Pico IIC communication

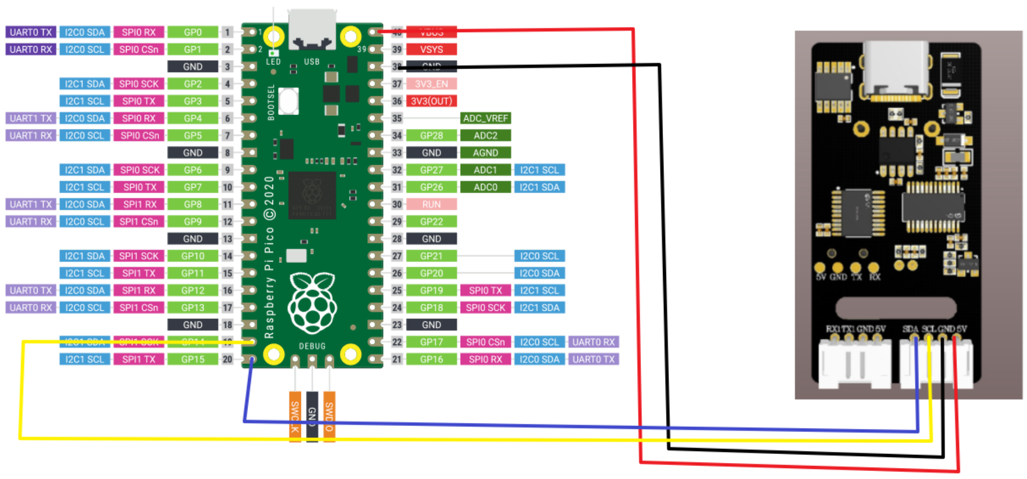
Note: The voice interaction module needs to be burned with factory firmware. If the voice chip has not been flashed with firmware after receiving it, it does not need to be burned

1. Experimental preparation

- Pico
- Voice interaction module
- Dupont line

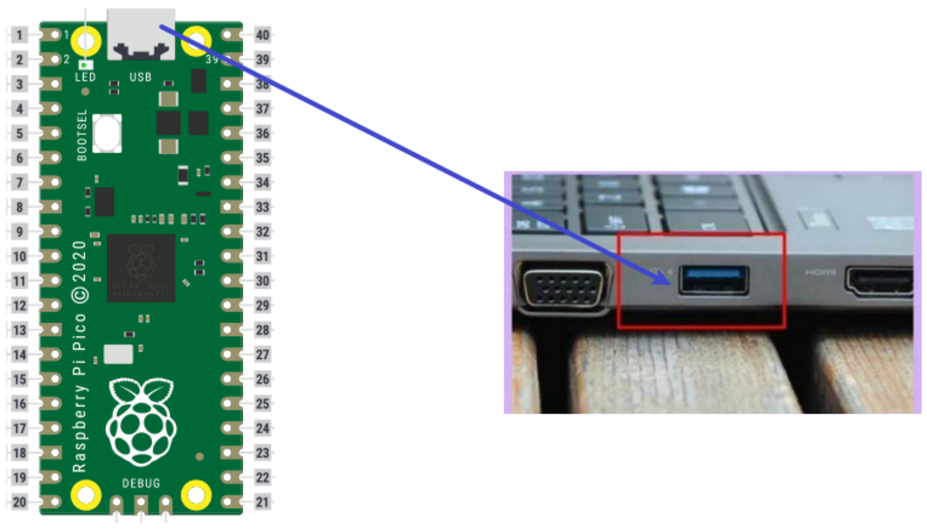
2. Wiring diagram

Pico	Voice interaction module
GP14	SDA
GP15	SCL
GND	GND
5V	5V

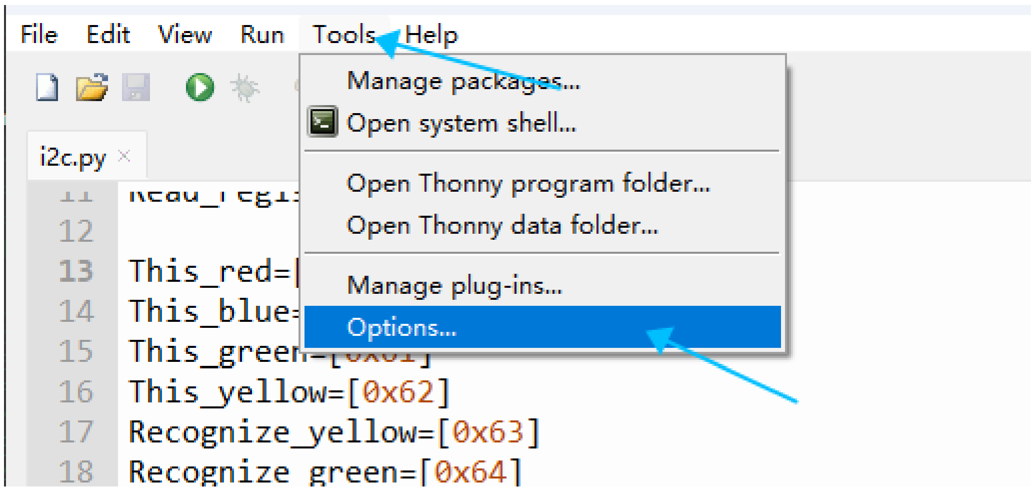


3. Program download

- Connect Pico to the computer using Type-C

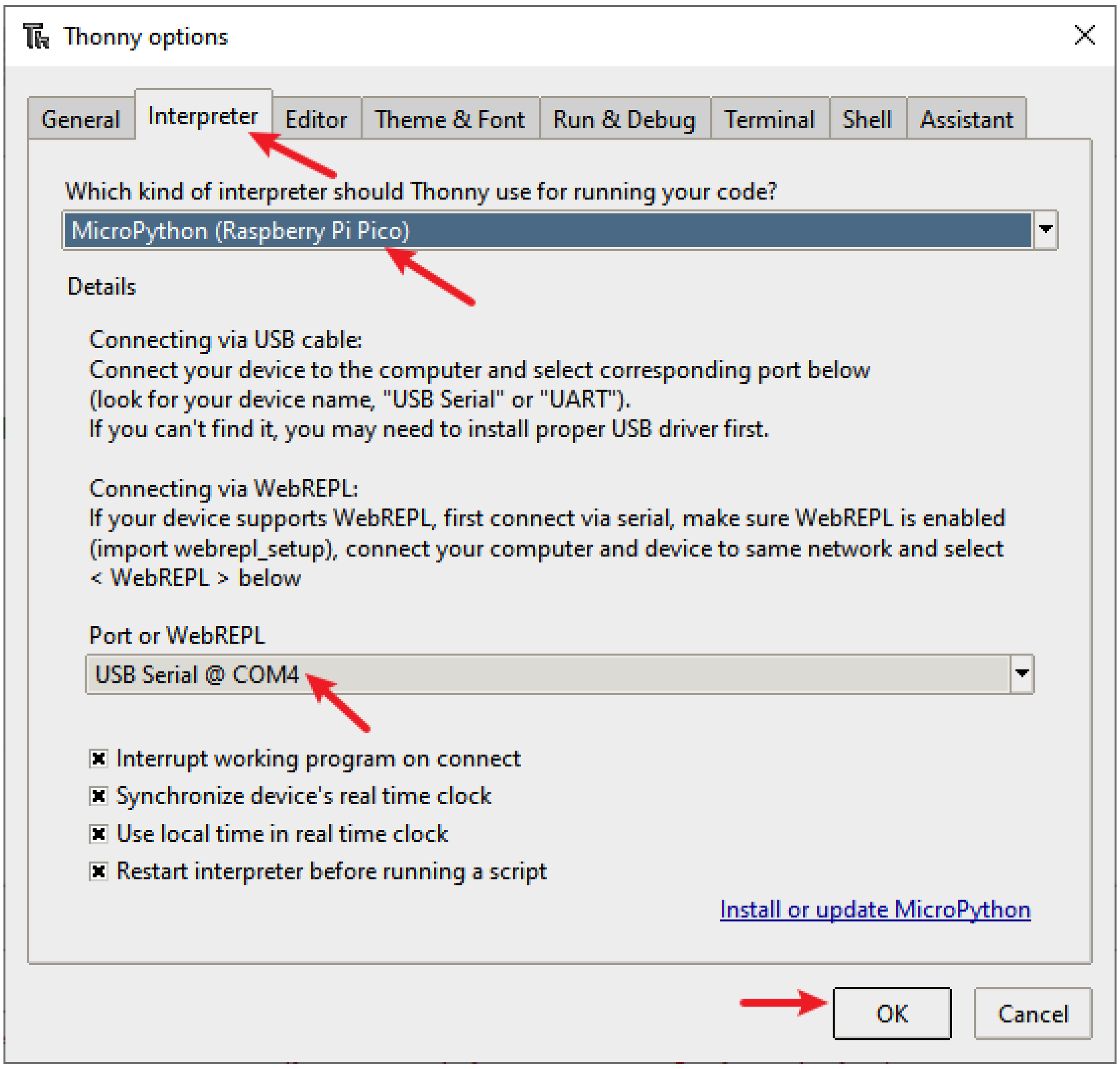


- Open the Thonny software, click Tools in the upper left corner, and select Options



```
This_red=[
This_blue=
This_green=[0x61]
This_yellow=[0x62]
Recognize_yellow=[0x63]
Recognize_green=[0x64]
```

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



Which kind of interpreter should Thonny use for running your code?

MicroPython (Raspberry Pi Pico)

Details

Connecting via USB cable:
Connect your device to the computer and select corresponding port below
(look for your device name, "USB Serial" or "UART").
If you can't find it, you may need to install proper USB driver first.

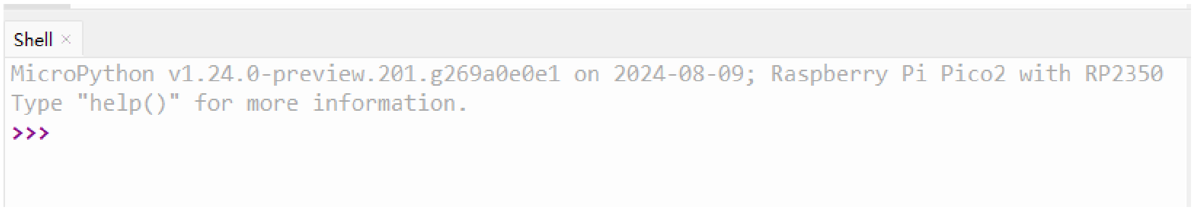
Connecting via WebREPL:
If your device supports WebREPL, first connect via serial, make sure WebREPL is enabled
(import webrepl_setup), connect your computer and device to same network and select
< WebREPL > below

Port or WebREPL

USB Serial @ COM4

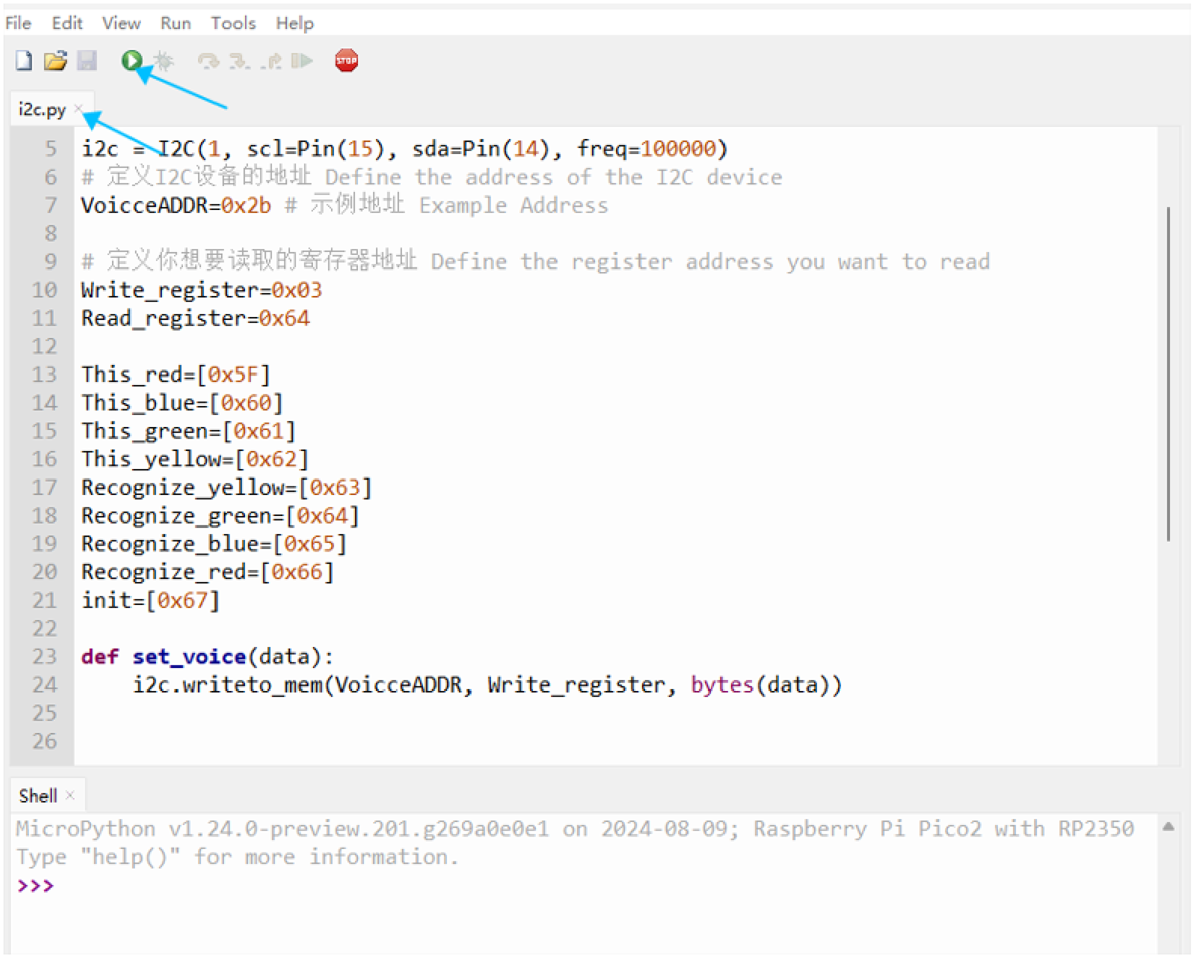
- [x] Interrupt working program on connect
- [x] Synchronize device's real time clock
- [x] Use local time in real time clock
- [x] Restart interpreter before running a script

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



```
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

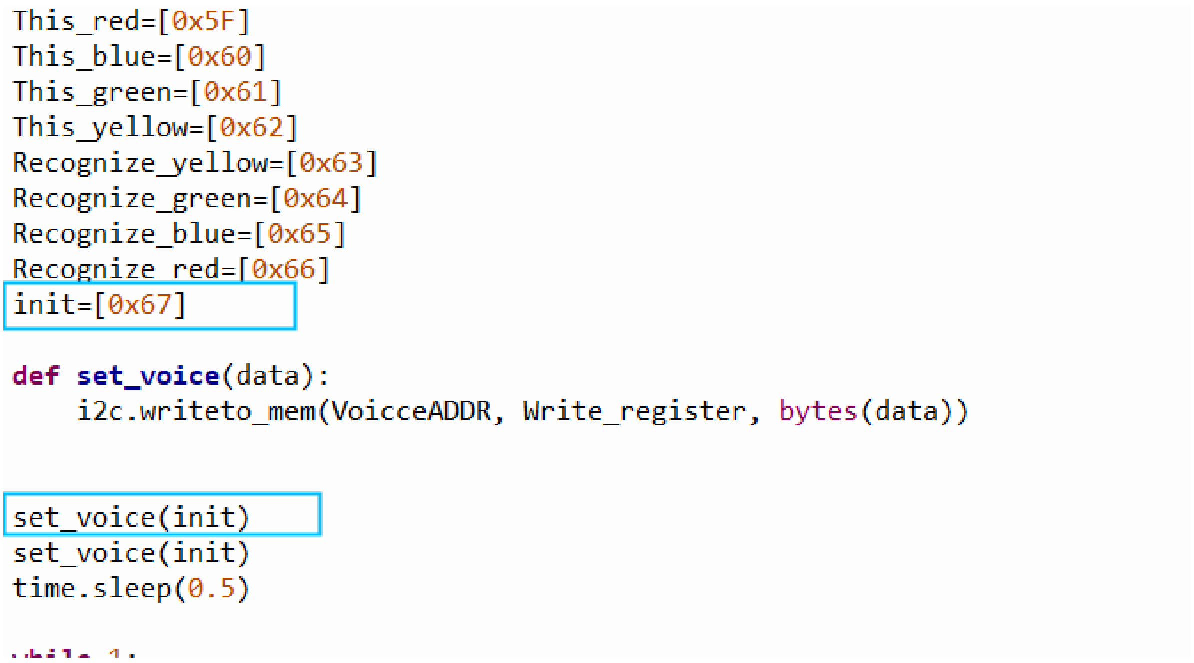


```
i2c = I2C(1, scl=Pin(15), sda=Pin(14), freq=100000)
# 定义I2C设备的地址 Define the address of the I2C device
VoicceADDR=0x2b # 示例地址 Example Address

# 定义你想要读取的寄存器地址 Define the register address you want to read
Write_register=0x03
Read_register=0x64

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]

def set_voice(data):
    i2c.writeto_mem(VoicceADDR, Write_register, bytes(data))
```

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

4.Achieve the effect

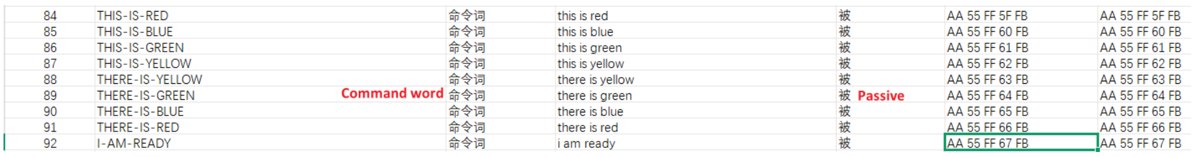
- Select the broadcast content by modifying the code in the program as shown below



```
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]

def set_voice(data):
    i2c.writeto_mem(VoicceADDR, Write_register, bytes(data))


set_voice(init)
set_voice(init)
time.sleep(0.5)
```

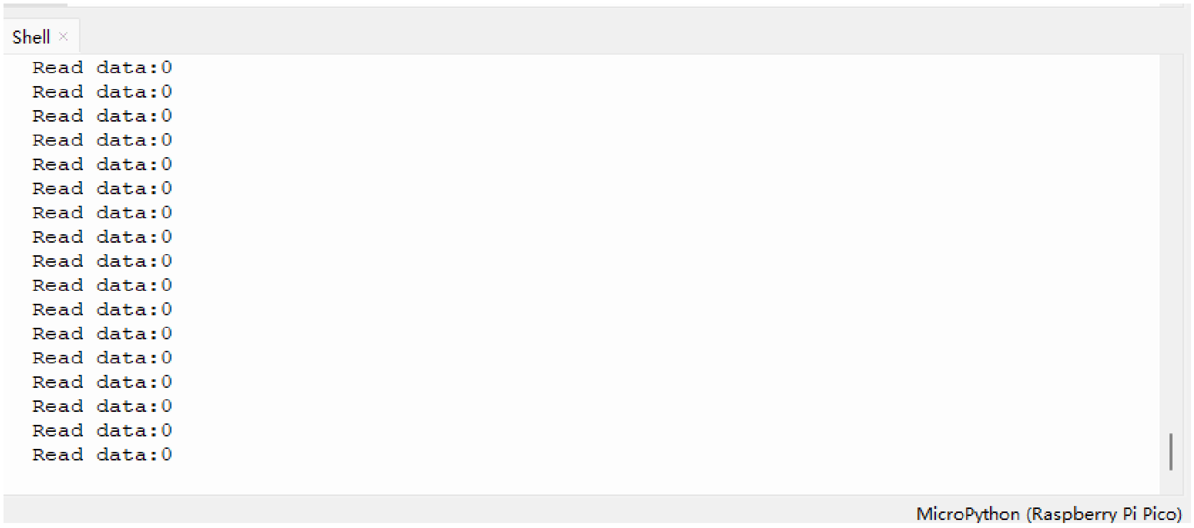
- The content of the broadcast can be viewed according to the **command word broadcast word protocol list V3_EN** file provided in the attachment,

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.



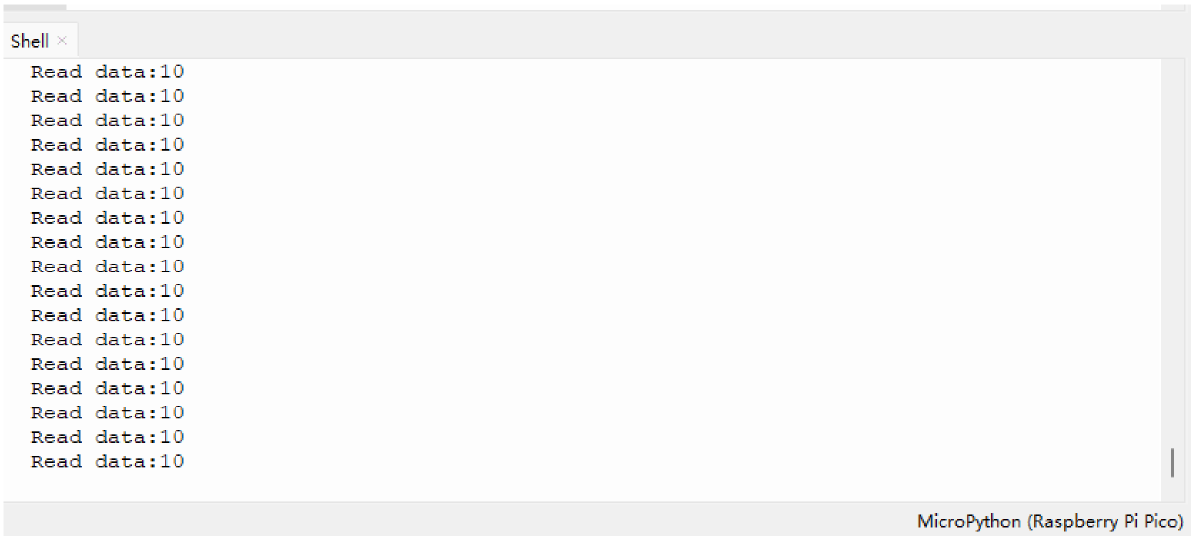
84	THIS-IS-RED		命令词	this is red	被		AA 55 FF 5F FB	AA 55 FF 5F FB
85	THIS-IS-BLUE		命令词	this is blue	被		AA 55 FF 60 FB	AA 55 FF 60 FB
86	THIS-IS-GREEN		命令词	this is green	被		AA 55 FF 61 FB	AA 55 FF 61 FB
87	THIS-IS-YELLOW		命令词	this is yellow	被		AA 55 FF 62 FB	AA 55 FF 62 FB
88	THERE-IS-YELLOW		命令词	there is yellow	被		AA 55 FF 63 FB	AA 55 FF 63 FB
89	THERE-IS-GREEN	Command word	命令词	there is green	被	Passive	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, you can see the terminal print the received command word ID



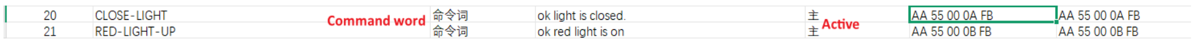
```
Shell
Read data:0
Read data:0
Read data:0
Read data:0
Read data:0
Read data:0
Read data:0
Read data:0
Read data:0
Read data:0
Read data:0
Read data:0
Read data:0
Read data:0
Read data:0
Read data:0
Read data:0
MicroPython (Raspberry Pi Pico)
```

- After I say the wake-up word to wake up, say **"close light"**, the debugging assistant will reply with the received ID: 10



```
Shell
Read data:10
Read data:10
Read data:10
Read data:10
Read data:10
Read data:10
Read data:10
Read data:10
Read data:10
Read data:10
Read data:10
Read data:10
Read data:10
Read data:10
Read data:10
Read data:10
Read data:10
MicroPython (Raspberry Pi Pico)
```

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



20	CLOSE-LIGHT	Command word	命令词	ok light is closed.	主	Active	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 AA FF represent the frame header of the protocol, the third byte represents the ID of the ten function words of the chip, and the fourth is the command word ID. Here you can see **"close light"** is hexadecimal 0A, so decimal will return 10. The fifth byte is the end frame.

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