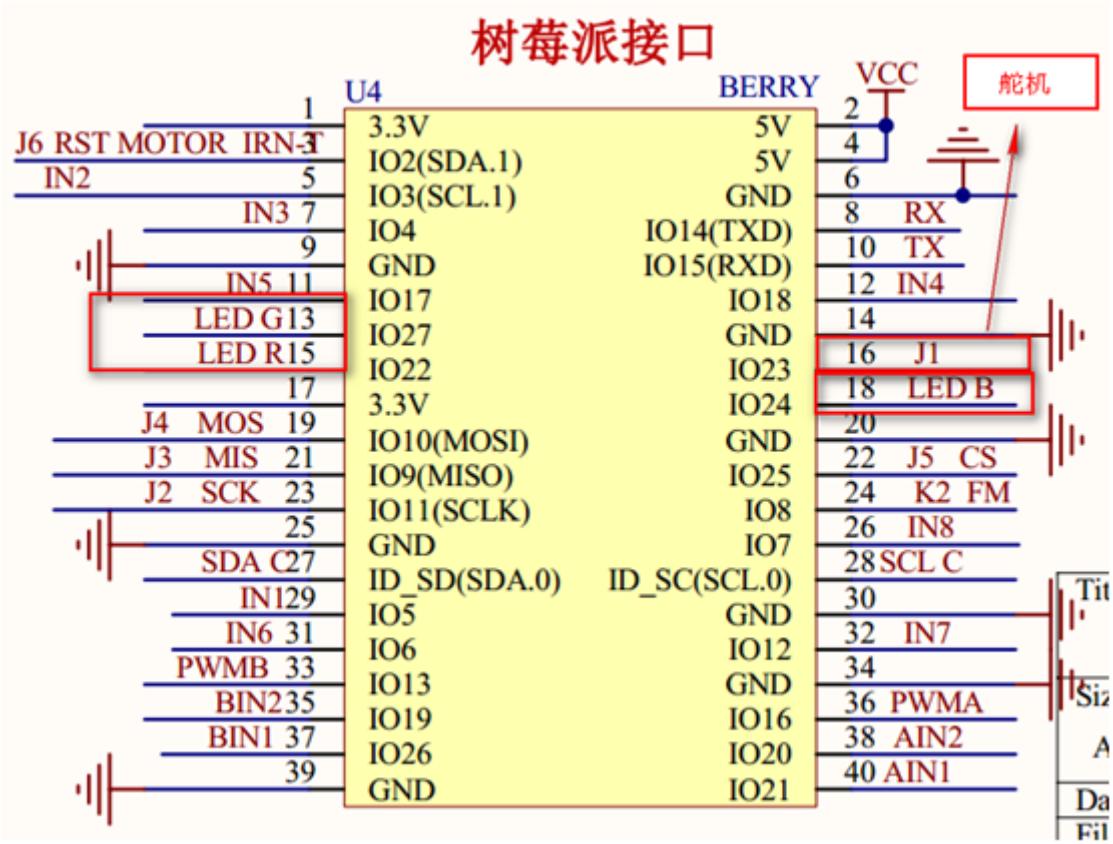
Chapter6 旋转舵机

实验目的

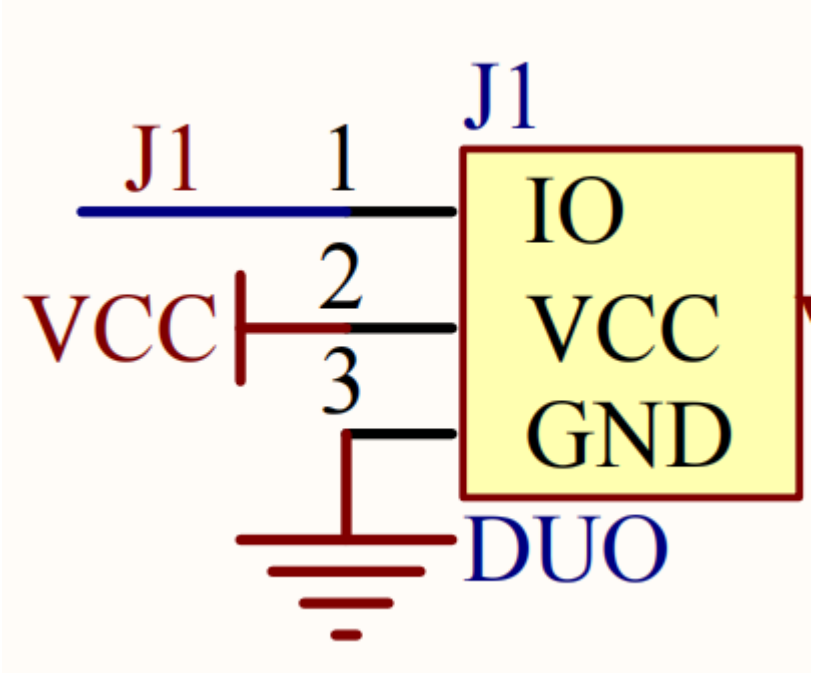
远程登录树莓派系统之后，运行程序，舵机0-180°旋转，LED颜色随舵机转动的角度变换。

实验原理

硬件原理



图一 树莓派接口

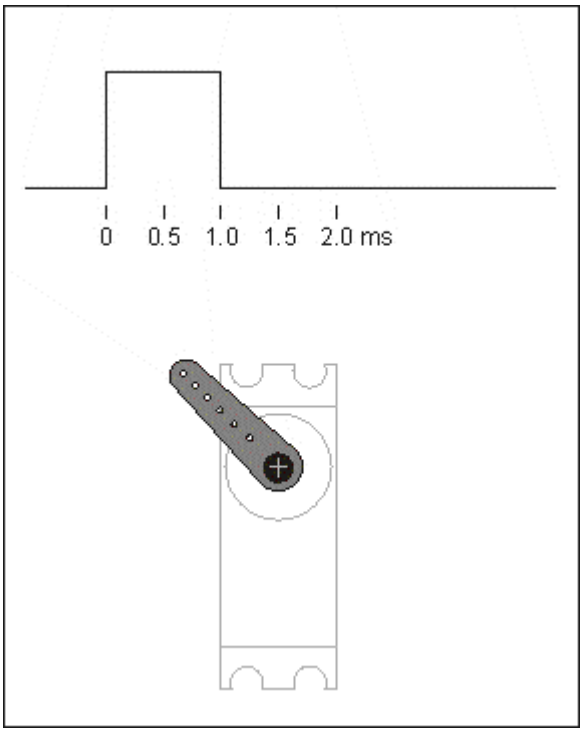


图二 SG90舵机模块接口

舵机转动原理

舵机工作原理：控制信号由接收机的通道进入信号调制芯片，获得直流的偏置电压。它内部有一个基准电路，产生周期为20ms，宽度为1.5ms的基准信号，将获得的直流偏置电压与电位器的电压比较，获得电压差输出。最后电压差的正负输出到电机驱动芯片决定电机的正反转。当电机转速一定时，通过级联减速此轮带动电位器旋转，使得电压差为0，电机停止转动。

舵机的控制：舵机接收的PWM信号频率为50HZ，即周期为20ms。当高电平的脉宽在0.5ms~2.5ms之间时舵机就可以对应旋转到不同的角度。本次实验的采用的舵机是180度转动，高电平部分用0.5ms-2.5ms范围来控制舵机0-180度。



脉冲高电平时间	舵机转动角度
0.5ms	0度
1ms	45度
1.5ms	90度
2ms	135度
2.5ms	180度

代码编写

有两种思路来控制舵机的旋转，简单的方法是直接使用pwm库控制占空比，另一种方法是编写脉冲函数来模拟占空比。

使用pwm库

```
#-*- coding:UTF-8 -*-
#本次舵机转动控制七彩灯控制舵机采用的是系统的pwm库
import RPi.GPIO as GPIO
import time

#RGB三色灯引脚定义
LED_R = 22
LED_G = 27
LED_B = 24

#舵机引脚定义
enginePin = 23

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

#忽略警告信息
GPIO.setwarnings(False)


#舵机引脚设置为输出模式
def init():
    global pwm_engine
    GPIO.setup(LED_R, GPIO.OUT)
    GPIO.setup(LED_G, GPIO.OUT)
    GPIO.setup(LED_B, GPIO.OUT)
    GPIO.setup(enginePin, GPIO.OUT)

   #设置pwm引脚和频率为50hz
    pwm_engine = GPIO.PWM(enginePin, 50)
    pwm_engine.start(0)


#根据转动的角度来点亮相应的颜色
```

```
def color_light(pos):
    #根据角度调节灯光
    pass

#舵机来回转动
def engine_twirl():
    for pos in range(181):
        pwm_engine.ChangeDutyCycle(2.5 + 10 * pos/180)
        color_light(pos)
        time.sleep(0.009)
    for pos in reversed(range(181)):
        pwm_engine.ChangeDutyCycle(2.5 + 10 * pos/180)
        color_light(pos)
        time.sleep(0.009)

#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()
    #舵机初始化向前
    pwm_engine.ChangeDutyCycle(2.5 + 10 * 90/180)
    while True:
        engine_twirl()

except KeyboardInterrupt:
    pass
pwm_engine.stop()
GPIO.cleanup()
```

脉冲函数模拟

```
#-*- coding:UTF-8 -*-
#本次舵机控制七彩灯采用的是自己定义的脉冲函数来
#产生pwm波形
import RPi.GPIO as GPIO
import time

#RGB三色灯引脚定义
LED_R = 22
LED_G = 27
LED_B = 24

#舵机引脚定义
enginePin = 23

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

```
#忽略警告信息
GPIO.setwarnings(False)

#RGB三色灯初始化为输出模式
#舵机引脚设置为输出模式
def init():
    GPIO.setup(LED_R, GPIO.OUT)
    GPIO.setup(LED_G, GPIO.OUT)
    GPIO.setup(LED_B, GPIO.OUT)
    GPIO.setup(enginePin, GPIO.OUT)

#定义一个脉冲函数，用来模拟方式产生pwm值
#时基脉冲为20ms，该脉冲高电平部分在0.5-
#2.5ms控制0-180度
def pulse(myangle):
    pulsewidth = (myangle * 11) + 500
    GPIO.output(enginePin, GPIO.HIGH)
    time.sleep(pulsewidth/1000000.0)
    GPIO.output(enginePin, GPIO.LOW)
    time.sleep(20.0/1000-pulsewidth/1000000.0)


#根据转动的角度来点亮相应的颜色
def color_light(pos):
    #根据角度调节灯光
    pass

#舵机来回转动
def servo_control_color():
    for pos in range(181):
        pulse(pos)
        corlor_light(pos)
        time.sleep(0.009)
    for pos in reversed(range(181)):
        pulse(pos)
        corlor_light(pos)
        time.sleep(0.009)

#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()
    while True:
        servo_control_color()

except KeyboardInterrupt:
    pass
GPIO.cleanup()
```