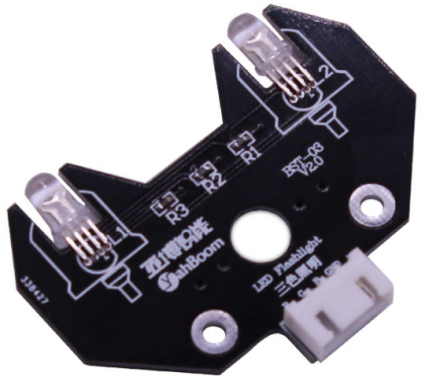
**1.Raspberry Pi platform-------Color\_LED**

1. **Preparation**



1-1 Raspberry Pi board



1-2 RGB module

1. **Purpose of Experimental**

After running the Color\_LED executable in the Raspberry Pi system and you can see the lights in 7 different colors.

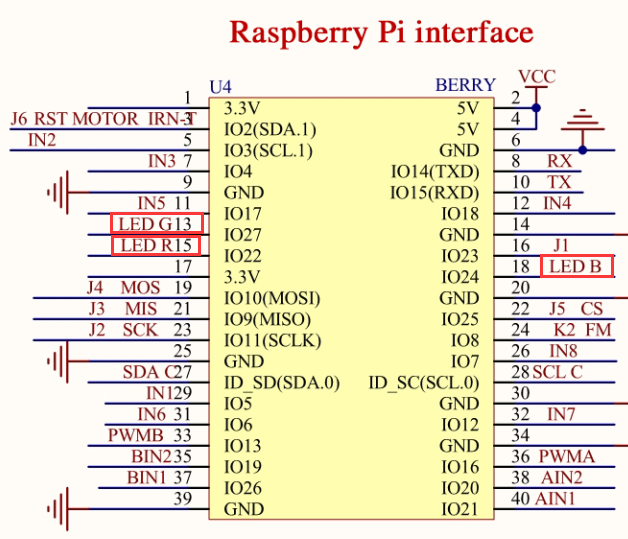
1. **Principle of experimental**

3 LEDs (red, green, blue) are packaged in the RGB lamp module. We can mix different colors(256\*256\*256) by controlling the brightness of the three LEDs.

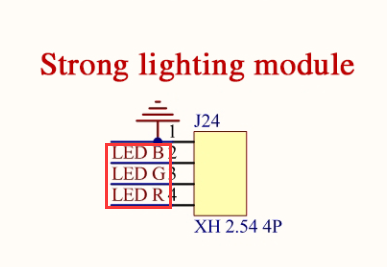
According to the circuit schematic, the RGB lamp is a common cathode LED, one pin is connect to GND, and the remaining three pins are respectively connected to the wiringPi port 3, 2, 5 on the Raspberry Pi board. Each LED needs to be connected in series with a 220Ω resistor as the current limiting resistor. We can control the LED by controlling the corresponding pin to be high level of Raspberry Pi board.

1. **Experimental Steps**

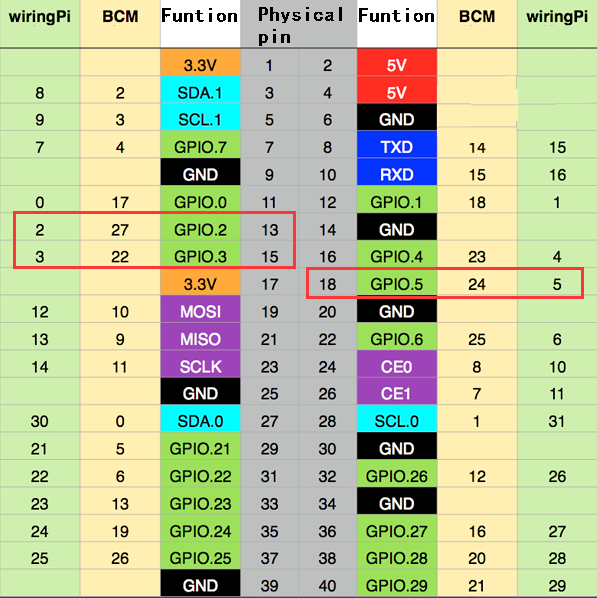
4-1 About the schematic



4-1 Raspberry Pi interface circuit diagram



4-2 RGB module interface circuit diagram



4-3 Raspberry Pi 40 pins comparison table

4-2 According to the circuit schematic:

LED\_R-----15(Physical pin)----- 3(wiringPi)

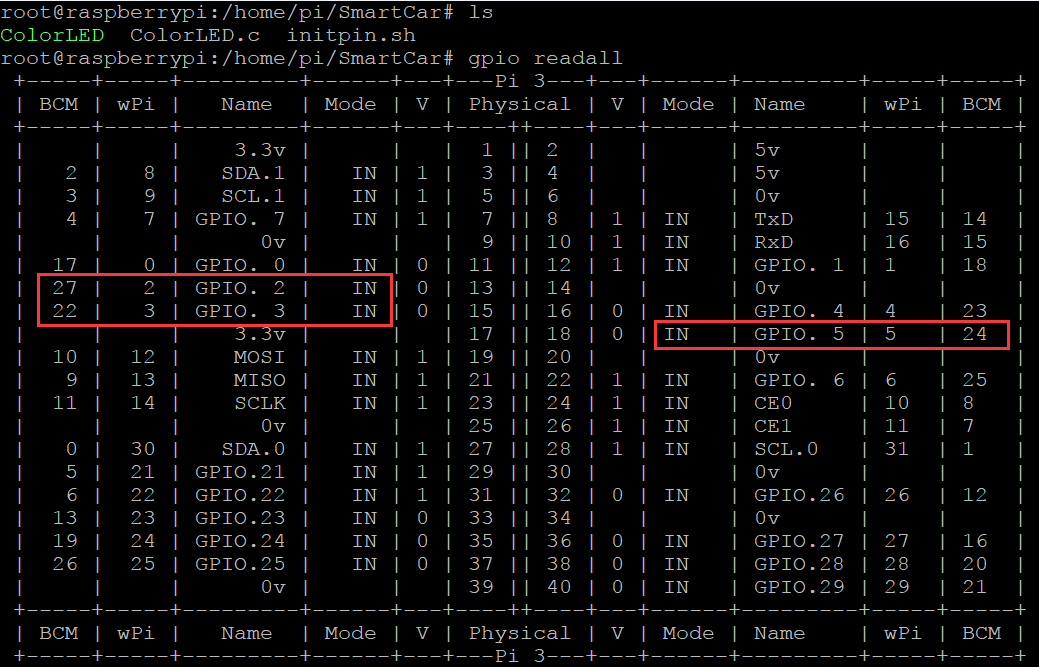
LED\_G-----13(Physical pin)----- 2(wiringPi)

LED\_B-----18(Physical pin)----- 5(wiringPi)

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

4-3 About the code

(1) Before compiling the code, we can see the mode and level state changes of the pins by inputting gpio readall. As shown in the figure below.

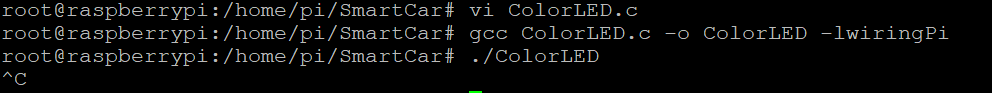


1. We need to compile this file in the Raspberry Pi system. (Note: we need to add -lwiringPi to the library file.)

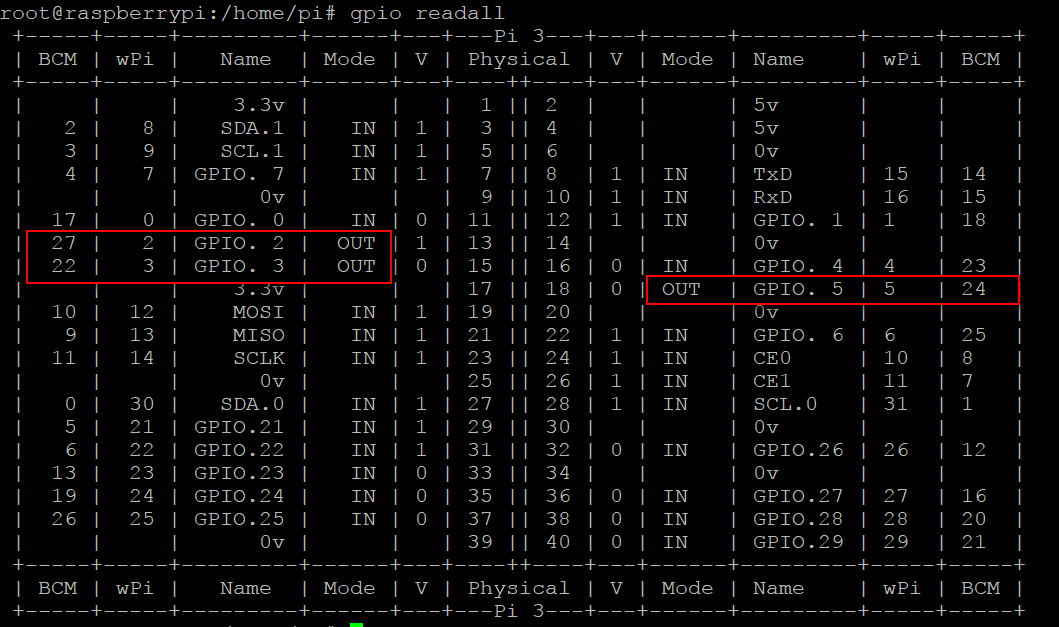
We need to input:gcc ColorLED.c -o ColorLED -lwiringPi

(3)We need to run the compiled executable file in the Raspberry Pi system.We need to input: ./ColorLED

As shown in the figure below.



(4)We can input: gpio readall to see the mode and level state changes of the pins. You will find that the level and mode of the corresponding pin has changed.

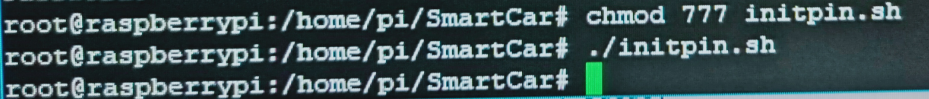


1. We can input: ctrl+c to stop this process, which mean is send a signal to the linux kernel to terminate the current process, but the state of the relevant pin is uncertain at this time, we also need to run a script to initialize all pins.

(Note:The initpin.sh script file is included in the SmartCar directory.)

You need to input:  chmod 777 initpin.sh

./initpin.sh



After completing the above steps, the experiment is over.