# Arduino 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

- Arduino development board
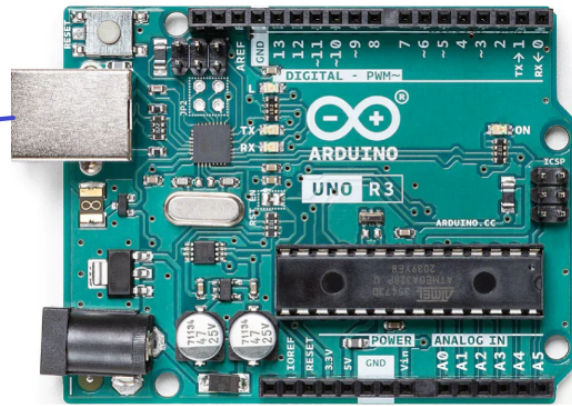- Voice interaction module
- Multiple Dupont wires

2. Wiring diagram

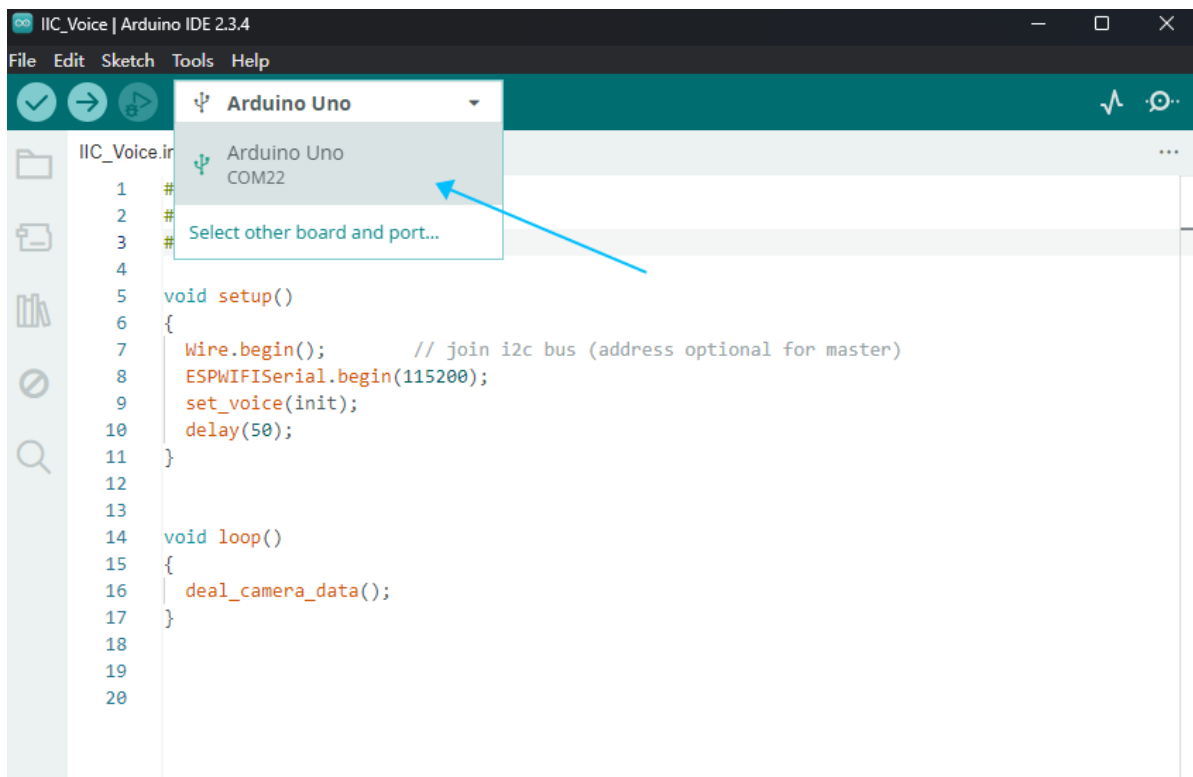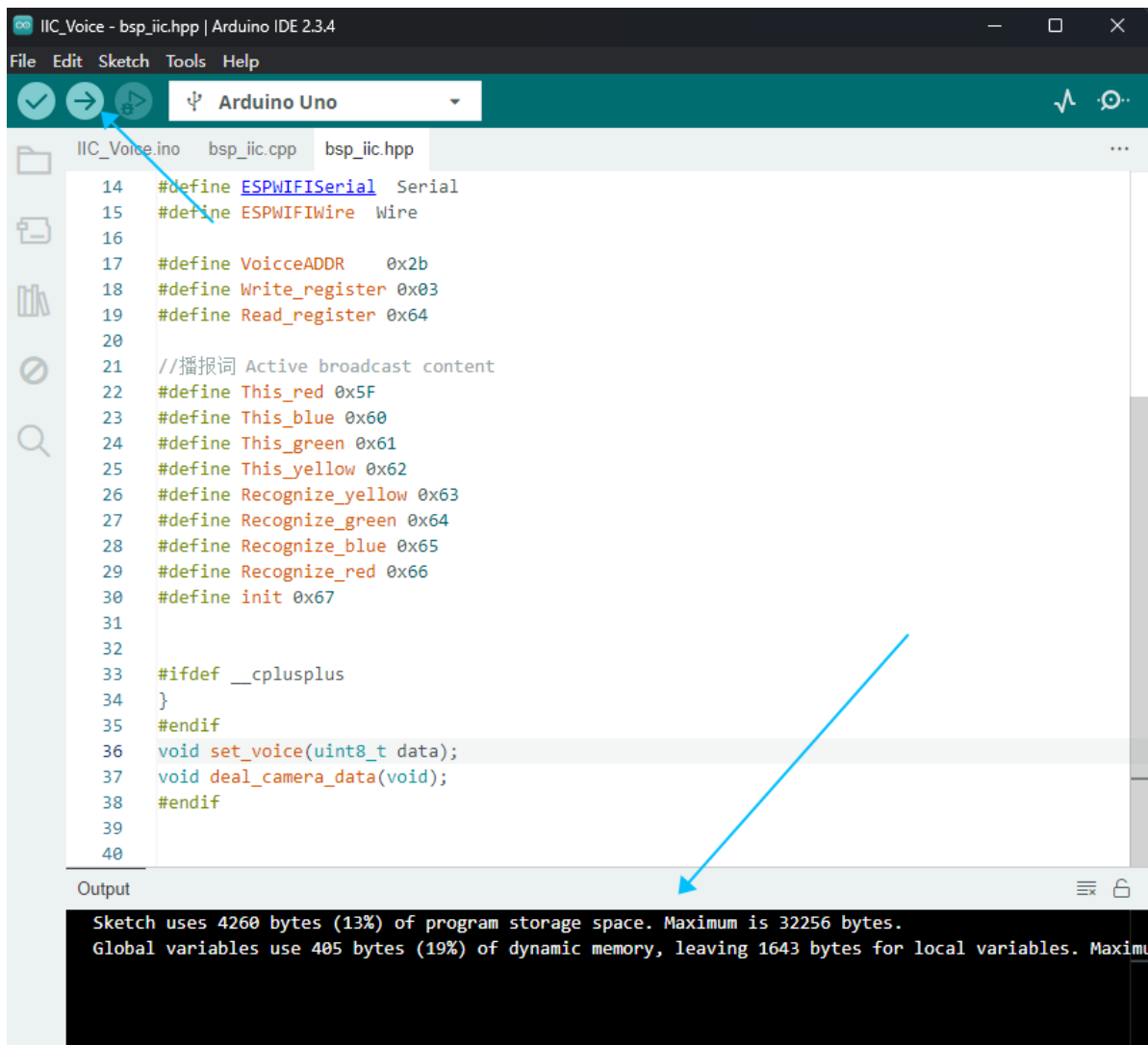| Arduino	Voice interaction module
SCL	SCL
SDA	SDA
GND	GND
5V	5V
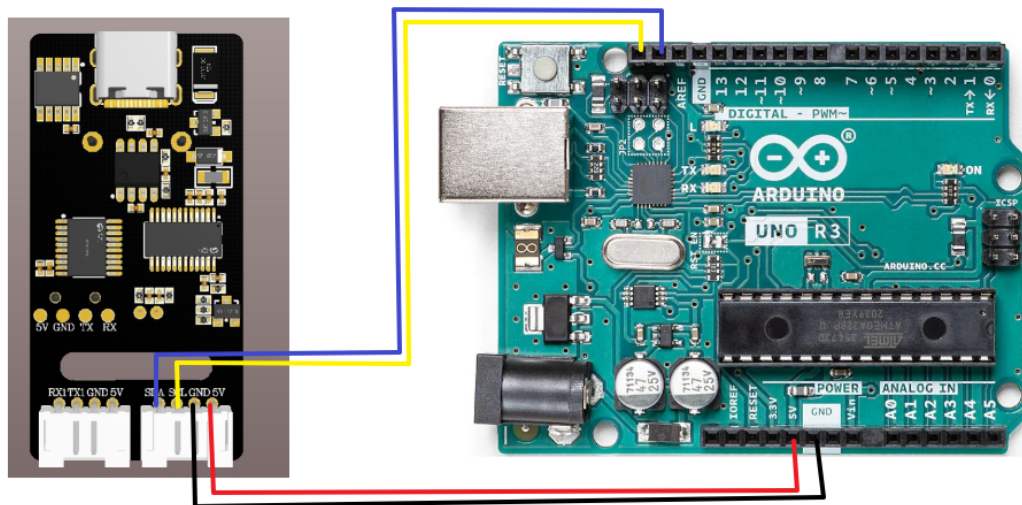
3. Program download

- Connect Arduino to the computer using a download cable

- Open the ArduinoIDE compilation platform, and you can see the serial port number, which is normal.

- Click the right arrow in the upper right corner to download the program, and you can see that the compilation is successful below

```
14    #define ESPWIFISerial   Serial
15    #define ESPWIFIWire   Wire
16
17    #define VoicceADDR      0x2b
18    #define Write_register 0x03
19    #define Read_register 0x64
20
21    //播报词 Active broadcast content
22    #define This_red 0x5F
23    #define This_blue 0x60
24    #define This_green 0x61
25    #define This_yellow 0x62
26    #define Recognize_yellow 0x63
27    #define Recognize_green 0x64
28    #define Recognize_blue 0x65
29    #define Recognize_red 0x66
30    #define init 0x67
31
32
33    #ifdef __cplusplus
34    }
35    #endif
36    void set_voice(uint8_t data);
37    void deal_camera_data(void);
38    #endif
39
40
```

Output

```
Sketch uses 4260 bytes (13%) of program storage space. Maximum is 32256 bytes.
Global variables use 405 bytes (19%) of dynamic memory, leaving 1643 bytes for local variables. Maximu
```

- Press the reset button and you can hear **"I am ready"**, indicating that the program has been burned in

4.Achievement effect

- You can select the broadcast content by modifying the code in the program as shown below

```
void setup()
{
  Wire.begin();          // join i2c bus (address optional for master)
  ESPWIFISerial.begin(115200);
  set_voice(init);
  delay(50);
}
```

```
//播报词 Active broadcast content
#define This_red 0x5F
#define This_blue 0x60
#define This_green 0x61
#define This_yellow 0x62
#define Recognize_yellow 0x63
#define Recognize_green 0x64
#define Recognize_blue 0x65
#define Recognize_red 0x66
#define init 0x67
```

- The content of the report can be viewed according to the **Command Word Broadcast Word Protocol List V3_EN** file provided in the attachment.
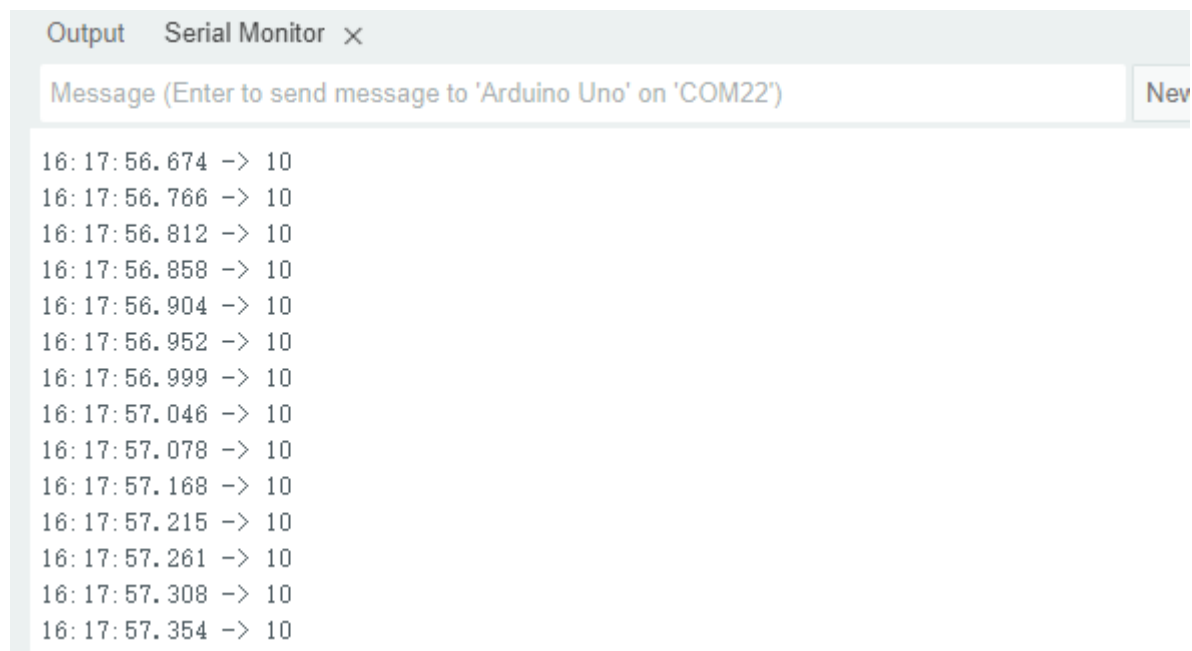
The first and second bytes AA FF represent the frame header of the protocol, the third byte FF represents the broadcast function, and the fourth is the ID of the broadcast content. Here you can see **"I am ready"** is 67 in hexadecimal, so in the program, send 0x67 to register 0x03 to broadcast the corresponding 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

- Open the IDE serial port, and the received command word ID will be printed
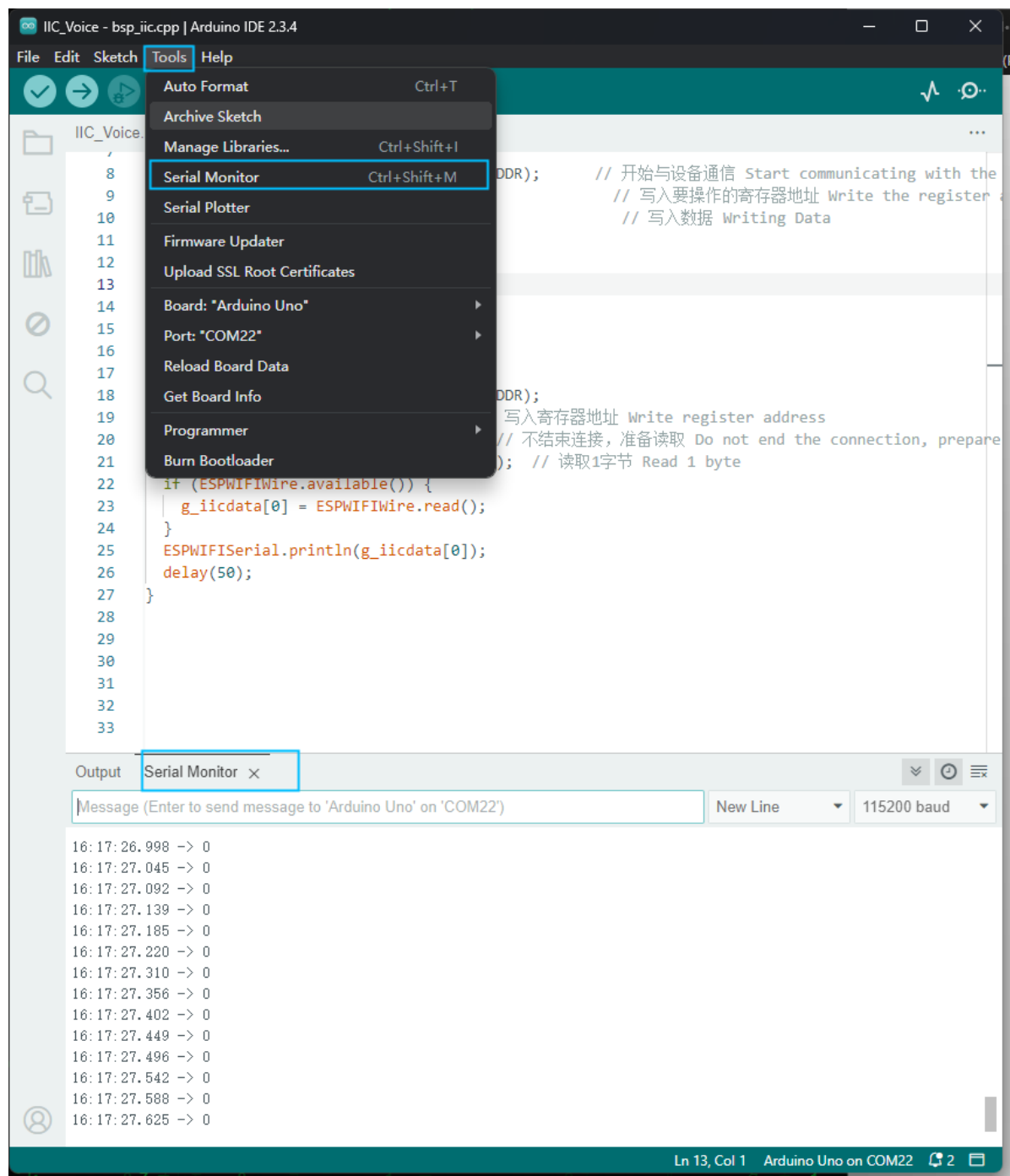
- After I say the wake-up word to wake up, say **"close light"**, the debugging assistant will reply with the received ID: 10

```
16:17:56.674 -> 10
16:17:56.766 -> 10
16:17:56.812 -> 10
16:17:56.858 -> 10
16:17:56.904 -> 10
16:17:56.952 -> 10
16:17:56.999 -> 10
16:17:57.046 -> 10
16:17:57.078 -> 10
16:17:57.168 -> 10
16:17:57.215 -> 10
16:17:57.261 -> 10
16:17:57.308 -> 10
16:17:57.354 -> 10
```

- 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