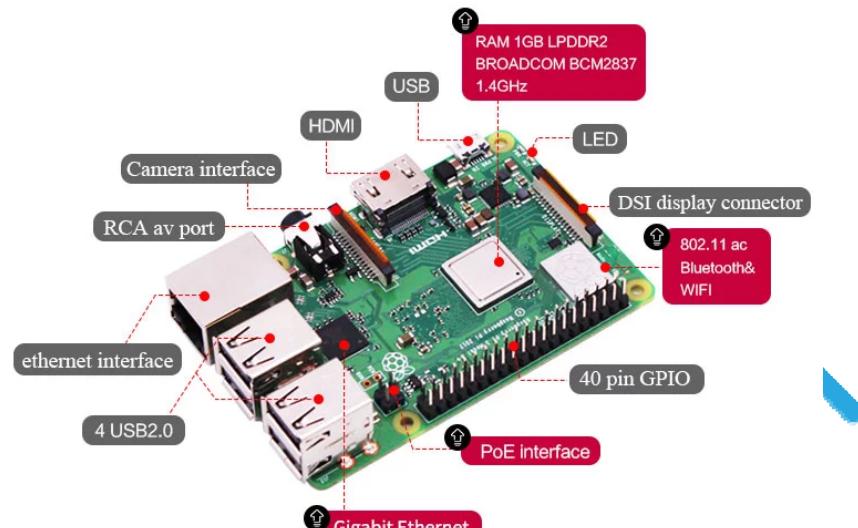


## 4.Raspberry Pi platform ----- ServoControlColor

### 1) Preparation



1-1 Raspberry Pi board



1-2 SG90 servo



1-3 RGB module

### 2) Purpose of Experimental

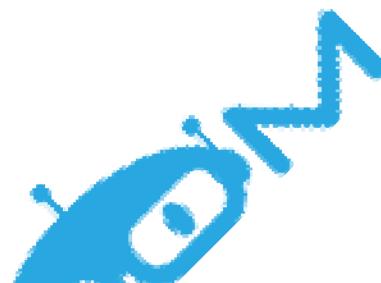
After running the ServoControlColor executable in the Raspberry Pi system. After the car will still for 0.5 s. The servo starts to turn and lights up different colors when turning to different angles.

### 3) Principle of experimental

The working principle of the servo: the control signal enters the signal modulation chip from the channel of the receiver to obtain the bias voltage of the DC. It has a reference circuit inside, which generates a reference signal with a period of 20ms and a width of 1.5ms. It will compares the DC bias voltage with the voltage of the potentiometer to obtain a voltage difference and output. The positive and negative of the voltage difference is outputted to the motor drive chip to determine the forward and reverse of the motor.

Servo rotation angle is by adjusting the duty ratios of PWM (pulse width modulation) signal. The standard PWM (pulse width modulation) signal has a fixed period of 20ms (50Hz). Theoretically, pulse width distribution should be between 1 ms to 2 ms, but in fact between pulse width can be 0.5 ms and 2.5 ms. Pulse width and the servo rotation angle  $0^\circ \sim 180^\circ$  corresponds, as shown below.

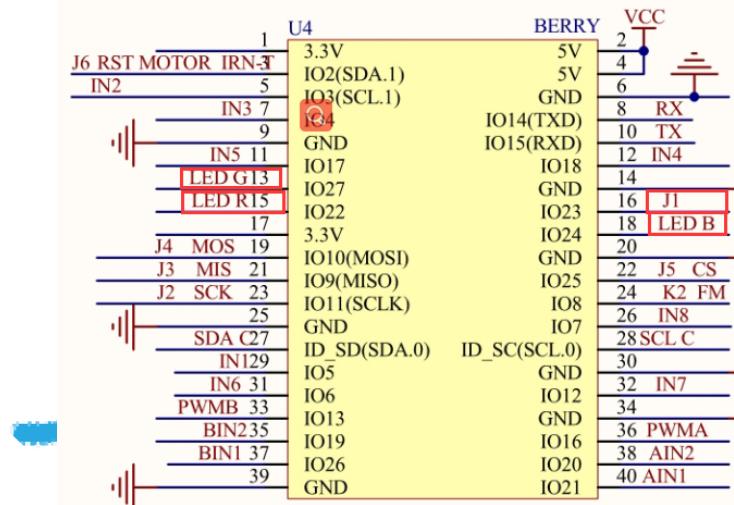
|            |      |
|------------|------|
| 0.5ms----- | 0°   |
| 1.0ms----- | 45°  |
| 1.5ms----- | 90°  |
| 2.0ms----- | 135° |
| 2.5ms----- | 180° |



### 4) Experimental Steps

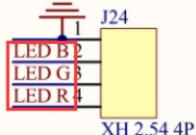
#### 4-1 About the schematic

Raspberry Pi interface



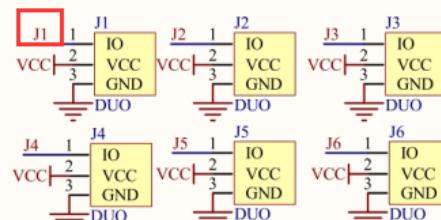
4-1 Raspberry Pi interface circuit diagram

### Strong lighting module



4-2 RGB module interface

### Servo interface



4-3 Servo interface

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

4-4 Raspberry Pi 40 pins comparison table

(Note: SG90 Servo is be connected to steering gear interface J1)

4-2 According to the circuit schematic:

LED\_R----15(Physical pin)---- 3(wiringPi)  
 LED\_G----13(Physical pin)---- 2(wiringPi)  
 LED\_B----18(Physical pin)---- 5(wiringPi)  
 J1---16(Physical pin)----4(wiringPi)

(Note: We use the wiringPi library to write code.)

4-3 About the code

(Note: A software PWM library in the wiringPi is required for this experimental.)

(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 ServoControlColor.c -o ServoControlColor -lwiringPi`

```
root@raspberrypi:/home/pi/SmartCar# gcc ServoControlColor.c -o ServoControlColor  
or -lwiringPi
```

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

As shown in the figure below.

```
root@raspberrypi:/home/pi/SmartCar# ./ServoControlColor
```

(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 SmartCar directory.)

You need to input: `chmod 777 initpin.sh`

`./initpin.sh`

```
root@raspberrypi:/home/pi/SmartCar# chmod 777 initpin.sh  
root@raspberrypi:/home/pi/SmartCar# ./initpin.sh  
root@raspberrypi:/home/pi/SmartCar#
```

After completing the above steps, the experiment is over.

