Electrical Overview

Year: 2015 **Semester:** Spring **Team:** 2 **Project:** R.I.S.K
Creation Date: February 4, 2015 **Last Modified:** February 6, 2015
Author: <redacted> **Email:** <redacted>

Assignment Evaluation:

Item	Score (0-5)	Weight	Points	Notes
Assignment-Specific Items				
Electrical Overview				
Electrical Considerations				
Interface Considerations				
System Block Diagram				
Writing-Specific Items				
Spelling and Grammar				
Formatting and Citations				
Figures and Graphs				
Technical Writing Style				
Total Score				

5: Excellent 4: Good 3: Acceptable 2: Poor 1: Very Poor 0: Not attempted

General Comments:

1.0 Electrical Overview

The primary computation component is a 32-bit, 64-pin microcontroller, which will handle all of the game logic, input processing, and output preparation. The web server host will maintain images and text and communicate with each player's mobile device. The inputs will be buttons and knobs for selecting and confirming actions. Output devices include 7-segment LEDs, RGB LEDs, and an LCD display. Each bit sent out to the displays corresponds to a light being lit or darkened.

2.0 Electrical Considerations

2.1 Operating Voltage

The main operating voltage of the game will be 3.3V. The microcontroller we have chosen (PIC32MZEC) allows a maximum of 3.6V to be supplied [1], and the RGB LEDs we have chosen require a minimum of 3.0V to operate (for green and blue; red requires only 1.8V) [2]. 3.3V falls within this range, and it should be easy to keep the produced voltage within our 3.0V to 3.6V limits. Although 3.4V would have been an acceptable choice, 3.3V supplies are more common, and therefore easier to find.
Two of the components, the LCD screen [3] and the Raspberry Pi [4], require a higher input voltage, 5V. Therefore, part of the board will be dedicated to these 5V components, and a converter and level translators will be required to power and interface with these components.

2.2 Operating Frequency

We would like to prevent the players of the game from noticing any flickering or other effects from updates to the displays. Initially, we thought that the entire lighting update interval should be less than 10ms. This would mean ensuring that all of the data bits leaving the microcontroller (24 bits per country * 42 countries ~= 1000) would have to be completed, leading to an output data rate of ~100kHz.
However, the chosen shift registers allow all of the bits to be shifted in and out, and a separate clocking signal pushes those bits to the output pins. Therefore, any flicker the users would notice would be limited by the time it takes that single clock pulse to propagate throughout the system (negligible). We would like the updates to appear instantaneous, or within about 1/20 of a second.
Through our own tests on the RGB LEDs, we determined that having the LEDs on about 10% of the time is sufficiently bright. Therefore, we would like to be able to pulse-width modulate in software to reduce the power consumption of the LEDs, with granularity down to at least 1/10 of 50ms. This leads to a frequency of about 200kHz. The microcontroller can handle SPI output at 50Mbp/s, and the shift registers can handle changes at 50MHz.

2.3 Power Supply

The power to the board will be supplied by a wall adapter running at 3.3V and 3A. Most of the components on the board require 3.3V to operate. These will consume 1470mA of the current supplied. The breakdown for this usage is:

- 131 x RGB LEDs[2] * 35mA * 10% duty cycle = 460mA
- 89 x 7-seg displays * 13mA/segment * 10% duty cycle = 810mA
- Microcontroller[1] = 200mA (absolute maximum on datasheet)

The board will also have a regulator for increasing the 3.3V input to a 5V output that can supply up to 320mA. The breakdown for this power (at 5V) is:

- LCD Display[3] = 80mA
- Raspberry Pi[4] = 140mA
- WiFi Adapter[4] = 100mA

3.0 Interface Considerations

3.1 Serial Interface

A serial interface will be used to send all of the lighting information to the country boards. This will be handled with three output pins from the microcontroller: one pin for the serial data, one pin for the s-clk (serial clock), and one for the r-clk (register clock). These separate clocks are both inputs to the shift register ICs. On each s-clk, the bits are shifted to the next position in the shift register. On each r-clk (pulsed once per lighting update), the bits are copied from the shift registers into the output latches connected to the output pins. This will require the use of one of the SPI modules on the microcontroller, as well as an additional GPIO pin for the r-clk.

3.2 Pi Interface

The Raspberry Pi will communicate with the microcontroller using SPI. The microcontroller has 4 sets of SPI wires and the Raspberry Pi has several available, and SPI minimizes the number of wires that will need to be level translated. We will set up the Raspberry Pi as the master for this connection and the microcontroller as the slave because the Pi only has master mode and the microcontroller supports both (with each SPI interface having an independent setting). The operation will still be as if the Pi is the slave since it is providing user input and displaying data that it receives. This connection has a limited maximum speed of 50Mb/s on the microcontroller's end with an expected usage of less than 1Mb/s.

4.0 Sources Cited:

[1] Microchip. (2013) "PIC32MZ Embedded Connectivity (EC) Family" [Online]. Available: http://www.mouser.com/ds/2/268/60001191B-271441.pdf [Feb. 6, 2015].

[2] China Young Sun LED Technology Co., Ltd. "Model No.: YSL-R596CR3G4B5C-C10 RED/GREEN/BLUE Triple Color LED" [Online]. Available: https://www.sparkfun.com/datasheets/Components/YSL-R596CR3G4B5C-C10.pdf [Feb. 6, 2015].

[3] Gravitech. "LCD-20x4Y- 20x4 Liquid Crystal Display with HD44780 driver" [Online]. Available: http://www.mouser.com/ds/2/595/LCD-20x4Y_Datasheet-257394.pdf [Feb. 6, 2015].

[4] "Power Supply," *raspberrypi.org*, Jul. 17, 2014. [Online]. Available: http://www.raspberrypi.org/documentation/hardware/raspberrypi/power/README.md [Feb. 6, 2015]

Appendix 1: System Block Diagram

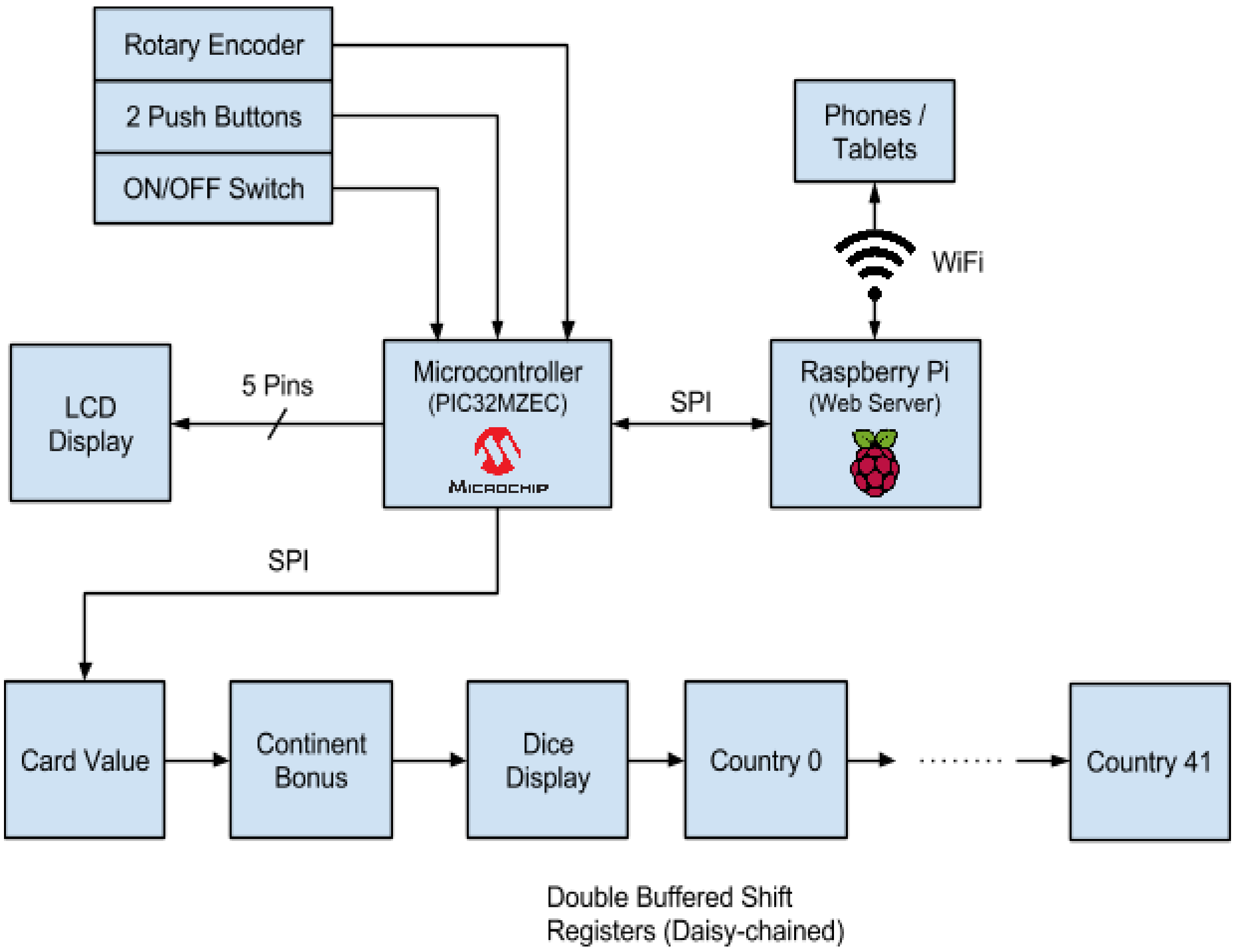

Figure 1