**Project 28：Rocker control light**

1. **Introduction**

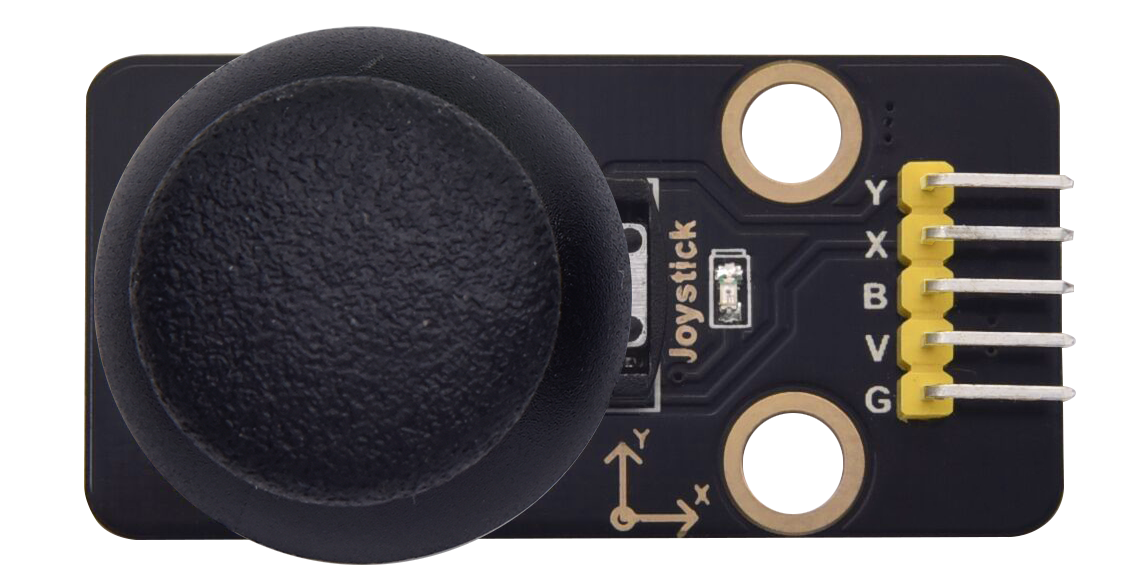
The joystick module is a component with two analog inputs and one digital input. It is widely used in game operation, robot control, drone control and other fields.

In this project, we will use a Raspberry Pi Pico and a joystick module to control RGB. You can have a deeper understanding of the principle and operation of the joystick module in practice.

1. **Components Required**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 17a6d7f241a04d4e932cb06b758197c4 | | _DSC2552 | | 4 | |
| Raspberry Pi Pico\*1 | | Raspberry Pi Pico Expansion Board\*1 | | Joystick Module\*1 | |
| 图层 2 拷贝 2 |  |  |  | 杜邦线-3 |  |
| RGB LED\*1 | 220ΩResistor\*3 | Jumper Wires | USB Cable\*1 | M-F Dupont Wires | Breadboard\*1 |

1. **Component Knowledge**



**Joystick module**: It mainly uses PS2 joystick components. In fact, the joystick module has 3 signal terminal pins, which simulate a three-dimensional space. The pins of the joystick module are GND, VCC, and signal terminals (B, X, Y). The signal terminals X and Y simulate the X-axis and Y-axis of the space. When controlling, the X and Y signal terminals of the module are connected to the analog port of the microcontroller. The signal terminal B simulates the Z axis of the space, it is generally connected to the digital port and used as a button.

VCC is connected to the microcontroller power output VCC (3.3V or 5V), GND is connected to the microcontroller GND, the voltage in the original state is about 1.65V or 2.5V. In the X-axis direction, when moving in the direction of the arrow, the voltage value increases, and the maximum voltage can be reached. Moving in the opposite direction of the arrow, the voltage value gradually decreases to the minimum voltage. In the Y-axis direction, the voltage value decreases gradually as it moves in the direction of the arrow on the module, decreasing to the minimum voltage. As the arrow is moved in the opposite direction, the voltage value increases and can reach the maximum voltage. In the Z-axis direction, the signal terminal B is connected to the digital port and outputs 0 in the original state and outputs 1 when pressed. In this way, we can read the two analog values and the high and low level conditions of the digital port to determine the operating status of the joystick on the module.

**Features:**

Input Voltage：DC 3.3V ~ 5V

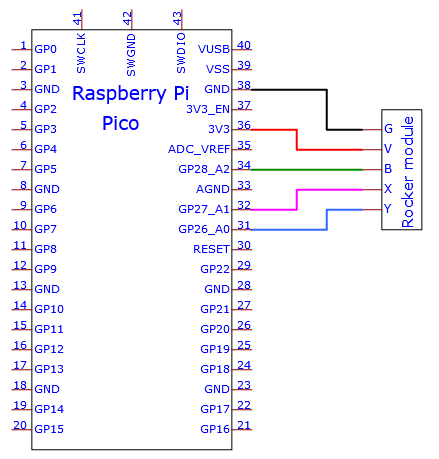
Output Signal：X/Y dual axis analog value +Z axis digital signal

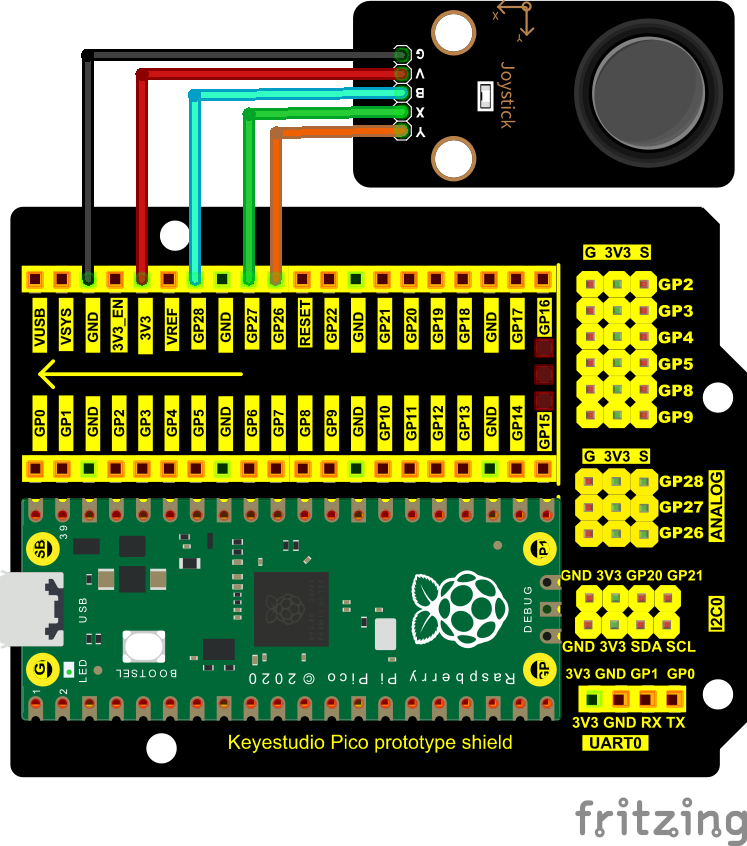
[Range](javascript:;) [of](javascript:;) [Application](javascript:;)：Suitable for control point coordinate movement in plane as well as control of two degrees of freedom steering gear, etc.

[product](javascript:;) [feature](javascript:;)s：Exquisite appearance, joystick feel superior, simple operation, sensitive response, long service life.

1. **Read the Value**

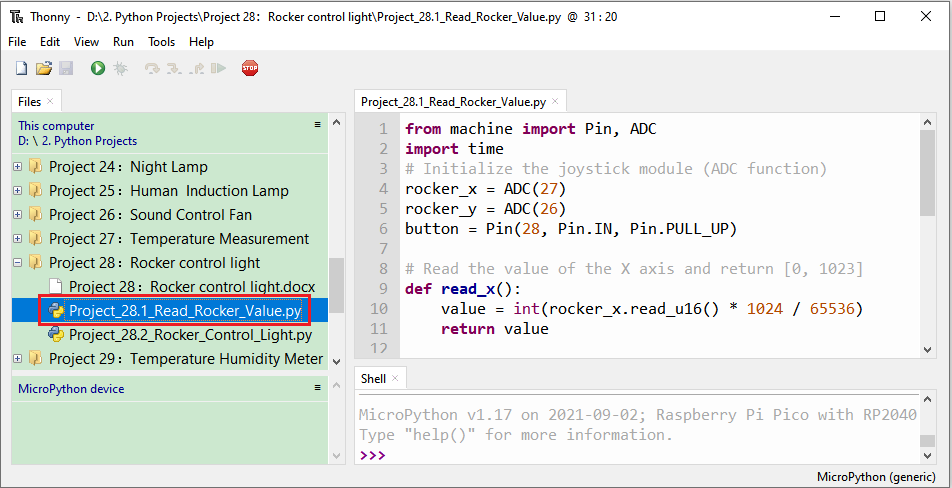
We have to use analog Raspberry Pi Pico pin IO to read the data from X or Y pins, and use digital IO port to read the values of the button. Please follow the wiring diagram below for wiring.





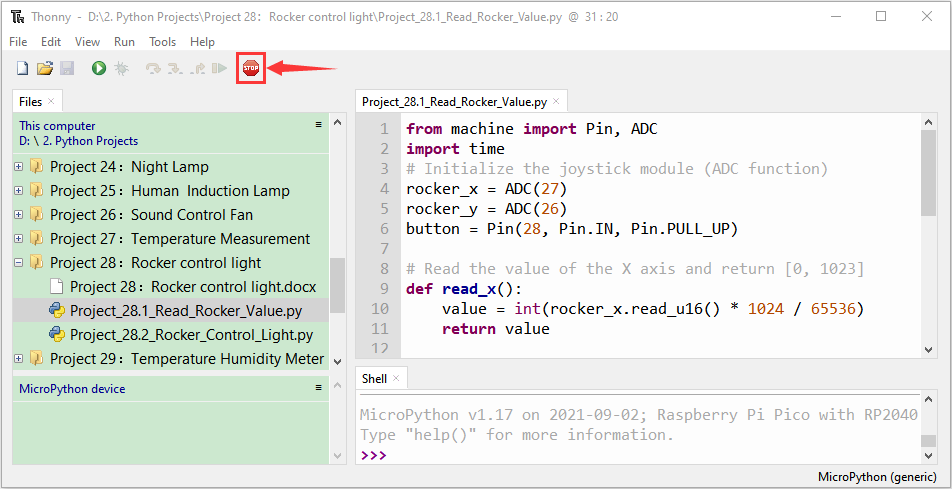
The code used in this project is saved in the file KS3020 Keyestudio Raspberry Pi Pico Learning Kit Ultimate Edition\2. Windows System\1. Python\_Tutorial\2. Python Projects\Project 28：Rocker control light. You can move the code to anywhere, for example, we can save the code in the Disk(D), the route is D:\2. Python Projects.

Open“Thonny”, click“This computer”→“D:”→“2. Python Projects”→“Project 28：Rocker control light”. And double left-click the“Project\_28.1\_Read\_Rocker\_Value.py”.

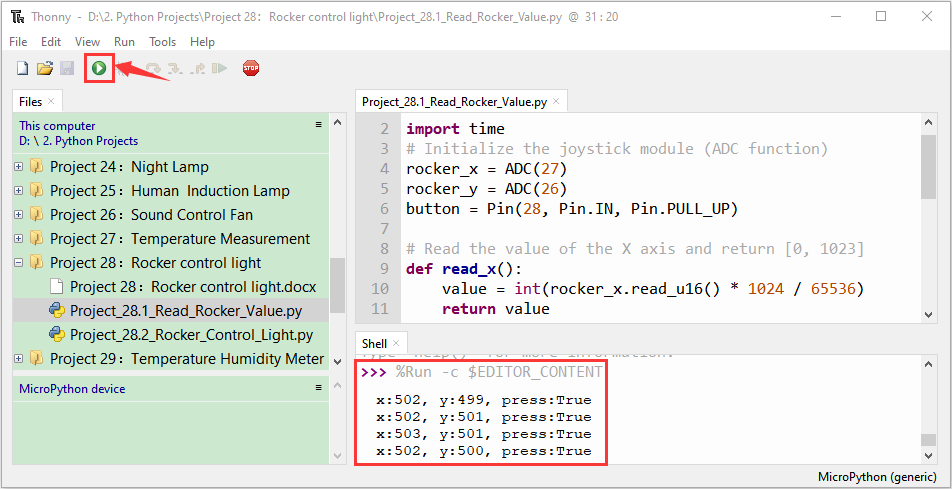


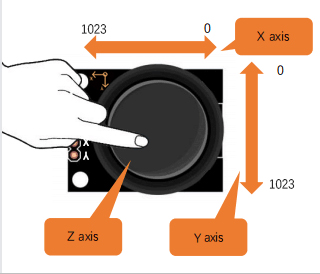
|  |
| --- |
| from machine import Pin, ADC  import time  # Initialize the joystick module (ADC function)  rocker\_x = ADC(27)  rocker\_y = ADC(26)  button = Pin(28, Pin.IN, Pin.PULL\_UP)    # Read the value of the X axis and return [0, 1023]  def read\_x():  value = int(rocker\_x.read\_u16() \* 1024 / 65536)  return value    # Read the value of Y axis and return [0, 1023]  def read\_y():  value = int(rocker\_y.read\_u16() \* 1024 / 65536)  return value    # Read the state of the button, press to return to True, release to return to False  def btn\_state():  press = False  if button.value() == 0:  press = True  return press    # Print the current value of the X axis,Y axis,Z axis cyclically.  while True:  value\_x = read\_x()  value\_y = read\_y()  state = btn\_state()  print("x:%d, y:%d, press:%s" % (value\_x, value\_y, state))  time.sleep(0.1) |

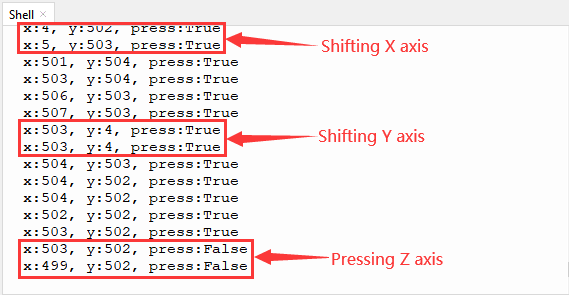
Ensure that the Raspberry Pi Pico is connected to the computer，click“Stop/Restart backend”.



Click “Run current script”, the code starts executing, we will see that the "Shell" window of Thonny IDE will print the analog and digital values of the current joystick. Moving the joystick or pressing it will change the analog and digital values in "Shell". Press“Ctrl+C”or click“Stop/Restart backend”to exit the program.

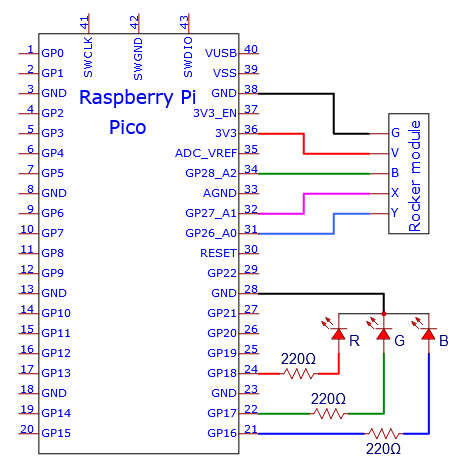


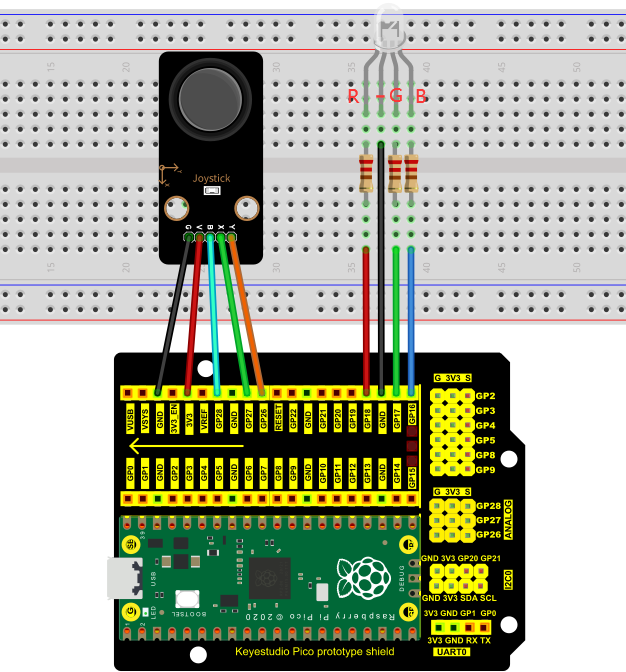




1. **Circuit Diagram and Wiring Diagram**

We just read the value of the joystick module. Now we need to do something with the joystick module and RGB, connecting according to the following diagram.

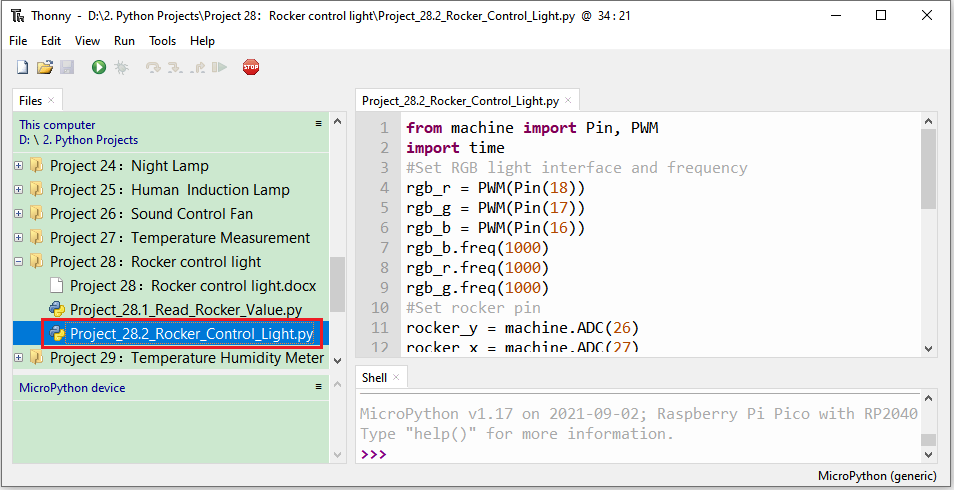




1. **Test Code**

The code used in this project is saved in the file KS3020 Keyestudio Raspberry Pi Pico Learning Kit Ultimate Edition\2. Windows System\1. Python\_Tutorial\2. Python Projects\Project 28：Rocker control light. You can move the code to anywhere, for example, we can save the code in the Disk(D), the route is D:\2. Python Projects.

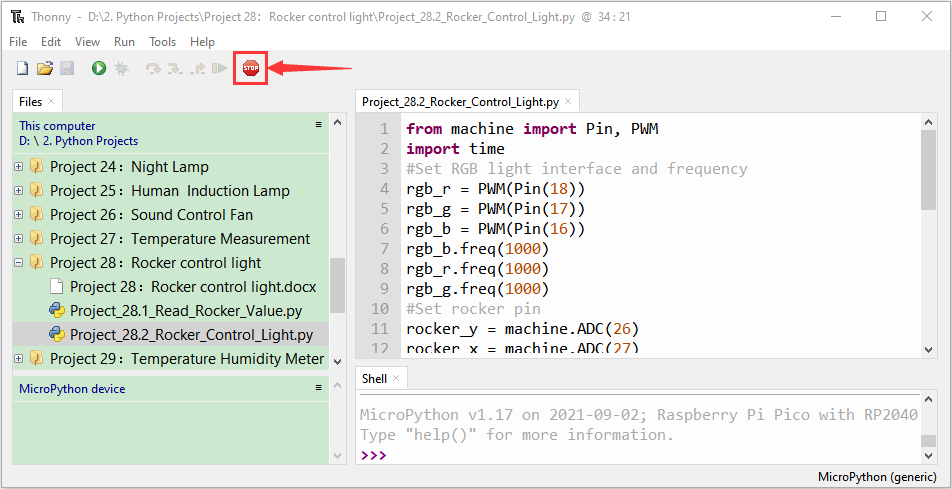
Open“Thonny”, click“This computer”→“D:”→“2. Python Projects”→“Project 28：Rocker control light”. And double left-click the“Project\_28.2\_Rocker\_Control\_Light.py”.



|  |
| --- |
| from machine import Pin, PWM  import time  #Set RGB light interface and frequency  rgb\_r = PWM(Pin(18))  rgb\_g = PWM(Pin(17))  rgb\_b = PWM(Pin(16))  rgb\_b.freq(1000)  rgb\_r.freq(1000)  rgb\_g.freq(1000)  #Set rocker pin  rocker\_y = machine.ADC(26)  rocker\_x = machine.ADC(27)  y=500  x=500  while True:  y = rocker\_y.read\_u16()#Get Y value of rocker  x = rocker\_x.read\_u16()#Get X value of rocker  if x < 6400: #left  rgb\_b.duty\_u16(0)  rgb\_r.duty\_u16(65535)  rgb\_g.duty\_u16(0)  elif x > 38400: #right  rgb\_b.duty\_u16(0)  rgb\_r.duty\_u16(0)  rgb\_g.duty\_u16(65535)  elif y < 6400: #down  rgb\_b.duty\_u16(65535)  rgb\_r.duty\_u16(0)  rgb\_g.duty\_u16(0)  elif y > 38400: #up  rgb\_b.duty\_u16(65535)  rgb\_r.duty\_u16(65535)  rgb\_g.duty\_u16(65535)  time.sleep(0.01) |

1. **Test Result**

Ensure that the Raspberry Pi Pico is connected to the computer，click“Stop/Restart backend”.



Click“Run current script”, the code starts executing, we will see that①

If the joystick is moved to the far left in the X direction, the RGB light turns red. ② If the joystick is moved to the far right in the X direction, the RGB light turns green. ③If the joystick is moved to the top in the Y direction, the RGB light turns white. ④If the joystick is moved to the bottom in the Y direction, the RGB light turns blue. Press“Ctrl+C”or click“Stop/Restart backend”to exit the program.

