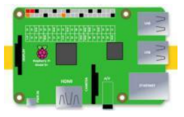
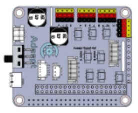
# Lesson 8 How to Control WS2812 LED

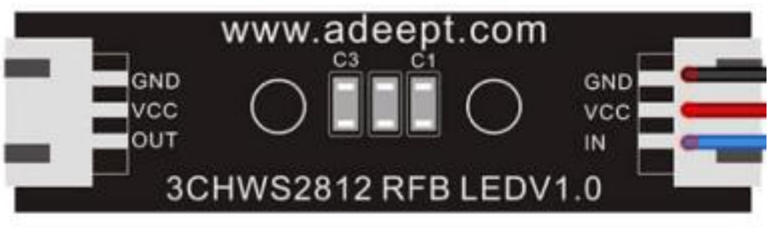
In this lesson, we will learn how to control WS2812 LED.

## 8.1 Components used in this course

| Components | Quantity | Picture |
|---|---|---|
| Raspberry Pi | 1 |  |
| Robot HAT | 1 |  |
| 3 pin wire | 1 | |
| WS2812 RGB LED | 1 | www.adeept.com<br>GND<br>VCC<br>OUT<br>GND<br>VCC<br>IN<br>3CHWS2812 RFB LEDV1.0 |

## 8.2 Introduction of WS2812 RGB LED

WS2812 RGB module is a low-power RGB tri-color lamp with integrated current control chip. Its appearance is the same as a 5050LED lamp bead, and each element is a pixel. The pixel contains an intelligent digital interface data latch signal shaping amplifier driving circuit, and also contains a high-precision internal oscillator and a 12V high-voltage programmable constant current control part, which effectively guarantees that the color of the pixel light is highly consistent.



WS2812 LED is a very commonly used module on our robot products. There are three WS2812 LEDS on each module. Pay attention to the direction of the signal line when connecting. The signal line needs to be connected to the “IN” port of WS2812 LED after being led from the Raspberry Pi. When the next WS2812 LED needs to be connected, we connect a signal wire drawn from the "OUT" port of the previous WS2812 LED with the "IN" port of the next WS2812 LED.

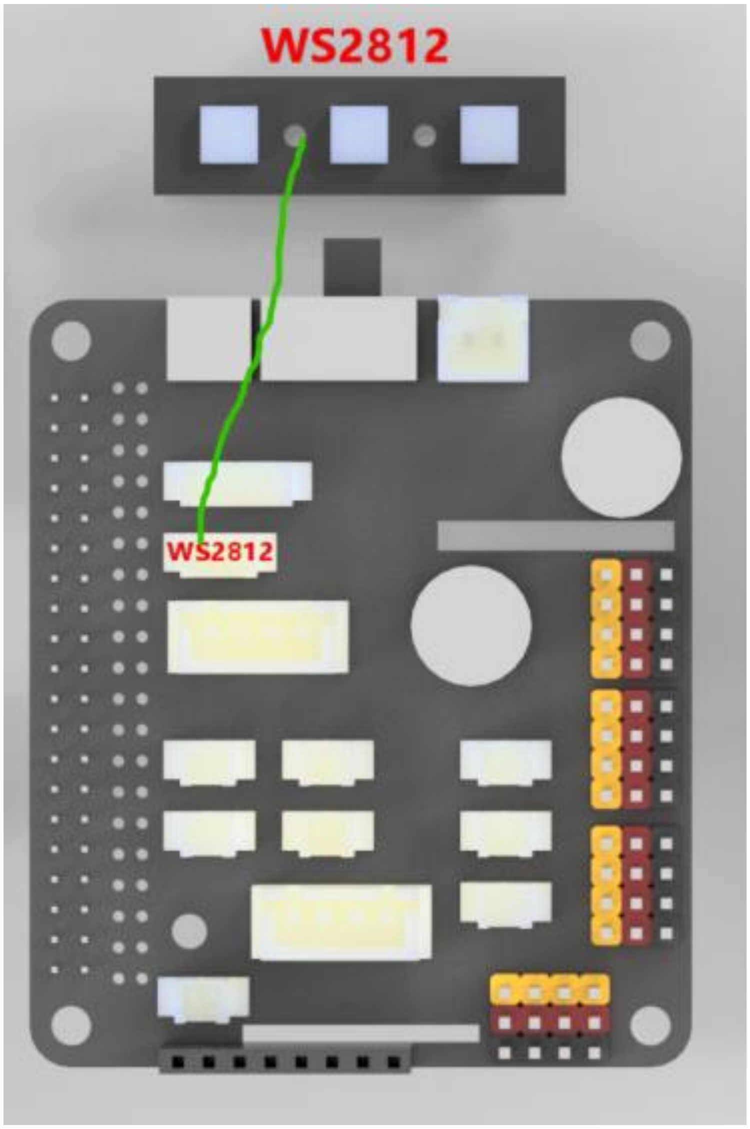
When using the Raspberry Pi to install the driver board RobotHAT, the WS2812 LED can be connected to the WS2812 interface on the RobotHAT using a 3pin cable.

We use a third-party library [rpi_ws281x] to control the WS2812 LED. You can learn about it via https://github.com/richardghirst/rpi_ws281x.

If you connect the WS2812 LED to the WS2812 interface of RobotHAT, the signal line is equivalent to connecting to the GPIO 12 of the Raspberry Pi.

## 8.3 Wiring diagram (Circuit diagram)

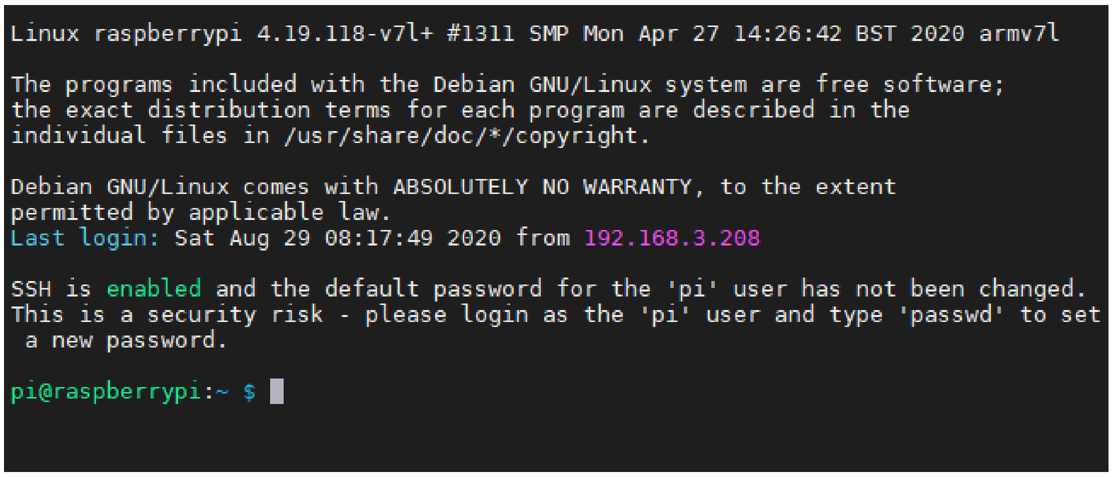
When the WS2812 LED is in use, the IN port needs to be connected to the WS2812 port on the RobotHAT driver board, as shown in the figure below:



## 8.4 How to control WS2812 LED

### 8.4.1 Run the code

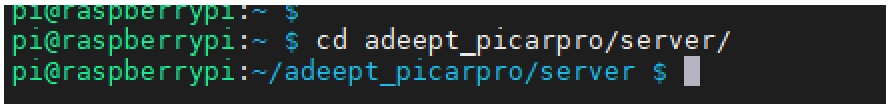
1. Remotely log in to the Raspberry Pi terminal.



```
Linux raspberrypi 4.19.118-v7l+ #1311 SMP Mon Apr 27 14:26:42 BST 2020 armv7l

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Sat Aug 29 08:17:49 2020 from 192.168.3.208

SSH is enabled and the default password for the 'pi' user has not been changed.
This is a security risk - please login as the 'pi' user and type 'passwd' to set
 a new password.

pi@raspberrypi:~ $
```

2. Enter the command and press Enter to enter the folder where the program is located:

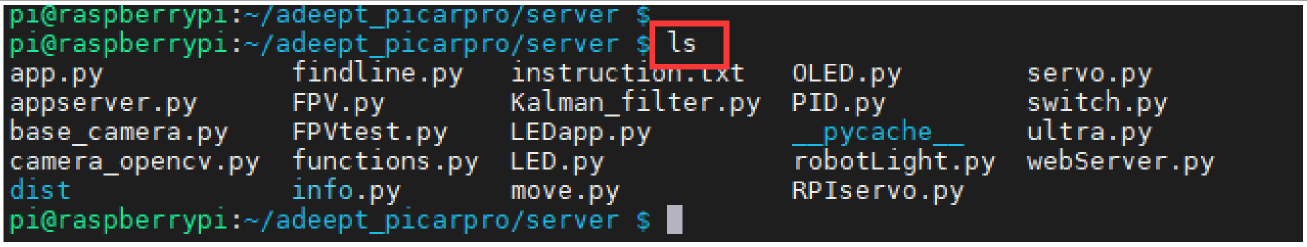
**cd adeept_picarpro/serve**r/



```
pi@raspberrypi:~ $
pi@raspberrypi:~ $ cd adeept_picarpro/server/
pi@raspberrypi:~/adeept_picarpro/server $
```

3. View the contents of the current directory file:

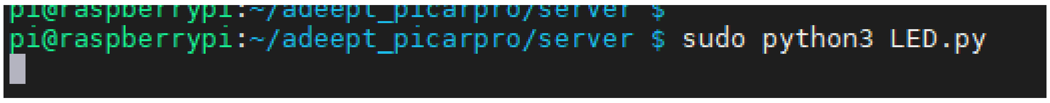
**ls**



```
pi@raspberrypi:~/adeept_picarpro/server $
pi@raspberrypi:~/adeept_picarpro/server $ ls
app.py            findline.py    instruction.txt   OLED.py         servo.py
appserver.py      FPV.py         Kalman_filter.py  PID.py          switch.py
base_camera.py    FPVtest.py     LEDapp.py         __pycache__     ultra.py
camera_opencv.py  functions.py   LED.py            robotLight.py   webServer.py
dist              info.py        move.py           RPIservo.py
pi@raspberrypi:~/adeept_picarpro/server $
```

4. Enter the command and press Enter to run the program:

**sudo python3 LED.py**



```
pi@raspberrypi:~/adeept_picarpro/server $
pi@raspberrypi:~/adeept_picarpro/server $ sudo python3 LED.py
```

5. After running the program successfully, you will observe that the WS2812 alternately flashing red, green and blue lights.

6. When you want to terminate the running program, you can press the shortcut key "Ctrl + C" on the keyboard.

## 8.5 Main code program

For the complete code, please refer to the file LED.py.

```
import time
from rpi_ws281x import *
```

Import dependency.

```
class LED:
        def __init__(self):
            self.LED_COUNT      = 16 #  Set to the total number of LED lights on the robot product, which can be more than the total number of LED lights connected to the Raspberry Pi
            self.LED_PIN        = 12 # Set as the input pin number of the LED lamp group
            self.LED_FREQ_HZ    = 800000
            self.LED_DMA        = 10
            self.LED_BRIGHTNESS = 255
            self.LED_INVERT     = False
            self.LED_CHANNEL    = 0

    # Use the configuration items above to create a strip
            self.strip = Adafruit_NeoPixel(
        self.LED_COUNT,
        self.LED_PIN,
        self.LED_FREQ_HZ,
        self.LED_DMA,
        self.LED_INVERT,
        self.LED_BRIGHTNESS,
        self.LED_CHANNEL
        )
    self.strip.begin()

    def colorWipe(self, R, G, B): # This function is used to change the color of the LED
        color = Color(R, G, B)
        for i in range(self.strip.numPixels()): # Only one LED light color can be set at a time, so a cycle is required
            self.strip.setPixelColor(i, color)
            self.strip.show() # After calling the show method, the color will really change
```

Build LED control class.

```
if __name__ == '__main__':
    led= LED()
    try:
        while 1:
            led.colorWipe(255, 0, 0)  #All lights turn red
            time.sleep(1)
            led.colorWipe(0, 255, 0)  # All lights turn green
            time.sleep(1)
            led.colorWipe(0, 0, 255)  # All lights turn blue
            time.sleep(1)
    except:
        LED.colorWipe(0,0,0)  #Turn off all lights
```

The above code will control all the WS2812 lights to switch among the three colors, press CTRL+C to exit the program.

If you want to control the color of a single light, you can use the following code to achieve, where i is the serial number of the light, the serial number of the first light connected to the signal line from the driver board is 0, and the serial number of the second light is 1. , And so on, R, G, B are the brightness corresponding to the three color channels:

```
LED.strip.setPixelColor(i, Color(R, G, B))
LED.strip.show()
```

Note: You must use the Color() method to pack the RGB value, and then pass it to setPixelColor().