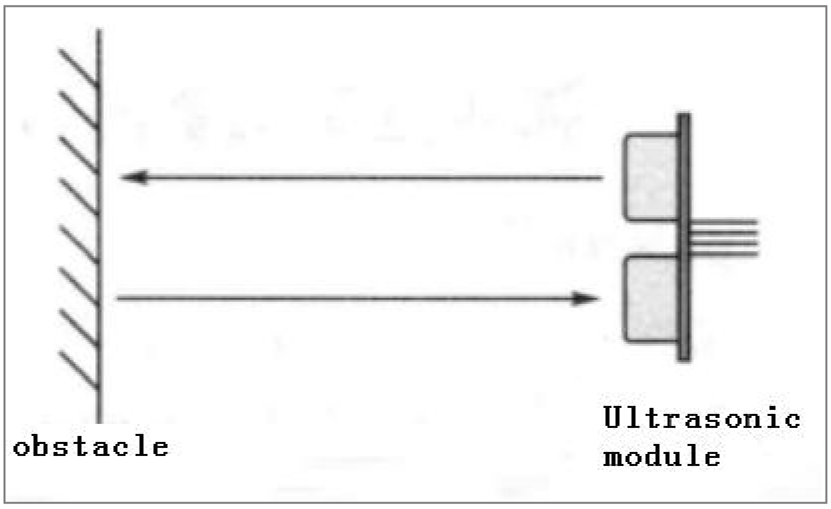
Hardware Control course--Ultrasonic avoid

1. Learning target

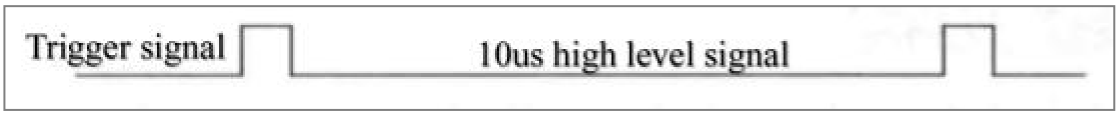
In this course, we will learn how to use ultrasonic module completes obstacle avoidance experiment.

2. Principle of experimental

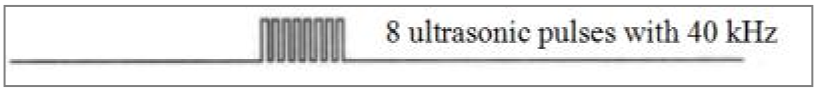
The ultrasonic module is a sensor that uses ultrasonic characteristics to detect the distance. It has two ultrasonic probes for transmitting and receiving ultrasonic waves. The range of measurement is 3-450 cm.



(1) You need to input a high level signal of at least 10us to the Trig pin to trigger the ranging function of the ultrasonic module.



(2) After the ranging function is triggered, the module will automatically send out 8 ultrasonic pulses with 40 kHz and automatically detect whether there is a signal return. This step is done internally by the module.



(3) When the module detects an echo signal, the ECHO pin will output a high level. The high level duration is the time from when the ultrasonic wave is sent to when it returns. You can calculate the distance by using the time function to calculate the high level duration.

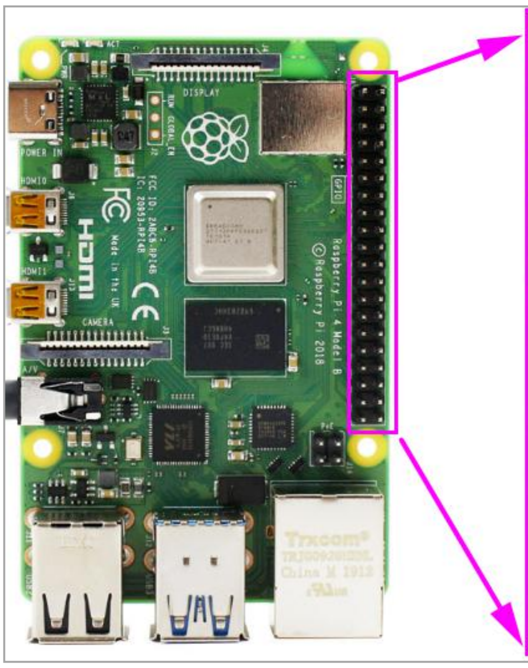
Formula: Distance = High level duration * Speed of sound(340M/S)/2.

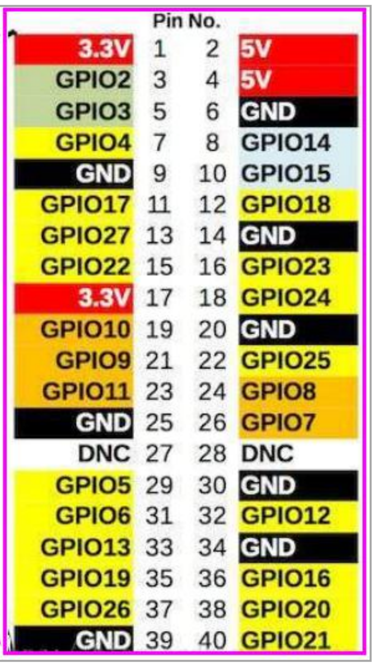
For the Raspbot car, we use 4 TT DC gear motors. They are driven by the TB6612 chip. The driver chip is not directly connected to the Raspberry Pi pins.
Raspberry Pi communicates with STM8 MUC through IIC, and then STM8 MCU drives TB6612 chip to drive the motor.

3. Coding method

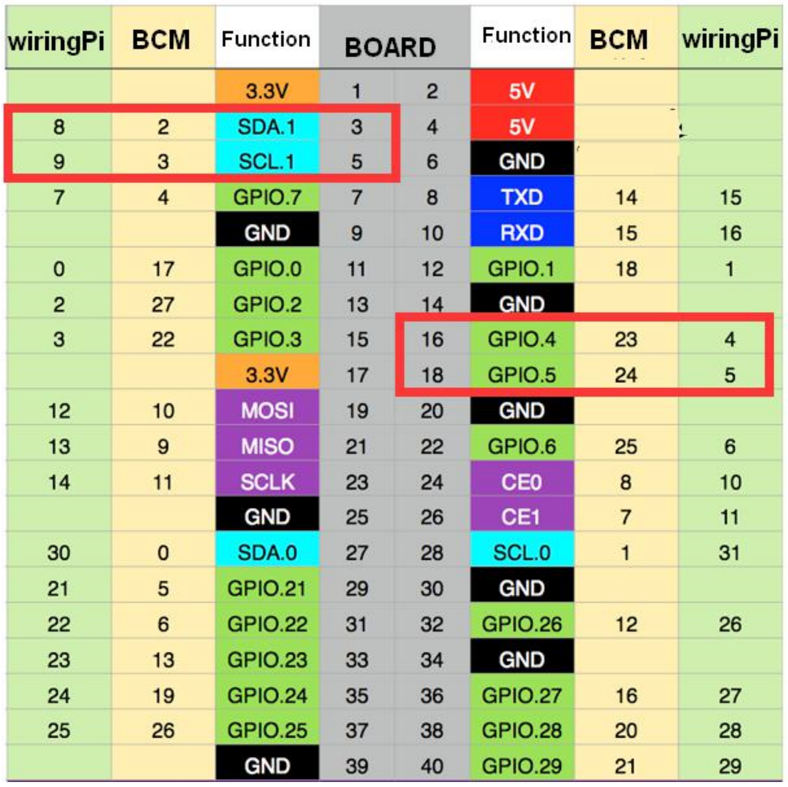
In this course, we use BOARD coding method.
According to the hardware interface manual, we see that the ultrasonic Echo and Trig pins are connected to the 18 and 16 pin of the Raspberry Pi board.
STM8 is connected to SDA.1, SCL.1 on the Raspberry Pi board.
We have provided a library text dedicated to driving motors and servos --YB_Pcb_Car.py.

The pin comparison table of Raspberry Pi as shown below.





	Pin No.		
3.3V	1	2	5V
GPIO2	3	4	5V
GPIO3	5	6	GND
GPIO4	7	8	GPIO14
GND	9	10	GPIO15
GPIO17	11	12	GPIO18
GPIO27	13	14	GND
GPIO22	15	16	GPIO23
3.3V	17	18	GPIO24
GPIO10	19	20	GND
GPIO9	21	22	GPIO25
GPIO11	23	24	GPIO8
GND	25	26	GPIO7
DNC	27	28	DNC
GPIO5	29	30	GND
GPIO6	31	32	GPIO12
GPIO13	33	34	GND
GPIO19	35	36	GPIO16
GPIO26	37	38	GPIO20
GND	39	40	GPIO21

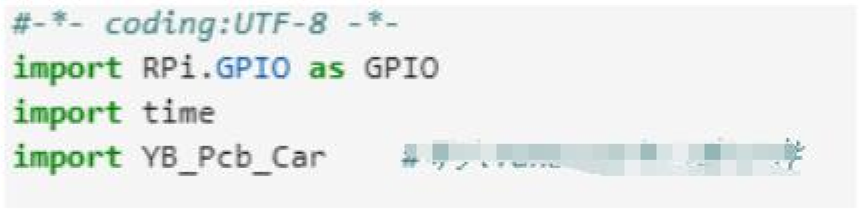


wiringPi	BCM	Function	BOARD		Function	BCM	wiringPi
		3.3V	1	2	5V		
8	2	SDA.1	3	4	5V		
9	3	SCL.1	5	6	GND		
7	4	GPIO.7	7	8	TXD	14	15
		GND	9	10	RXD	15	16
0	17	GPIO.0	11	12	GPIO.1	18	1
2	27	GPIO.2	13	14	GND		
3	22	GPIO.3	15	16	GPIO.4	23	4
		3.3V	17	18	GPIO.5	24	5
12	10	MOSI	19	20	GND		
13	9	MISO	21	22	GPIO.6	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. About code

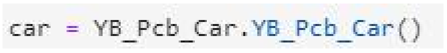
Path: /home/pi/Yahboom_project/Raspbot/2.Hardware Control course/5.Ultrasonic avoid/Ultrasonic avoid.ipynb

1) Import time, GPIO and YB_Pcb_Car library

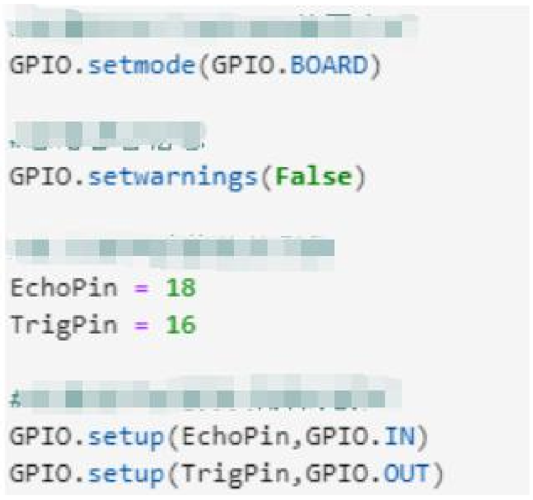


```
#-*- coding:UTF-8 -*-
import RPi.GPIO as GPIO
import time
import YB_Pcb_Car    [redacted]
```

2)Set the GPIO coding mode , define ultrasonic module pin, and define the car class is used to drive motors or servos.



```
car = YB_Pcb_Car.YB_Pcb_Car()
```



```
[redacted]
GPIO.setmode(GPIO.BOARD)

[redacted]
GPIO.setwarnings(False)

[redacted]
EchoPin = 18
TrigPin = 16

[redacted]
GPIO.setup(EchoPin,GPIO.IN)
GPIO.setup(TrigPin,GPIO.OUT)
```

3)Define the ultrasonic ranging function
We need to set a high level signal of at least 10us to the TRIG pin of the ultrasonic module to trigger the module's ranging function.

```python
def Distance():
    GPIO.output(TrigPin,GPIO.LOW)
    time.sleep(0.000002)
    GPIO.output(TrigPin,GPIO.HIGH)
    time.sleep(0.000015)
    GPIO.output(TrigPin,GPIO.LOW)

    t3 = time.time()

    while not GPIO.input(EchoPin):
        t4 = time.time()
        if (t4 - t3) > 0.03 :
            return -1
    t1 = time.time()
    while GPIO.input(EchoPin):
        t5 = time.time()
        if(t5 - t1) > 0.03 :
            return -1

    t2 = time.time()
    time.sleep(0.01)
    #print ("distance_1 is %d " % (((t2 - t1)* 340 / 2) * 100))
    return ((t2 - t1)* 340 / 2) * 100
```

4) Define the ultrasonic ranging optimization program to optimize the test results.

```python
def Distance_test():
    num = 0
    ultrasonic = []
    while num < 5:
            distance = Distance()
            #print("distance is %f"%(distance) )
            while int(distance) == -1 :
                distance = Distance()
                #print("Tdistance is %f"%(distance) )
            while (int(distance) >= 500 or int(distance) == 0) :
                distance = Distance()
                #print("Edistance is %f"%(distance) )
            ultrasonic.append(distance)
            num = num + 1
            #time.sleep(0.01)
    #print ('ultrasonic')
    distance = (ultrasonic[1] + ultrasonic[2] + ultrasonic[3])/3
    #print("distance is %f"%(distance) )
    return distance
```

5) Define the avoid function to complete the ultrasonic obstacle avoidance function of the car.

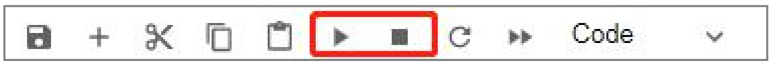
```
def avoid():
    distance = Distance_test()
    if distance < 15 :
        car.Car_Stop()
        time.sleep(0.1)
        car.Car_Spin_Right(100,100)
        time.sleep(0.5)
    else:
        car.Car_Run(100,100)
```

6) Call the obstacle avoidance function in loop to complete the obstacle avoidance experiment.
When we click the stop button, we exit the loop, stop the car, and clear the car class and GPIO.

```
try:
    while True:
        avoid()
except KeyboardInterrupt:
    pass
car.Car_Stop()
del car
print("Ending")
GPIO.cleanup()
```

5. Running code

Click the button shown in the figure below to run the program on the Jupyter Lab interface



6. Experimental phenomenon

After the program runs, we can see that the car will move forward, and it will avoid obstacles when it encounters obstacles in front of it.