

Pico serial communication

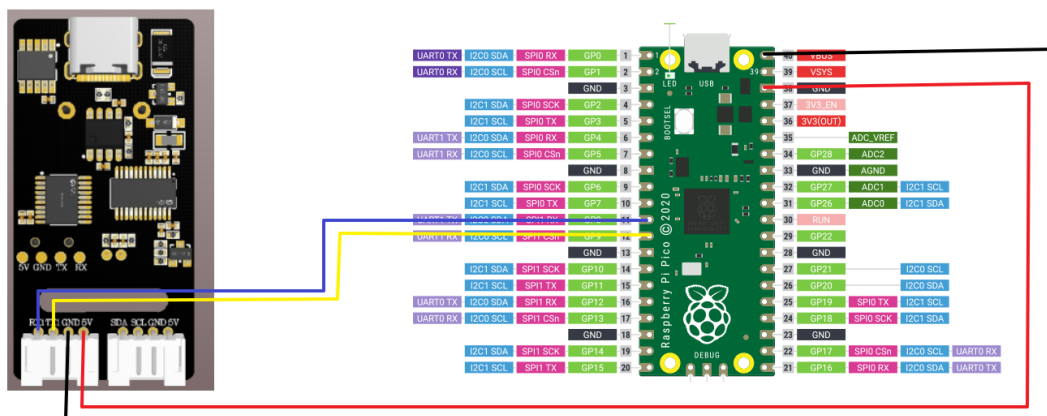
Note: The voice interaction module needs to be burned with factory firmware. If the voice chip has not been flashed with firmware after receiving it, it does not need to be burned

1. Experimental preparation

- Pico
- Voice interaction module
- Dupont line

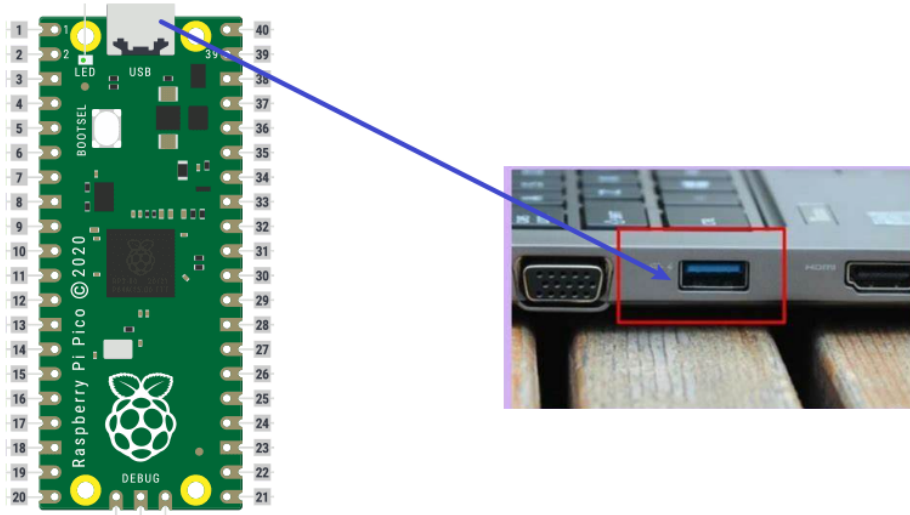
2. Wiring diagram

Pico	Voice interaction module
4	RX
5	TX
GND	GND
5V	5V

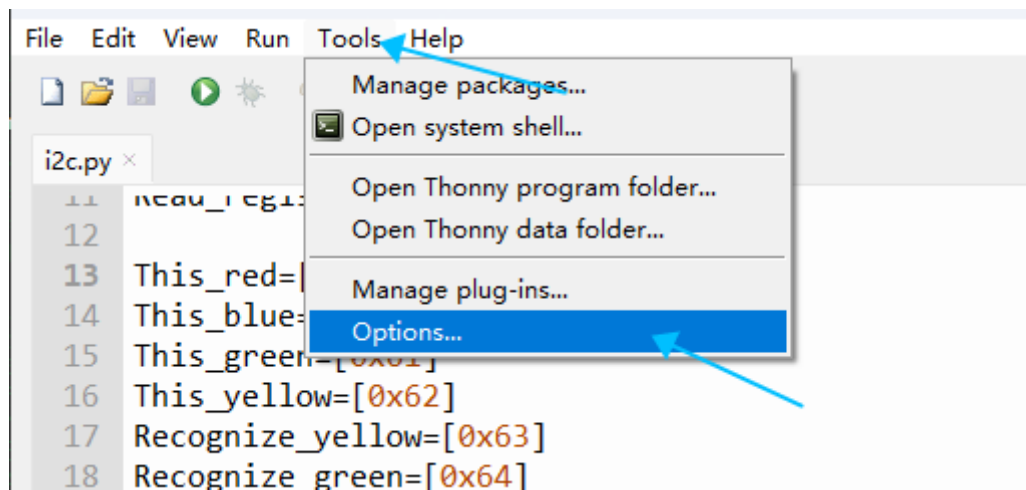


3. Program download

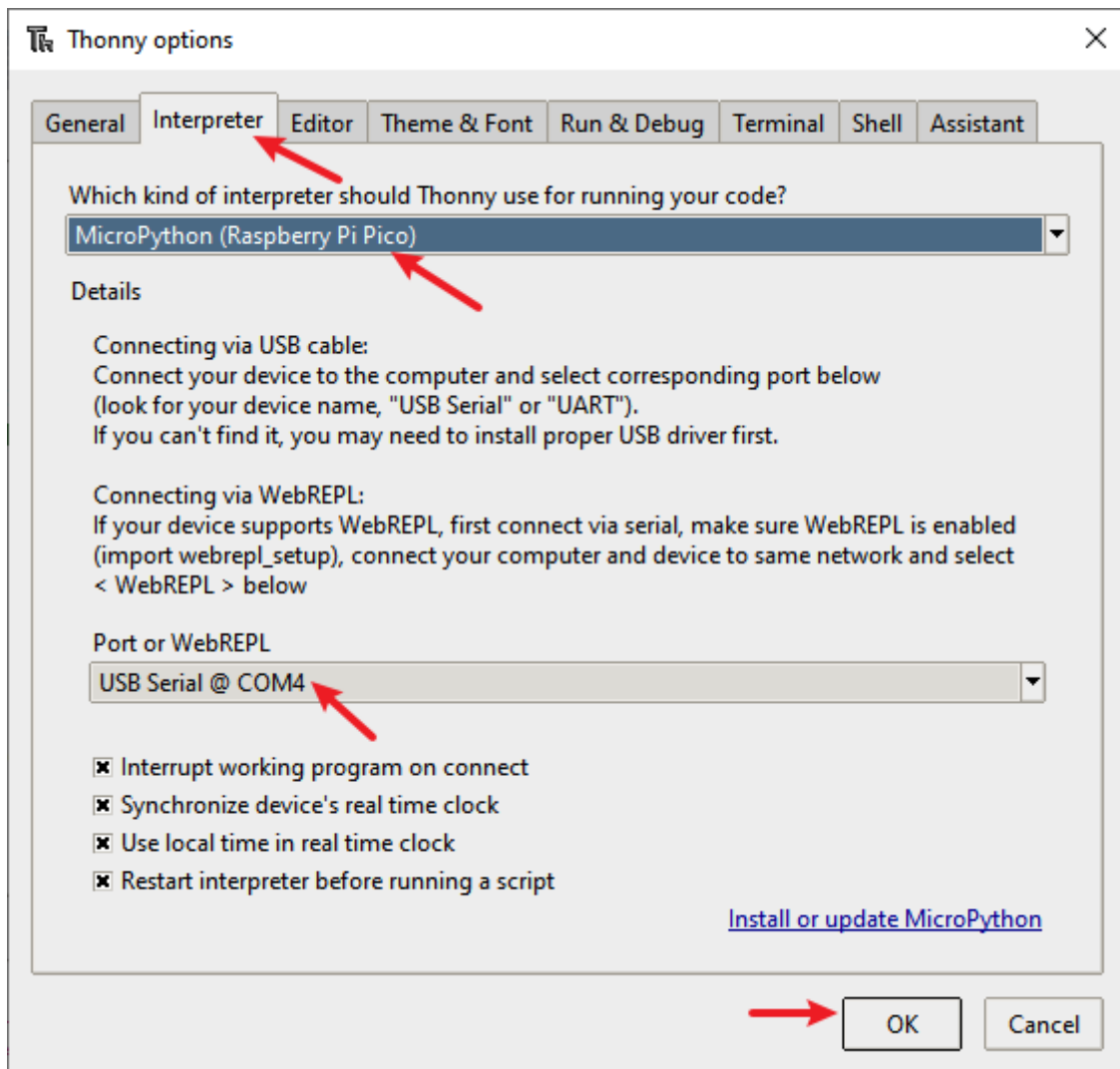
- Connect Pico to the computer using Type-C



- Open the Thonny software, click Tools in the upper left corner, and select Options



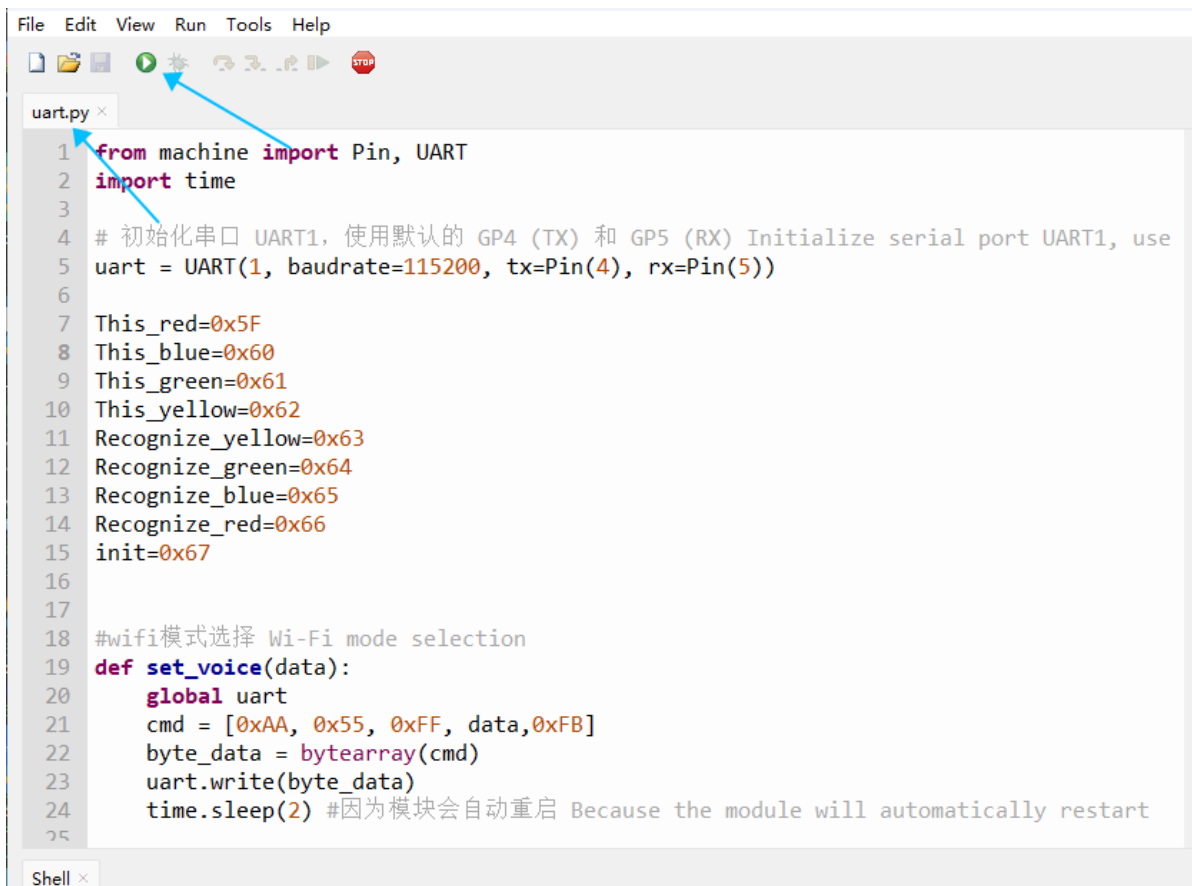
- Select interpreter, select the corresponding serial device in the Port below, and then click OK



- The following picture appears on the terminal, indicating that the connection is correct

```
Shell x
MicroPython v1.24.0-preview.201.g269a0e0e1 on 2024-08-09; Raspberry Pi Pico2 with RP2350
Type "help()" for more information.
>>>
```

- Open the corresponding py file, click the run button above, and hear **I am ready**, which means that the program is running



```
File Edit View Run Tools Help
uart.py x
1 from machine import Pin, UART
2 import time
3
4 # 初始化串口 UART1, 使用默认的 GP4 (TX) 和 GP5 (RX) Initialize serial port UART1, use
5 uart = UART(1, baudrate=115200, tx=Pin(4), rx=Pin(5))
6
7 This_red=0x5F
8 This_blue=0x60
9 This_green=0x61
10 This_yellow=0x62
11 Recognize_yellow=0x63
12 Recognize_green=0x64
13 Recognize_blue=0x65
14 Recognize_red=0x66
15 init=0x67
16
17
18 #wifi模式选择 Wi-Fi mode selection
19 def set_voice(data):
20     global uart
21     cmd = [0xAA, 0x55, 0xFF, data, 0xFB]
22     byte_data = bytearray(cmd)
23     uart.write(byte_data)
24     time.sleep(2) #因为模块会自动重启 Because the module will automatically restart
25
Shell x
```

4. Achieve the effect

- Select the broadcast content by modifying the code in the program as shown below

```
This_red=0x5F
This_blue=0x60
This_green=0x61
This_yellow=0x62
Recognize_yellow=0x63
Recognize_green=0x64
Recognize_blue=0x65
Recognize_red=0x66
init=0x67
```

```
#wifi模式选择 Wi-Fi mode selection
def set_voice(data):
    global uart
    cmd = [0xAA, 0x55, 0xFF, data, 0xFB]
    byte_data = bytearray(cmd)
    uart.write(byte_data)
    time.sleep(2) #因为模块会自动重启 Because the module will automatically restart

try:
    set_voice(init)
    while 1:
        if uart.any():
```

- The content of the broadcast can be viewed according to the **command word broadcast word protocol list V3_EN** file provided in the attachment,

where the first and second bytes are AA FF indicates the frame header of the protocol, the third byte FF indicates the broadcast function, and the fourth is the ID of the broadcast content. Here you can see that "I am ready" is 67 in hexadecimal, so in the program, sending 0x67 to register 0x03 can broadcast the corresponding content. The fifth byte is the end frame.

84	THIS-IS-RED	命令词	this is red	被	AA 55 FF 5F FB	AA 55 FF 5F FB
85	THIS-IS-BLUE	命令词	this is blue	被	AA 55 FF 60 FB	AA 55 FF 60 FB
86	THIS-IS-GREEN	命令词	this is green	被	AA 55 FF 61 FB	AA 55 FF 61 FB
87	THIS-IS-YELLOW	命令词	this is yellow	被	AA 55 FF 62 FB	AA 55 FF 62 FB
88	THERE-IS-YELLOW	命令词	there is yellow	被	AA 55 FF 63 FB	AA 55 FF 63 FB
89	THERE-IS-GREEN	命令词	there is green	被	AA 55 FF 64 FB	AA 55 FF 64 FB
90	THERE-IS-BLUE	命令词	there is blue	被	AA 55 FF 65 FB	AA 55 FF 65 FB
91	THERE-IS-RED	命令词	there is red	被	AA 55 FF 66 FB	AA 55 FF 66 FB
92	I-AM-READY	命令词	i am ready	被	AA 55 FF 67 FB	AA 55 FF 67 FB

- After clicking Run, when I say the wake-up word to wake up, and say "**close lights**", the debugging assistant will reply to receive 0A

```

MicroPython v1.24.0-preview.201.g269a0e0e1 on 2024-08-09; Raspberry Pi Pico2 with RP2350
Type "help()" for more information.
>>> %Run -c $EDITOR_CONTENT

Received: 00
Received: 0a

```

- At this time, you can open the attached **Command Word Broadcast Word Protocol List V3_EN** file to view the "Turn off the lights" protocol

20	CLOSE-LIGHT	命令词	ok light is closed.	主	AA 55 00 0A FB	AA 55 00 0A FB
21	RED-LIGHT-UP	命令词	ok red light is on	主	AA 55 00 0B FB	AA 55 00 0B FB

The first and second bytes are AA FF indicates the frame header of the protocol. The third byte indicates the ID of the ten function words of the chip. The fourth is the command word ID. Here you can see that "**close light**" is hexadecimal 0A. The fifth byte is the end frame.

- If you say other command words, the serial port debugging assistant will also print the corresponding command word ID. You can try it yourself