1 Introduction

1.1 Objective

Ticket to Ride is a fun game to play with friends, but when it comes to counting your score, it can put a strain on your relationship. The scoring process includes tallying your scores obtained from laying down train cars on certain paths and trusting that your friends did it correctly. It also includes the longest path bonus, which can take a while to count and determine whose path is longer. Destination cards can be confusing to determine. Besides keeping track of the score, it is also tedious to place the individual train cars on the board to claim a route. This can take up a lot of time and slows the game down unnecessarily.

Our solution to this problem is to automate the scoring process of this game. Automation will greatly increase enjoyment, alleviating the most cumbersome and time consuming part of the whole experience. The automation will include indicating the current player’s turn, showing current scores, and automatically calculating end-game scores including bonuses from destination cards and longest path. Instead of manually placing train cars on the board, LEDs will automatically be illuminated when a player captures a route. The color of the LEDs will indicate which player controls the route.

1.2 Background

From personal experience, tallying score is the worst part of Ticket to Ride. This sentiment is shared by Jacob Bryan (current TA for this course), who first suggested we implement the solution to this problem. There is feasible demand for a product like this, as there are automated versions of popular board games out there already, such as electronic Monopoly and The Game of Life. There was also a project in 2013 that automated the scoring process of Settlers of Catan that performed well - showing that there was a demand for automated scoring processes for board games like Ticket to Ride.

1.3 High-Level Requirements

- Train spaces should light up when the corresponding path is claimed.
- The game must be able to automatically calculate and display the players’ score during the game, as well as automatically factor in longest path bonus and destination cards at the end of the game.
- The game should automatically end when a player’s number of train cars dips to 2 or below.
2 Design
2.1 Block Diagram

The user input, control, and LED modules will take care of lighting the LEDs when a path is taken. The MCU will have dedicated memory to be able to store player scores as well as calculate scores in real time. The MCU will also keep track of number of train cars for each player, allowing for an automatic end to the game when the number of train cars goes below 2 for any player. The MCU accomplishes this by receiving signals from the control module that indicate what action each player took on their turn. These signals are then decoded by the MCU and used to update the game state. The LCD module is connected to the MCU and is used to give feedback to players about their current point totals. This screen will indicate when the game has ended and also show all of the players’ point totals so that the winner is determined immediately.
2.2 Physical Design

Figure 2: The Physical Board

Above is an image of the Ticket to Ride board game. Buttons will be placed at each of the different cities (the orange dots on the board) and will be used to indicate that a player is taking control of a rail line between two cities. The buttons will also allow a player to enter in the two endpoints listed on a destination card drawn. In addition to the buttons on the cities, there will be a few more buttons located in the lower left corner of the board allowing the users to indicate the number of players, pass the turn along to the next player, and indicate that they are entering a destination card rather than claiming a rail. Instead of placing the plastic trains to claim a route, LEDs will light up along each route to indicate that it has been claimed. The color of the LEDs on the path will be different depending on which player took control of the route. An LCD screen will be placed in the lower right corner of the board (where the scoring details are now located) and will display the current point total for each player and indicate which player is currently taking their turn. There will be a one inch deep wooden housing constructed that the game board will rest in - all of the wiring and electrical components can be placed in the housing to achieve a polished look in the finished product.

2.3 Functional Overview
2.3.1 User Interface Module

The user interface module will output a data signal to the control system module based on the specific buttons pressed by the user. The user interface will consist of the buttons that are laid on each city. These will be used to input destination cards as well as take a path during your turn. Users can claim a path by pressing down two city buttons simultaneously. The control module will handle the button presses and update the current player’s score accordingly. In addition to buttons on the cities, there will be buttons used to indicate the number of players, end a player’s turn, and switch to a different mode to enter destination cards. All of these button
presses will be handled by the control module and the MCU module will update the game state appropriately.

2.3.2 Control System Module

The control module will take a data signal input from the user interface module, as well as output a power line to the LED module and a data signal to the MCU module. The control module will include state machines and set control signals necessary for gameplay, as well as any button presses that occur in the user interface, such as number of players, end turn, and destination card input mode. When a set of buttons are pressed, the control module will determine if the move is valid. If the move is not valid, the module sends a ‘not valid’ signal to the MCU. If the move is valid, the control module will encode the path that was claimed, send the command to the MCU block, then turn on the LEDs associated with the path.

2.3.3 LED Module

The LED block will be used to signify the current player, current mode of input, and who has claimed which path. This will consist of multi-colored LEDs under each tile on the board. The LEDs will be directly controlled by the users button inputs. LEDs can also be used to give the users feedback about the choice they are currently making. For example, some cities are connected by a dual set of train lines. If a player is trying to claim one of these two lines (and neither has yet been claimed) the LEDs for one line could blink, indicating that the player needs to select between the two lines. The buttons could be pressed to switch the selected line, and the LEDs would be turned on continuously when a player has selected which path they are claiming. The LEDs will be given power at the discretion of the control module.

2.3.4 MCU Module

The MCU module will receive commands from the control module - in the form of a data input - for what move was made by the current player, as well as number of players and current mode. The MCU block will keep track of cumulative user score and output a data signal to the LCD display. This module will also be responsible for calculating and assigning the longest route bonus as well as handling the destination cards that each player had entered in. The control system module will tell the MCU module what specific move was taken and the MCU will then have to update the game state to handle this action. The MCU will communicate with the LCD unit to display information to the players. If an error was received from the control unit, this will be decoded and displayed on the LCD in a human readable error message. Updates on the scores and current player turn can also be sent to the LCD for display.

2.3.5 LCD Module

The LCD display will receive current score, turn, and mode from the MCU and output to show the real-time score calculations. These pieces of information will be formatted in such a way that they will be human readable on the LCD screen. Power for the screen will be provided directly from the power module. These messages will display any errors to the users and also make it easy to check everyone's current score.
2.3.6 Power Module

The power module will consist of AA batteries and be used to send a power line to the whole board. Voltage regulators and current-regulating resistors will be used to stabilize power. Diodes can be used to prevent any damage to the electronics if the batteries are inserted backwards. It is important to keep power consumption to a minimum to maintain battery life, as the need to plug into a wall outlet severely detracts from the experience of playing a board game.

2.4 Block Requirements

<table>
<thead>
<tr>
<th>Module</th>
<th>Requirement</th>
</tr>
</thead>
<tbody>
<tr>
<td>User-Input</td>
<td>● Provide debounced signals to the control module representing the user input.</td>
</tr>
<tr>
<td></td>
<td>● Signals should be provided at 3V with a tolerance of ± 0.5V.</td>
</tr>
<tr>
<td>Control System</td>
<td>● Encode signals from the User-Input module to be sent to the MCU.</td>
</tr>
<tr>
<td></td>
<td>● Provide 3V with a tolerance of ± 0.5V to the correct LEDs (based on user input) in less than 0.25 s.</td>
</tr>
<tr>
<td>LED</td>
<td>● Receive 3V with a tolerance of ± 0.5V and limit current using resistors.</td>
</tr>
<tr>
<td></td>
<td>● Display the paths taken, number of players, and the current turn.</td>
</tr>
<tr>
<td></td>
<td>● LEDs must be visible in a well-lit room.</td>
</tr>
<tr>
<td>MCU</td>
<td>● Decode signals from the control module to calculate current scores.</td>
</tr>
<tr>
<td></td>
<td>● Run an algorithm to determine success of destination cards and longest path bonus.</td>
</tr>
<tr>
<td></td>
<td>● Output game-state information to the LCD screen.</td>
</tr>
<tr>
<td>LCD</td>
<td>● Receive input from MCU to display current scores, turn, and number of cars left.</td>
</tr>
<tr>
<td>Power</td>
<td>● Output 3V with a tolerance of ± 0.5V from AA alkaline batteries.</td>
</tr>
</tbody>
</table>

Table 1: List of Requirements by Module
2.5 Risk Analysis

The control block is the block that poses the most risk to the completion of this project. The control unit needs to control LEDs, encode path data, and send it to the MCU for score calculation. It is the hardest module to implement solely because of the complexity of the hardware involved. There are a large number of routes that all have to produce a unique code that can be sent to the MCU. This unit will also be responsible for checking whether or not a move is valid. It will have to keep track of which routes are claimed and implement logic to check that the move a player has entered does not interfere with an existing route. Both of these components will require a significant amount of hardware design that can be time consuming and difficult to troubleshoot. The MCU module could also pose some issues as neither of us are very familiar with using an MCU, but we know C and have used an Arduino before so there should be enough background knowledge to carry us through that module.

3 Ethics and Safety

We will uphold the IEEE code of ethics\(^1\) during the development of our project. Any intellectual property concerns about building upon an existing board game will be avoided as we do not intend to sell the final product. Our project has very minimal safety concerns, and could even improve upon the safety of the original board game. We will be eliminating the need for small game pieces and thus will be able to remove a choking hazard for small children. All of our electronics will be safely housed underneath the board in an insulated environment. We will take care that there are no exposed electrical components that could harm users. We will also put in a fail-safe to protect our circuit should the batteries be put in the wrong way, so there will be no chance of fire or frying our circuit.

References