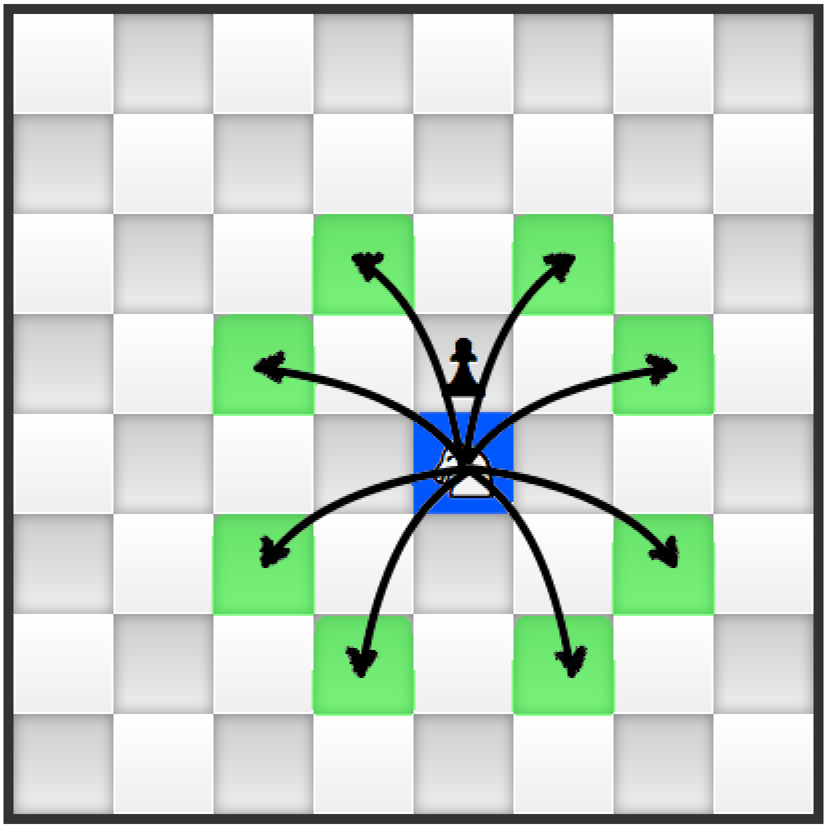

Figure 5.2-1
Available moves shown by green LEDs

The accurate control of the LEDs will be key to the success of the Knight Light LED Chess Set. The sheer amount of LEDs makes this task much more challenging than just wiring a few LEDs to general I/O pins and giving them an on or off signal. RGB LEDs will be used to provide the ability to use different colors instead of just selecting a single color for every LED in the chess board. TI has published schematics for their TLC59116EVM-390 Evaluation Module that can be extremely helpful when designing the LED/LCD Driver configuration. Excerpts from the datasheet are shown below in will be shown below in Figure 5.2-2.

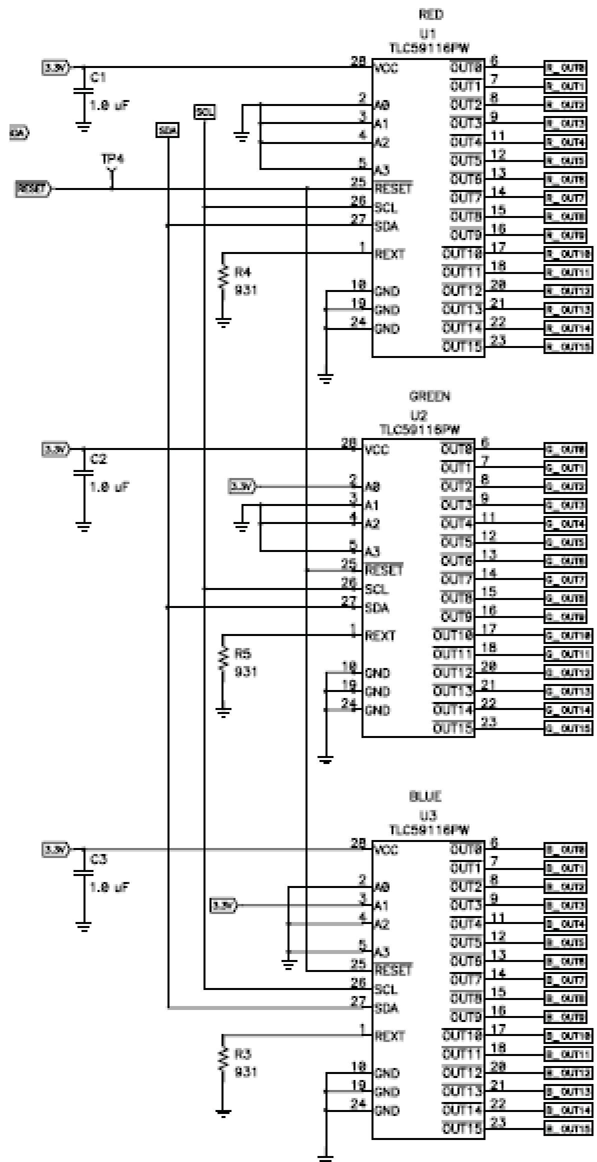

Figure 5.2-2
TLC59116EVM-390 Evaluation Module
Reprinted with permission from Texas Instruments.

This section of the datasheet helps give a general idea of what pins need to be grounded, which need resistors, and where they should be connected to the power source. The REXT pin allows the selection of the output current, A0-A3 are the address pins, SCL and SDA are the connectors to the I2C bus, and the output pins are all shown on the right for easier design purposes. The following figure shows the output pins going to a collection of RGB LEDs. The capacitors help to stabilize the electrical system and ensure the needed voltage is always available.

Figure 5.2-3, shown below, is a diagram of the TI TLC59116 Evaluation Board.

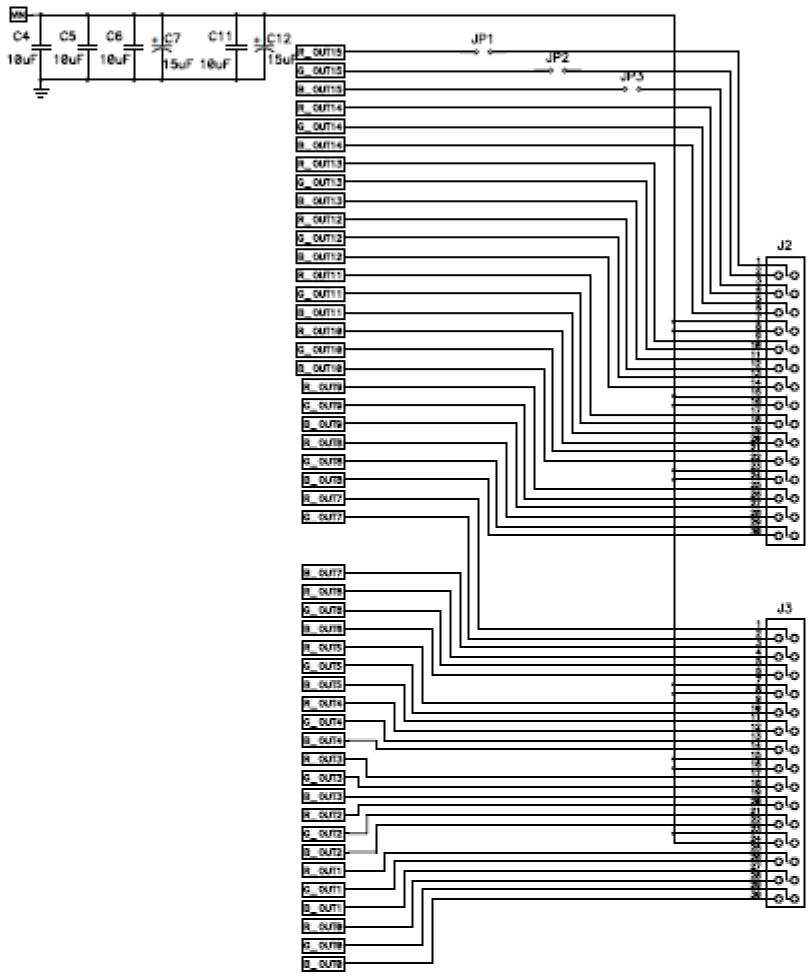

Figure 5.2-3
TI TLC59116 Evaluation Board
Reprinted with permission from Texas Instruments.

After considering the examples from TI as well as other independent research, the decided design for the LED display will be shown below. First is figure x, displaying the TLC59116 LED Driver. The REXT pin is used to determine the current magnitude for the output pins. Pins A0, A1, A2, and A3 are used for hardware select; they can be thought of as a 4 bit binary number. The 12 drivers will be selected by an input voltage on a pin to represent a 1 and grounding the pin to represent a 0.

The pin setup for the TLC59116 is shown in Figure 5.2-4 below.

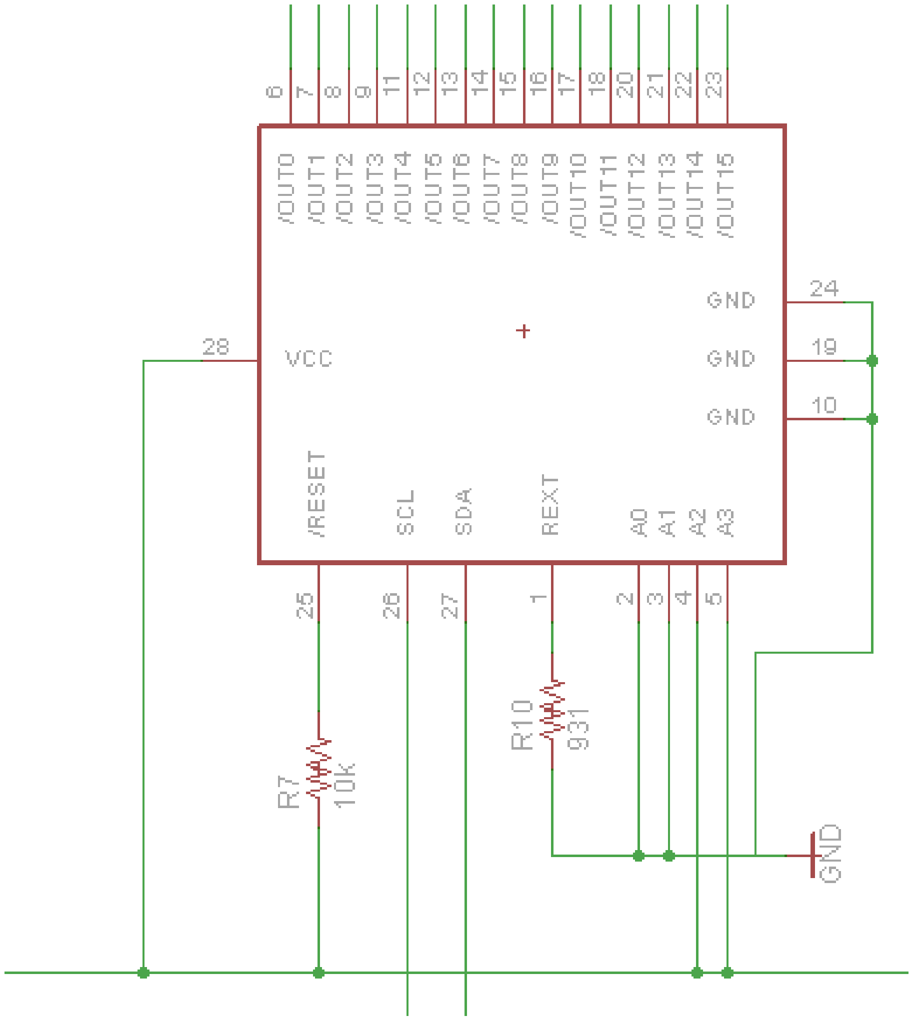

Figure 5.2-4
TLC59116

The overall display will be grouped into four-by-four sections of RGB LEDs that will be controlled by 3 drivers. Each driver will represent one of the three colors in that set of LEDs. The sections of 3 drivers and 16 LEDs will each make up a quarter of the entire display.

One of the 3 driver display quarters can be seen in Figure 5.2-5, while the entire configuration for the LED display will be shown in Figure 5.2-6.

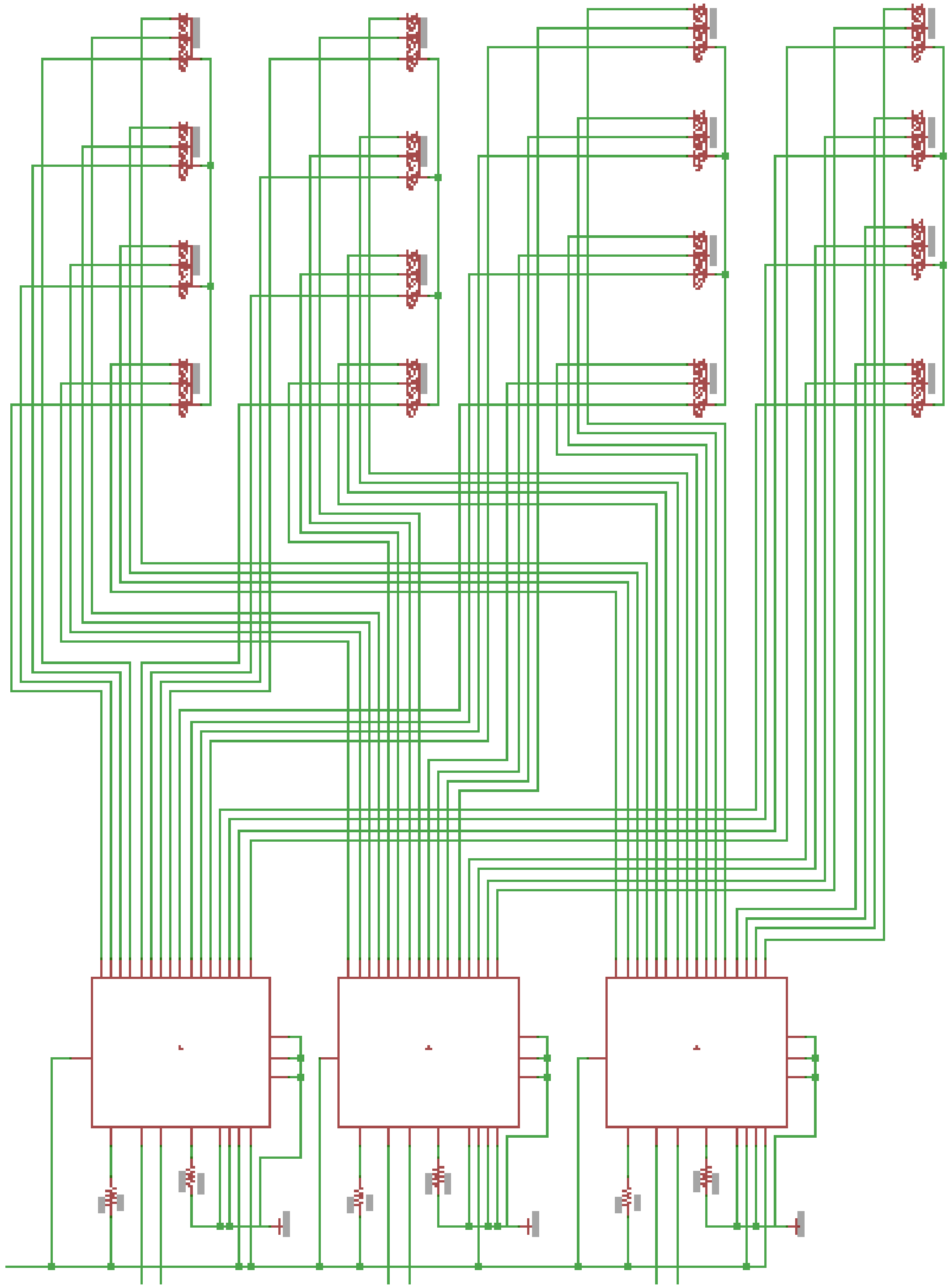

Figure 5.2-5
Single 3-Driver LED Arrangement Cluster

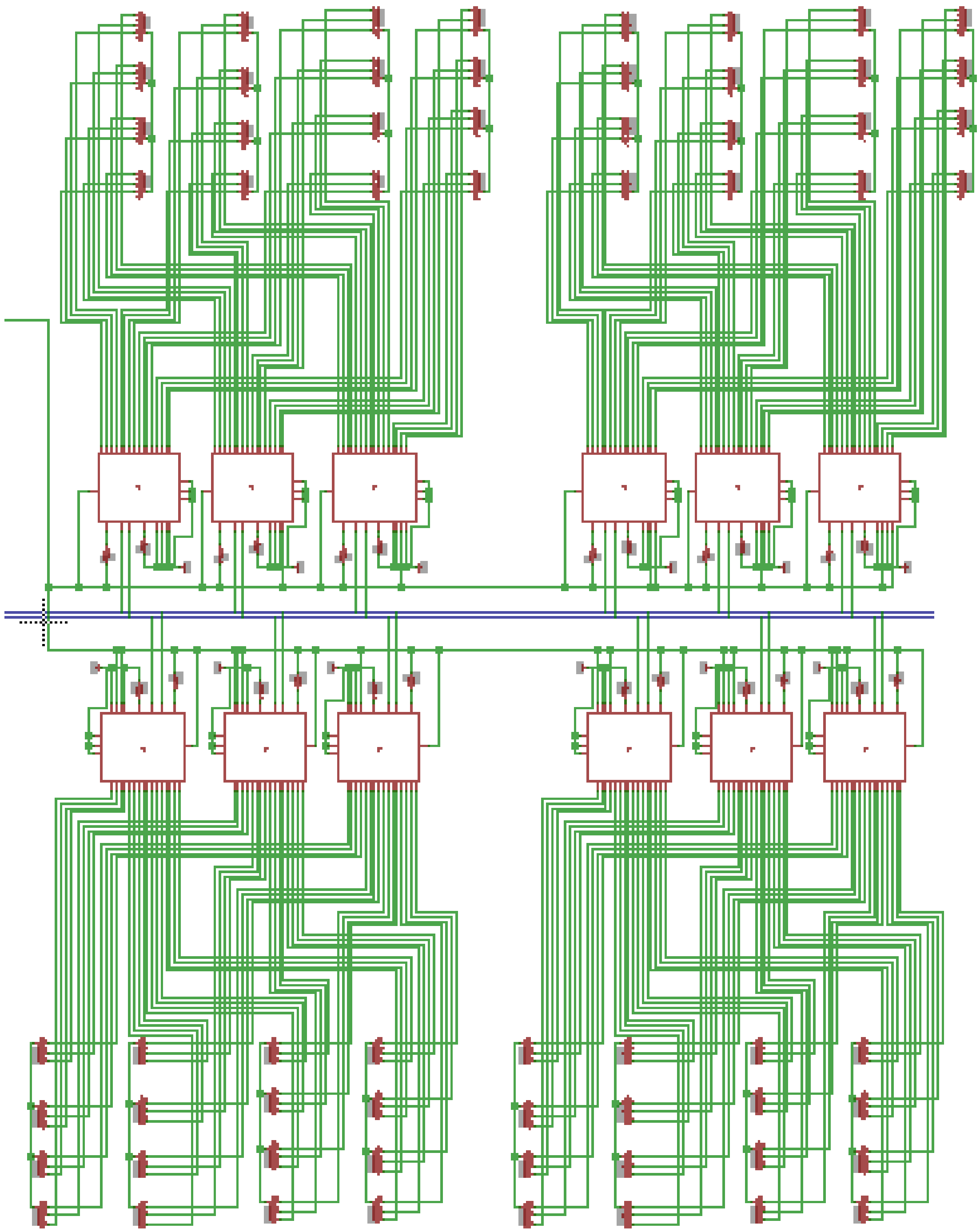

Figure 5.2-6
Entire Conceptual LED Arrangement

The figures above show the initial concept schematics developed, these had to be basically scrapped in the interest of having PCBs manufactured. All of the connections remain the same, however they will be separated between multiple PCBs. This will be further discussed in the Microcontroller Layout section.

5.3 Piece Placement Sensing

If LEDs are the backbone of the display, the QRD1114 sensors are the backbone of game piece tracking. Every square on the game board must have a sensor embedded in it to watch for pieces to enter the square or leave it. The sensors will, of course, be arranged in an eight-by-eight matrix to mimic the design of a chess board. Much like the LEDs, the sensors will be sectioned into four-by-four quarters that will be wired to a peripheral and then to the microcontroller. The MCP23017 I/O expander has address pins just like the LED drivers and will be selected in the same manner. Similar to the display section, the sensor subsystem had to be reevaluated and separated between multiple PCBs. The following figures show the initial design concepts.

Figure 5.3-1 shows the configuration of the I/O expanders.

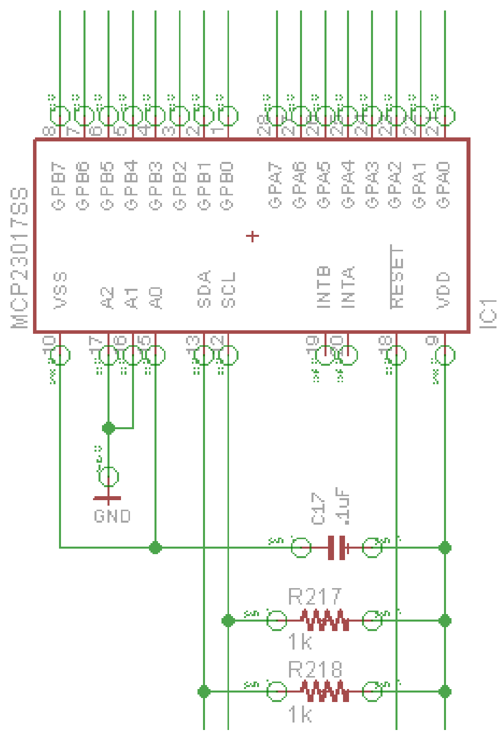

Figure 5.3-1
I/O Expander Configuration

The overall layout of the I/O expanders and sensor subsystem will be shown in figures 5.3-2 and 5.3-3.

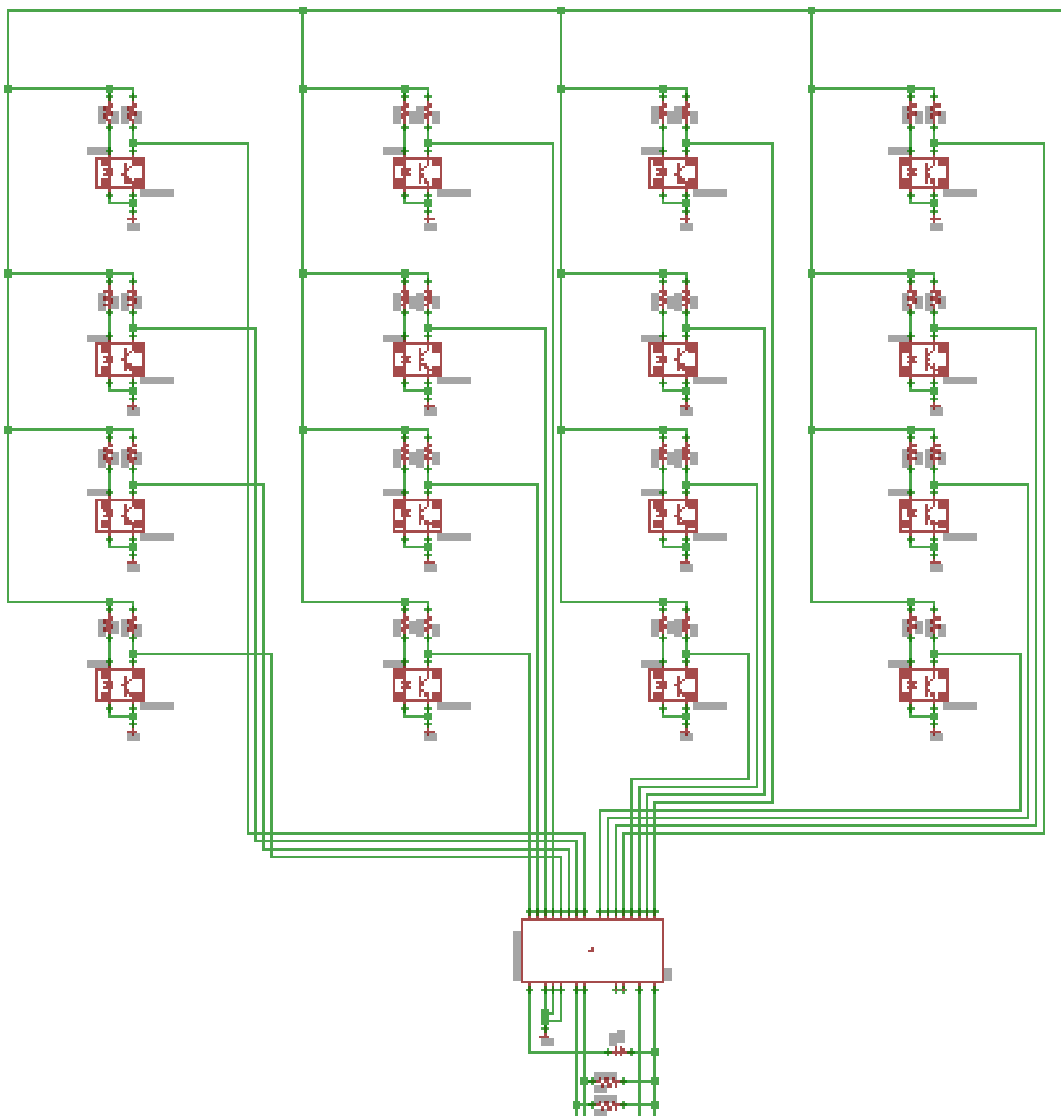

Figure 5.3-2
I/O Expander Subsystem Quarter

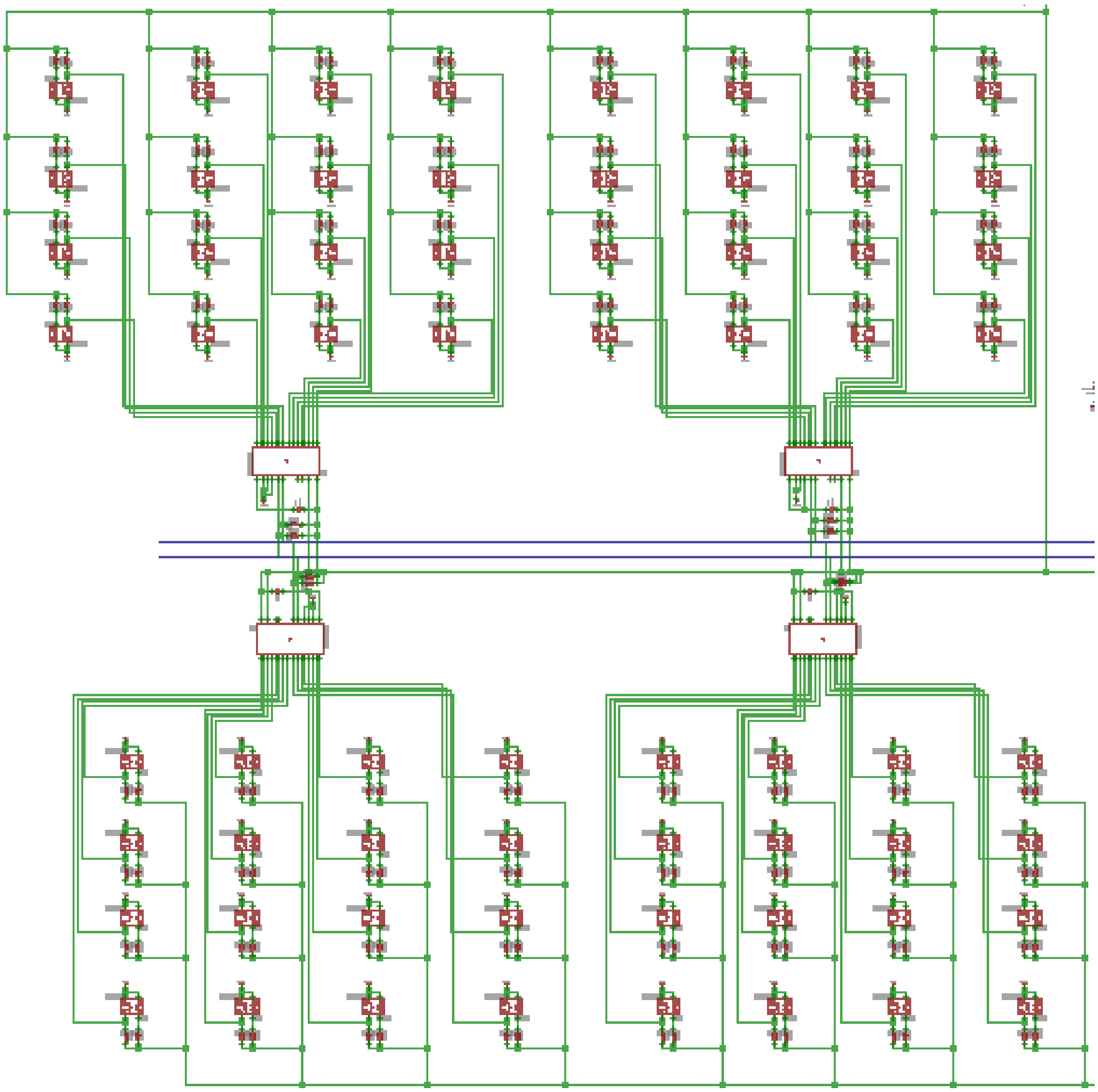

Figure 5.3-3
Full I/O Expander Subsystem Configuration

5.4 General Game Logic

The software will follow a chess game as it is played out on the chessboard using inputs sent from the hardware. Sensors on the chessboard will trigger functions inside the software. The application will process this information and provide output to the hardware to trigger the illumination of LEDs, among other things.

The software, designed using the C programming language, will contain a number of functions that completely encompass the rules of chess. The software will be able to determine if a move is valid or not based on many parameters.

The software will keep track of all chess piece positions on the board at all times during the game.

5.4.1 Algebraic Notation

Algebraic notation is a movement language and is an essential tool for tracking piece movements in a chess. The software running on the Knight Light Chess set will keep track of all piece moves of each player for playback, as well as aiding the software in determining the current locations of pieces on the board.

The language is simple and uses labeled files and ranks on a chessboard in a similar fashion to x and y coordinates used on a two-dimensional Cartesian plane. Labeled files and ranks are discussed in detail in section 4.3.1 General Chess Gameplay.

Capital letters are assigned to all piece types besides the pawn. K for king, Q for queen, R for rook, B for bishop, and N for knight. If a Bishop were to move into the tile d4, the move will be named Bd4. If a pawn were to move into f5, the move would not be prepended with a capital letter and would simply remain f5.

5.4.1.1 Ambiguity

Because this notation only describes the type of piece moving and the tile location to which it is moving into, a problem can occur when two pieces of the same type and color are able to move into the same tile.

If the files in which both pieces reside differ, the file will be included in the move's notation. For example, if one knight occupies d2 and and another king occupies g1, both have the opportunity to move into tiles f3. The move will be specified as

Ndf3 or Ngf3 depending on which knight moved from which location.

5.4.2 Piece Movement Algorithms

By using a set of structures in the software, chess piece types will be constrained to a set rules by which they can move. These allowed movements can be described and programmed as a set of algorithms.

The characteristics and movements for each piece type are discussed in detail in section 4.3.1 General Chess Gameplay.

Table 5.4.2-1, shown below, outlines the four different moves a pawn can make on a chessboard. During any given move, the pawn must move forward in the y-direction. Pawns can only capture other pieces by moving diagonally and in the positive y-direction.

Table 5.4.2-1
Pawn Movement Algorithms

Movement	X-axis (along a rank)	Y-axis (along a file)
Normal Move	n/a	y + 1
Attacking Move (left)	x + 1	y + 1
Attacking Move (right)	x - 1	y + 1
First Move Option	n/a	y + 2

In Table 5.4.2-2, below, the knight's moves are defined. Knights move in L-shaped patterns of either two tiles horizontally and one tile vertically or two tiles vertically and one tile horizontally.

Table 5.4.2-2
Knight Movement Algorithms

Movement	X-axis (along a rank)	Y-axis (along a file)
1 (clockwise)	x + 1	y + 2
2	x + 2	y + 1
3	x + 2	y - 1
4	x + 1	y - 2
5	x + 1	y - 2
6	x - 2	y - 1
7	x - 2	y + 1
8	x + 1	y + 2

In Table 5.4.2-3, the bishop's movements are set. Bishops can move diagonally only. This can be described as moving the same number of tiles in along the x and y axises, while only altering the signs of these movements.

Table 5.4.2-3
Bishop Movement Algorithms

Movement	X-axis (along a rank)	Y-axis (along a file)
Right and Up	x + A	y + A
Right and Down	x + B	y - B
Left and Down	x - C	y - C
Left and Up	x - D	y + D

Unlike bishops, rooks can only move horizontally and vertically. In Table 5.4.2-4, shown below, the rook's movement algorithms are defined.

Table 5.4.2-4
Rook Movement Algorithms

Movement	X-axis (along a rank)	Y-axis (along a file)
Up	n/a	y + A
Right	x + B	n/a
Down	n/a	y - C
Left	x - D	n/a

Queens take the movement traits of both the rooks and the bishops. In Table 5.4.2-5, shown below, both of the previous tables are meshed together to create the superset of moves a powerful queen can perform.

Table 5.4.2-5
Queen Movement Algorithms

Movement	X-axis (along a rank)	Y-axis (along a file)
Up	n/a	y + A
Right and Up	x + B	y + B
Right	x + C	n/a
Right and Down	x + D	y - D
Down	n/a	y - E
Left and Down	x - F	y - F
Left	x - G	n/a
Left and Up	x - H	y + H

Kings, just like queens, can move in any direction. The only difference is that the king is limited to just one tiles in these directions, whereas the queen can move an unlimited number of tiles. Below, in Table 5.4.2-6, the king's movement algorithms are shown.

Table 5.4.2-6
King Movement Algorithms

Movement	**X-axis (along a rank)**	**Y-axis (along a file)**
Up	n/a	y + 1
Right and Up	x + 1	y + 1
Right	x + 1	n/a
Right and Down	x + 1	y - 1
Down	n/a	y - 1
Left and Down	x - 1	y - 1
Left	x - 1	n/a
Left and Up	x - 1	y + 1

5.4.3 Gameplay Pseudo-Code

```
while(gameOver == false) {
     while(playerTurn) {
          if (player is in checkmate) {
               gameOver = true;
          }
          if (player is in check) {
               Update LCD -> "[Player color]'s in Check";
               Illuminate King's tile RED;
               Illuminate attacking piece's tile RED;
               Scan and wait for hardware input;
               validateMove(player, moveArray, check);
          }
          Update LCD -> "[Player color]'s Turn";
          Scan and wait for hardware input;
          validateMove(player, moveArray, normal);
     }
}
void validateMove(player, moveArray, status) {
     if (player picks up a piece) {
          Illuminate piece's original position BLUE;
          Determine available moves for that piece;
          Illuminate available tiles GREEN;
          Scan and wait for hardware input;
          if (player puts down piece) {
               Deluminate all tiles;
          }
          if (player makes an invalid move) {
               Illuminate invalid tile RED;
          }
          if (player makes a valid move) {
               Illuminate valid tile BLUE;
               Update game piece coordinates;
               End turn;
          }
     }
}
```

5.5 Artificially Intelligent Chess Computer

If the one-player mode is selected on the device, an artificially intelligent computer player will be available to play against. The same basic rules of chess will apply; the game will start with white moving first, each player will take turns moving and capturing pieces, and the game will end in either a checkmate or a stalemate.

5.5.1 Chess Piece Relative Value

The single most important function of an artificially intelligent computer chess player is the accurate detection of good chess moves. A feature of the AI that can help tremendously in this the department is a predefined set of relative values for each of the different chess piece types. The AI can use this programmed information to target nearby enemy pieces that have a higher relative value score. Using this sort of high-value piece targeting paradigm, a stronger AI can be developed.

Many chess experts have analyzed the game and studied how important different chess pieces are to a player based on many different metrics and thousands of test games. The table below, Table 5.5.1 Standard Valuations, is the most common standard value assignments for chess type pieces. The king is usually left out of such value assignments. The values below will be used in the AI for this product.

Table 5.5.1
Standard Valuations

Pieces	Point Values
Pawn	1
Knight	3
Bishop	3
Rook	5
Queen	9

5.6 Power Source

To ensure there is a sufficient supply of input voltage and current, a wall wart and a barrel jack will be used to provide power to the Knight Light LED Chess Set. The power line will go to an ON/OFF switch and then be regulated to 3.3v and 5v for the different subsystems of the project. Two of the 5v regulators are used to help distribute heat and ensure that the LED and sensor subsystems have enough current coming to them. Figures 5.6-1, 5.6-2, and 5.6-3, below, show the power input and voltage regulation sections of the circuitry.

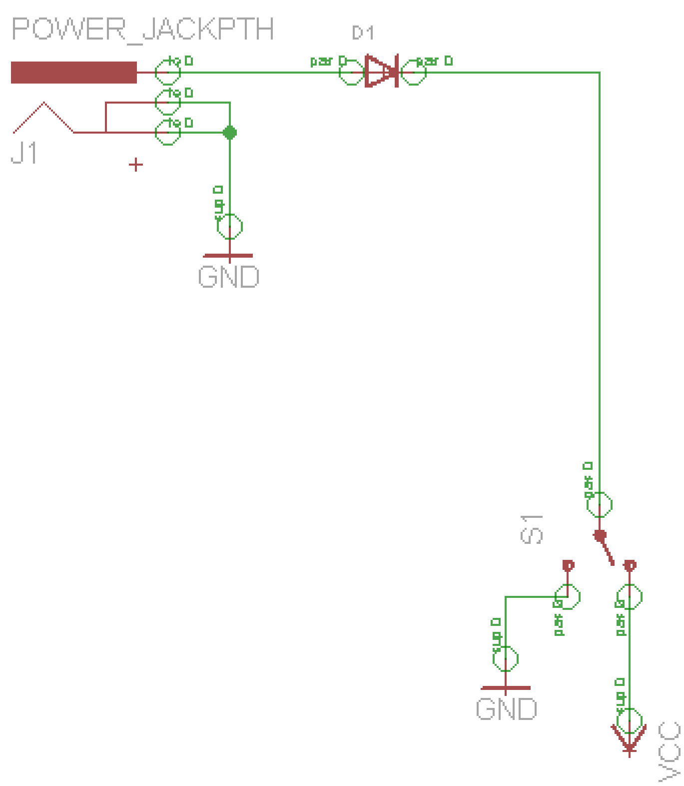

Figure 5.6-1
Barrel Jack to ON/OFF Switch

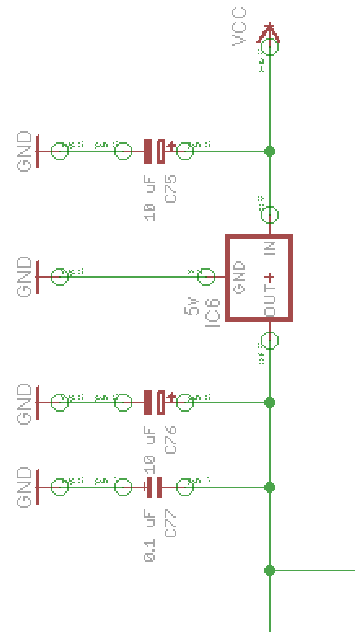

Figure 5.6-2
5V Regulator Circuit

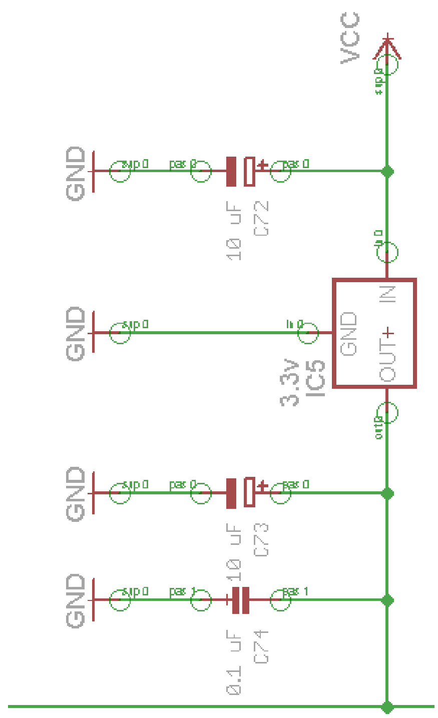

Figure 5.6-3
3.3V Regulator Circuit

5.7 LCD Display and Buttons

The LCD screen can be simply wired up through the use of GPIO pins. The push buttons for determining the game mode can similarly just be wired through GPIO. The initial conceptual wiring for each of these components can be seen in figures 5.7-1 and 5.7-2 below.

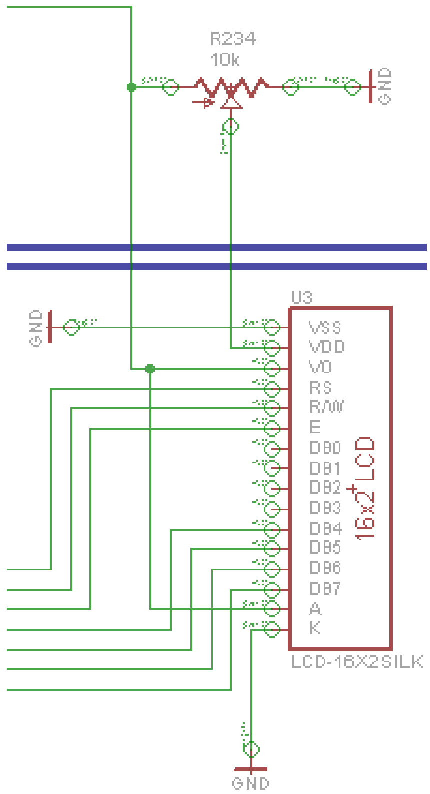

Figure 5.7-1
LCD Screen Wiring

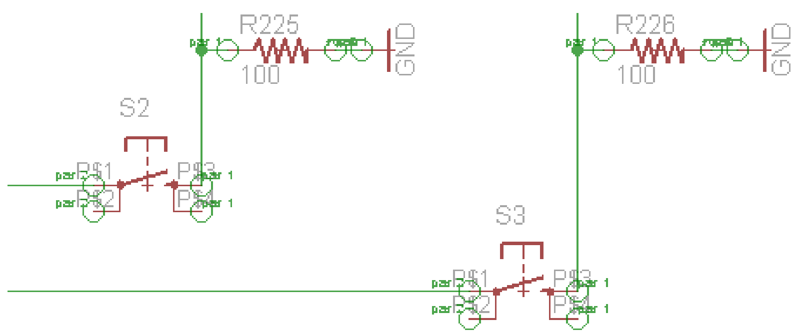

Figure 5.7-2
Mode Buttons

5.8 Microcontroller Layout

The layout of the entire project gets somewhat complicated because of the amount of wires needed for the large amount of LEDs and sensors. The Microcontroller also needs to be wired for a JTAG connection, a RESET circuit, and an oscillator crystal for the clock. Figure 5.8-1 shows the JTAG layout, 5.8-2 shows the reset circuit, 5.8-3 shows the simple crystal setup, and 5.8-4 shows the theoretical entire layout.

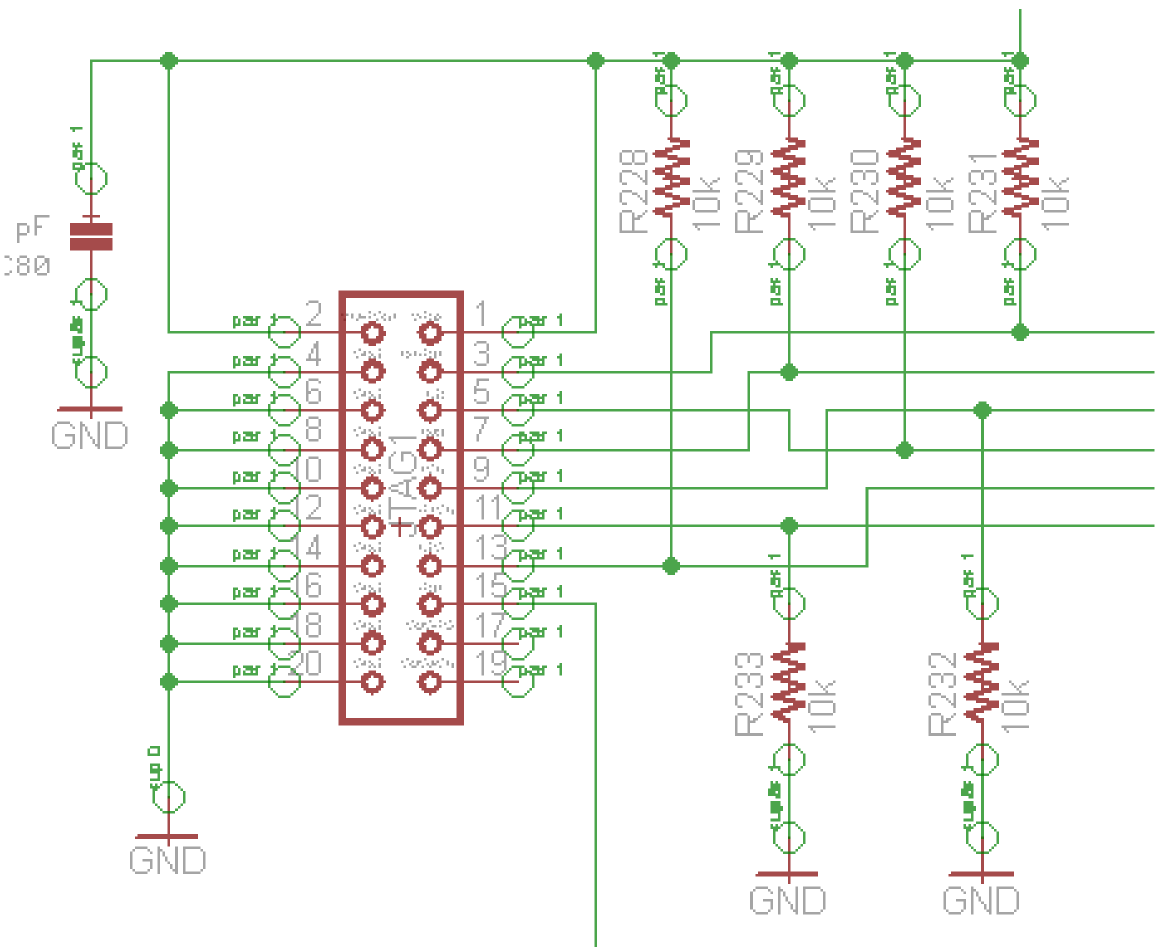

Figure 5.8-1
JTAG Wiring

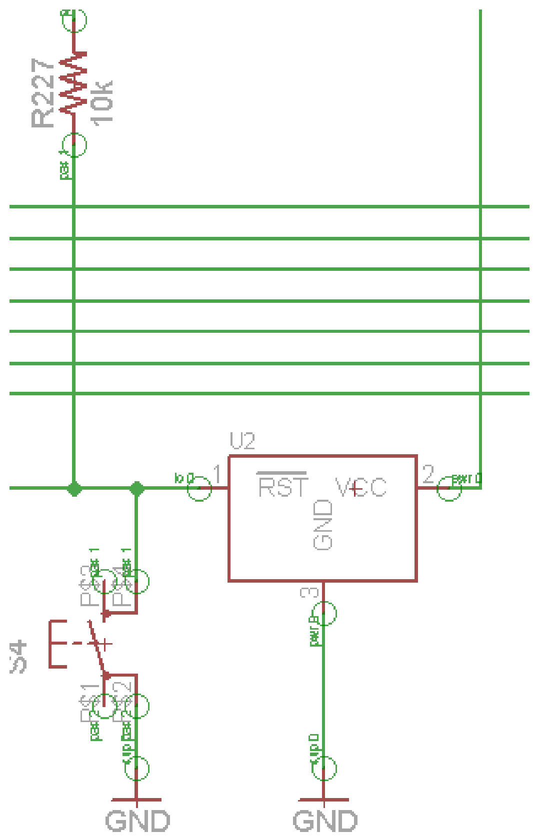

Figure 5.8-2
RESET Circuit

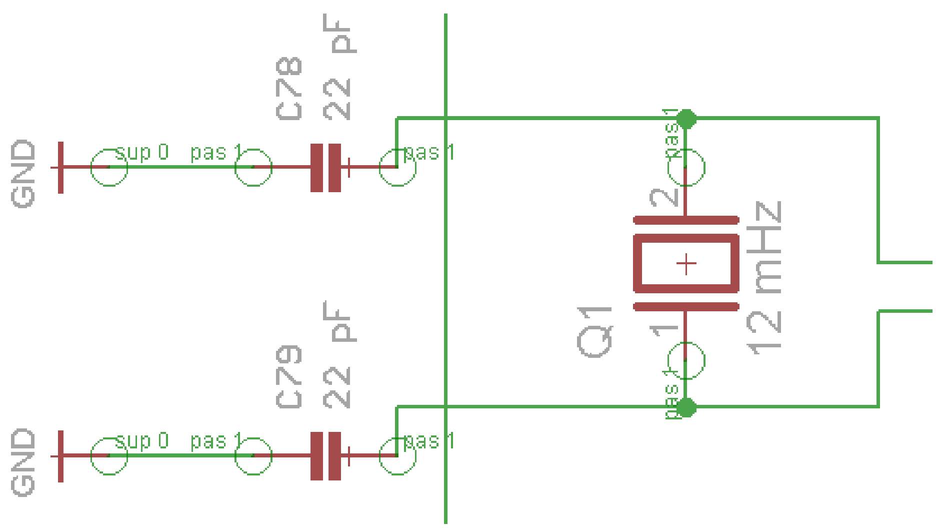

Figure 5.8-3
Crystal Oscillator for CLCK

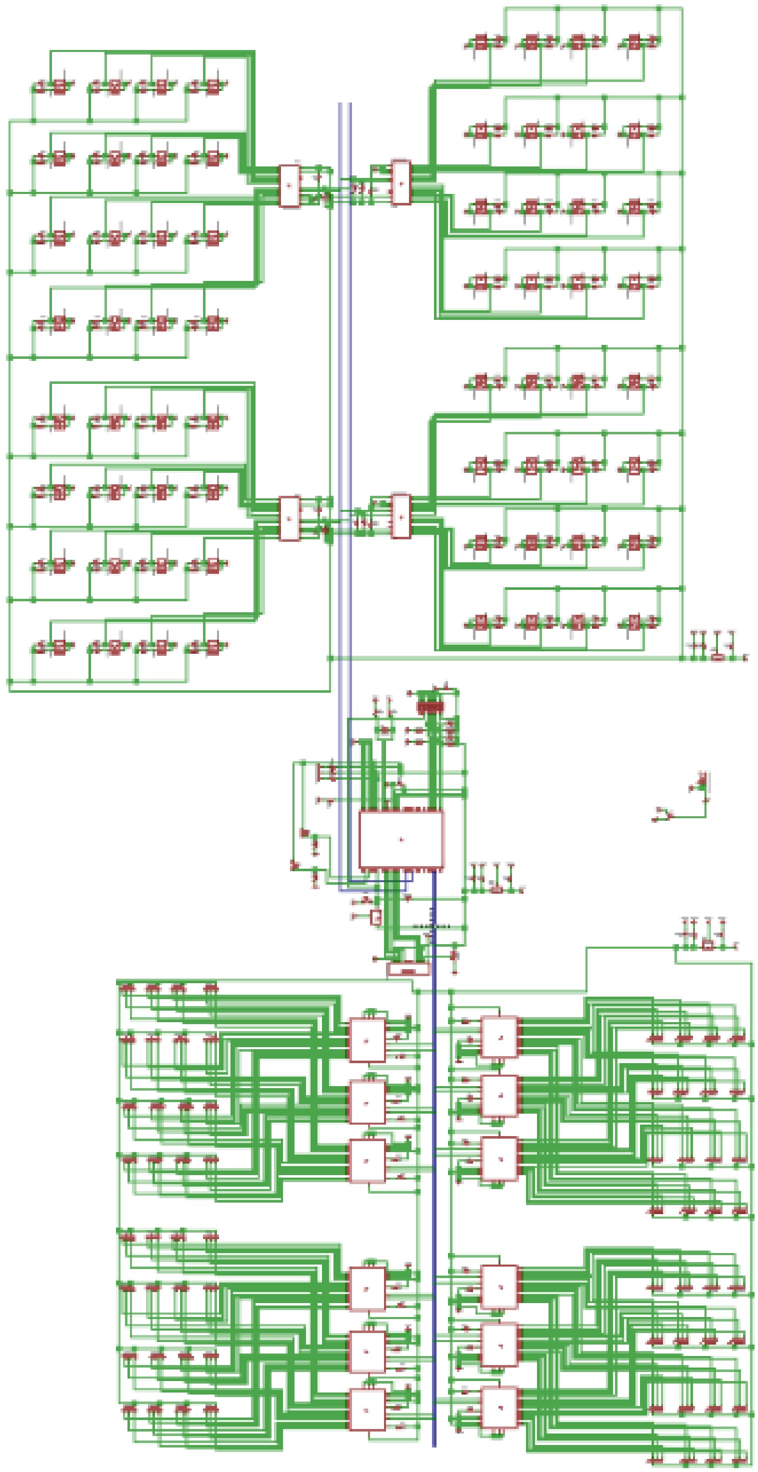

Figure 5.8-4
Theoretical Entire Schematic

When PCB design discussions started it became apparent that with the free version of Eagle it would not be possible to design a PCB that could hold all of the LEDs, sensors, and other components. The direction of the project shifted to the idea of using separate PCBs for the major subsystems of the chess board. Three boards will be designed; one will support the microcontroller, JTAG connection, and power supply, one will support the LED Drivers, and the last will support the I/O Expanders. Because of the amount of LEDs and sensors, they will not be directly on a PCB but on a perforated board that will be the size of the playing surface. The PCBs will contain either headers or soldering pads to connect to the LEDs and sensors. As of now the plan is to use soldering pads, however, if during the PCB design process it is found that headers will work better, the schematics will be updated accordingly. Figures 5.8-5, 5.8-6, and 5.8-7 show the schematics for each major subsystem.

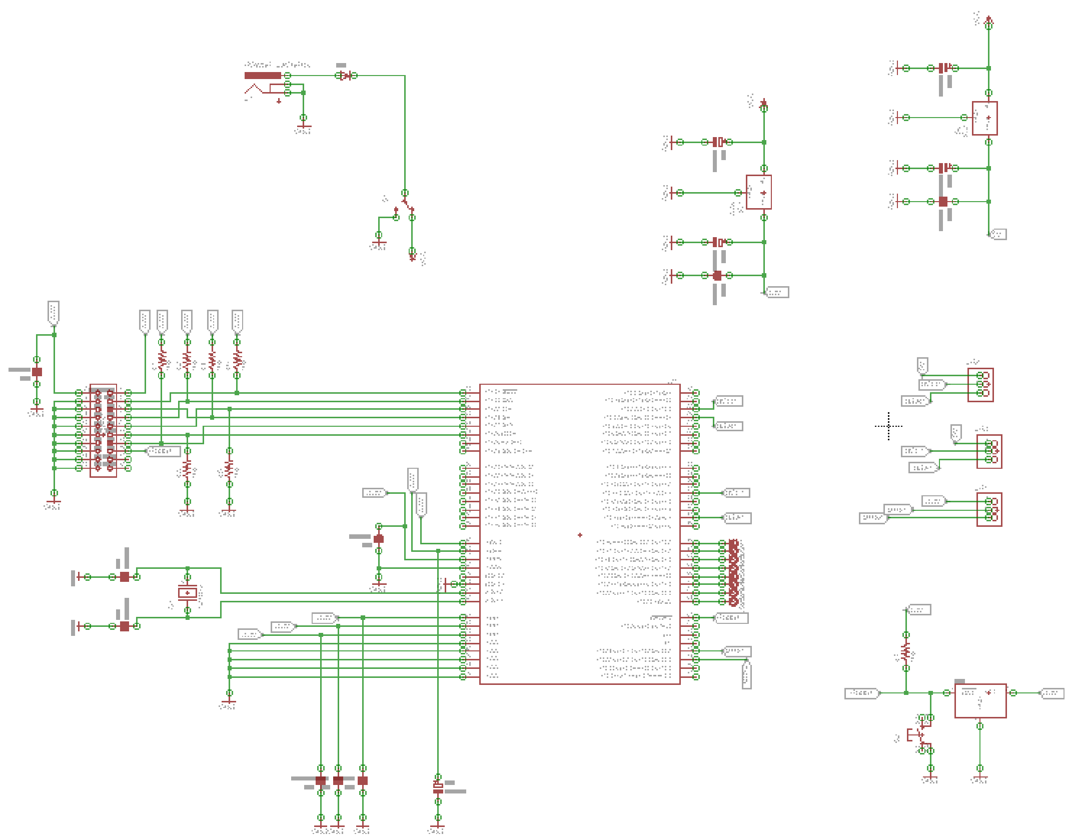

Figure 5.8-5
Main Micro Circuit Schematic

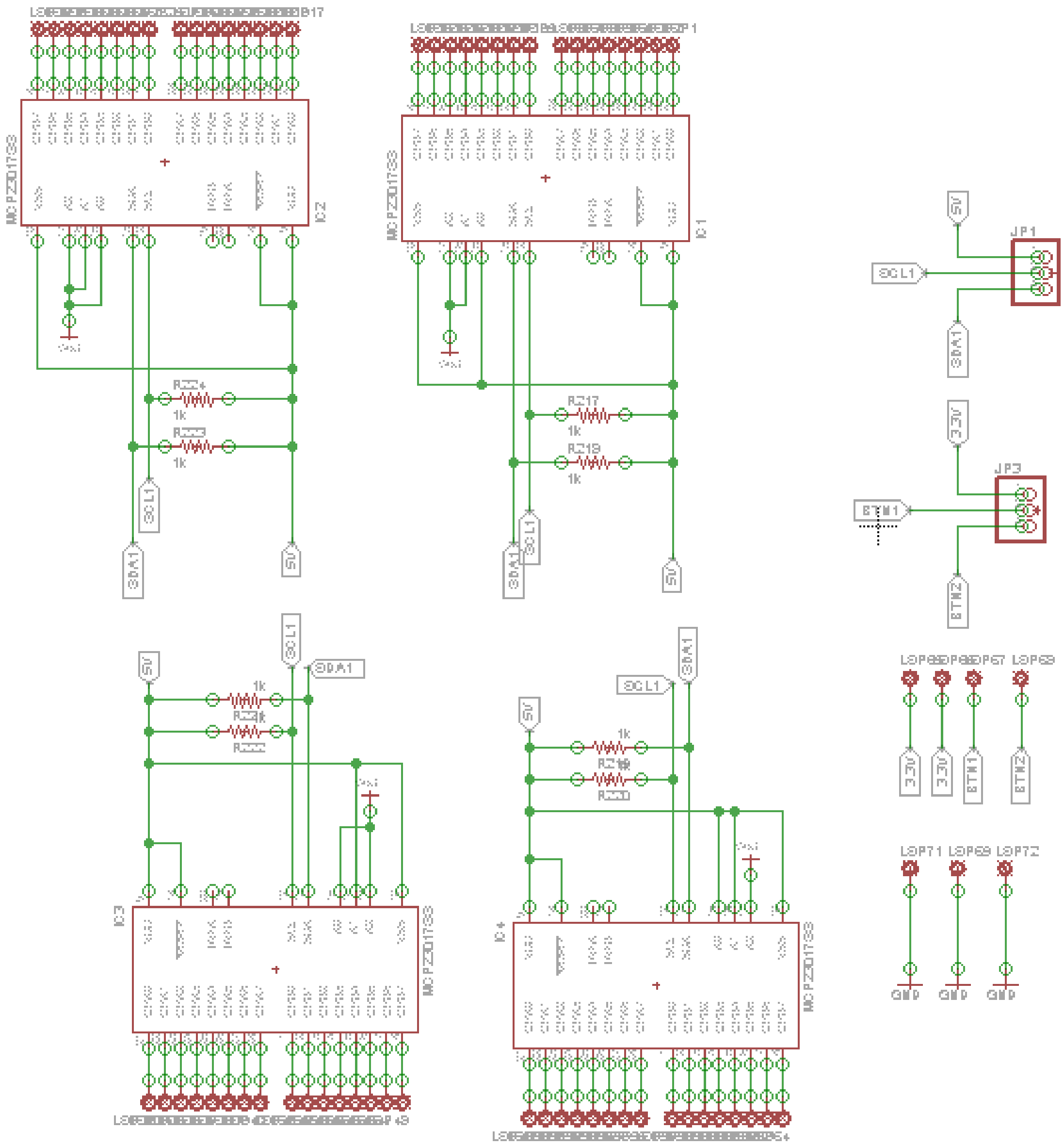

Figure 5.8-6
I/O Expander and Buttons Schematic

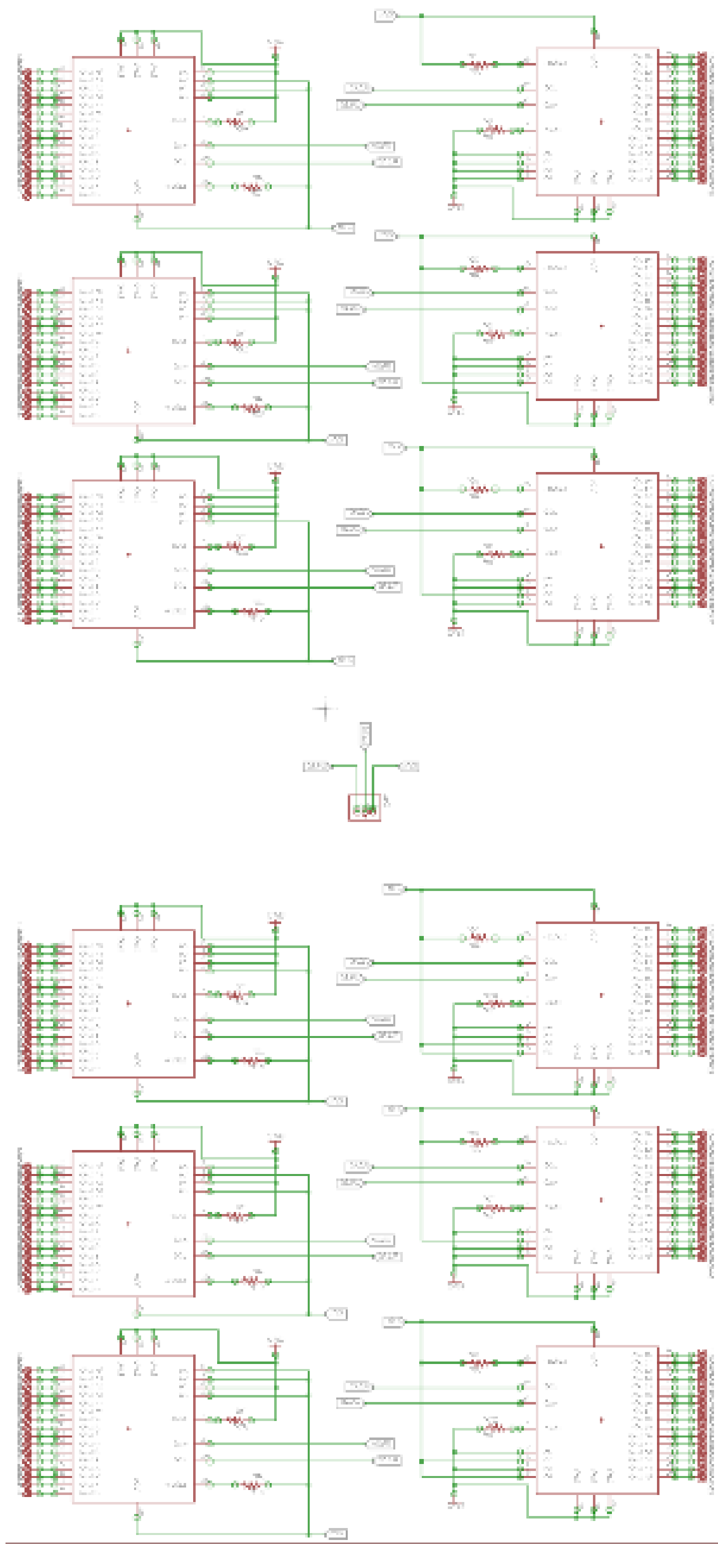

Figure 5.8-7
LED Driver Schematic

5.9 PCB Plan

Schematics will be turned into PCB designs that will be manufactured to hold the main components of the chess board. Possible manufacturers include ExpressPCB and Advanced Circuits. In the coming weeks designs will be finalized and sent to companies for estimates. A decision will be quickly made so an order can be placed to start putting the board together. While waiting for estimates and eventual manufacturing, parts will be ordered and a development board will be used to test out the parts and how the subsystems work. Some PCB design has been started and figures 5.9-1 and 5.9-2 show the progress so far.

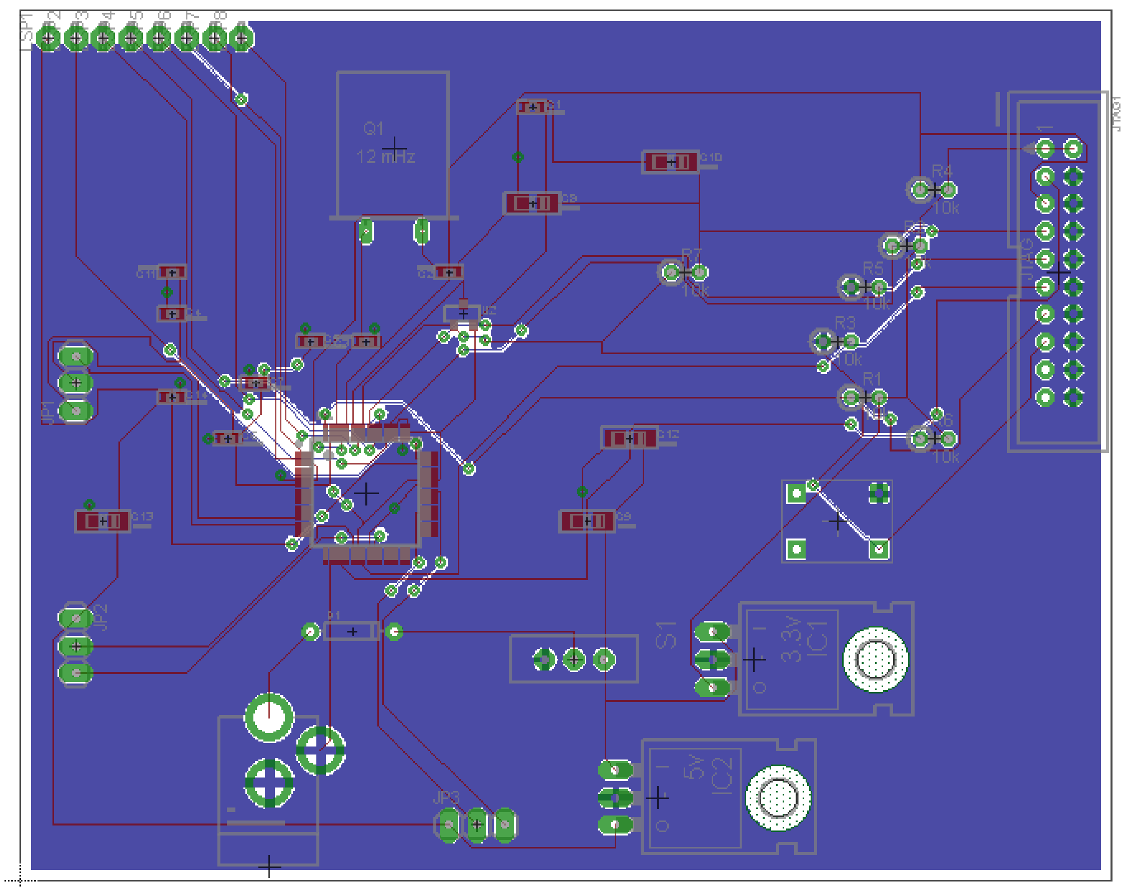

Figure 5.9-1
Main PCB Circuit Layout

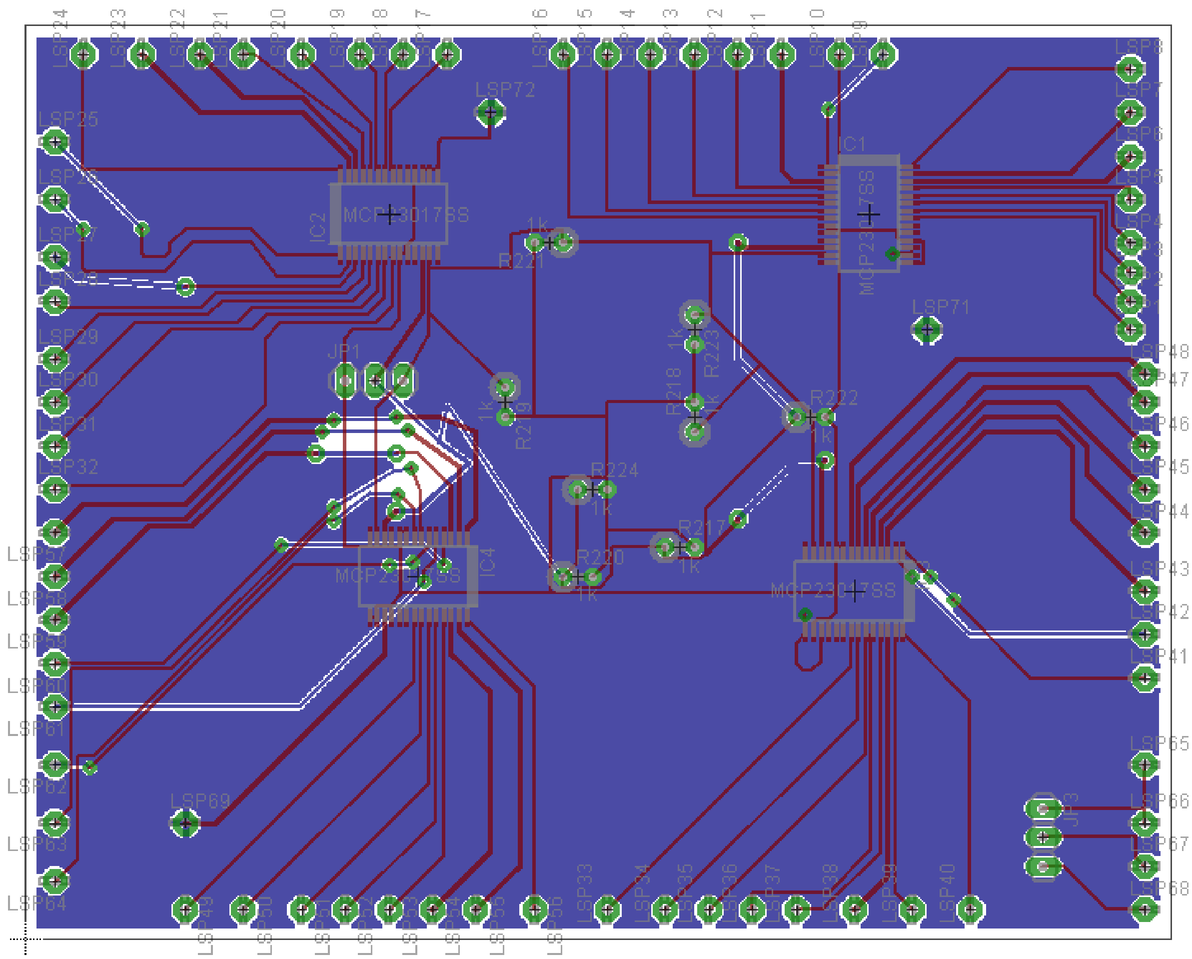

Figure 5.9-2
I/O Expander PCB Circuit Layout

6 Design Summary

The design of this project is intended to gracefully intertwine the software and hardware components. Software will be written to the LPC2148 microcontroller that will control the different hardware components. These two facets must work seamlessly together to create a quality prototype.

6.1 Software Design Summary

The LPC2148 microcontroller supports a range of software. Through research, it has been determined that the C programming language will offer this product the most flexibility and performance.

Because the C language does not support the concept of object-oriented classes, a set of six structures will be used to define game piece properties. The core of these property characteristics are the movements algorithms allowed by the six different types of chess pieces.

In the 4.3.1 General Chess Gameplay section of this document, the chess game grid is described as an eight-by-eight array of sixty-four tiles. By treating the game board as a two-dimensional Cartesian coordinate plane, the software can keep track of each piece's position on the board by maintaining a set of thirty-two structure data objects that include an x-y coordinate location to each piece and a boolean to define the piece's liveliness. By referencing a piece's position along a row (rank) using an x-coordinate and a piece's position along a column (file) using a y-coordinate, the software can use a set of predefined algorithms to verify the movements of chess pieces.

It will also be beneficial to maintain a separate eight-by-eight two-dimensional array to mirror the chessboard and its piece locations. By cross-referencing this array with the array of specific piece locations constantly throughout the game, the software will be able to determine when valid moves are made by the players.

6.2 Hardware Design Summary

The hardware has been divided into four subsystems that will be contained on separate boards and connected via headers or wires. The microcontroller subsystem will be on one PCB that will contain the microcontroller, power supply, JTAG connector, and headers that will go to the other PCBs. The second subsystem on a PCB will be the I/O Expander subsystem that is used to interface with the QRD1114 sensors. The last PCB will contain the LED Driver subsystem that will be used to interface with the large amount of LEDs. The final subsystem will be contained on a perforated board and will include the matrices of LEDs and sensors. The components will be placed accordingly and wired appropriately on the perforated board and then connected to the appropriate subsystem PCB.

7 Project Prototype Testing

To create a successful prototype there must be an appropriate testing plan. Hardware must be tested to ensure that every component exhibits the properties that the design plan is based upon. A single faulty component could ruin an entire design or make the developers think there is something wrong with perfectly fine software. Software must be tested extensively to ensure that the correct data is recorded, manipulated appropriately, and that it sends the correct outputs to the hardware. If there is a mistake in either the hardware or software side of a prototype it can keep the prototype from being functional.

7.1 Hardware Test Environment

In order to have a successful hardware testing environment, it is necessary that the software is fully tested and debugged before the hardware components can be checked. Successful software implementation is a vital source to drive the correct functions to the microcontroller and its components for the electronic chess board. Software programming, testing and debugging will be done using a JTAG debugger or through the setup of the microcontroller's USB port. Code that is written in C must be translated into machine code done by a IDE, such as Eclipse, which is a free software IDE that is setup to work with the chess board's microcontroller. The JTAG debugger or the USB port will transfer the translated code from the IDE to the microcontroller for use with hardware components. Proper software is necessary so that the hardware components are connected to the corresponding ports in which the testing needs to take place. Separately each port, pin and device will be checked for proper functionality and correctness. It is crucial to keep track of which ports are being used and which ports correspond to what device. Incorrect information may alter the output of the hardware components or alter the programmer's decisions on correct code. Each hardware component will need a different strategy for testing so that the desired output is produced.

Popular hardware testing environments include drawing diagrams and schematics both by hand and computer software, calculating desired results such as voltages and currents so that testing is purposeful and last using a bread board for prototyping before committing permanent hardware components to the chess board system.

Creating diagrams such as schematics and free-hand maps will help with the understanding of the mapping of the devices such as LEDs and keep track of the locations of those devices under test. Programs such as Xilinx and National Instruments Circuit Design can provide the software to allow replication of the chess board system hardware virtually before implementing any real design. Logic of the pins and devices can be depicted here and tested thoroughly before any real implementation is made. Handmade diagrams can be useful while testing. For example, the microcontroller can be simplified by marking with a pen and paper the location and functionality of each port. Correct schematics will allow us to test input and output of the system along with providing other useful information.

It will be beneficial to test voltages and currents using software design tools to match up to human calculations. When matching the human theoretical values to software theoretical values a low percent of error such as less than 5% will be accepted so that testing is done correctly. Many times when there is an error with either theoretical values, that means one of two things, either the schematic was wrong or the human theoretical calculations are off. When either of the two results occur, each process will need to be re-implemented in order to ensure that the testing is valid.

Once diagrams and schematics have been tested, and values are correctly matched, the next step is to test the results using a breadboard in which individual components can be utilized and tested before making any permanent changes. While developing with a breadboard testing will consist of using wires, multi-meters and correct circuit configuration along with the hardware components in order to test the devices properly. Here the theoretical values must correlate to the tested values and desired results must be met. Testing multiple devices will also be important so that there is consistency in results such as with the LEDs. Other testing will be device sensitive and is discussed in the sections below such as magnet sensing, LED/LCD output, and power source testing. It is important to note that not all testing will be done the exact same way.

7.2 Hardware Specific Testing

It is critical to isolate components and test specific functions in hardware to make sure that the device functions properly and will consistently do so. The individual testing of each component will allow the developers to precisely correct any errors since many components and devices rely on each other.

7.2.1 Chess Piece Sensing

In order to test the sensing of the chess pieces the chess board will contain optical detectors and phototransistors that are aligned with each position on the board. Proper testing of these components will allow the board to appropriately detect the chess pieces. Chess piece sensing is crucial for the game to operate correctly. Without sensing the chess pieces successfully the board will essentially only be useful for nonelectric chess games.

Setting up a circuit that will allow the testers to appropriately utilize the simplest form of sensing such as whether the piece is present or not which will be crucial for more complex testing of other components. Having the circuit respond with an output of success will give the developers the promise that at least the board will sense the pieces. After this has been accomplished more rigorous testing can be completed such as playing chess to prove that not only does it sense the piece but it shows available moves to both players based on the results of the game. In the next section there is more detail of how the sensors will be tested for proper gameplay and piece detection.

7.2.1.1 Optical Detectors and Phototransistors Sensing

In order to test the optical detectors and the phototransistors the first thing that needs to be done is setting up a single sensor with the development board. Once the simple circuit is set up the next step is to test this device on different colored surfaces in which information about the level of detection can be determined with addition of the amount of light present for detection. With the optical detectors, the device produces an infrared light that it will use to detect the presence of a chess game piece or the absence of that particular piece. The amount of light reflected back to the optical detector will let the device know what type of information to relay to the system. If there is a piece present, the light reflected back will be much stronger than when there is not a piece in the spot. By testing different surfaces and different colors, it will allow the developers to determine a threshold of a maximum or minimum amount of light to expect to emitted back to the sensor. This will help program the sensors for maximum efficiency for detecting chess pieces, Determining what colors and surfaces work best with the sensors will be important for proper development.

Another important test to help determine the threshold of infrared light present will be testing the device in different operating environments. One scenario will be in a well lit environment such a in the daylight sun, another will be a dim environment and last will be a dark environment to simulate the game being playing at night. Each of these different environments will produce varying levels of infrared light strength that the sensors should produce and detect. This will allow the developers to determine sensor strengths in all possible gaming situations. It will also be just as important to try the sensors with cutouts to simulate the housing that the sensors will be enclosed in after full development. Testing the sensors is an important part of the development of the electronic LED/LCD chess board. Proper piece detection will allow the software and hardware to cooperate and work seamlessly in unison for proper functionality. It will allow the software to dictate to the hardware which device to enable for such functionality as LED lights and LCD display.

7.2.2 LED and LCD Testing

The Knight Light LED Chess Set communicates to the users through a series of lights and displays. It is critical that all of these components work properly, especially the RGB LEDs instead the chess board housing. By isolating each component, its functionality can be tested thoroughly.

7.2.2.1 LED Testing

When testing an LED the circuit setup is just as important as the software that corresponds to the LED. Proper circuitry is important because without having a correct circuit setup there won't be much to observe even if you have absolutely correct code. Setting up the circuit can be sometimes be trivial and other times cause many issues. It is important to understand how the LED will work inside of the chess board before the circuit can be set up. Knowing the LED and some important concepts such as maximum and minimum current, output voltage, and resistance will be very important. All LEDs have two wires coming them. One is the anode which allows current to flow into the LED and the other is the cathode in which the current escapes the LED. Current will need to be set to a specific limit depending on the part that is used. Having a current that is too high will cause the LED to have a short lifetime in which the chess board LEDs will need to be replaced at a much higher rate or it can cause the LED to immediately burn out. The opposite is also true having a current that is too small which will cause the LED to become dim and not shine as bright. For the aesthetic features of the chess board it will look much more presentable having LEDs that are optimized to their full potential brightness. One way of finding the correct LED is by using Ohm's law (R=I/V) to find the correct resistance needed to optimize the LED by letting just the right amount of current through to the LED. A resistor needs to be placed in series with the LED for correct operation. Connecting an LED directly to a power source is sure to ruin the component. So knowing the voltage and current of the LED is important to finding the correct resistor that needs to be combined with the LED. Most of the times the LEDs will come with those values but if need be there are other methods to solve that issue. Next step after all of these value have been discovered is to setup the simple circuit and either start programming the corresponding port and pin to the LED or if the software has already been created, it is as simple as turning on the device and seeing if the LED lights and isn't dimmed.

7.2.2.2 LCD Testing

Testing LCDs is much similar to testing the LEDs except the LCD has more complexity by enabling multiple pins for input/output information between the microcontroller and the LCD. In order to create a successful chess board with an integrated LCD it is best to understand the different pins of the LCD and their functionality. There are two power pins located on the LCD, one which will connect to the power source and the other to the ground, a contrast pin, which will allow programmers to adjust the code to change the LCD of the chessboard, and most of the other pins designated with a D# will be used for controlling the output so that the users can view message to the screen. There are also additional pins such as the optional backlight pin and the ever important read/write pin which will be discussed in detail below.

When testing an LCD is it best to setup the simple circuit using the breadboard much like that of an LED and then connecting it to the microcontroller. Next step is to test the output of the microcontroller with simple programs that test all characters and numbers. Then we test full sentences and see if they are properly translated to screen output. Another thing to note when testing an LCD is that the microcontroller is typically much faster than the LCD so there will need to be delays set for the LCD to in order to allow the LCD to properly read the signals from the program and timely write the output. Testing different time increments will be important so that there isn't a huge delay from signal input to screen output. Much of the testing can be done by setting pins to specific values such as turning the read/write pin on and off and with correct sequences. "Read" sends messages from the LCD to the microcontroller and "write" sends messages from the microcontroller to the LCD. Knowledge of the pins is important so that when testing the the LCD for the electronic chess board it ensures the correct pins are programmed and switched at sequential times.

7.2.3 Power Source Testing

When testing for power one must have a fixed power source that gives the microcontroller and other devices the correct amount of power so that it will function properly. Providing too much power will strain the system and will cause the devices to die much faster or even immediately depending the severity of the over-the-top power source. Providing not enough power may cause other additional problems such as devices working slower or LEDs for instance not to shine as bright. Testing the power source will be important for performance optimization.

Most manufacturers will provide the voltage range in which each device can handle. Setting up the correct circuit with the specific components that help regulate voltage such as resistors will limit the amount of power that enter each device. Meeting the standards of the manufacturers voltage lower and upper limits allows for proper functionality. For example, the LPC2148 microcontroller allows for only a range of voltage from 3.3V to 5.0V with a plus/minus 0.5C on either side.

Testing the circuits components can be as easy as using a voltmeter. A voltmeter should be accessible by the developers at a school lab but there are also companies that rent them for a low cost so there is no need to purchase this testing machinery which can range from $8 dollars for the least expensive up to a couple hundred dollars for the high end accuracy machines. Testing first the power source directly will give a great deal of information for how to setup the circuit. Once it is understood how much power will be constantly supplied, it is then up to the developers to research the amount of power that must be supplied for proper functionality of each device such as the LEDs, the LCD message board and other related components. For instance each microcontroller will have specifications of what type of power source the device can accept whether the power comes from a wall unit, a USB or even battery and solar powered. However regardless of where the power comes from, it is important to test the amount of power supplied to the electronic devices of the chess board.

7.3 Software Test Environment

When testing the software of the LED/LCD chess board it is crucial that all of the developers are on the same page. Understanding different sections of the code, commenting and allowing each developer to own a part of the code will generate a productive testing environment. First thing to consider for a testing environment is to make sure that each software developer has the same IDE (Integrated Development Environment). With most IDE's it is important to choose one that can easily transfer the code that is engineered from the IDE to the microcontroller for functionality of the chess board. There are many great IDE tools available out there which makes the decision much more difficult than just choosing one at random. The IDE needs to compile the language that will be used for the software. In the case of this chess board, C is the standardized language of choice and almost every IDE will compile C. It also helps if the IDE is dedicated to the device so that an environment doesn't need to be created for each component because this process typically isn't trivial when developing with a microcontroller. An ideal IDE is one that is open source, free, easy to use, dedicated and connects to the microcontroller through the JTAG connection.

A common IDE is Eclipse which is widely used from the school environment to the professional setting. Eclipse appears to be a perfect fit for the ideal testing of the LPC2148 microcontroller environment. Not only is Eclipse an IDE that is both open source and free but once a development board is purchased it typically comes with a plugin that is can be used to directly dedicate the Eclipse IDE for the microcontroller programming and testing. By simply downloading the plugin available with the development board, it will make testing the microcontroller and other devices as easy as plugging in the development board to the USB port of the computer being used for development of the software. It will provide visual tools that can assist with the testing and provide insight to which devices aren’t functioning correctly. Most of the testing will be visual and the correct output will determine if the software is appropriate for each device.

Source control is another tool that will be used extensively when testing code. In order to have successful testing and programming this project must incorporate source control. It will enable each user to take copies of the code and develop specific test that may or may not work on a local computer. The reason that source control is key to successful testing is that a developer may make a mistake along the way and without source control the developer can not rollback the changes made and start from the initial code. Source control will also allow the developer to commit changes to the project for each person to upload once a specific device or component has been programmed and tested correctly. Popular source control programs include SVN repository which is already installed in the Eclipse IDE package. Another option is to keep all the code on an external server such as github or by using another host such as Apache Subversion which is an open source tool that can be downloaded from the internet. In order to make things easy, the best choice for source control is using the SVN repository to commit any changes in which the testing was successful for the group to share. One important aspect of source control is only manipulating source code that isn’t being changed by another member of the group. Working in the same file will cause merge conflicts from code that has changed while two or more developers are working on the same lines of code. This issue can often take a long time to correct and can cause much frustration amongst a group of coders and testers.

7.4 Software Specific Testing

The importance of testing software is crucial for the development of all of the components of the chess board. Not only will the chess algorithms fail to produce a successful chess game but if the software isn't tested and debugged there will be issues with the hardware. Having the correct software environment along with testing different components individually will allow the testers and developers to pinpoint errors and correct them accordingly.

7.4.1 Piece Tracking

In order to test the chess piece tracking of the software it is to be assumed that the Optical Sensors are all working properly since much of the hardware testing will come before the software. Some of the hardware and software testing will occur simultaneously if bugs are found along the way. Once the piece sensing is accomplished, meaning that the chessboard senses the presence or absence of a chess piece, it is now time to test the tracking of each piece which comes naturally by playing chess. The software will need to be uploaded to the development board for testing. The algorithms associated with each piece along with the sensors of the board will determine the tracking of the pieces when engaged in a game of chess. The testing will have to prove that multiple games with varying scenarios can produce a successful outcome. Knowledge of how chess is played in addition to how each piece moves will allow for successful testing. The board must recognize the precise moves that a piece can make, where it has moved to and if it has been removed by a move made from the opponent. If the piece is removed than the chess board will no longer track the piece and it will recognize that the piece has been taken by the opponent. If the piece remains on the board tracking will continue and it will also show possible moves that are available due to the precise tracking. Once this testing is complete, the chess board will have a great foundation for future improvements and serve as a base for further functionality such as adding in an A.I component so the user can play chess against a simulated computer opponent. Additional software modifications made need to be made if any issues occur and will all be handled differently depending on the problem at hand during testing.

7.4.2 Artificial Intelligence Tracking

Providing the human chess player with a computer player component will allow the user to play chess against a simulated computer. No longer will the chess player need someone in order to play a game of chess. Testing will be roughly the same as the piece tracking section. Most of the testing will be done by simply playing chess and having a knowledge of how the game is played in order to develop a formidable computer opponent. It is first crucial to test the chess board thoroughly for the tracking of all of the pieces. Once this testing is successful, it is then possible to come up with algorithms for a computer player, in which the user will have to move the opponents pieces either based on predetermined moves or give the user a choice to place the piece where the board shows possible moves designated by LED lights. The user will be able to act out different scenarios to develop a greater sense of the game and learn how to overcome certain obstacles that develop while the game is played against the computer. With the same desired outcome as that of piece tracking, the chess board will have to recognize where each piece has been moved during gameplay, when pieces have been removed and possible moves for both the user and the computer. It seems a majority of this testing is visual and needs to play out with correct chess logic in order to have a successful game against the computer. Additional software manipulation may be needed depending on issues that arise during testing.

8 Administrative

8.1 Senior Design 1 Milestones

For the research portion of this phase, much of the it was done simultaneously with the writing of the research paper. Splitting up the amount of information to research allowed each member to specialize on a specific part of the project and then bring any new knowledge to group for discussion. All of the research allowed the group to get a better understanding of how to attack the problem of designing the chess board for Senior Design II. It is now clear how development will begin and what specific parts are going to be needed to complete the development phase of this project. Below, in Table 8.1, are the details and dates for project milestones during the first semester of the project.

Table 8.1
Senior Design 1 Project Milestones

Project Part	Start Date	End Date	Duration
Research Microcontrollers	9/11/2012	11/11/2012	61 Days
Research Sensors	9/11/2012	11/11/2012	61 Days
Research RGB LEDs and LED Drivers	9/11/2012	11/11/2012	61 Days
Research I/O Expanders	9/11/2012	11/11/2012	61 Days
Research soldering and project design	9/11/2012	11/11/2012	61 Days
Research Input/Output Control and Brainstorm initial algorithm ideas	9/11/2012	11/11/2012	61 Days
Borrow a develop development board and chip for practice	9/25/2012	11/25/2012	61 Days
Research PCB Design	10/5/2012	11/25/2012	51 Days

8.2 Senior Design 2 Milestones

Since none of the group members have built an electronic LED/LCD chess board, it is a little tough to estimate the amount of time it will take exactly to build this project. However, from the research portion of this project it is now much clearer roughly how much time it will take to complete the development of the chess board. For the amount of days to completion, it is better overestimate the days until completion than it is too underestimate the amount of days and not complete the project. The group is planning on coming short on most of the deadlines so that any issues or problems can be addressed with the proper attention needed. Below, in Table 8.1, showing the projected milestones and the amount of days it will take to complete each task.

Table 8.2
Senior Design 2 Project Milestones

Project Part	Start Date	End Date	Duration
Acquire Microcontroller, Sensors, LEDs, development board	1/8/2012	1/30/2012	22 Days
Acquire LED Drivers, I/O Expanders	1/8/2012	1/30/2012	22 Days
Acquire magnets, chess board/pieces, and any other last parts needed	1/8/2012	1/30/2012	22 Days
Microcontroller Programming	1/30/2012	2/13/2012	14 Days
Test Microcontroller	2/13/2012	2/22/2012	9 Days
Test Microcontroller with Sensors and Outputs	2/22/2012	3/3/2012	9 Days
Housing for Final Prototype	2/13/2012	3/3/2012	18 Days

PCB for Prototype	3/3/2012	3/24/2012	21 Days
Assemble Prototype	3/24/2012	4/12/2012	18 Days
Test Prototype	4/12/2012	4/18/2012	6 Days
Fix Prototype	4/18/2012	4/24/2012	6 Days

8.3 Budget and Finances

When considering the budget of the electronic LED/LCD chess board it is important to understand the scope of this project and similar products that are out on the market in order to correctly evaluate the cost that will be eventually absorbed for development. This project needs to be within the realms of something affordable for both the developers and potential customers. This project is not being sponsored as of now by anyone other than the students that are developing the LED/LCD chessboard. All of the developers have decided to keep the chessboard optimized in terms of performance but also for budget. It will take time narrowing down the precise parts and specific combination of parts that will benefit both the performance and budget. This project will not be limited by any finances other than what the developers are willing to spend. A general idea of parts that will be needed in order to create a successful project will allow the developers to assess an estimate of the cost. However, as many projects don't go perfectly the first time it is safe to assume mistakes and additional costs that aren't seen at first glance. Hopefully the additional cost aren't initially underestimated. Continuing research will benefit the overall understanding of the products but it will also help to eliminate extra cost that are unnecessary. Shopping around and comparing products and prices is going to give this project the edge it needs to be notable and affordable. Below in Table 9.3, is a summary of the many needed components for the chess board to operate correctly and efficiently.

Table 8.3
Budget Summary Table

Part	Quantity	Price Range
Microcontroller LPC2148	1	$15
LED / LCD		
RGB LEDs (25 pack)	3	$19.95
LED Drivers - TLC59116	12	$1.63
LCD Monitor	1	$14.00
Sensors / Magnets		
Touch Sensor	2	$2.00 - $3.00
I/O Expander MCP23016	4	$2.00
QRD1114	64	$1.00
Power / Programming		
JTAG Programmer	1	$51.95
USB Cable	1	$4.00
Power Regulator	2	$1.95
Development Board	1	$40.00
Testing / Misc.		
Perforated Board	1	$20.00
Chess Pieces/Board	1	$10.00
Additional Components (resistors, cap, PCB)	N/A	$150.00
Total Estimate		$470

8.3 Project Summary

For development of an LED/LCD electronic chessboard hours of purposeful research was the prerequisite to the formation of the group's understanding of the overall detailed project. It is vital for success in the next phase to have a complete understanding of all aspects of our planned project. From this research each developer was able to get a hands on understanding of the various components. Now it is much more apparent how each part must come together in unison to produce the desired output and product. Whether it was an in-depth look at the hardware components or the software structure each piece was broken down to its core in order to fully comprehend the specs, advantages and challenges. It order to summarize the project it is best to have a look at all of the different components and how they relate to one another. The electronic LED/LCD chess board will consists of two different sides, hardware and software that will control the chess game and all of the features.

Starting with the hardware, the main component is the microcontroller which serves to control devices such as the LEDs, the LCD message board, the sensors, timers, and the power source. The microcontroller is essential for the increased functionality that typical non-electrical chessboards don't provide. The board will have a total of sixty-four LED lights (eight-by-eight board) that will light up depending on the individual chess game and will designate possible moves, the placement of pieces for a continued game, and certain game situations such as when a person is in checkmate. The sensors will help with recognition of whether a piece was picked up or is present on the board and the timers will time each person's game length. The ports of the microcontroller will be designated to operate these devices depending on their functionality and needs such as power output and amount of pins needed. However, none of this will be possible without the software which will serve as the backbone of the all the hardware components.

The chess board software will allow the microcontroller to control each device properly. The microcontroller needs the correct programming in order to organize the functionality of each device that was introduced above. Designating specific pins for the devices and matching up those pins with the programming of the microcontroller is a matter of organization and preparedness. The microcontroller will be programmed in C. It is unfortunate that we can not use object oriented programming but the next best choice C, will provide a great logical foundation for the chess board. In addition to controlling each device, the software will also have complex chess algorithms assigned to each piece so that the LEDs light up correctly for possible moves with the awareness of other pieces on the board and ones that have been removed. Moving on to the next phase of development will require a great deal of detail, focus and time. Coordinating all parts of the chess board will heavily weigh on the research of this phase.

8.4 Conclusion

The research accomplished for Senior Design I will prove to be the essential backbone of the development for Senior Design II and the creation of the Knight Light LED chess board. This group will be implementing the extensive research to create an electronic LED/LCD chess board. All of the group members have been able to obtain beneficial knowledge that will pertain to both this course as well as many other environments outside of the classroom. Senior Design has taught each of the team members valuable information to apply to the creation of an electronic LED/LCD chess board provided that only fundamentals were previously known about creating such a challenging project. The information present in this document is a detailed explanation of the parts, components, processes and devices associated with an LED/LCD chess board. Each of the group members now know the complexity behind the development of the chess board project. It will not be an easy task. In order to keep progressing during the next phase of development it is important that the group continually stays on task and follows the deadlines set out for the project. If deadlines are compromised during the next phase, the risk of not completing the project comes in play. Not meeting deadlines and risking completion of the project is our biggest challenge for the semester however with organization, structure, communication and teamwork this project should go very smoothly. It is going to be a pleasure to put all of the combined knowledge of the group to use and create a successful project.

9 Appendix

9.1 Content Authorization

Authorization by ChessBaron.co.uk

From: baron.turner@gmail.com [baron.turner@gmail.com] on behalf of Baron Turner | ChessBaron [mail@chessbaron.co.uk]
Sent: Monday, November 26, 2012 5:14 PM
To: NickDeSantis54@knights.ucf.edu
Subject: Chess Computers

Nick

I've approved your request for a pic of the Excalibur GM:
http://www.chessbaron.co.uk/gallery/CMD2005/CMD2005_large.jpg

But... as a chess/engineering student, what do you think about helping us with our intention to manuafacture a range of chess computers - particularly one similar to the Citrine: http://www.chessbaron.co.uk/chess-CMD2002.htm. To be of use, you'd need to be pretty hot with C or C++ and a strong chess player yourself. Is that you?

Kind regards

Baron Turner

ChessBaron.co.uk

Authorization by Microchip.com

From: Academic@microchip.com [Academic@microchip.com]
Sent: Wednesday, November 28, 2012 2:43 PM
To: NickDeSantis54@knights.ucf.edu
Subject: RE: MCP23016 Datasheet Image Use

Hello Nick:

Thank you for your email. As I understand it, you should be ok as long as these are for educational purposes. You may wish to have a look at our legal site at www.microchip.com/legal for specifics. If you have any questions, please contact our legal team at:legal.department@microchip.com.

Regards,
Microchip's Academic Program Team
Need more information on Microchip's Academic Program or would like to become a Partner? Visit ***www.microchip.com/academic*** for more information.

From: Nick DeSantis [mailto:NickDeSantis54@knights.ucf.edu]
Sent: Monday, November 26, 2012 1:22 PM
To: Academic
Subject: MCP23016 Datasheet Image Use

Good afternoon, I'm a senior Computer Engineering at UCF and I'm currently working on a report for my senior design class. I wanted to find out how to get permission to use images from the datasheet for the MCP23016 in my report. If you could redirect me to the right person or grant me permission for the use I would greatly appreciate it, thanks!

Authorization by Texas Instruments

Texas Instruments is pleased to provide the information on these pages of the World Wide Web. We encourage you to read and use this information in developing new products.

TI grants permission to download, print copies, store downloaded files on a computer and reference this information in your documents only for your personal and non-commercial use. But remember, TI retains its copyright in all of this information. This means that you may not further display, reproduce, or distribute this information without permission from Texas Instruments. This also means you may not, without our permission, "mirror" this information on your own server, or modify or re-use this information on another system.

TI further grants permission to non-profit, educational institutions (specifically K-12, universities and community colleges) to download, reproduce, display and distribute the information on these pages solely for use in the classroom. This permission is conditioned on not modifying the information, retaining all copyright notices and including on all reproduced information the following credit line: "Courtesy of Texas Instruments". Please send us a note describing your use of this information under the permission granted in this paragraph. Send the note and describe the use according to the request for permission explained below.

Authorization by MadCatz.com

From: Michelle Rodriguez [tsupport@madcatz.com]
Sent: Tuesday, December 04, 2012 11:04 AM
To: nickdesantis54@knights.ucf.edu
Subject: [#44813]: Image use

Hello Nick,

Thank you for patiently waiting! We appreciate you contacting us about this inquiry. There would be no problem if you decided to use our image of the Mephisto Chess Trainer. If you have any further questions or concerns please feel free to ask. Have a great day and best of luck on your report!

Thank you,

-

Michelle Rodriguez
Technical Support

Authorization by SparkFun.com

Product Photos: SparkFun product photos may be used without permission for educational purposes (research papers, school projects, etc.). However, permission must be granted for commercial use and proper credit to SparkFun must be given. For inquiries about the use of our product photos or permission to use them, please contact marketing@sparkfun.com.