TCP-Client Example

Routine explanation

> If network-related routines run abnormally, please try to turn off the system firewall. Since the operation of different systems is different, I will not introduce it in detail here. You can search on Baidu for [How to turn off the firewall in xxx system]

Introduction

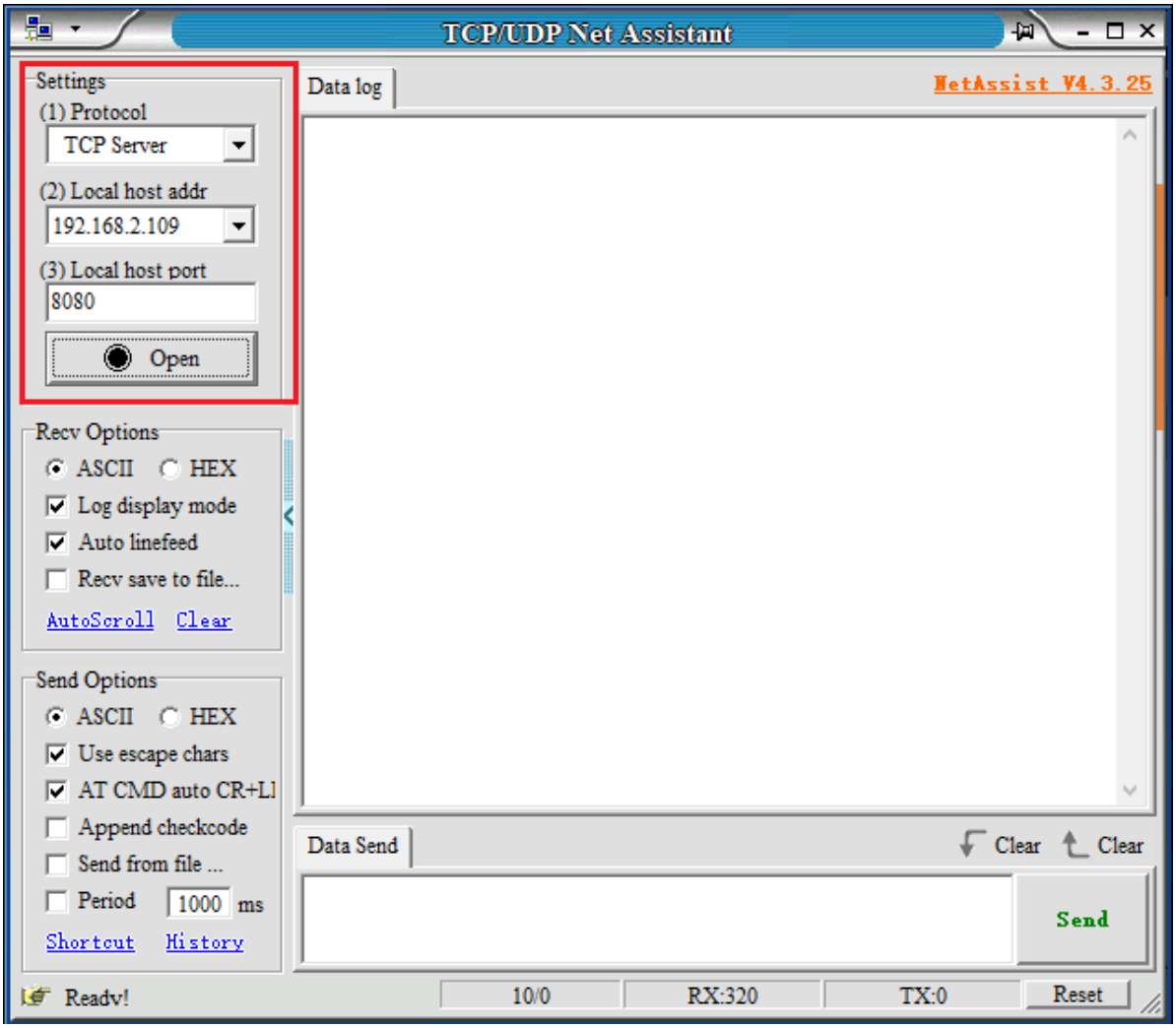
In this section, we use K230 as a TCP client.

We open the example code in this section and modify the WIFI connection information part in the code to your home WIFI or mobile hotspot

```
def  run_tcp_client ():
    ##########################################
    if  not  connect_wifi ( "Yahboom2" , "yahboom890729" ):
        return
    ##########################################
```

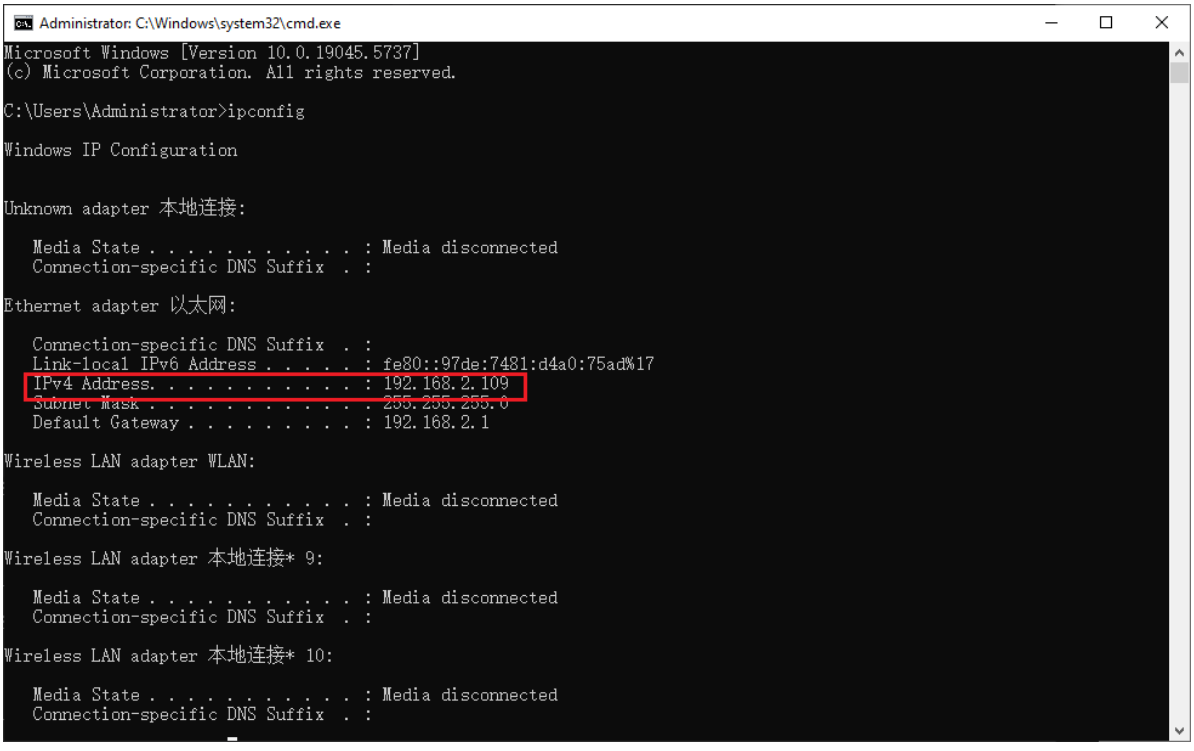
Then we open the network debugging assistant NetAssist.exe in the supporting tools

Enter the information in the area on the upper left corner and click Open



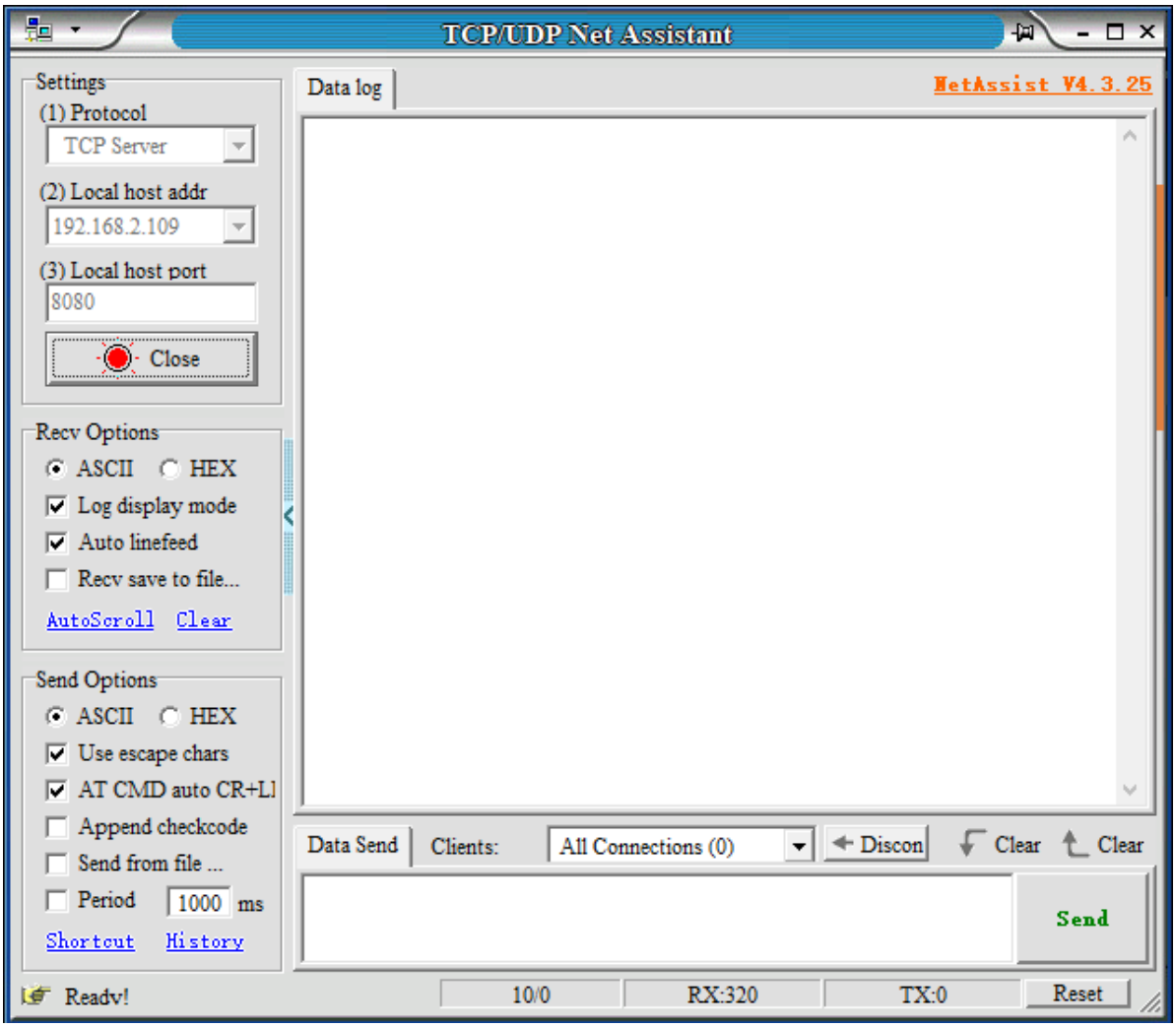
Note that the local host address here can be viewed by executing the ipconfig command in the command prompt. The operation methods for each system version are different. You can search for how to open the command prompt for your own computer version. Here we take Windows 11 as an example

After opening it, if you are connected via a network cable, find [Ethernet Adapter Ethernet] and find the [IPv4 Address] in this column. This is the local host address we need to fill in.



Similarly, if you are connected to WIFI, find the [IPv4 Address] in the [Wireless LAN Adapter WLAN] section.

Then we go back to the network debugging assistant and make sure the current server is turned on:



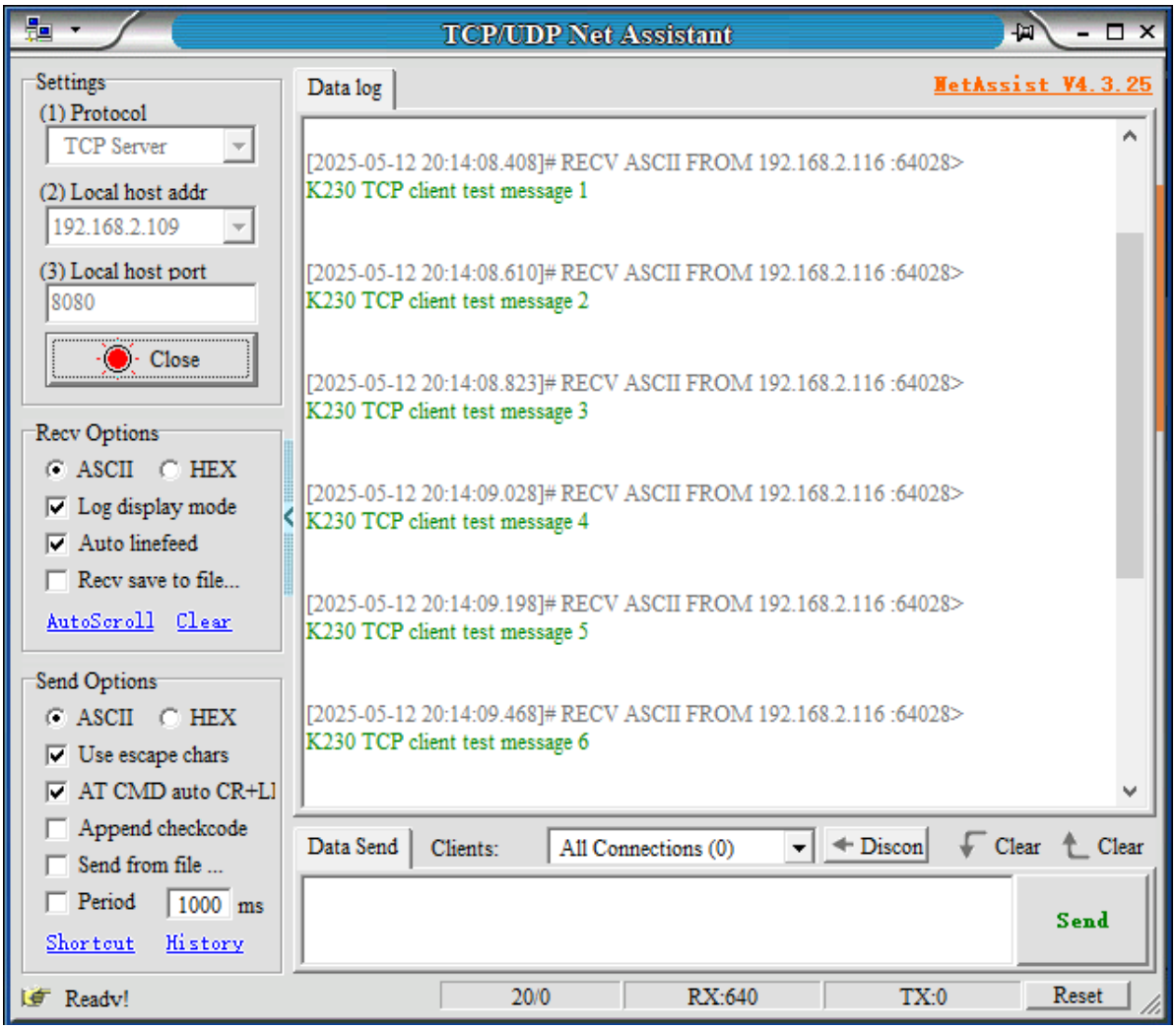
Then we go back to our example in CanMV IDE, find the server details, and change server_ip to the IP of the server we just started

```
# Server details / 服务器详情
server_ip = "192.168.2.83"
server_port = 8080
```

After completion, we click Run and you can see the following output

```
Connected! IP: 192.168.2.87 / 已连接! IP地址: 192.168.2.87
Connecting to 192.168.2.83:8080 / 正在连接到 192.168.2.83:8080
Sending message 0 / 正在发送消息 0
Sending message 1 / 正在发送消息 1
Sending message 2 / 正在发送消息 2
Sending message 3 / 正在发送消息 3
Sending message 4 / 正在发送消息 4
Sending message 5 / 正在发送消息 5
Sending message 6 / 正在发送消息 6
Sending message 7 / 正在发送消息 7
Sending message 8 / 正在发送消息 8
Sending message 9 / 正在发送消息 9
Connection closed / 连接已关闭
```

At the same time, the network debugging assistant also displays relevant content



Code Explanation

For the complete code and bilingual comments, please refer to [Source Code/11.Network/03.tcp/tcp_client.py]

1. Import module part:

```
import network  # For WiFi functionality / 用于WiFi功能
import socket  # For TCP/IP communications / 用于TCP/IP通信
import os      # For system functions / 用于系统功能
import time    # For adding delays / 用于添加延时
```

Four key modules are imported here:

- network: used for WiFi connection management
- Socket: Provides TCP/IP network communication function
- os: provides system-related functions
- time: Provides delay function

2. WiFi connection function:

```
def connect_wifi(ssid, password):
    """
    Connect to WiFi network with given SSID and password
    连接到指定SSID和密码的WiFi网络
    """
```

This function implements the WiFi connection function:

- Use network.WLAN(network.STA_IF) to initialize WiFi to station mode
- Use wifi.connect() to connect to the specified WiFi network
- Contains a retry mechanism, with a maximum of 10 connection attempts
- Returns the IP address after successful connection, returns None if failed

Main process:

- Initialize WiFi
- Try to connect
- Wait for the connection to complete (up to 10 seconds)
- Return connection result

3. TCP client main function:

```
def run_tcp_client():
    """
    Run TCP client to send test messages
    运行TCP客户端发送测试消息
    """
```

This function implements the core functionality of the TCP client:

- First call connect_wifi() to connect to WiFi
- Creating a TCP Socket
- Connect to the specified server
- Send 10 test messages
- Handle exceptions and ensure resources are released correctly

Key steps:

- Create a socket: `socket.socket(socket.AF_INET, socket.SOCK_STREAM)`
  - AF_INET indicates the use of IPv4
  - SOCK_STREAM indicates the use of TCP protocol
- Connect to the server: `sock.connect((server_ip, server_port))`
- Send a message: `sock.send(message.encode())`
- Use try-finally to ensure the socket is closed properly

4. Main program entry:

```
if __name__ == "__main__":
    run_tcp_client()
```

This is the entry point of the program, and when the script is run directly it will execute the run_tcp_client() function.

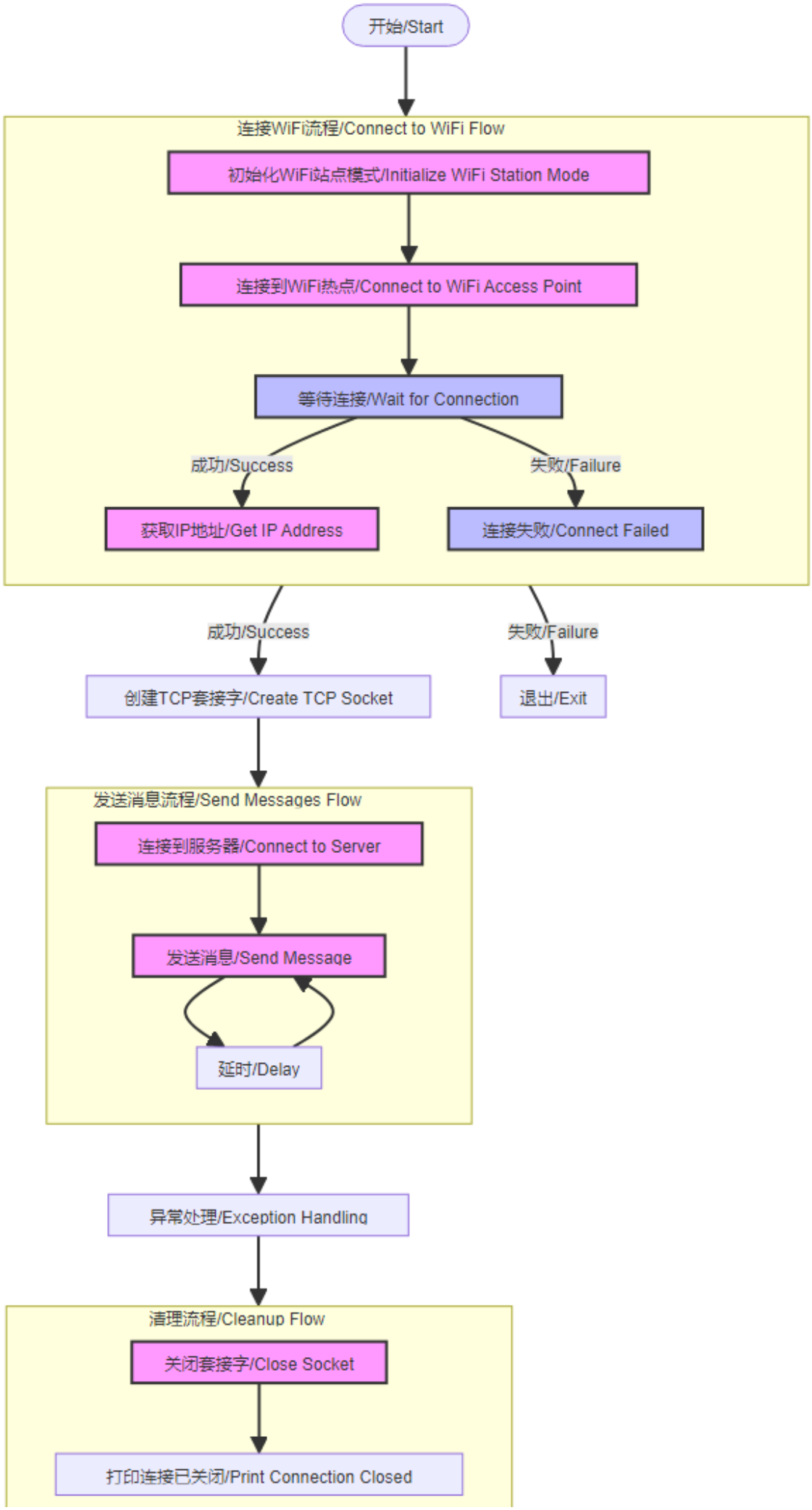
5. Error handling and resource management:

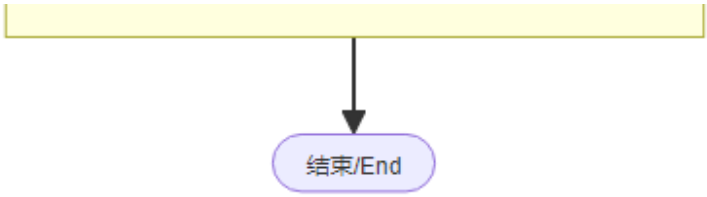
- Use try-except to catch possible network exceptions
- Use a finally block to ensure that the socket is always closed
- Have proper error handling when WiFi connection fails

Things to note when using the code:

1. Need to change the WiFi SSID and password
2. You need to set the correct server IP address and port
3. Make sure the target server is running and accepting connections
4. Consider the stability of the network environment

flow chart


开始/Start
连接WiFi流程/Connect to WiFi Flow
初始化WiFi站点模式/Initialize WiFi Station Mode
连接到WiFi热点/Connect to WiFi Access Point
等待连接/Wait for Connection
成功/Success
失败/Failure
获取IP地址/Get IP Address
连接失败/Connect Failed
成功/Success
失败/Failure
创建TCP套接字/Create TCP Socket
退出/Exit
发送消息流程/Send Messages Flow
连接到服务器/Connect to Server
发送消息/Send Message
延时/Delay
异常处理/Exception Handling
清理流程/Cleanup Flow
关闭套接字/Close Socket
打印连接已关闭/Print Connection Closed


结束/End