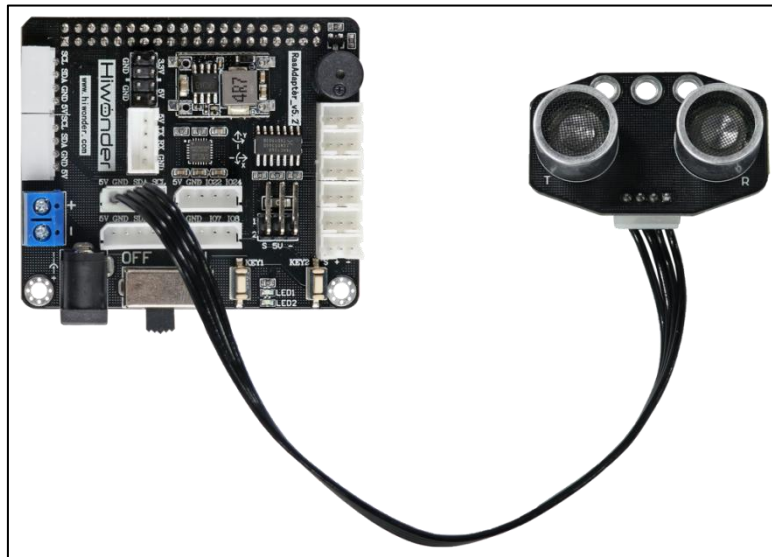


Lesson 4 Ultrasonic sensor

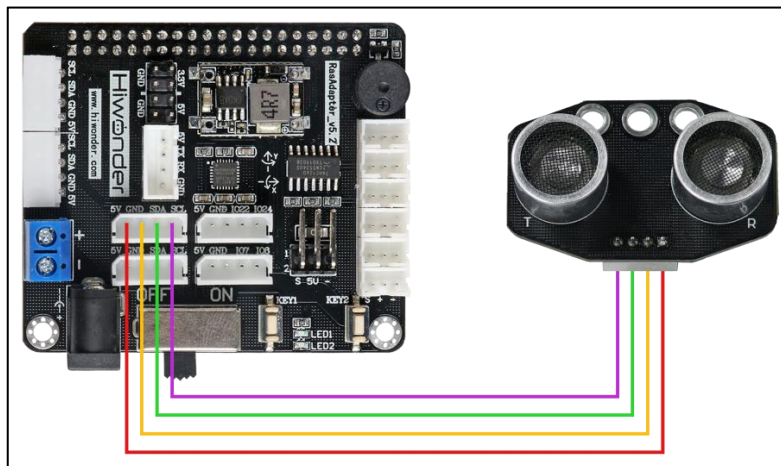
1. Getting Ready

Prepare an ultrasonic sensor and connect it to any one of IIC ports on Raspberry Pi expansion board through a 4PIN cable. The wiring effect is as follow:



Note: Please do not insert forcefully because 4PIN wire uses anti-reverse plugging design.

In addition, 4 female to female Dupont lines can also be used to connect the ultrasonic sensor to Raspberry Pi expansion board, as the figure shown below:



2. Module Usage

In this lesson, the distance measurement chip of the ultrasonic sensor integrates ultrasonic transmitter circuit, ultrasonic receiver circuit and digital processing circuit, etc. The module uses IIC communication ports, which can take advantage of IIC communication to read the measured distance.

In addition, the ultrasonic probe integrates two RGB lights, which can not only adjust the brightness of the light, but also realize the color change by modifying the parameters of three color channels of red (R), green (G) and blue (B).

3. Working Principle


Firstly, set the distance measurement. Then control the on and off of the RGB light by changing the high and low levels. Finally, control the displayed light color by changing the value of each color component.

The source code of program is located in:

/home/pi/TonyPi/Extend/Sensor/Sonar_RGBD.py

```
31 if __name__ == "__main__":
32     while True:
33         time.sleep(1)
34         if s.getDistance() != 99999:
35             print("Distance:", s.getDistance(), "mm")
36             distance = s.getDistance()
37
38             if 0.0 <= distance <= 100.0:
39                 s.setRGBMode(0)
40                 s.setRGB(1, (255,0,0))
41                 s.setRGB(0, (255,0,0))
42
43             if 100.0 < distance <= 150.0:
44                 s.setRGBMode(0)
45                 s.setRGB(1, (0,255,0))
46                 s.setRGB(0, (0,255,0))
47
48             if 150.0 < distance <= 200.0:
49                 s.setRGBMode(0)
50                 s.setRGB(1, (0,0,255))
51                 s.setRGB(0, (0,0,255))
52
53             if distance > 200.0:
54                 s.setRGBMode(0)
55                 s.setRGB(1, (255,255,255))
56                 s.setRGB(0, (255,255,255))
```

4. Operation Steps

- 1) Click  in the upper left corner to open the terminal. Enter command “cd TonyPi/Extend/Sensor/” and press “Enter” to come to the directory of the game programmings.

```
pi@raspberrypi:~ $ cd TonyPi/Extend/Sensor/
```

- 2) Enter “sudo python3 Sonar_RGBD.py” command, and then press “Enter” to start the game.

```
pi@raspberrypi:~ $ cd TonyPi/Extend/Sensor/  
pi@raspberrypi:~/TonyPi/Extend/Sensor $ sudo python3 Sonar_RGBD.py
```

- 3) If want to close this program, press “Ctrl+C”. You can try multiple time if fail to close.

5. Project Outcome

After the program is started, the obstacle is placed in front of the ultrasonic sensor. Then the terminal interface will print the measured distance and RGB light will light up the corresponding color light. The corresponding relation between the light color and the distance range is as follow:

When the distance is less than 100mm, RGB light will display red light.

When the distance is more than 100mm and less than 150mm, RGB light displays green light.

When the distance is more than 150mm and less than 200mm. RGB light displays blue light.

When the distance is more than 200mm, RGB light displays white light.

6. Function Extension

6.1 Modify the Measured Distance

We can modify the distance range corresponding to the RGB light. Take modifying the distance range “150.0<distance<=200.0” corresponding to blue light to “150.0<distance<=250.0”, and modifying the distance range “distance>200” corresponding to white light to “distance>250” as example. Please refer to the following steps:

1) Open LX terminal, and then enter command “cd TonyPi/Extend/Sensor/” to come to the directory of the game programmings.

```
pi@raspberrypi:~ $ cd TonyPi/Extend/Sensor/
```

2) Enter “sudo vim Sonar_RGBD.py” command, and then press “Enter” to start the game.

```
pi@raspberrypi:~ $ cd TonyPi/Extend/Sensor/
pi@raspberrypi:~/TonyPi/Extend/Sensor $ sudo vim Sonar_RGBD.py
```

3) Find the code shown in the figure below.

```
36         distance = s.getDistance()
37
38         if 0.0 <= distance <= 100.0:
39             s.setRGBMode(0)
40             s.setRGB(1, (255,0,0)) #两边 RGB设置为红色
41             s.setRGB(0, (255,0,0))
42
43         if 100.0 < distance <= 150.0:
44             s.setRGBMode(0)
45             s.setRGB(1, (0,255,0)) #两边 RGB设置为绿色
46             s.setRGB(0, (0,255,0))
47
48         if 150.0 < distance <= 200.0:
49             s.setRGBMode(0)
50             s.setRGB(1, (0,0,255)) #两边 RGB设置为蓝色
51             s.setRGB(0, (0,0,255))
52
53         if distance > 200.0:
54             s.setRGBMode(0)
55             s.setRGB(1, (255,255,255)) # 两边 RGB设置为白色
56             s.setRGB(0, (255,255,255))
```

4) Press “i” key on keyboard to enter the edit mode.

```

pi@raspberrypi: ~/TonyPi/Extend/Sensor
文件(F) 编辑(E) 标签(T) 帮助(H)
34     if s.getDistance() != 99999:
35         print("Distance:", s.getDistance() , "mm")
36         distance = s.getDistance()
37
38         if 0.0 <= distance <= 100.0:
39             s.setRGBMode(0)
40             s.setRGB(1, (255,0,0)) #两边 RGB设置为红色
41             s.setRGB(0, (255,0,0))
42
43         if 100.0 < distance <= 150.0:
44             s.setRGBMode(0)
45             s.setRGB(1, (0,255,0)) #两边 RGB设置为绿色
46             s.setRGB(0, (0,255,0))
47
48         if 150.0 < distance <= 200.0:
49             s.setRGBMode(0)
50             s.setRGB(1, (0,0,255)) #两边 RGB设置为蓝色
51             s.setRGB(0, (0,0,255))
52
53         if distance > 200.0:
54             s.setRGBMode(0)
55             s.setRGB(1, (255,255,255)) # 两边 RGB设置为白色
56             s.setRGB(0, (255,255,255))
-- 插入 --
39,1 底端

```

5) Modify data as the figure shown below:

```

38         if 0.0 <= distance <= 100.0:
39             s.setRGBMode(0)
40             s.setRGB(1, (255,0,0)) #两边 RGB设置为红色
41             s.setRGB(0, (255,0,0))
42
43         if 100.0 < distance <= 150.0:
44             s.setRGBMode(0)
45             s.setRGB(1, (0,255,0)) #两边 RGB设置为绿色
46             s.setRGB(0, (0,255,0))
47
48         if 150.0 < distance <= 250.0:
49             s.setRGBMode(0)
50             s.setRGB(1, (0,0,255)) #两边 RGB设置为蓝色
51             s.setRGB(0, (0,0,255))
52
53         if distance > 250.0:
54             s.setRGBMode(0)
55             s.setRGB(1, (255,255,255)) # 两边 RGB设置为白色
56             s.setRGB(0, (255,255,255))
-- 插入 --
56,30 底端

```

6) After modification, press “Esc” and enter “:wq” (please note that the colon is in front of wq), and then press “Enter” to save the modified content.

```

49             s.setRGBMode(0)
50             s.setRGB(1, (0,0,255)) #两边 RGB设置为蓝色
51             s.setRGB(0, (0,0,255))
52
53         if distance > 250.0:
54             s.setRGBMode(0)
55             s.setRGB(1, (255,255,255)) # 两边 RGB设置为白色
56             s.setRGB(0, (255,255,255))
:wq

```


6.2 Customize RGB Color

Similarly, we can change the RGB light color. Take changing the color of RGB light from white to orange as example. The specific operation step is as follow:

- 1) Please refer to “6.1 Modify the Mesured Distance” to open game program file
- 2) Find the code as the figure shown below in the opening interface.

```

34         if s.getDistance() != 99999:
35             print("Distance:", s.getDistance() , "mm")
36             distance = s.getDistance()
37
38             if 0.0 <= distance <= 100.0:
39                 s.setRGBMode(0)
40                 s.setRGB(1, (255,0,0)) #两边 RGB设置为红色
41                 s.setRGB(0, (255,0,0))
42
43             if 100.0 < distance <= 150.0:
44                 s.setRGBMode(0)
45                 s.setRGB(1, (0,255,0)) #两边 RGB设置为绿色
46                 s.setRGB(0, (0,255,0))
47
48             if 150.0 < distance <= 200.0:
49                 s.setRGBMode(0)
50                 s.setRGB(1, (0,0,255)) #两边 RGB设置为蓝色
51                 s.setRGB(0, (0,0,255))
52
53             if distance > 200.0:
54                 s.setRGBMode(0)
55                 s.setRGB(1, (255,255,255)) # 两边 RGB设置为白色
56                 s.setRGB(0, (255,255,255))

```

- 3) Modify the value of RGB to change the light color.
“setRGB(1,(255,255,255))” and “setRGB(0,(255,255,255))” are modified to
“setRGB(1,(255,127,0))” and “setRGB(0,(255,127,0))”, as the figure shown below:

```

47
48         if 150.0 < distance <= 200.0:
49             s.setRGBMode(0)
50             s.setRGB(1, (0,0,255)) #两边 RGB设置为蓝色
51             s.setRGB(0, (0,0,255))
52
53         if distance > 200.0:
54             s.setRGBMode(0)
55             s.setRGB(1, (255,127,0)) # 两边 RGB设置为橙色
56             s.setRGB(0, (255,127,0))
-- 插入 --

```

RGB value refers to the red, green and blue components in a certain color.

Theoretically, the three primary colors which are red, green and blue can be mixed in different proportion to make any color. The closer of the RGB value of a color, the closer it is to gray or black and white, and the larger the value, the whiter it is, and vice versa, the darker it is.

For example, the value of B is the largest in RGB (150, 152, 183), which means the color contains more blue color, so it can be identified as a grayish blue.

1) After modification, press “Esc” and enter “:wq” (please note that the colon is in front of wq), and then press “Enter” to save the modified content.

```
52
53         if distance > 200.0:
54             s.setRGBMode(0)
55             s.setRGB(1, (255,127,0)) # 两边 RGB 设置为橙色
56             s.setRGB(0, (255,127,0))
:wq
```