IN1901

Microcontroller Based Application Development Project

Final Report

OMI THE TRUMPS

Supervisor :- Mr. B. H. Sudantha

Group No : 34 **Group Name :** OneOneZero

Index Number	Name
234052X	H.B.C.Dhananjaya
234223A	S.D. Weerasinghe
234235L	W.M.D.N. Wijerathne
234118G	M.H.N.Kumarasiri
234056M	P.D.M.Dilshari

Bachelor of Science Honours in Information Technology (BScHons (IT))
Faculty of Information Technology
University of Moratuwa

Content

1 Introduction

Omi, a traditional card game celebrated for its strategic depth and interactive gameplay, has been a popular pastime across generations. However, the game is often associated with several limitations, including the time intensive nature of manual card distribution, inadequate privacy in card management, and the potential for dishonest practices. Additionally, the lack of technological integration has constrained its relevance and appeal in the context of an increasingly digital society.

This project, titled *Omi the Trumps*, seeks to address these limitations by introducing a hardware-based solution that incorporates digital and interactive features. By employing private LED panels, wired remote controls, and an embedded system powered by a Raspberry Pi, the project aims to modernize the traditional Omi card game while preserving its strategic integrity. This approach aspires to enhance the gameplay experience, ensuring greater efficiency, security, and engagement for contemporary players.

1.1 Problem in Brief

Traditional Omi gameplay faces several challenges that hinder its appeal and fairness. The manual card management process is time-consuming and prone to human errors, disrupting the flow of the game. Additionally, the lack of privacy often allows players to unintentionally or intentionally view others' cards, compromising the integrity of the gameplay. Cheating risks further exacerbate this issue, as the absence of secure mechanisms enables dishonest practices. Moreover, the lack of technological integration makes the game less attractive to modern audiences who expect innovative and engaging experiences. These limitations underscore the need for a solution that blends the timeless strategic appeal of Omi with modern hardware innovations to ensure fair, efficient, and captivating gameplay.

1.2 Significance of Study

The significance of this study lies in its ability to modernize the traditional Omi card game while preserving its cultural and strategic essence. By addressing issues such as time-consuming card distribution, lack of privacy, and opportunities for cheating, the project enhances the gameplay experience through secure, automated, and interactive mechanisms. Additionally, the integration of technologies such as private LED panels and a Raspberry Pi-based embedded system ensures the game's relevance in the digital era, promoting fair play and engaging a tech-savvy audience. This study serves as a valuable framework for blending tradition with innovation.

1.3 Aim and Objectives

Aim:

To modernize the Omi card game by integrating advanced hardware features, thereby enhancing fairness, privacy, and engagement while preserving its traditional essence.

Objectives:

- Streamline card distribution to save time.
- Enhance privacy for discreet card management.
- Integrate anti-cheating mechanisms for fair play.
- Combine traditional gameplay with modern innovations.
- Provide a user-friendly, engaging gaming experience.

2 Literature Study

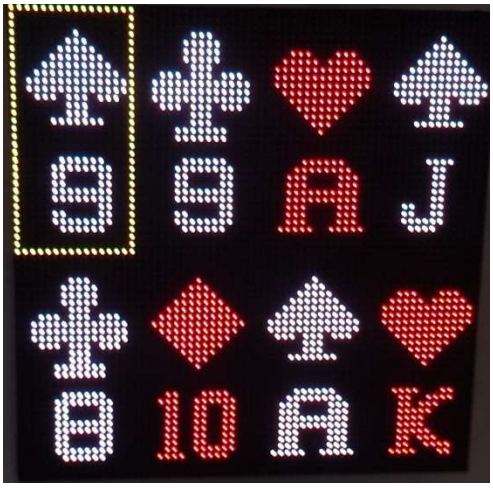
Previous research in digitizing card games has primarily focused on single player digital applications or basic two-dimensional interfaces, often falling short in addressing the needs of multiplayer interaction or integrating advanced hardware. The *Omi the Trumps* project addresses these shortcomings through a sophisticated three-dimensional, hardware-driven approach. It employs a Raspberry Pi to manage game control, including inputs, logic, and LED panel updates. RFID sensors securely track token transactions for accurate scoring, while LED panels provide private and clear card displays, ensuring player privacy. Additionally, wired remotes offer intuitive gameplay and seamless card selection. Together, these features create an immersive and engaging gaming experience, effectively overcoming the traditional limitations of Omi.

3 Proposed Solution

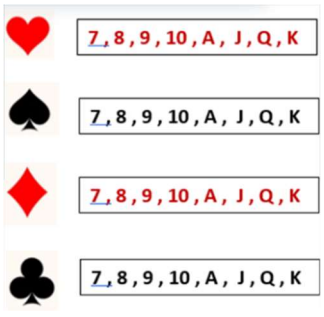
The Omi the Trumps project introduces an innovative approach to the classic Omi card game. By integrating hardware features and advanced game logic, it creates a fair, engaging, and modernized gameplay experience.

3.1 Features of the Proposed Solution

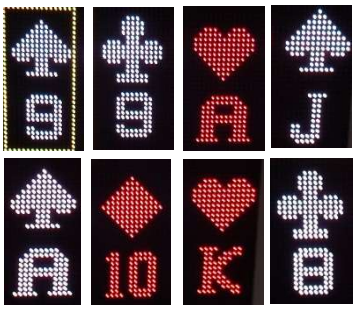
Each player has an individual display to view their cards securely.



64 x 64 RGB LED Matrix Panel



Four Suits

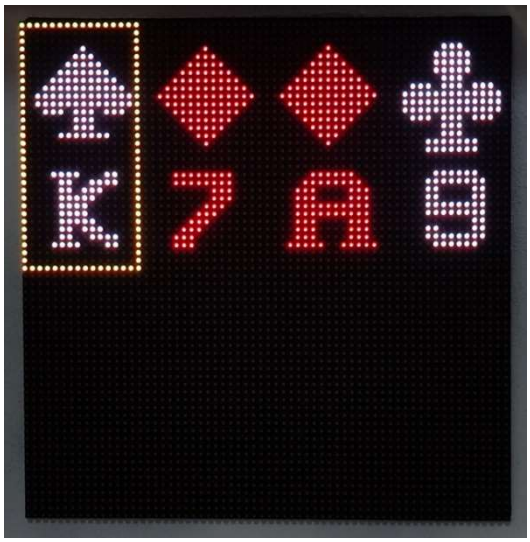


16 x 32 LED Slots

- Yellow background indicates the current selection card.

3.1.1 Private LED Panels:

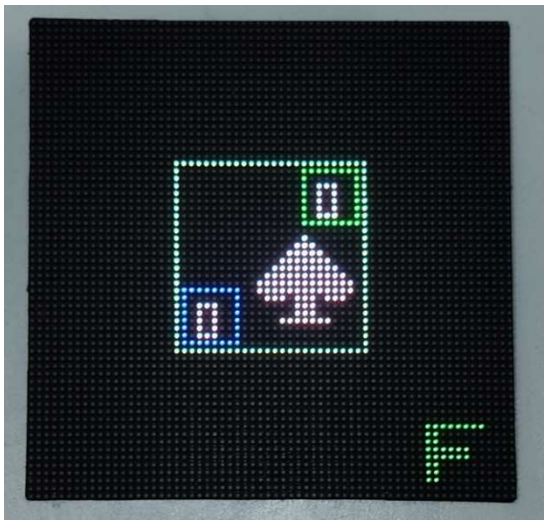
- Initially, Team B's Player 2 receives their first four cards. The other players cannot see their cards at this stage.



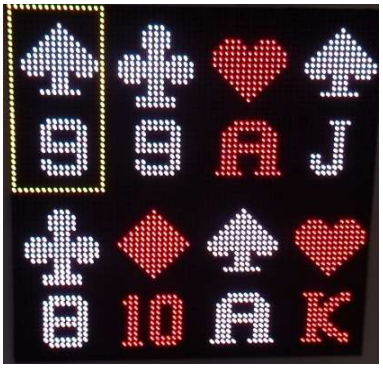
- During this time, the main display prompts player to select their trump card.

- When he chooses the trump, the main display shows it like this.

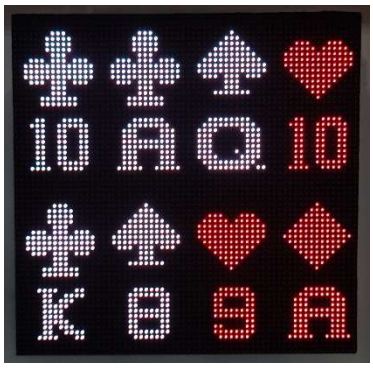
- After that, Trump stays in the middle and creates a marks panel around it.



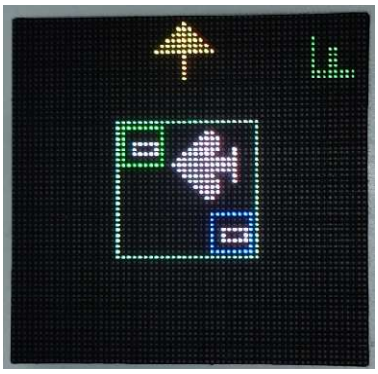
- Then, all users' cards are displayed on their respective screen.
- And they can navigate through their remotes to a select a card.

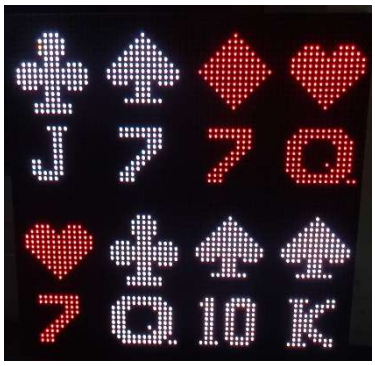


Team B
Player 2

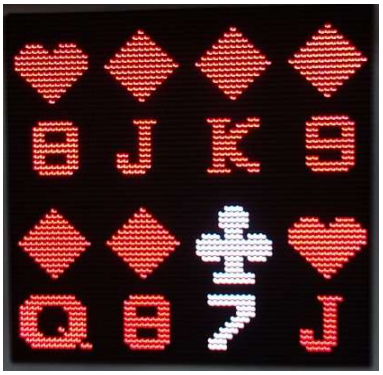


Team A
Player 3



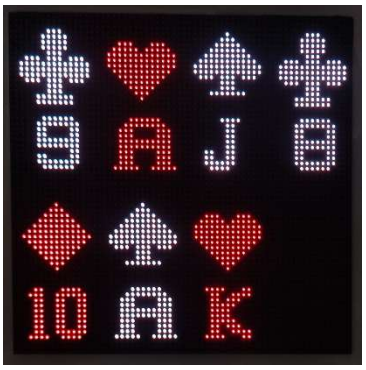


Team A
Player 1

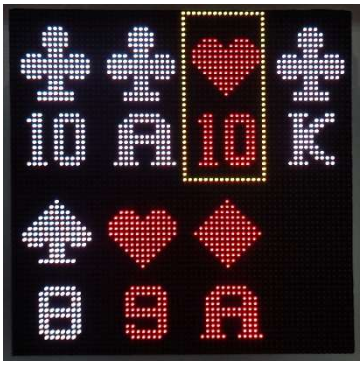


Team B
Player 4

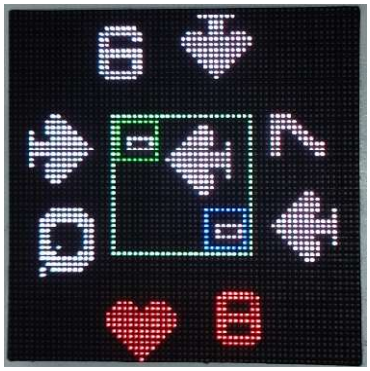
- After selecting their cards, they will be displayed on the main screen one by one.

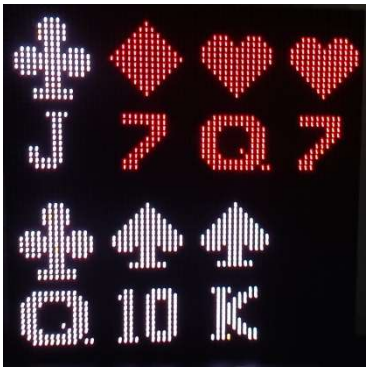


Team B
Player 2

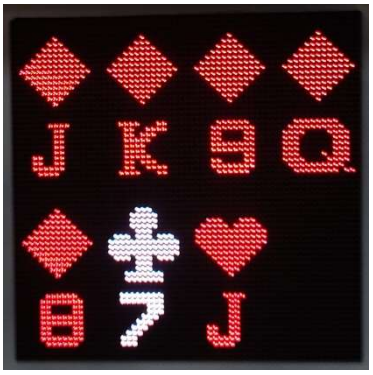


Team A
Player 3





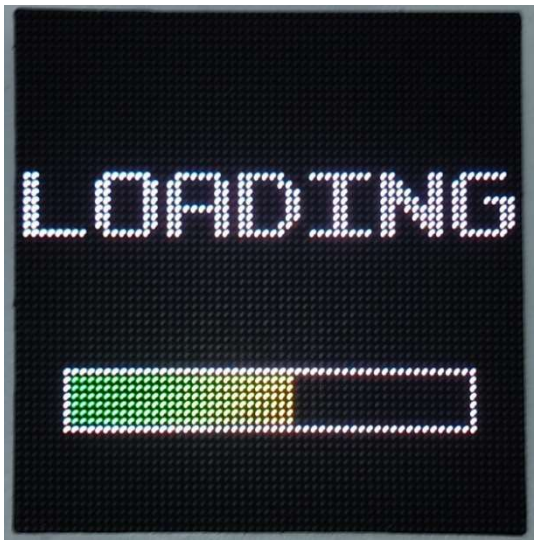
Team A
Player 1

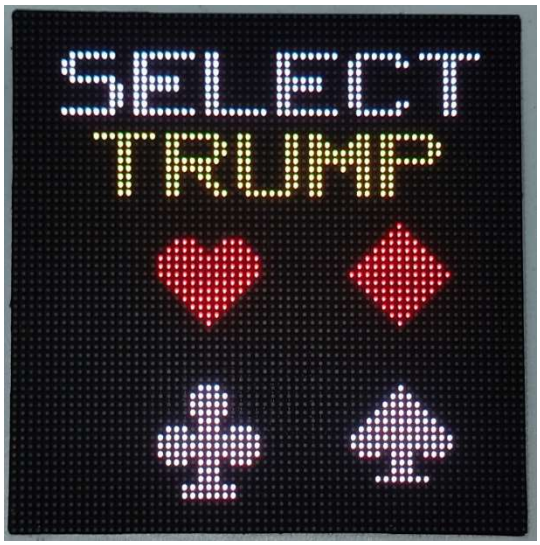


Team B
Player 4

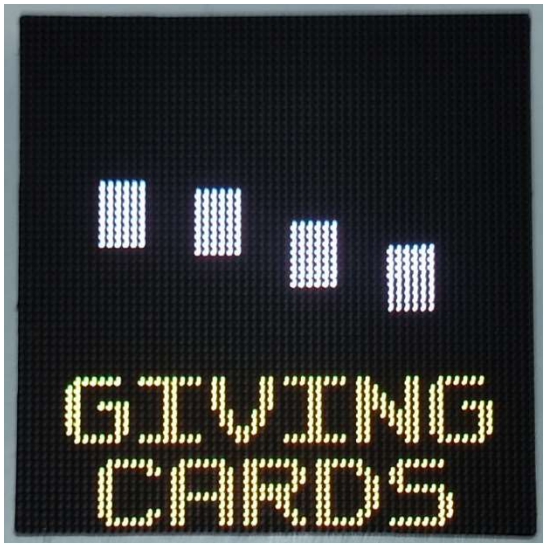
3.1.2 Central Display:

- When the game starts, the introduction and welcoming are shown on this display.

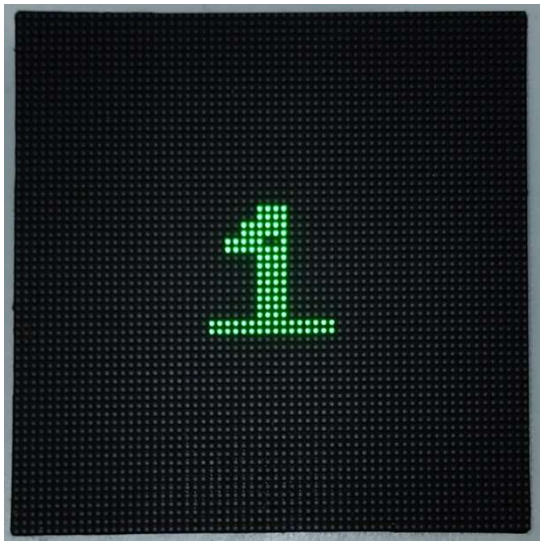




- After the trump is chosen by the player, all cards are given to the players and the game begins.



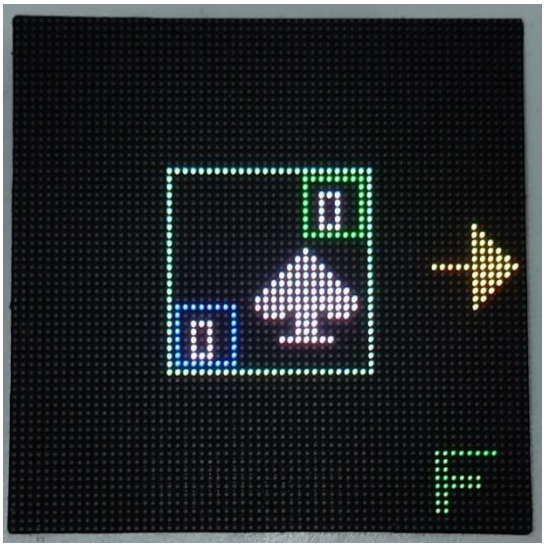
(While this happens, each player's cards are shown on their own display.)

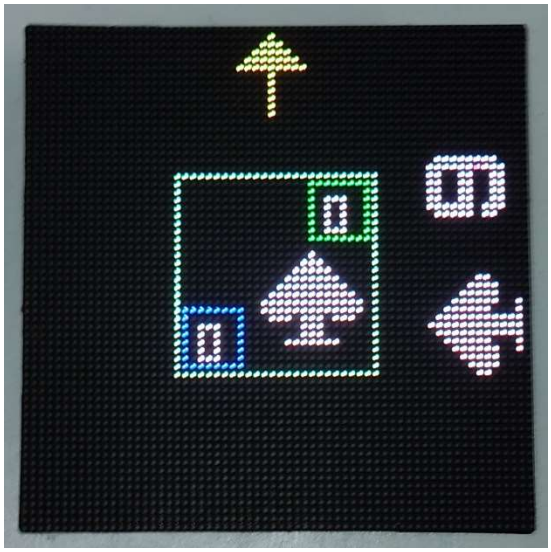


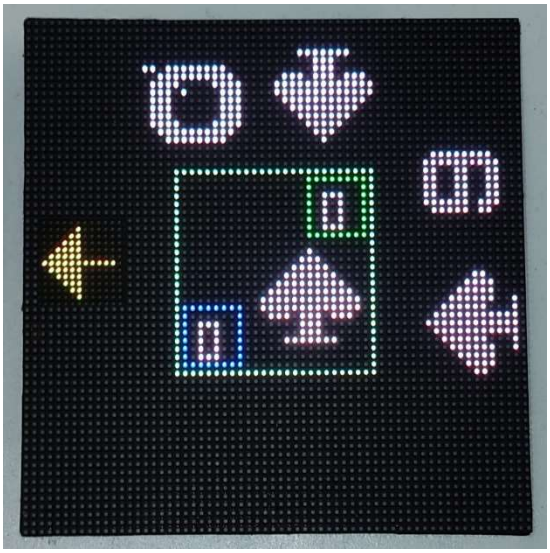
(And there's a countdown before the game begins.)

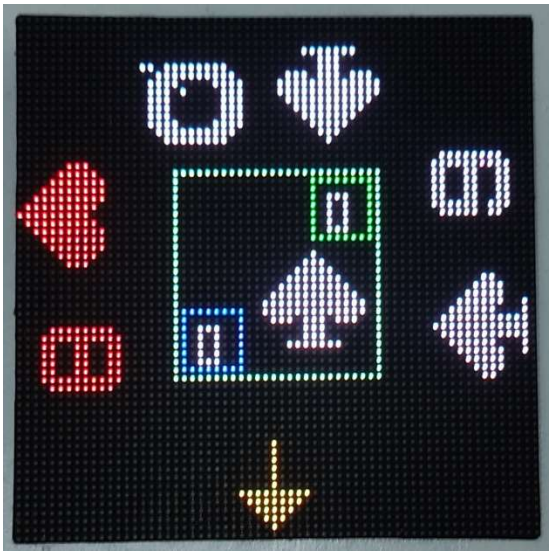
After that, the main display looks as follows:

- The trump is in the middle, and each team's marks are shown around it.
- The green "F" symbol means players can play full court in this hand.
  (Full court will be explained later.)
- The blinking yellow arrow indicates which player has to play.







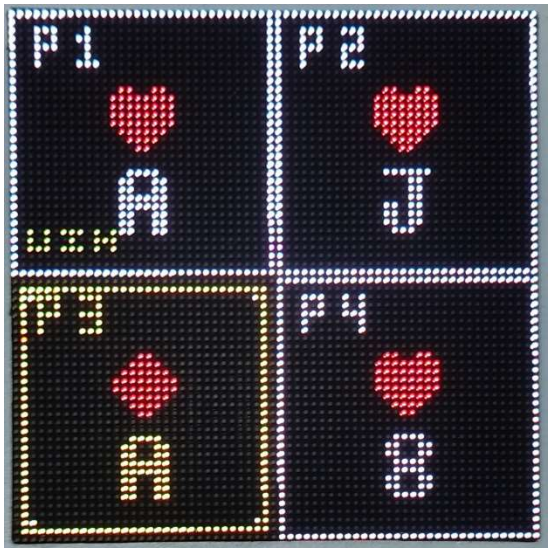


The following things are also shown on the central display.

- When an invalid card is selected.

- When a new round begins.

- To show the previous hand.



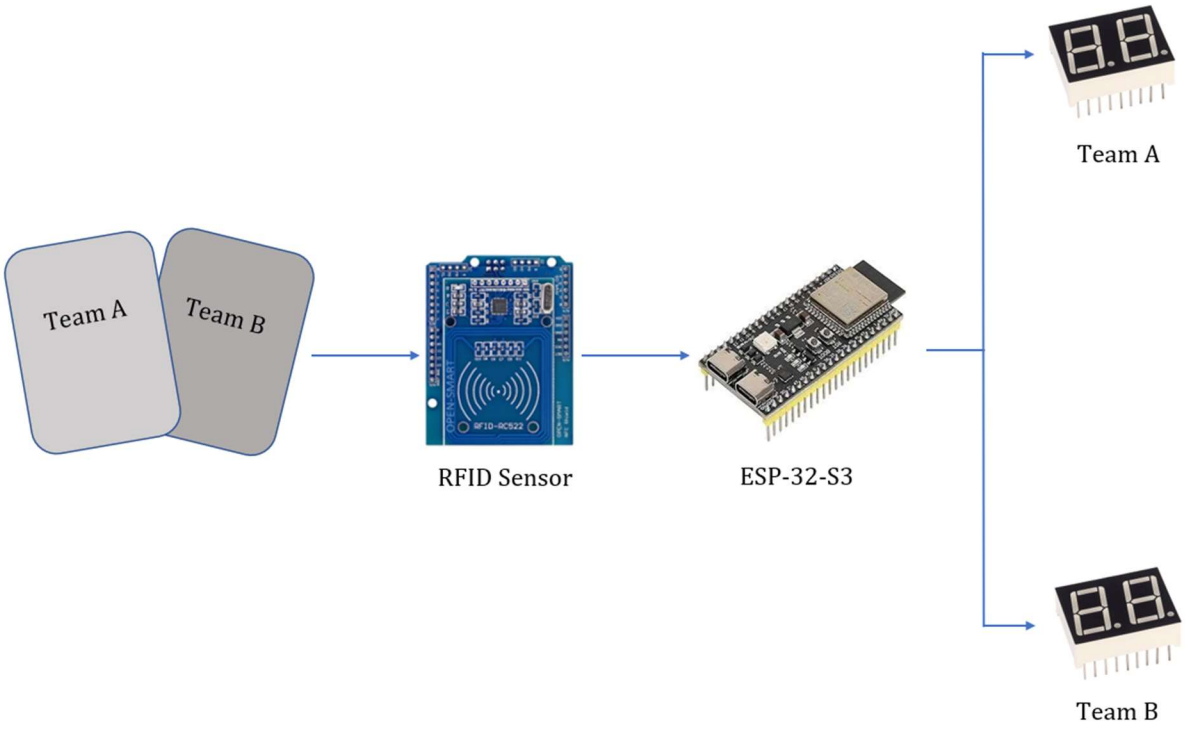
To see the previous hand, any player can ***navigate to the bottom card row on their display and press the down arrow button on their remote.***

3.1.3 Wired Remotes

- Players can easily navigate and select cards using handheld remotes.

3.2 RFID Token System:

- Tracks and deducts tokens in real-time, ensuring fair and dynamic gameplay.



3.3 Game Logic Implementation:

- Automated Card Distribution to ensures randomness and eliminates human errors.
- Facilitates scoring and rule enforcement through a Raspberry Pi.
- Above shows how the cards are mainly distributed randomly and how they are displayed on the central panel. Now, let's dive into the game logic.

3.3.1 Basic Game

- Each player gets 8 cards for a match, so players have 8 rounds to play.
- The main panel shows the marks, indicating how many rounds each team has won.
- When all 8 rounds are over, the team that has won the most rounds wins a match.

- The above image shows a scenario where Team A's Player 1 won a round, and Team A won the match against Team B, as displayed on the central panel.

- If both teams have the same points after 8 rounds, it is a draw.

- When a team loses all ten tokens, the game is over. The team with the remaining tokens wins the game.

3.3.2 Token Function

- When a match ends, the losing team must give tokens.
- Each team starts with 10 tokens. The losing team gives the specified number of tokens to the game by scanning their RFID card.
- When a player scans their card through the RFID, it is shown on their 2-digit 7-segment display and also on the central display
  (The ESP32-S3 and Raspberry Pi are connected through Wi-Fi.)
- The following shows how it is displayed on the central panel & 2-digit 7-segments.

- This is how tokens are reduced live on the main display.

3.3.3 Full Court Mode

When a player, other than the trump-declaring player sees a chance to win all 8 rounds, they can turn on Full Court mode. The catch is, if they fail to win even one round, the Full Court mode is finished, and the player's team must give 3 tokens.

- To activate Full Court mode, the player must **press the select button on their remote before the first player selects their card.** Then it shows like this.

3.3.4 Half Court mode

After the trump is declared, the first four cards are distributed to each player. If a player, other than the trump-declaring player, sees a chance to win all four rounds, they can activate Half Court mode. The catch is, if they fail to win even one round, the half Court mode is finished, and the player's team must give 2 tokens.

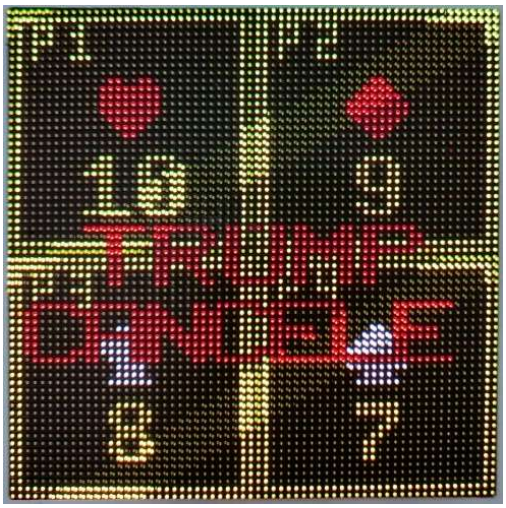
- To activate Half Court mode, the main panel displays a message after the first four cards are distributed, with a 5-second countdown, asking whether anyone will play Half Court. If you do, press the enter button on your remote.

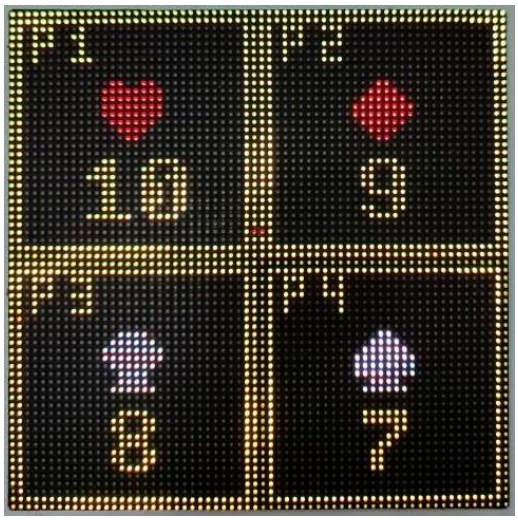
- When Half Court is being played, the activating player's teammate does not play that round. Their display remains empty during that time.

3.3.5 Cancelling Trump

When a player gets a chance to select the trump, if all 4 cards are equal or under value 10, the player has the option to cancel that trump.

- To activate this, the player must navigate to the card from the left side and press the up arrow button on the remote.





- When cancelling the trump, the main display shows the 4 cards the player received and cancels the trump.
- After cancelling the trump, that player can select the trump again for the same round.

3.4 Components and Technologies required for the proposed solution

3.4.1 Raspberry Pi 4 Model B

- The **Raspberry Pi 4 Model B** serves as the core processing unit of the *Omi the Trumps* system . This powerful single – board computer is responsible for controlling the game logic , managing real - time display updates , handling player input through wired remotes , and communicating with peripheral modules like the RFID token system and 7 - segment displays. It offers enhanced performance and robust I/O capabilities, making it ideal for our interactive, hardware-intensive game environment.

- **Processor:** Broadcom BCM2711, Quad-core Cortex-A72 @ 1.5GHz
- **RAM** **:** 2GB LPDDR4
- **Ports** **:** 2× USB 3.0, 2× USB 2.0, 2× Micro-HDMI, Gigabit Ethernet, USB-C power
- **Wireless** **:** Wi-Fi 802.11ac, Bluetooth 5.0
- **GPIO** **:** 40-pin header
- **Power** **:** 5V/3A via USB-C

3.4.2 64 x 64 RGB LED Matrix Panel

- This panel is used to display each player's cards privately with bright and colorful visuals. It consists of 4096 RGB LEDs arranged in a 64×64 grid and supports smooth rendering of card graphics.

- **LED Type** : SMD2121
- **Resolution** : 64×64 (4096 LEDs)
- **Voltage** : 5V DC
- **Interface** : HUB75
- **Input Pins** :VCC,GND,DataInput

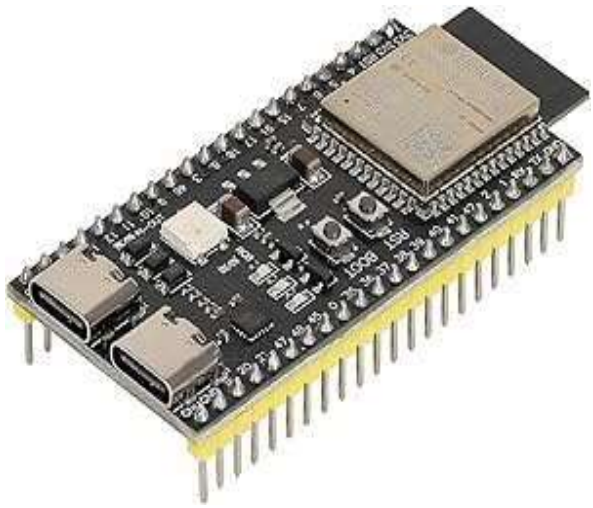
- Its high brightness and resolution ensure clear card visibility and an engaging gameplay experience.

3.4.3 ESP32-S3

- The **ESP32 - S3** module is used to process RFID token inputs .It communicates with the RFID sensor to detect token IDs and controls the 7-segment displays in real time.

- **Processor:** Dual-core 32-bit LX7 processor up to 240 MHz
- **Connectivity:** Wi-Fi 802.11 b/g/n up to 150 Mbps, Bluetooth 5.0 LE with mesh support
- **Pin Count:** 44 physical pins (including power & ground)
- **Usable GPIOs:** 36 GPIO pins (GPIO0-21, GPIO26-48)
- **Interfaces:** I2C, SPI, UART
- **Operating Voltage:** 3.0-3.6V

- Its speed and efficient communication interfaces make it ideal for handling token logic separate from the main Raspberry Pi controller.



3.4.4 RFID Sensor (RC522)

- **Function:** Used to detect and read RFID tokens when they are brought near the sensor.
- **How it works:** Utilizes radio frequency to communicate with RFID tokens within its range.
- **Unique ID Reading**: Each RFID token contains a unique identifier (UID). The RC522 reads this UID when a card is tapped or held near the reader.
- **Communication with Microcontroller:** Sends the UID data via the SPI (Serial Peripheral Interface) to the ESP32-S3 for further processing.
- **Use in the project:**
    - Acts as a player/token identifier in the game.
    - Helps manage turns by detecting which player tapped the RFID card.
    - Can be used to trigger specific game events (e.g., start a round, validate a move, update score).

3.4.5 5V 40A SMPS Power supply

- A **5V 40A Switched - Mode Power Supply (SMPS)** with a metal Casing is used to power the entire system. This supply provides a stable 5V output with up to 40A of current , sufficient to run high – demand components such as multiple 64×64 RGB LED panels, the Raspberry Pi, remotes, sensors, and other modules simultaneously.

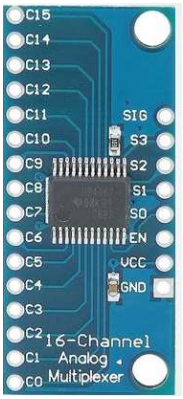
- The SMPS is compact , energy – efficient , and equipped with built-in protection mechanisms . Its metal casing aids in heat dissipation and ensures safe , reliable operation even under continuous or heavy load conditions—making it ideal for our hardware – integrated, multiplayer game environment.

3.4.6 2 Digit seven segment (Common Cathode)

- Display the token count of Team A and Team B in real-time.

3.4.7 16-Channel Analog/Digital Multiplexer (CD74HC4067)

- Two CD74HC4067 multiplexers are used to connect 4 wired remotes, with 2 remotes per multiplexer. Each chip allows 16 inputs to be read through a single signal line using 4 control pins, helping reduce GPIO usage on the Raspberry Pi.

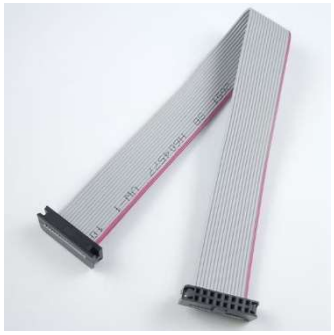


- **Channels** : 16
- **Control Pins:** 4 (S0–S3)
- **Signal Pin** : 1 (SIG)
- **Voltage** : 2V–6V
- **Signal Type** : Analog or digital
- **Use Case :** Efficiently reads inputs from 4 remotes via 2 multiplexers
- This setup enables scanning multiple buttons with minimal GPIOs.

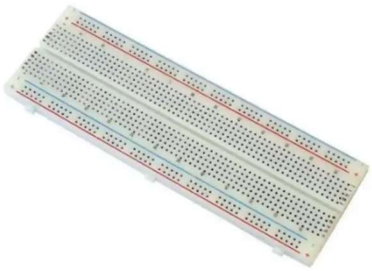
3.4.8 Wired Remote

- Simple controllers with buttons for card selection

3.4.9 Other electronic Accessories

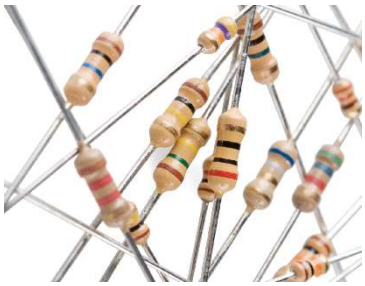


IDC Cables



Bread Board

JST Connectors



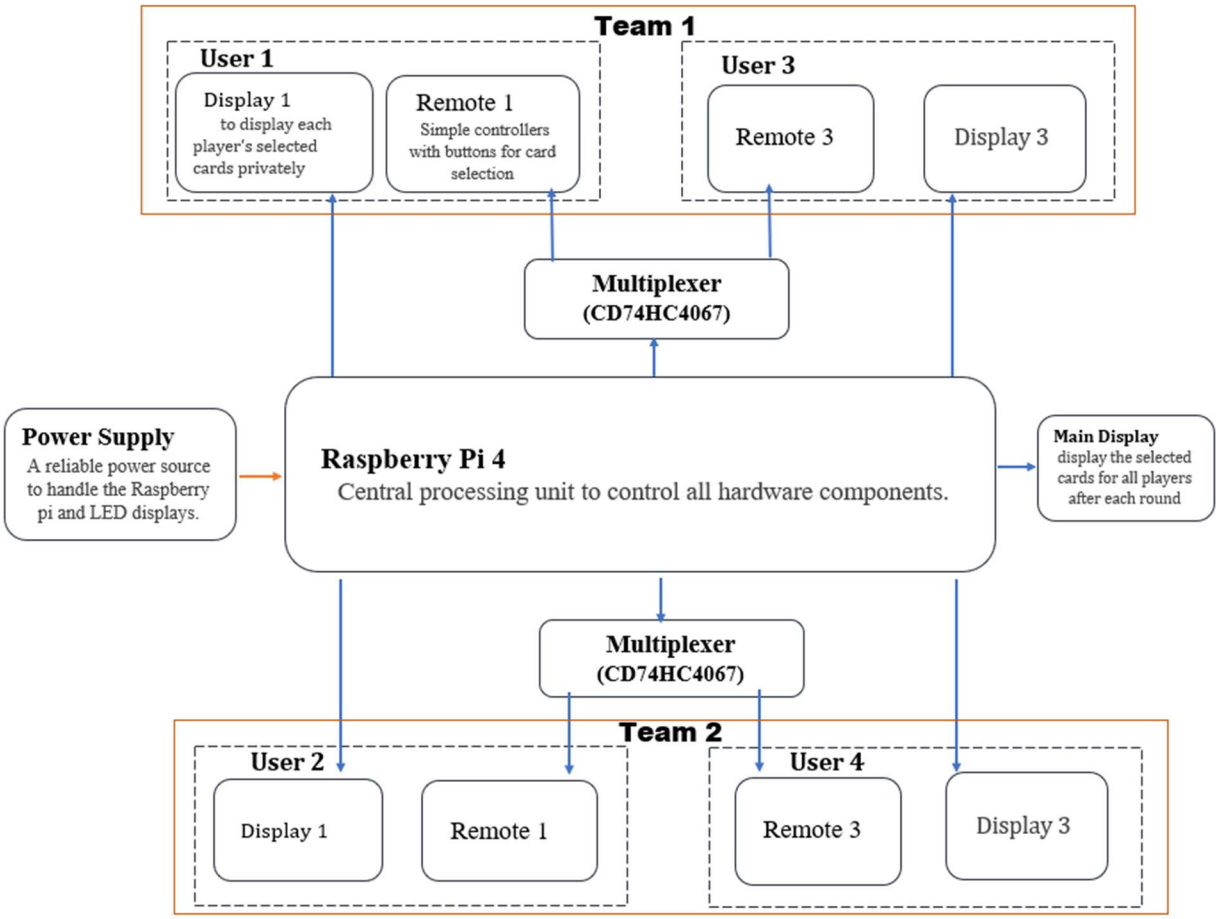
Resistors

Tactile Push Button

Jumper Wires

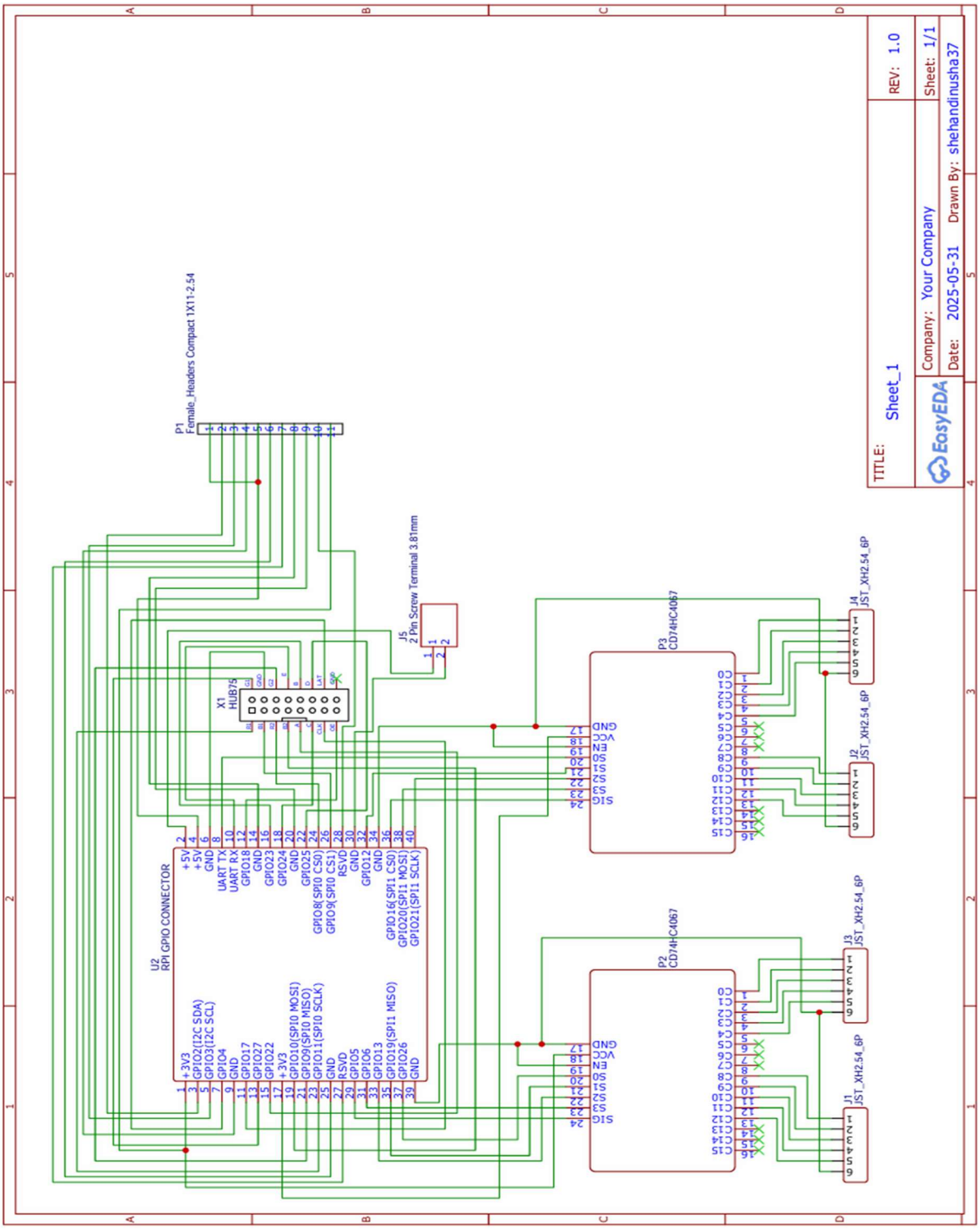
4 Analysis & Design

4.1 Block Diagram

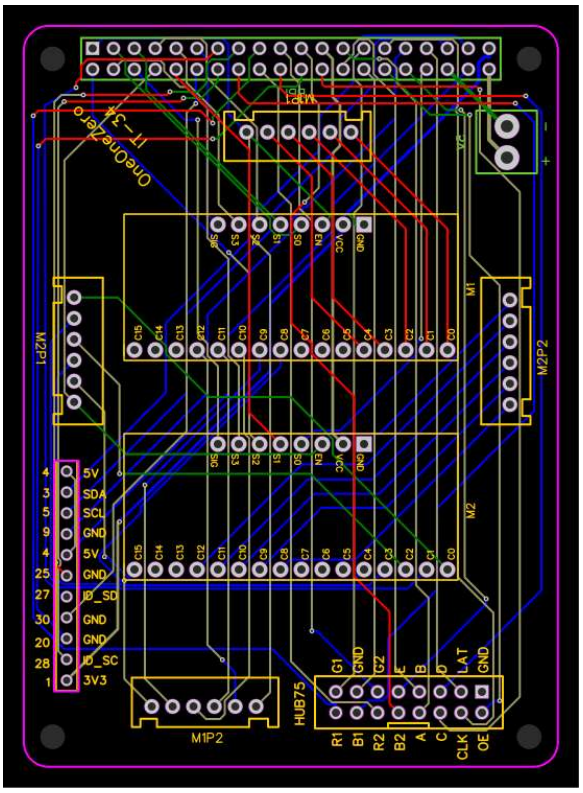


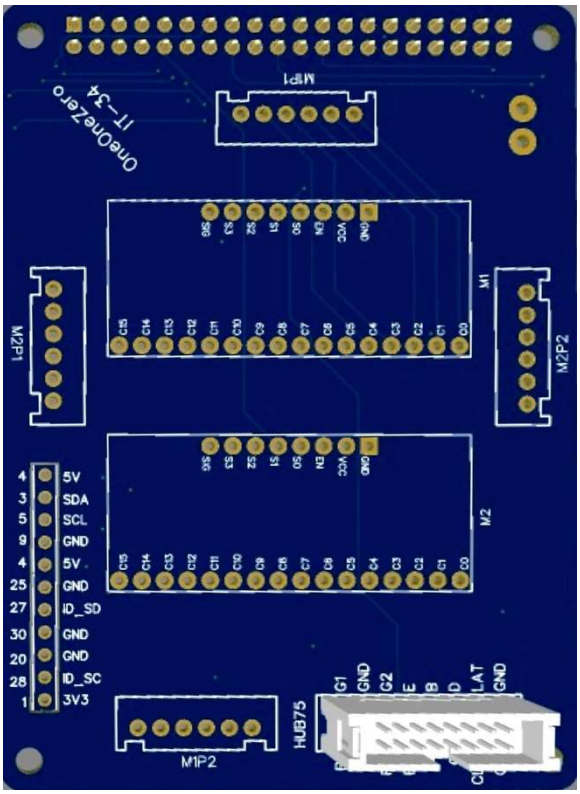
4.2 Solution Design

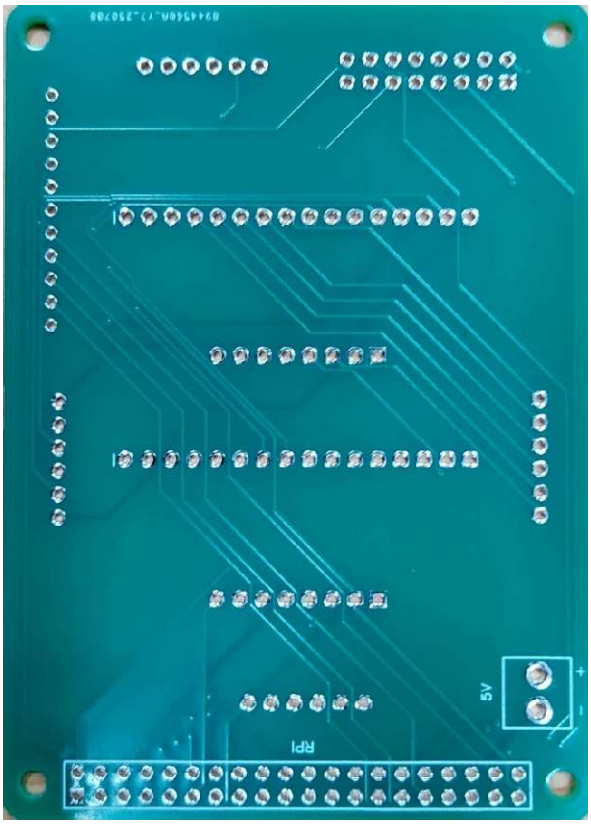
4.2.1 LED panel configuration

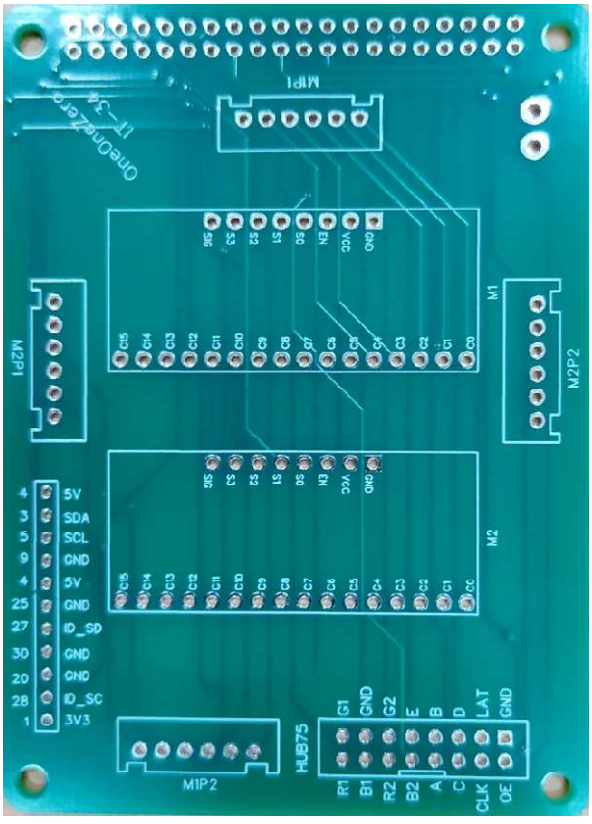


Circuit Diagram


OneOneZero
IT-34
M1P1
M2P1
M2P2
M1
M2
M1P2
HUB75
5V
SDA
SCL
GND
ID_SD
ID_SC
3V3
G1
G2
LAT
CLK
OE

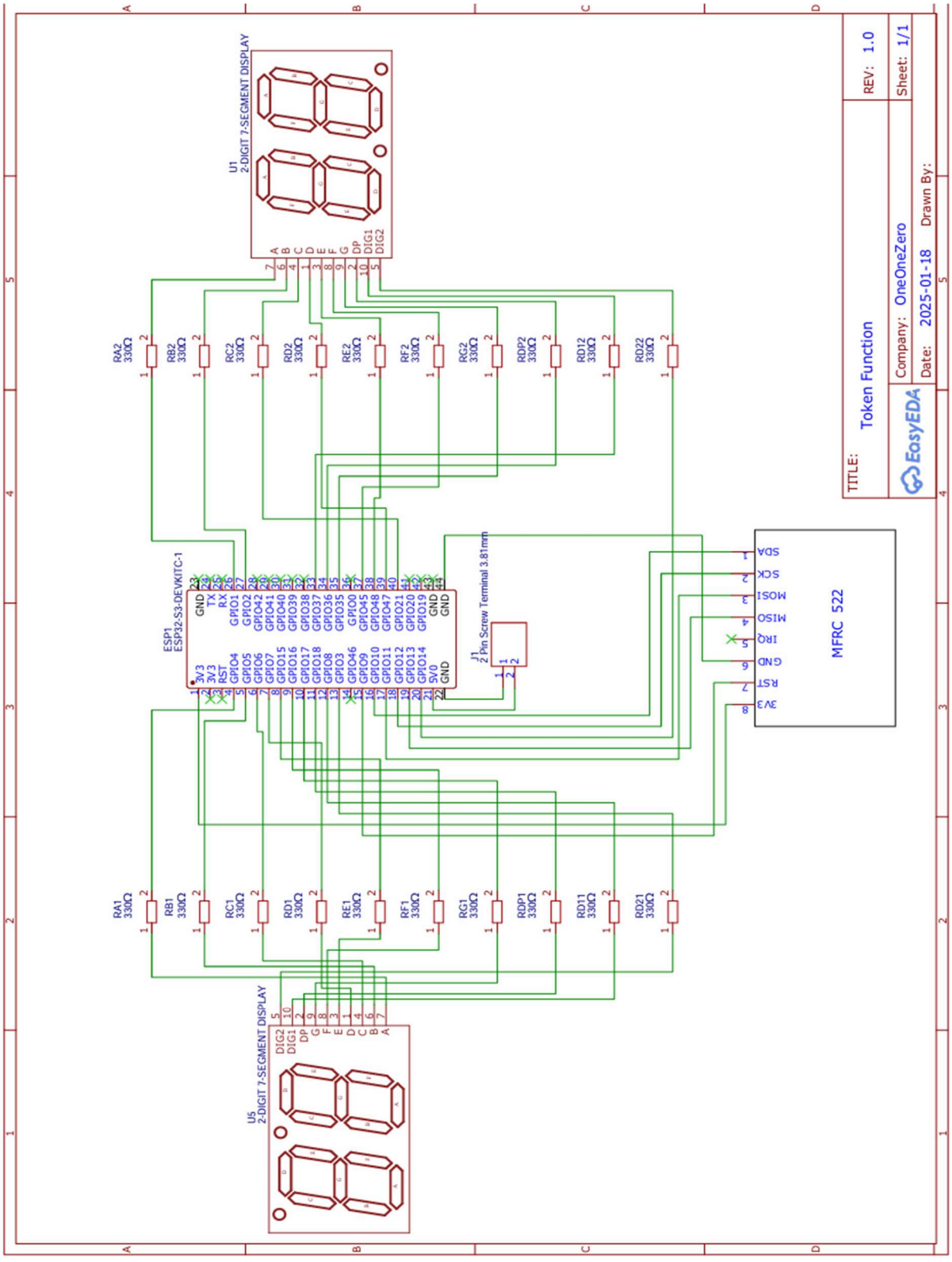

OneOneZero
IT-34
M1P1
M2P1
M2P2
M1
M2
M1P2
HUB75
5V
SDA
SCL
GND
ID_SD
ID_SC
3V3
G1
G2
LAT


RPI
5V

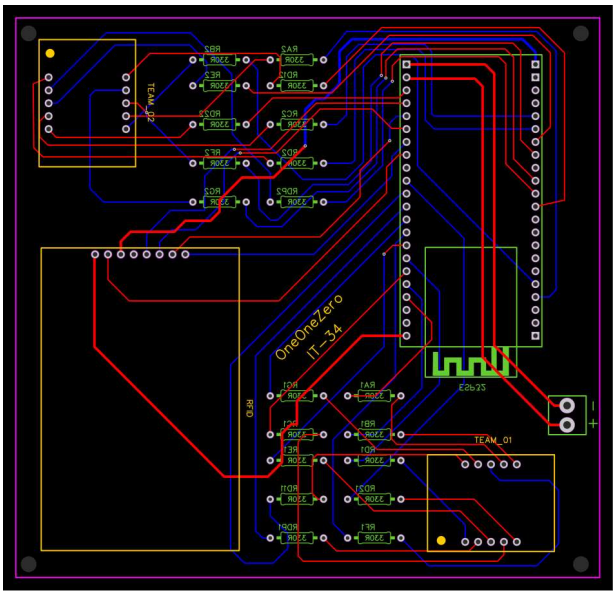

OneOneZero
IT-34
M1P1
M2P1
M2P2
M1
M2
M1P2
HUB75
5V
SDA
SCL
GND
ID_SD
ID_SC
3V3
G1
G2
LAT

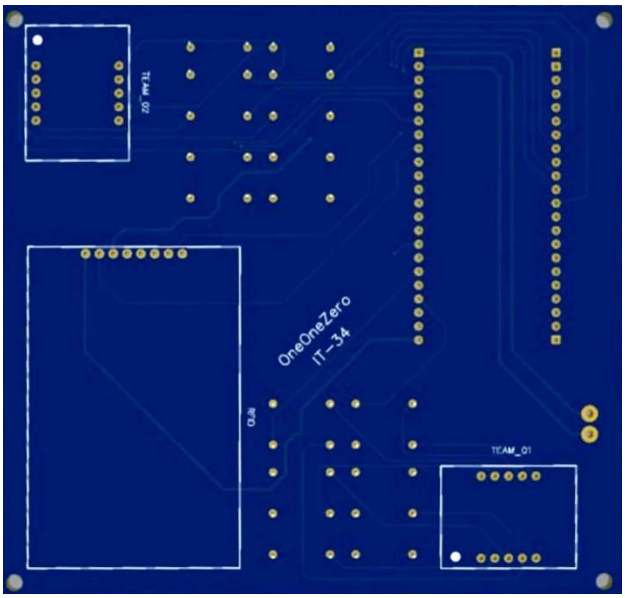
PCB Design

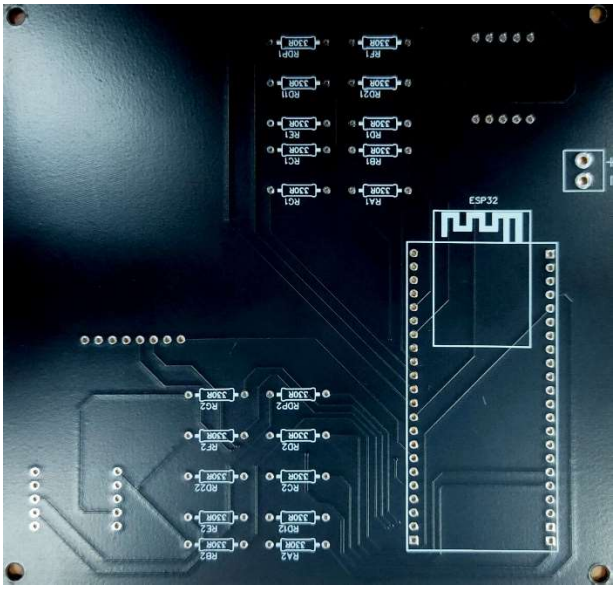
4.2.2 RFID Token System

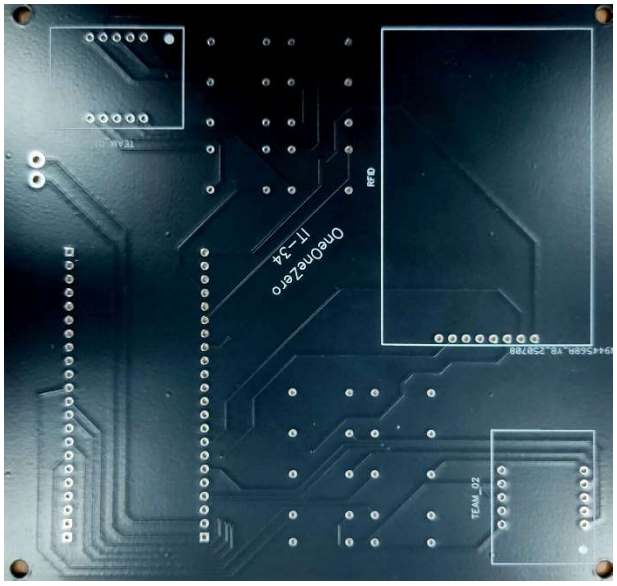


Circuit Diagram



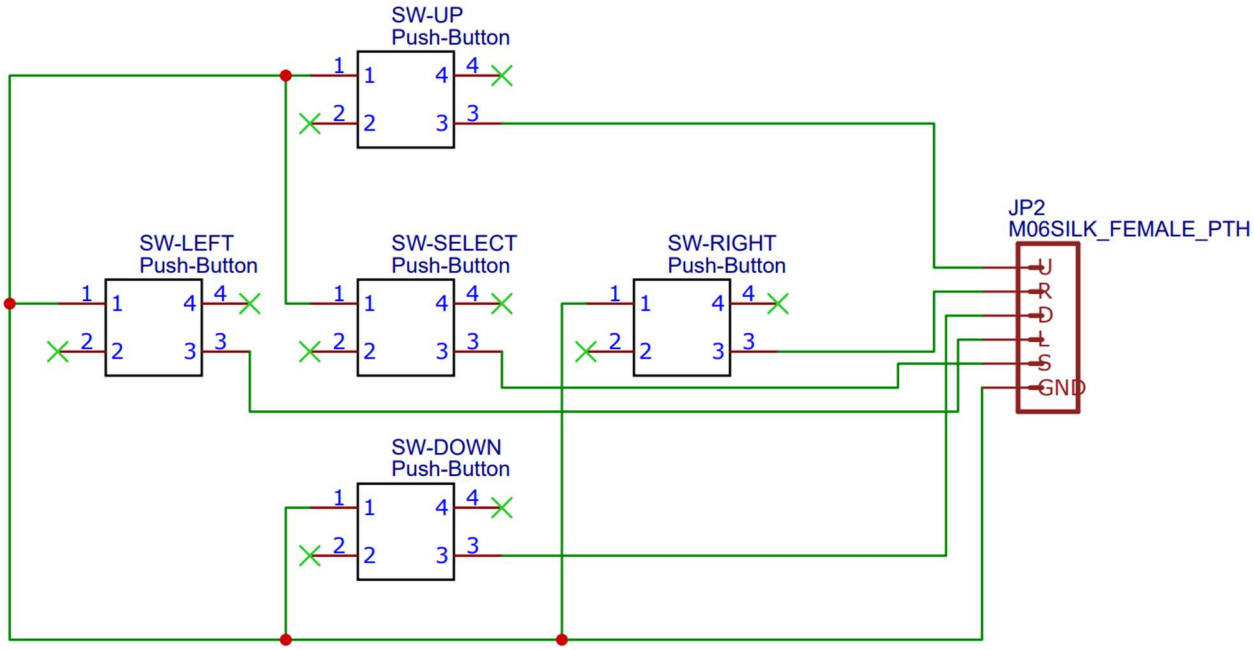




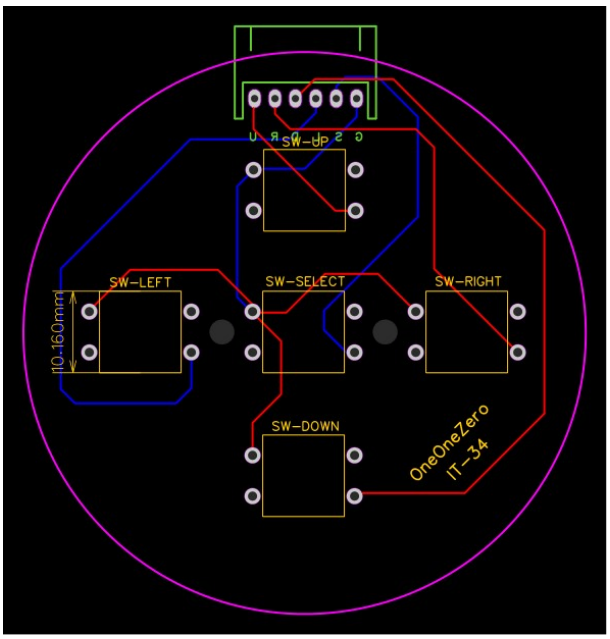


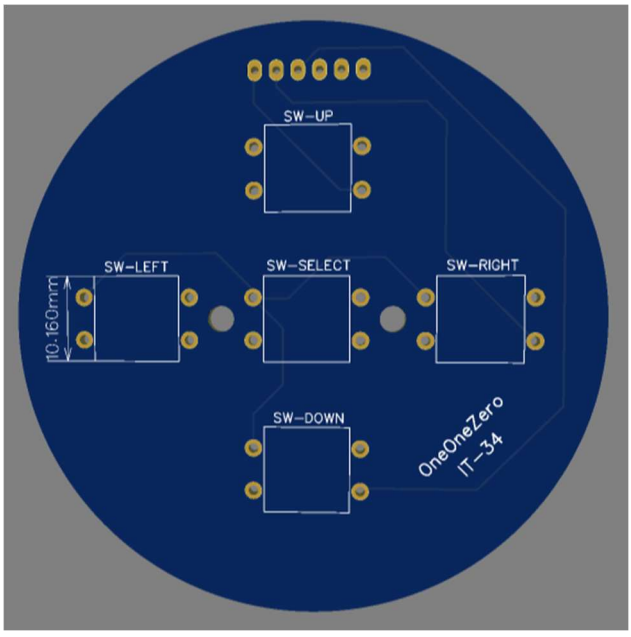
PCB Design

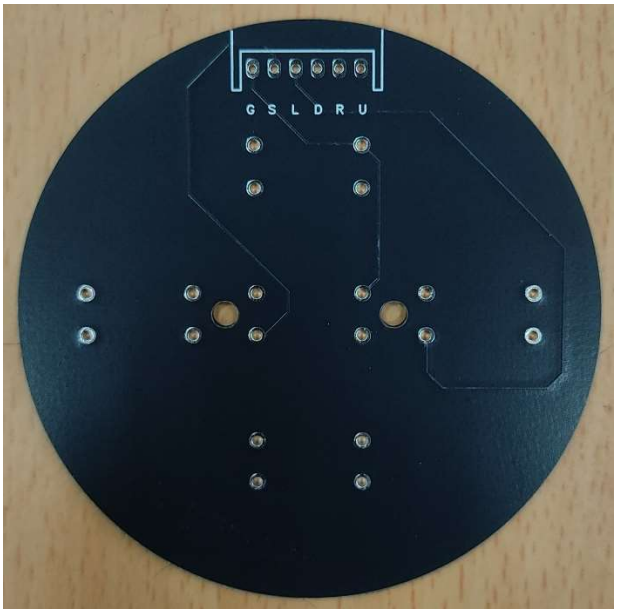
4.2.3 Remote Configuration

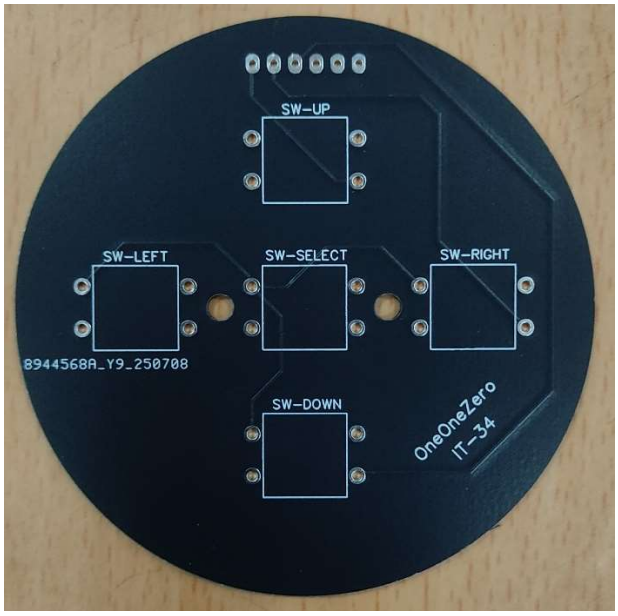


Circuit Diagram






G S L D R U


SW-UP
SW-LEFT
SW-SELECT
SW-RIGHT
8944568A_Y9_250708
SW-DOWN
OneOneZero
IT-34

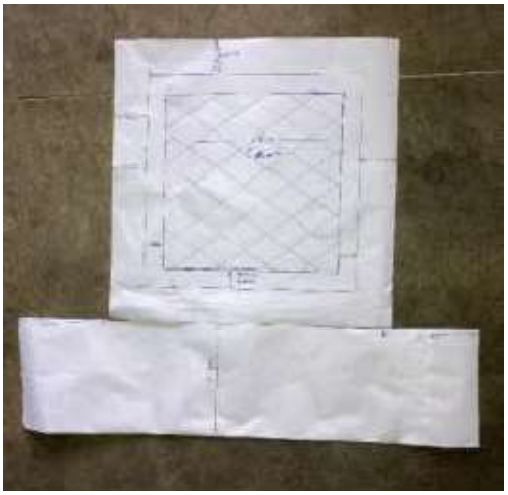
PCB Design

4.3 End Product

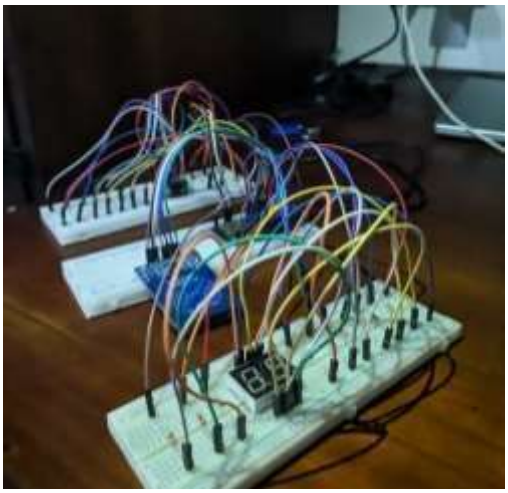
5 Resources

Component	Quantity	Price
Raspberry Pi 4 Model B	1	15900.00
64 x 64 RGB LED Matrix Panel	5	22500.00
ESP 32 S3	1	1200.00
Accessories For Wired Remote	4	6000.00
2 Digit seven segment	2	70.00
5V 40A SMPS Power Supply	1	2850.00
RFID Sensor	1	450.00
Multiplexer (CD74HC4067)	2	400.00
PCB Design		10500.00
3D Wooden Frame	1	8000.00
Total		67870.00

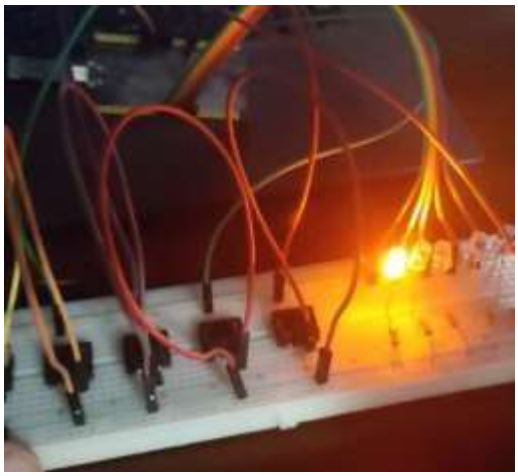
6 Testing and Implementation



- Drawing the sketch for the model.



- Complete the token function.

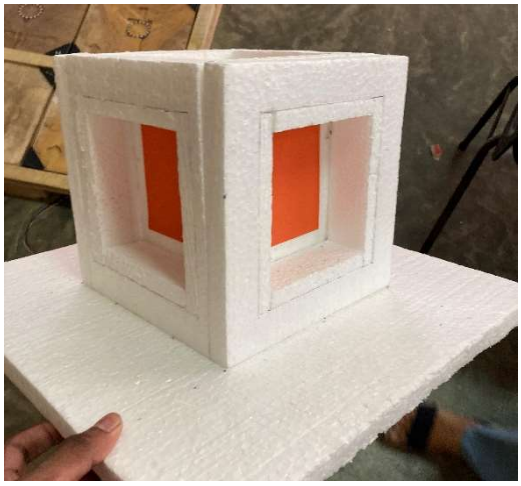


- Testing remote buttons by lighting some LEDs

- Soldering the RFID sensor pins.

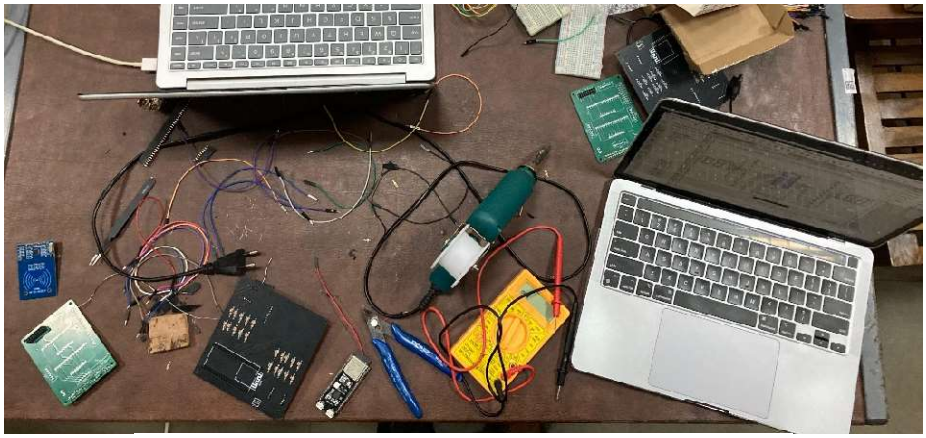
- Displaying the cards through LED

- Installing the OS. panels.

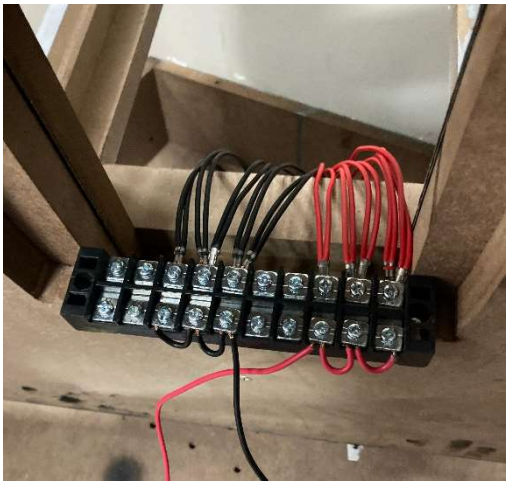


- Building a Prototype for the Model

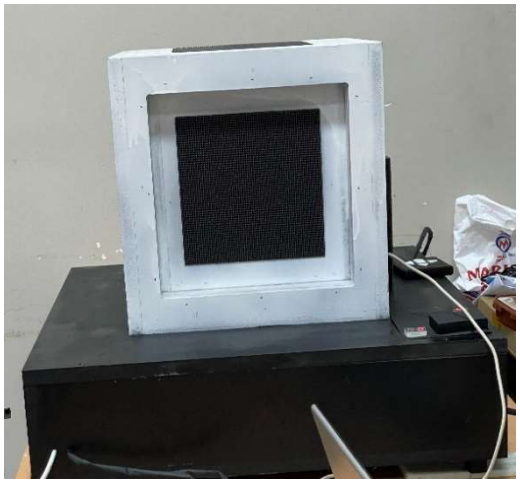
- Building the Model



- Testing and Soldering PCBs



- Wiring the LED panels



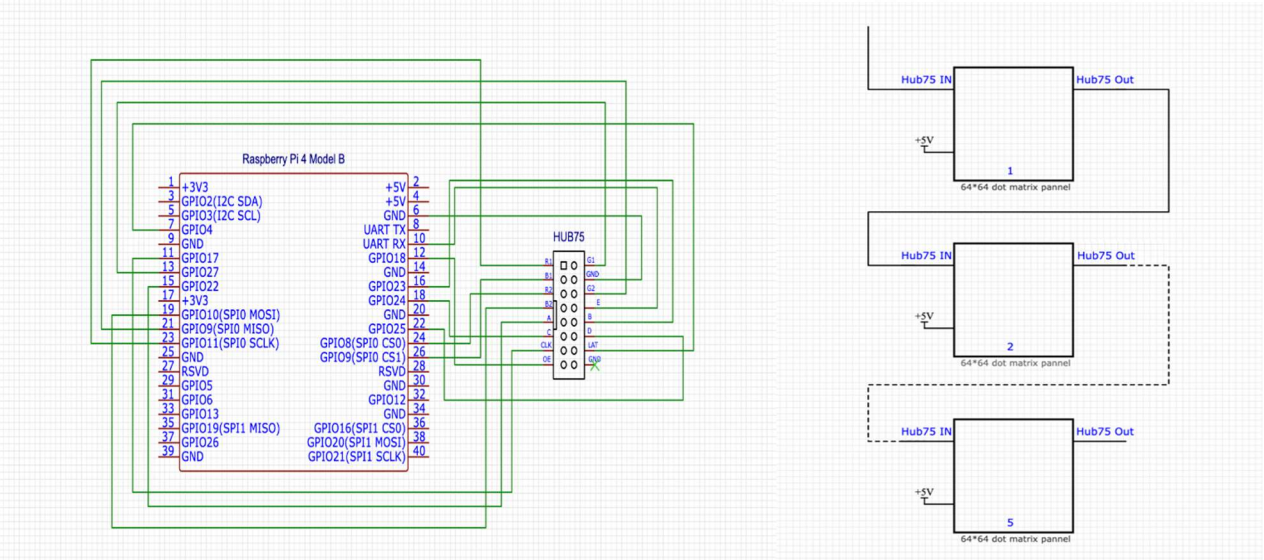
- Assembling and Testing

7 Individual Contribution

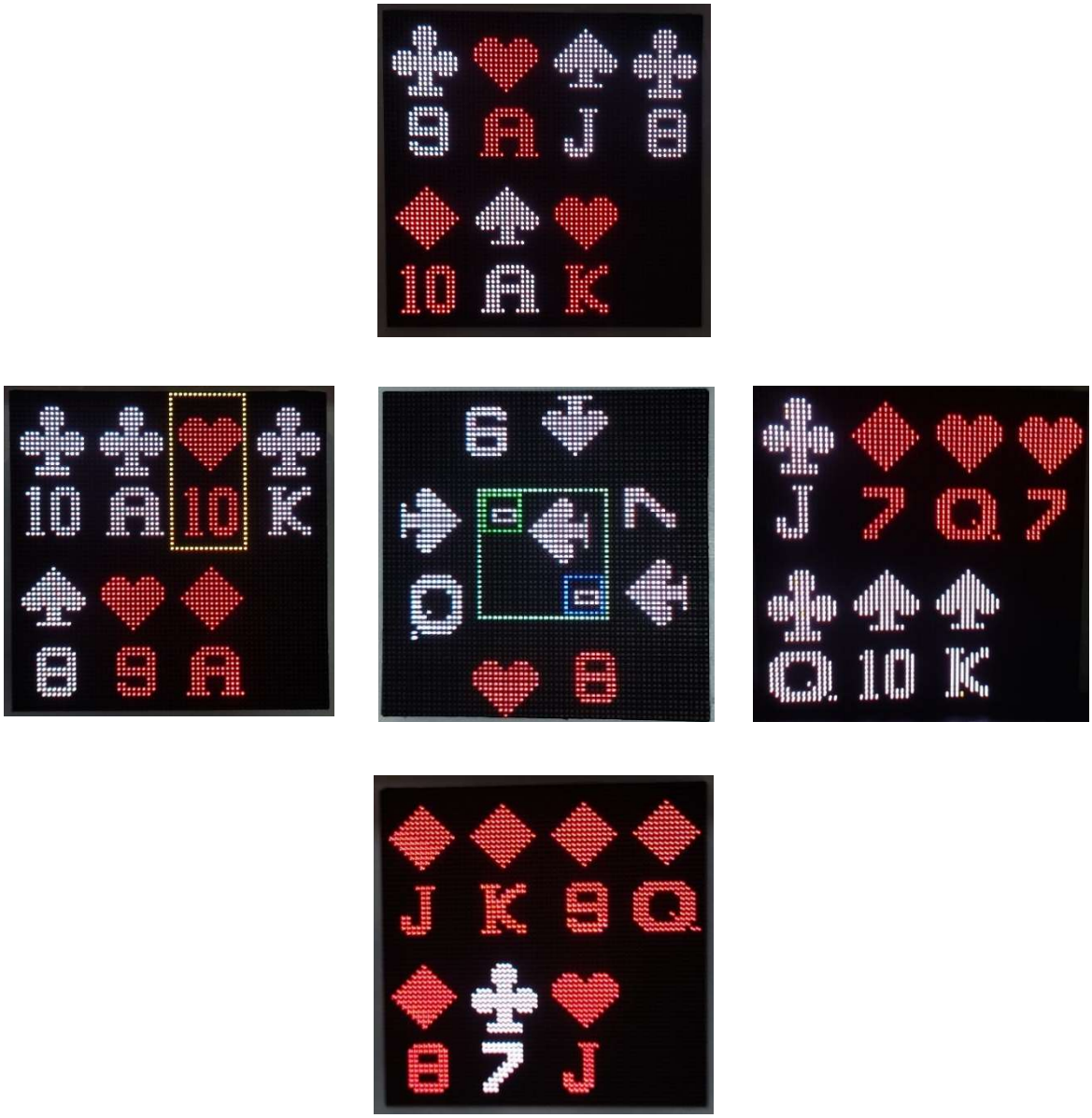
234052X - H.B.C. Dhananjaya

During this project, I took part in setting up the Raspberry Pi 4 board to establish the foundation of the entire system. This device serves as the central processing unit, overseeing the game logic and facilitating connections with all the hardware components (panels, remotes, token function). My primary responsibility was to configure the Raspberry Pi with the dot matrix panels and create basic game logic. The game wants substantial processing power to manage all computations autonomously. The processing demands of the game had to be handled by the Pi itself to maintain a stable environment and responsive enough to handle user interactions. To prepare the Raspberry Pi, I installed the latest version of the Raspbian operating system. This is the software system that helps the Raspberry Pi run programs efficiently and without problems.

I connected the 64x64 RGB LED Matrix Panel (3mm pitch) panels in a linear arrangement using IDC cables through the hub 75 ports. It can control individual lights by addressing their row and column in software through the rpi-rgb-led-matrix library. The hardware uses multiplexing with row selection lines and serially shifted data lines to light up specific LEDs in a fast cycle.During the panel setup process, I implemented code to avoid display flickering. To ensure optimal functionality, I utilized specialized libraries to configure the display cards. I defined bitmaps for the card symbols, letters, and numbers. Furthermore, I divided one panel into eight sections to display 8 cards for each player. In main panel I divide sections and display scores and selected cards.



In parallel, I had to develop the Basic Omi gaming concept. I used python to develop game logic. This involved the development of fundamental mechanics and the formulation of the overarching game rules. **When developing fundamental game mechanics I** included how the game progresses through rounds (called tricks), how players select cards, and how these selections affect the game state. The primary panel displays the triumph, selected cards and number of winning rounds for each team. This aimed to enhance the game's entertainment value and provide a more dynamic and interactive user experience.



Panel configuration and basic game logic running on 5 panels

The basic game's rules are as follows:

- Normal Game (8 tricks):
- Trump team wins 5+ tricks: Trump team gets 1 token
- Non-trump team wins 5+ tricks: Non-trump team gets 2 tokens
- All 8 tricks (Kapothi): Winner gets 3 tokens
- 4-4 tie: No tokens awarded

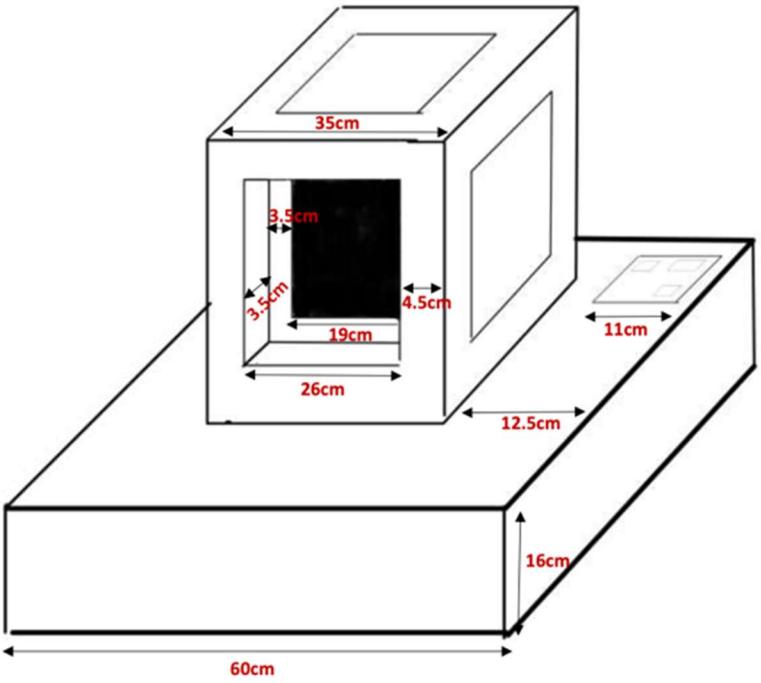
The Raspberry Pi updates the LED matrix panels and token function to show selected cards, the winning team in each round, current scores, and tokens earned, providing a dynamic and interactive user interface. Since the Raspberry Pi handles all this processing internally, the game logic had to be efficient and responsive enough to manage user inputs and update the display in real time without delay or instability.

To power the dot matrix panels and token function part, I opted for a 5V 40A Power Supply. One 64x64 RGB LED Matrix Panel (3mm pitch) use 5V and 4A. there are 5 panels used in this project. This power supply was the most suitable choice, as it provided sufficient power to all the components within the system, including the LED matrix panels and token function. I tested the whole setup to make sure it works without problems. I ensured the SMPS gave enough backup power so that even on maximum load conditions, the system works perfectly. This setup was very crucial to ensure the overall system's reliability and efficiency for which no power-related problems should reduce the gaming experience.

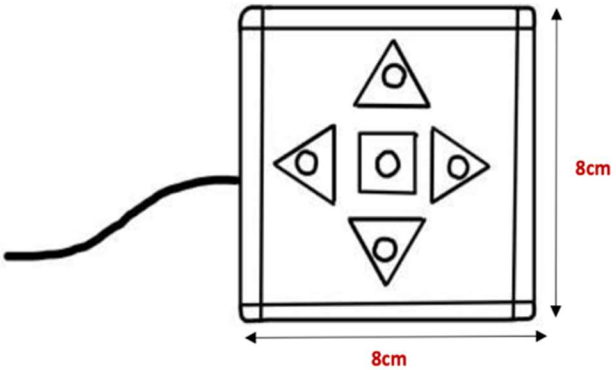
I also led to create suitable structure design for implement the LED matrix Pannels. The structure is like a box that keeps the panels, Raspberry Pi, and power supply safe and firmly in place. I used 3D design tools to create this structure. This helped us plan exactly how all parts fit together before building it. The final design is strong and neat, making sure the game hardware doesn't get damaged during use and that it looks good too. There are 4 displays for 4 users.in this case each player cannot see others displays.

Everyone can see main display.so I think and create suitable design for our final product. Remote design is unique design, and it can easily handle with one hand.

Structure Design

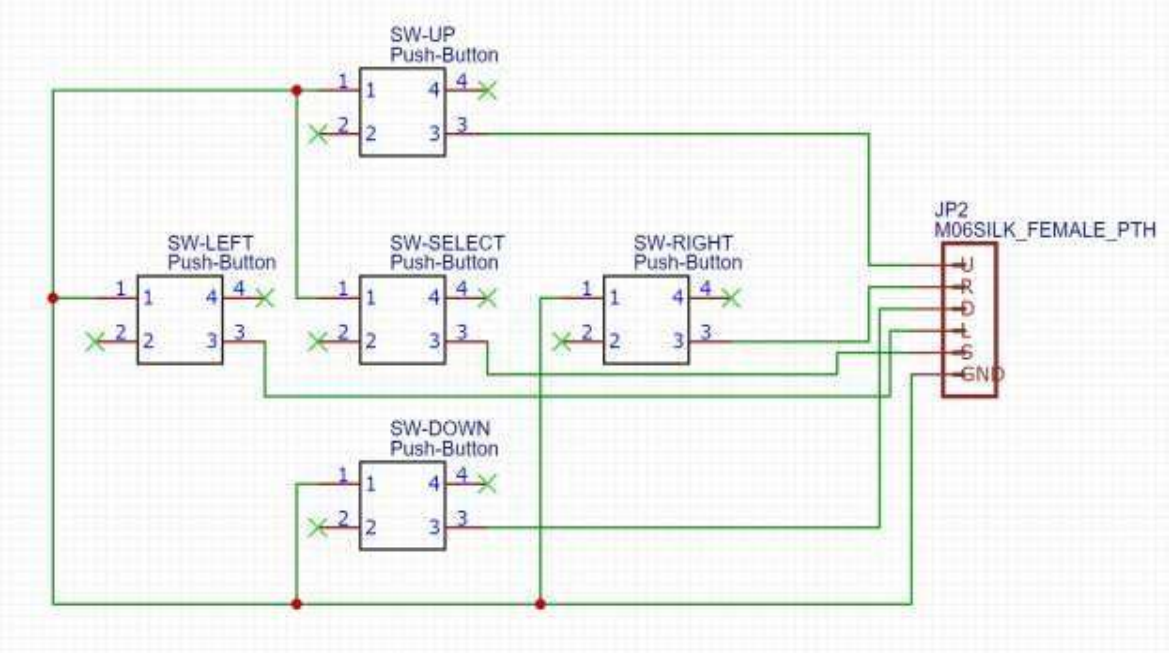


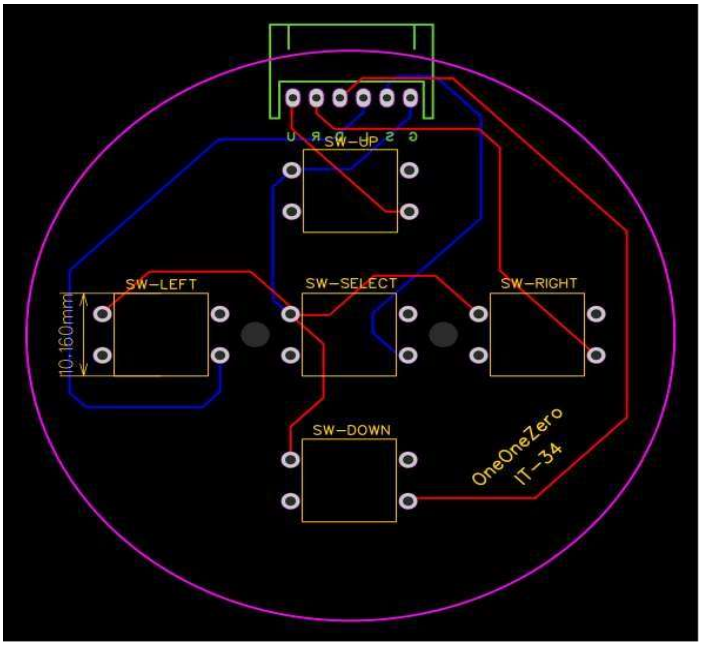
Remote Design



234056M - P.D.M. Dilshari

In this project, my primary responsibility was the design and implementation of the remote circuit and its printed circuit board (PCB), which functioned as the direct interface between the players and the game logic of *Omi the Trumps*. The circuit schematic and PCB layout were developed using the EasyEDA design platform for four individual remotes, each equipped with five tactile push buttons corresponding to the directional inputs (Up, Down, Left, Right) and a selection input (Enter).





These buttons were assigned to the channels of two CD74HC4067 16-channel analog/digital multiplexers, enabling the scanning of all 20 input signals using a minimal number of Raspberry Pi general-purpose input/output (GPIO) pins—specifically, four control lines (S0–S3) and one signal line (Z) per multiplexer, with the enable (EN) pins permanently grounded to ensure continuous operation.

The circuit employed pull-up logic, utilizing the Raspberry Pi's internal pull-up resistors so that an unpressed button produced a HIGH signal and a pressed button produced a LOW signal, thereby improving noise immunity and simplifying the wiring process.

In the PCB layout, signal traces were kept short and physically separated from high-current power lines to minimize electromagnetic interference, while a continuous ground plane was incorporated to enhance signal integrity. Additionally, connection headers were positioned at the board edges to facilitate efficient system integration.

Following fabrication, all components, including push buttons, IC sockets, and connectors, were soldered onto the PCB, and each channel was tested using GPIO monitoring scripts to ensure precise input detection, effective debouncing, and overall responsiveness.

Troubleshooting procedures included conducting continuity checks, remapping input channels, and refining trace routing for improved performance. The completed design is presented in the Remote Configuration Circuit Diagram and the Remote PCB Layout , clearly illustrating the integration of multiplexers, button arrays, and Raspberry Pi connections that were developed to deliver low-latency, reliable, and durable input handling for smooth gameplay.

234223A - S.D. Weerasinghe

My primary responsibility in this project was **designing and implementing the remote control input system using multiplexers (CD74HC4067) and Raspberry Pi GPIO pins**. This module reads multiple button inputs from different player remotes efficiently and reliably by managing hardware pin control, signal reading, and software debouncing.

Hardware Components

- CD74HC4067 Multiplexer: Two 16-channel multiplexers (M1 and M2) are used to reduce the number of GPIO pins required to read buttons from multiple remotes. Each multiplexer selects one of its 16 inputs to route to a single signal output pin for the Raspberry Pi.
- Raspberry Pi GPIO: Uses 5 GPIO pins per multiplexer to control select lines (S0–S3) and receive the button press signal (SIG).
- Remotes: Two remotes per multiplexer, each with buttons mapped to specific multiplexer channels.

How It Works (Simplified Explanation)

- Each multiplexer has 16 "channels" corresponding to potential inputs.
- The Raspberry Pi sets the multiplexer’s four select pins (S0–S3) to binary values that choose which button (channel) to "listen" to.
- The multiplexer outputs the state of the selected button on the SIG pin.
- Because buttons use an inner pull-up resistor, the SIG pin is LOW (0) when pressed and HIGH (1) when not pressed.
- This approach lets the system read many buttons using fewer GPIO pins by cycling through each button channel rapidly.

Key Functions in Code

- setup_gpio: Initializes GPIO pins for multiplexers as inputs or outputs.
- select_channel: Converts a button number (0-15) to a binary signal to control multiplexer select pins.
- read_button: Reads the state of a button by selecting the channel and checking the SIG pin.
- get_player_input: For a given player, scans their remote's buttons, applies a debounce filter to handle button "bounce" noise, and returns the first detected pressed button or None if no button is pressed.
- cleanup: Resets GPIO pins on program exit.

Debouncing

To prevent registering a single button press multiple times due to mechanical bounce, the code ignores repeated presses on the same button within 0.2 seconds after the initial press.

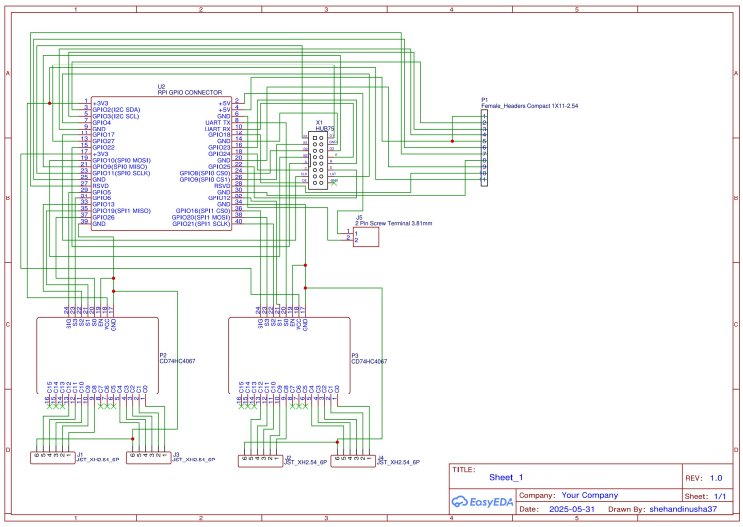
Why CD74HC4067 Multiplexers Are Used Instead of I2C Expanders

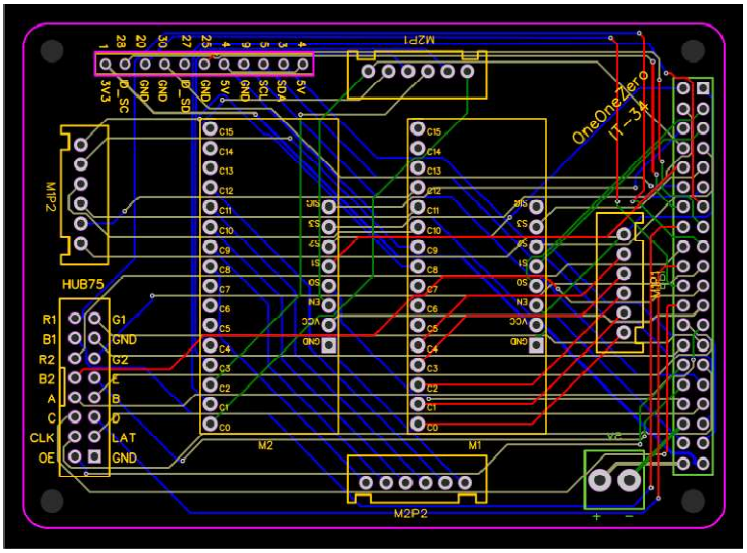
During interim evaluations, we proposed using an I2C-based 16-bit I/O expander (such as the MCP23016) but chose the CD74HC4067 multiplexer for the following reasons:

- Only 5 GPIO pins per multiplexer are needed versus requiring an I2C communication setup.
- Supports both digital and analog signals.
- Simpler hardware setup, no I2C protocol overhead or libraries.
- Cheaper and easier to manage with direct GPIO control on the Raspberry Pi.
- Sequential channel reading is sufficient for this project's button inputs.

And also I specifically **designed a four-layer PCB to efficiently connect the Raspberry Pi, multiplexers, and four player remotes**. Due to the tight space constraints—since the board needs to fit into a compact enclosure—using four layers allowed for more straightforward and reliable signal routing without tangled or excessively long traces.

By opting for a four-layer design, I was able to keep the board dimensions small while maintaining clean, robust connections between all critical components. This approach also makes assembly easier and provides better signal integrity compared to a two-layer board, especially when dealing with the multiple remote inputs and control signals required for the multiplexers.





In addition to the hardware and PCB design, I also **implemented key software features** to enhance gameplay clarity and interactivity:

- **Previous Hand Display Logic:**

  The core code functions included:

  - display_last_trick_cards_corrected():
    This function displays the cards from the most recently completed trick on the game's main LED panel so that all players can review what was played. It divides the screen into four quadrants, one for each player, and in each quadrant shows the player label (P1–P4), the suit symbol, and the rank of the card they played. A border and cross-lines visually separate the quadrants. The display stays visible for a set duration (about 6 seconds), allowing everyone to clearly see the previous trick before play continues.

- **Game Animations:**
  I designed and programmed the animation sequences that make gameplay smoother and visually appealing. These include card movement, reveals, and highlight effects that help players intuitively follow game actions.

  Main code functions for animations:

  - display_trick_winner_animation(winner_player, winning_team, duration)

    Handles the animated display when a specific player (e.g., Player 1 in Team A) wins a trick.Shows pulsating team colors, celebration effects, and winner text.

  - new_round_animation_enhanced()

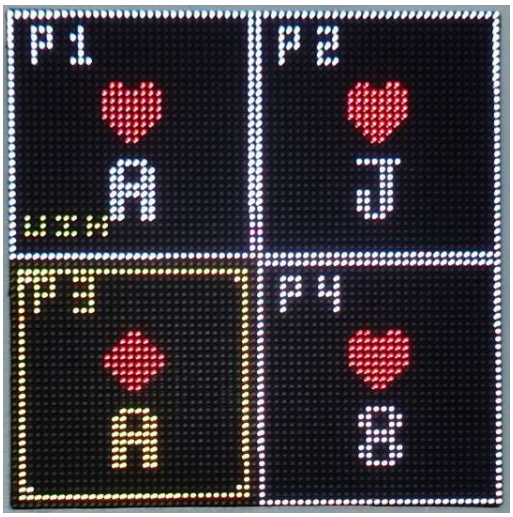
    Plays a multi-phase animated transition showing *"NEW ROUND BEGINS"* with effects and round number display

  - show_spectacular_game_over_animation(losing_team, winning_team)

    Creates a multi-stage "grand finale" animation with fireworks, patterns, and pulsing victory text for the winning team.

- Triggered when check_game_over_condition() detects that a team's token count has reached zero.
- Ends with display_play_again_prompt() if players wish to restart.

These functions were carefully integrated into the game loop, ensuring that both the previous hand display and game animations are triggered exactly at the right times for clear, engaging gameplay. This work ensures not only accuracy in game state representation but also an enjoyable user experience.



234118G - M.H.N. Kumarasiri

I designed and implemented the token function display system using two-digit common-cathode seven-segment displays. This system clearly showed the token counts for both Team A and Team B in real time, allowing players, referees, and spectators to easily track the match progress. It improved game transparency and contributed to a more engaging and professional presentation.

My work began with the design of the display circuit. I ensured that each two-digit common-cathode display could be connected properly to the ESP32-S3 microcontroller. This process involved mapping the correct GPIO pins for each display segment and digit control line, as well as selecting appropriate current-limiting resistors to protect the LED segments from excessive current. I also took into account the overall power requirements to ensure compatibility with the 5V supply from the SMPS.

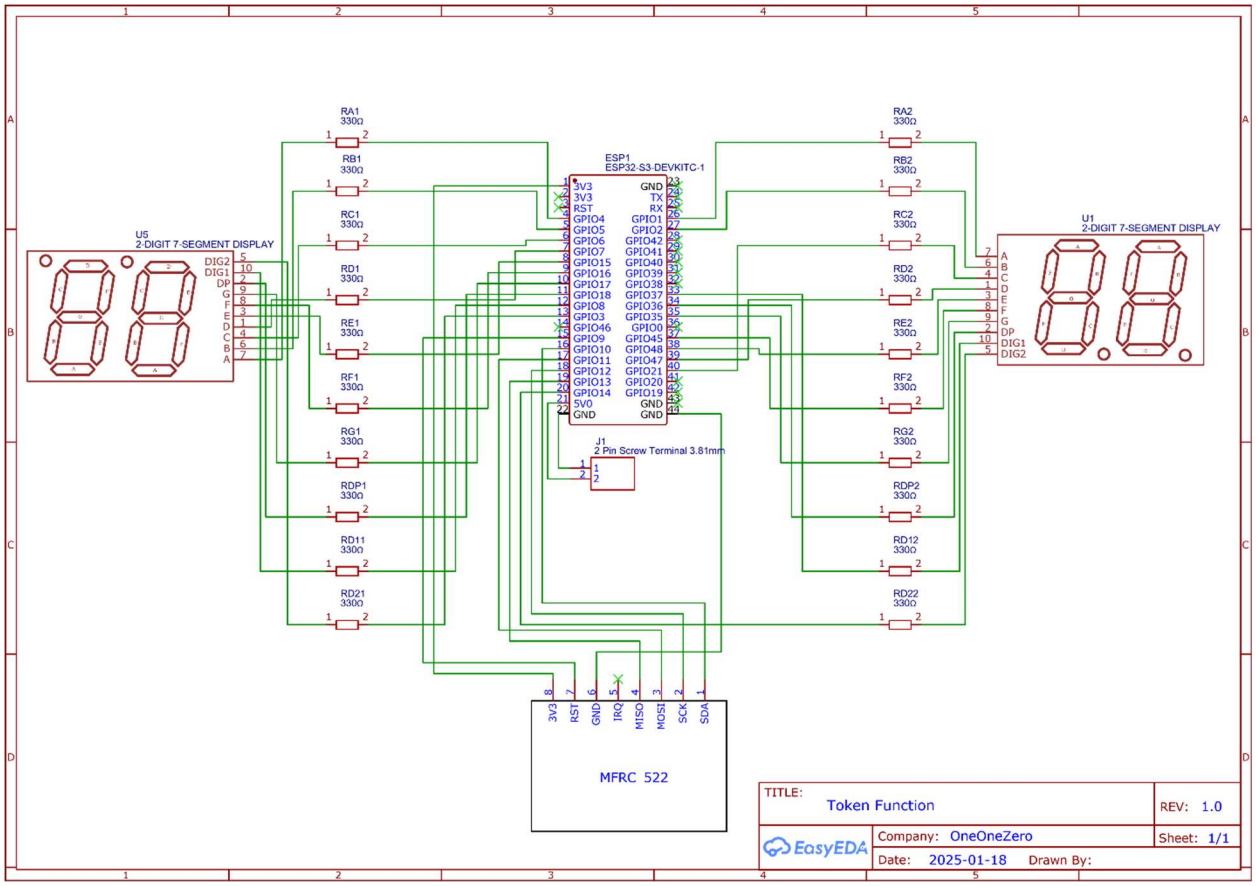
	ESP 32 pin for	
Segment Pin	**Segment 1**	**Segment 2**
a	4	1
b	5	2
c	6	21
d	7	47
e	15	48
f	16	45
g	17	35
dp	18	36
Digit 1 (DG1)	8	37
Digit 2 (DG2)	3	14

ESP32-S3 Pin Mapping for Two-Digit Common-Cathode Seven-Segment Display

After finalizing the design, I moved on to hardware assembly. This stage included soldering the displays, resistors, and connection wires according to the printed circuit board (PCB) layout. I verified that the common cathode pins were connected securely to the ground lines and that the power lines from the SMPS provided a steady and reliable 5V supply. To ensure correctness, I tested the displays in stages: first by lighting individual segments to confirm wiring accuracy, then by enabling each digit in sequence to confirm that the multiplexing worked as intended. Through this careful testing process, I ensured that there were no dead segments, incorrect connections, or visible flicker during normal operation.

The final stage of my work involved integrating the display system into the complete game hardware. I collaborated with the team members responsible for the token logic to ensure that the displayed values updated instantly whenever a token was added or removed. This integration was critical in maintaining accuracy, as the token counts had to reflect the actual game state without delays. Once integrated, I performed multiple game simulations to confirm that the display updated consistently under different game scenarios.

Through this process, my contribution ensured that the token function display system was reliable, easy to read, and seamlessly integrated into the project. The use of a well-designed hardware circuit, proper assembly techniques, and effective integration made the system both functional and visually professional, significantly improving the user experience for everyone interacting with the game system.

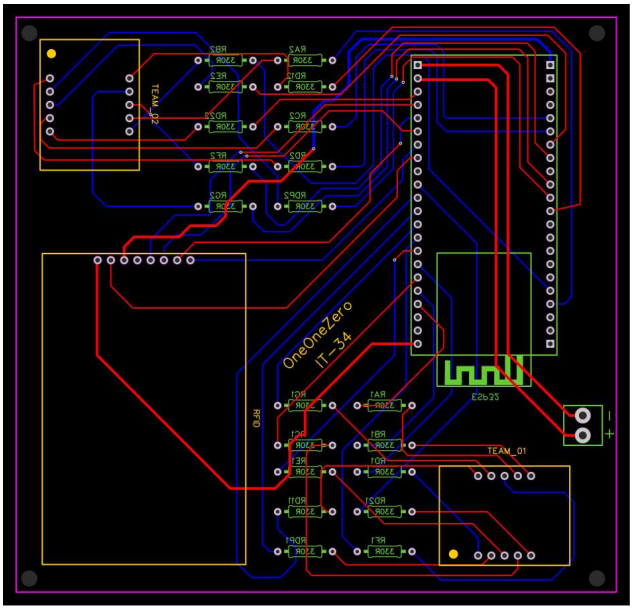


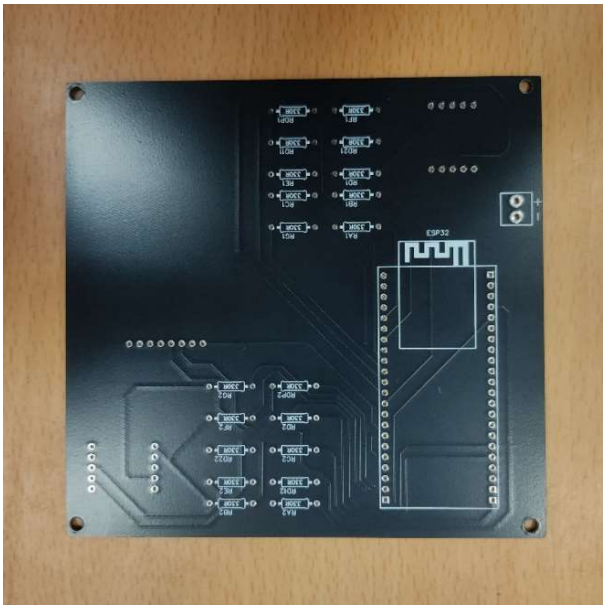
RFID Token System Circuit Diagram

PCB Design and Hardware Integration

I took full ownership of designing the custom PCB layout that houses the ESP32, RFID module, seven-segment displays, and supporting circuitry. This was particularly challenging because I had to ensure proper signal routing while maintaining compact dimensions to fit within our game setup

The PCB design required careful consideration of power distribution, with separate power lines for the displays and logic circuits. I used wider traces (0.5mm) for power lines and standard traces (0.25mm) for data signals to minimize voltage drop and ensure stable operation. The layout also incorporated proper grounding techniques to reduce electromagnetic interference, which was crucial for reliable RFID operation.





234235L - W.M.D.N. Wijerathne

I developed the complete hardware–software bridge for the RFID Token System, designing a custom ESP32 based PCB with an RFID scanner and seven-segment displays. I implemented token deduction logic via the MFRC522, added scan delay and card recognition, and built a TCP-based WiFi link to the Raspberry Pi for real-time game sync. I also integrated key game logic half/full court modes, cancel trump, and previous trick display ensuring smooth coordination between physical scanning and automated gameplay.

1. RFID Token Management System Design and Implementation

My primary responsibility centered around designing and implementing the complete RFID-based token management system that forms the core of our card game automation. This involved extensive work with the MFRC522 RFID module integrated with the ESP32 microcontroller to create a reliable card scanning and token deduction system.

I configured the RFID reader using SPI communication protocol, with careful attention to pin assignments (SS_PIN 10, RST_PIN 9, SCK_PIN 12, MISO_PIN 13, MOSI_PIN 11) to ensure stable card detection. I implemented a 2-second delay between scans to avoid accidental multiple deductions when players held their cards too long on the reader.

The system successfully differentiates between Team A and Team B cards using hardcoded UID recognition. When a valid card is scanned, the system immediately deducts one token from the appropriate team's count and updates the seven-segment displays in real-time. I also added comprehensive error handling for scenarios like insufficient tokens, wrong team cards, and system malfunctions.

2. WiFi Communication Bridge Between ESP32 and Raspberry Pi

One of my major contributions was establishing robust WiFi communication between the ESP32 token management system and the Raspberry Pi running the main game logic. I developed a custom protocol that allows real-time synchronization of game state and token counts between both systems.

The ESP32 acts as a TCP server on port 8080, while the Raspberry Pi connects as a client using my custom ESP32WiFiManager class. I implemented automatic network discovery, connection management, and heartbeat monitoring to ensure the connection remains stable throughout extended gameplay sessions. The system can recover from temporary network interruptions and automatically reconnect when the connection is restored.

I designed the communication protocol to handle various message types including START_SCAN, SCAN_PROGRESS, SCAN_COMPLETE, and token count updates. Each message follows a structured format that ensures reliable data transmission and proper error handling on both ends.

3. Game Logic Integration and Special Game Modes

My work extended significantly into the game logic implementation, particularly for advanced game modes and features that make our system unique. I developed the logic for half court mode, full court mode, cancel trump functionality, and the previous trick display system.

For half court mode, I implemented the detection logic that identifies when a player press enter button, automatically triggering the half court option display. The system calculates which team gets defeated and determines the appropriate number of token scans required (2 cards for half court wins).

The full court mode implementation was more complex, requiring tracking of all 8 tricks and determining when a single player wins all tricks. I coded the logic that identifies the full court winner and triggers thess appropriate token scanning sequence (3 cards for full court victories).

I also implemented the cancel trump feature, which allows the trump selector to cancel their selection if all their first four cards are rank 10 or below. This required careful integration with the card dealing logic and user interface animations.

4. Token Display System and User Interface

The seven-segment display system was entirely my responsibility, from hardware configuration to software implementation. I configured two separate displays for Team A and Team B token counts, using common cathode configurations with appropriate current-limiting resistors.

The display refresh system was particularly challenging because I needed to achieve maximum brightness while maintaining stable digit multiplexing. I implemented an optimized refresh routine that cycles through both displays at high frequency (every 2ms) to eliminate flickering and ensure clear visibility under various lighting conditions.

I also integrated the display updates with the game state, so token counts automatically reflect the current game situation. The displays show real-time feedback during token scanning sequences, providing immediate visual confirmation to players.

5. System Integration and Testing

Throughout the project, I was responsible for integrating all my components with the rest of the team's work. This involved extensive testing of the communication protocols, debugging timing issues between the ESP32 and Raspberry Pi, and ensuring that the token management system remained synchronized with the game state.

I conducted comprehensive testing scenarios including network disconnections, rapid token scanning, simultaneous game mode switches, and edge cases like players running out of tokens mid-game. Each test revealed areas for improvement, leading to more robust error handling and better user experience.

6. Challenges and Solutions

The project presented several significant technical challenges that required innovative solutions. The most critical issue was ensuring reliable RFID card detection while preventing accidental double-scans. I solved this by implementing a time-based filtering system that ignores subsequent reads within 2 seconds of a successful scan.

WiFi connectivity proved problematic in environments with multiple competing networks. I addressed this by implementing automatic IP discovery, static IP fallback options, and comprehensive connection monitoring with automatic recovery procedures.

Power management was another major challenge, particularly ensuring stable operation of the seven-segment displays under varying load conditions. I resolved this by implementing proper power supply decoupling and optimizing the display refresh timing to minimize current spikes.

The integration between game logic and token management required careful synchronization to prevent race conditions and ensure consistent state between systems. I implemented a message queuing system with acknowledgment protocols to guarantee reliable state updates.

Through systematic debugging and iterative improvements, these challenges were successfully resolved, resulting in a stable and reliable token management system that enhances the overall gaming experience while maintaining the integrity of the traditional card game rules.

8 Software Implementation

The following are the three main codes in this project.

- *omi_the_trumps.py – Omi LED Game Controller (Main Game Logic)*

  Handles the full rules, flow, and graphics for the Omi card game on a multi-panel RGB LED matrix with remote controls.

  Implements animations, score tracking, special modes (Half/Full Court), and integrates with the token system.

  Communicates with the ESP32 over Wi-Fi to trigger and process token scanning events.

- *token_function.txt – ESP32 Token Manager (RFID + Display + TCP Server)*

  Runs on the ESP32 to manage token counts for both teams using RFID card scans.

  Drives two seven-segment displays to show live token counts with optimized brightness refresh.

  Hosts a TCP server to communicate with the Raspberry Pi game system, sending scan progress, token updates, and game-over alerts.

- *esp32_wifi_manager.py – Python ESP32 TCP Communication Client*

  Provides a robust TCP client for the Raspberry Pi to connect to the ESP32 Token Manager.

  Handles discovery, connection, reconnection, heartbeats, and parsing of incoming messages.

  Exposes methods for the game logic to start scans, reset tokens, request status, and receive scan/token updates in real time.

All the codes are in the following GitHub repository :

https://github.com/Shehan-Dinusha/Omi-The-Trumps.git

9 References

- Smith, J. (2020). Digitizing Traditional Games. Game Tech Journal, 34(2), 56-78.
- Raspberry Pi Foundation. (2023). Raspberry Pi Projects and Applications.
- Arduino Community. (2019). Applications of RFID in Gaming.
- TRONIC.LK
- https://www.adafruit.com/
- https://www.waveshare.com/wiki/RGB-Matrix-P2.5-64x64