EMBEDDED AND UBIQUITOUS SYSTEMS

ENGINEERING BOOK:

THE GAME

ESCOLA
POLITÈCNICA SUPERIOR
UNIVERSITAT DE LLEIDA

José Ramon Ariza Pérez

Mihaela Alexandra Buturuga

Eric López Seguí

Bachelor’s Degree in Computer Engineering

10th January 2025

University of Lleida

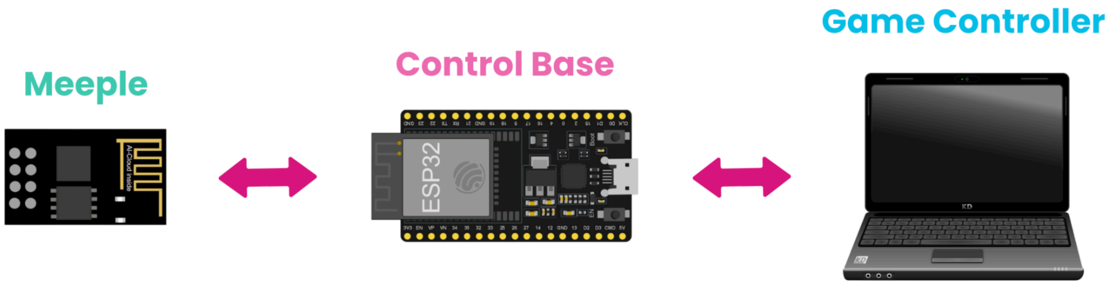
Course 2024 - 2025

Index

1. Introduction

The **e-Game Board** project aims to combine embedded systems, IoT communication, and creative game design to develop an interactive electronic board game. Each player controls a **meeple** that represents their position on the board and interacts with a **control base** displaying their game status and actions.

The game incorporates components such as ESP-01 and ESP32 microcontrollers, sensors, LEDs, and an LCDdisplay to handle player interactions. **Wifi** and **MQTT protocol** are used for communication between components.



The primary goal is to develop a functional game that includes different mechanics that allow us to work with the different components in harmony.

This document outlines the milestones, components, game rules, and the development process.

2. Project Milestones

This section details the chronological progression of the e-Game Board project, highlighting key accomplishments and challenges at each stage.

The e-Game Board project was carefully planned and executed in several stages, with each milestone contributing to the development of a fully functional and interactive game. This section outlines the specific goals, priorities, dependencies, and effort allocation for each milestone.

2.1. Initial Exploration and Component Familiarization

The project began with an initial exploration phase focused on familiarizing ourselves with the ESP32 and ESP01 microcontrollers. As this was our first experience with these components, we dedicated time to understanding their functionalities, pinouts, and basic programming.

We experimented with sample code to control individual components, including blinking LEDs, reading button inputs, generating buzzer sounds, and displaying messages on the LCD screen. Similarly, we tested the meeple's LED and Hall effect sensor to establish a foundation for later integration.

This hands-on approach proved crucial for building a solid understanding of the hardware and software building blocks.

Duration: 8 hours

Objective: Gain practical experience with the ESP32 and ESP01 microcontrollers and their peripherals.

Tasks:

- Experimented with basic code examples to control LEDs, buttons, the buzzer, and the LCD screen.
- Tested the meeple's LED and Hall effect sensor for position detection.

Dependencies: None; foundational milestone for all subsequent work.

Responsible member: All

Priority: High; established the groundwork for component integration.

2.2. Game Design and Rule Definition

The second milestone focused on defining the game itself. We had to brainstorm different game ideas and after settling on a “Mario Party” inspired idea, we worked on establishing the gameplay mechanics and rules.

This stage aimed to define what would be happening during our game, and once that was decided we also had to define how players interact with the board, accumulate points, play minigames, how the game ends, etc.

Duration: 8 hours

Objective: Define the core game mechanics, rules, board layout, cell types, and mini-game concepts.

Tasks:

- Brainstormed game ideas and mechanics.
- Designed the 4x4 cyclic board layout and cell distribution.
- Defined the rules for player movement, point accumulation, and winning conditions.
- Conceptualized the mini-games to be included.

Dependencies: None; directly followed the initial exploration phase and laid the groundwork for software and hardware integration.

Responsible member: All

Priority: High. This was the foundational milestone upon which all subsequent development depended.

2.3. Network Connectivity and MQTT Integration

With a grasp of the individual components, we shifted our focus to establishing network connectivity and integrating the MQTT communication protocol.

Initially, we used an online MQTT broker for testing purposes, sending and receiving basic messages between devices.

We also implemented the necessary logic on the control base (ESP32) to receive and display MQTT messages on the LCD screen. This involved using queues to manage incoming messages and creating structured functions and tasks to handle different message types and functionalities. This milestone ensured reliable communication between the different components of the game.

Duration: 4 hours

Objective: Establish reliable Wi-Fi connectivity and integrate the MQTT communication protocol for seamless component interaction.

Tasks:

- Connected components to an online MQTT broker for testing.
- Implemented message handling on the control base (ESP32) using queues and structured tasks.
- Configured the LCD to display incoming data.

Dependencies: Initial Exploration and Component Familiarization.

Responsible member: All

Priority: High; critical for enabling communication between system components.

2.4. Dockerized Broker and Game Controller Prototype

To enhance development and testing, we created a Docker Compose environment incorporating a custom Dockerfile for the MQTT broker and a Python-based game controller prototype.

This setup allowed us to use a local broker and gain more control over the communication infrastructure. With the communication pipeline established, we started developing the basic structure and flow of the game controller logic, focusing on core game mechanics such as turn management and cell event handling.

Simultaneously, we worked on the control base (ESP32) to synchronize the game flow between the controller and the physical board.

Duration: 4 hours

Objective: Develop a local MQTT broker and a Python-based game controller to manage game logic and synchronization.

Tasks:

- Created a Docker Compose environment with a custom Dockerfile for the MQTT broker.
- Developed a Python-based prototype for managing turn logic and cell events.
- Coordinated with the ESP32 to ensure data synchronization.

Dependencies: Network Connectivity and MQTT Integration.

Responsible member: Mihaela, José Ramon

Priority: Medium; essential for establishing game logic infrastructure.

2.5. Game Controller and Control Base Synchronization

This milestone focused on refining the interaction between the game controller and the control base. We implemented the logic for handling cell events, player turns, and score updates, ensuring seamless communication and data synchronization between the Python-based controller and the ESP32-based control base. This stage involved rigorous testing and debugging to ensure a smooth and responsive gameplay experience.

Duration: 60 hours
Objective: Refine the interaction between the Python game controller and the ESP32 control base for smooth gameplay.

Tasks:

- Implemented logic for handling cell events, managing player turns, and updating scores.
- Tested communication protocols for consistent data exchange.

Dependencies: Dockerized Broker and Game Controller Prototype.

Responsible member: José Ramon

Priority: High; necessary for ensuring gameplay reliability.

2.6. Mini-Game Development

With the core game mechanics in place, we began developing the mini-games. Initially, we collaborated on the development of the first mini-game to solidify our understanding of the integration process. Following this, we divided the remaining three mini-games among the team members, working concurrently to develop and test them independently. This parallel development approach accelerated the implementation of the mini-game components.

Duration: 30 hours

Objective: Develop four mini-games to enhance the interactive elements of the board game.

Tasks:

- Collaboratively developed the first mini-game.

- Assigned remaining mini-games to team members for parallel development. **Dependencies:** Game Controller and Control Base Synchronization.

Responsible member: All

Priority: Medium; integral for enriching the game experience.

2.7. Meeple Integration and Refinement

This stage involved integrating the meeple (ESP01) into the overall game system. We implemented the Wi-Fi and MQTT connection logic for the meeple and ensured that the Hall sensor readings were correctly translated into MQTT messages for position tracking.

Additionally, we made minor updates and quality-of-life improvements, such as adding new sound effects, improving feedback for players whose turn it wasn't, and enhancing the user interface.

Duration: 14 hours

Objective: Integrate the meeple (ESP01) with the system and ensure precise position tracking.

Tasks:

- Configured Wi-Fi and MQTT connectivity for the ESP01.
- Translated Hall sensor readings into MQTT messages for position updates.
- Enhanced player feedback with sound effects and improved interaction.

Dependencies: Mini-Game Development.

Responsible member: Mihaela, José Ramon

Priority: High; critical for finalizing the core gameplay elements.

2.8. Documentation and Final Presentation

The final milestone focused on preparing the project documentation, including this engineering book, technical documentation outlining the system architecture and communication protocols, and organizing the project source code. We also prepared the final presentation to showcase the project's functionalities and design decisions.

Duration: 8 hours

Objective: Compile comprehensive documentation and prepare for the project presentation.

Tasks:

- Prepared the engineering book, technical specifications, and source code.
- Designed the final presentation to showcase project features.

Dependencies: All previous milestones.

Responsible member: Eric

Priority: High; ensures clear communication of project results and facilitates future development.

Each milestone was designed to build upon the previous one, ensuring a logical and efficient progression toward project completion. By adhering to this structured approach, we successfully integrated complex components and delivered a cohesive, interactive e-Game Board system.

3. Component Breakdown

This section provides a detailed overview of each implemented component, including its purpose, development time, and key takeaways.

3.1. Meeple (ESP-01)

- **a) Goal:** To track the player's position on the board and transmit movement data to the game controller.
- **b) Dedication Time:** 17 hours (including initial familiarization, Hall sensor integration, Wi-Fi/MQTT connectivity, and debugging).
- **c) Conclusion:** Successfully implemented movement detection using the Hall effect sensor and established reliable MQTT communication for position updates. Also came up with an idea to overcome lack of console debug logs by using MQTT message to handle the device's status.

3.2. Operational Base (ESP32)

- **a) Goal:** To display game information, handle player input, provide feedback through the buzzer and LCD, and communicate with the game controller.
- **b) Dedication Time:** 21 hours (including initial setup, LCD and buzzer integration, button input handling, and MQTT communication).
- **c) Conclusion:** The ESP32 proved to be a powerful platform for managing the complexities of the operational base. The use of FreeRTOS greatly facilitated multitasking, enabling concurrent handling of display updates, button inputs, and MQTT communication.

3.3. Game Controller (Python)

- **a) Goal:** To manage the overall game flow, enforce game rules, handle player turns, and coordinate interactions between the meeples and operational bases.
- **b) Dedication Time:** 78 hours (including initial prototyping, game logic implementation, MQTT communication, mini-game integration, and debugging).
- **c) Conclusion:** The Python-based game controller provided a flexible and efficient platform for implementing the game logic. This was the biggest most complex part of the project, and relying on Python helped us make this process simpler. Also making the game controller manage almost all of the game logic, allowed for faster integration of

updates, helping greatly reduce time spent on uploading new code to the microcontrollers.

3.4. MQTT Broker (Docker)

- **a) Goal:** To facilitate reliable message exchange between the game components using the MQTT protocol.
- **b) Dedication Time:** 4 hours (including Docker setup, configuration, and integration with the game controller and embedded devices).
- **c) Conclusion:** Using a Dockerized MQTT broker simplified the deployment and management of the communication infrastructure. This allowed for a more controlled and reproducible testing environment.

4. Game Definition and Rules

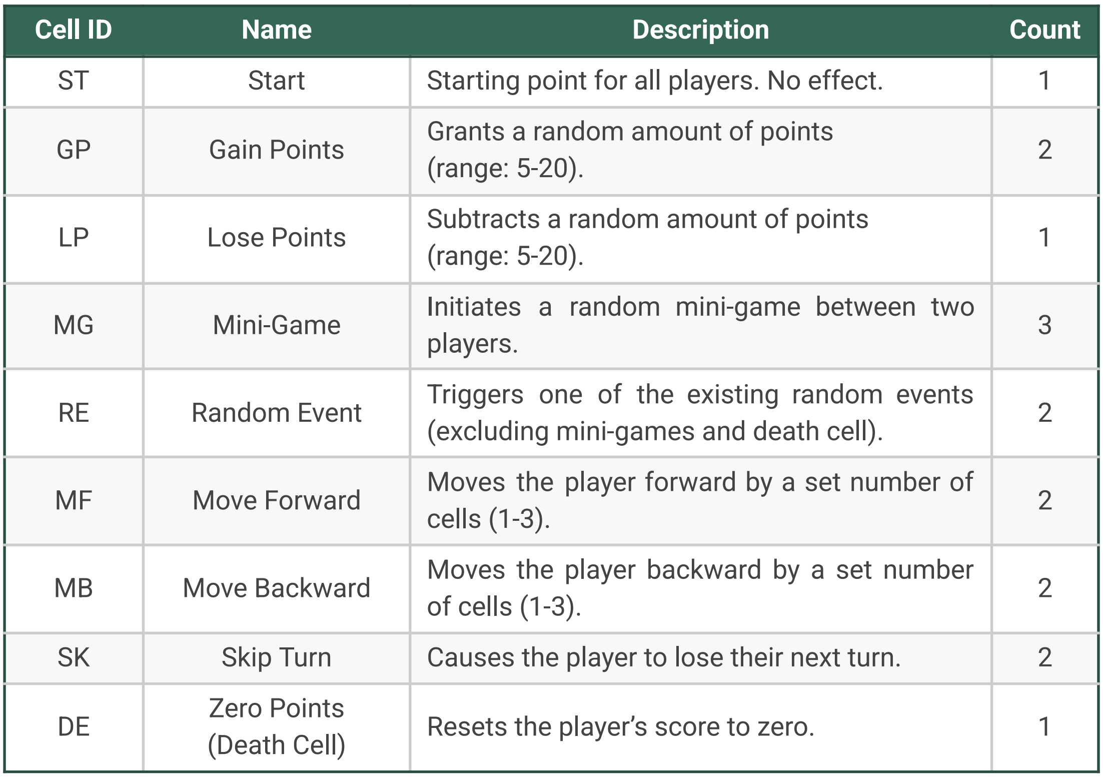
4.1. Overview

In this 4x4 board game, players move meeples across a cyclic board filled with different types of cells. Each cell impacts the players differently, offering points, penalties, mini-games, or other unique events.

The first player to reach **50 points** wins the game.

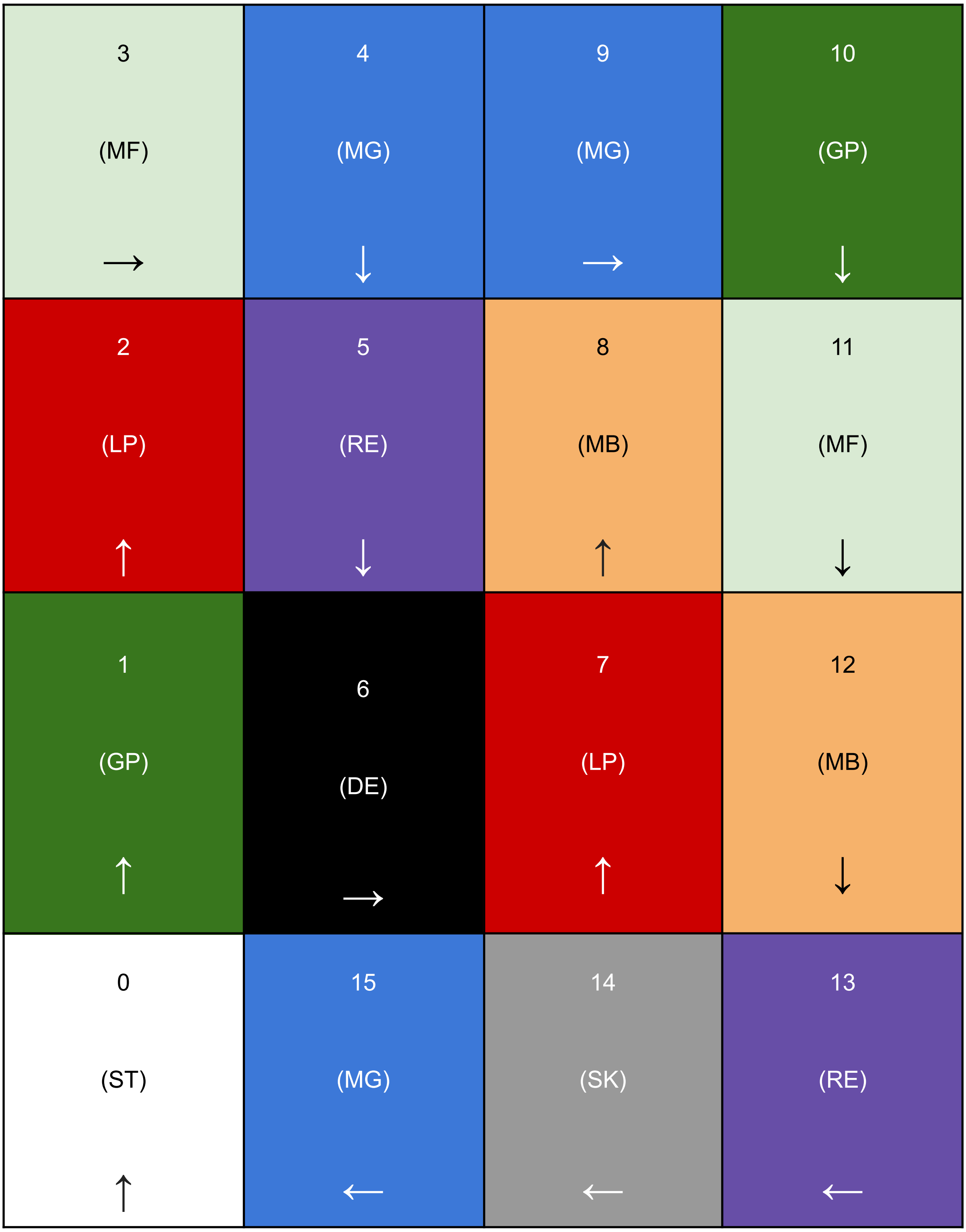
4.2. Game Board Setup

- **Board Dimensions**: 4x4 grid, a cyclic board where players move in a predefined path.
- **Cell Distribution**: The board's cells are organized to ensure a balanced mix of events, challenges, and points, as follows:



Cell ID	Name	Description	Count
ST	Start	Starting point for all players. No effect.	1
GP	Gain Points	Grants a random amount of points (range: 5-20).	2
LP	Lose Points	Subtracts a random amount of points (range: 5-20).	1
MG	Mini-Game	Initiates a random mini-game between two players.	3
RE	Random Event	Triggers one of the existing random events (excluding mini-games and death cell).	2
MF	Move Forward	Moves the player forward by a set number of cells (1-3).	2
MB	Move Backward	Moves the player backward by a set number of cells (1-3).	2
SK	Skip Turn	Causes the player to lose their next turn.	2
DE	Zero Points (Death Cell)	Resets the player's score to zero.	1

4.3. Game Board Design



3 (MF) ⟶	4 (MG) ↓	9 (MG) ⟶	10 (GP) ↓
2 (LP) ↑	5 (RE) ↓	8 (MB) ↑	11 (MF) ↓
1 (GP) ↑	6 (DE) ⟶	7 (LP) ↑	12 (MB) ↓
0 (ST) ↑	15 (MG) ⟵	14 (SK) ⟵	13 (RE) ⟵

4.4. Mini-Games

Minigames allow players to compete against other players.

A random mini-game is selected when a player lands on a mini-game cell. The following mini-games are available:

1. **Hot Potato**
    - **Objective**: Avoid being the player holding the button when time runs out.
    - **Rules**: The control base generates a hidden countdown timer. Players press the button to pass the potato, not knowing the exact time remaining. The buzzer beeps faster as the timer approaches zero, and the player holding the button when it reaches zero loses the game.

      It's reminiscent of the classic "Hot Potato" game.
2. **Number Guesser**
    - **Objective**: Guess a hidden target number without going over.
    - **Rules**: Players have to guess a number between a specified range, without knowing the control base’s hidden number. The player with the closest guess, without exceeding the target, wins the round.

      It's reminiscent of the classic "The Price is Right" game ("Precio Justo" in Spanish).
3. **Tug of War**
    - **Objective**: Pull the virtual rope onto your screen.
    - **Rules**: Players repeatedly press the button to move a virtual rope closer to their side on the control base LCD screen. The first player to pull the rope past a threshold wins.

      It's reminiscent of the classic "Tug of War" game ("Tira y Afloja" in Spanish).
4. **Last Stick Standing**
    - **Objective**: Be the player who removes the last stick.
    - **Rules**: The game starts with a random number of virtual sticks (e.g., 10). Players take turns removing 1 or 2 sticks each. The player who removes the last stick wins the game.

      The LCD screen displays the remaining sticks as lines.

      It's reminiscent of the classic "Nim" game.

Here's a summary table of the mini-games:

Mini-Game	Objective	Rules
Hot Potato	Avoid holding the button when time runs out	Hidden countdown timer. Players press button to decrement time; buzzer beeps faster as timer nears zero.
Number Guesser	Guess a hidden number without going over	Players guess a number within a range. Closest guess without exceeding the target wins.
Tug of War	Pull the virtual rope onto your screen	Players repeatedly press button to pull a rope on LCD screen; first to pull past threshold wins.
Last Stick Standing	Be the player to remove the last stick	Random number of sticks; players take turns removing 1 or 2. Last to remove a stick wins.

We also had decided to add more games and had already defined the rules and objectives. However, we ended up implementing only the four games presented above. The discarded game ideas were:

- **Blind Timer**
    - **Objective**: Estimate a target time without a clock.
    - **Rules**: The control base generates a target time (in seconds). Each player holds the button for the time they believe matches the target. The player closest to the target time wins.

      This game tests players' internal sense of time.
- **Rock, Paper, Scissors**
    - **Objective**: Outplay the opponent in a classic Rock, Paper, Scissors match.
    - **Rules**: Both players select Rock, Paper, or Scissors. Standard Rock, Paper, Scissors rules determine the winner.
- **Quick Reflexes**
    - **Objective**: Be the fastest to react when the buzzer sounds.
    - **Rules**: Players wait for an audio cue from the control base. As soon as the buzzer goes off, players must press their buttons as quickly as possible. The first player to react and press the button wins the game. However, if a player presses too early, they’re disqualified from the round.

4.5. Turn Mechanics

The game follows a turn-based system, where players take turns moving their meeples along the board. The turn mechanics are as follows:

1. When a player's turn starts, their meeple and control base LEDs light up.
2. Then, the player rolls a random movement (1-6 squares) for their meeple.
3. The player moves their meeple cell by cell, following the path on the board. As they advance, the LCD screen displays the current cell's number and the number of movements remaining.

   Moving through cells does not trigger their effects until the player lands on the final cell of their movement.
4. When the movement count reaches zero, the LCD screen displays the cell's event and its effects on the player.
   - If the player lands on a mini-game cell, a game is chosen randomly from the available mini-games and its name appears on the screen, and the game starts.
   - If the player lands on a win/lose points cell, or the zero points cell, the player's score is updated accordingly on the screen.
   - If the player lands on a move forward/backward cell, the counter is updated accordingly, and the player continues their turn, moving forwards or backwards the specified number of cells. The last cell they land on triggers its event.
   - If the player lands on a skip turn cell, the player loses their next turn.
   - If the player lands on a random event cell, a random event is chosen from the available events, and its name appears on the screen, following the mechanics described above.

Different sounds and animations accompany each event, mini-game, or action, enhancing the player's experience.

5. Conclusions

The e-Game Board project successfully demonstrated the integration of embedded systems, networking, and game design principles to create a functional and engaging interactive board game.

The project provided valuable practical experience with microcontroller programming, MQTT communication, and the challenges of coordinating multiple interconnected devices.

The final product, while not without its limitations, achieved the core objective of creating a playable game incorporating various interactive elements.

5.1 Main Challenges

Several challenges were encountered during the development process, including:

MQTT Message Overflow

The initial implementation of the Tug of War mini-game generated a high volume of MQTT messages, leading to frequent overflows and communication delays. This issue was mitigated by modifying the gameplay to reduce the message frequency. However, it revealed the importance of considering message rate and potential bottlenecks in the MQTT communication pipeline.

Hall Sensor Inconsistency

The Hall effect sensor used for meeple movement detection showed some inconsistencies, sometimes failing to register the presence or removal of a magnet. This led to significant debugging efforts, particularly challenging due to the lack of a console for real-time feedback on the ESP01.

Additionally, the absence of a dedicated power supply for the ESP01 during initial stages made us rely on the ESP32 for power supply. This also contributed to connectivity issues, as we think the ESP32 was sometimes providing insufficient power to handle the expensive Wifi and MQTT tasks, resulting in frequent disconnections, further complicating the debugging process.

ESP01 Debugging Limitations

Additionally, the limited debugging capabilities of the ESP01 significantly delayed development.

Each small code modification required reuploading and testing, a time-consuming process that slowed down progress. A dedicated debugging setup for the ESP01 would have greatly improved development efficiency. We did create a console-less debugging system through MQTT messages, but it wasn’t informative for connection problems.

5.2 Obtained Results

Despite the challenges, the project achieved its primary goals:

- **Functional and Engaging Game:** The team successfully developed a fully functional and entertaining interactive board game, a significant accomplishment considering our limited prior experience with microcontrollers and embedded systems.
- **Practical Experience:** The project provided invaluable hands-on experience in embedded systems development, MQTT communication, and collaborative project management.

5.3 Future Work

Several areas for future development and improvement have been identified:

- **Additional Mini-Games:** Implement the remaining mini-game concepts (Blind Timer, Rock, Paper, Scissors, Quick Reflexes) to further diversify the gameplay.
- **Enhanced User Experience:** Improve the game's overall user experience by incorporating more engaging sound effects, visual feedback, and animations. This would enhance player immersion and enjoyment.
- **MQTT Keep-Alive Mechanism:** Implement a keep-alive mechanism for MQTT messages to detect disconnected devices and provide appropriate feedback to the game controller. This would enhance the robustness of the communication system.
- **Hardware Enhancements:**
  - Consider using a larger LCD screen to display more information and reduce the need for strict string length management.
  - Incorporating different colored LEDs would also enhance visual feedback and provide more opportunities for creative game design.
- **Multiplayer Support (Beyond 2 Players):** Refactor the codebase to fully support more than two players, enhancing the game's social aspect.
- **Improved Meeple Design:** Explore alternative designs for the meeple component and wiring to improve portability and ease of use during gameplay. A more robust and compact meeple design would enhance the overall user experience.

By addressing these areas for improvement, the e-Game Board project can be further refined and expanded, providing an even more engaging and interactive gaming experience.