Chapter12 超声波测距

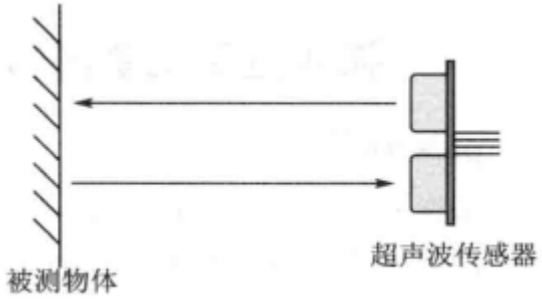
实验目的

远程登录树莓派系统之后，运行程序，按下启动按键K2，启动超声波避障功能，根据小车与障碍物的距离进行转向、直行以及掉头等动作，以避开障碍物前进。

实验原理

测距原理

超声波模块是利用超声波特性检测距离的传感器。其带有两个超声波探头，分别用作发射和接收超声波。

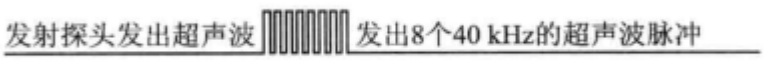


图一 超声波发射和接收示意图

工作原理：当TRIG脚输出至少10us的高电平信号时，触发超声波模块的测距功能。

图二 超声波模块发送触发信号

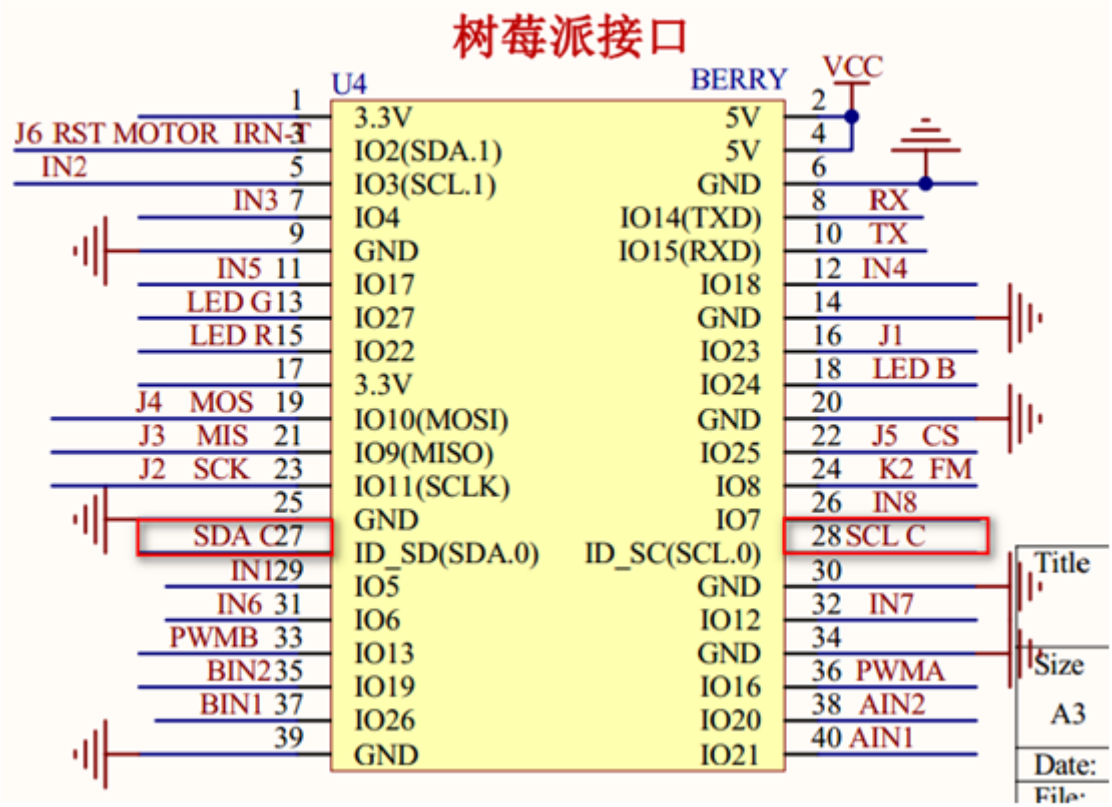
测距功能触发后，模块将自动发出 8 个 40kHz 的超声波脉冲，并自动检测是否有信号返回，这一步由模块内部自动完成。



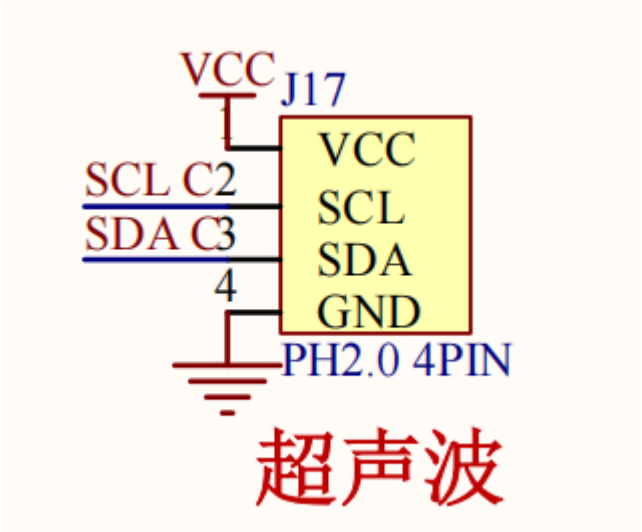
图三 超声波脉冲

一旦检测到有回波信号，ECHO引脚会输出高电平。**高电平持续的时间就是超声波从发射到返回的时间。**此时可以使用时间函数计算出echo引脚高电平的持续时间，即可计算出距被测物体的实际距离。 $$ 距离=高电平时间*声速(340M/S)/2 $$

硬件原理



图四 树莓派接口



图五 超声波模块接口

引脚名称	说　明
Vcc	电源 5 V
Trig	触发引脚
Echo	回馈引脚
Gnd	地

图六 超声波模块引脚

Echo引脚对应SDA接口，Trig引脚对应SCL接口。

代码实现

```
#-*- coding:UTF-8 -*-
import RPi.GPIO as GPIO
import time

#超声波引脚定义
EchoPin = 0
TrigPin = 1

#设置GPIO口为BCM编码方式
GPIO.setmode(GPIO.BCM)

#忽略警告信息
GPIO.setwarnings(False)

#电机引脚初始化为输出模式
#按键引脚初始化为输入模式
#超声波引脚初始化
def init():
    GPIO.setup(key,GPIO.IN)

    GPIO.setup(EchoPin,GPIO.IN)
    GPIO.setup(TrigPin,GPIO.OUT)

#超声波函数
def get_distance():
    GPIO.output(TrigPin,GPIO.HIGH)
    time.sleep(0.000015)
    GPIO.output(TrigPin,GPIO.LOW)

    while not GPIO.input(EchoPin):
        pass
    t1 = time.time()
    while GPIO.input(EchoPin):
        pass
    t2 = time.time()
    print("distance is {} cm".format( ((t2-t1)*340/2)*100 ) )
    time.sleep(0.01)
    return ((t2 - t1)* 340 / 2) * 100

#延时2s
time.sleep(2)

#try/except语句用来检测try语句块中的错误,
#从而让except语句捕获异常信息并处理。
try:
    init()
    key_scan()
    while True:
        distance = get_distance()
        if distance > 50:
```

```
            run(MAX_SPEED, MAX_SPEED)    #当距离障碍物较远时全速前进
        elif 30 <= distance <= 50:
            run(MID_SPEED, MID_SPEED)      #当快靠近障碍物时慢速前进
        elif distance < 30:
            spin_right(SPIN_MAX_SPEED, SPIN_MAX_SPEED)
            time.sleep(0.4)  #当靠近障碍物时原地右转大约90度
            brake()
            time.sleep(0.001)

        distance = get_distance()  #再次测试判断前方距离
        if distance >= 30:
            run(MID_SPEED, MID_SPEED)        #转弯后当前方距离大于25cm时前进
        elif distance < 30:
            spin_left(ZERO_SPEED, SPIN_MAX_SPEED)
            time.sleep(0.7)    #转弯后前方距离小于25cm时向左原地转弯180度
            brake()
            time.sleep(0.001)

        distance = get_distance() #再次测试判断前方距离
        if distance >= 30:
            run(MID_SPEED, MID_SPEED)          #转弯后当前方距离大于25cm时前进
        elif distance < 30:
            spin_left(ZERO_SPEED,SPIN_MAX_SPEED)    #转弯后前方距离小于25cm时向左原地
转弯90度
            time.sleep(0.35)
            brake()
            time.sleep(0.001)

except KeyboardInterrupt:
    pass
pwm_PWMA.stop()
pwm_PWMB.stop()
GPIO.cleanup()
```