Team 10

MagiChess

Jack Deguglielmo, Samantha Klein, Weishan Li, Sai Thuta Kyaw

Advisor: Shira Epstein
Meet the team

Sai Thuta Kyaw
Electrical Engineer

Samantha Klein
Electrical Engineer

Jack Deguglielmo
Computer Engineer

Weishan Li
Computer Engineer

Shira Epstein
Faculty Team Advisor
Problem Statement

For centuries, the game of chess has been played by two players sitting across a chessboard. The advent of digital technology in the last decades has brought virtual chess to computers and mobile phones and for the first time, this has allowed players to be anywhere across the world.

Digital chess lacks:
- A physical aspect/satisfaction of seeing and moving your own pieces

Physical chess lacks:
- Ability to play from anywhere and with anyone
Our Solution

We’ve decided to close the gap between physical and digital chess. To do this, we plan to create a chess board that allows users to play with an AI or a remote human opponent.

Plan:

● Sense location of chess pieces on the board
● Interface with LiChess server
● Automate piece moving
Preliminary System Specifications (Design-agnostic)

- Mechanically move a piece to destination cell
- Remove/replace a piece to/from game board
- Provide visual feedback
  - Game setup, tutorial
  - Game announcements
  - Highlights previous move
- Provide audio feedback
  - Notification alerts
- Play versus remote opponent
- Playback previous games
- Includes buffer zone to store captured pieces
- Topple the King after checkmate
Preliminary System Specifications (Quantitative)

- Total system dimensions: no larger than 32.5 in x 30 in x 8in (80 cm x 74 cm x 15 cm)
- Speed of XY plotter: 5 - 8 cm/s
  - Speed increased due to better stepper drivers
  - Absolute maximum time taken for a move 25s
  - Move each pieces under 10s more than half of the time
- Weight: Under 50lbs
  - Upgrading from wood to more robust aluminium frame
Vision statement for our working prototype:
Our vision for the working prototype and progress for Magichess includes several key functionalities. We will integrate subsystems described in MDR (LiChess API conversation with physical movement of gantry). We will have a complete assembly and wood frame of our board as well as communication between Pi and (at least) two 328p working as intended.
Proposed CDR Deliverables

Key aspects of our prototype:

- System able to detect Chess piece movement made by the user.
- System able to communicate with LiChess the movement made by the user.
- System able to move chess pieces around with Electromagnet and Gantry System with a reasonable success rate.
- Fully functional graphical user interface
- Completed frame and mechanical assembly of the chessboard and gantry
Demos for Integrated System

Raspberry Pi and Gantry Making Moves
Making Physical Moves

Raspberry Pi and Fast Scanning Hall Sensors Detecting movement.

Raspberry Pi A
Gantry MCU PCB
XY Gantry

GitHub Repo

Raspberry Pi B
Sensor MCU Board
Long Sensor PCB

Geographic Isolation
MDR Demos

MDR:

- **Fast Scanning**
  - Multiplexed 4 Hall sensors

- **GUI**
  - Game playing limited to application
  - No audio, text feedback

- **Movement**
  - Simple X and Y axis movement
  - Mini Testbench to test different materials
  - Non-optimized Path Planning
Current Prototype

Left: Soldiered Hall Sensors on Sensor PCB

Right: Sensor PCBs setup and wiring. Breadboard Power Rails are used for wire connections only.
Pre-MDR: Fast Scanning

- Fast Scanning
  - Multiplexed 4 Hall sensors
  - Used one 4 2x1 mux
  - No communication with Pi
Fast Scanning Demo
Pre-MDR: GUI

Lacking
- End game handling
- Audio
- User interaction
- Text feedback on game updates
GUI Updates

Integrated Audio Feedback!
Pre-MDR: Movement

- Barebone, noisy and shaky gantry
- Mini Testbench with manual movements
Post-MDR Development: Gantry

- Pencil Marker
- Contact Switches
Current Prototype

Left: MagiChess frame, electromagnet and monitors for testing

Right: MagiChess Frame and Gantry
Gantry and GUI Demo
CDR Accomplishments

- Fast Scanning
  - Multiplexed 64 Hall sensors
  - Communicate with Pi
  - Able to detect move

- Movement
  - Smooth and quieter movements
  - Self-Calibrate and communication with Pi

- GUI
  - Integrated with different subsystems
  - Added Audio/Text feedback
CDR Deliverables

- System able to communicate with LiChess the movement made by the user.
- System able to detect chess piece movement made by the user
- System able to move chess pieces around with Electromagnet and Gantry System with a reasonable success rate.

- Fully functional graphical user interface
  - Audio Integrated
  - Optimized for touch display and added features

- Completed frame and mechanical assembly of the chessboard and gantry
  - Wooden frame as seen in the current prototype
  - Upgrading from wood to Aluminium
Software Diagram - Game State

(A) gameStart, challengeDeclined, ...

(B) gameState, chatLine, ...
1. Gantry Control PCB

1. Microcontroller
2. Reset Button
3. Level Shifters
4. ICSP Port
5. Power Ports
6. Status LEDs + Resistors
7. Extra Ports
8. 12V Power Input
2. Sensor Board PCB
3. Sensor Control PCB

1. Microcontroller
2. Power Input
3. Reset Button
4. Level Shifter
FPR Plan

- Migrate from wood to aluminium extrusion frame with plywood + plexiglass sides - April 10th
- 3D-Print Chess pieces with velvet bottoms and embedded magnet
- Migrate from header pins to JST connectors
- Order new PCB to minimize wiring sensor boards
- **Optimize software**
  - error handling
Plans for Testing Prototype

- Stress test gantry with simulated game play for 4 hours
- Stress test hall sensors with real game play
- Play socially-distanced chess with strangers
- Test and record failures to perform root cause analysis

Testing timing
Pi saves the “distance” for average and max move for a typical game

Use mathematical modeling to calculate the timings

(See additional slides)
Plans for Hardening Prototype

- Retrying failed moves certain number of time
- Add option for User intervention to correct the physical gamestate
- Occasionally resetting the gantry
- Monitor thermals
Plans for FPR Demo

Play a game over the internet

We challenge YOU to a game of chess!
## Responsibilities post CDR

<table>
<thead>
<tr>
<th>Jack</th>
<th>Sam</th>
<th>Sai</th>
</tr>
</thead>
<tbody>
<tr>
<td>● Raspberry Pi interfaces with 328Ps</td>
<td></td>
<td></td>
</tr>
<tr>
<td>● Analysis of gantry move time</td>
<td>● Evaluate the use of other protocols over software UART</td>
<td></td>
</tr>
<tr>
<td>● Replay/resume/reset game</td>
<td>● Refine Fast Scanning</td>
<td></td>
</tr>
<tr>
<td>● Altium Lead</td>
<td>● Budget Manager</td>
<td></td>
</tr>
<tr>
<td>Weishan</td>
<td></td>
<td></td>
</tr>
<tr>
<td>● Refine and add features to GUI</td>
<td></td>
<td>● Team Coordinator</td>
</tr>
<tr>
<td>● Improve and debug communication between Pi and 328p’s</td>
<td></td>
<td>● Final Frame Assembly</td>
</tr>
<tr>
<td>● Replayable Games</td>
<td></td>
<td>● Testing and Hardening Movement</td>
</tr>
</tbody>
</table>
# Total Spending

<table>
<thead>
<tr>
<th>Description</th>
<th>Amount</th>
</tr>
</thead>
<tbody>
<tr>
<td>Total Spending (Fall Semester)</td>
<td>264.83</td>
</tr>
<tr>
<td>Stepper Motor Drivers</td>
<td>40.1</td>
</tr>
<tr>
<td>Prototype (DigiKey)</td>
<td>29.49</td>
</tr>
<tr>
<td>JLC PCB</td>
<td>49.36</td>
</tr>
<tr>
<td>PCB Population (DigiKey)</td>
<td>57.19</td>
</tr>
<tr>
<td>PCB Population (Newark)</td>
<td>41.84</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td><strong>482.81</strong></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Description</th>
<th>Amount</th>
</tr>
</thead>
<tbody>
<tr>
<td>Budget Extension</td>
<td>150</td>
</tr>
<tr>
<td>Misc. Order for Assembly</td>
<td>86.59</td>
</tr>
<tr>
<td>JLC PCB (Revision + Power)</td>
<td>25.6</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td><strong>112.19</strong></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Description</th>
<th>Amount</th>
</tr>
</thead>
<tbody>
<tr>
<td>Remaining Budget</td>
<td>55</td>
</tr>
</tbody>
</table>
## Gantt Chart After CDR

<table>
<thead>
<tr>
<th>Task</th>
<th>Team Member</th>
<th>Mar 28 - Apr 3</th>
<th>Apr 4 - Apr 10</th>
<th>Apr 11 - Apr 17</th>
<th>Apr 18 - Apr 24</th>
<th>Apr 25 - May 1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bug Fix</td>
<td>Jack</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Training/Replay</td>
<td>Jack</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Bug Fix</td>
<td>Wei</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Training/Replay</td>
<td>Wei</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>System Integration</td>
<td>Sam</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Bug Fix</td>
<td>Sam</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Final Frame Assembly</td>
<td>Sai</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>System Integration</td>
<td>Sai</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
External Links

Team Website

All Demo Videos Playlist

Github Repo
Thank You