Pinewood Derby Finish Line Detection System

by

Cody Clayton _______________________________

Robert Schreibman _________________________________

A Technical Report Submitted to the Faculty of Electrical Engineering Colorado School of Mines

Submitted in partial fulfillment for the requirements of EENG 383 – Microcomputer Architecture and Interfacing

December 12, 2016
Chapter 1: Design Goals

1.1 Need Statement

In 1953, the very first Pinewood Derby race took place in California at the Manhattan Beach Clubhouse. Since then, this competition amongst cub scouts has become very popular and is now an annual tradition [1]. Pinewood Derby race is a competition that allows young aspiring engineers to design a miniature car used for racing along a downhill track. The object of this race is simple. Build a unique car by adjusting the shape, size, and weight distribution of a wooden block that will allow the car to reach the finish line in the fastest time. The cars are propelled by gravity alone and must comply with design specifications according to the ACCC AWANA Grand Prix Event Rules and Procedures [2]. It is estimated that since the very first race, there have been over 500 million kids that have build cars and participated in Pinewood Derby races. As a result of an increasing interest in this event, the rules and regulations have become very precise and there has been a demand for ways to accurately measure which Derby car finished first. Today, official competition timing systems can cost upwards of $1,000 [3]. However, at this price, many communities are simply unable to purchase a system and so other methods are used to determine the winner.

Some of the alternative methods used to determine the winner for the Pinewood Derby race can have a margin of error that is subjective. One method is to have a judge watch the finish line to monitor the order of the cars and determine a winner. The problem with this is that it can be difficult to accurately discern the order of the cars especially since they can finish within just millimeters of each other. In addition, the judge might also have issues remembering the exact order. Adding in more judges can help but it can also lead to more disputes since human observations can yield imprecise results. Likewise, another method uses a camera to record the finish line and is replayed in slow motion. This is a better solution, but it can still be difficult to determine the winner due to the low frame rates of inexpensive cameras. This uncertainty can cause arguments that ruin the experience for everyone.

There is a need for an inexpensive system that can accurately detect the winner of a Pinewood Derby race and then display the results quickly and clearly.

1.2 Marketing Requirements

The solution needs to be:
- Able to determine race results accurately
- Easy to turn on and reset
- Easy to read the results
- Easy to assemble and disassemble
- Inexpensive
1.3 Level-0 Description

Figure 1 shows the basic input/output functionality to the system. The inputs will consist of four different signals, one from each lane, and based off of the chronological order of the signals, there will be RGB LEDs used to display the order of the color of the corresponding lane. Table 1 is provided to explain the some of the other components of the Interface.

![Figure 1: The Level-0 Diagram for pine car derby finish line detector.](image)

<table>
<thead>
<tr>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Colored Lanes</td>
<td>One car will be on each lane. The cars start at the top of the track and are released mechanically. The first one to cross the finish line is declared the winner.</td>
</tr>
<tr>
<td>Reset Button</td>
<td>Resets the LED display when pressed. This will be used in between race heats.</td>
</tr>
<tr>
<td>Power Switch</td>
<td>The switch to turn on the device.</td>
</tr>
<tr>
<td>1st-4th Place LEDs</td>
<td>The LEDs will all be white when the system is ready, and then the LEDs will be updated as the cars finish. This will display the winner to the user. The order of LED color display will show the user the order in which the derby cars finished.</td>
</tr>
</tbody>
</table>

Table 1: The Level-0 Input/Output Table for pine car derby finish line detector.
Chapter 2: Detailed Design

2.1 Level 1 Description

We will be using the HCS12 microcontroller, and a phototransistor array to determine the winner of the race. The results will then be displayed on an array of WS2811 RGB LEDs. The amount of light entering each lane is converted into a voltage by the phototransistor board. These voltages are then fed into 4 of the analog inputs on the HCS12. When the voltage reaches a certain threshold we will know a car has crossed the line. The HCS12 will output the proper signal to the LED array to display the results. The Level 1 diagram and I/O descriptions for each module are shown below in figure 2 and tables 2 through 4 respectively.

Figure 2: The Level-1 Diagram for Figure 1.
Module | Phototransistor Board I/O Descriptions
--- | ---
**Inputs** | The light coming into the phototransistor in each lane.
**Outputs** | The phototransistor for each lane will output an analog voltage between 0v and 5v called PT0-3.
**Behavior** | When there is no car above the phototransistor it will output a high voltage because of the lights above the track. When a car passes over the finish line the voltage will drop to near 0v.

Table 2: The I/O table representing the Phototransistor Board module.

Module | HCS12 I/O Descriptions
--- | ---
**Inputs** | PT0-3: Analog voltage signal from each lane.
             Reset: Signal from push button.
             Power Switch: Input supplying power to the system.
**Outputs** | PTT0 will output the correct waveform to supply data to the WS2811 array.
**Behavior** | When the voltage on one of the analog pins drops by a threshold amount the HCS12 will mark that lane as finished and send the proper data to illuminate the correct LED with that lane’s color.

When the reset line is asserted the HCS12 will reset all of the LEDs to white. This signifies that the device is ready for a race to occur.

The power switch line provides power to the HCS12. The device will not function without this enabled.

Table 3: The I/O table representing the HCS12 module.

Module | WS2811 Array I/O Descriptions
--- | ---
**Inputs** | LED Data: An input array that conveys the order of lanes.
**Outputs** | RGB LEDs: Displays the colors corresponding to input array value.
**Behavior** | The WS2811 LEDs will remain white until an input signal from the HCS12 is used to tell the LEDs which of the four colors to turn into: red, blue, green, yellow.

Table 4: The I/O table representing the WS2811 LED array module.
2.2 Calculations

We will be using an array of WS2811 LEDs as the display. The WS2811 is an RGB LED that is controlled through a data_in line. The LED can then take the data and pass it to other WS8211 chips wired in series. These LEDs will then require very precise timing to properly propagate the signal to LEDs further down the chain. Data must be sent at 400 kHz. The table below shows the signals required to send data to the WS2811[4]. Figure 3 shows a diagram of these signals.

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>T0H</td>
<td>0 code, high voltage time</td>
<td>0.5 μs ±150ns</td>
</tr>
<tr>
<td>T1H</td>
<td>1 code, high voltage time</td>
<td>1.2 μs ±150ns</td>
</tr>
<tr>
<td>T0L</td>
<td>0 code, low voltage time</td>
<td>2.0 μs ±150ns</td>
</tr>
<tr>
<td>T1L</td>
<td>1 code, low voltage time</td>
<td>1.3 μs ±150ns</td>
</tr>
<tr>
<td>RES</td>
<td>Reset, low voltage time</td>
<td>Above 50μs</td>
</tr>
</tbody>
</table>

Table 5: This table shows the signal properties to communicate with the WS2811 LED

Since we are using such a high frequency we will need a 1:1 prescaler. Table 6 below shows the calculations for the number of counts required to wait for the correct amount of time to pass. This number of counts will be used to generate the signals in Figure 3.
<table>
<thead>
<tr>
<th>Name</th>
<th>Counts</th>
<th>Equation</th>
</tr>
</thead>
<tbody>
<tr>
<td>T0H</td>
<td>12</td>
<td>([0.5\mu s] \times \left[ \frac{1s}{10^6\mu s} \right] \times \left[ \frac{24 \times 10^6\text{clks}}{1s} \right] \times \left[ \frac{1\text{cnt}}{1\text{clk}} \right] = 12 \text{ cnts} )</td>
</tr>
<tr>
<td>T1H</td>
<td>29</td>
<td>([1.2\mu s] \times \left[ \frac{1s}{10^6\mu s} \right] \times \left[ \frac{24 \times 10^6\text{clks}}{1s} \right] \times \left[ \frac{1\text{cnt}}{1\text{clk}} \right] = 29 \text{ cnts} )</td>
</tr>
<tr>
<td>T0L</td>
<td>48</td>
<td>([2.0\mu s] \times \left[ \frac{1s}{10^6\mu s} \right] \times \left[ \frac{24 \times 10^6\text{clks}}{1s} \right] \times \left[ \frac{1\text{cnt}}{1\text{clk}} \right] = 48 \text{ cnts} )</td>
</tr>
<tr>
<td>T1L</td>
<td>31</td>
<td>([1.3\mu s] \times \left[ \frac{1s}{10^6\mu s} \right] \times \left[ \frac{24 \times 0\text{clks}}{1s} \right] \times \left[ \frac{1\text{cnt}}{1\text{clk}} \right] = 31 \text{ cnts} )</td>
</tr>
<tr>
<td>RES</td>
<td>1200</td>
<td>([50\mu s] \times \left[ \frac{1s}{10^6\mu s} \right] \times \left[ \frac{24 \times 10^6\text{clks}}{1s} \right] \times \left[ \frac{1\text{cnt}}{1\text{clk}} \right] = 1200 \text{ cnts} )</td>
</tr>
</tbody>
</table>

Table 6: Calculations to determine the correct number of timer counts for each part of the signal.

The HCS12 will send a reset signal followed by a string of 24 bits, coded like Figure 3, for each of the four LEDs which correspond to each finishing position. This will update each LED in the array. The first 8 of the 24 bits contain the Red value of the LED with the MSB first. The remaining 16 bits hold the 8 bit values for Green and Blue respectively.

### 2.3 Technical Requirements

The finish line detection system should be able to detect the order of the derby cars to within a few millimeters. This distance was used for two reasons. It is sufficient enough for the race results and this distance is about accuracy limitation of lining up the phototransistors in a line. It is likely that the phototransistors will not be perfectly lined up and will have a positioning margin between one millimeter and one centimeter due to human placement. Because of this, the most accurate waveform needed will be 1ms long since the average derby car finishes at about 3m/s.

\[
\left( 1 \text{ mm} \times \left( \frac{1m}{1000\text{mm}} \right) \times \left( \frac{1s}{3m} \right) \right) = 1\text{ms}
\]

which is closest to a one to one prescaler. The system will use the counter overflow which will give us a sampling rate of 366 Hz. This is the minimum needed to detect the 1ms difference.

The finish line detection system needs to be able to display the race results in a manner that is easy to read. The RGB LEDs will be placed left to right and will change colors according to the order of the cars. Labeling 1 through 4 will be placed below the LEDs from left to right. The colors will be very distinct and will match the colors of the lanes. These components should make the outcome easy to interpret by a judge.
In order to make the device easy to use there will only be 2 buttons. An on/off switch and a reset button. The user will know the device is turned on because the RGB LEDs will default to white to show the system is on and running.

Setting up the device will be simple as well. There will only be two cables, a power cable and an ethernet cable. This should be a quick and intuitive connection from the wall outlet to the microcontroller and then to the LED display.

The entire device will cost less than $50 when the expensive SSMI board, which we are required to use, is replaced with an inexpensive microcontroller allowing our users to afford the system over the expensive alternative.

2.4 Bill of Materials

<table>
<thead>
<tr>
<th>Material</th>
<th>Quantity</th>
<th>Acquisition Status</th>
<th>Details</th>
</tr>
</thead>
<tbody>
<tr>
<td>WS2811 RGB LEDs</td>
<td>4</td>
<td>Acquired</td>
<td></td>
</tr>
<tr>
<td>Super Bright LEDs</td>
<td>4</td>
<td>Acquired</td>
<td></td>
</tr>
<tr>
<td>Phototransistors</td>
<td>4</td>
<td>Acquired</td>
<td></td>
</tr>
<tr>
<td>HCS12 NanoCore</td>
<td>1</td>
<td>Using Schools</td>
<td></td>
</tr>
<tr>
<td>Resistors - 10kΩ</td>
<td>4</td>
<td>Using Schools</td>
<td></td>
</tr>
<tr>
<td>Resistor - 470Ω</td>
<td>1</td>
<td>Using Schools</td>
<td></td>
</tr>
<tr>
<td>Push Button</td>
<td>1</td>
<td>Acquired</td>
<td></td>
</tr>
<tr>
<td>Power Switch</td>
<td>1</td>
<td>Acquired</td>
<td></td>
</tr>
<tr>
<td>10ft Ethernet Cable</td>
<td>1</td>
<td>Acquired</td>
<td></td>
</tr>
<tr>
<td>Ethernet Connectors</td>
<td>2</td>
<td>Acquired</td>
<td></td>
</tr>
</tbody>
</table>
Chapter 3: Implementation

3.1 Milestone I

Tests for the Phototransistor Module
1. Move objects across a Phototransistor from 0.5cm away and observe a value change displayed on PuTTY.
2. Repeat with 2nd Phototransistor
3. Repeat with 3rd Phototransistor
4. Repeat with 4th Phototransistor

Tests for the WS2811 RGB LED Module
1. Send code that should make the LED0 red and observe.
2. Send code that should make the LED1 blue and observe.
3. Send code that should make the LED2 yellow and observe.
4. Send code that should make the LED3 green and observe.

Milestone I Requirements:

The first step for creating the detection system is to get the individual modules functioning correctly. Since the phototransistor board will need to detect pinewood derby cars as they pass over, the HCS12 will need to be able to read two distinct value ranges from each of the individual phototransistors. This will be done by covering up each individual phototransistor and making sure the values read are nice and distinct. In order to make sure the WS2811 Array module of RGB LEDs functions properly, each of the LEDs will need to be able to turn into any of the four colors red, blue, yellow, green. Figure 2 below shows how the modules interact with each other. Table 7 shows the phototransistor module tests that were conducted, and Figure 4 shows the results of those tests which were captured using PuTTY. Table 8 shows the WS2811 module tests that were conducted, and Figure 5 shows the results of those.
<table>
<thead>
<tr>
<th>Module</th>
<th>Test Name</th>
<th>Input</th>
<th>Expected Output</th>
<th>Actual Output</th>
<th>Results</th>
</tr>
</thead>
<tbody>
<tr>
<td>Phototransistor</td>
<td>PT Sensitivity Test 1</td>
<td>Move objects across a PT from 0.5cm away and observe a value change displayed on PuTTY.</td>
<td>Initial Value &lt; 20 Final Value &gt; 100</td>
<td>198</td>
<td>PASS</td>
</tr>
<tr>
<td>PT Sensitivity Test 2</td>
<td>2nd PT</td>
<td></td>
<td>Initial Value &lt; 20 Final Value &gt; 100</td>
<td>197</td>
<td>PASS</td>
</tr>
<tr>
<td>PT Sensitivity Test 3</td>
<td>3rd PT</td>
<td></td>
<td>Initial Value &lt; 20 Final Value &gt; 100</td>
<td>204</td>
<td>PASS</td>
</tr>
<tr>
<td>PT Sensitivity Test 4</td>
<td>4th PT</td>
<td></td>
<td>Initial Value &lt; 20 Final Value &gt; 100</td>
<td>193</td>
<td>PASS</td>
</tr>
</tbody>
</table>

Figure 2: The Level-1 Diagram for Figure 1.
Table 7: First Module Tests

Figure 4: Phototransistor signals captured four times

Results:
Using PuTTY, the four signals are read while each individual phototransistor was covered up. There is a very distinct high and low value for when the derby car passes over.
<table>
<thead>
<tr>
<th>Module</th>
<th>Test Name</th>
<th>Input</th>
<th>Expected Output</th>
<th>Actual Output</th>
<th>Results</th>
</tr>
</thead>
<tbody>
<tr>
<td>WS2811 RGB LEDs</td>
<td>LED Color Settings Test 1</td>
<td>Send code that should make the LED0 red and observe.</td>
<td>LED0 should be red</td>
<td>RED</td>
<td>PASS</td>
</tr>
<tr>
<td></td>
<td>LED Color Settings Test 2</td>
<td>Send code that should make the LED1 blue and observe.</td>
<td>LED1 should be green</td>
<td>GREEN</td>
<td>PASS</td>
</tr>
<tr>
<td></td>
<td>LED Color Settings Test 3</td>
<td>Send code that should make the LED2 yellow and observe.</td>
<td>LED2 should be blue</td>
<td>BLUE</td>
<td>PASS</td>
</tr>
<tr>
<td></td>
<td>LED Color Settings Test 4</td>
<td>Send code that should make the LED3 green and observe.</td>
<td>LED3 should be yellow</td>
<td>YELLOW</td>
<td>PASS</td>
</tr>
</tbody>
</table>

Table 8: Second Module Tests

Figure 5: Left photo shows passing tests. Right photo shows that the LEDs can change colors.

Results:
The LEDs have no problem changing into the needed colors which means all the tests pass.
3.2 Milestone II

Tests to Verify Correct Module Interactions.

1. Have the photoresistors set off in the order of red, blue, yellow, green and observe the LEDs for that order.
2. Have the photoresistors set off in the order of yellow, green, red, blue and observe the LEDs for that order.
3. Have the photoresistors set off in the order of green, yellow, blue, red and observe the LEDs for that order.
4. Press the reset button to make sure all the LEDs change back to white.

Milestone II Requirements:

After the individual modules are functioning independently, the next step is to combine the subsystems into a single unit. The values read from the Phototransistor Board now need to be transmitted to the HCS12 where it will be processed and converted to a format that will tell the LEDs on the WS2811 what colors to display. Figure 9 shows the results of the combined system tests, and Figure 6 shows the accuracy of the device.

<table>
<thead>
<tr>
<th>Module</th>
<th>Test Name</th>
<th>Input</th>
<th>Expected Output</th>
<th>Actual Output</th>
<th>Results</th>
</tr>
</thead>
<tbody>
<tr>
<td>Combined System</td>
<td>Derby Car Finish Order: red, blue, yellow, green</td>
<td>Blocked Light from Derby Cars activates the input signal</td>
<td>Display: red, blue, yellow, green</td>
<td>red, blue, yellow, green</td>
<td>PASS</td>
</tr>
<tr>
<td>Derby Car Finish Order: yellow, green, red, blue</td>
<td>Blocked Light from Derby Cars activates the input signal</td>
<td>Display: yellow, green, red, blue</td>
<td>yellow, green, red, blue</td>
<td>PASS</td>
<td></td>
</tr>
<tr>
<td>Derby Car Finish Order: green, yellow, blue, red</td>
<td>Blocked Light from Derby Cars activates the input signal</td>
<td>Display: green, yellow, blue, red</td>
<td>green, yellow, blue, red</td>
<td>PASS</td>
<td></td>
</tr>
<tr>
<td>Reset Button changes LEDs to white</td>
<td>Blocked Light from Derby Cars activates the input signal</td>
<td>All LEDs white</td>
<td>All white</td>
<td>PASS</td>
<td></td>
</tr>
</tbody>
</table>

Table 9: LED display tests corresponding to the finish line detection.

Results:
The system functioned exactly as intended as all of the tests passed. The system properly detected the finishing sequence and displayed the correct colors. The reset button changes all of the LEDs back to white, indicating the system is ready for the next race.
3.3 Final Implementation
Tests to verify that the system functions properly.

1. Run cars down track and record finish order, and observe that the proper colors are displayed.

Figure 6: Waveform showing two cars crossing the finish line 1.3ms apart.

Results:
The system functioned exactly as intended. The finishing positions of each lane were recorded accurately. Figure 6 shows that the detection system can capture two distinct cars 1.3ms apart. Since the derby cars will be traveling on average 3m/s, the detection will be accurate to within 2mm.
References


Appendix A: Running the Project

How to use the Pinewood Derby Finish Line Detection System:

- Plug in the power cable to the barrel jack on the control box
- Plug in the black side of the ethernet cable to the control box
- Plug in the other side of the ethernet cable to the sensor board
- Plug in the power connectors from the sensor board to the light bar
- Switch the control box on
  - The lights should light up all white if everything is good
  - If the lights are not white the light bar is not properly aligned
    - Readjust the light bar to be over the sensors and press reset
- Run the race
- Press the reset button to run the next race
  - The lights will go back to white when it is ready for a new race