Pinewood Derby Timer Documentation

Last updated: 3/15/2018
By: John Swensen (jpswensen@gmail.com)

Overview

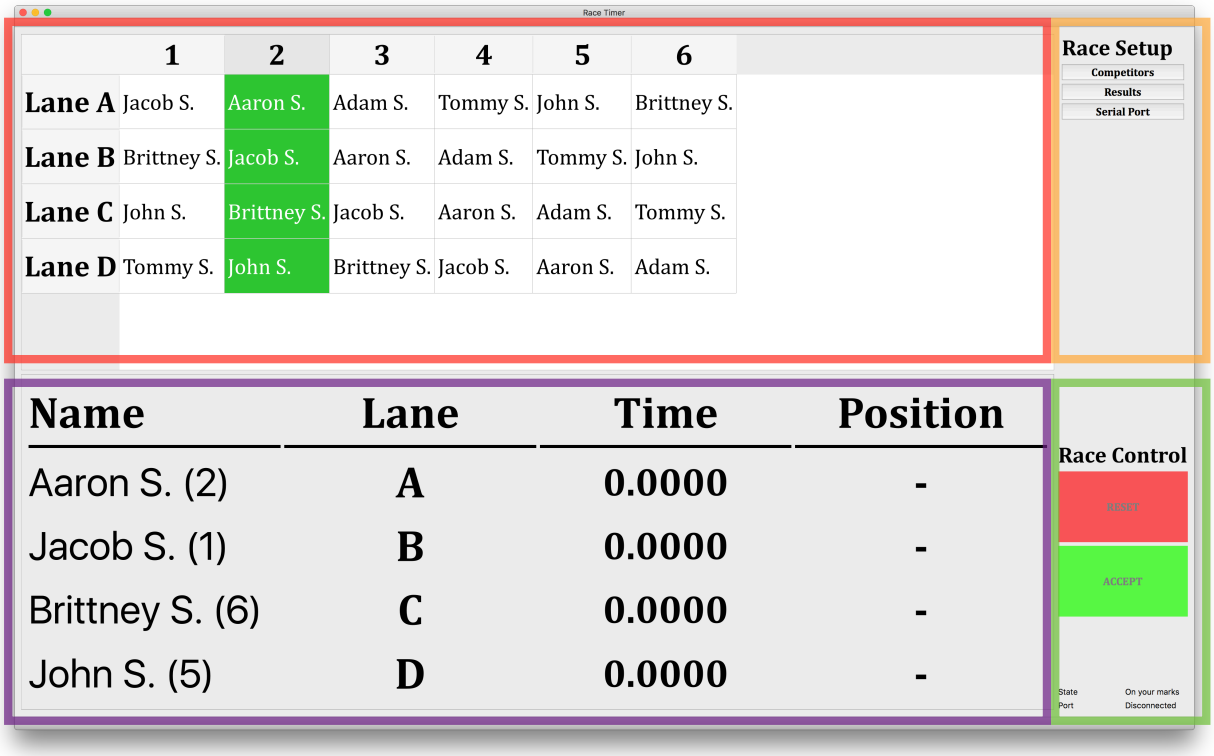
Non-technical overview

This is a very basic pinewood derby timer. Among its simple features, it can do the following:

- load/save race information from/to a comma-separated values (CSV) file
- run or re-run each heat of the race
- display and print the results.

It assumed that the track has four lanes and is based on the principle of every car running on every lane and the winner is that average of the times. It is generally understood that even really good tracks have some lanes that are better than others.

The main interface consists of 4 primary regions as seen in the following screen capture. The upper-left region (highlighted in red) contains a complete list of all the heats for the current set of race participants. The lower-left region (highlighted in purple) contains the details of the current heat. The upper-right region (highlighted in orange) contains buttons for configuring the race participants, opening a window with the complete race results, and for configuring the serial port communications with the timer hardware. The lower-right region (highlighted in green) contains



buttons for controlling the current heat and some informational text about the current state of the timer hardware and the serial port. Each of these components will be explained later.

Technical overview

For the technically inclined, it is an Arduino-compatible ESP8266 microcontroller board (https://www.wemos.cc/product/d1-mini-pro.html) with one pin interrupt for the start gate switch and four pin interrupts for the IR finish line sensors (https://www.amazon.com/gp/product/B01I57HIJ0). The PC timer software is written using the QT toolkit for cross-platform compatibility. It uses only standard QT toolkit libraries, so it should easily compile for MacOS, Windows, and Linux. The repository for the code can be found at https://github.com/jpswensen/SunnysidePWDTimer.

Quick Start

Import/Save from CSV file

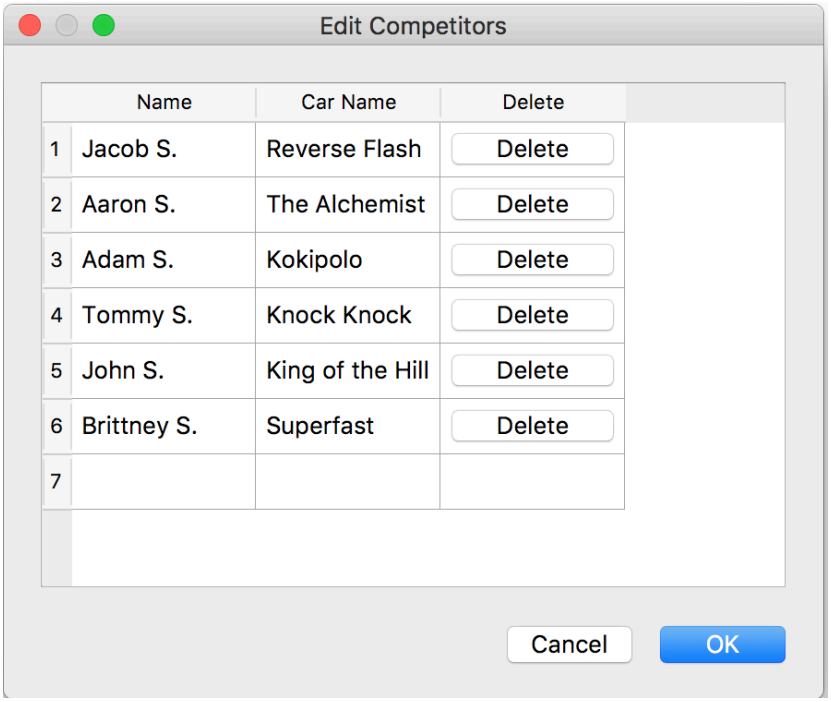
While you can type in the contestants from a Competitors window within the app, many packs have a car check-in the day before. So, the software provides the alternative approach of entering the data into a spreadsheet as saving in CSV form. The import functionality of the timer software requires the follow fields per row: participant name, car name, lane A time, lane B time, lane C time, lane D time.

As the software has still had limited testing, I would recommend saving the race results after every heat is run. In that manner, if there is a crash of the software, then the times from the races that have been completed will not be lost.

Competitors Window

By clicking the Competitors button in the upper-right of the main window, it will open the dialog box shown below that allows the race manager to add and delete participants
WARNING: If the competitors button is used after the race has begun to add/remove a participant, it will clear out all the race time for heats that have already been completed. This is because the addition or removal of a racer will re-arrange how the heats are organized.

Heats Panel of the Main Window



The heats panel is populated by the list of participants, whether from the Competitors window or

	1	2	3	4	5	6
Lane A	Jacob S.	Aaron S.	Adam S.	Tommy S.	John S.	Brittney S.
Lane B	Brittney S.	Jacob S.	Aaron S.	Adam S.	Tommy S.	John S.
Lane C	John S.	Brittney S.	Jacob S.	Aaron S.	Adam S.	Tommy S.
Lane D	Tommy S.	John S.	Brittney S.	Jacob S.	Aaron S.	Adam S.

loaded from a CSV file. With N participants, there will be N heats.

When the race times for a race heat have been accepted, a red square will appear next to the participant names in any heat that is complete. The race administrator must click on a column to change to a different heat. The current heat is highlighted in green, as shown in the previous figure.

Current Heat Panel of the Main Window

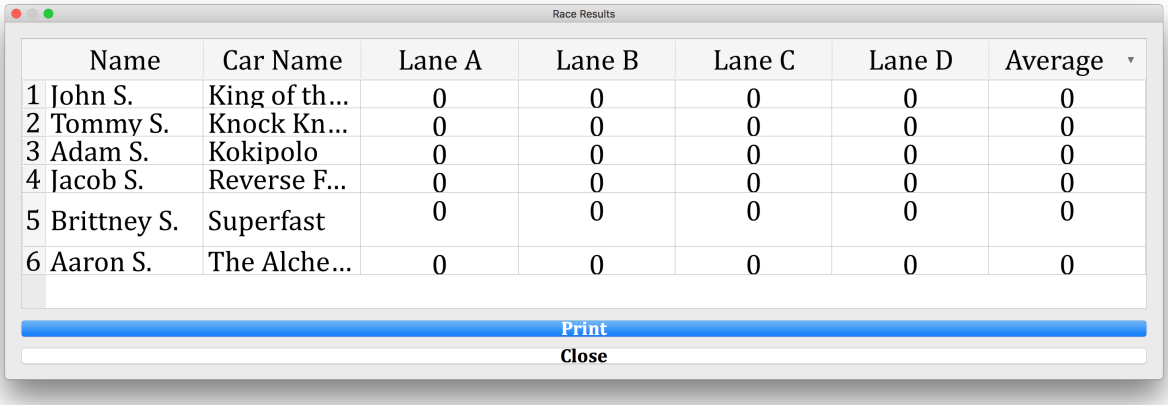
The current heat panel is populated based on which column is selected in the Heats Panel (as described above). This panel not only shows the car number, but also the lane indicator. We recommend having each car place a number on the back of the car during registration that matches the row in the CSV file. Additional put the (A,B,C,D) label on the back of each lane of the track. This will ensure that at the start of each heat it is easy to ensure each car is in the right lane. When the start gate is opened, each participant's time begins to increase equally until each car crossed the finish line and the time for that lane stops. The participant's finish position is also displayed. A screenshot of this lower-left panel is shown below. If the heat was successful, the race administrator pressed the "Accept" button to add the times for the current heat and marks the heat as completed in the Heats Panel.

Results Window

The Results window allows the race administrator to show the final results. This window allows the administrator to sort times by clicking on the header for the row. Some racers may be

Name	Lane	Time	Position
Jacob S. (1)	A	0.0000	-
Brittney S. (6)	B	0.0000	-
John S. (5)	C	0.0000	-
Tommy S. (4)	D	0.0000	-

interested to know if they were the fastest on any individual lane, even if they weren't the overall

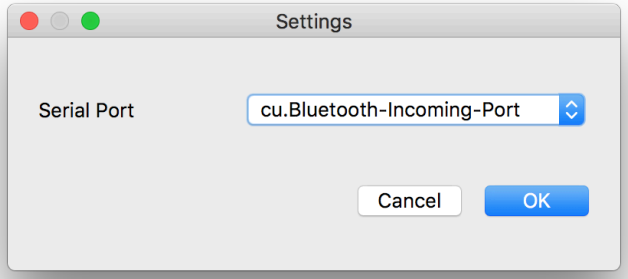


winners.

Serial Port Window

The Serial Port window allows the user to select the serial port used to communicate with the timer. Even though the timer is plugged into your computer using a USB cable, it shows up as a serial port. The easiest way to determine which serial port gets created when the USB cable is

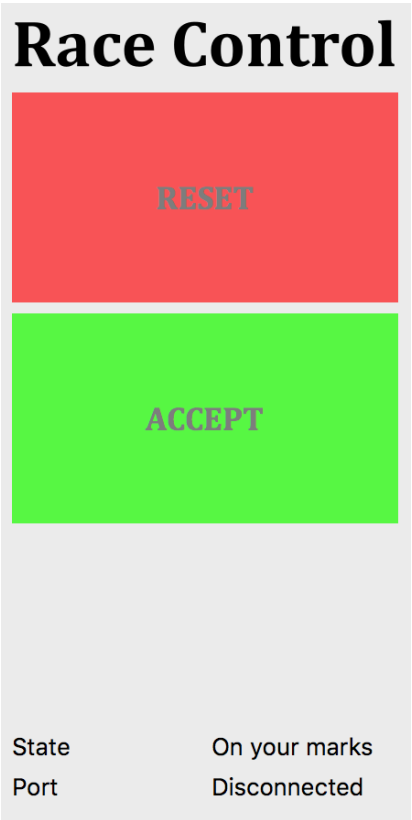
plugged in is to first open the Serial Port window before you plug in the timer. Then after you



plug in the timer, open the Serial Port window again and see which serial port in the list is new.

Race Control Panel

The Race Control panel is used to control the current heat. Due to current communications issues between the timer software and the timer hardware, after the heat has been changed, press the Reset button multiple times until the Time column of the Heat Panel is all zeros



(0.0000) and the Position column is empty (‘-‘).

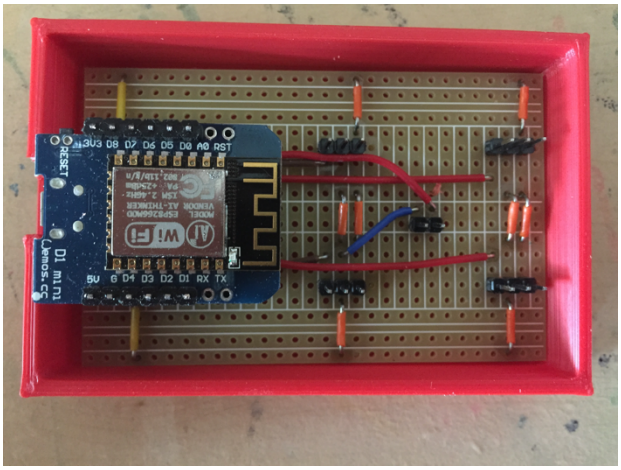
After the Reset button has been pressed, the State text will change to “On your marks (close gate)”. Once the start gate has been closed, the State text will change to “Get set”. The timer is now just waiting for the gate to be opened.

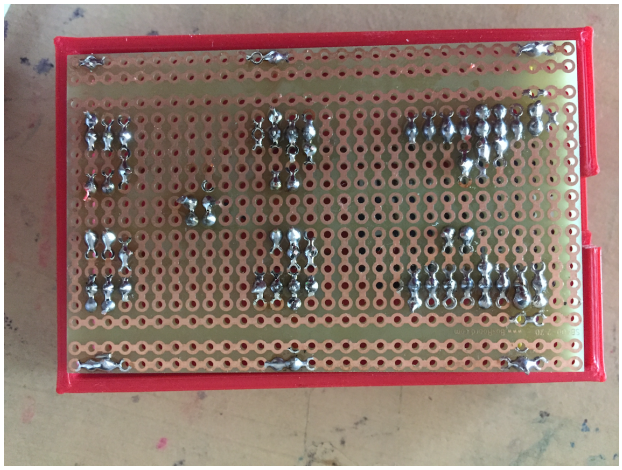
The Accept button will store the results of the current heat and update the Heats Panel.

Hardware Connections

This timer supports 4 tracks and a start gate switch. The start gate switch should be configured in the normally closed configuration (I might make that a settings option at a future time).

I soldered up a quick mainboard for connecting the sensors and the gate switch to my microcontroller (eventually I would like to give a simple EaglePCB design file for a mainboard also).





In this board, the connections should be

Lane D Lane C

Gate Switch

Lane B Lane A

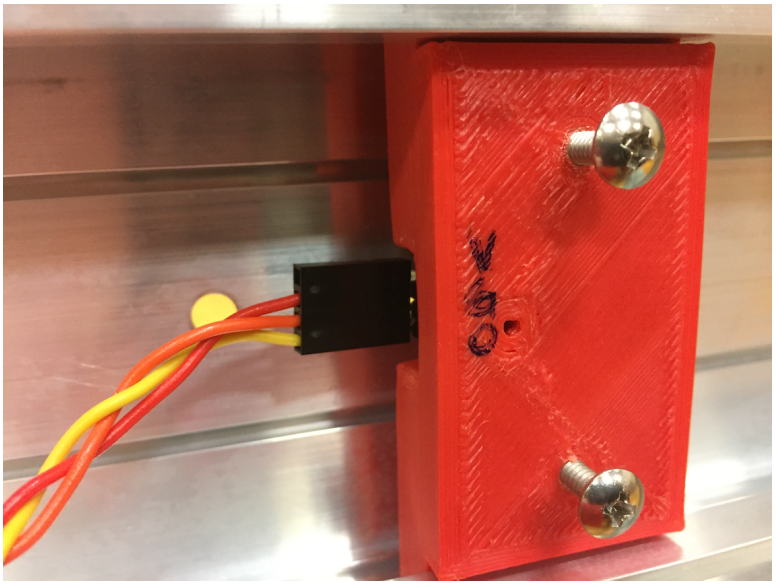
On each sensor header, the order of pins is

Signal, Ground, Power

For the gate switch, the polarity doesn't matter because it either connects or disconnect power from the pin (the Wemos board has a weak pull-down, so no additional resistor is necessary). We use a simple binder clip to hold the gate switch in a position where the switch is pressed when the gate is armed. TODO: Figure out whether the gate is wired normally open (NO) or normally closes (NC). I can't remember.

The end-gate sensors are attached to a 3D printed part that slides into each lane slot on the bottom of the track, as shown below:

For the connection to the sensor, look at the installed sensors to figure out how the cables are supposed to be connected. On our track, we wrote the Power, Ground, and Signal onto the 3D printed part, as shown below:



This image shows the pin ordering on the sensor so that even if you can't see the board within the 3D printed part for the finish line, you can still determine how it should be plugged in.