Lapcounter with Raspberry pi

Paul Stahr

January 19, 2022

1 Hardware

We used the raspberry pi 3 together with the official touchscreen-display. Other configurations were not tested but will work most propably without much configuration. Of course there are many different ways to detect if a car is driving through the start-line. We used a led which collects light which comes from above. Using the slot itself with a light barrier was promising but turned out to be not reliable.

1.1 Components

Item	Amount	Article number	Price
Raspberry PI 4 B 2 GB All-In-Bundle	1	RPI 4B 2GB ALLIN	66,50
7" Touchscreen Display	1	RASPBERRY PI 7TD	64,50
NE 555 DIP	slots	NE 555 DIP	0.17
Resistor 470Ω	slots	K-O RD12JN471T52	0.08
Resistor 10k	slots	K-O RD12JN103T52	0.08
Resistor 100k	slots	K-O RD12JN104T52	0.08
Led V 510 Flat Led Rot	slots	V 510	0.10
Phototransistor BPW 42	slots	BPW 42	0.29
LED 3MM GN LED	1	LED 3MM GN	0.07
Bluetooth keyboard		(1)	
MT3608	1		3.29

Table 1.1: Table of components

1.2 Wiring

To connect the display power, the illumination and the senors we will use the gpio of the raspberry pi. For the connection from the photodiode to the input-circuit we will use usb-connectors. The circuit 1.2 pulls the voltage down for a defined amount of time, and doesn't allow retriggering during this time. In theory it would be possible to connect the raspberry pi directly to the photodiode, but I have not tested it. Also there would be some drawbacks. First you would be dependent on hardware interrupts, because a polling approach could miss the time when the light barrier is interrupted. Secondly this solution is much easier to debug, because you can see if an input was triggered.

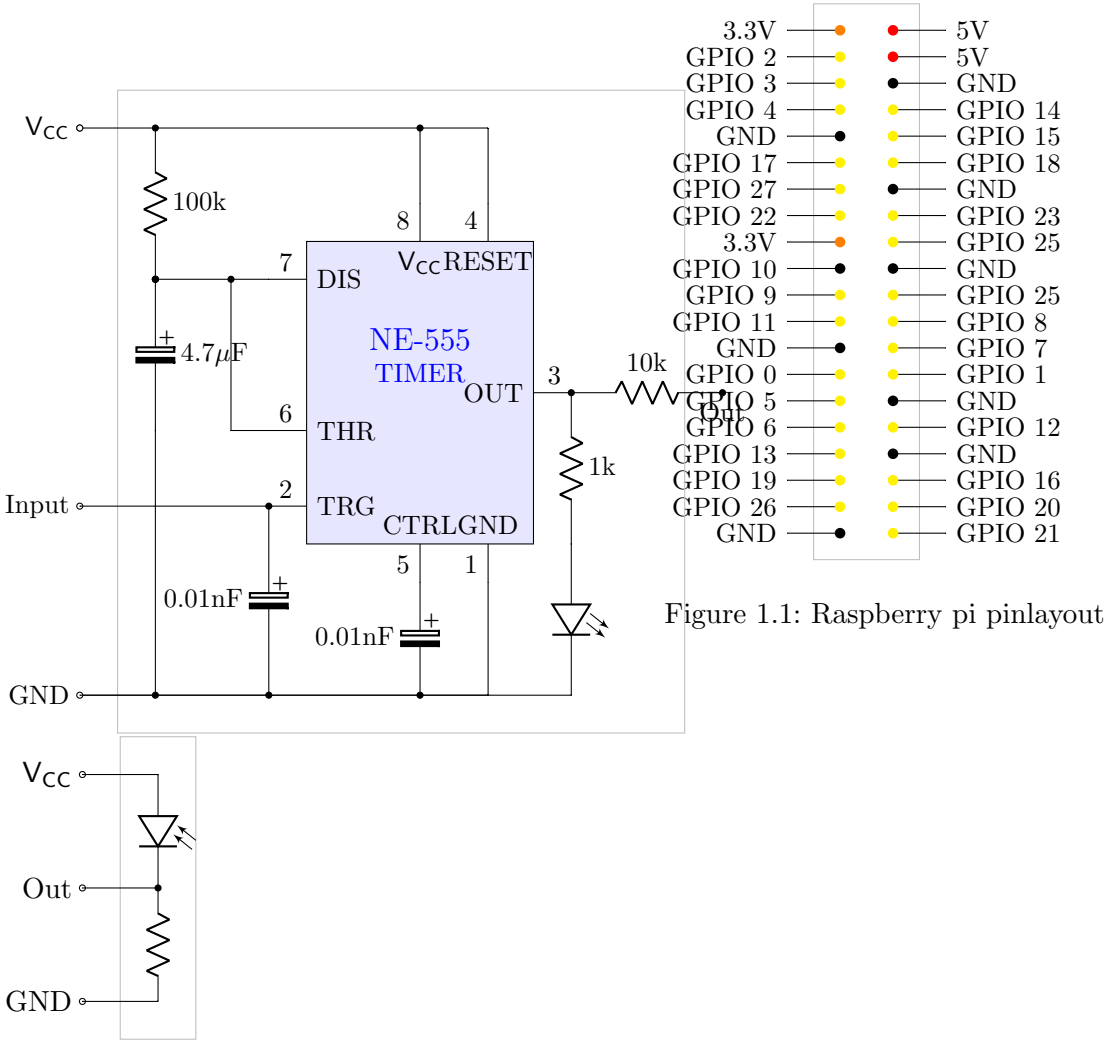
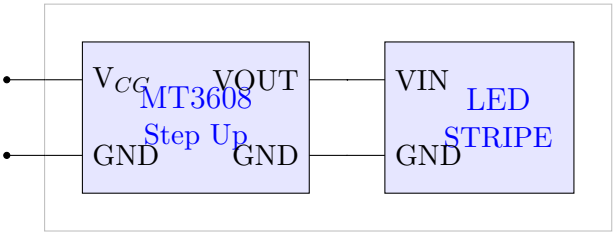


Figure 1.1: Raspberry pi pinlayout

Figure 1.2: Wiring digram

1.2.1 Usb

For the connections to the light barrier and the electronic I used usb connectors, namely the Type-A and Mini-A ones. I adapted the power supply lanes but used the data lanes for the light-barrier-signal.



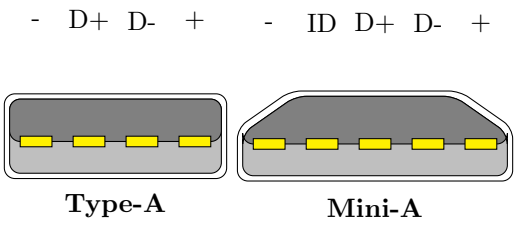


Figure 1.3: USB-Pins

2 Software

2.1 Compiling

The program uses Allegro to show the results. In theory it should be possible to use it on any machine, which supports linux and opengl, I tested it on a x86 platform and on the raspberry pi. If you are using debian, please install the following packages:

Listing 2.1: Required packages

```
sudo aptitude install liballegro5-dev freeglut3-dev libboost-
   thread-dev libboost-system-dev
```

Then build with either

Listing 2.2: Compile

```
mkdir -p build; make -j 4 program
```

Or if you are using the rasperry-pi

Listing 2.3: Compile Pi

```
mkdir -p build; make -j 4 programpi
```

2.2 Controls

There are 4 inputs at which the program listenens to which can generate events which will be handled by the program.

Input	Menu Control	Car Control
Light Barier	no	yes
Mouse	yes	no
Keyboard	yes	yes
Html-Webside	yes	yes
Gamepad	yes	no

2.3 Fast Race

The common mode is the Fast Race. You can select up to 4 players and an arbitrary amount of laps.

2.4 Tournee Planer

The program contains a simple algorithm to calculate plans for tournees. You can adjust the numbers of players, slots, races and equaly hard slots. Keep in mind, that mathematically not all combinations of input can work.

2.5 Settings

2.6 Network access

Connecting to the server will return you a Html-document containing the most important data.