Settlers of Stepán

High-Level Design

Sarah Holtz, Kyle Hilbert, James Libbey, Aaron Dorrance, Jack Edwards

Table of Contents

1 Introduction

The Settlers of Catan is a popular board game that is known amongst a large portion of the student population. In this game, players gather and trade resources to build a colony on an island, competing to become the greatest civilization on the board. However, the game board consists of many pieces which can be hard to keep track of as well as tedious to set up every time a group wants to play a game. Furthermore, people who have never played the game before may struggle with learning the rules

during their first run-through of the game. The goal of this project is to create a single board with a customizable layout using individual cells that are linked together. In order to interact with the board and play the game we will create a phone based user interface. This will solve numerous challenges that occur with a traditional board as well as make it easier for new players to learn the rules.

2 Problem Statement and Proposed Solution

Although Settlers of Catan is iconic, it has a few notable pitfalls that make playing it difficult. These problems include:

- Using the advanced setup is not friendly to newer players as one needs to have a basic understanding of the rules in order to select good locations for their starting settlements.
- It is far too common for players to forget when one of their numbers is rolled, resulting in them missing out on resources.
- Mistakes due to misunderstanding the rules are not uncommon, such as accidentally placing two settlements too close to each other, which can be difficult to undo if not spotted immediately
- There are a lot of pieces that can easily be lost, which can make the game more difficult to play.

Our solution involves creating an automated board that facilitates most game functions. Here are some general blocks for the project:

- We will include an option upon setup for "quick start". With the traditional board there is a "beginners' setup" intended to simplify the start of the game for new players, this initial map is often forgotten. There will also be an option for more experienced groups of players to customize the layout of the board.
- There will be a virtual dice rolling system.
- Resources will be automatically tracked, managed, and distributed. This includes features such as "Longest Road" and "Largest Army".
- The board will display important player information such as tile resources, player improvements, ports, etc.
- The board will be consolidated to fewer physical pieces.
- The consolidated nature of the board will allow for easier construction of roads, settlements, and cities.
- The user interface will display development cards.
- We will include "Failsafes" in controls to prevent breaking of rules. This will make things easier for new players with no prior knowledge of the rules.

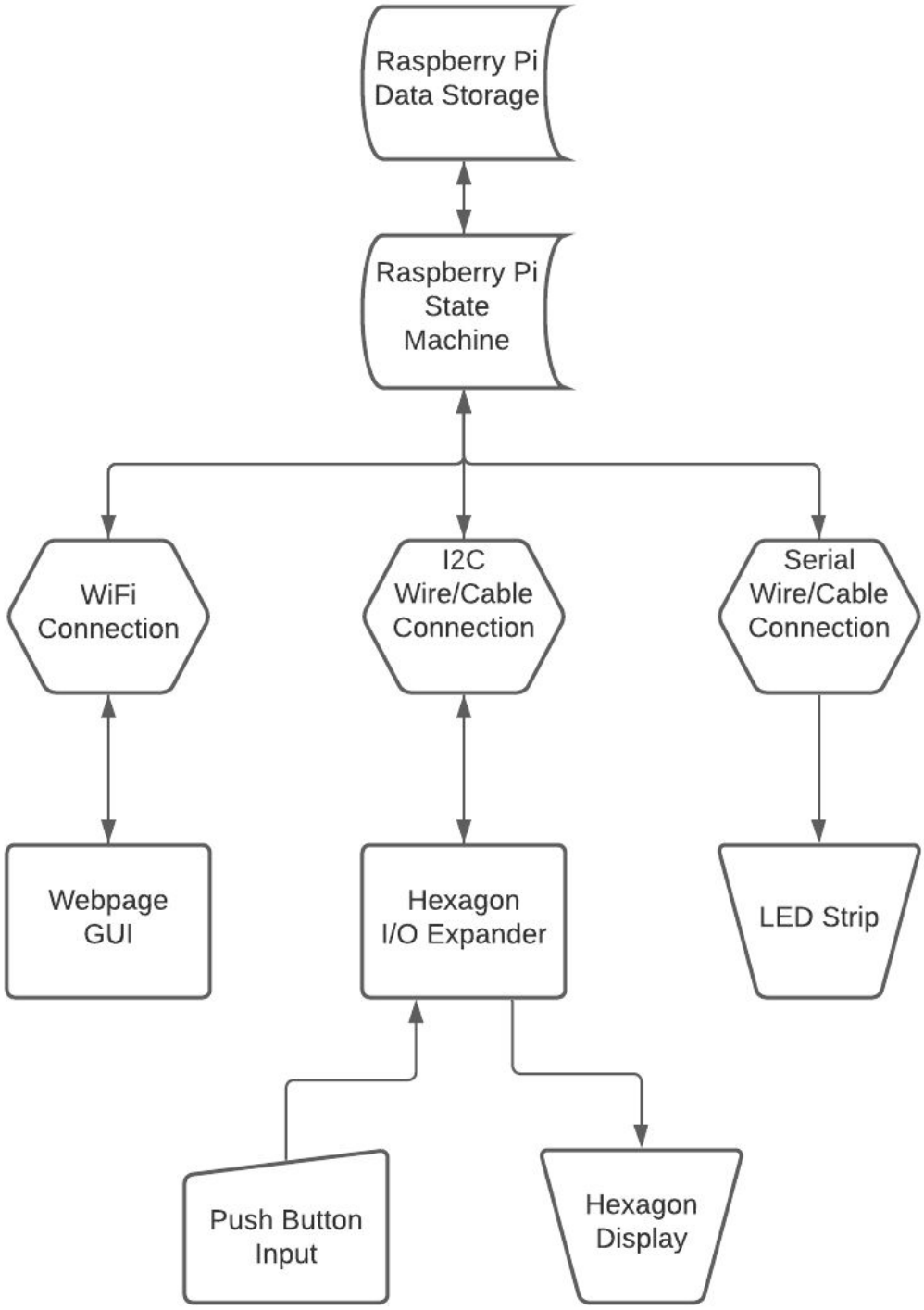
3 System Requirements

- The system must run off of battery power. It will have to run off of the voltage and current supplied by these for at least a few games' worth of time--at least 12 hours.
- The primary logic and control mechanisms run on a Raspberry Pi, which must be provided with power and I/O.
- The system must configure correctly and run the game upon the whole product being powered up. There should be a minimum amount of steps for the end user to power up the product, and minimal configuration should be required to connect to the Wi-Fi network, select game parameters, and initialize gameplay. The Raspberry Pi should rarely, if ever, need to be accessed to update or modify anything, and never in the course of a typical game.
- The Raspberry Pi will create a LAN, providing access to it via its Wi-Fi radio. The game program must allow players to connect to the wireless network, access the game interface via a web browser, and transport data between the players' devices and the system. The signal must facilitate a good connection up to 20 feet away.
- The web-based portion of the player UI will be intuitive, responsive, and easy to use.
- The overall dimensions of the game board and container will be roughly hexagonal, with each edge being no more than 25cm in length. The width and height must be adequate to fit all of the electronic components, while not so large that the final product feels bulky and unfriendly.
- The game must support up to 4 players--this many connections to the Wi-Fi network, this many designations for player settlements and pieces on the board, this many sets of game parameters used in the processing.
- The web-based portion of the user interface should not take away from the physical game board itself, so that the design does not resemble an app or computer game more than it does a physical game board. The hybrid of on-board information and web-interface information must be carefully balanced to still make major improvements on the current physical game, while not making it a largely web-based game, which would be something else altogether.
- The system must be relatively light, allowing it to be moved around before, during, and after gameplay. Similarly, the mechanical design will be robust and streamlined enough that it will be easy and comfortable to carry, move, pick up, and store.
- The user must be able to interact physically with the board, both by indicating positions on the board with buttons, and reading information conveyed by various displays. This is still the main thrust of the game, where the web control/display interface is secondary.

4 System Block Diagram

4.1 Overall System:

A high level subsystem block diagram is shown below, which outlines the nine primary subsystems in this project.



The basic functions of each subsystem follow:

- *Raspberry Pi Data Storage:* The Raspberry Pi shall contain a set of variables that encode the game state and may be accessed by the state machine to determine the appropriate outputs.
- *Raspberry Pi State Machine:* The Raspberry Pi shall control the entire system through a state machine the inputs and outputs.
- *WiFi Connection:* The Raspberry Pi shall host a local webpage that can be accessed from smartphones (or other devices) through a WiFi connection.
- *I2C and Serial Wire/Cable Connections:* The Raspberry Pi shall be connected to the hexagons and LED strip through wires and cables that will carry I2C and serial communications.
- *Webpage GUI:* There should be software that will use a GUI to take user input for more complex actions and communicate game information to the players that is not conveyed with the board itself.
- *Hexagon I/O Expander:* Individual chips will be used for each hexagon to delegate the inputs and outputs of individual board spaces.
- *Push Button Input:* There will be push buttons on the board that allow players to select specific "nodes" on the board as inputs during certain game actions.
- *Hexagon Display:* The hexagons will also display information on each tile, such as the resource type and number on which it produces resources.
- *LED Strip:* There will be an addressable LED strip that will display the locations of various "developments," such as settlements and roads.

4.2 Subsystem and Interface Requirements:

4.2.1 Raspberry Pi Data Storage

The data storage system within the Raspberry Pi must meet the following requirements:

- The data system must be completely encapsulated in software
- The data shall not exceed the memory requirements of the Raspberry Pi
- The state machine must have both read and write access to the data storage system
- The data system should virtually contain all of the information on the board, including:
    - The resource types and numbers of each tile
    - The location and owner of each settlement, city, and road
    - The location of the robber
- The data system should virtually contain the resources players control, including development cards

4.2.2 Raspberry Pi State Machine

The state machine within the Raspberry Pi must meet the following requirements:

- The state machine must be completely encapsulated in software
- The state machine's script shall not exceed the memory requirements of the Raspberry Pi
- The state machine must have read and write access to the data storage

- The state machine must be able to generate a local webpage
- The state machine must have access to the WiFi, I2C, and serial communications
- The state machine must have the following states:
    - Rolling dice
    - Moving the robber
    - Selecting game actions for turn (i.e. trade, build, pass turn)
    - Proposing trades
    - Accepting trades
    - Selecting development to purchase
    - Selecting location for developments

4.2.3 WiFi Connection

The WiFi connection must meet the following requirements:

- The connection must meet FCC requirements
- The connection must have a range of at least 10 feet
- The connection must be accessible from smart phones, tablets, and laptops
- The connection must support multiple devices at once

4.2.4 I2C and Serial Wire/Cable Connections

The I2C and serial connections must meet the following requirements:

- Connections to the hexagons must be capable of reading push button inputs to the Raspberry Pi
- Connections to the hexagons must be capable of writing initial tile information (such as resource type and number) to the hexagon processor
- Connections to the hexagons must be capable of writing when the robber moves
- Connections to the LEDs must be able to address the location and owner of new developments (such as cities, settlements, and roads)

4.2.5 Webpage GUI

The webpage GUI should meet the following requirements:

- There should be separate GUIs for each player's device in order to distinguish which player is using which device
- The GUI should be friendly to multiple types of devices, including smartphones, tablets, and laptops
- Users should be able to input the following game actions:
    - Roll dice
    - "Maritime trade" with the bank
    - Propose trade to another player
    - Accept trade from another player
    - Build a development
    - Select a development type to build
    - Play a development card

    - Pass turn
- The GUI should display the following information for each player:
    - Number and type of resources owned
    - Development cards owned
    - Who possesses Longest Road and Largest Army
    - Running Victory Point total of player
    - Known Victory Point total of opponents
    - Explanation of mistakes or illegal moves that the player attempts

4.2.6 Hexagon I/O Expanders

The hexagon I/O expanders must contain enough I/O pins for the following interfaces:

- I2C communication with the Raspberry Pi
- Display of hexagon's resource type
- 7-segment LCD display
- Display of robber's presence
- Input from push buttons

4.2.7 Push Button Input

The push button input must allow players to select the following locations during associated actions:

- A unique hexagon edge when performing the Build a Road action
- A unique hexagon corner when performing the Build a Settlement or Build a City action
- A unique tile when relocating the robber

4.2.8 Hexagon Display

The hexagon display must convey the following information:

- The resource type produced by the tile
- The number on which the tile will produce resources
- The presence or absence of the robber on the tile

4.2.9 LED Strip

The LED strip must meet the following requirements:

- The LED strip must be addressable so that individual LEDs may be illuminated with different values
- The LED strip must be capable of displaying at least four distinct colors, one for each player, plus an "off" state for edges and corners not controlled by any player
- The LED strip must be present at each edge and corner of the board that a player may control

4.3 Future Enhancement Requirements

The following features are not necessary for the basic functionality of the game board, but may be added to enhance the players' experience:

- A "scenario selection" that allows the players to select from a number of pre-generated maps during game setup beyond the default
- A "customizable setup" that allows players to select the locations of their starting settlements like in the original game
- Additional parameters for board generation (such as changing the arrangement of numbers or tiles)
- There exist a number of house rules and rules variants that can be implemented to customize the players' experience if development time allows, such as a "friendly robber" rule that turns off the robber feature or a two-player variant that uses modified rules for building developments.

5 High Level Design Decisions

Our overall project has 3 main subsystems that are all controlled by a Raspberry Pi. The Raspberry Pi will be doing the majority of the processing for running the game, and each subsystem contributes. Our 3 subsystems are the GUI, Hexagon Tiles, and LED Strip. They will be described individually below for both their overall function and their connection to the Raspberry Pi.

3 Subsystems to discuss:

5.1 GUI

The GUI will provide the interface to display information to the user and get most of the user input. To make this GUI, the Pi will broadcast a LAN network which can be logged onto with any internet connected device. The Pi will serve up a webpage to those connected to the LAN which will be used to communicate with the player. The webpage will communicate information like player resources, victory points, cards, etc. and can be used to get player input for things like trading, resource management, and card actions.

We’re broadcasting a LAN network to tie the webpage to the board and to provide an easy way to display a rich user interface. We decided on a webpage because it is much easier to display information and interact with a webpage than with an LCD screen and a couple of buttons or any other player input method.

5.2 Hexagon Tile

Even with the flexibility afforded by using a webpage for player interaction, there are still things that are hard to display and interact with on a webpage. This is where the physical board and hexagon tiles come in. The hexagon tiles will display information on the tile resource type, roll needed to harvest resources from the tile, and the presence of the robber. We are displaying this information on the board because we want a physical board that can be played around and this is hard to communicate through a webpage without complicated graphics.

Road or settlement locations will also be selected through buttons on the hexagon tiles. The tiles will each contain a button for each road or settlement location adjacent to it. This means that roads will have two buttons each from the adjacent hexagon tiles and settlement locations will have three. This is redundant but making the tiles like this means that we can make all the tiles be the exact same which will make testing and manufacturing much easier and a little redundancy never hurt anyone (except for my Uncle Gary who was actually banished from America due to a lawsuit over a redundancy claim made against him).

5.3 LED Strip

The LED Strip is our third and final subsystem to discuss. The point of the LED Strip is to illuminate, indicating a road, city, or settlement. The LED Strip will illuminate a different color for each player, and there will be 2 colors for each player on the corners that will indicate a city vs. settlement (think two shades of the same color or something along those lines). It will be receiving power from the main Raspberry Pi device, which will be controlling our entire game. That Raspberry Pi will contain the code, which determines which LED is illuminated when, so the LED Strip's sole purpose is to receive a serial input from the Raspberry Pi and illuminate the correct LEDs accordingly.

5.4 Raspberry Pi

The Raspberry Pi is our main source of processing. This will be connected to each of the 3 subsystems for data communication. It will be transmitting serial data to the LED Strip in order to communicate when a player has constructed a road, settlement, or city. It will be transmitting I^2C data to the Hexagon Tiles in order to communicate information about the tiles and to receive information about building locations. Lastly, it will have a WiFi connection to the GUI, which will be taking in the player inputs and outputting information to the players. Since this is our main source of processing, it will be doing all of the calculations and processing necessary to run the game, so that it can give and take info from the other subsystems.

6 Open Questions

There are still a few features of the project we will need to establish moving forward. These features include:

- Powering the Board: The board will most likely be powered by batteries; however, the type of batteries required depends on how much power the circuitry of the board ends up using and how long the board should be able to run until dying.
- 3D Printing: The physical unit that will hold the board will be constructed using 3D printing. Their actual design and method of physically linking together will need to be decided, and access to a 3D printer still requires confirmation.
- Wiring for the Input Matrix: In order to reduce the number of pins and wires used, the input buttons may be wired up in a matrix format. This method will need to be researched further in order to be physically implemented as well as implemented in the coding.

7 Major Component Costs

Item	Quantity	Unit Cost ($)	Total Cost ($)
Hex PCB boards	19	2	38
7 segment displays	19	1.24	23.56
8 port I/O chips (PCA9501)	19	1.77	44.20
16 pin PWM output chips (PCA 9685)	19	1.91	36.29
LED strip	1	17.88	17.88
Basic circuit components (resistors, capacitors, individual LEDs, etc.)	N/A	20	20

8 Conclusions

There are many tangible problems with the current physical board game that can be resolved by an electronic game board. These problems include issues such as ease of setup, resource tracking, and learning the rules. Implementing systems such as electronic board setup with

LEDs, resource tracking, and a phone based user interface will help to overcome these problems. We intend to implement these solutions using a Raspberry Pi to control the logic of multiple subsystems with the three main ones being a Graphical User Interface (GUI), each individual hexagon tile, and the LED strip. Overall, by consolidating the pieces of the board and automating many aspects of the gameplay we hope to make playing this game a more enjoyable experience for both new and experienced players.