Review the assignment due date

final-project-skeleton

Team Number: 24

Team Name: TEAM |= (1<<TEAM_NAME)

Team Member Name	Email Address
Bowen Wang	wangbw@seas.upenn.edu
Alexander Yu	ayu2126@seas.upenn.edu
Benjamin Abt	bencabt@seas.upenn.edu

GitHub Repository URL: https://github.com/upenn-embedded/final-project-f25-f25-final_project-t24.git

GitHub Pages Website URL: [for final submission]*

Final Project Proposal

1. Abstract

We aim to build a pair of chess boards with chess pieces that can communicate moves made by one human player to the other board and have the other board move the piece. The boards will achieve this by using hall effect sensor arrays under the board surface to detect moves made human players, send the move to the paired board, and that board will use a gantry rail system to actualize the move. This way, two human players play physical chess against each other in potentially two separate rooms.

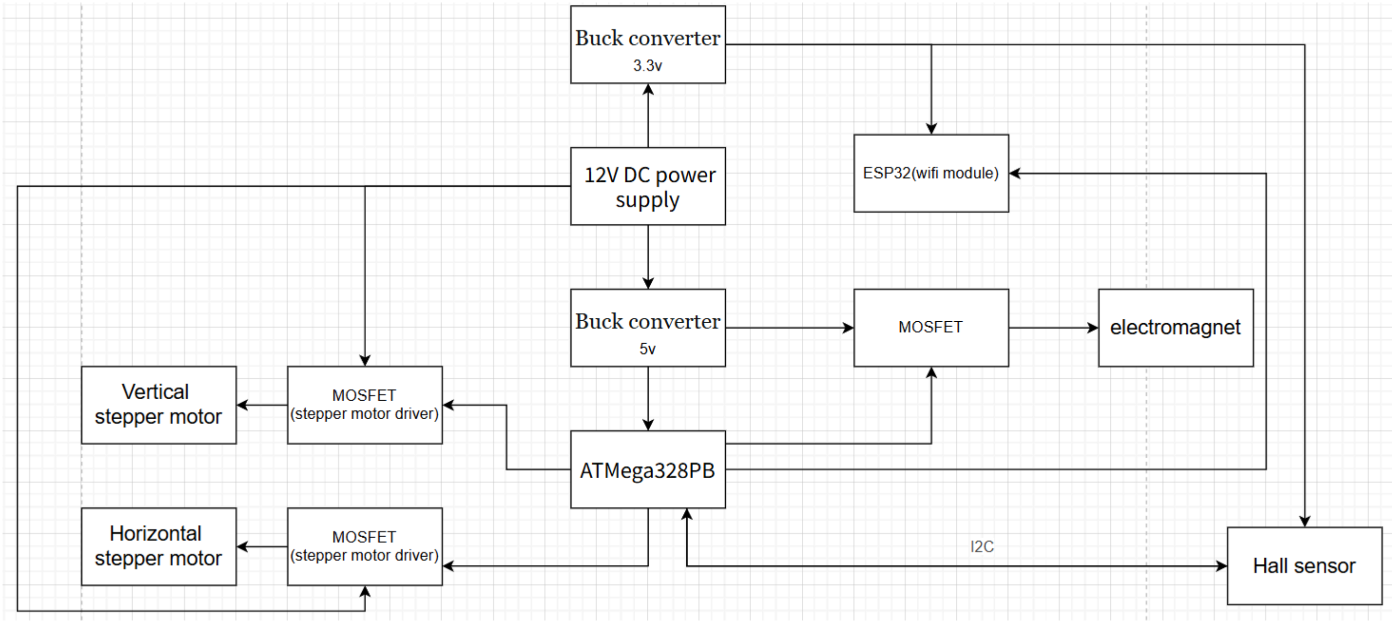
An additional feature would be that the player could change a setting and play against a computer on their own board.

2. Motivation

If you like to play chess but not on chess dot com, then this is for you. We aim to adapt and build a known game to be played by humans separated by distance on physical boards with real game pieces instead of a mouse or finger and screen.

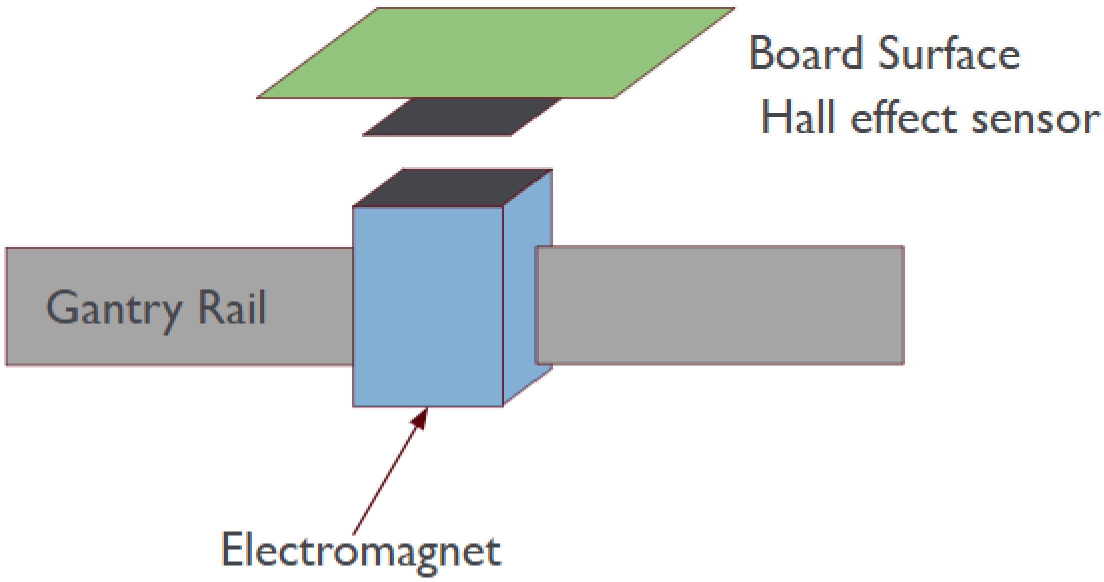
3. System Block Diagram

Show your high level design, as done in WS1 and WS2. What are the critical components in your system? How do they communicate (I2C?, interrupts, ADC, *etc.)? What power regulation do you need?*

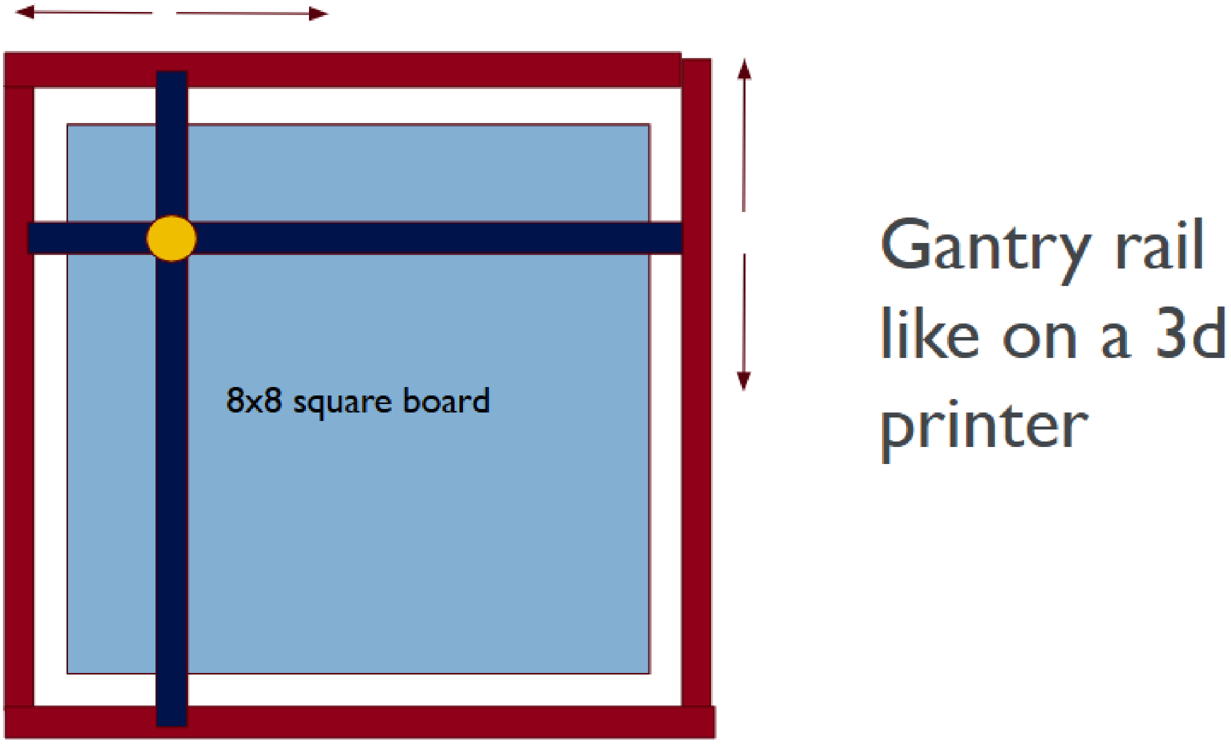


4. Design Sketches

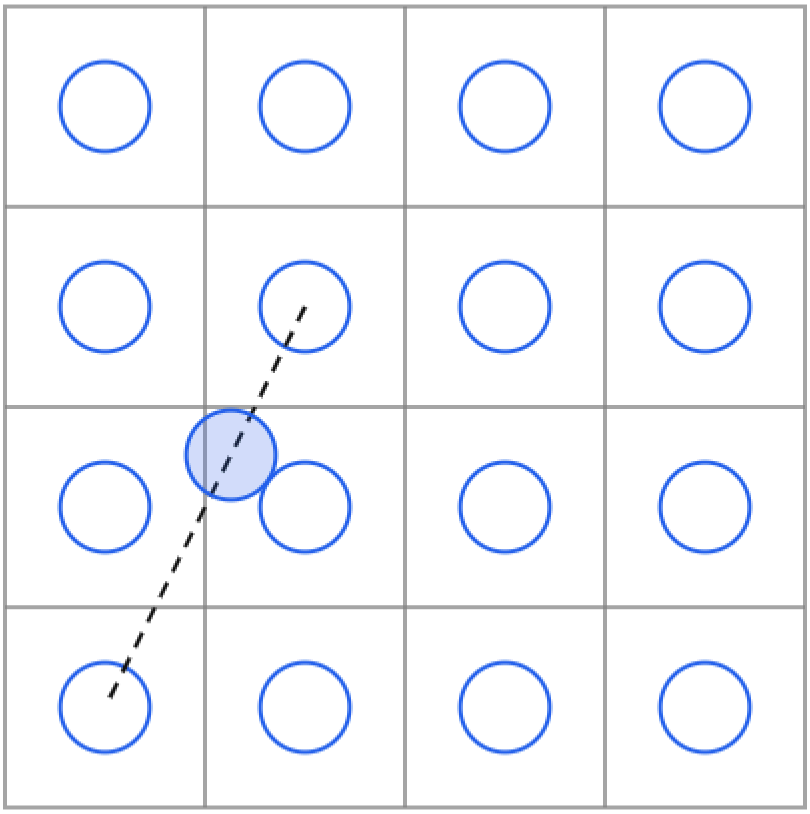
What will your project look like? Do you have any critical design features? Will you need any special manufacturing techniques to achieve your vision, like power tools, laser cutting, or 3D printing? Submit drawings for this section.



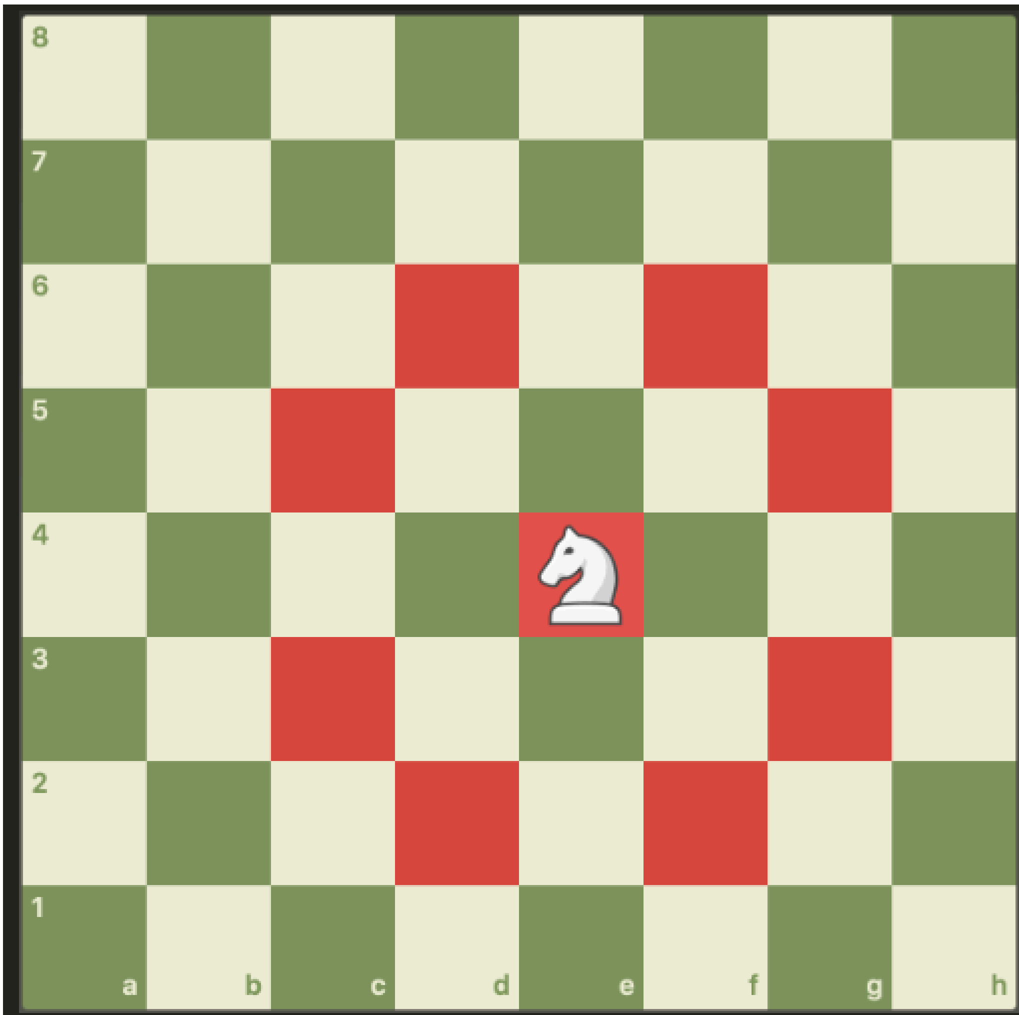
Single virtical cell of the 64



Gantry rail visualization



Maximum radius of piece relative to width of cell is R=0.22



The knight moves in an "L" shape.

5. Software Requirements Specification (SRS)

Formulate key software requirements here. Think deeply on the design: What must your device do? How will you measure this during validation testing? Create 4 to 8 critical system requirements.

These must be testable! See the Final Project Manual Appendix for details. Refer to the table below; replace these examples with your own.

5.1 Definitions, Abbreviations

Here, you will define any special terms, acronyms, or abbreviations you plan to use for hardware

5.2 Functionality

ID	Description
SRS-01	The time between a move and the end of software computation will be within 2s.
SRS-02	After playing multiple games, there should be zero instances where the incorrect piece is moved, the correct piece is moved to the wrong position, or no movement is captured.
SRS-03	Upon a user pressing an e-stop, the system should stop moving within .5s
SRS-04	Signal disconnections should last for no more than 10s.

6. Hardware Requirements Specification (HRS)

Formulate key hardware requirements here. Think deeply on the design: What must your device do? How will you measure this during validation testing? Create 4 to 8 critical system requirements.

These must be testable! See the Final Project Manual Appendix for details. Refer to the table below; replace these examples with your own.

6.1 Definitions, Abbreviations

Here, you will define any special terms, acronyms, or abbreviations you plan to use for hardware

6.2 Functionality

ID	Description
HRS-01	The electromagnet must be operable through the chess board (1cm).
HRS-02	Locational accuracy of the pieces must be within a 1mm radius.
HRS-03	No piece motion should take more than 10s.

ID	Description
HRS-04	No piece should collide with another piece when moving (test over 3 games).
HRS-05	No piece should fall over at any instance (test over 3 games).

7. Bill of Materials (BOM)

In order to detect the pieces, Hall effect sensor is used to detect the magnet under the pieces. Because the selected Hall effect sensor uses I2C to communicate with MCU, the multiple sensor can be connected in series.

Electromagnet is used to move the pieces. Because there is a magnet under the pieces, the pieces can be picked up when the electromagnet is opened and put down when the electromagnet is closed.

Stepper motor is used to move the gantry rail, The stepper motor has a 1.8 degree step angle, and this motor can move the pieces accurately.

The Wifi module can help the chessboard to communicate with computer.

Cost may become an issue. If it does we can switch from a gantry set up to a electromagnet array and also scale down the scale from 2 8x8 boards to 2 4x4 or 3x3 game boards, just enough to demonstrate full piece movements.

https://docs.google.com/spreadsheets/d/1hQGFuSGjS97sqDZma7hITm9WjsHUTzoeXTmpe9kDE0k/edit?gid=2071228825#gid=2071228825

8. Final Demo Goals

We will be able to run a game of chess between two users on two seperate boards._

9. Sprint Planning

You've got limited time to get this project done! How will you plan your sprint milestones? How will you distribute the work within your team? Review the schedule in the final project manual for exact dates.

Milestone	Functionality Achieved	Distribution of Work
Sprint #1	Wired Component communication Wireless communication	
Sprint #2	Hall effect array piece detection Movment system and driver built	
MVP Demo	Piece movment code Remote play code	
Final Demo	Single player game code	

This is the end of the Project Proposal section. The remaining sections will be filled out based on the milestone schedule.

Sprint Review #1

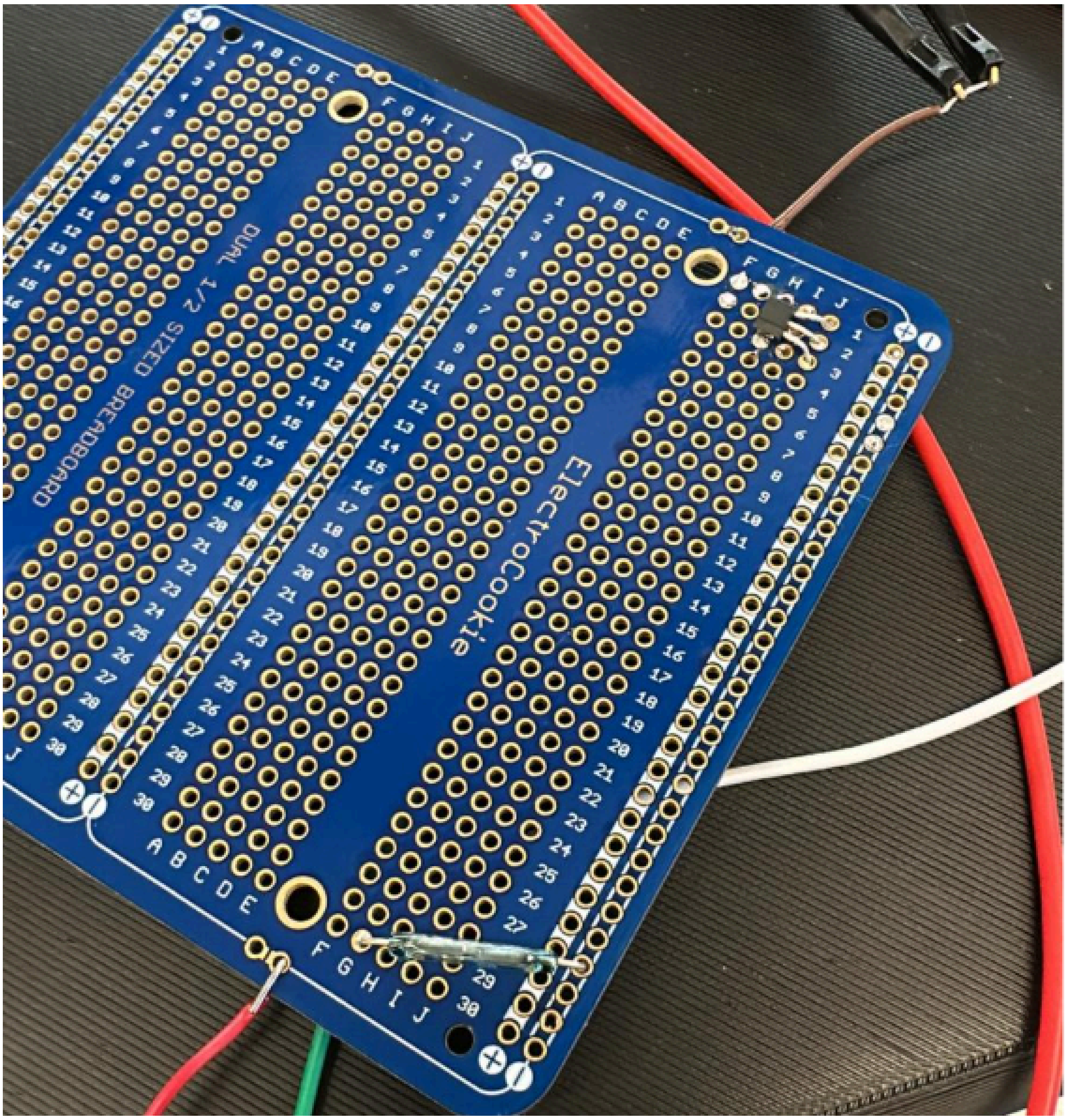
Last week's progress

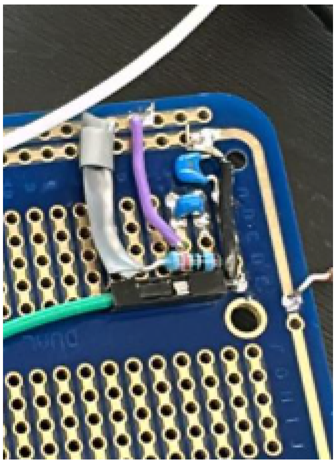
We soldered two trial boards of a single game board cell. One for Hall effect sensor utilization, and one for reed switch utilization. We evaluated that using reed switches would greatly reduce the number of work hours required and updated our BOM to reflect that. We considered some software solutions and talked about how we can do memory management of game datastructures.

One proposal is to use a uint8_t arrays of size 32. Each uint8_t will be encoded to contain a 3bit x cordinate, 3bit y coordinate, 1 bit moving/not-moving, 1 bit for alive/captured.

We can use the array index to track of each piece i.e. index <0:7> are white pawns, index<8:9> are white rooks, etc.

Thus in our 2kb of memory, we only allocate 32 bytes for keeping track of the entire game state.





Current state of project

We have purchased our parts, but our parts have not arrived yet.

Since our parts have not arrived yet, we tested some parts from detkin, like hall effect sensor and reed switches.

We have a idea for tracking the game state, and we can test them after we have built our chessboard.

Next week's plan

Flesh out software skeleton code

Build an mini sensor array with detkin parts to test ATMega communication. (1 - 2 weeks) Will finish together.

Definition of "done": Build the communication with ATMega.

Write some code for stepper motor and driver and tast it after we receive our parts. (1 week) Will finish together.

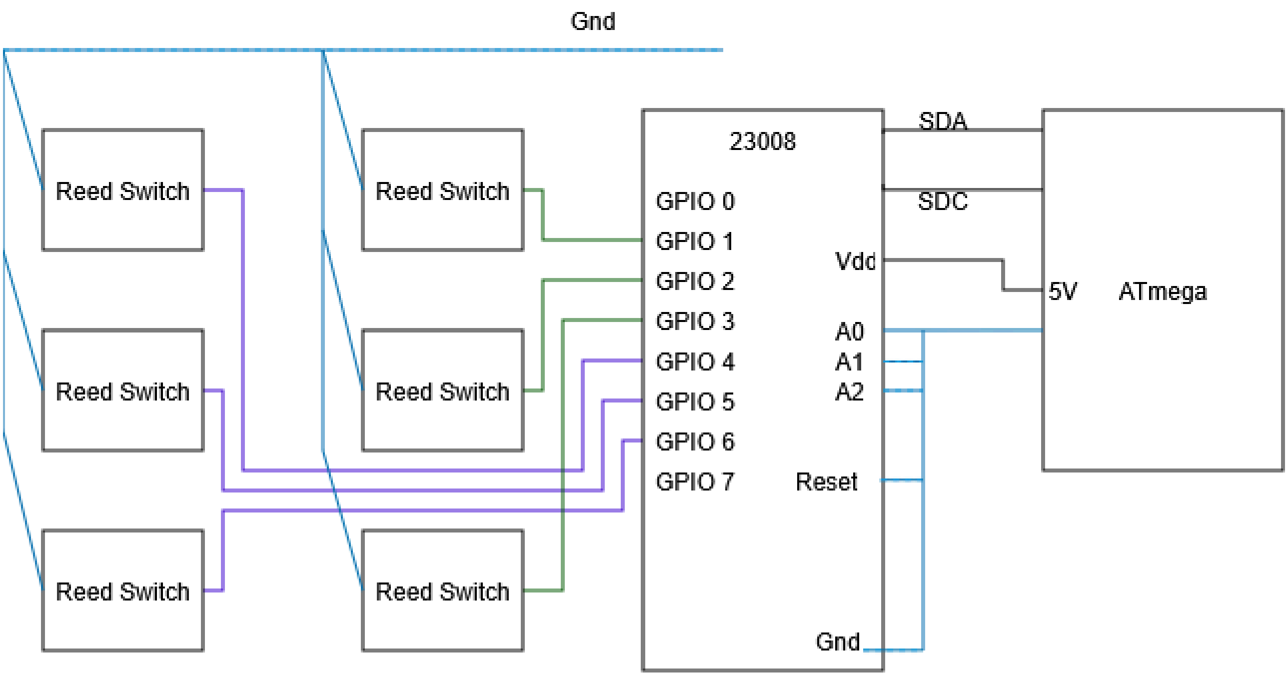
Definition of "done": control motor to rotate and stop it when the switch is pushed.

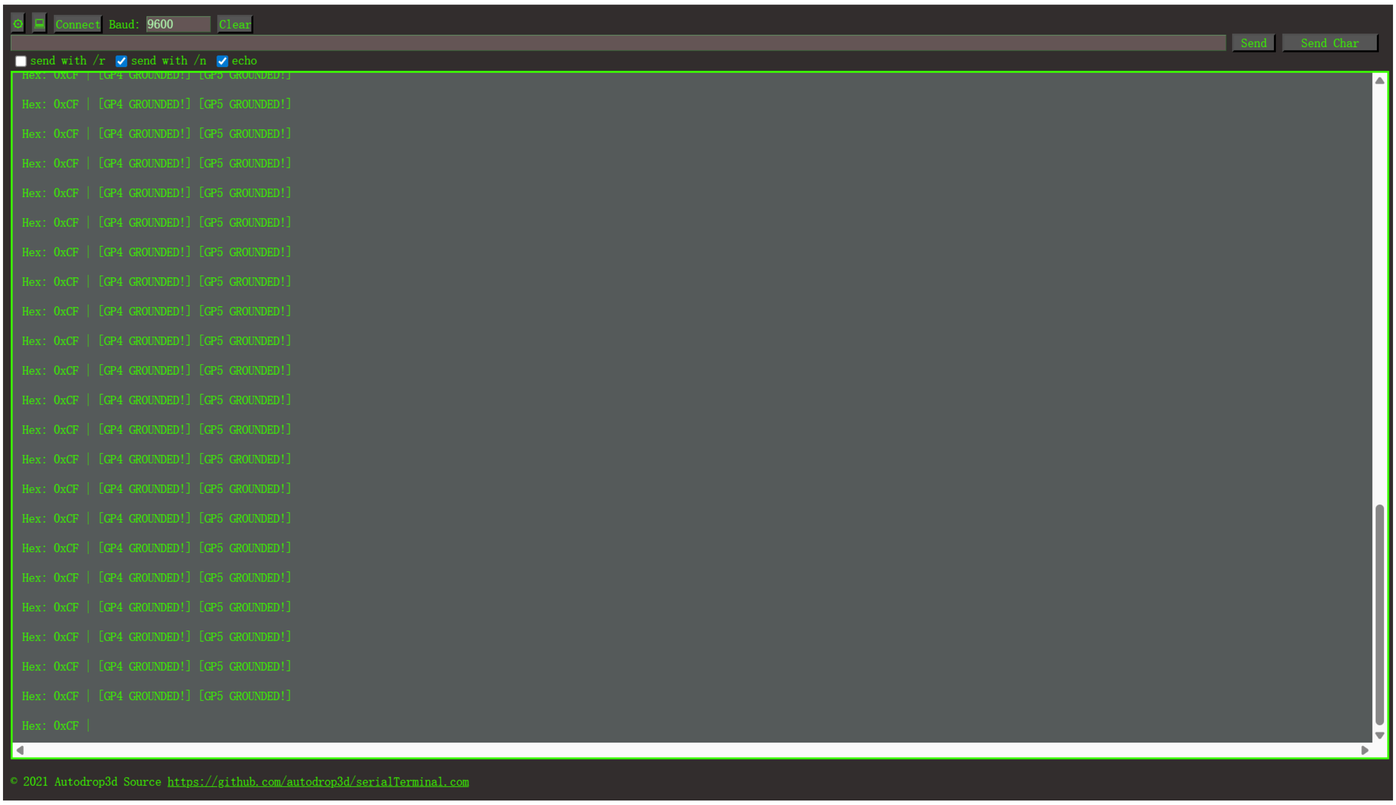
Sprint Review #2

Last week's progress

Built a basic 2x3 version of our reed switch array using 10-20AT reed switches from detkin. Wrote some test code to test the circuit.

Test Code





Current state of project

Gantry components all arrived including the electromagnet. We are still waiting on reed swtiches from Arrow and IO expanders and Muxes from Mouser. We have successfully proven that our reed switch array method will work and have basic scalable software to iterface with said array. If our GPIO expanders don't arrive in time, we can get away with using a number of the 23008s in detkin.

Next week's plan

Write the software to drive stepper motors using our motor drivers. (1 - 2 weeks) Will finish it together.

Definition of "done": Drive the stepper motor to rotate and stop it when the switch is pressed.

Start printing electromagnet taxi and gantry brackets.

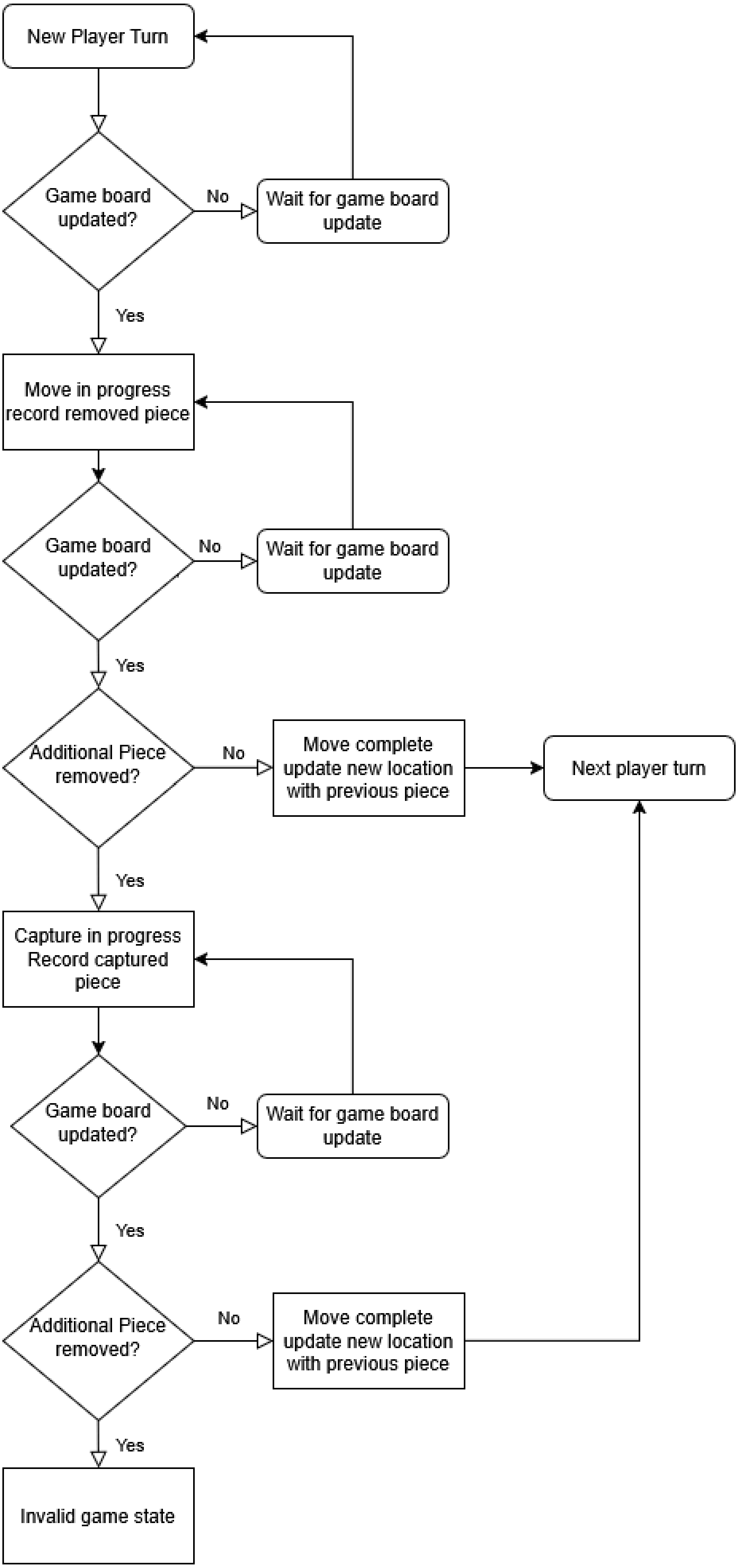
Build the gantrey enough to mount motors. (1 - 2 weeks) Will finish it together.

A reach goal would be to demonstrate control of the gantry using a reed switch array. Chess game state plan has been finalized.

Connect multiple IO expanders in series and monitor the status of more reed swtiches. (1 week) Will finish it together.

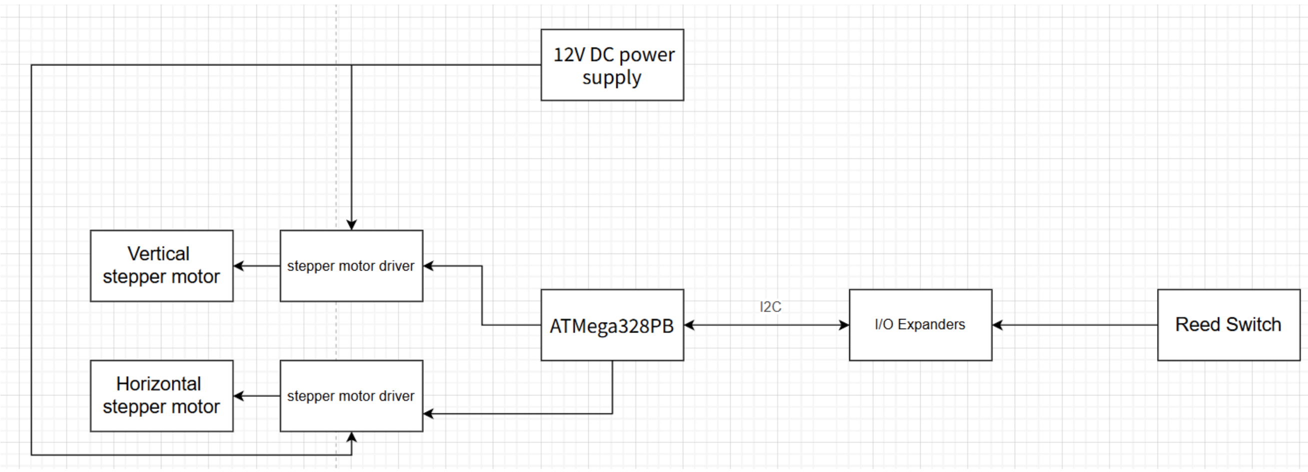
Definition of "done": Monitor the status of at least 10 reed swtiches at the same time.

The game board requires a de-elevated "graveyard" region for captured pieces. Since the MCU only has boolian states for each game square, we require additional button controls and human interaction procedure to indicate the difference between captureing specific pieces. So we will create a test circuit with both a switch array and buttons to show the game capture logic can differinciate a basic movement and a piece capture.



MVP Demo

1. Show a system block diagram & explain the hardware implementation.



   Now we have the I/O expanders to read the state of read switch. We use 8 I/O expanders, and each I/O expanders ican be connect to 8 read switches. We use two stepper motor drivers to drive two stepper motors and use 12v power source to provide 12v for the drivers.

2. Explain your firmware implementation, including application logic and critical drivers you've written.

   Our firmware will scan the state of read switch and save the state of of read switch by the array. We have the function to send pulses to motor and control the number of turns that motor rotate.

3. Demo your device.

4. Have you achieved some or all of your Software Requirements Specification (SRS)? Yes, we have achieved some of SRS. Our state scanning is fast and the motor can be stopped by the switch.

   1. Show how you collected data and the outcomes.

5. Have you achieved some or all of your Hardware Requirements Specification (HRS)? Our assembly was not complete, so we did not test it.

   1. Show how you collected data and the outcomes.

6. Show off the remaining elements that will make your project whole: mechanical casework, supporting graphical user interface (GUI), web portal,

etc.

We will Implement the gantry and electromagnet, and assemble them with motor, board and the read switch later.

7. What is the riskiest part remaining of your project? The magnets of the chess pieces and electromagnet might influence the chess pieces on the path.
    1. How do you plan to de-risk this? We need to bulid a larger chessboard and control the power of an electromagnet.
8. What questions or help do you need from the teaching team?

Final Project Report

Don't forget to make the GitHub pages public website! If you’ ve never made a GitHub pages website before, you can follow this webpage (though, substitute your final project repository for the GitHub username one in the quickstart guide): https://docs.github.com/en/pages/quickstart

1. Video

[Insert final project video here]

- The video must demonstrate your key functionality.
- The video must be 5 minutes or less.
- Ensure your video link is accessible to the teaching team. Unlisted YouTube videos or Google Drive uploads with SEAS account access work well.
- Points will be removed if the audio quality is poor - say, if you filmed your video in a noisy electrical engineering lab.

2. Images

[Insert final project images here]

Include photos of your device from a few angles. If you have a casework, show both the exterior and interior (where the good EE bits are!).

3. Results

What were your results? Namely, what was the final solution/design to your problem?

3.1 Software Requirements Specification (SRS) Results

Based on your quantified system performance, comment on how you achieved or fell short of your expected requirements.

Did your requirements change? If so, why? Failing to meet a requirement is acceptable; understanding the reason why is critical!

Validate at least two requirements, showing how you tested and your proof of work (videos, images, logic analyzer/oscilloscope captures, etc.).

ID	Description	Validation Outcome
SRS-01	The IMU 3-axis acceleration will be measured with 16-bit depth every 100 milliseconds +/-10 milliseconds.	Confirmed, logged output from the MCU is saved to "validation" folder in GitHub repository.

3.2 Hardware Requirements Specification (HRS) Results

Based on your quantified system performance, comment on how you achieved or fell short of your expected requirements.

Did your requirements change? If so, why? Failing to meet a requirement is acceptable; understanding the reason why is critical!

Validate at least two requirements, showing how you tested and your proof of work (videos, images, logic analyzer/oscilloscope captures, etc.).

ID	Description	Validation Outcome
HRS-01	A distance sensor shall be used for obstacle detection. The sensor shall detect obstacles at a maximum distance of at least 10 cm.	Confirmed, sensed obstacles up to 15cm. Video in "validation" folder, shows tape measure and logged output to terminal.

4. Conclusion

Reflect on your project. Some questions to address:

- What did you learn from it?
- What went well?
- What accomplishments are you proud of?
- What did you learn/gain from this experience?
- Did you have to change your approach?
- What could have been done differently?
- Did you encounter obstacles that you didn’t anticipate?
- What could be a next step for this project?

References

Fill in your references here as you work on your final project. Describe any libraries used here.