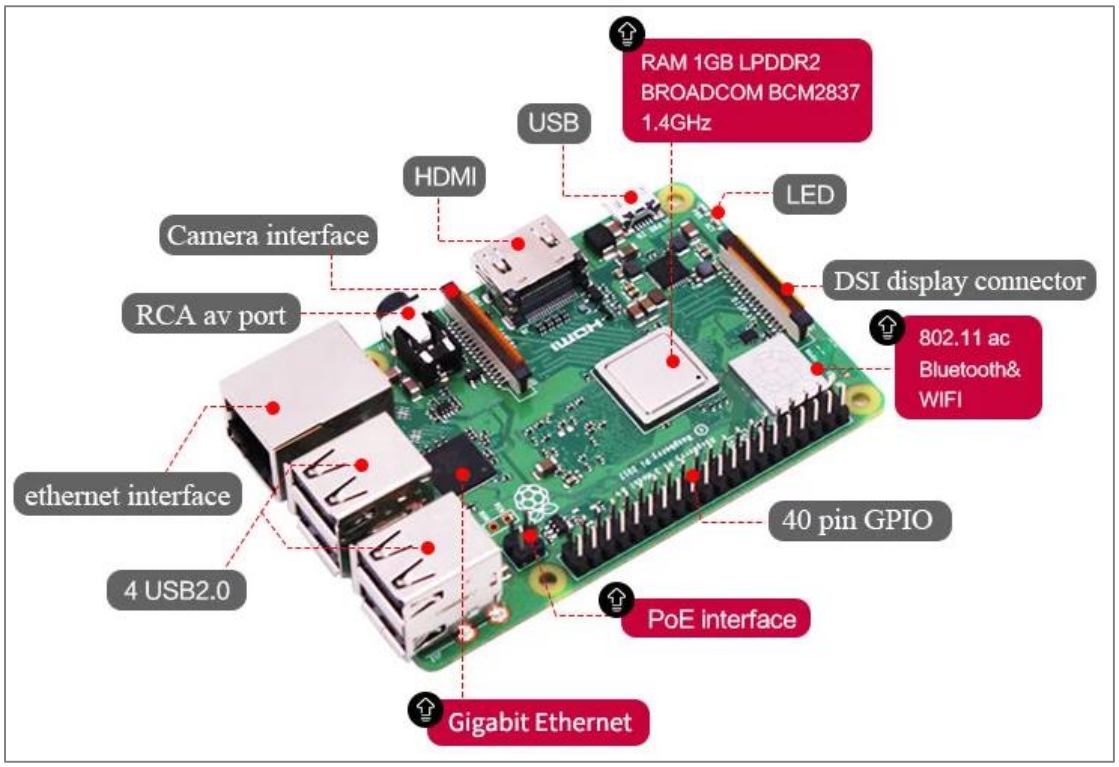
1.Raspberry Pi platform-------whistle

1) Preparation



1-1 Raspberry Pi board

1-2 Buzzer module

2) Purpose of Experimental

After running the whistle executable in the Raspberry Pi system, the car will whistle.

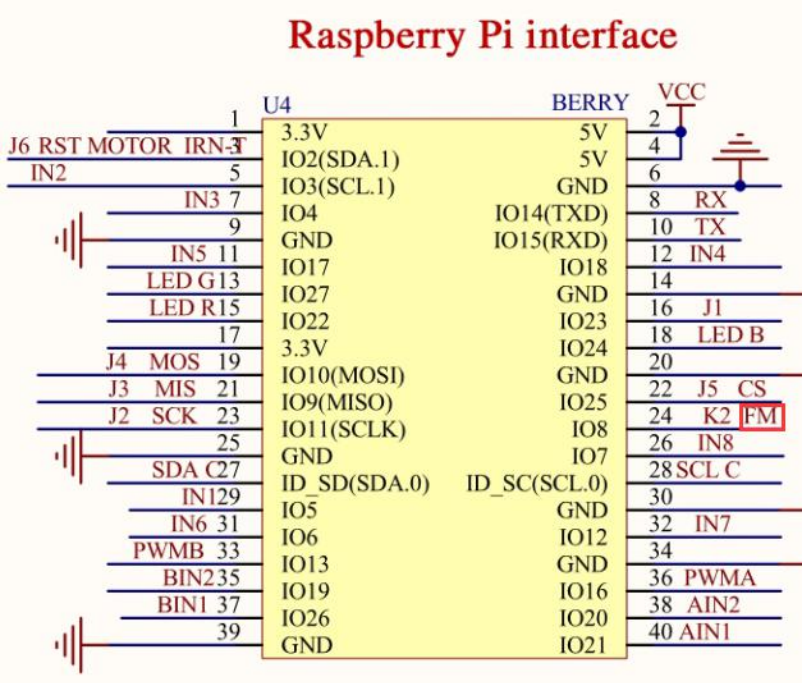
3) Principle of experimental

Buzzers are divided into two types: "active buzzers" and "passive buzzers". Active means that they possess a multi-vibrator inside. It only needs to provide the working voltage externally, it can emit a fixed frequency sound. Passive means that there is no internal oscillation source, and an external drive circuit is required to provide a certain frequency of the drive signal.

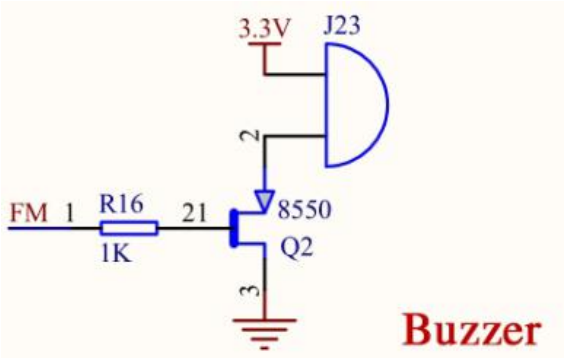
In this experiment, we will use active buzzer.

4) Experimental Steps

4-1 About the schematic



4-1 Raspberry Pi board



4-2 Buzzer module

According to the circuit schematic diagram, positive electrode of the positive buzzer is connected to 3.3V, the negative electrode is connected to 8550 triode, the collector of the triode is connected to GND, the collector of the triode is connected in series with a 1K resistor, and the other end of the resistor is connected to the physical pin 24 of the Raspberry Pi.

wiringPi	BCM	Funtion	Physical pin		Funtion	BCM	wiringPi
		3.3V	1	2	5V		
8	2	SDA.1	3	4	5V		
9	3	SCL.1	5	6	GND		
7	4	GPIO.7	7	8	TXD	14	15
		GND	9	10	RXD	15	16
0	17	GPIO.0	11	12	GPIO.1	18	1
2	27	GPIO.2	13	14	GND		
3	22	GPIO.3	15	16	GPIO.4	23	4
		3.3V	17	18	GPIO.5	24	5
12	10	MOSI	19	20	GND		
13	9	MISO	21	22	GPIO.6	25	6
14	11	SCLK	23	24	CE0	8	10
		GND	25	26	CE1	7	11
30	0	SDA.0	27	28	SCL.0	1	31
21	5	GPIO.21	29	30	GND		
22	6	GPIO.22	31	32	GPIO.26	12	26
23	13	GPIO.23	33	34	GND		
24	19	GPIO.24	35	36	GPIO.27	16	27
25	26	GPIO.25	37	38	GPIO.28	20	28
		GND	39	40	GPIO.29	21	29

4-3 Raspberry Pi 40 pins comparison table

4-2 According to the circuit schematic:

FM-----24(Physical pin)----- 10(wiringPi)

4-3 About the code

Please view .py and.c file

A. For .c code

1) We need to compile this file in the Raspberry Pi system. (Note: we need to add -lwiringPi to the library file.)

We need to input: gcc whistle.c -o whistle -lwiringPi

2) We need to run the compiled executable file in the Raspberry Pi system.We need to input: ./whistle

As shown in the figure below.

```
pi@raspberryPi:~/TrikeBotCar $ gcc whistle.c -o whistle -lwiringPi
pi@raspberryPi:~/TrikeBotCar $ ./whistle
```

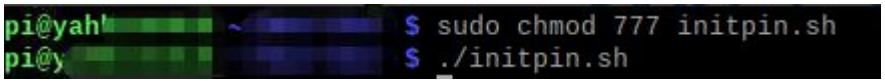
3)We can input: ctrl+c to stop this process, which mean is send a signal to the linux kernel to terminate the current process, but the state of the relevant pin is uncertain at this time, we also need to run a script to initialize all pins.

(Note:The initpin.sh script file is included in the TrikeBotCar/python_code directory.)

You need to input:

sudo chmod 777 initpin.sh

./initpin.sh



B. For python code

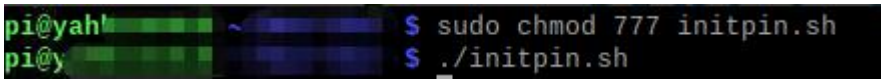
1) We need to input following command to run python code.

python whistle.py

2) We can input: ctrl+c to stop this process, which mean is send a signal to the linux kernel to terminate the current process, but the state of the relevant pin is uncertain at this time, we also need to run a script to initialize all pins.

3) You need to input: sudo chmod 777 initpin.sh

./initpin.sh



After completing the above steps, the experiment is over.

5) Experimental phenomenon

When we run the program, the car will whistle.