2-CH UART TO ETH

From Waveshare Wiki
Jump to: navigation, search

Overview

2-CH UART TO ETH

(http://www.waveshare.com/2-CH-UART-TO-ETH.htm?amazon)

2-CH Serial UART TTL to Ethernet Converter

Introduction

The 2-CH UART TO ETH provides an easy way to communicate between UART TTL and RJ45 Ethernet, allowing bi-directional transparent data transmission. It is easy to use, along with features like fast data rate, low power consumption, high stability, suits applications such as safety & security, IoT, and so on.

More (http://www.waveshare.com/2-CH-UART-TO-ETH.htm)

Features

- Embedded Ethernet MAC and PHY layers.
- Bi-direction transparent data transmission between UART and Ethernet.
- 10/100M, full-duplex/half-duplex auto-negotiation Ethernet interface, 802.3-compliant.
- Auto-MDI/MDIX, detect and switch cable type automatically.
- Supports DHCP auto-obtained IP and DNS domain access.
- Network parameter configuration like chip operating mode, port, IP, via host computer software or UART command.
- Four operating modes: TCP CLIENT, TCP SERVER, UDP CLIENT, UDP SERVER.
- 2-CH UART concurrent operation, standalone transparent transmission for each channel.
- Full-duplex or half-duplex UART communication supports RS485 RX/TX auto switch.
- Supports virtual serial port software (provided).
- KEEPALIVE mechanism support.

Specifications

- Operating voltage: 3.3V/5V
- Operating current: 140mA
- Operating mode: TCP/UDP
- UART baud rate: 300bps ~ 921.6Kbps
- UART TTL level: 3.3V/5V compatible

- Storage temperature: -55°C ~ 125°C
- Operating temperature: -40°C ~ 85°C
- Dimensions: 53.00 x 22.00mm

Pinout

5V/3V3	power positive, use any pin of 5V/3.3V pins for power supply
GND	ground
RST1	external reset input, low active
RESET	restore to factory setting, detected when chip power on, low active
TXD1	asynchronous UART 1 data output (enabled by default)
TXD2	asynchronous UART 2 data output (disabled by default)
RXD1	asynchronous UART 1 data input (enabled by default)
RXD2	asynchronous UART 2 data input (disabled by default)
CFG0	UART config mode, low level to enter configuration, high level to exit configuration
RUN	Operating status indicating pin
DIR1	UART 1 RS485 RX/TX switch (connects to external RS485 controller)
DIR2	UART 2 RS485 RX/TX switch (connects to external RS485 controller)

2-CH-UART-TO-ETH-4.jpg (/wiki/File:2-CH-UART-TO-ETH-4.jpg)

CH9121

CH9121 is the chip for network serial port transparent transmission, which can realize the two-way transparent transmission of serial data and network data. It supports 4 working modes such as TCP CLIENT, SERVER, UDP CLIENT, and SERVER, and the serial port baud rate is 300bps 921600bps. Before use, it is necessary to configure the network parameters and serial port parameters of the chip through the host computer software

NetModuleConfig.exe or serial commands. After the configuration is completed, CH9121 will save the parameters to the internal storage space. After the chip is reset, CH9121 will work according to the saved configuration values.

Default configuration

The UART2 is disabled, UART1 works in TCP CLIENT mode by default. The default setting of ETH is as below:

- Device IP: 192.168.1.200
- Subnet Mask: 255.255.255.0
- Gateway: 192.168.1.1
- Device Port: 2000
- Target IP: 192.168.1.100
- Target port: 1000

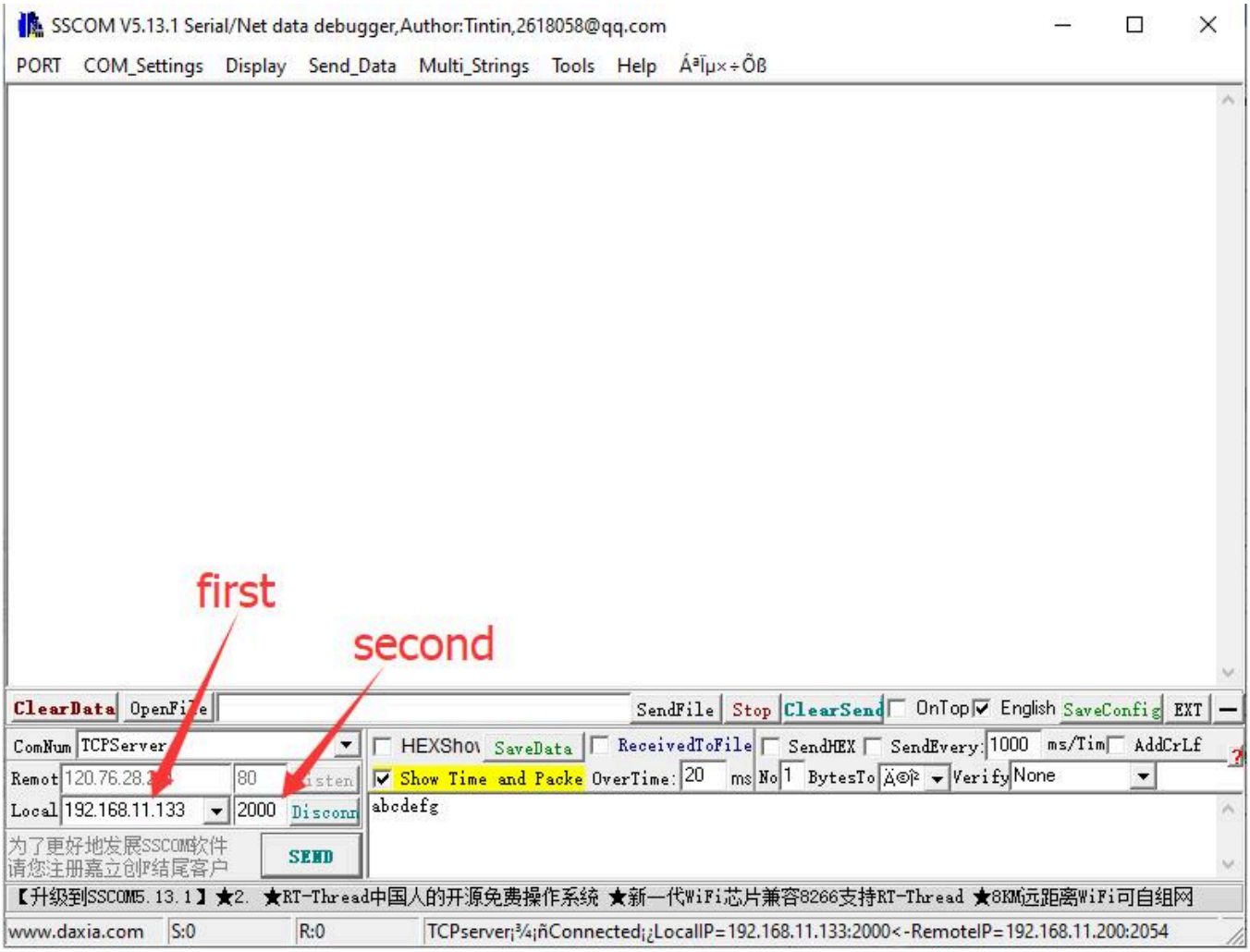
The default setting of the UART1 is as below:

- Baud rate: 9600
- Timeout: 0
- data bit: 8
- Stop bit: 1
- Parity: None

- Clearing buffer: Never

TCP CLIENT

1. Select SSCOM to select the protocol type as TCPServre, and then view the target IP and target port number through SSCOM (that is, 1 and 2 in the figure).



(/wiki/File:SSCOM-1.jpg)

2. Run NetModuleConfig.exe to set the module parameters, as follows:

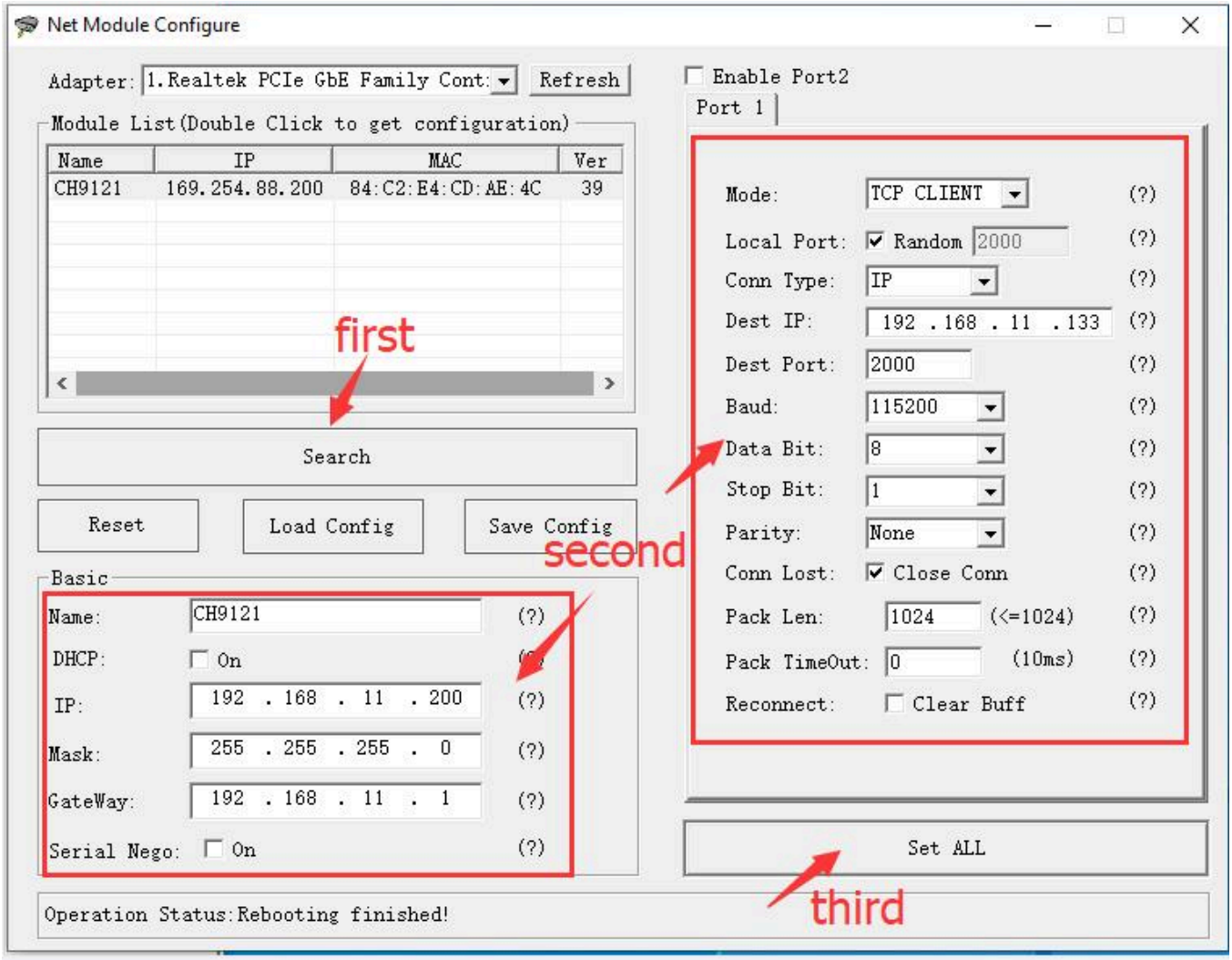
(1) Click "Search Device", and the device list will show the modules in the subnet;

(2) Double-click the module in the device list, modify the parameters on the left, and then click "Configure Device Parameters";

(3) After the configuration is completed, the module will restart, wait a while, click "Search Device", find the module, and view the configuration result.

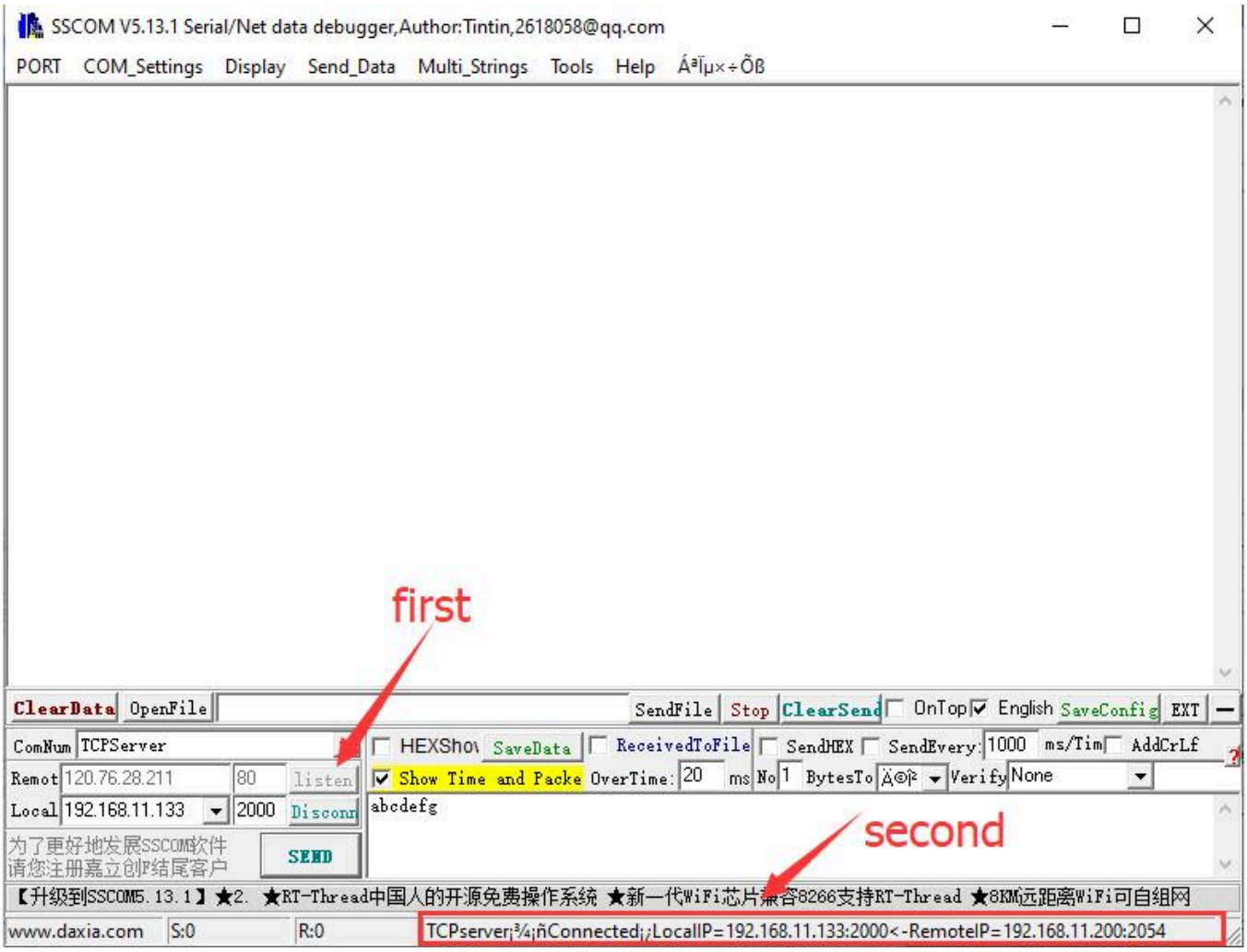
(4) Modify the module parameters according to actual needs, for example: set the network mode to TCP CLIENT, destination IP, destination port, and TCP SERVER.

The IP and port are the same, and then modify the baud rate and other parameters as required. Then click Configure Device Parameters.



(/wiki/File:NetModuleConfig-1.jpg)

3. After the configuration is complete, click on SSCOM to connect to it.



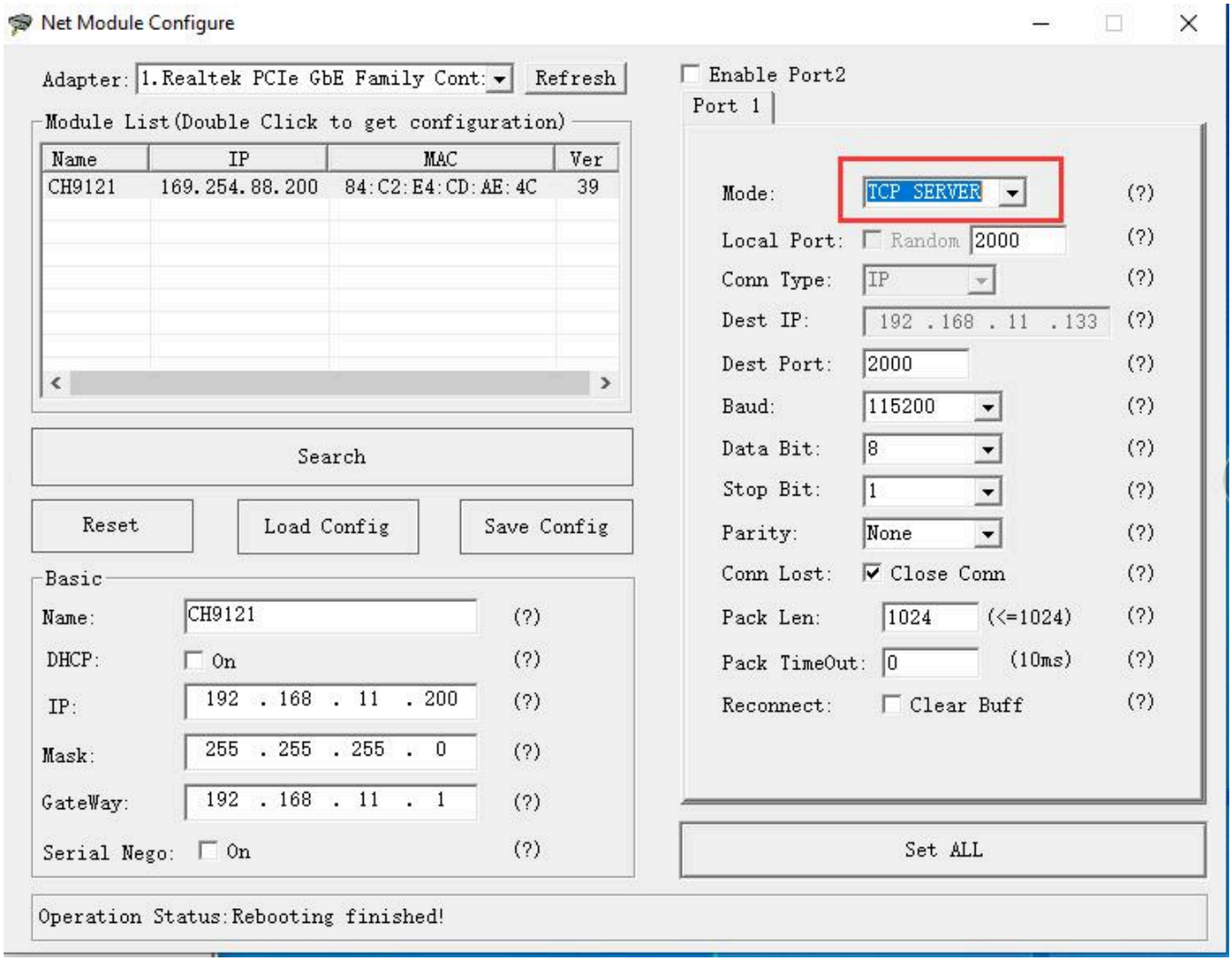
(/wiki/File:SSCOM-2.jpg)

TCP SERVER

1. Run NetModuleConfig.exe to set the module parameters, as follows:

(1) Click "Search Device", the device list will show the modules in the subnet;

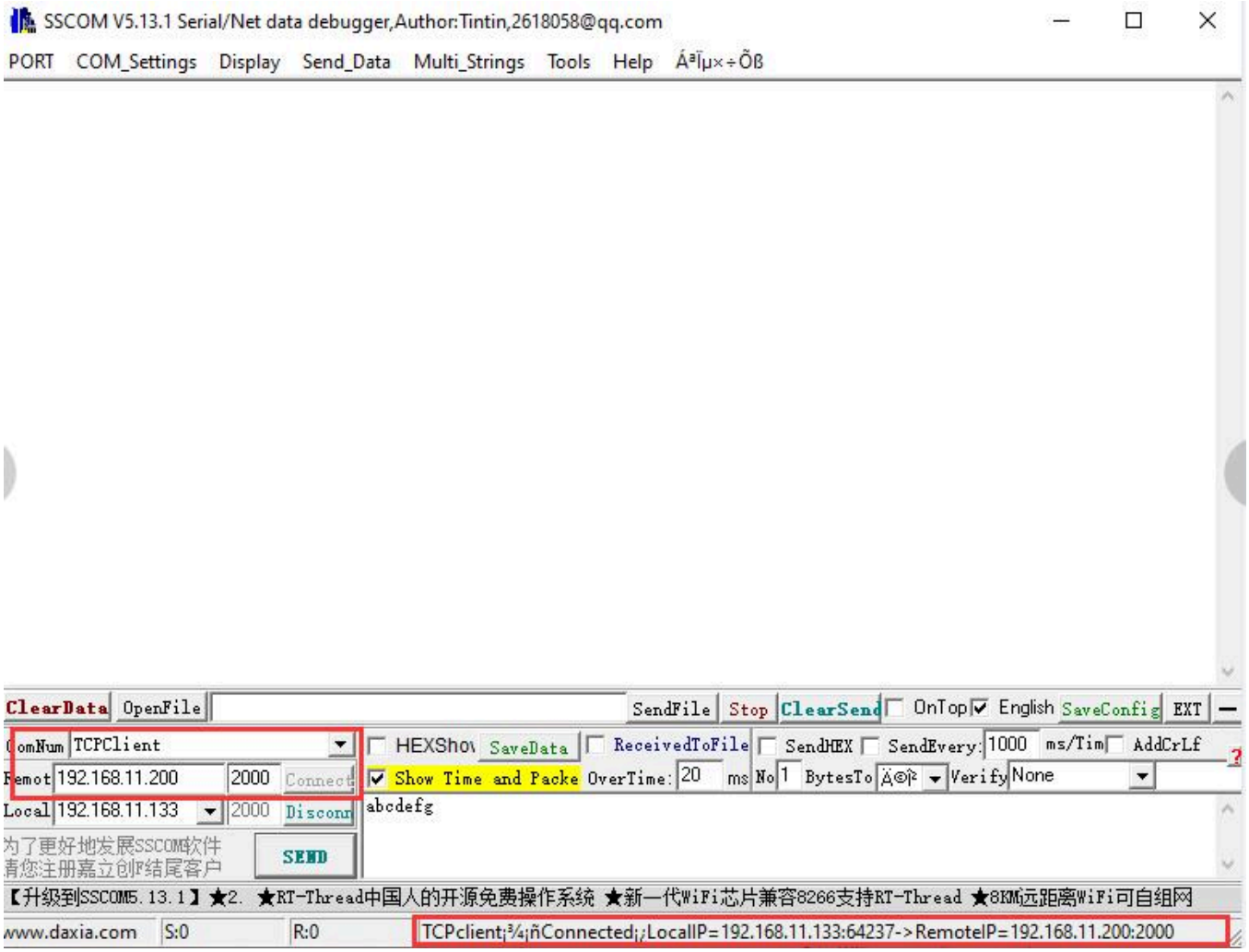
(2) Double-click the module in the device list, modify the network mode to TCPServre on the left, then modify the module parameters as required, and finally click "Configure Device Parameters".



(/wiki/File:NetModuleConfig-2.jpg)

2. Connect to the module configured as TCP SERVER through SSCOM, the connection will be shown as

follows:



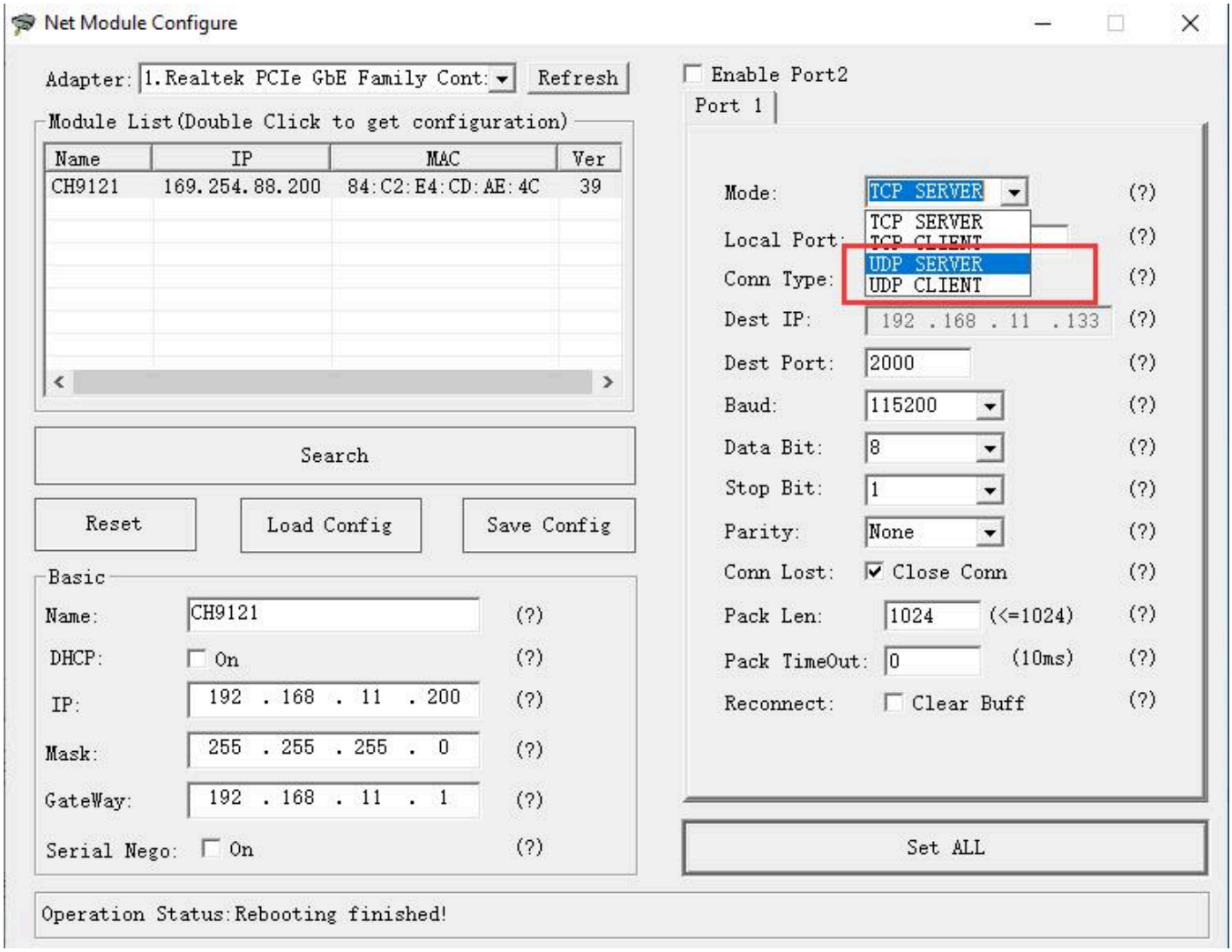
(/wiki/File:SSCOM-3.jpg)

UDP CLIENT/SERVER

1. Run NetModuleConfig.exe to set the module parameters, as follows:

(1) Click "Search Device", the device list will show the modules in the subnet;

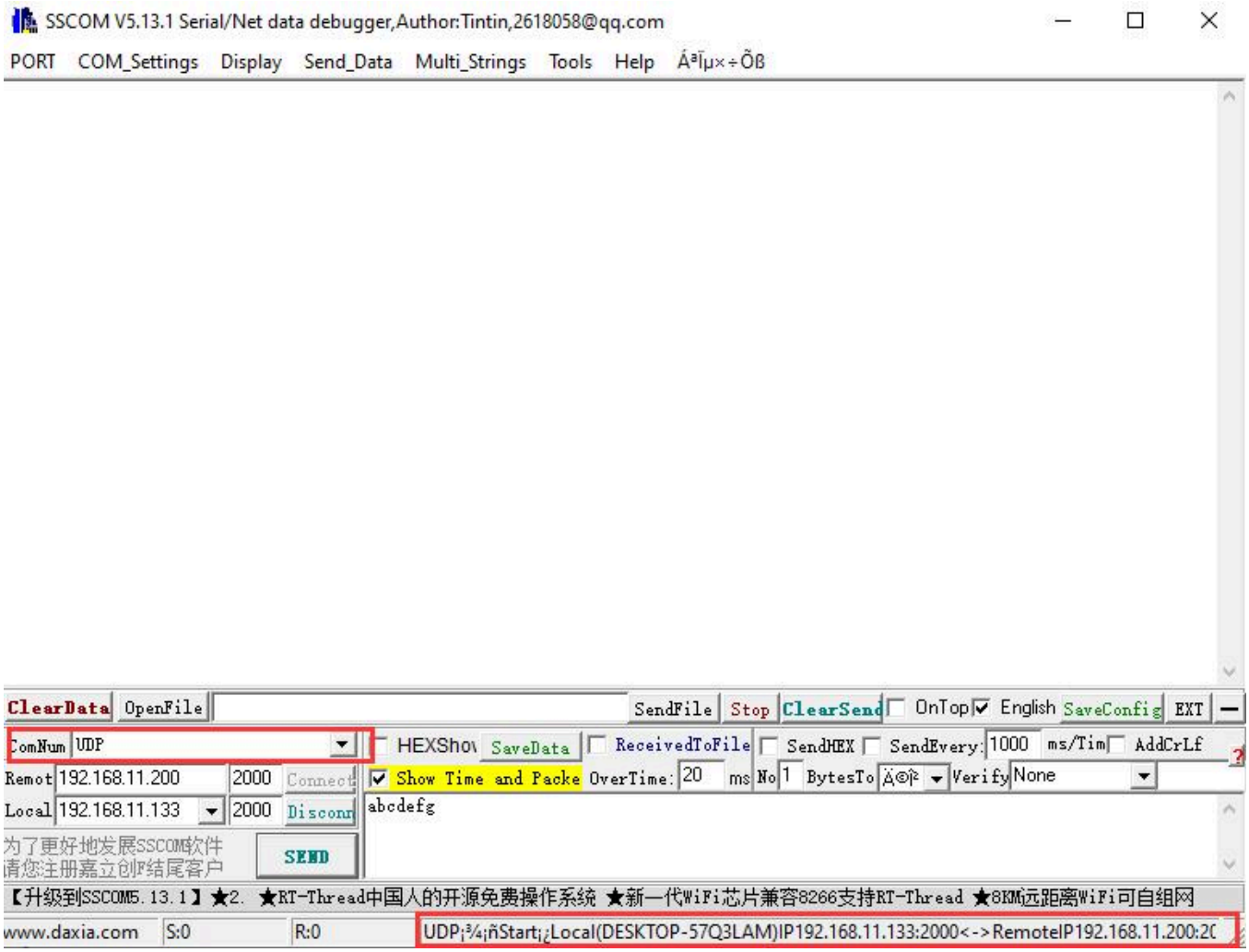
(2) Double-click the module in the device list, modify the network mode to UDP CLIENT/Servre on the left, then modify the module parameters as required, and finally click "Configure Device Parameters".



(/wiki/File:NetModuleConfig-3.jpg)

2. Connect the module configured as UDP CLIENT/Servre through SSCOM (the UDP mode of SSCOM will automatically connect, no matter if your mode is configured as CLIENT/Servre), the connection will be shown

as follows:



(/wiki/File:SSCOM-4.jpg)

Working with Pico

Hardware connection

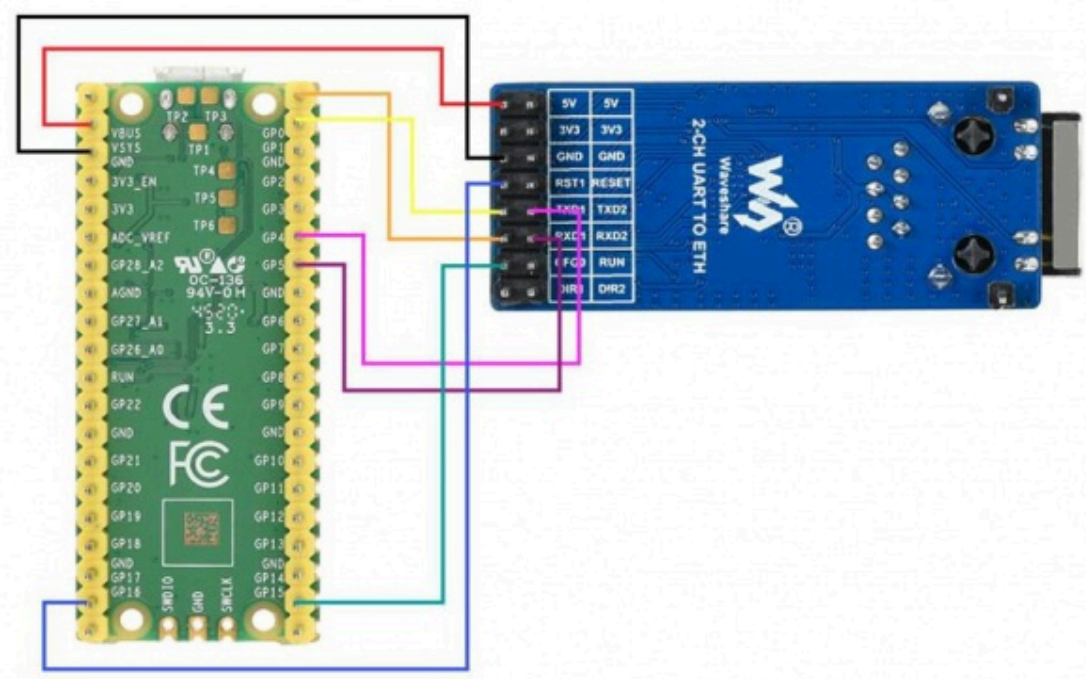
You can connect according to the following table.

Correspondence of Pico connection pins

ETH	Pico	Function
5V	VSYS	Power input
GND	GND	Power ground
RXD1	GP0	Serial data input
TXD1	GP1	Serial data output
RXD2	GP4	Serial data input
TXD2	GP5	Serial data output
CFG0	GP14	Network configuration enable pin

RST1	GP17	Reset

Direct connection



(/wiki/File:2-CH_UART_TO_ETH-Pico.jpg)

Program download

Open the Raspberry Pi terminal and execute:

```
sudo apt-get install p7zip-full
cd ~
sudo wget  https://files.waveshare.com/upload/3/37/2-CH_UART_TO_ETH_CODE.7z
7z x Pico_ETH_CH9121_CODE.7z -o./2-CH UART TO ETH_CODE
cd ~/2-CH UART TO ETH_CODE
cd Pico/c/build/
```

Demo

C

- The following tutorial is for the operation on the Raspberry Pi, but due to the multi-platform and portable characteristics of cmake, it can be successfully compiled on the PC, but the operation is slightly different and needs to be judged by the user.

To compile, make sure to be in the c directory:

```
cd ~/2-CH UART TO ETH_CODE/Pico/C/
```

2-CH UART TO ETH_CODE/Pico/C/Serial Port Parameter Configuration: Used to configure the mode through the serial port.

2-CH UART TO ETH_CODE/Pico/C/RX_TX: Used to send and receive information, and return what is received.

Enter one of the folders to create and enter the build directory, and add the SDK: Where ../../pico-sdk is the directory of your SDK. There is a build in our sample program, just enter it directly.

```
cd build
export PICO_SDK_PATH=../../pico-sdk
(Note: Be sure to write the path where your own SDK is located)
```

Execute cmake to automatically generate Makefile.

```
cmake ..
```

Executing make to generate executable files, the first compilation time is relatively long.

```
make -j9
```

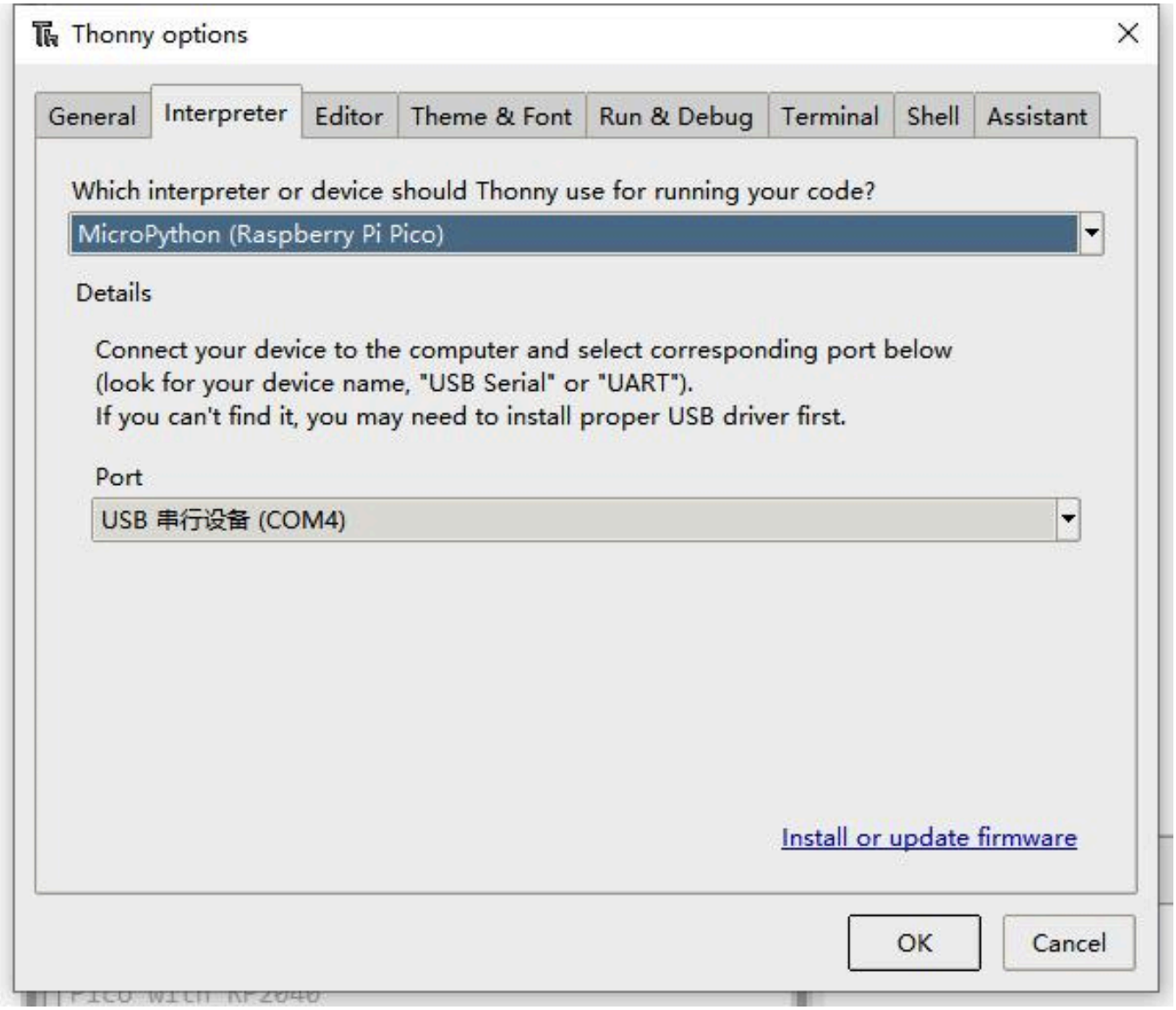
After the compilation is complete, the uf2 file will be generated. Press and hold the button on the Pico board, connect Pico to the USB port of the Raspberry Pi via the Micro USB cable, and then release the button. After connecting, the Raspberry Pi will automatically recognize a removable disk (RPI-RP2). Copy the main.uf2 file in the build folder to the recognized removable disk (RPI-RP2).

```
cp main.uf2 /media/pi/RPI-RP2/
```

Python

Use in PC Environment

- Press and hold the BOOTSET button on the Pico board, connect Pico to the USB port of the computer through the Micro USB cable, and release the button after the computer recognizes a removable hard disk (RPI-RP2).
- Copy the rp2-pico-20210418-v1.15.uf2 file in the python directory to the recognized removable disk (RPI-RP2).
- Open Thonny IDE (note: use the latest version of Thonny, otherwise there is no support package for Pico, the latest version under Windows is v3.3.3).
- Click Tools -> Settings -> Interpreter, and select Pico and the corresponding port as shown in the figure.



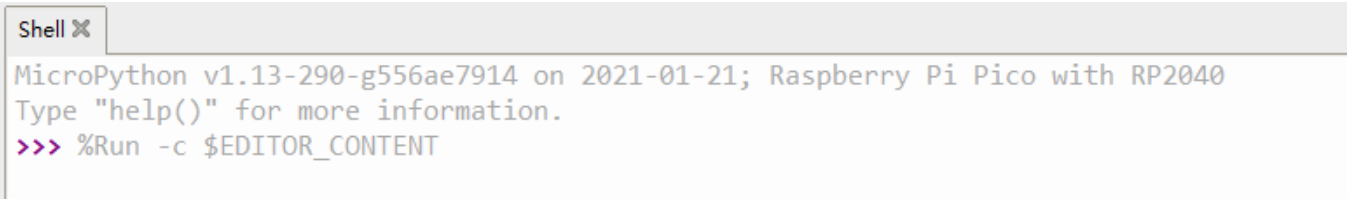
(/wiki/File:Pico-lcd-0.96-img-config.png)

This example provides two demos:

Serial Port Parameter Configuration.py: This demo is used to configure the mode through the serial port.

RX_TX.py: This is used to send and receive information, and return what is received.

- File -> Open -> RX_TX.py, click to run, as shown in the figure below:



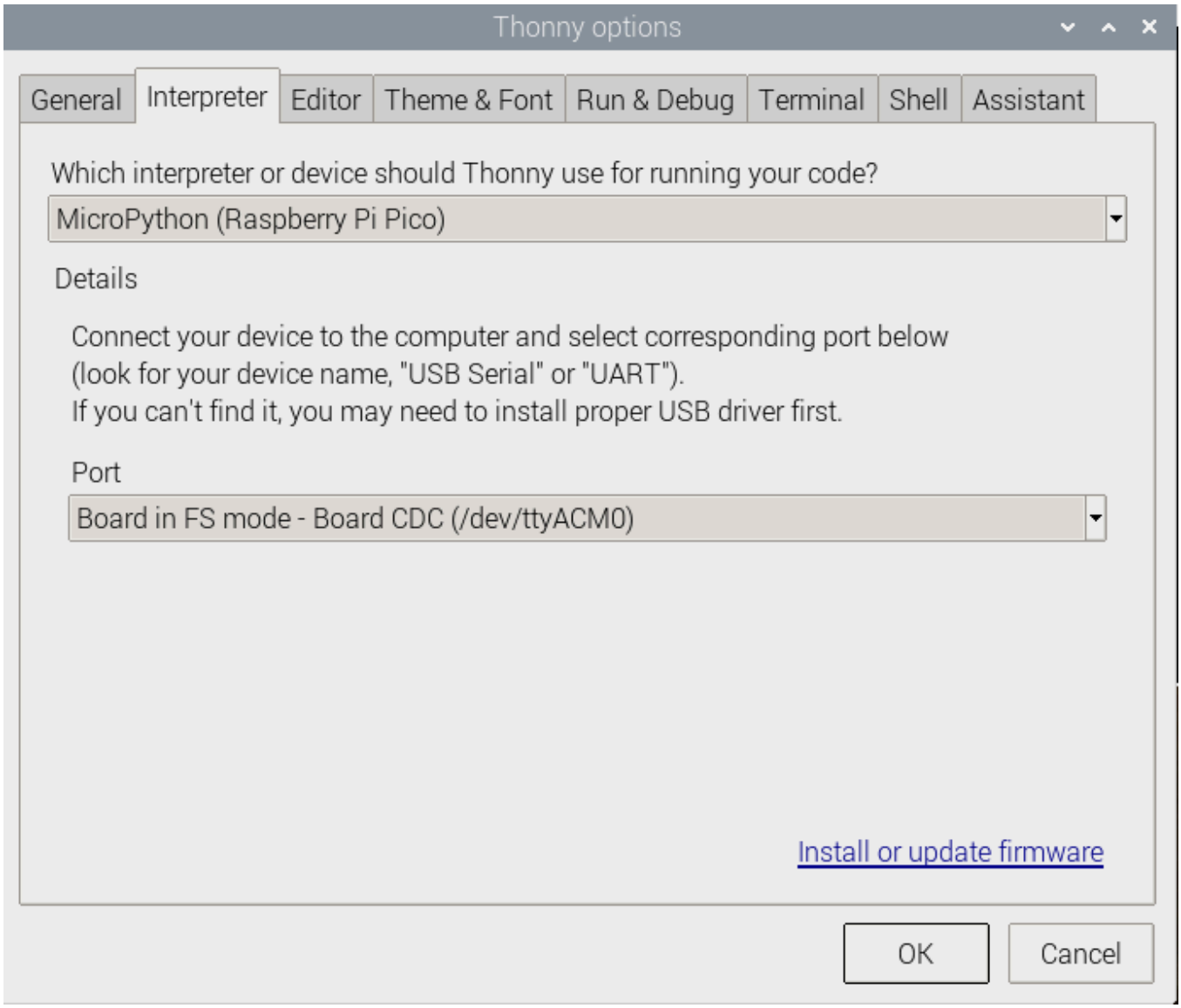
(/wiki/File:Pico-lcd-0.96-img-run.png)

Use in the Raspberry Pi environment

- The process of flashing the firmware is the same as on Windows. You can choose to copy the rp2-pico-20210418-v1.15.uf2 file into pico on your PC or Raspberry Pi.
- Open Thonny IDE on the Raspberry Pi Mountain (click the Raspberry logo -> Programming -> Thonny Python IDE), you can check the version information in Help->About Thonny.

To ensure that your version has the Pico support package, you can also click Tools -> Options... -> Interpreter to select MicroPython (Raspberry Pi Pico and ttyACM0 port.

As the picture shows:



(/wiki/File:Pico-lcd-0.96-img-config2.png)

If your current Thonny version does not have a pico support package, enter the following command to update Thonny IDE.

```
sudo apt upgrade thonny
```

Click File -> Open...-> python/RX_TX.py and run the script.

Code analysis

C

Configure the parameters through the serial port (modify according to your needs):

- Data type:

```
#define UCHAR unsigned char
#define UBYTE uint8_t
#define UWORD uint16_t
#define UDOUBLE uint32_t
```

- Module initialization:

```
void CH9121_init(void);
```

- This program only has a simple configuration. If you need to configure other functions, you can refer to the serial port control command and configure it yourself:

```
UCHAR CH9121_Mode            //Mode selection
UCHAR CH9121_LOCAL_IP[4]     //Local IP
UCHAR CH9121_GATEWAY[4]      //Gateway
UCHAR CH9121_SUBNET_MASK[4]  //Subnet mask
UCHAR CH9121_TARGET_IP[4]    //Target IP
UWORD CH9121_PORT1           //Local port
UWORD CH9121_TARGET_PORT     //Target port
UDOUBLE CH9121_BAUD_RATE     //Serial port baud rate
```

- According to the serial port control command, the following functions can be used to configure the parameters:

```
void CH9121_TX_4_bytes(UCHAR data, int command);  //Used for mode, whether the por
t is random, whether the port is disconnected from the network, whether to clear t
he serial port data, whether to open DHCP, //whether to open the serial port 2

void CH9121_TX_5_bytes(UWORD data, int command);  //Used to set the port number of
the serial port
void CH9121_TX_7_bytes(UCHAR data[], int command);//Used to set IP, subnet mask, g
ateway
void CH9121_TX_BAUD(UDOUBLE data, int command);   //Used to set the baud rate of t
he serial port
void CH9121_Eed();  //Update configuration parameters to EEPROM, execute configura
tion, reset 9121, leave configuration mode
```

Python

Users only need to modify the values shown below in Serial Port Parameter Configuration.py to configure the serial port parameters of the module:

```
MODE = 1  #0:TCP Server 1:TCP Client 2:UDP Server 3:UDP Client
GATEWAY = (169, 254, 88, 1)    # GATEWAY
TARGET_IP = (169, 254, 88, 17) # TARGET_IP
LOCAL_IP = (169,254,88,70)     # LOCAL_IP
SUBNET_MASK = (255,255,255,0)  # SUBNET_MASK
LOCAL_PORT1 = 5000             # LOCAL_PORT1
LOCAL_PORT2 = 4000             # LOCAL_PORT2
TARGET_PORT = 3000             # TARGET_PORT
BAUD_RATE = 115200             # BAUD_RATE
```

Provide WiringPi, RPI (Python) library demos.

Working with RPI

Hardware Connection

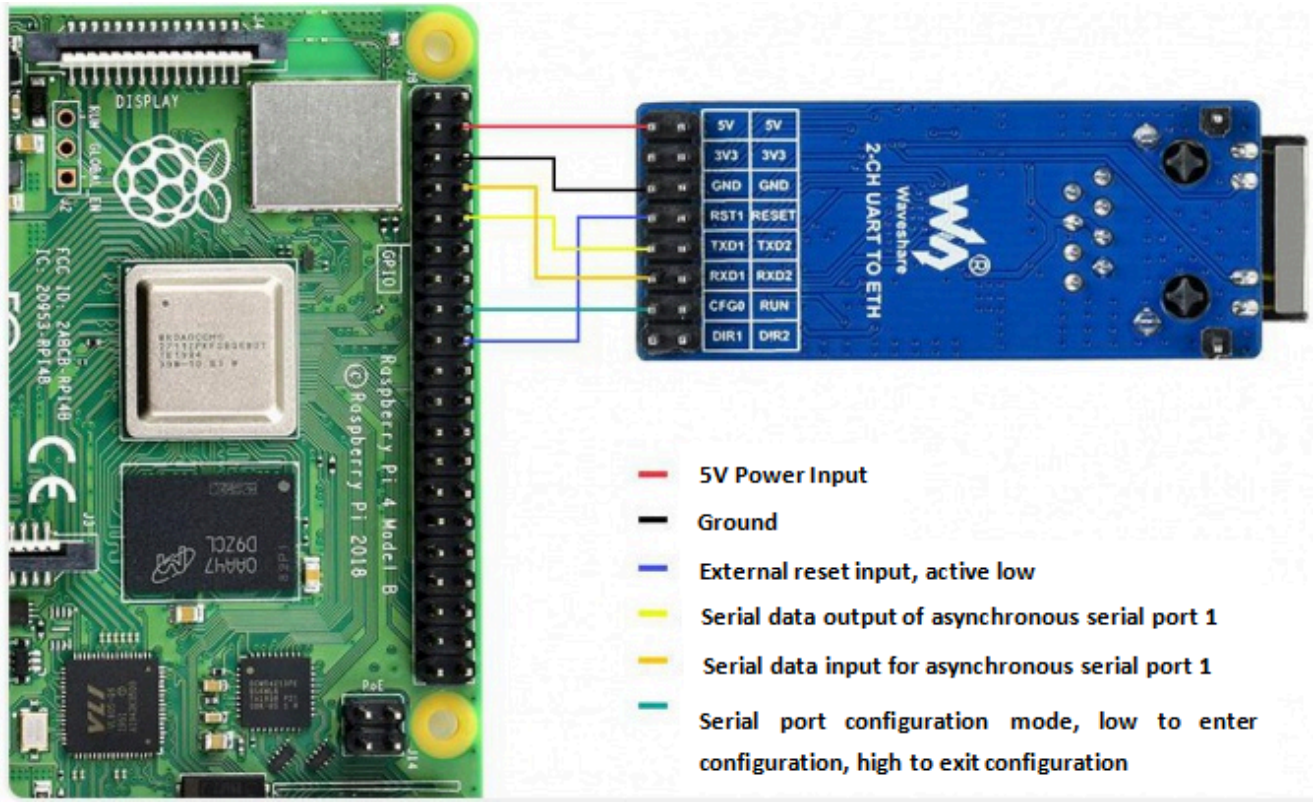
Please refer to the pin correspondence table below:

Raspberry Pi connection pin correspondence

ETH	Raspberry Pi	
	Wiring encoding	Board physical pin number
5V	5V	5V
GND	GND	GND
RXD1	15	8
TXD1	16	10
CFG0	4	16

RST1	5	18

Direct connection



(/wiki/File:2-CH-UART-TO-ETH-RPI001.png)

Enable UART

Open the Raspberry PI terminal, execute the following command to enter the Raspberry Pi configuration:

```
sudo raspi-config
```

You need to disable the login shell and enable the srial port hardware.

Choose Interfacing Options -> Serial -> No -> Yes:

L76X GPS Module rpi serial.png (/wiki/File:L76X_GPS_Module_rpi_serial.png)

Open the /boot/config.txt file and find the following configuration statement to enable the serial port, if not, add it at the end of the file:

```
enable_uart=1
```

Install Library

If you use the bookworm system, you can only use lgpio library, bcm2835 and wiringPi can't be installed and used.

BCM2835

```
#Open the Raspberry Pi terminal and run the following command
wget http://www.airspayce.com/mikem/bcm2835/bcm2835-1.71.tar.gz
tar zxvf bcm2835-1.71.tar.gz
cd bcm2835-1.71/
sudo ./configure && sudo make && sudo make check && sudo make install
# For more, you can refer to the official website at: http://www.airspayce.com/mik
em/bcm2835/
```

WiringPi

```
#Open the Raspberry Pi terminal and run the following command
cd
sudo apt-get install wiringpi
#For Raspberry Pi systems after May 2019 (earlier than that can be executed withou
t), an upgrade may be required:
wget https://project-downloads.drogon.net/wiringpi-latest.deb
sudo dpkg -i wiringpi-latest.deb
gpio -v
# Run gpio -v and version 2.52 will appear, if it doesn't it means there was an in
stallation error

# Bullseye branch system using the following command:
git clone https://github.com/WiringPi/WiringPi
cd WiringPi
. /build
gpio -v
# Run gpio -v and version 2.70 will appear, if it doesn't it means there was an in
stallation error
```

lgpio

```
#Open the Raspberry Pi terminal and run the following command
wget https://github.com/joan2937/lg/archive/master.zip
unzip master.zip
cd lg-master
sudo make install
#Raspberry PI 5
```

```
sudo apt install python3-rpi-lgpio
#For more details, you can refer to https://github.com/gpiozero/lg
```

- Install the Python function library:

```
#python2
sudo apt-get update
sudo apt-get install python-pip
sudo apt-get install python-pil
sudo apt-get install python-numpy
sudo pip install RPi.GPIO
sudo pip install spidev
#python3
sudo apt-get update
sudo apt-get install python3-pip
sudo apt-get install python3-pil
sudo apt-get install python3-numpy
sudo pip3 install RPi.GPIO
sudo pip3 install spidev
```

Download Test Demo

Open the Raspberry Pi terminal and execute:

```
sudo apt-get install p7zip-full
wget https://files.waveshare.com/upload/3/37/2-CH_UART_TO_ETH_CODE.7z
7z x 2-CH UART TO ETH_CODE. 7z -O./2-CH UART TO ETH_CODE
sudo chmod 777 -R 2-CH UART TO ETH_CODE
cd 2-CH UART TO ETH_CODE/RPi
```

Demo usage

C

```
cd WiringPi
```

- Serial Port Parameter Configuration: Used to configure the mode through the serial port.
- RX_TX: Used to send and receive information, and return what is received.

If you configure the parameters through the upper computer, you can directly run RX_TX to test whether there is a problem with the transmission and whether the packet is lost. Otherwise, you can use the Serial Port Parameter Configuration to configure the device parameters. (Functions will be introduced later).

```
#Configure device parameters
cd Serial Port Parameter Configuration
make clean
make
./CH9121_Config
#Run the send and receive program
cd ..
cd RX_TX
make clean
make
./CH9121_RX_TX
```

Python

```
cd Python
```

- Serial Port Parameter Configuration.py: Used to configure the mode through the serial port.
- RX_TX.py: Used to send and receive information, and return what is received.

```
python3 Serial Port Parameter Configuration.py #You can directly run the following statement through the configuration parameters of the host computer, otherwise mod ify the parameters in the Serial Port Parameter Configuration.py according to your needs, and then run it.
python3 RX_TX.py
```

Code Analysis

C

Configure the parameters through the serial port (modify according to your needs):

- This demo only has a simple configuration. If you need to configure other functions, you can refer to the serial port control command and configure it yourself:

```
uint8_t CH9121_Mode            //Mode selection
uint8_t CH9121_LOCAL_IP[4]     //Local IP
uint8_t CH9121_GATEWAY[4]      //Gateway
uint8_t CH9121_SUBNET_MASK[4]  //Subnet mask
uint8_t CH9121_TARGET_IP[4]    //Target IP
uint16_t CH9121_PORT1          //Local port
uint16_t CH9121_TARGET_PORT    //Target port
uint32_t CH9121_BAUD_RATE      //Serial port baud rate
```

- According to the serial port control command, the following functions can be used to configure the parameters:

```
void CH9121_TX_4_bytes(UCHAR data, int command);  //Used for mode, whether the por
t is random, whether the port is disconnected from the network, whether to clear t
he serial port data, whether to open DHCP, //whether to open the serial port 2

void CH9121_TX_5_bytes(UWORD data, int command);  //Used to set the port number of
the serial port
void CH9121_TX_7_bytes(UCHAR data[], int command);//Used to set IP, subnet mask, g
ateway
void CH9121_TX_BAUD(UDOUBLE data, int command);   //Used to set the baud rate of t
he serial port
void CH9121_Eed(); //Update configuration parameters to EEPROM, execute configurat
ion, reset 9121, leave configuration mode
```

Python

Users only need to modify the values shown below in Serial Port Parameter Configuration.py to configure the serial port parameters of the module:

```
MODE = 1 #0: TCP Server 1: TCP Client 2: UDP Server 3: UDP Client
GATEWAY = (169, 254, 88, 1)    # GATEWAY
TARGET_IP = (169, 254, 88, 17) # TARGET_IP
LOCAL_IP = (169,254,88,70)     # LOCAL_IP
SUBNET_MASK = (255,255,255,0)  # SUBNET_MASK
LOCAL_PORT1 = 5000             # LOCAL_PORT1
LOCAL_PORT2 = 4000             # LOCAL_PORT2
TARGET_PORT = 3000             # TARGET_PORT
BAUD_RATE = 115200             # BAUD_RATE
```

This example has been tested on Arduino Uno. Just connect to Arduino uno as shown in the table below.

Working with Arduino

Hardware Connection

You can connect according to the following table.

Arduino connection pin correspondence

ETH	Arduino	Function
5V	5V	Power input
GND	GND	Power ground
RXD1	TX	Serial data input
TXD1	RX	Serial data output
CFG0	D2	Network configuration enable pin

RST1	D4	Reset

Direct connection

2-CH UART TO ETH-Arduino.png (/wiki/File:2-CH_UART_TO_ETH-Arduino.png)

Install and compile the software (Windows tutorial)

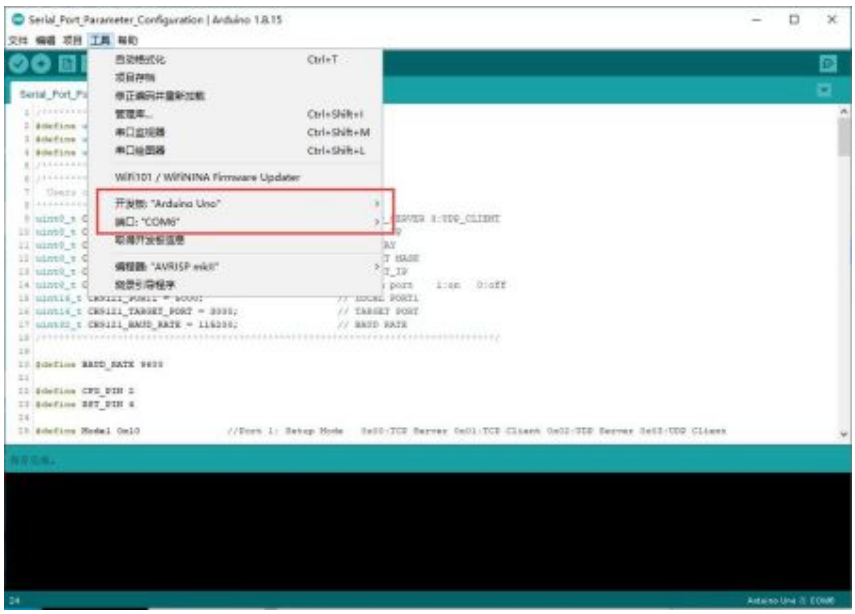
- Arduino IDE installation tutorial (https://www.waveshare.com/wiki/Arduino_ide_download)

Run the demo

Download the demo in the information we provide, unzip it, and then enter the 2-CH UART TO ETH_CODE/Arduino directory.

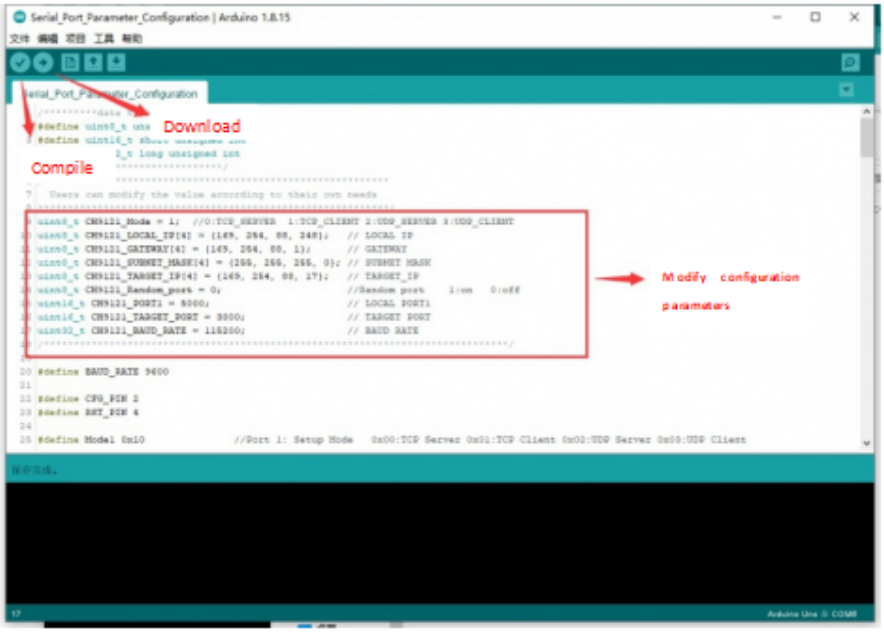
If you don't want to configure the parameters through the host computer, you can run the project in Serial_Port_Parameter_Configuration and double-click to open the .ino file.

Choose your development board and the corresponding port.



(/wiki/File:CH9121-Arduino-2.jpg)

Change the corresponding parameters according to your needs, and then compile and download (you don't need to connect the RXD and TXD of ETH to the Arduino during the download process, otherwise the download will fail), as shown below:



(/wiki/File:CH9121-Arduino-001.PNg)

After configuration, you can compile and download the project in RX_TX, and double-click to open the .ino file.

After the download is successful, run SSCOM to connect to the ETH module, and you can send data to ETH, and ETH will return everything you sent to it.

(Remember to connect RXD and TXD to Arduino.)

(/wiki/File:CH9121-Arduino-3.jpg)

Code Analysis

Configure the parameters through the serial port (modify according to your needs):

- Data type:

```
#define uint8_t unsigned char
#define uint16_t short unsigned int
#define uint32_t long unsigned int
```

- This demo only has a simple configuration. If you need to configure other functions, you can refer to the serial port control command and configure it yourself:

```
uint8_t CH9121_Mode            //Mode selection
uint8_t CH9121_LOCAL_IP[4]     //Local IP
uint8_t CH9121_GATEWAY[4]      //Gateway
uint8_t CH9121_SUBNET_MASK[4]  //Subnet mask
uint8_t CH9121_TARGET_IP[4]    //Target IP
uint16_t CH9121_PORT1          //Local port
uint16_t CH9121_TARGET_PORT    //Target port
uint32_t CH9121_BAUD_RATE      //Serial port baud rate
```

- According to the serial port control command, the following functions can be used to configure the parameters:

```
void CH9121_TX_4_bytes(UCHAR data, int command);  //Used for mode, whether the por
t is random, whether the port is disconnected from the network, whether to clear t
he serial port data, whether to open //DHCP, whether to open the serial port 2

void CH9121_TX_5_bytes(UWORD data, int command);  //Used to set the port number of
the serial port
void CH9121_TX_7_bytes(UCHAR data[], int command);//Used to set IP, subnet mask, g
ateway
void CH9121_TX_BAUD(UDOUBLE data, int command);   //Used to set the baud rate of t
he serial port
void CH9121_Eed(); //Update configuration parameters to EEPROM, execute configurat
ion, reset 9121, leave configuration mode
```

STM32

The sample demo we provide is based on STM32F103RBT6, and the connection method provided is also the pin of the corresponding STM32F103RBT6. If there is a need to transplant the demo, please connect according to the actual pin.

Hardware Connection

STM32F103ZET connection pin correspondence

ETH	STM32
5V	5V
GND	GND
RXD1	PC10
TXD1	PC11

CFG0	PC12
RST1	PD2

Take our STM32F103RBT6 as an example, the connection is as follows:

2-CH UART TO ETH-STM32.png (/wiki/File:2-CH_UART_TO_ETH-STM32.png)

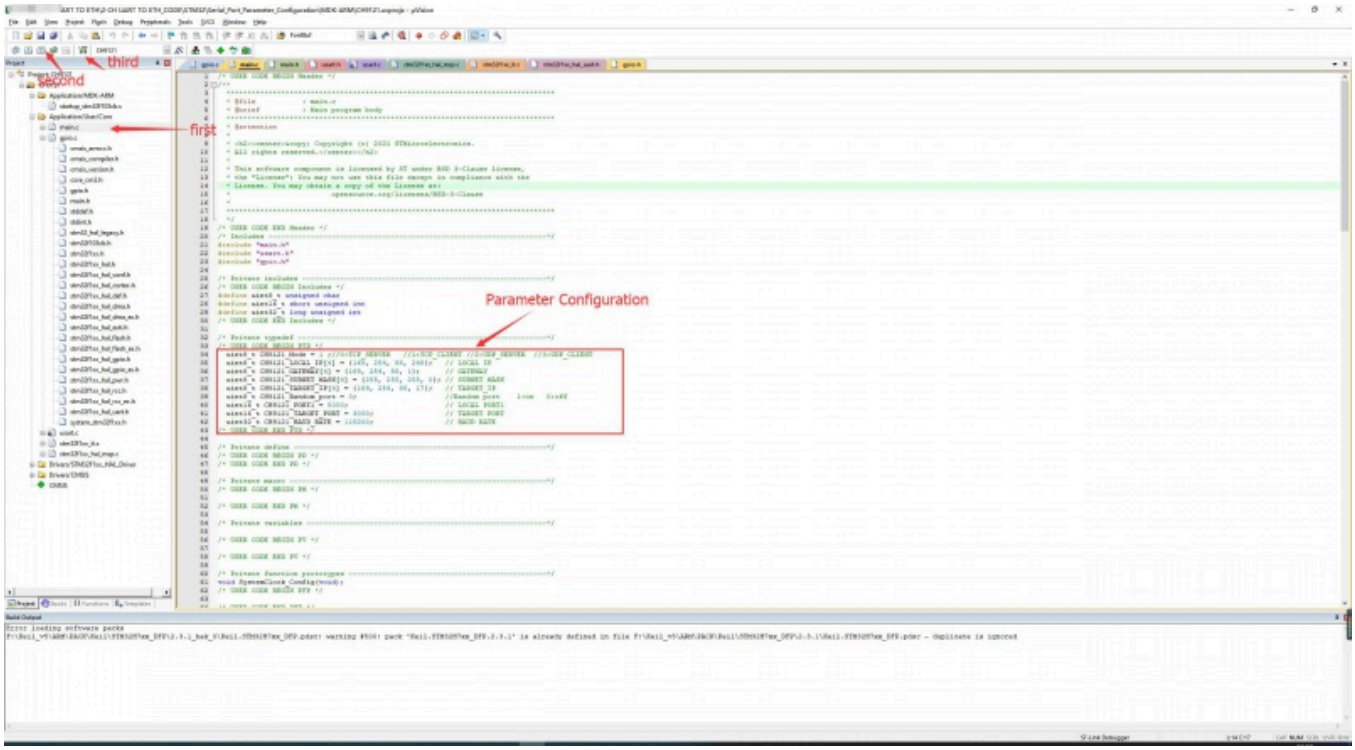
Software Description

The demo is developed based on the HAL library. Please download the demo in the data, find the STM32 program file directory, and open STM32\Serial_Port_Parameter_Configuration\MDK-ARM. CH9121.uvprojx under the directory, you can see the demo.

CH9121 STM32 1.jpg (/wiki/File:CH9121_STM32_1.jpg)

Open main.c, you can see the demo that can be modified by the user, modify the corresponding parameters according to your own needs, and then recompile and download.

(This function is mainly used to configure the serial port parameters. If you use the upper computer configuration, you can directly run the demo in the RX_TX file to test the data transmission and reception, whether it will lose packets, etc.)

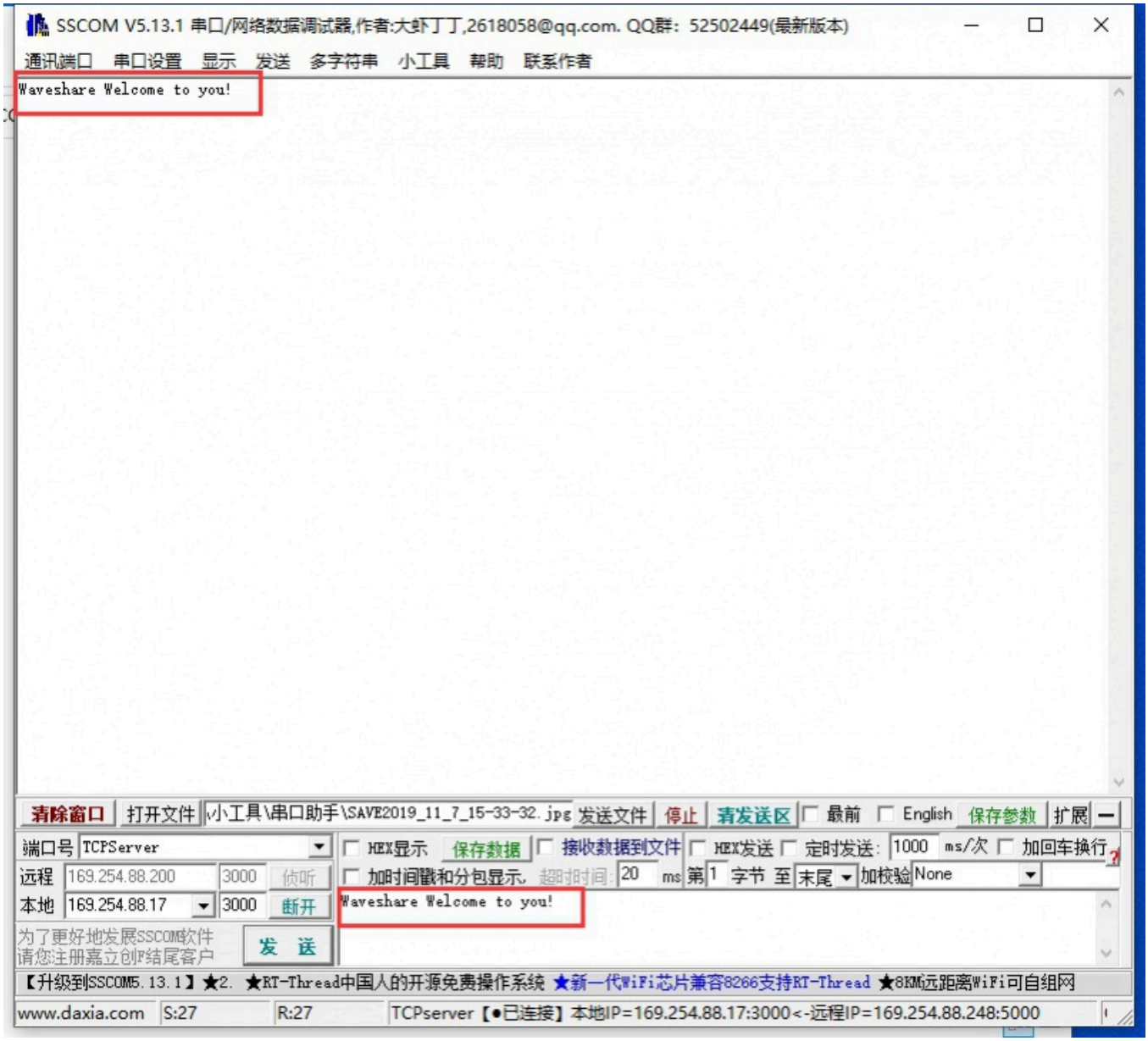


(/wiki/File:CH9121_STM32_2.jpg)

After the parameters are configured, download the demo in the RX_TX folder to STM32.

After the download is successful, run SSCOM to connect to the ETH module, and you can send data to ETH,

and ETH will return everything you sent to it.



(/wiki/File:CH9121-Arduino-3.jpg)

Code Analysis

- This demo only has a simple configuration. If you need to configure other functions, you can refer to the serial port control command and configure it yourself:

```
uint8_t CH9121_Mode            //Mode selection
uint8_t CH9121_LOCAL_IP[4]     //Local IP
uint8_t CH9121_GATEWAY[4]      //Gateway
uint8_t CH9121_SUBNET_MASK[4]  //Subnet mask
uint8_t CH9121_TARGET_IP[4]    //Target IP
uint16_t CH9121_PORT1          //Local port
uint16_t CH9121_TARGET_PORT    //Target port
uint32_t CH9121_BAUD_RATE      //Serial port baud rate
```

- According to the serial port control command, the following functions can be used to configure the parameters:

```
void CH9121_TX_4_bytes(UCHAR data, int command);  //Used for mode, whether the por
t is random, whether the port is disconnected from the network, whether to clear t
he serial port data, whether to open DHCP, //whether to open the serial port 2
void CH9121_TX_5_bytes(UWORD data, int command);  //Used to set the port number of
the serial port
void CH9121_TX_7_bytes(UCHAR data[], int command);//Used to set IP, subnet mask, g
ateway
void CH9121_TX_BAUD(UDOUBLE data, int command);   //Used to set the baud rate of t
he serial port
void CH9121_Eed(); //Update configuration parameters to EEPROM, execute configurat
ion, reset 9121, leave configuration mode
```

Resources

- Schematic (https://files.waveshare.com/upload/5/5a/2-CH_UART_TO_ETH_SCH.pdf)
- Demo (https://files.waveshare.com/upload/3/37/2-CH_UART_TO_ETH_CODE.7z)
- CH9121 Datasheets (https://files.waveshare.com/upload/a/a9/CH9121DS1.PDF)
- CH9121 AT Commands (https://files.waveshare.com/upload/e/ef/CH9121_SPCC.pdf)
- Software (https://files.waveshare.com/upload/b/bc/Configuration_software.7z)

Support

Technical Support

If you need technical support or have any feedback/review, please click the **Submit Now** button to submit a ticket, Our support team will check and reply to you within 1 to 2 working days. Please be patient as we make every effort to help you to resolve the issue.
Working Time: 9 AM - 6 PM GMT+8 (Monday to Friday)

Submit Now (https://service.waveshare.com/)