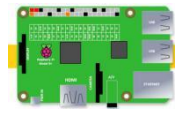
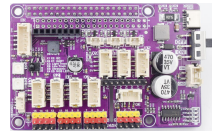




Lesson 10 How to Control WS2812 LED

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

10.1 Components used in this Course

Components	Quantity	Picture
Raspberry Pi	1	
Aadept Robot HAT V3.1	1	
3 pin cable	1	
WS2812 RGB LED	1	

10.2 Introduction of WS2812 LED

WS2812 LED 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.**

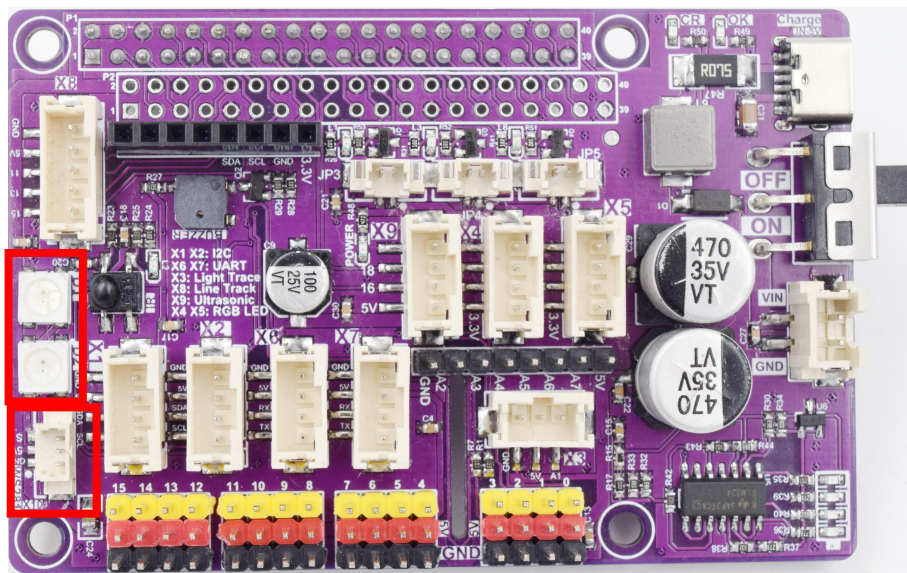
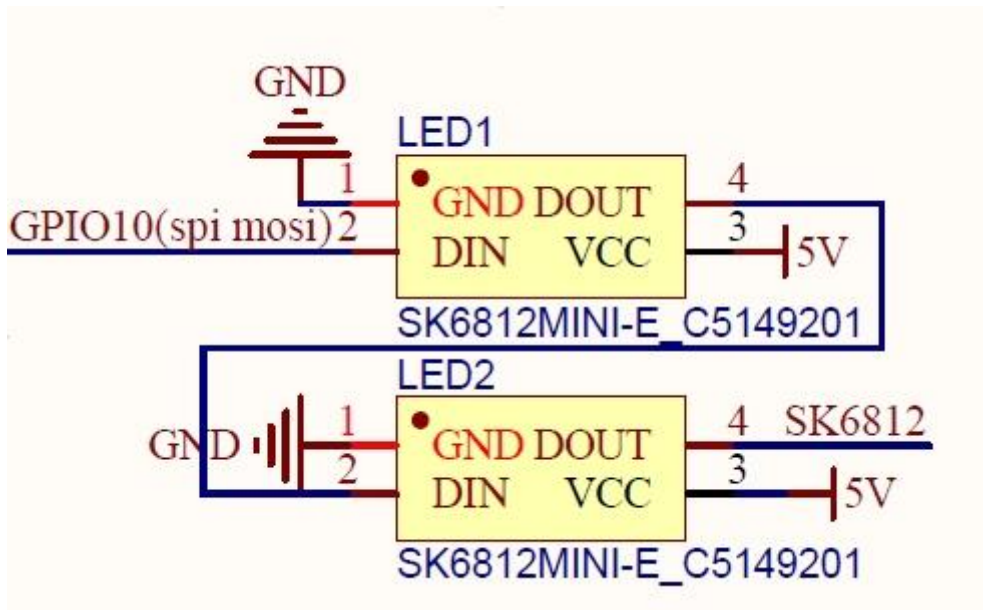
When using the Raspberry Pi to install the driver board Adeept Robot HAT V3.1, the WS2812 LED can be connected to the WS2812 interface on the Adeept Robot HAT V3.1 using a 3pin cable.

If you connect the WS2812 LED to the WS2812 interface of Adeept Robot HAT V3.1, the signal line is equivalent to connecting to the GPIO 10 of the Raspberry Pi.

10.3 Wiring Diagram

When the WS2812 LED is in use, the IN port needs to be connected to the WS2812 port on the Adeept Robot HAT V3.1 driver board.

Adeept Robot HAT V3.1 has 2 WS2812 LEDs on board, which are located at the front of the GPIO10 pin. When you use the WS2812 Port interface to connect one or more WS2812 LED modules, the first and second (0 and 1) in the code control the two WS2812LEDs on the board. Starting from the 3rd one is used to control the extended WS2812LED light. (2,3,4,...)



10.4 How to Control WS2812 LED

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 adept_picar-b2/examples/
```

```
pi@raspberrypi:~ $
pi@raspberrypi:~ $ cd adept_picar-b2/examples/
pi@raspberrypi:~/adept_picar-b2/examples $
```

3. View the contents of the current directory file:

```
ls
```

```
pi@raspberrypi:~/adept_picar-b2/examples $ ls
01_LED.py    03_servo.py  05_RGB.py    06_ws2812.py  08_lineTracking.py  10_BatteryLevelMonitoring.py
02_buzzer.py 04_motor.py  06_Spi_WS2812.py 07_ultra.py  09_lightTracking.py
```

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

```
sudo killall python3
```

```
pi@raspberrypi:~/adept_picar-b2/server $
pi@raspberrypi:~/adept_picar-b2/server $ sudo killall python3
```

If an error is reported when running the 06_Spi_WS2812.py program, please check the Q&A of this tutorial.

```
sudo python3 06_Spi_WS2812.py
```

```
pi@raspberrypi:~/adept_picar-b2/examples $ sudo python3 06_Spi_WS2812.py
spidev version is 3.5
spidev device as show:
/dev/spidev0.0 /dev/spidev0.1 /dev/spidev10.0
```

5. After running the program successfully, you will observe that the WS2812 alternately flashing lights of different colors.
6. When you want to terminate the running program, you can press the shortcut key "Ctrl + C" on the keyboard.

10.5 Main Code

For the complete code, please refer to the file [06_Spi_WS2812.py](#).

```
001 import spidev
002 import threading
003 import numpy
004 from numpy import sin, cos, pi
005 import time
006 class Aadept_SPI_LedPixel(threading.Thread):
007     def __init__(self, count = 8, bright = 255, sequence='GRB', bus = 0, device = 0, *args, **kwargs):
008         self.set_led_type(sequence)
009         self.set_led_count(count)
010         self.set_led_brightness(bright)
011         self.led_begin(bus, device)
012         self.lightMode = 'none'
013         self.colorBreathR = 0
014         self.colorBreathG = 0
015         self.colorBreathB = 0
016         self.breathSteps = 10
017         #self.spi_gpio_info()
018         self.set_all_led_color(0,0,0)
019         super(Aadept_SPI_LedPixel, self).__init__(*args, **kwargs)
020         self.__flag = threading.Event()
021         self.__flag.clear()
022     def led_begin(self, bus = 0, device = 0):
023         self.bus = bus
024         self.device = device
025         try:
026             self.spi = spidev.SpiDev()
027             self.spi.open(self.bus, self.device)
028             self.spi.mode = 0
029             self.led_init_state = 1
030         except OSError:
```

```

031         print("Please check the configuration in /boot/firmware/config.txt.")
032         if self.bus == 0:
033             print("You can turn on the 'SPI' in 'Interface Options' by using 'sudo raspi-config'.")
034             print("Or make sure that 'dtoverlay=spi=on' is not commented, then reboot the Raspberry Pi.
035 Otherwise spi0 will not be available.")
036         else:
037             print("Please add 'dtoverlay=spi{}-2cs' at the bottom of the /boot/firmware/config.txt, then
038 reboot the Raspberry Pi. otherwise spi{} will not be available.".format(self.bus, self.bus))
039             self.led_init_state = 0
040
041     def check_spi_state(self):
042         return self.led_init_state
043
044     def spi_gpio_info(self):
045         if self.bus == 0:
046             print("SPI0-MOSI: GPIO10(WS2812-PIN) SPI0-MISO: GPIO9 SPI0-SCLK: GPIO11 SPI0-CE0:
047 GPIO8 SPI0-CE1: GPIO7")
048         elif self.bus == 1:
049             print("SPI1-MOSI: GPIO20(WS2812-PIN) SPI1-MISO: GPIO19 SPI1-SCLK: GPIO21 SPI1-CE0:
050 GPIO18 SPI1-CE1: GPIO17 SPI0-CE1: GPIO16")
051         elif self.bus == 2:
052             print("SPI2-MOSI: GPIO41(WS2812-PIN) SPI2-MISO: GPIO40 SPI2-SCLK: GPIO42 SPI2-CE0:
053 GPIO43 SPI2-CE1: GPIO44 SPI2-CE1: GPIO45")
054         elif self.bus == 3:
055             print("SPI3-MOSI: GPIO2(WS2812-PIN) SPI3-MISO: GPIO1 SPI3-SCLK: GPIO3 SPI3-CE0:
056 GPIO0 SPI3-CE1: GPIO24")
057         elif self.bus == 4:
058             print("SPI4-MOSI: GPIO6(WS2812-PIN) SPI4-MISO: GPIO5 SPI4-SCLK: GPIO7 SPI4-CE0:
059 GPIO4 SPI4-CE1: GPIO25")
060         elif self.bus == 5:
061             print("SPI5-MOSI: GPIO14(WS2812-PIN) SPI5-MISO: GPIO13 SPI5-SCLK: GPIO15 SPI5-CE0:
062 GPIO12 SPI5-CE1: GPIO26")
063         elif self.bus == 6:
064             print("SPI6-MOSI: GPIO20(WS2812-PIN) SPI6-MISO: GPIO19 SPI6-SCLK: GPIO21 SPI6-CE0:
065 GPIO18 SPI6-CE1: GPIO27")
066
067     def led_close(self):
068         self.set_all_led_rgb([0,0,0])
069         self.spi.close()
070
071     def set_led_count(self, count):
072         self.led_count = count

```

```

073     self.led_color = [0,0,0] * self.led_count
074     self.led_original_color = [0,0,0] * self.led_count
075
076     def set_led_type(self, rgb_type):
077         try:
078             led_type = ['RGB', 'RBG', 'GRB', 'GBR', 'BRG', 'BGR']
079             led_type_offset = [0x06, 0x09, 0x12, 0x21, 0x18, 0x24]
080             index = led_type.index(rgb_type)
081             self.led_red_offset = (led_type_offset[index]>>4) & 0x03
082             self.led_green_offset = (led_type_offset[index]>>2) & 0x03
083             self.led_blue_offset = (led_type_offset[index]>>0) & 0x03
084             return index
085         except ValueError:
086             self.led_red_offset = 1
087             self.led_green_offset = 0
088             self.led_blue_offset = 2
089             return -1
090
091     def set_led_brightness(self, brightness):
092         self.led_brightness = brightness
093         for i in range(self.led_count):
094             self.set_led_rgb_data(i, self.led_original_color)
095
096     def set_ledpixel(self, index, r, g, b):
097         p = [0,0,0]
098         p[self.led_red_offset] = round(r * self.led_brightness / 255)
099         p[self.led_green_offset] = round(g * self.led_brightness / 255)
100         p[self.led_blue_offset] = round(b * self.led_brightness / 255)
101         self.led_original_color[index*3+self.led_red_offset] = r
102         self.led_original_color[index*3+self.led_green_offset] = g
103         self.led_original_color[index*3+self.led_blue_offset] = b
104         for i in range(3):
105             self.led_color[index*3+i] = p[i]
106
107     def set_led_color_data(self, index, r, g, b):
108         self.set_ledpixel(index, r, g, b)
109
110     def set_led_rgb_data(self, index, color):
111         self.set_ledpixel(index, color[0], color[1], color[2])
112
113     def set_led_color(self, index, r, g, b):
114         self.set_ledpixel(index, r, g, b)

```



```

115         self.show()
116
117     def set_led_rgb(self, index, color):
118         self.set_led_rgb_data(index, color)
119         self.show()
120
121     def set_all_led_color_data(self, r, g, b):
122         for i in range(self.led_count):
123             self.set_led_color_data(i, r, g, b)
124
125     def set_all_led_rgb_data(self, color):
126         for i in range(self.led_count):
127             self.set_led_rgb_data(i, color)
128
129     def set_all_led_color(self, r, g, b):
130         for i in range(self.led_count):
131             self.set_led_color_data(i, r, g, b)
132         self.show()
133
134     def set_all_led_rgb(self, color):
135         for i in range(self.led_count):
136             self.set_led_rgb_data(i, color)
137         self.show()
138
139     def write_ws2812_numpy8(self):
140         d = numpy.array(self.led_color).ravel()           #Converts data into a one-dimensional array
141         tx = numpy.zeros(len(d)*8, dtype=numpy.uint8)    #Each RGB color has 8 bits, each represented by a
142         uint8 type data
143         for ibit in range(8):                             #Convert each bit of data to the data that the spi
144         will send
145             tx[7-ibit::8]=((d>>ibit)&1)*0x78 + 0x80      #T0H=1,T0L=7, T1H=5,T1L=3  #0b11111000 mean
146             T1(0.78125us), 0b10000000 mean T0(0.15625us)
147             if self.led_init_state != 0:
148                 if self.bus == 0:
149                     self.spi.xfer(tx.tolist(), int(8/1.25e-6))    #Send color data at a frequency of 6.4Mhz
150                 else:
151                     self.spi.xfer(tx.tolist(), int(8/1.0e-6))      #Send color data at a frequency of 8Mhz
152
153     def write_ws2812_numpy4(self):
154         d=numpy.array(self.led_color).ravel()
155         tx=numpy.zeros(len(d)*4, dtype=numpy.uint8)
156         for ibit in range(4):

```



```

157         tx[3-ibit::4]=((d>>(2*ibit+1))&1)*0x60 + ((d>>(2*ibit+0))&1)*0x06 + 0x88
158     if self.led_init_state != 0:
159         if self.bus == 0:
160             self.spi.xfer(tx.tolist(), int(4/1.25e-6))
161         else:
162             self.spi.xfer(tx.tolist(), int(4/1.0e-6))
163
164     def show(self, mode = 1):
165         if mode == 1:
166             write_ws2812 = self.write_ws2812_numpy8
167         else:
168             write_ws2812 = self.write_ws2812_numpy4
169         write_ws2812()
170
171     def wheel(self, pos):
172         if pos < 85:
173             return [(255 - pos * 3), (pos * 3), 0]
174         elif pos < 170:
175             pos = pos - 85
176             return [0, (255 - pos * 3), (pos * 3)]
177         else:
178             pos = pos - 170
179             return [(pos * 3), 0, (255 - pos * 3)]
180
181     def hsv2rgb(self, h, s, v):
182         h = h % 360
183         rgb_max = round(v * 2.55)
184         rgb_min = round(rgb_max * (100 - s) / 100)
185         i = round(h / 60)
186         diff = round(h % 60)
187         rgb_adj = round((rgb_max - rgb_min) * diff / 60)
188         if i == 0:
189             r = rgb_max
190             g = rgb_min + rgb_adj
191             b = rgb_min
192         elif i == 1:
193             r = rgb_max - rgb_adj
194             g = rgb_max
195             b = rgb_min
196         elif i == 2:
197             r = rgb_min
198             g = rgb_max

```

```
199         b = rgb_min + rgb_adj
200     elif i == 3:
201         r = rgb_min
202         g = rgb_max - rgb_adj
203         b = rgb_max
204     elif i == 4:
205         r = rgb_min + rgb_adj
206         g = rgb_min
207         b = rgb_max
208     else:
209         r = rgb_max
210         g = rgb_min
211         b = rgb_max - rgb_adj
212     return [r, g, b]
213
214     def police(self):
215         self.lightMode = 'police'
216         self.resume()
217
218     def breath(self, R_input, G_input, B_input):
219         self.lightMode = 'breath'
220         self.colorBreathR = R_input
221         self.colorBreathG = G_input
222         self.colorBreathB = B_input
223         self.resume()
224
225     def resume(self):
226         self.__flag.set()
227
228
229     def breathProcessing(self):
230         while self.lightMode == 'breath':
231             for i in range(0, self.breathSteps):
232                 if self.lightMode != 'breath':
233                     break
234                 self.set_all_led_color(self.colorBreathR*i/self.breathSteps,
235 self.colorBreathG*i/self.breathSteps, self.colorBreathB*i/self.breathSteps)
236                 #self.show()
237                 time.sleep(0.03)
238             for i in range(0, self.breathSteps):
239                 if self.lightMode != 'breath':
240                     break
```

```
241         self.set_all_led_color(self.colorBreathR-(self.colorBreathR*i/self.breathSteps),
242 self.colorBreathG-(self.colorBreathG*i/self.breathSteps),
243 self.colorBreathB-(self.colorBreathB*i/self.breathSteps))
244         #self.show()
245         time.sleep(0.03)
246     def policeProcessing(self):
247         while self.lightMode == 'police':
248             for i in range(0,3):
249                 self.set_all_led_color_data(0,0,255)
250                 self.show()
251                 time.sleep(0.05)
252                 self.set_all_led_color_data(0,0,0)
253                 self.show()
254                 time.sleep(0.05)
255             if self.lightMode != 'police':
256                 break
257             time.sleep(0.1)
258             for i in range(0,3):
259                 self.set_all_led_color_data(255,0,0)
260                 self.show()
261                 time.sleep(0.05)
262                 self.set_all_led_color_data(0,0,0)
263                 self.show()
264                 time.sleep(0.05)
265             time.sleep(0.1)
266
267
268     def lightChange(self):
269         if self.lightMode == 'none':
270             self.pause()
271         elif self.lightMode == 'police':
272             self.policeProcessing()
273         elif self.lightMode == 'breath':
274             self.breathProcessing()
275
276     def run(self):
277         while 1:
278             self.__flag.wait()
279             self.lightChange()
280             pass
281
282
```

```
283
284 if __name__ == '__main__':
285     import time
286     import os
287     print("spidev version is ", spidev.__version__)
288     print("spidev device as show:")
289     os.system("ls /dev/spi*")
290
291     led = Adeept_SPI_LedPixel(8, 255)          # Use MOSI for /dev/spidev0 to drive the lights
292
293     try:
294         if led.check_spi_state() != 0:
295             led.set_led_count(8)
296             led.set_all_led_color_data(255, 0, 0)
297             led.show()
298             time.sleep(0.5)
299             led.set_all_led_rgb_data([0, 255, 0])
300             led.show()
301             time.sleep(0.5)
302             led.set_all_led_color(0, 0, 255)
303             time.sleep(0.5)
304             led.set_all_led_rgb([0, 255, 255])
305             time.sleep(0.5)
306
307             led.set_led_count(12)
308             led.set_all_led_color_data(255, 255, 0)
309             for i in range(255):
310                 led.set_led_brightness(i)
311                 led.show()
312                 time.sleep(0.005)
313             for i in range(255):
314                 led.set_led_brightness(255-i)
315                 led.show()
316                 time.sleep(0.005)
317
318             led.set_led_brightness(20)
319             while True:
320                 for j in range(255):
321                     for i in range(led.led_count):
322                         led.set_led_rgb_data(i, led.wheel((round(i * 255 / led.led_count) + j)%256))
323                         led.show()
324                         time.sleep(0.002)
```

```
    else:
        led.led_close()
except KeyboardInterrupt:
    led.led_close()
```