

9 External interface experimental

9 External interface experimental

9.1 the experimental goals

9.2 experimental procedure

9.3 experimental results

9.4 the experiments are summarized

9.1 the experimental goals

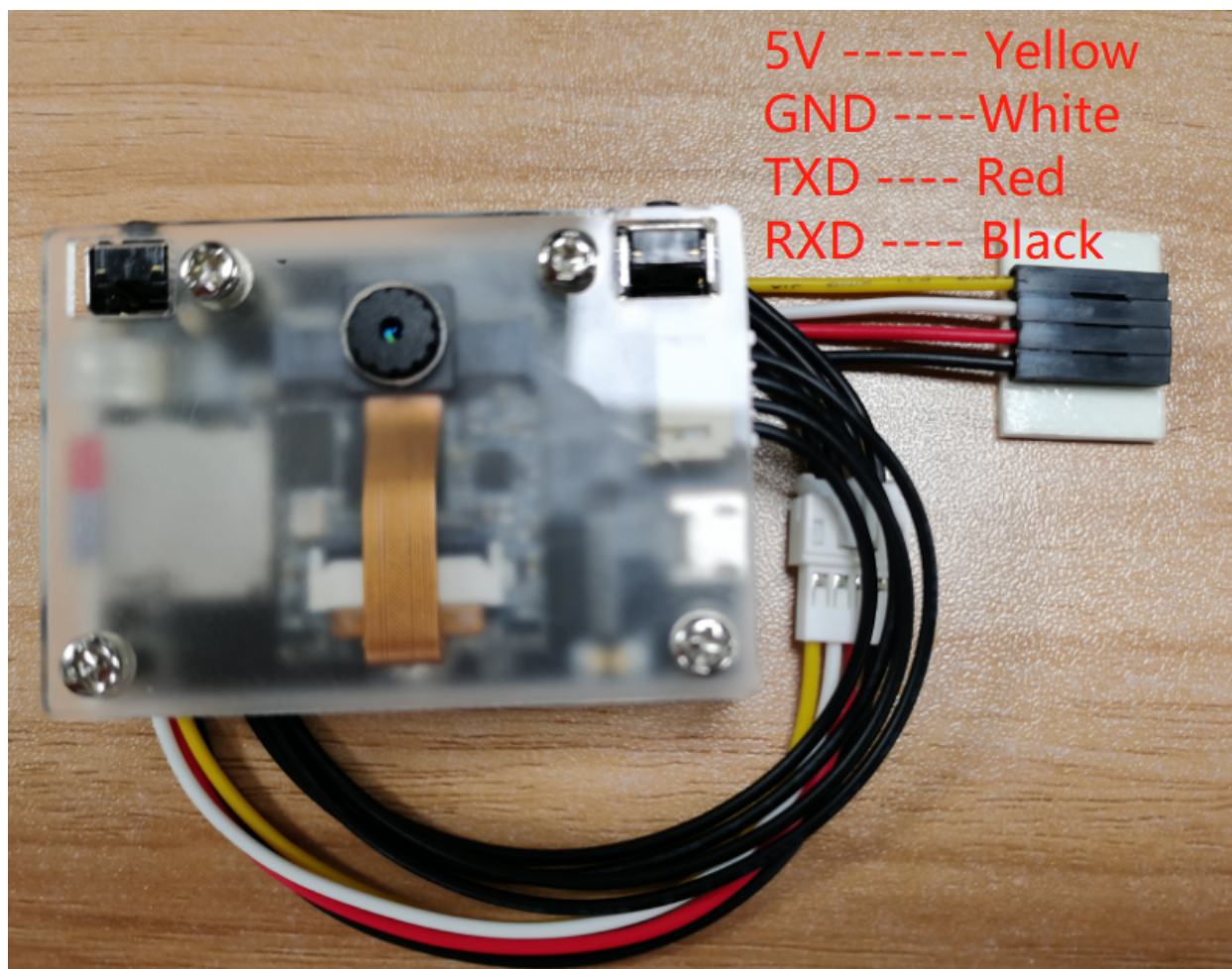
In this lesson, you will learn how to communicate with microPython using an external serial port.

The reference code path for this experiment : CanMV\03-Hardware\serial.py

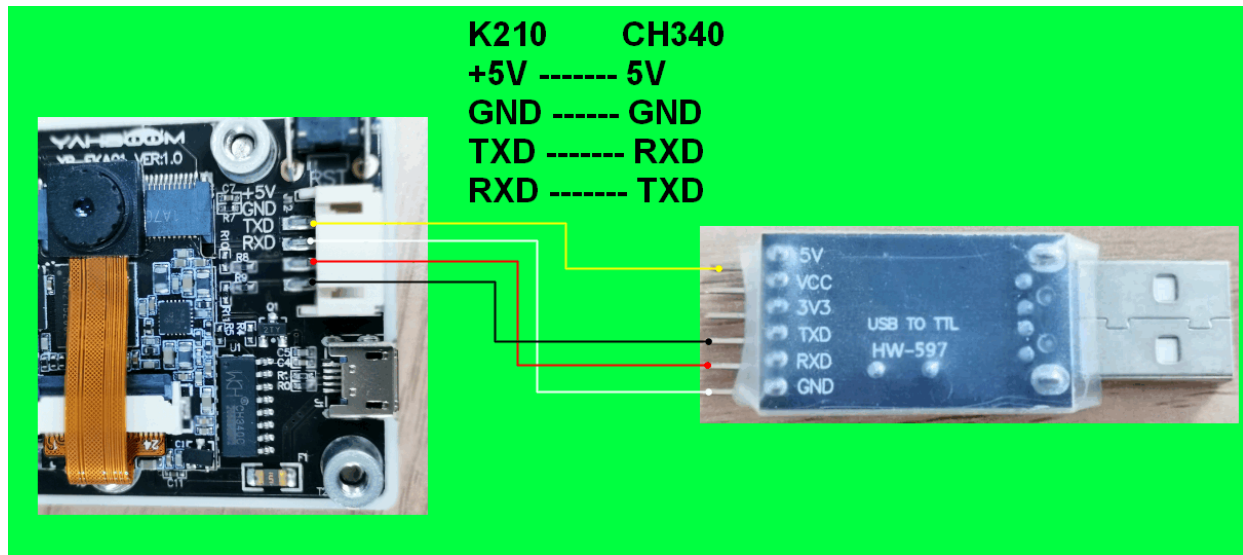
9.2 experimental procedure

The factory firmware of the module has been integrated with the external serial port control module. If you have downloaded other firmware, please burn it back to the factory firmware before doing the experiment.

After using the matching wire connection, the corresponding color of the external pin is:



The CH340 module needs to be used to connect the external interface of the module, and the pin connection is shown in the following figure.



1. Import ybserial from modules.

```
from modules import ybserial
import time
```

2. Create an object of ybserial and call it serial

```
serial = ybserial()
```

3. Through the external serial port character 1 and newline character.

```
serial.send_byte(0x31)
serial.send_byte(0x0D)

array = [0x30, 0x31, 0x32, 0x33, 0x0D]
serial.send_bytearray(array)
```

Wherein, `serial.send_byte(data)`: Means to send one byte of data through the external serial port. The 'data' parameter ranges from 0-255.

`serial.send_bytearray(array)`: Send a byte array through the external serial port. The argument 'array' is an array of bytes.

4. 通过外置串口发送“Hello Yahboom”字符串。

```
text = 'Hello Yahboom'
num = serial.send(text)
print("num:", num)
```

Wherein, `num = serial.send(string)`: 表示通过外置串口发送字符串，返回值是字符串的长度。

5. Create two command strings to send alternately, with count representing the number of times to send.

```
num = 0
count = 0
CMD_1 = "$A#"
CMD_2 = "$BB#"
```

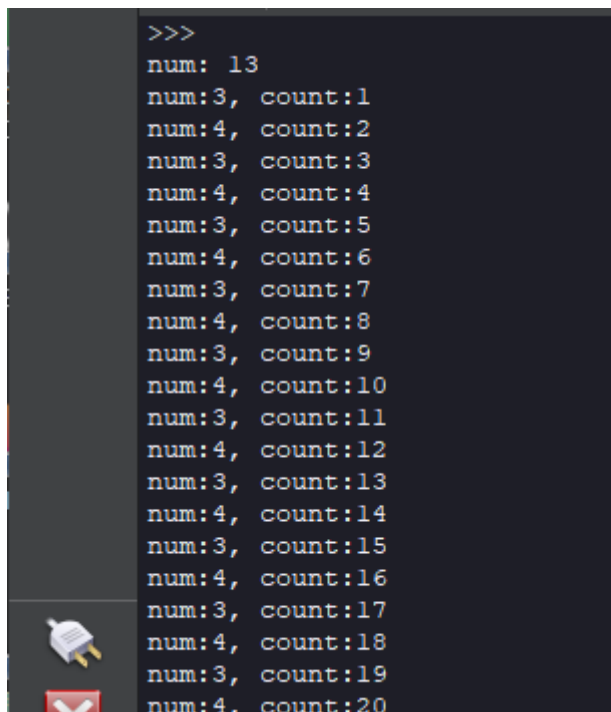
6. Create a new while loop, the count is automatically added and the data is sent once every second. When the count is odd, the command CMD_1 is sent, and when the count is even, the command CMD_2 is sent.

```
while True:
    time.sleep_ms(1000)
    count = count + 1
    if count % 2 == 1:
        num = serial. send(CMD_1)
    else:
        num = serial. send(CMD_2)
    print("num:%d, count:%d" % (num, count))
```

9.3 experimental results

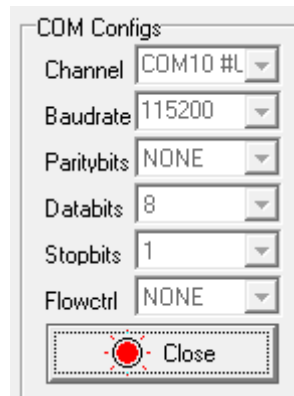
Connect the K210 module to the computer through the microUSB data cable, CanMV IDE click the connect button, after the connection is completed click the Run button to run the routine code. You can also download the code as main.py and run it in the K210 module.

Open the IDE at the bottom of the `Serial Terminal` That you can see every second print data once.

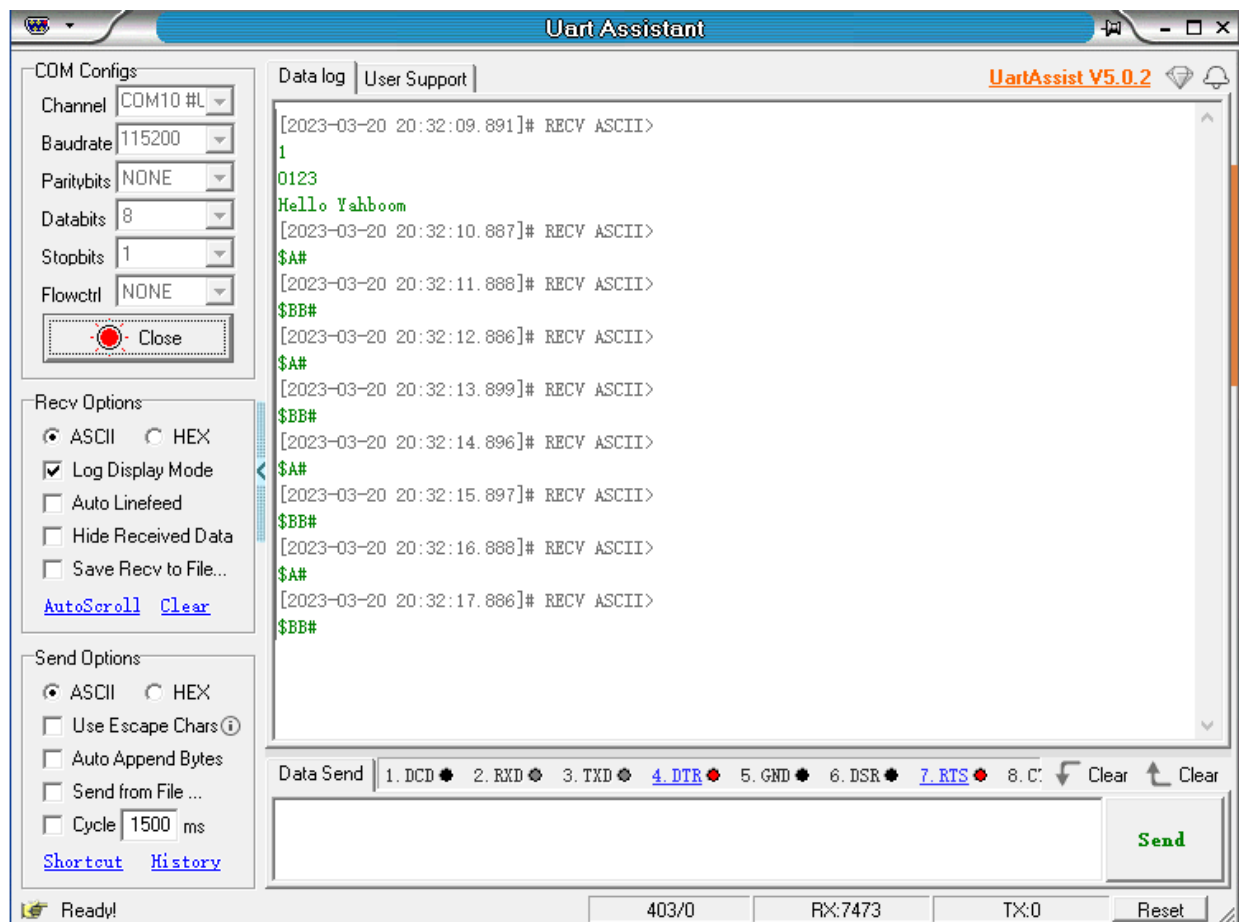


```
>>>
num: 13
num:3, count:1
num:4, count:2
num:3, count:3
num:4, count:4
num:3, count:5
num:4, count:6
num:3, count:7
num:4, count:8
num:3, count:9
num:4, count:10
num:3, count:11
num:4, count:12
num:3, count:13
num:4, count:14
num:3, count:15
num:4, count:16
num:3, count:17
num:4, count:18
num:3, count:19
num:4, count:20
```

Open the CH340 is connected to the serial port assistant, serial assistant configured as follows: baud rate 115000, no parity bit, 8 bits data, 1 bit stop, no flow control.



You can see the serial port assistant every 1 second received sentence of command.



9.4 the experiments are summarized

Use CanMV IDE, with the factory firmware write a good MicroPython syntax, the default already configured an external serial port, you'll need to initialize the serial port object can be used.

CH340 module needs to be manually connected to the line, insert the computer will produce another port, you need to and K210 port distinguish.