TEAM 08 PROPHYLAXIS

Project Proposal and Feasibility Study (PPFS)
12/11/15

Team 08
Alexis Bonnema, Nicholas Bramer, Paul Brouwer,
Derek De Young, Curtis Kortman

Team Advisor
Professor Mark Michmerhuizen

Senior Design Project – Engineering 339 – Fall 2015
© 2015, Calvin College, and Alexis Bonnema, Nicholas Bramer, Paul Brouwer, Derek De Young, Curtis Kortman
Executive Summary:

Chess is an ancient game that has not seen significant change for many years. While there are some chess variations dating back to the 13th century, the version played today was largely determined near the end of the 15th century (Calvo, 1998). This project will show how modern technology can meld with a historic game to enhance the interaction and experience with the game. The group will design and build a prototype of an automated chessboard.

The design of the chessboard could take on many modern influences like bright displays, lots of graphics, or motion and high-speed action. However, the group decided to respect some of the more humble origins of the game, and create an interface that adds vast capabilities to the game, without altering the feel or environment associated with chess. The selected design includes several goals. The first goal is to create a physical chess game where one player can play a computer. Additionally, the board will provide capability for a remote chess player with a cell phone app to play the person sitting with the board. The second main goal is that the chess pieces move without a visible movement mechanism to preserve the traditional look and feel of chess. To achieve a traditional character the chessboard will include a minimalistic interface to start the computerized games and time the turns. Furthermore, the chessboard will be made of wood. Finally, to improve accessibility, a stretch goal of the project would be to add voice control to the chess table.

The team working on the chessboard consists of five Senior Electrical and Computer Engineering majors from Calvin College. The team members are Curtis Kortman, Derek De Young, Nick Bramer, Paul Brouwer, and Alexis Bonnema. Calvin College professor Mark Michmerhuizen serves as the group’s mentor, giving advice and monitoring the group’s progress. Gentex employee Eric Walstra has also generously offered his services as an industrial consultant giving insight into project management.

The group will design and build the prototype by May 2016. The group will have to maintain a budget set forth by the engineering department of $500. Treasurer Nick Bramer judiciously approves purchases to verify features that can be sufficiently implemented, while minimizing fiscal waste. To achieve the goal of implementing the project within the budget, the team has contacted the sales manager, Jim Kortman, at Donald Engineering Inc. who is obtaining many of the parts for the team at discount.

In addition to working on design and construction, the project will include presentations, documentation, reports, and the maintenance of a website. The team has tracked project progress since its inception in September of 2015. Dictated in this report is the detailed explanation of decisions and progress thus far. The completion of a majority of the software required for an artificial intelligence to make moves on the board marks the majority of progress made so far. To date, the scope of the project has remained intact since its start in September. The group must remain diligent in sticking to schedules and identifying risks and roadblocks to implement all of the different desired features. To achieve this goal at present, more research and testing will be done for sensing the position of chess pieces and motion control. Furthermore, the team will begin on the 3D model of the physical board while it is waiting for more parts to arrive.
Table of Contents

1. Introduction .................................................................................................................. 1
   1.1. Project Background ............................................................................................... 1
   1.2. Team Description ................................................................................................. 1
       1.2.1. Alexis Bonnema ......................................................................................... 2
       1.2.2. Nicholas Bramer ....................................................................................... 2
       1.2.3. Paul Brouwer .............................................................................................. 2
       1.2.4. Derek De Young ......................................................................................... 2
       1.2.5. Curtis Kortman ........................................................................................... 3

2. Project Management .................................................................................................... 4
   2.1. Team Organization .............................................................................................. 4
       2.1.1. Software Team ............................................................................................ 4
       2.1.2. Board and Pieces Team ............................................................................ 4
       2.1.3. XY-Table Team .......................................................................................... 4
   2.2. Schedule and Work Hours .................................................................................. 4
   2.3. Budget ................................................................................................................ 4
   2.4. Method of Approach .......................................................................................... 7

3. System Design ............................................................................................................ 9
   3.1. System Requirements ......................................................................................... 9
       3.1.1. Software Requirements ............................................................................ 9
       3.1.2. Hardware Requirements .......................................................................... 9
       3.1.3. Performance Requirements ..................................................................... 9
   3.2. System Architecture ......................................................................................... 10
       3.2.1. Physical Architecture ............................................................................... 10
   3.2.2. Software Architecture ................................................................................. 11

4. Christian Perspective .................................................................................................. 15
   4.1. Biblical Perspective ........................................................................................... 15
   4.2. Design Norms .................................................................................................... 15
       4.2.1. Caring ......................................................................................................... 15
       4.2.2. Trust ........................................................................................................... 15
       4.2.3. Integrity .................................................................................................... 16

5. Business Plan .............................................................................................................. 17
   5.1. Mission for the Company .................................................................................. 17
   5.2. Potential Customers & Target Market .............................................................. 17
   5.3. Competitive Strategy ....................................................................................... 17
   5.4. Market Sizes & Trends ..................................................................................... 17
   5.5. Business Strategy & Distribution ..................................................................... 17
   5.6. SWOT Analysis ................................................................................................. 18
       5.6.1. Strengths .................................................................................................... 18
       5.6.2. Weaknesses ............................................................................................... 18
       5.6.3. External Opportunities ............................................................................. 18
       5.6.4. External Threats ........................................................................................ 18
   5.7. Uniqueness & Existing/Potential Competitors .................................................. 18

6. Design Alternatives and Selection ............................................................................. 20
   6.1. Chess-piece Movement ...................................................................................... 20
       6.1.1. General Requirements .............................................................................. 20
       6.1.2. Alternatives ............................................................................................... 20
       6.1.3. Selection .................................................................................................... 20
6.2. Microcontroller ...........................................................................................................21
  6.2.1. General Requirements .........................................................................................21
  6.2.2. Alternatives .........................................................................................................21
  6.2.3. Selection ...............................................................................................................21
6.3. Board ..........................................................................................................................23
  6.3.1. General Requirements .........................................................................................23
  6.3.2. Alternatives .........................................................................................................23
  6.3.3. Selection ...............................................................................................................23
6.4. Piece Tracking ............................................................................................................24
  6.4.1. General Requirements .........................................................................................24
  6.4.2. Alternatives .........................................................................................................24
  6.4.3. Selection ...............................................................................................................24
7. Integration and Testing .................................................................................................26
  7.1. Reed Switches ..........................................................................................................26
    7.1.1. Single Reed Switch Testing ..............................................................................26
    7.1.2. Reed Switch Array Testing ............................................................................26
  7.2. Electromagnet ..........................................................................................................27
  7.3. XY-Table Motors ......................................................................................................28
    7.3.1. Sequential Motor Movement ..........................................................................28
    7.3.2. Parallelized Motor Movement ...........................................................................29
  7.4. Software ...................................................................................................................30
8. Conclusion ......................................................................................................................31
  8.1. Potential Risks & Issues ........................................................................................31
  8.2. Project Status ..........................................................................................................31
  8.3. Project Feasibility ....................................................................................................32
  8.4. Summary ..................................................................................................................32
9. Acknowledgements .......................................................................................................33
10. References ....................................................................................................................34
  10.1. Data Sheets ............................................................................................................34
11. Appendices ..................................................................................................................35
  11.1. Appendix 1: Gantt Chart ......................................................................................35
  11.2. Appendix 2: Hourly Work Breakdown ..................................................................36
Table of Figures

Figure 1 - Senior Design Team 08 .................................................................1
Figure 2 - Remaining Balance after each Purchase ........................................5
Figure 3 - Multiplayer Architecture including the Web API ................................10
Figure 4 - Physical Hardware Block Diagram ...............................................11
Figure 5 - Software Hierarchy ......................................................................13
Figure 6 - Reed Switch Array Test Circuit Diagram .......................................27
Figure 7 - Loop-based step control ................................................................28
Figure 8 - PWM with Interrupts Code Architecture .......................................29
Figure 9 - Project breakdown showing hours worked by team members ..........36
Figure 10 - Percentage of hours spent on different project parts .....................37
Figure 11 - Running sum of the work completed separated by part of project ......37

Table of Tables

Table 1 - Listing of Donated Items and Values ...............................................5
Table 2 - Detailed List of Ordered Parts ..........................................................6
Table 3 - Decision Matrix for Project Management Software ..........................7
Table 4 - Decision Matrix for the Selection of Piece Movement Mechanism ....20
Table 5 - Decision Matrix for Selecting a Microcontroller ...............................21
Table 6 - Decision Matrix for Piece Tracking ...............................................24
Table 7 - Parallelized Motor Tests ..................................................................30
Table 8 - Hourly Break down of Projects .....................................................36
1. Introduction

1.1. Project Background
The Senior Design Project is part of Engineering Courses 339 and 340: the capstone courses in the Engineering program. For the design project, a team of three to five members work together to develop a solution to a design problem. Students choose their own design project. Team 08 chose to design and prototype an automatic chessboard with artificial intelligence (AI) capabilities.

1.2. Team Description
The name chosen for this team is Prophylaxis. Prophylaxis is a term used in chess, meaning "to thwart an opponent's plan before it has begun." The Prophylaxis team consists of five senior engineering students of the electrical & computer concentration. The team based the idea of an automated chessboard on the combination of their desires to work with a physical system and with a system that was internet connected. The members of the team are pictured below in Figure 1.

Figure 1 - Senior Design Team 08
1.2.1. Alexis Bonnema
Alexis Bonnema is a senior studying Electrical/Computer Engineering at Calvin College and pursuing a minor in Mathematics. She grew up in South Holland, Illinois, a south suburb of Chicago. This past summer, she worked at Calvin College in a research position concerning the highly adjustable power supply for an array of high-power LEDs. This included opportunities for working with printed circuit board design and assembly as well as microcontroller software.

Alexis is currently on the leadership team for the college's chess club, and also enjoys ballroom dancing, writing, crocheting, and board and video games with friends.

1.2.2. Nicholas Bramer
Nick Bramer is a senior at Calvin College studying Electrical/Computer Engineering as well as pursuing a Computer Science Minor. During the summer of 2014, he worked at Gentex on the production line in the bending area, which led into a summer internship in 2015. During the internship, he had the opportunity to program ABB robots, write VB and jscript code, work with Citect and Iconics graphics, wire panels, and do ladder logic programming with PLCs.

In his spare time, Nick spends a great deal of his time playing soccer, as well as furthering his knowledge of software languages, and different operating systems.

1.2.3. Paul Brouwer
Paul Brouwer is a senior Electrical & Computer Engineering major at Calvin College, and is also pursuing minors in Mathematics and Computer Science. During the summer of 2014, Paul worked at Steelcase, Inc's Innovation Management Office, researching how to apply the Internet of Things to a manufacturing setting. In the summer of 2015, he researched with Dr. Larry Pileggi at Carnegie Mellon University, developing a non-volatile D-latch using magnetic tunnel junctions.

Outside of engineering, Paul has taken piano lessons for 16 years and continues to do so at Calvin. He has been involved with the Calvin Gospel Choir, and he currently sings with the Oratorio Society. His hobbies also include playing tennis and looking into video game development.

1.2.4. Derek De Young
Derek De Young is a senior at Calvin College studying Electrical/Computer Engineering. During the summer of 2014 and 2015, he had the opportunity to intern at Federal-Mogul Powertrain. There he had the opportunity to work on and optimize multiple databases dealing with the lifecycle of their products, as
well as integrating their design software into an existing database. He also is passionate about web and mobile app development.

In his spare time, he enjoys playing intramural sports such as basketball and volleyball, as well as researching new and upcoming technologies and learning more about web development.

1.2.5. Curtis Kortman

Curtis Kortman is a senior at Calvin College studying Electrical/Computer Engineering. During the past summers, he had the opportunity to work at Johnson Controls, (now Visteon), and perform research at Calvin College regarding building energy efficiency and the Internet of Things. He has worked on several projects including designing an audio circuit for dash chimes, designing an automated test rig for testing automotive batteries, and implementing a sensor network at Calvin College to monitor energy usage.

During his spare time, Curtis enjoys being outside and active while fishing, hunting, biking, or hiking. He also enjoys working with his hands on different projects around the house, as well as transportation like mopeds and bikes.
2. Project Management

2.1. Team Organization
With a team of five members, it was imperative that the team communicated early who was going to be working on what part of the project. To accomplish this the team divided into three main sections.

2.1.1. Software Team
The software team is composed of Derek De Young and Nicholas Bramer. It is their duty to get the upper level code working. This includes the tasks of interfacing with the chess engine and sending commands to the lower level classes to get the current state of the board, move the motors, and turn on the magnet. Along with the physical board’s code, the software team is also working on an iOS app and a web application programming interface (API) so players can play against the chessboard from anywhere in the world.

2.1.2. Board and Pieces Team
Paul Brouwer and Alexis Bonnema worked on interfacing with the physical board and magnets. They worked on testing reed switches, magnets, and electromagnets, and researched how to correctly wire and use an array of reed switches to determine the location of pieces on the board. In addition, the team determined the necessary parameters for building a chessboard and pieces with embedded magnetics.

2.1.3. XY-Table Team
Curtis Kortman took charge of designing and building the XY-Table. It is his duty to get the code working that interfaces the motors to the upper level code written by the software team. He then has to develop the 3D model of the XY-Table so he can machine and assemble the different components to make the physical finished table.

2.2. Schedule and Work Hours
Appendix 1 shows a Gantt chart with the proposed schedule that Team 08 is following. Appendix 2 shows the hourly work breakdown for the team as well as the total hours worked on different aspects of the project.

2.3. Budget
The team’s assigned budget is $500. In a desire to be good stewards of the resources available, the team desires to adhere to this amount. To achieve this goal the team has made contact with industrial fluid power and motion company Donald Engineering. So far, the company has provided them with stepper motors, drivers, linear rail systems, and motor mounts free of charge. The donated items have amounted
to a significant amount of money, delineated in Table 1. Figure 2 shows the remaining balance of Team 08’s budget.

Table 1 - Listing of Donated Items and Values

<table>
<thead>
<tr>
<th>Items</th>
<th>Qty</th>
<th>Price/Piece</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>Nema 23 Closed Loop Stepper Motor</td>
<td>1</td>
<td>$257.00</td>
<td>$257.00</td>
</tr>
<tr>
<td>Nema 23 Stepper Motor</td>
<td>1</td>
<td>$42.00</td>
<td>$42.00</td>
</tr>
<tr>
<td>MBC12101 Stepper Motor Drivers</td>
<td>2</td>
<td>$93.00</td>
<td>$186.00</td>
</tr>
<tr>
<td>linear Rails</td>
<td>3</td>
<td>$65.00</td>
<td>$195.00</td>
</tr>
<tr>
<td>Carriages</td>
<td>4</td>
<td>$42.00</td>
<td>$168.00</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td></td>
<td></td>
<td><strong>$ 848.00</strong></td>
</tr>
</tbody>
</table>

Figure 2 - Remaining Balance after each Purchase

See Table 2 for a more detailed list of the budget.
Table 2 - Detailed List of Ordered Parts

<table>
<thead>
<tr>
<th>Date</th>
<th>Team member</th>
<th>Description</th>
<th>Debit</th>
<th>Credit</th>
<th>Balance</th>
</tr>
</thead>
<tbody>
<tr>
<td>9/27/15</td>
<td>All</td>
<td>Beginning Balance</td>
<td>0.00</td>
<td>500.00</td>
<td>$500.00</td>
</tr>
<tr>
<td>9/30/15</td>
<td>Paul</td>
<td>Sensor Reed Switch SPST</td>
<td>6.93</td>
<td>0.00</td>
<td>$493.07</td>
</tr>
<tr>
<td>9/30/15</td>
<td>Paul</td>
<td>Sensor Reed Switch SPST</td>
<td>8.77</td>
<td>0.00</td>
<td>$484.30</td>
</tr>
<tr>
<td>9/30/15</td>
<td>Paul</td>
<td>Diode Gen Purp 110V 200 mA DO35</td>
<td>1.84</td>
<td>0.00</td>
<td>$482.46</td>
</tr>
<tr>
<td>9/30/15</td>
<td>Curtis</td>
<td>5.6lbs DC 12V Holding Electromagnet Lift Solenoid</td>
<td>5.51</td>
<td>0.00</td>
<td>$476.95</td>
</tr>
<tr>
<td>9/30/15</td>
<td>Paul</td>
<td>Reed Switch SPST 7-30AT</td>
<td>0.62</td>
<td>0.00</td>
<td>$476.33</td>
</tr>
<tr>
<td>9/30/15</td>
<td>Paul</td>
<td>Reed Switch SPST 10-50AT</td>
<td>0.50</td>
<td>0.00</td>
<td>$475.83</td>
</tr>
<tr>
<td>9/30/15</td>
<td>Paul</td>
<td>Reed Switch 200V 10W-20-30AT</td>
<td>0.74</td>
<td>0.00</td>
<td>$475.09</td>
</tr>
<tr>
<td>9/30/15</td>
<td>Paul</td>
<td>Reed Switch Molded 200V 25-40AT</td>
<td>0.89</td>
<td>0.00</td>
<td>$474.20</td>
</tr>
<tr>
<td>9/30/15</td>
<td>Paul</td>
<td>Reed Switch 200V 10W-10-15AT</td>
<td>0.60</td>
<td>0.00</td>
<td>$473.60</td>
</tr>
<tr>
<td>9/30/15</td>
<td>Paul</td>
<td>Magnet 1/4&quot; DIA X 1/16&quot; Thick</td>
<td>0.21</td>
<td>0.00</td>
<td>$473.39</td>
</tr>
<tr>
<td>9/30/15</td>
<td>Paul</td>
<td>Magnet 1/4&quot; DIA X 1/8&quot; Thick</td>
<td>0.38</td>
<td>0.00</td>
<td>$473.01</td>
</tr>
<tr>
<td>9/30/15</td>
<td>Paul</td>
<td>Magnet 1/2&quot; DIA X 1/8&quot; Thick</td>
<td>1.28</td>
<td>0.00</td>
<td>$471.73</td>
</tr>
<tr>
<td>10/30/15</td>
<td>Paul</td>
<td>DC 24V Holding Electromagnet Lift Solenoid</td>
<td>9.98</td>
<td>0.00</td>
<td>$461.75</td>
</tr>
<tr>
<td>10/30/15</td>
<td>Paul</td>
<td>11LB DC12V Holding Electromagnet Lift Solenoid</td>
<td>6.83</td>
<td>0.00</td>
<td>$454.92</td>
</tr>
<tr>
<td>11/7/15</td>
<td>Curtis</td>
<td>DC Switching Converter Power Supply</td>
<td>15.58</td>
<td>0.00</td>
<td>$439.34</td>
</tr>
<tr>
<td>11/7/15</td>
<td>Curtis</td>
<td>Raspberry Pi 2.0</td>
<td>38.72</td>
<td>0.00</td>
<td>$400.62</td>
</tr>
<tr>
<td>11/7/15</td>
<td>Curtis</td>
<td>EasyAcc Micro USB Charger</td>
<td>8.99</td>
<td>0.00</td>
<td>$391.63</td>
</tr>
<tr>
<td>11/7/15</td>
<td>Curtis</td>
<td>Samsung 16GB Micro SDHC</td>
<td>7.64</td>
<td>0.00</td>
<td>$383.99</td>
</tr>
<tr>
<td>11/7/15</td>
<td>Curtis</td>
<td>Adafruit: Bluetooth 4.0 USB Module (v2.1 Back-Compatible)</td>
<td>11.95</td>
<td>0.00</td>
<td>$372.04</td>
</tr>
<tr>
<td>11/27/15</td>
<td>Curtis</td>
<td>Open Ended 6mm Width GT2 Belt</td>
<td>1.80</td>
<td>0.00</td>
<td>$370.24</td>
</tr>
<tr>
<td>11/27/15</td>
<td>Curtis</td>
<td>2GT 20 Tooth 6.35mm Bore Pulley</td>
<td>1.85</td>
<td>0.00</td>
<td>$368.39</td>
</tr>
<tr>
<td>11/27/15</td>
<td>Curtis</td>
<td>UM GT2 Pulley 20 Tooth 5mm or 6mm Bore</td>
<td>0.40</td>
<td>0.00</td>
<td>$367.99</td>
</tr>
<tr>
<td>11/27/15</td>
<td>Curtis</td>
<td>Idler Pulley Post 6mm</td>
<td>0.20</td>
<td>0.00</td>
<td>$367.79</td>
</tr>
<tr>
<td>11/27/15</td>
<td>Curtis</td>
<td>Timing Belt Tensioner Spring</td>
<td>1.80</td>
<td>0.00</td>
<td>$365.99</td>
</tr>
<tr>
<td>11/27/15</td>
<td>Curtis</td>
<td>Toothed clamp for open ended belt</td>
<td>0.90</td>
<td>0.00</td>
<td>$365.09</td>
</tr>
<tr>
<td>11/27/15</td>
<td>Curtis</td>
<td>KR13 Needle Roller Bearing</td>
<td>0.65</td>
<td>0.00</td>
<td>$364.44</td>
</tr>
<tr>
<td>11/27/15</td>
<td>Curtis</td>
<td>Limit Switch with Roller</td>
<td>0.65</td>
<td>0.00</td>
<td>$363.79</td>
</tr>
<tr>
<td>12/9/15</td>
<td>Curtis</td>
<td>Arduino Nano w/ usb cable</td>
<td>7.99</td>
<td>0.00</td>
<td>$355.80</td>
</tr>
<tr>
<td>12/10/15</td>
<td>Curtis</td>
<td>Bluetooth Low Energy (BLE 4.0) - nRF8001</td>
<td>19.95</td>
<td>0.00</td>
<td>$335.85</td>
</tr>
<tr>
<td>11/7/15</td>
<td>Curtis</td>
<td>Adafruit Bluetooth 4.0 USB Module</td>
<td>11.95</td>
<td>0.00</td>
<td>$323.90</td>
</tr>
</tbody>
</table>
2.4. Method of Approach

Before work began, the team chose a design methodology. The team evaluated many different project management software packages, and decided on using Asana as seen in Table 3. Asana’s philosophy is that one person is accountable for each task. In this way, no task is lacking clear ownership, and so its completion is not jeopardized by a lack of understanding of who is in charge. Team 08's design methodology consisted of dividing the work into different smaller projects between all of the members of Team 08. Each person took charge of one aspect of the entire project, and within the project, one person took charge of each sub task. Section 2.1, Team Organization, details the division of roles.

Each week the members of Team 08 meet to discuss progress of each of the sections of the project and progress of the project as a whole. Communication outside of meetings occurs through Asana. The team also utilizes Asana to create a schedule with due dates for small tasks and entire projects. Asana can give great reminders to team members to make sure projects stay on track. For communicating documents, OneDrive stores all documents so that team members can work on the reports and other files simultaneously. In addition to these programs, GitHub provides version history and merge capabilities to store all of the team’s code in an effective manner. This program ensures the safety of the team’s code, and allows the team to return to previous versions in the case anything gets broken (DeYoung, Derek; Bramer, Nick, 2015).

Research is another big aspect of the project. Much of the research consisted of simply testing parts; however, for the more expensive parts, such as the microcontroller, a great deal of online research occurred before buying the parts. As said before, the research method used for the cheaper parts, such as the electromagnet and reed switches, consisted of buying a variety of parts and testing them under different conditions, in order to see which part best-suited Team 08's design needs.

Throughout all of Team 08’s, design work, Christian perspective and design norms held great importance in deciding methods of approach and implementation. As seen in the Christian Perspective section in this report, Colossians 3:17 gives the commandment to do everything in the name of Jesus Christ, "giving
thanks to God the Father through him." Team 08 took this calling seriously throughout the design of this project and made sure to incorporate this calling into every decision and methodology. This calling had a great impact on the way that Team 08's members communicated with each other, as communication was done in a respectful and loving manner. In addition to this calling, the design norms of integrity, trust, and caring played a great role in the methodology of this project. The caring design norm pushed Team 08 to make decisions based on the customer's needs and wants out of the chess board, while the trust design norm pushed Team 08 to create a quality chess board that would work 100% of the time, or at least very close. The integrity design norm pushed Team 08's members to be honest with each other, while also being honest with the customer about the progress of the project and problems that occurred throughout the project. The integrity design norm lined up with Team 08's calling seen in Colossians 3:17, and had the same effect on the methodology of the project as well.
3. System Design

3.1. System Requirements
The system that the team develops should not be unnecessarily complex, and contain no vestigial pieces. In the end, the team strives for the simplest implementation of the following system requirements of both software and hardware.

3.1.1. Software Requirements
The team chose to follow the following four table objectives:

- The table shall determine the current location of the chess game pieces.
- The table should determine the player’s desired move via voice
- The table shall allow chess to be played against artificial intelligence
- The table should provide a web API and mobile application for multiplayer chess

3.1.2. Hardware Requirements
The team chose to follow the following three hardware objectives:

- The hardware shall determine if a piece is on top of each square of the game board.
- The hardware shall move chess pieces around the chessboard
- The hardware should convert speech to electrical signals

3.1.3. Performance Requirements
The board should be able to move pieces at a reasonable speed, where completing a move takes no longer than 2.5 seconds for a non-capturing move and no longer than 6 seconds for a move where a piece is captured. The speed will also be slow enough to ensure reliable performance.

When moving pieces, the board should not make noise above the level of hushed conversation. In this way, noise will not interfere with the players’ enjoyment or focus on the game. Power consumption will be limited to a reasonable amount and the mechanics of the board will function reliably.

Voice recognition will be powerful enough to allow players to speak at a rate nearly as fast as standard speech while still providing accurate translation. The app will have a delay under 2 seconds when sending a desired move to the board.
3.2. System Architecture

The architecture to achieve the aforementioned requirements is detailed in a system overview below. The architecture is divided into two areas: software and hardware. Even though the software and hardware are intertwined, they can be thought of separately. The system architecture also includes a web application programming interface (API,) which will be used for a multiplayer chess game through phone applications. Although the web API and phone applications may be a part of the system architecture, they are not required in order to play chess. Figure 3 shows the architecture including the web API game server and phone applications.

![Multiplayer Architecture including the Web API](image)

**Figure 3 - Multiplayer Architecture including the Web API**

3.2.1. Physical Architecture

The components that constitute the physical architecture include a microprocessor, an XY-Table, a playing surface, a piece-sensing mechanism, a button panel, one or more microphones, and power supplies. Each of these is detailed below, noting how they interact within the overall architecture. A graphical summary is portrayed below in Figure 4; refer to Figure 4 to understand how the components listed below are connected.
3.2.1.1 Raspberry Pi
The Raspberry Pi microcontroller connects the otherwise disparate components of the design. The Raspberry Pi controls the motor drivers, electromagnet, and the LED lights. By controlling the motor drivers and electromagnet, the Raspberry Pi controls the movement of the chess pieces. The microprocessor also reads in the Reed Switch Array and inputs from the user such as voice and button presses to start a new game and select the game mode.

3.2.1.2 Motor System
The motor system will be able to move an electromagnet underneath the playing surface in order to move pieces around the board. In order to accomplish this, motor one will move the electromagnet in the x direction, and motor two in the y direction. The motor drivers controlled by the Raspberry Pi will control and power the two motors.
3.2.1.3 **Electro Magnet**

The electromagnet will be used to drag the pieces around the board. The electromagnet’s power will be turned on and off by the Raspberry Pi. 12V will power the electromagnet. The team will have to ensure the magnet does not retain any residual magnetic field that will interfere with pieces or reed switches when not in use.

3.2.1.4 **Playing Surface**

In order to play the game of chess, a 64 square (8x8) board is needed, as well as room on the sides to put captured pieces. The squares should be sized near regulation size, which is 2 ¼”, making sure there is enough room for the motor system and the electromagnet to move pieces past each other. The playing surface will cover over everything in the block diagram of Figure 4 (above).

3.2.1.5 **Reed Switch Array**

For the Chess engine to interpret a human move, it will track the pieces on the board. To track the pieces we will track whether or not a piece is currently on top of a square at the beginning and end of each turn. A piece’s presence on top of a square is detected by the reed switch array. A high signal represents the presence of a piece. A low signal represents the absence of a piece. The reed switch array outputs its data to the Raspberry Pi.

3.2.1.6 **Button Panel**

Due to the variety of playing options the board will be capable of, as well as chess-related options such as pawn promotions, buttons will be needed for users to select game modes and decide what piece to promote a pawn to that has crossed to the other side of the board. These buttons will be directly connected to the microprocessor.

3.2.1.7 **Microphone(s)**

One or more microphones will be required in order to implant voice-activated piece movement. These will only record human speech as directed by button presses (on the button panel), and will connect directly to the microprocessor.

3.2.1.8 **Power Supplies**

Several different supply voltage levels will be required in order to power the Raspberry Pi (5V), motors (24V), and electromagnet (12V). The power supplies will be plugged into a wall for operation.

3.2.2 **Software Architecture**

This section discusses the architecture and structure of the software. The team used a combination of open source software and custom written software. The open source software includes the chess engine. The
code that was written by the team includes modifications to the chess engine, classes for interfacing with hardware, the XY-Table class and the chess table class that encapsulates all of the code for the physical chess table. The overall hierarchy of the code classes can be seen in Figure 5. The orange colored boxes seen above represent classes that are interacting directly with hardware.

![Figure 5 - Software Hierarchy](image)

3.2.2.1 Chess Engine
The chess engine chosen is a lightweight open source command line chess game (Buffett, n.d.). The program has the capability of allowing a user to play a game of chess against an artificial intelligence (AI) at multiple levels of difficulty. This artificial intelligence will still be the backbone to the automated chessboard. Multiple features were added to the chess engine to allow for better piece tracking as this was key to allowing physical movements on the chessboard.

3.2.2.2 Hardware Interfacing Classes
Multiple classes were written to interface between the hardware and the chess game. These classes are imperative to the success of the project as they connect the hardware to the software that is running the chessboard.

3.2.2.2.1 Motor Class
The motor class takes a command that turns the motor shaft either clockwise or counter clockwise for a given distance.

3.2.2.2.2 Magnet Class
The magnet class takes a command that enables/disables the electromagnet. These are referred to as a grab command and a release command.
3.2.2.3 **Board Class**

The board class is able to read the current status of the board by reading the array of reed switches that are placed under the board. It will provide the upper levels of code with a matrix of ones and zeroes telling what square of the board has a piece on it.

3.2.2.3 **XY-Table Class**

The XY-Table class encapsulates the motor and magnet classes to provide an easy interface of communicating to the XY-Table. This class provides the ability to go to any coordinate on the XY axis, as well as the ability to grab and release the magnate.

3.2.2.4 **Chess Table Class**

The chess table class will encapsulate the XY-Table class as well as all the user input classes to the board such as switches and buttons (TBD). This class will be the main interface between the physical board and the chess game and artificial intelligence. The class is able to take move commands generated from the chess engine to control the board as well as respond by reading the current state of the board and returning a move to send back into the chess engine.
4. **Christian Perspective**

4.1. **Biblical Perspective**

Colossians 3:17 says “And whatever you do, whether in word or deed, do it all in the name of the Lord Jesus, giving thanks to God the Father through him.” This verse presents the team’s approach to designing and prototyping the chessboard. Through every step in the design process, the team aims to work towards designing a product that will present a Christian perspective. Although this product does not solve an existing problem, it provides a form of recreation that is environment friendly in design and functionality.

4.2. **Design Norms**

The Calvin College Engineering Department encourages the integration of design norms into every project that the students encounter during their college career and to continue this integration into their work careers. These design norms are not necessarily the “Christian” way to do engineering but help raise questions in the design process that require the use of the Christian faith.

4.2.1. **Caring**

The design norm of caring must be considered all throughout the project process. Fundamentally, caring means to not only think of others but also to act on kind thought and put words into practice.

For a project involving an automated chessboard, caring is most readily seen in the potential of such a product to provide for those who have a hard time finding chess opponents. This product allows such players to battle wits with a fully functional local artificial intelligence or a remote player using the mobile app. In addition, through voice control or an app, this board will make the game of chess more accessible to those with disabilities who are less inclined or unable to move pieces by hand.

Caring also involves creating a product that is aesthetically pleasing and intuitive to use. Elements of caring are even found in such details as the weight and size of the board and the noise (or hopefully lack thereof) generated when the board moves pieces.

4.2.2. **Trust**

Trust is a key element that customers look for, but trust as a design norm involves not only the marketing of the final product but also the production process. Transparency is a key element of trust, and transparency is required concerning the final product and the process it took to get there.
The idea of an automated chessboard instantly leads to questions relating to trust: Will it handle all legal chess moves? Can the board move pieces move fast enough to play with a turn clock? Is the electromagnet reliable? Will the board make reasonably intelligent moves? Questions such as these will be answered outright, with data to back up the claims. Information on how decisions about the design were made will also be disclosed.

4.2.3. Integrity

Integrity as a design norm invokes the idea of delightful harmony in all aspects of a design. A design that displays integrity is one that is aesthetically pleasing, joyful to use, and is holistic in that the entire design works together and displays a beauty of form and function.

For an automated chessboard, integrity can first be seen in the visual appearance of the board and its pieces, as they ought to complement each other harmoniously. The integration of buttons and other features must also display uniformity of style, and voice control will need to be intuitive and enjoyable for every user.

The movement of pieces shall display integrity through consistency of movement speed and reasonable accelerations and decelerations when starting and stopping a move. The movement speed shall be fast enough to play chess with a turn clock, but not too fast as to be off-putting. Integrity will also be shown through app design by creating an intuitive user interface, understandable by both the youth and the aged.
5. Business Plan

5.1. Mission for the Company
Prophylaxis’ mission is to create the most immersive chess experience available with a physical chessboard.

5.2. Potential Customers & Target Market
Prophylaxis is targeting novice and expert chess players alike. With this system, novice chess players would be able to increase their skills by playing against various levels of the chess engine. For expert chess players, it can be hard to find a competitor who is at the same talent level as they are. The Prophylaxis system would let the expert player play against an excellent chess engine or an online player while still using a physical chessboard. Prophylaxis’ board could also be used for the physically handicapped giving them the ability to speak to the board to dictate moves without the need to physically move the chess pieces.

5.3. Competitive Strategy
Prophylaxis seeks to compete based on differentiation. As there is no large presence in the market for automated chessboards, competition based on cost is not a consideration. From the few boards that do have built in chess engines, the team seeks to differentiate their product by physically moving the chess pieces, as well as having the ability to dictate moves and play via mobile application.

5.4. Market Sizes & Trends
The market of smart board games is very small compared to other gaming markets. Prophylaxis is trying to bring more attention to the market by bridging the gap between virtual game board games on mobile apps the physical board games. Prophylaxis plans to release an iOS app that is a fully featured, multiplayer chess game. The app will be able to connect with the Prophylaxis chessboard, which may bridge the gap between virtual and physical gaming.

5.5. Business Strategy & Distribution
Prophylaxis will distribute their product in two ways. First, the board will be available via internet sellers such as Amazon, and shipped to customers using their services. Second, the board will be shown at chess tournaments and tech shows; this will show off the board to niche markets to gain interest. The combination of these two distribution channels should provide Prophylaxis with enough sales capacity.
5.6. **SWOT Analysis**

5.6.1. **Strengths**
The Prophylaxis automated chessboard will have the capability of automatically moving chess pieces via voice control, via an app using a web server, with artificial intelligence, and through manual movements. In addition, the Prophylaxis will be aesthetically pleasing to the user, and will allow the user to choose which features of the Prophylaxis the user wants to use, in order to better suit the user’s wants and needs.

5.6.2. **Weaknesses**
The Prophylaxis chessboard will be heavy, require battery charging or access to a power outlet, and cost a lot compared to conventional chess sets. These limitations shift its appeal away from individual players and toward groups or organizations with greater resources and motivations.

5.6.3. **External Opportunities**
Chess is a historically significant game, and its popularity is relatively stable. The Prophylaxis team, therefore, does not need to worry as much about passing trends among their user base. Also, because it is based on chess, the product potentially appeals to a diverse audience around the world. Other automated chessboards may have some of the same features as the Prophylaxis; however, none has been found that have all of the features that the Prophylaxis plans to have.

5.6.4. **External Threats**
The Prophylaxis chessboard will include features that other automated chessboards have already adopted. Due to this, some of the features of the automated chessboard will have less of a “wow” factor, and may not operate as well as other more expensive, single feature automated chessboards for a given feature.

5.7. **Uniqueness & Existing/Potential Competitors**
A small of products featuring an automated chessboard have been on the market in the past or are currently on the market, yet research has revealed no product as fully functional as that which is proposed by Team 08.

One notable board is the Mirage, from Excalibur Electronics. The board was first produced in 1997, but is no longer on the market. The Mirage included the capability of playing versus an artificial intelligence, and used an electromagnet to move the pieces. Each square contained a pressure sensor so that the artificial intelligence would know the most recent move of the human player. One of the major weaknesses of the board, however, was the unreliability of some the moves, as other pieces would
occasionally be knocked out of position. The Mirage also was not capable of voice control, nor did it feature app support.

Another similar project, but one that is currently in development, is Chess Automated. Chess Automated also allow for playing against an artificial intelligence on a physical board, and also features an app that allows players to play via the internet from anywhere in the world. The Chess Automated team has currently finished four prototypes, and boasts a sleek design. Similar to the Mirage, Chess Automated uses pressure sensors for piece tracking but does not have the capability of using voice control (Chess Automated, n.d.).

The uniqueness of Team 08’s proposed product, then, is based on seamless integration of a wider variety of features than any other design, past or present.
6. Design Alternatives and Selection

6.1. Chess-piece Movement

6.1.1. General Requirements

Requirements for chess piece movement are detailed in Section 3.1, and are summarized as follows: The chess pieces shall have the capability of automatic movement. The pieces shall move without a visible movement mechanism. The movement should be smooth and quiet. The movement mechanism should occupy a small space beneath the chessboard, and not weigh too much. The board should be portable for playing in different rooms.

6.1.2. Alternatives

The team discussed the use of different movement mechanisms. The proposed mechanisms included linear actuators, stepper motors with ball screws, stepper motors with timing belts, purely magnetic movement, and a claw-like device to pick and place the pieces.

6.1.3. Selection

Stepper motors with a timing belt fit the needs of the project the best. The decision matrix (Table 4) shows some of the different qualities that went into making the decision. Ultimately, the stepper motors with ball screws proved to be a suboptimal solution because of cost and the fear of oscillation with too long of a screw shaft (Automation, RAF, 2013). Using linear actuators did not instill confidence with accuracy without expensive components, and it did not offer a compact solution. Purely magnetic movement also looked expensive, and required an overly complicated implementation for what the team was trying to accomplish.

\[ \text{Table 4 - Decision Matrix for the Selection of Piece Movement Mechanism} \]

<table>
<thead>
<tr>
<th>Option</th>
<th>Compact</th>
<th>Feasibility</th>
<th>Cost</th>
<th>Sound</th>
<th>Precision</th>
<th>Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>Linear Actuators</td>
<td>40</td>
<td>90</td>
<td>65</td>
<td>50</td>
<td>45</td>
<td>62</td>
</tr>
<tr>
<td>Ball Screws</td>
<td>80</td>
<td>65</td>
<td>60</td>
<td>90</td>
<td>100</td>
<td>77</td>
</tr>
<tr>
<td>Timing Belts</td>
<td>80</td>
<td>90</td>
<td>75</td>
<td>90</td>
<td>90</td>
<td>85</td>
</tr>
<tr>
<td>Purely Magnetic</td>
<td>100</td>
<td>30</td>
<td>50</td>
<td>100</td>
<td>100</td>
<td>70</td>
</tr>
</tbody>
</table>
6.2. Microcontroller

6.2.1. General Requirements

The microcontroller must be powerful enough to handle at least 64 reed switches and python software including the chess engine, piece tracking, and other functions needed in the software. The microcontroller must be user friendly and will allow the python code to be easily added into the microcontroller. The microcontroller also needed to have enough pins to handle all of the reed switches, motors, buttons, and anything else such as LEDs that could be added in the future.

6.2.2. Alternatives

Arduino Mega, Raspberry Pi 2.0 and other Raspberry Pi models.

6.2.3. Selection

A decision matrix used in the decision of the microcontroller may be seen in Table 5 below.

<table>
<thead>
<tr>
<th>Option</th>
<th># of Pins</th>
<th>Clock Speed</th>
<th>Display</th>
<th>User-Friendly</th>
<th>Pre-Knowledge</th>
<th>Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>Raspberry Pi A/B</td>
<td>60</td>
<td>75</td>
<td>100</td>
<td>90</td>
<td>95</td>
<td>81</td>
</tr>
<tr>
<td>Raspberry Pi A+/B+</td>
<td>80</td>
<td>85</td>
<td>100</td>
<td>92</td>
<td>95</td>
<td>89</td>
</tr>
<tr>
<td>Raspberry Pi 2.0</td>
<td>90</td>
<td>95</td>
<td>100</td>
<td>94</td>
<td>95</td>
<td>94</td>
</tr>
<tr>
<td>Arduino Mega</td>
<td>95</td>
<td>85</td>
<td>100</td>
<td>85</td>
<td>50</td>
<td>85</td>
</tr>
</tbody>
</table>

Raspberry Pi 2.0 outperformed the alternatives. Listed below are some of the Raspberry Pi 2.0 specifications.

- A 900MHz quad-core ARM Cortex-A7 CPU
- 1GB LPDDR2 SDRAM
- 4 USB ports
- 40 GPIO pins
- Full HDMI port
- Ethernet port
- Combined 3.5mm audio jack and composite video
- Camera interface (CSI)
- Display interface (DSI)
- Micro SD card slot
- VideoCore IV 3D graphics core
One of the main reasons the Raspberry Pi 2.0 was selected was due to its 40 GPIO pins. More pins are ideal in the choice of the microcontroller due to the great amount of components that will need to be attached to the Raspberry Pi 2.0 via the pins. The Raspberry Pi A and B have 17 pins and the Raspberry Pi A+ and B+ both have 26 pins. These amounts of pins may have not been enough to handle all of the reed switches and then the possibility of buttons and LEDs. The Raspberry Pi 2.0 also had better specifications, all around, than the other Raspberry Pi models. This also factored into the choice of the Raspberry Pi 2.0 because the amount of RAM, USB ports, and other factors were not fully known when the decision was made, and because of this it was safer to choose the Microcontroller with the better specifications.

The Arduino boards were also considered and the Arduino Mega (ATmega2560) did have enough I/O pins, having 54 pins; however, the other specifications were not as good as the Raspberry Pi 2.0. For instance the clock speed of the Arduino Mega is 16 MHz compared to 900 MHz (quad-core) with the Raspberry Pi 2.0. Other microcontrollers were also looked into; however, none matched the performance of the Raspberry Pi 2.0 while also having enough pins to support all of the reed switches and LEDs that would be needed.

In addition to the performance of the microcontroller, the usability and user-friendliness of the microcontroller was also of great importance in the selection of the microcontroller. The Raspberry Pi has been used by some of the team members of Team 08 and has proven to be very user-friendly. Also, the remaining members of Team 08 were interested in learning how to use a Raspberry Pi microcontroller as it is a common microcontroller for small project use.
6.3. **Board**

6.3.1. **General Requirements**

The board shall be thin enough for the reed switches to work efficiently and for the electro magnet to work efficiently as well. The board shall hold the pieces and the pieces shall be able to slide across the board easily.

6.3.2. **Alternatives**

The board may be made of wood, plastic, metal, or glass. There are various choices for the thickness of the board.

Various square sizes are also possible and square size is typically chosen based on the size of the pieces.

6.3.3. **Selection**

A wooden surface was chosen for its simple elegance and rigidity at widths around ¼” to 3/8”. Wooden pieces are also easier to embed magnets within than metal or glass pieces, and for aesthetic purposes the board and pieces were desired to be made of the same material. Wood is also considered a more elegant and classic look for a chessboard than plastic, which was the only other material relatively easy to embed magnets within.

The square sizes for the board were chosen to be 2 ¼”, in order to accommodate 1” diameter chess pieces. This would provide the pieces room to slide past each other, as in the case where a knight on the first move wants to move past the row of pawns. 2 ¼” is also a standard size for chess board squares.
6.4. Piece Tracking

6.4.1. General Requirements

Chess pieces are easily tracked and their positions are stored in a database for use throughout the game by the software.

6.4.2. Alternatives

Choices include reed switches, hall effect sensors, RFID chips, and pressure sensors, as well as an above-the-board camera to monitor the board position.

6.4.3. Selection

Reed switches were chosen to implement piece tracking because of their relatively low cost and for being easier to build into the board design than the alternatives, as shown in the decision matrix below in Table 6.

<table>
<thead>
<tr>
<th>Option</th>
<th>Cost</th>
<th>Layout</th>
<th>Size</th>
<th>Precision</th>
<th>Pre-Knowledge</th>
<th>Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reed Switches</td>
<td>90</td>
<td>80</td>
<td>95</td>
<td>90</td>
<td>70</td>
<td>87</td>
</tr>
<tr>
<td>Hall Effect Sensors</td>
<td>85</td>
<td>70</td>
<td>75</td>
<td>95</td>
<td>70</td>
<td>80</td>
</tr>
<tr>
<td>RFID chips</td>
<td>70</td>
<td>80</td>
<td>90</td>
<td>95</td>
<td>70</td>
<td>82</td>
</tr>
<tr>
<td>Pressure Sensors</td>
<td>90</td>
<td>80</td>
<td>95</td>
<td>75</td>
<td>70</td>
<td>84</td>
</tr>
<tr>
<td>Camera</td>
<td>50</td>
<td>70</td>
<td>85</td>
<td>100</td>
<td>70</td>
<td>75</td>
</tr>
</tbody>
</table>

The main reason for not choosing RFID-based piece tracking was because of the relatively high cost of RFID tags, as one tag per piece would be required, and a method of sensing that could cover every square would have to be determined, which would likely have involved using an RFID reader below every square—a very expensive sensing method.

Pressure sensors would also be expensive, as each square would need a pressure sensor to tell if a piece was on it. In addition, the pressure sensor would have to be built into the board surface and would likely increase the width of the board beyond the practical range of the electromagnet to reliably attract and move pieces.

Utilizing an above-board camera would have an advantage in that the different types of pieces could also be distinguished, rather than simply telling if a square is occupied or not. The disadvantages of the camera are the need for complex image processing and for an addition to the board for mounting the camera above the playing surface. This would erode the aesthetic quality of the board, as without the camera the
chess pieces themselves would be the tallest items in the design and the board would look and feel more like a standard chessboard.

Hall effect sensors were promising in that they could work with the already-existing magnets embedded in the chess pieces, but they would also be more expensive than reed switches or pressure sensors. Reading an array of 64 to 100 hall effect sensors would be difficult due to their analog measurements, compared to the digital readings (open or closed) from reed switches.

The reed switches, then, were the best method that was not overly expensive nor detracted from the functional and aesthetical qualities of the board. Since the reed switches are small, shaped similar to resistors, and relatively inexpensive, an array of reed switches with one per square can be implemented. In order for a large array of reed switches to align with the squares of the chess board without the use of excessive connection points, it is common for reed switches are arranged in rows and columns. In this configuration, each reed switch must be connected to a diode in order to avoid false positives. Although these diodes add to the cost of the reed switch array option, this option still has a reasonable cost while maintaining a thin surface width of the board for the electromagnet to work effectively.

Reed switches also have the advantage that the pieces already need to have magnets embedded within them so that they can be moved by the electromagnet. The reed switches, then, use the already-existing magnets in the pieces for the purpose of sensing piece locations: the magnetic fields from the magnets in the pieces cause reed switches directly below the pieces to close so that current can flow through. When no piece is present above the square, then, the reed switch remains open and no current can flow through it. This allows for testing by checking for open or closed circuits as they result from the state of the reed switches.
7. Integration and Testing

7.1. Reed Switches

7.1.1. Single Reed Switch Testing

Initial testing of the reed switches was accomplished by connecting the reed switch to be tested to an ohmmeter to be able to tell when the reed switch was open or closed. All reed switches acquired were classified as "nominally open", so they only close and allow current to pass when in the presence of a strong enough magnetic field.

The first test was to figure out the maximum range of the reed switches by placing each reed switch as if it was embedded below the board and testing how far away the magnet could be placed in order for the reed switch to close. Three different magnets and five different reed switches were tested.

The second test was to determine the range of sensitivity for a reed switch at a set vertical distance away from a magnet (a distance of 1/4"), as if the reed switch was embedded horizontally below the board and the magnet was on top of the board moving horizontally. The range of sensitivity for the reed switches to close was determined to be roughly circular, with two circles of sensitivity each offset from different ends of the reed switch. All combinations of the three magnets and five reed switches were tested.

The results of these tests showed that reed switches that could close given smaller magnetic fields were optimal, and that large magnets were better than small magnets. Both of these result in the largest possible sensitivity radius for the reed switch, which was about 1 inch.

Tests were additionally performed with and without a 3/16” wooden board between the magnet and reed switches, and these tests demonstrated that the wooden board had negligible effect on the magnet field and sensitivity range of the reed switches. The most promising magnet and reed switch combination resulted in approximately a one inch sensing radius at each end of the reed switch when sensing through a 3/16” board. This used the largest magnet (1/2” diameter) with the most sensitive reed switches.

7.1.2. Reed Switch Array Testing

In order to effectively analyze an array of reed switches with a minimal number of I/O pins, the testing strategy developed was to test one row of the board at a time, giving it a HIGH input and rest LOW. Then, the columns of the array can be read, and columns that are HIGH contain a piece in that row and columns that are LOW do not. For each reed switch, a diode must be placed in series in order to not allow current to flow in the wrong direction during the testing process.
Since only one row is tested at a time, a decoder can be used to shrink the number of GPIO pins needed on the Raspberry Pi. A 12-by-8 array is needed for the final design, due to the necessity of having a 2-by-8 bank of squares on each side of the chessboard for captured pieces. Therefore, a 4-to-16 decoder could be used to control the 12 rows of the board, and the total number of GPIO pins needed would only be 12 (4 for the rows, 8 for the columns).

![Figure 6 - Reed Switch Array Test Circuit Diagram](image)

A small 3-by-2 array of reed switches was created for testing purposes, and a 2-to-4 decoder was used to control the 3 rows. This circuit diagram is shown above. The array was successfully tested through running Python code on the Raspberry Pi, and code was able to locate the closed reed switches and create a matrix of the state of the board, filled with 0s where there is no piece and 1s where pieces reside.

### 7.2. Electromagnet

For initial testing of the electromagnet, two 3/16" boards were placed together to create a 3/8" board, and magnets were placed on top of the board and the electromagnet was moved below the board. The 3/8" separation is larger than the width of the actual board in order to account for the space needed to embed the reed switches below the board.

Tests were performed by adding weight above the magnets to simulate the weight of chess pieces and to approximate the actual amount of friction for wooded pieces with felt. The electromagnet was found to attract the biggest magnet from 1/2" and move it reliably with the additional weight. The smaller magnets that would be embedded in the pieces, however, were not reliable for moving the additional weight.
Two more electromagnets were acquired that were more powerful than the original electromagnet. Testing revealed that both worked well with the 1/2” magnet, although further testing utilizing an actual polished surface and wooden pieces with felt will be needed to make the final decision on which electromagnet to use. In addition, the weight of the chosen electromagnet will be a factor, as the weight may affect the XY-Table performance. Therefore, the electromagnets will also need to be tested on a working XY-Table prototype.

7.3. **XY-Table Motors**

The stepper motors turn in 1/2000 rotations increments. These increments are called steps. Steps are controlled by sending a pulse to the “clock+” input of our stepper motor drivers. Sending a precise number of steps ensures the accuracy of the stepper motors’ location. The timing of the steps is also critical. The speed of the steps must ramp up as the motor starts, and slow down as the motor stops. This ensures that the motors do not lose their position due to inertia.

Team 08 encountered a problem with the motors early on, as the motors generated a lot of heat. The team understood this is from the fact that the motors always run at full current, even when off, but the temperature was unsafe. The motors reached temperatures too hot to touch. Using a potentiometer, the team adjusted the current through the motors 4A down to 1.5A. The motors still functioned well, and were safe to touch. The torque will have to be monitored in the final implementation to maintain an optimum balance of heat generation and sufficient power.

7.3.1 **Sequential Motor Movement**

7.3.1.1 **For Loops**

Team 08 initially decided to use Python loops to send the steps to the motors. Figure 7 shows an example of the architecture implemented first (a rough sketch – it would not actually work).

```py
for (steps = 0; steps < stepsToTurn; steps++):
    stepPin = HIGH
    wait(1)
    stepPin = LOW
    wait(1)
```

*Figure 7- Loop-based step control*

This solution failed because it did not have good timing performance. The time between each pulse was variable. The team hypothesized background processes running in Linux consumed processor time, and caused variability in the timing. Explicitly observing the timing proved, but the gaps in timing could be heard by loud chunking sounds and vibrations in the motors.
### 7.3.1.2 PWM with Interrupts

Since the timing was not consistent with *for loops* and wait statements, the team decided to use the software PWM function provided with the GPIO library of the Raspberry Pi. Calling the PWM for the same amount of time did not allow for precise numbers of rotation, so the team counted the number of steps issued from the PWM with an interrupt. This architecture, shown in Figure 8, proved the best results. The rotation never had errors larger than one degree, and it enabled smooth motion.

```c
steps = 0
detectInterrupt(stepPin)  # set an interrupt on the step pin
PWM(stepPin)              # turn on the PWM signal

handleInterrupt:
    steps++                # handle interrupts on the step pin
    if (steps == stepsToTurn) # turn off the pwm after desired number of steps
        stopPWM(stepPin)
```

*Figure 8 - PWM with Interrupts Code Architecture*

### 7.3.2 Parallelized Motor Movement

A big part of the motor control is being able to move both the X motor and the Y motor at the same time to allow for diagonal movements for pieces such as the queen and bishops. Both multiprocessing and multithreading were researched and tested in order to see which would be a better fit for implementing the parallelization of the motors. Through various tests, it was found that neither multithreading or multiprocessing on the Raspberry Pi would work well because when the processes would run at the same time the motors would experience a great deal of chunking and the motors would run at different speeds, even with the PWM with interrupts architecture. This again appeared to be because there were background processes running at the same time. Several tests confirmed that one motor consistently moved slower than the other did. The speed of the motors was also variable from test to test. Finally, the motors would not move one revolution accurately. Table 7 shows the poor results of these tests.
Since parallelization was a failure, Team 08 decided to order Arduinos for motor controllers. The Arduinos run sequentially with one core (with no background tasks!). Since we want a parallel execution, the Raspberry Pi will call programs on each of the Arduinos simultaneously creating a parallelized process. Each Arduino will control one motor; one Arduino will control the x motor and the other will control the y motor. The Raspberry Pi and the Arduino can communicate via a USB-Serial connection.

### 7.4. Software

In order to test the software effectively, a test file was created. The test file tests a get move method, which is used to obtain a move from a matrix of 1’s and 0’s and an updated matrix of 1’s and 0’s. The updated matrix is obtained from the reed switches that are connected to the GPIO ports, and the matrix for comparison is obtained from the chess engine’s board. The chess engine’s board is converted into a matrix of 1’s and 0’s for comparison while the reed switch data is also made into a matrix of 1’s and 0’s.

The test file was found to be very effective in testing the get move method. Various board layouts were created for testing and different situations were tested. Special moves have yet to be implemented in the testing class; however, special move testing will be implemented in the future. These special moves are moves such as castling or pawn promotions.
8. Conclusion

8.1. Potential Risks & Issues

Throughout much of the project, many risks and issues will arise, and to better prepare for these risks and issues Team 08 has taken the time to look into and decipher the potential risks and issues that may arise. One of the biggest issues that may arise is the lack of functionality with the reed switches. The reed switches are a key part of this project, causing this potential risk to be a significant risk. The reed switches need to have a strong enough magnetic field so that they may pick up a signal from the piece that is in the square, in which the reed switch is designated to, but at the same time, the reed switch cannot be too strong as to pick up a signal from a piece in an adjacent square. Another potential issue concerns the electromagnet’s movement of the chess pieces. If the electromagnet is too strong it may drag other pieces with it while it is moving one piece, and if it is too weak then the electromagnet will not be able to move the chess pieces effectively. Another potential risk is with the motors used in the xy-table. The motors will need to be set up in a way that they do not try to start up instantly and stop instantly, but instead have a more gradual start up and stop. Potential problems may arise in figuring out the algorithm that will be needed for this to happen. In addition to these potential risks and issues, one of the most important issues that may need to be dealt with is issues regarding the integration of all of the parts into one final system. The hardware and software will need to be integrated together and there may be a great deal of issues that arise from this integration that will need to be dealt with.

8.2. Project Status

Team 08 plans to have a fully functioning, automatic moving chessboard with artificial intelligence, and the ability to move via voice recognition, an app using a web server, manually, and on its own using artificial intelligence, done by May. As of now, Prophylaxis is moving along at a good rate and should be complete by May. The base of the software, including the chess engine and code to interact with the chess engine and hardware is mostly complete. The parts for the xy-table are being obtained and research has been done on the step-motors that will be used for the xy-table. The layout of the reed switches and diodes has been chosen and testing on the reed switches and electromagnet is being done. The actual board development has yet to be started; however, different designs have been researched. App development is being postponed until later in the project, while the web server creation has just begun. Overall, the project is moving at a good rate and the chessboard should be complete by May.
8.3.  Project Feasibility
As stated before, Team 08 plans to have a fully functioning, automatic moving chessboard with artificial intelligence, and the ability to move via voice recognition, an app using a web server, manually, and on its own using artificial intelligence, done by May. This plan appears to be quite feasible, as Team 08 has been moving along at a steady rate, which may be seen in the status section of this report. Potential risks and issues may appear later in the process of created the automated chessboard; however, solutions for much of the potential risks and issues have been created in case the risks and issues do arise. Team 08 has also left a great deal of time open for “debugging.” Team 08 is aware that problems will arise and because of this Team 08 has planned to start the integration stage a significant amount of time before May. This will allow for any issues that may arise to be solved, as well as allowing for any extra add-ons that may be wanted.

8.4.  Summary
In conclusion, Team 08 is aware of many potential risks and issues that may arise and has done well in planning for these issues. Team 08 will combine each of its member’s skill sets to create a chess board that is pleasing to its user, will function properly, and will show the underlining design norms that were used in the creation of the automated chessboard. The chessboard will be complete by the month of May and will meet all of the requirements Team 08 has previously set for the chessboard. Regarding a more important requirement, Team 08 will follow the biblical perspective and calling, and strive to bring honor and glory to God through the research, design steps, and final design of the Prophylaxis automated chessboard.
9. Acknowledgements

Team 08 acknowledges the support and guidance of Mark Michmerhuizen, the team’s advisor, and Eric Walstra, the team’s industrial consultant.

Jim Kortman was also helpful to the team through his providing of equipment and materials for building an XY-Table. The team thanks Theo Voss for his input and encouragement. Additional thanks goes to Bob DeKraker for handling parts orders and delivering the ordered components to the team.
10. References


10.1. Data Sheets


http://www.fastech.co.kr/bbs/eng/product.php?mode=view2&uid=1 (56M-A)

http://www.capterra.com/project-management-software/

11. Appendices

11.1. Appendix 1: Gantt Chart

This is the proposed schedule for Team 08.
11.2. Appendix 2: Hourly Work Breakdown

Table 8- Hourly Break down of Projects

<table>
<thead>
<tr>
<th>Member</th>
<th>Documents</th>
<th>Software</th>
<th>XY Table</th>
<th>Reed Board</th>
<th>Hardware</th>
<th>Research</th>
<th>Team Meetings</th>
<th>Totals</th>
</tr>
</thead>
<tbody>
<tr>
<td>Derek</td>
<td>34.5</td>
<td>95.5</td>
<td>12</td>
<td>0</td>
<td>0</td>
<td>10</td>
<td>9.5</td>
<td>161.5</td>
</tr>
<tr>
<td>Nick</td>
<td>50.5</td>
<td>64</td>
<td>8</td>
<td>3</td>
<td>0</td>
<td>16</td>
<td>6.5</td>
<td>148</td>
</tr>
<tr>
<td>Curtis</td>
<td>18</td>
<td>14.5</td>
<td>20.5</td>
<td>0</td>
<td>6</td>
<td>6</td>
<td>14</td>
<td>79</td>
</tr>
<tr>
<td>Paul</td>
<td>23</td>
<td>3</td>
<td>0</td>
<td>28.5</td>
<td>7.5</td>
<td>8</td>
<td>10</td>
<td>80</td>
</tr>
<tr>
<td>Alexis</td>
<td>17.5</td>
<td>1.5</td>
<td>0</td>
<td>8.5</td>
<td>1.5</td>
<td>3</td>
<td>6.5</td>
<td>38.5</td>
</tr>
<tr>
<td>Totals</td>
<td>143.5</td>
<td>178.5</td>
<td>40.5</td>
<td>40</td>
<td>15</td>
<td>43</td>
<td>46.5</td>
<td>507</td>
</tr>
</tbody>
</table>

Figure 9- Project breakdown showing hours worked by team members
Figure 10 - Percentage of hours spent on different project parts

Figure 11 - Running sum of the work completed separated by part of project