GameFrame

Group 16
Allen Chion
Frank Weeks
Israel Soria
Levi Masters

Motivation

- Each of our members wanted to implement certain skills and gain experience to put down on our résumé, such as AI
- Most projects are about something being useful, so we wanted to build something fun
- Help inspire future generations about STEM
- Provide a mobile arcade style experience

Goals and Objectives

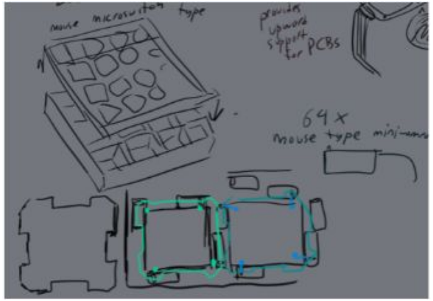
- Create a small and portable board game device
- Lightweight
- Relatively easy to use
- Long battery life

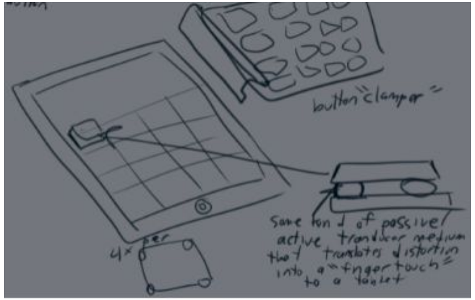
Specifications

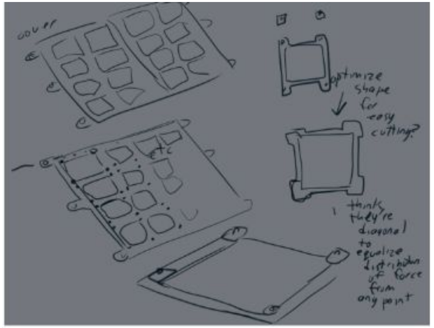
Weight	Less than 2lb
Dimensions	Box: 15.925in by 10.125in by 5.55in
Battery life	2-4hrs
Speed	Less than 5ms response time
Monetary	Should not exceed $400
Software	Should be able to at least play chess

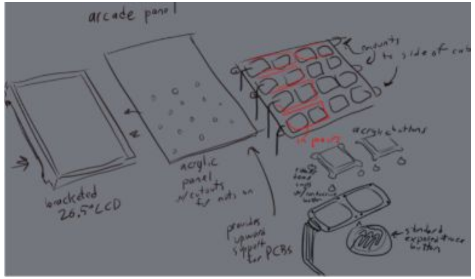
Design Diagram

- Cut costs
- Sustainability
- Button satisfaction
- Functional controller systems



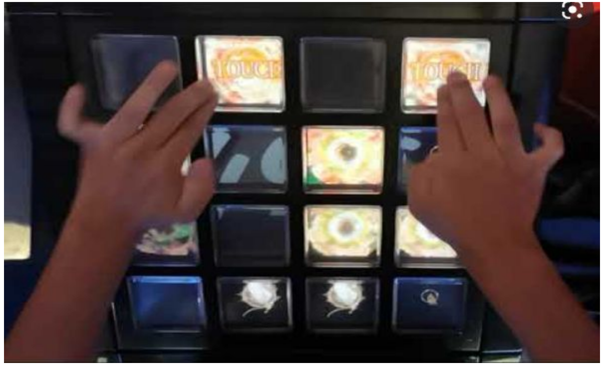

button "clamper"
4x per
Some kind of passive/active transducer medium that translates distortion into a "finger touch" to a tablet


cover
etc
optimize shape for easy cutting?
1. thinks they're diagonal to equalize distribution of force from any point


arcade panel
mounts to side of cab
acrylic buttons
bracketed 26.5" LCD
acrylic panel w/cutouts for nuts on
provides upward support for PCBs
standard expensive arcade button

Approach

- We wanted this to be portable
- Use a 8x8 button layout for the user to interact with it
- Similar to how players interact with Jubeat, but all of the UI elements will also be done through these buttons
- For example, the chess pieces will appear on each block and pressing the button will select that piece
- The screen will be under the button layout

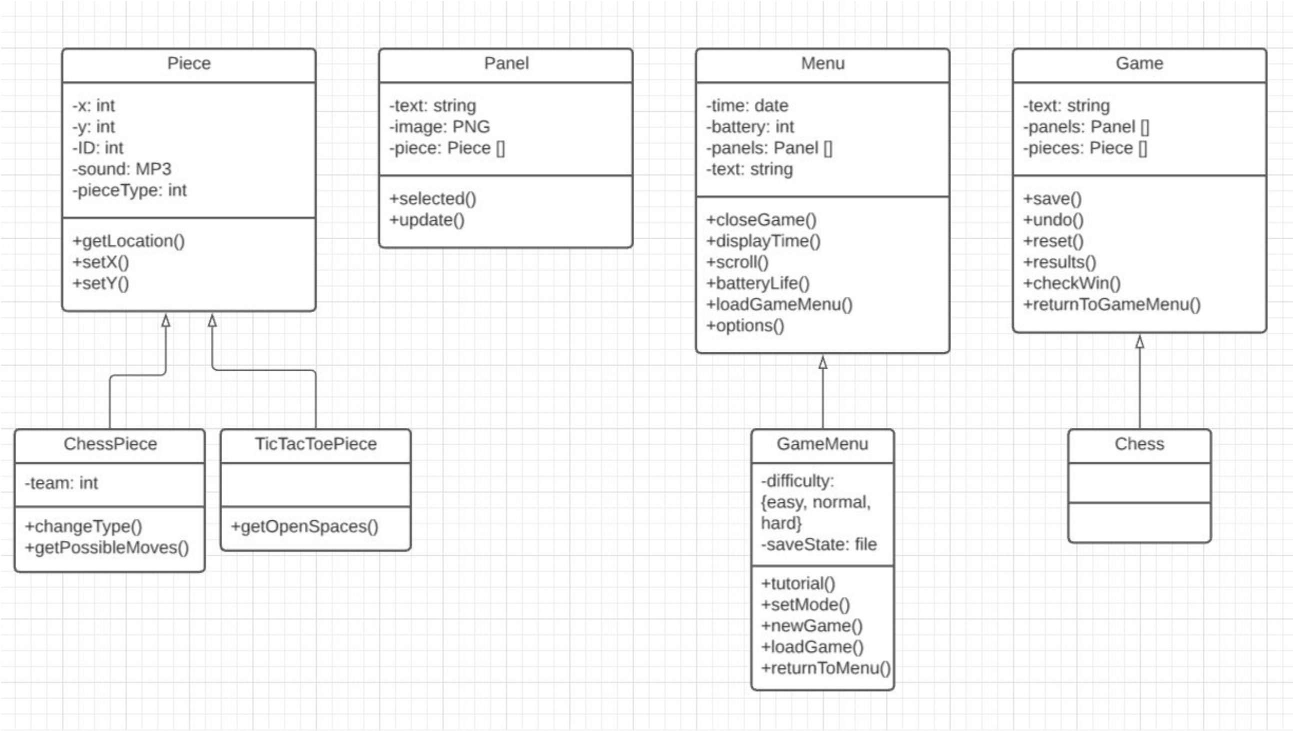


Software

Tools

- Python
  - Keep everything the same
  - Assortment of libraries
- Github
  - Keep track of our code
- Discord
  - Communication

Class Diagram



Piece

-x: int
-y: int
-ID: int
-sound: MP3
-pieceType: int

+getLocation()
+setX()
+setY()

Panel

-text: string
-image: PNG
-piece: Piece []

+selected()
+update()

Menu

-time: date
-battery: int
-panels: Panel []
-text: string

+closeGame()
+displayTime()
+scroll()
+batteryLife()
+loadGameMenu()
+options()

Game

-text: string
-panels: Panel []
-pieces: Piece []

+save()
+undo()
+reset()
+results()
+checkWin()
+returnToGameMenu()

ChessPiece

-team: int

+changeType()
+getPossibleMoves()

TicTacToePiece

+getOpenSpaces()

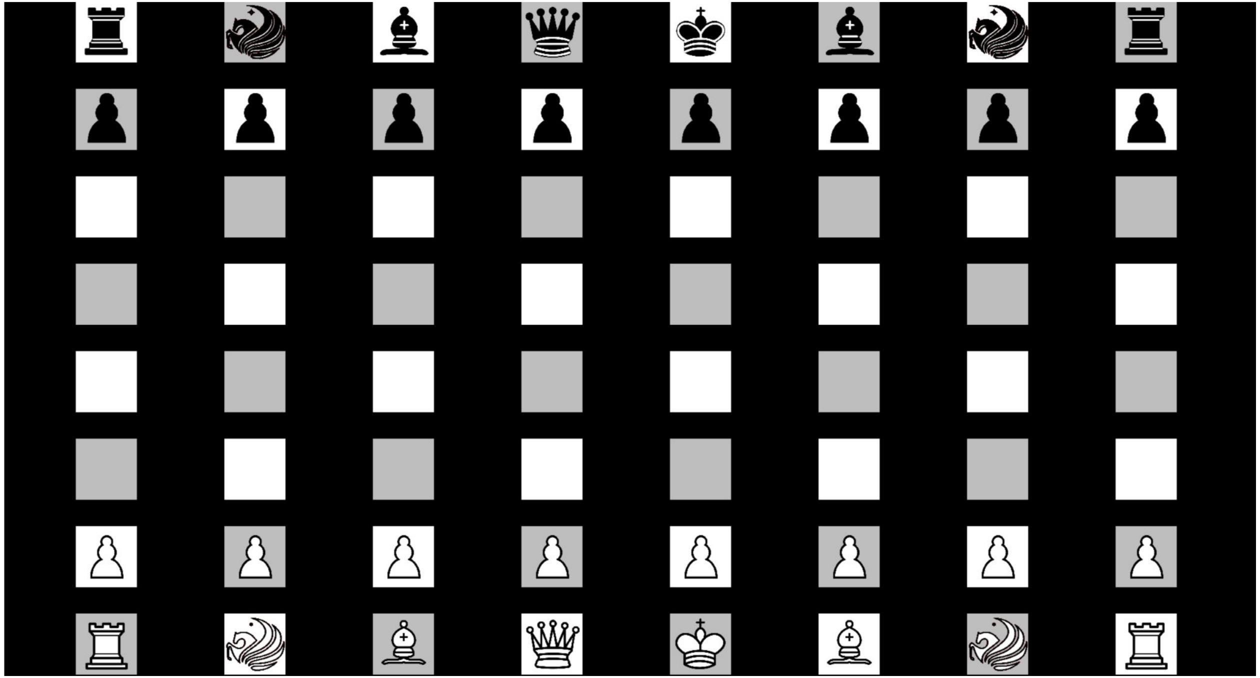
GameMenu

-difficulty: {easy, normal, hard}
-saveState: file

+tutorial()
+setMode()
+newGame()
+loadGame()
+returnToMenu()

Chess

User Interface

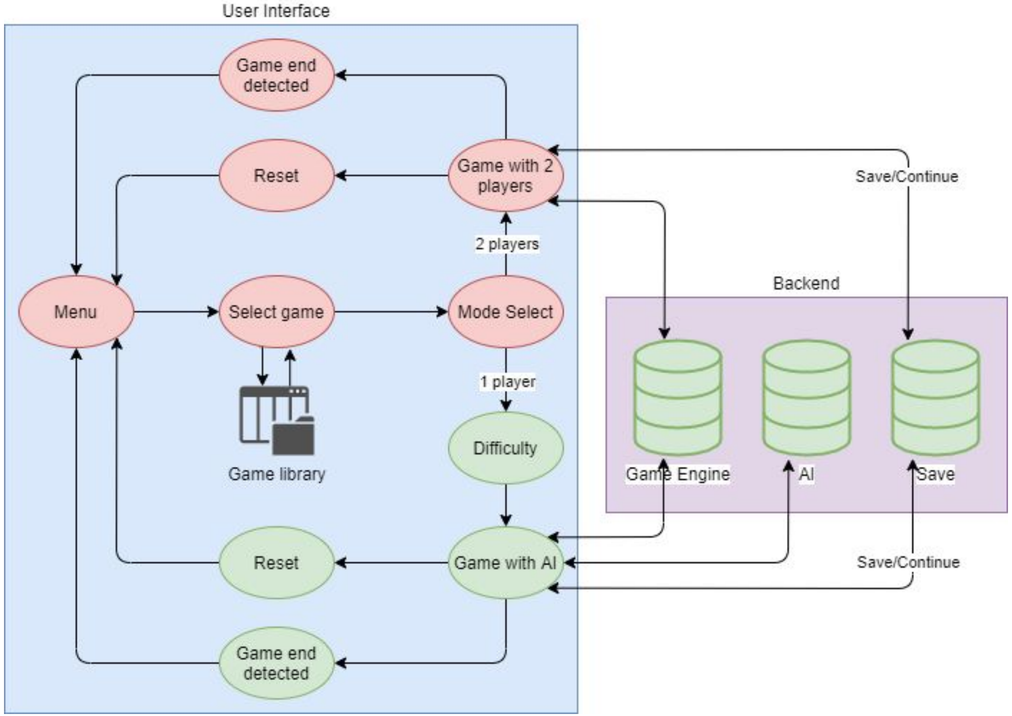


Game Example

Game Engine Testing

- Human testing to validate automated testing
    - Script of input games
    - GUI to test games
- Human testing on physical board to ensure proper usage of input and output
- Automated testing on physical board to ensure output is working correctly

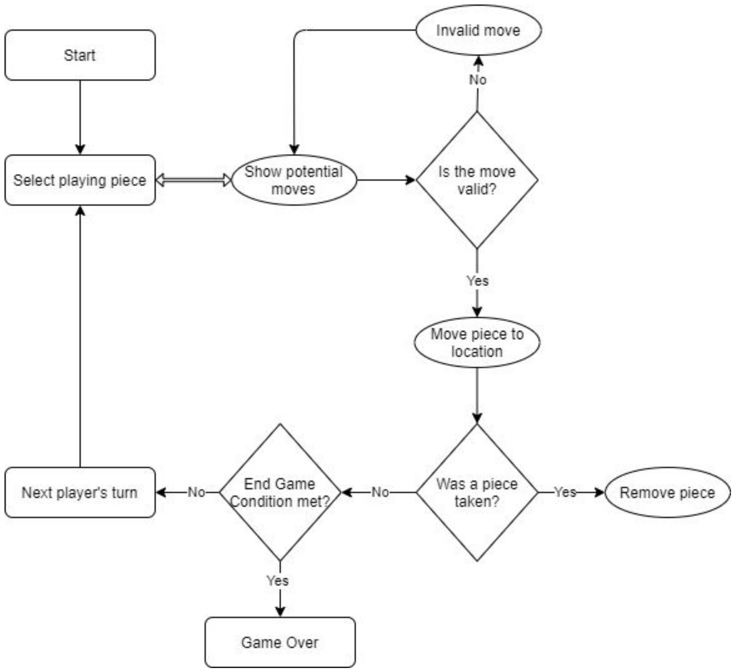
Game Engine Design



- Games are created in “select game” function
- AI is activated for solo play
- Save game is available for one game at a time
- Game engine gives “check_win()” function call after each move

Game Engine Design

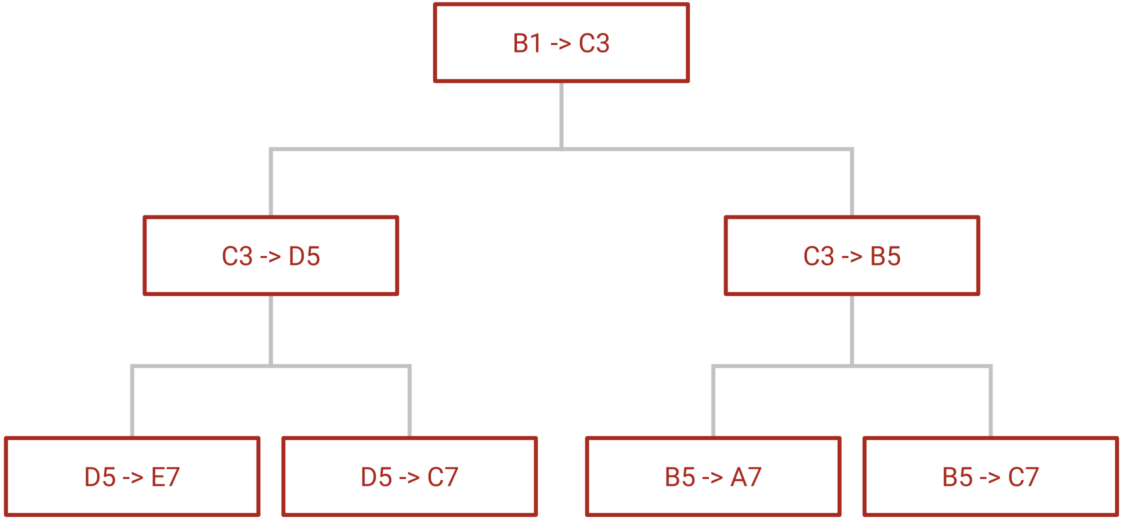
- Each tile is given a state, including what piece is present
- Each piece tells the tile valid moves
- The game engine presents the player with possible moves
- Invalid moves are ignored (no action taken, remains player's turn)
- State of pieces stored in array with attributes



AI (Player vs. Computer)

- Python/PyCuda is the main language used to program the Jetson nano
- backtracking/minimax hybrid is used for most AI computer gaming applications
- Different levels of depth are given at each stage of difficulty
  - This emulates human playing experience fairly similarly
- Monitored throughout development to gage difficulty levels

AI Backtracking Algorithm



- Backtracking
- Try a bunch of simulated games
- Choose best outcome probability
- Simulates AI and Player moves

Hardware

Hardware - Overview

Block Diagram

Microcontroller

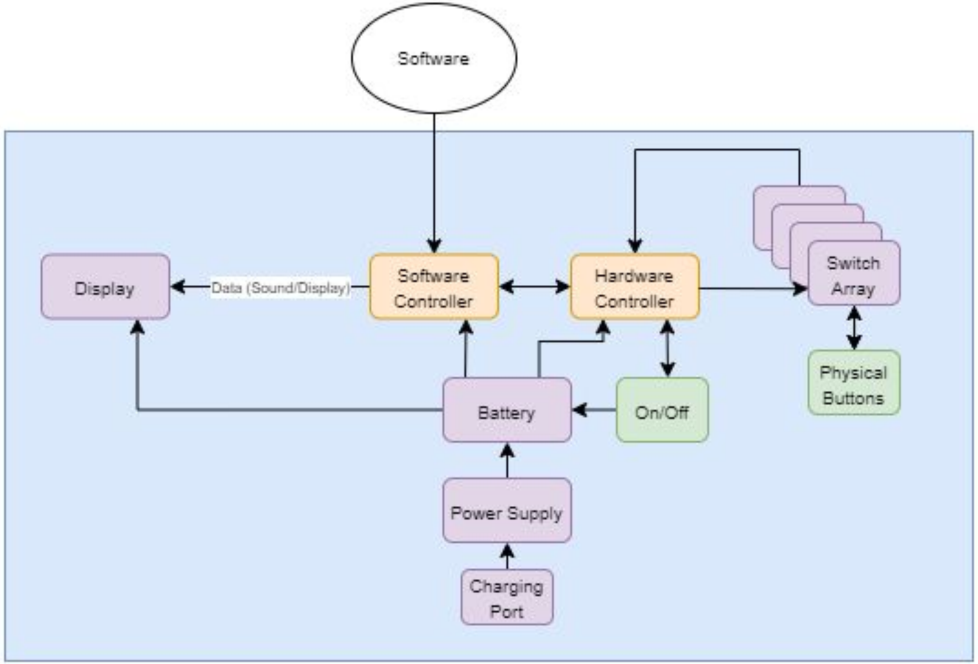
Switches

LCD

Power

PCB

Hardware - Block Diagram



Hardware - Microcontroller

MSP 430

Functions:

Communicating with Jetson Nano

Controlling and decoding switches

Hardware - Microcontroller

MSP 430 - G2ET

Reasons:

- Simple and familiar to use
- Cheap ($10-15)
- Low power (1.3-400μA)
- DIP-Socket compatible chip is available

Hardware - Shift Registers

64 buttons means 64 wires

Solution:

- 74HC595 (8-bit shift register IC)
    - One output line
- Shift register require clocks
    - Use a pin to output from MSP 430
- Two pins output for switch decoding

Hardware - Buttons/Switches

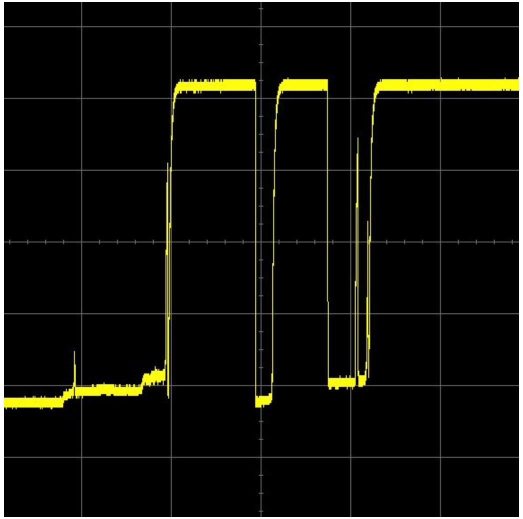
Custom Buttons

Copper Wire Grid

Switch contacts

Rubber actuators

Hardware - Switches - Debounce



Switching states can create bounce

Originally thought more important

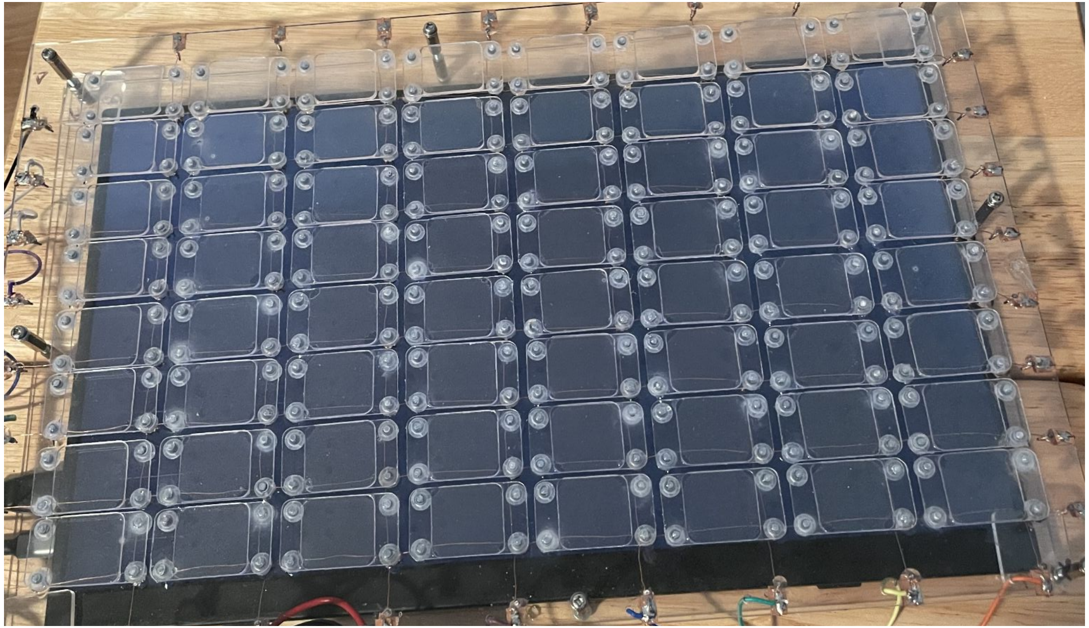
Implemented RC spot into PCB

Ended up being useful

Hardware - LCD

Features:

- HDMI compatible
- Built in sound
- 15”



Hardware - Power

Battery

- 12 VDC, 3 A max, 66Wh
- About 3-4.5 hour life

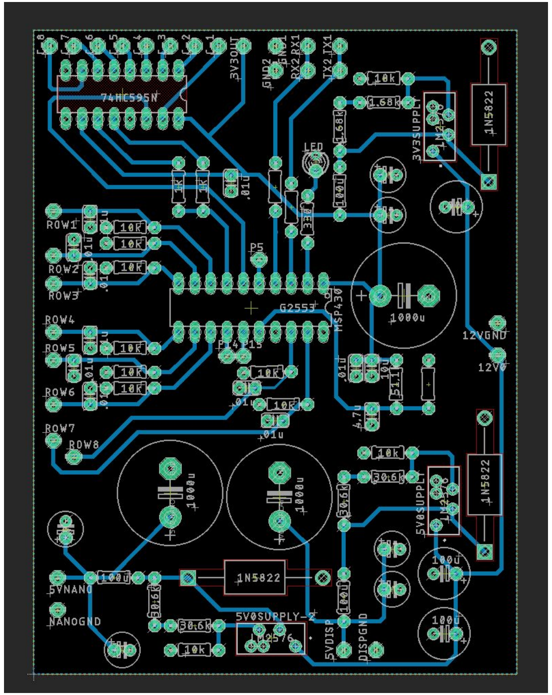
Voltage regulation

- LM2576 x 3
- Heatsinks

Hardware - PCB

PCB:

- Main
    - MSP430, Regulators, Shift register



Hardware - Testing

Power testing

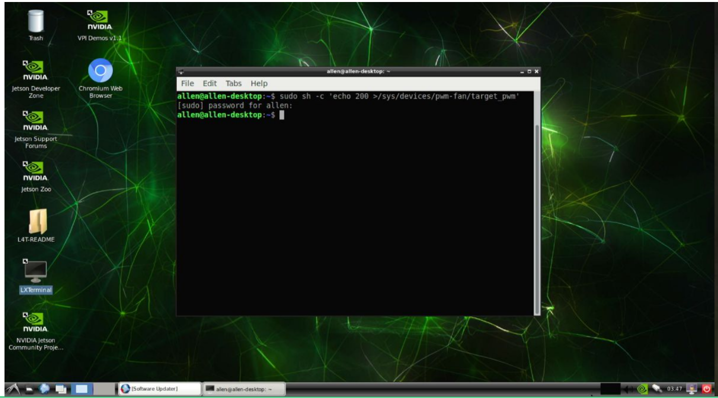
Temperature Testing

Battery Life testing

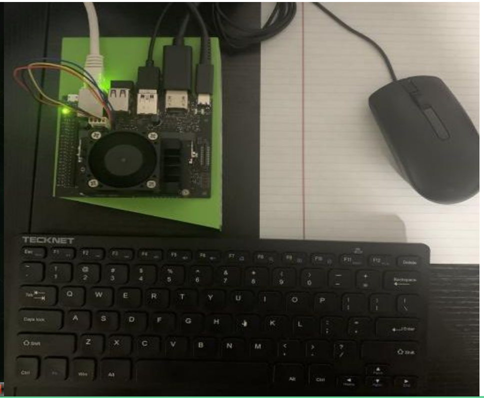
RS232-USB MSP430 Testing (UART)
Nano Python Script Testing (UART)

Jetson Nano

- We decided to go with the 2GB version because we reasoned that we would not need more than that to run simple board-style games since the 4GB version costs a lot more
- Built for AI, so this will be running our software
- Hardware can be controlled directly using the Linux terminal or running a script
- Python wrapper built in makes coding it a lot easier.

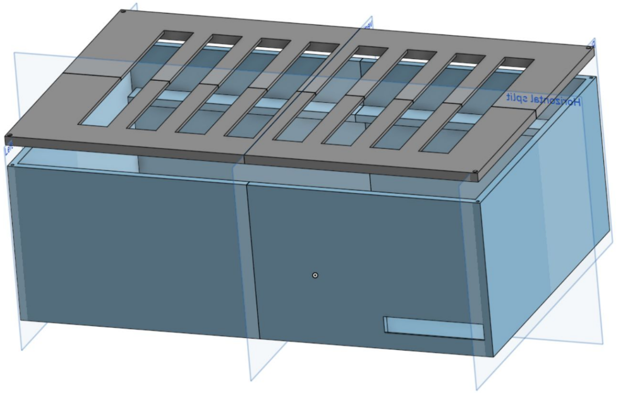


```
allen@allen-desktop:~$ sudo sh -c 'echo 200 >/sys/devices/pwm-fan/target_pwm'
[sudo] password for allen:
allen@allen-desktop:~$
```



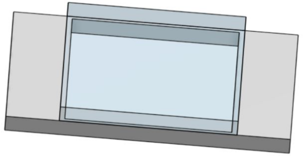
Housing

- PLA
- Box: 15.925in by 10.125in by 5.25in
- Top Frame: 15.925in by 10.125in by 5.25in by 0.3in
- Split into four cross sections due to 3D printing limitations

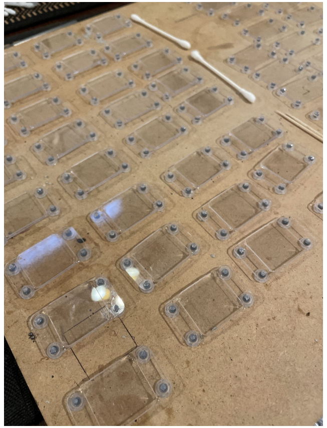


Buttons

- 1.6 in. by 0.94 in. by 0.08 in.
- Top tile piece is 1 in. by 0.94 in.
- Custom cut from PETG with CNC router



Initial Design



Final design

Design Constraints

- Economic
  - Since this project is not financed, all the funding is coming from ourselves.
  - Ideally spend the least amount of money possible, but we do not plan on cutting too many corners.
- Political/Ethical
  - We have to limit the games that are public domain and do not breach any copyright.
  - No rhythm games that use music since we do not want to deal with licensing.
  - We don’t want to just copy someone else’s game.
- Health and Safety
  - Ensuring the device does not heat up too much.
  - No exposed electronics.

Challenges

Screen limitations

- We originally wanted a big square LCD panel, but it was hard to find one and they were super expensive
- We had to drop the square design of most board games as a result:
  - Each button had their sides increased by 0.3in to accommodate the 0.3in actuators for button activation to avoid obstructing the middle viewing area, which was already a small 1in by 1in.
  - The display board for playing games is now unconventionally rectangular.
- The case design had to be relatively huge to house all our electronics, so a lot of the portability was sacrificed.

Hardware Challenges

Flipped PCB

- ICs on bottom

Feedback on wrong node

- Resistors on bottom

Lesson:

- 1-Layer PCBs are great!

Software Challenges

- Integration with hardware
- Debugging
- Multiple code iterations
- Neural network for AI was above skill level
  - Switched to using backtracking instead
- AI backtracking difficulties
  - Cloning boards
  - Opposition moves prediction
  - Quantifying moves
- Changing inputs

Budget and Financing

Item	Cost
NVIDIA Jetson Nano 2GB Developer Kit DC 5V cooling fans (2 pack)	$77.79
3D printing material	$24
Copper Wire	$9
Fan	$4
MSP430g2553	$15
4”x6” PETG Sheets	$14/10

Budget and Financing (Cables and Connectors)

Name	Price
Breakout to USB-C connector	$8
RS232 Breakout connector x 2 + Female-female RS232 connector	$16
Breakout to USB-A connector	$8/2
Dupont Cables	$14
Battery Connector	$9.50
Mini-hdmi to HDMI cable	$9
USB-RS232 (For initial testing)	$11

Budget and Financing (PCB Etching)

Name	Price
Ferris Chloride (from radio shack)	$13
4”x6” Copper Plate	$18/10

Name	Price
Rubber actuators	$3.6/20
Super glue	$3
Epoxy	$18
Drill bits for cnc milling	$16/10

Budget and Financing (BOM)

Name	Price
Misc RLC/Diode/heat sink/cheaper ICs/etc	~$20-30
Misc mechanical hardware components (ie screws/washers etc)	~$30
Acrylic Sheets	$13
15” LCD (with cables)	$136
Battery	$38

Budget and Financing (BOM)

Name	Price
Liquid tape	$8
Vector boards for alignment	$18/6
Plexiglass tiles	$12.50/20

Total	$568.39

Work Distribution:

- Levi Masters:
    - Game engine and AI
    - Backend
- Israel Soria:
    - UI elements
    - Front end
- Frank Weeks
    - Hardware
    - Hardware and software integration
- Allen Chion
    - CAD Design
    - Debugging