User Manual and Test Guidde

Modbus Master Function

1. FUNCTIONS

The Modbus Master function is used for the product to initiate Modbus polling. It queries the Modbus Slave device status and sends it to the server.

There is simplified and complete version of modbus master function, the difference is as following.

- Simplified version supports max 10 query command, complete version does not have any limit
- Simplified version supports JSON or HEX format upload, complete version support JSON
- Simplified version supports any query protocol (not only modbus, can be set any command), complete version support modbus only
- All HF IOT Device support the simplified version, but only part IOT Device support complete version.

The following products support Modbus Master complete version.

Ethernet + Wi-Fi + serial port: HF2211, HF2221, HF9610

Ethernet + Wi-Fi + IO Control: HF6208

Ethernet + Wi-Fi + 4G + serial port: HF2421, HF2421G, HF9624

Ethernet +4G+ serial port: EG46

For basic products usage and config procedure, see operation guide for detail, this doc won't guide the basic usage.

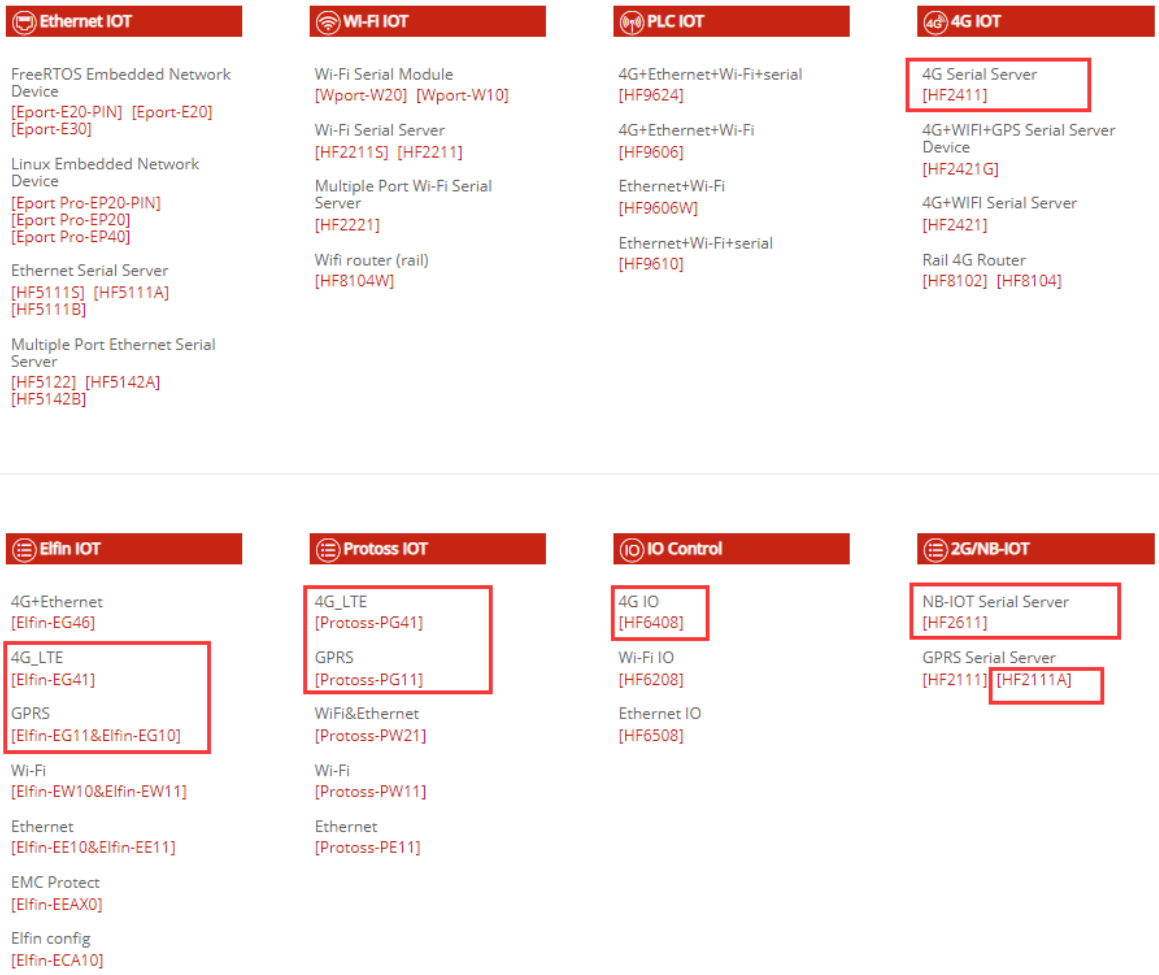
Related tools can be downloaded from the following address:

http://www.hi-flying.com/index.php?route=download/category&path=1_4

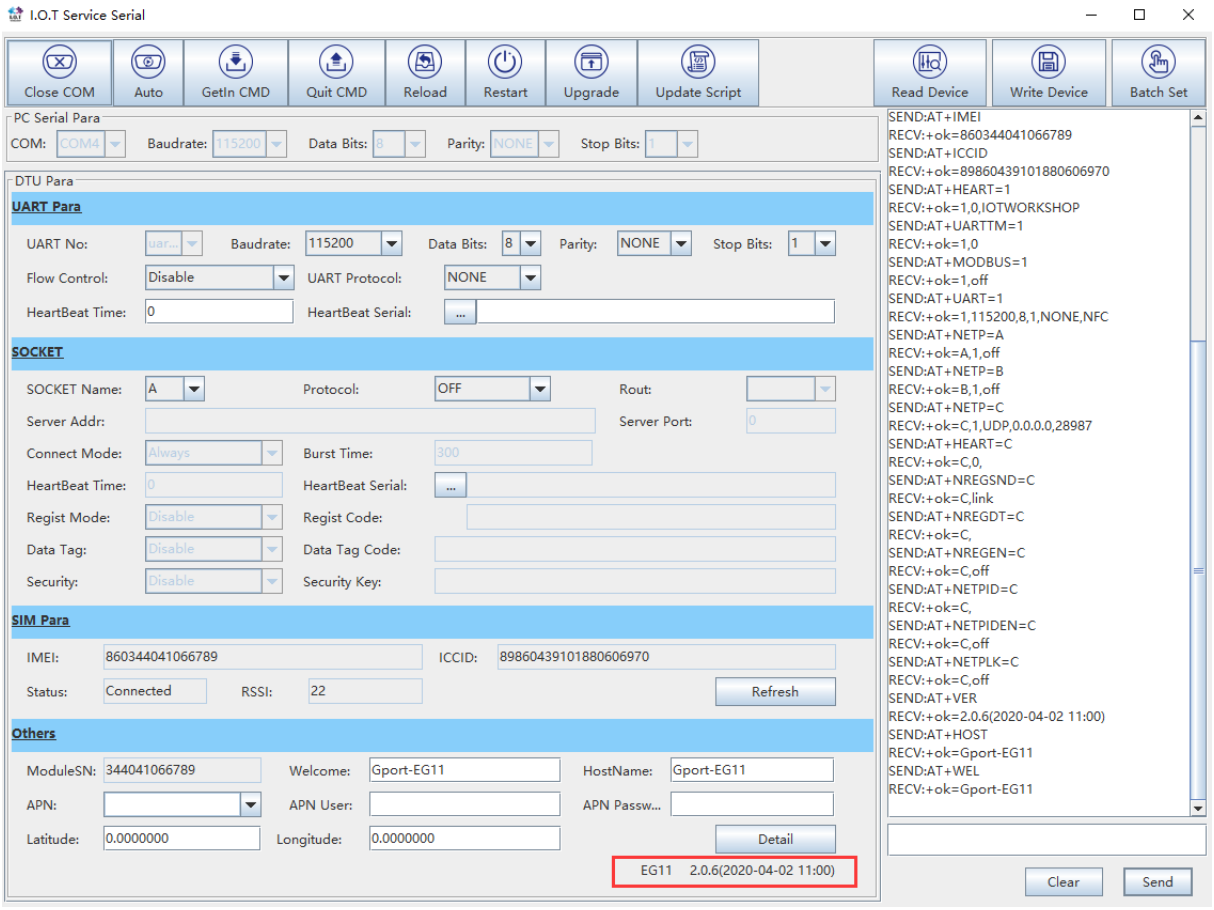
2. SIMPLIFIED MODBUS MASTER TEST

2.1. Cellular Network Products Test

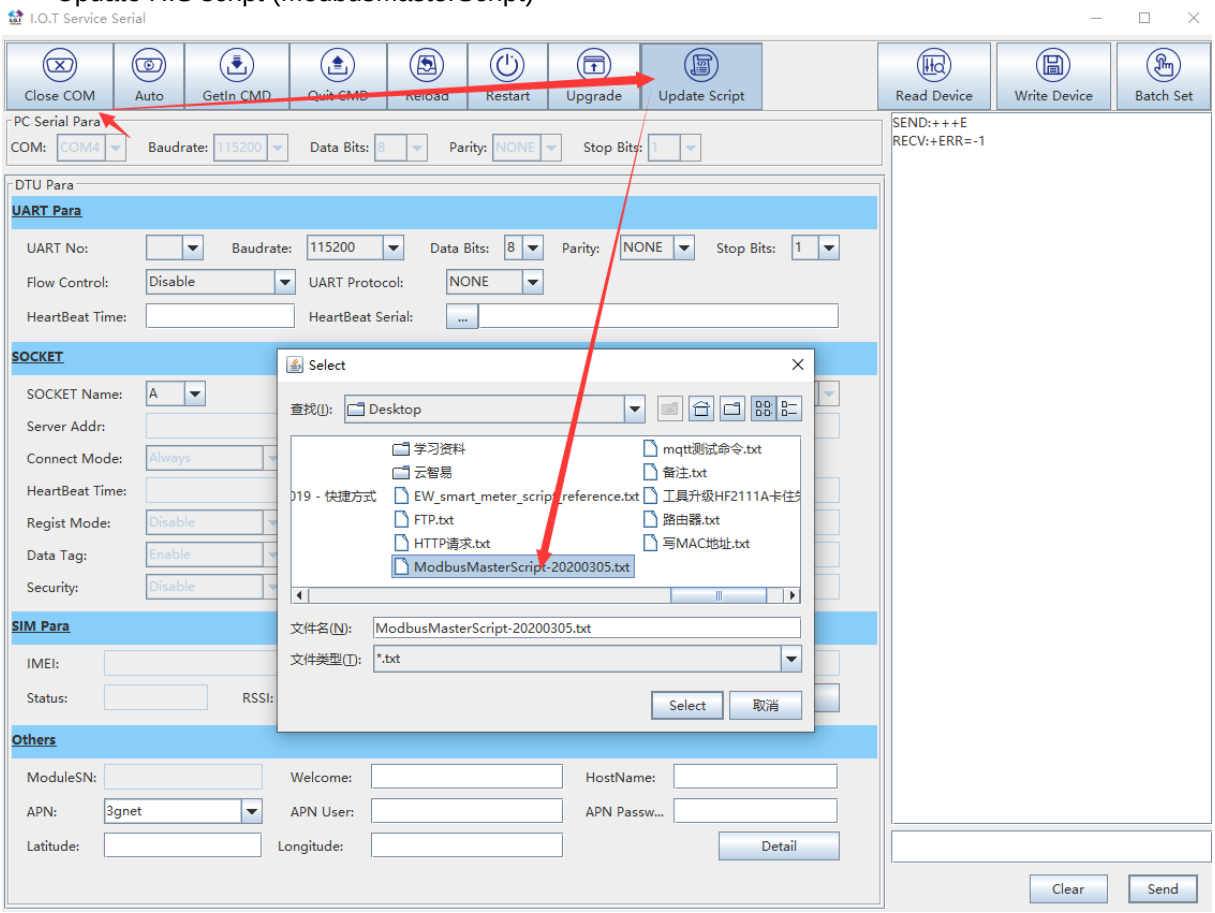
Cellular network products use AT command to config, the detailed product type is as following, this case take EG11 for example.



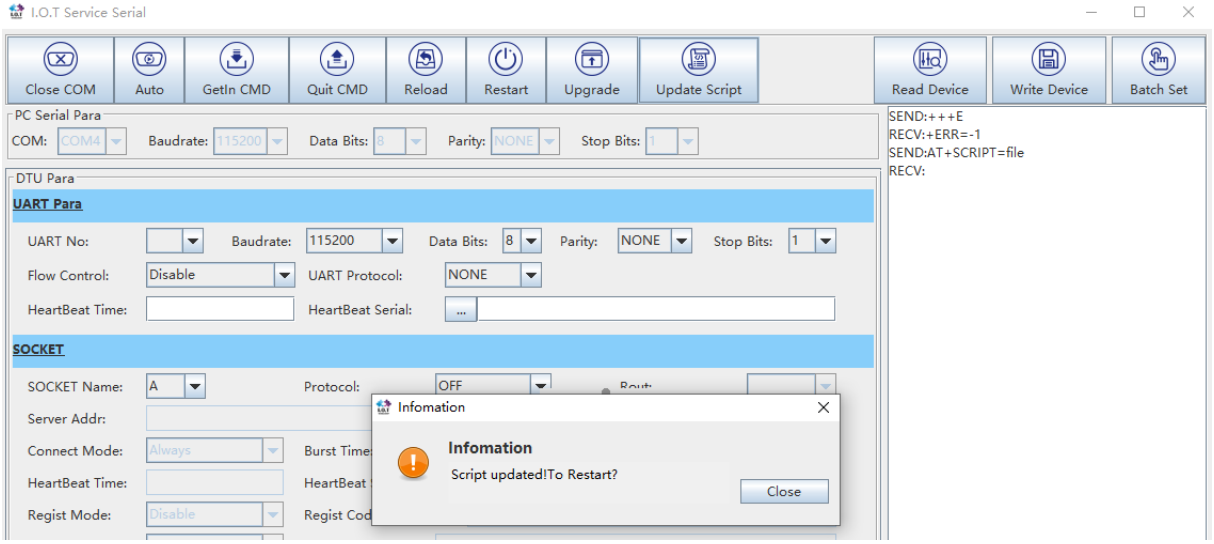
Upgrade firmare to 2.0.6 version.



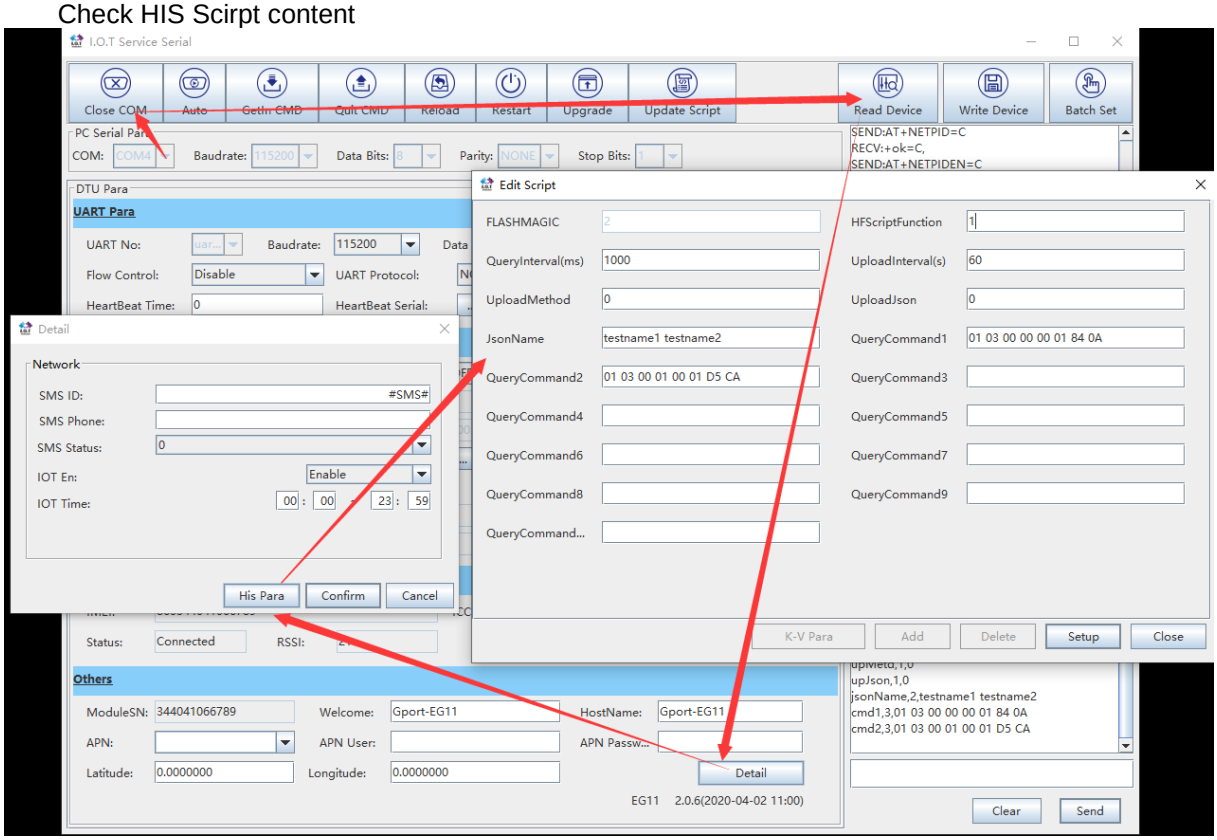
Update HIS script (ModbusMasterScript)



Restart products if update success.



Check HIS Scirpt content



FLASHMAGIC: HIS Para initial flag, if the flag is not the same as previous, then all the HIS para will be initialized to the scrpt file content. For example, when first update this script, all the HIS para will be initialized, when second time update script, the HIS para won't be initialized, keep the previous.

HFScription: Query function enable or disable, 1: Enable Query. 0: Disable Query.

QueryInterval(ms): The interval time of each query command, range from 200ms to 60000ms

UploadInterval(s): The interval time of several query commands, range from 10sec to 86400 sec.For example,

Query1->QueryInterval->Query2…->Query10->UploadInterval->Query1->QueryInterval->….

UploadMethod: 0 or 1，0: Upload immediately when receive response of each Query Command packet, 1: Upload after receive all the Query Command response packet.

UploadJson: 0 or 1, 0: Upload the response packet. 1: transform the response packet to JSONand upload.

Upload Json: Only valid when Upload Method is 1, Name number should be same as query command number. maximum 10characters for each command.

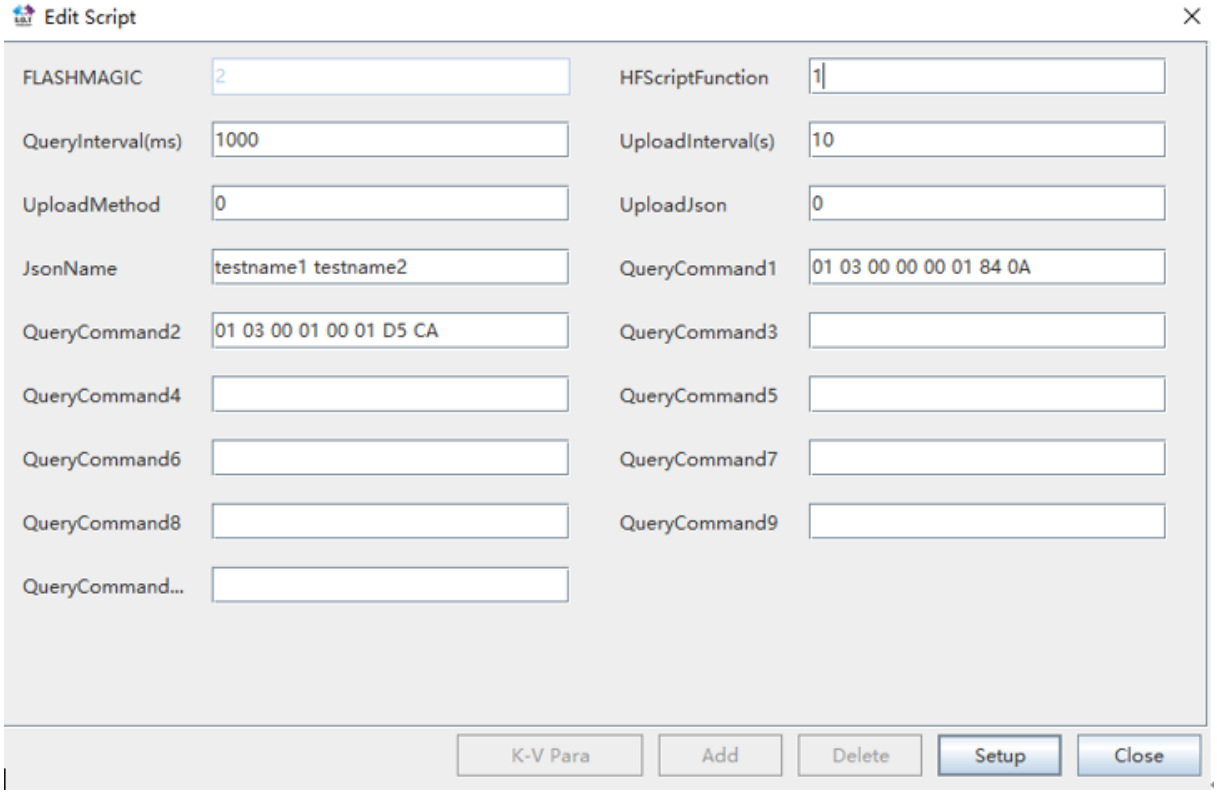
- UploadMethod set to 0, Upload Method set to 1 example.

```
Upload 1：  //value is the response packet in ASCII format
{
        "key": "Name1",
        "value": "xxxxxxxx"
}
Upload 2：
{
        "key": "Name2",
        "value": "xxxxxxxx"
}
```

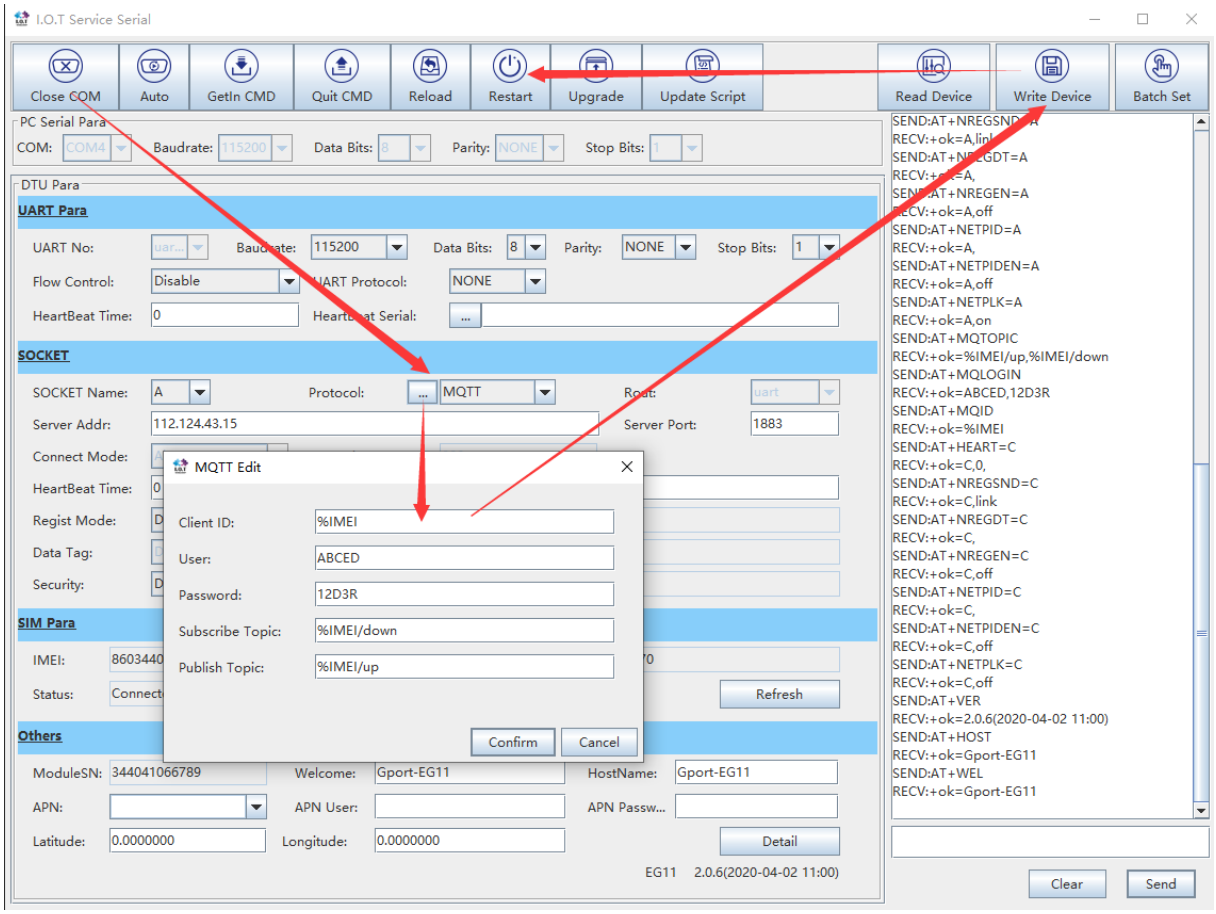
- UploadMethod set to 1, Upload Method set to 1 example.

```
Upload
{
        "Packets": [{
                "key": "Name1",
                "value": "xxxxxxxx"
        }, {
                "key": "Name2",
                "value": "xxxxxxxx"
        }, {
                "key": "Name3",
                "value": "xxxxxxxx"
        }]
}
```

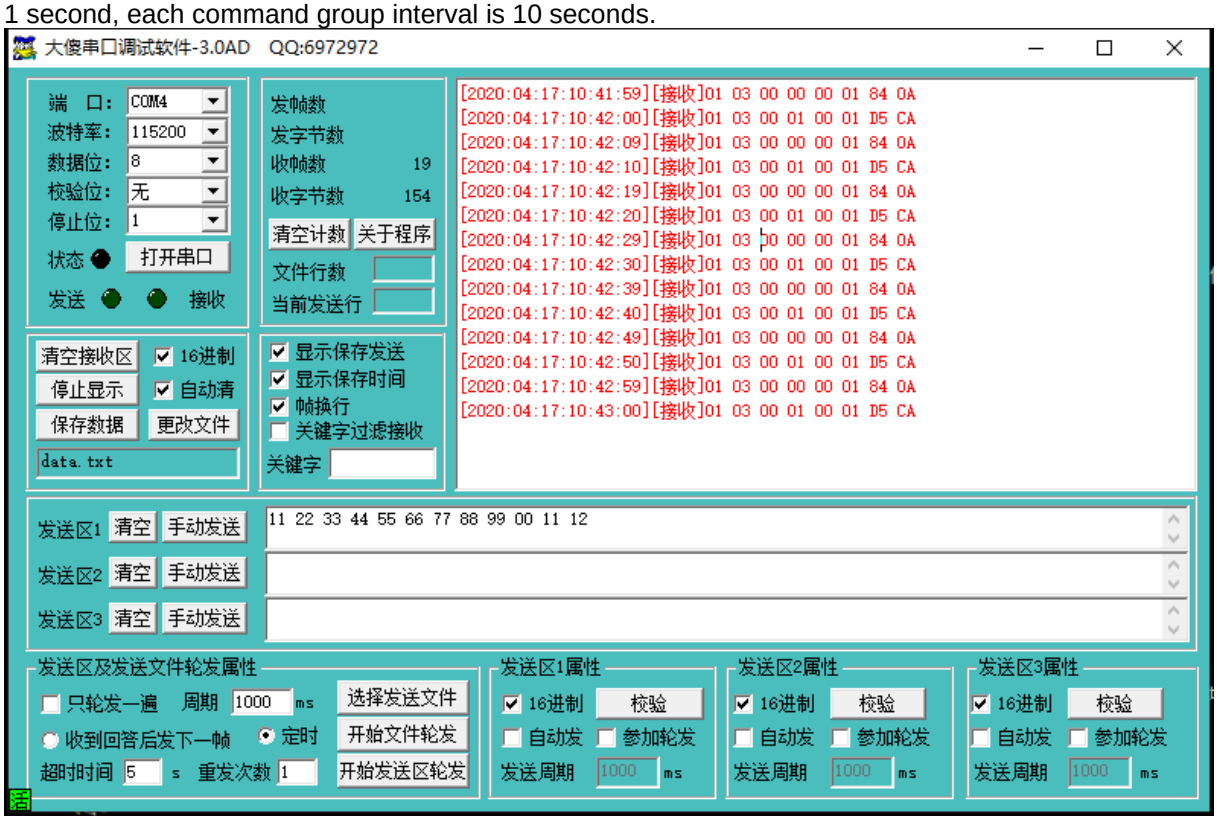
QuerryCommand: Hex format, max 10 commands, 80 bytes max for each command.



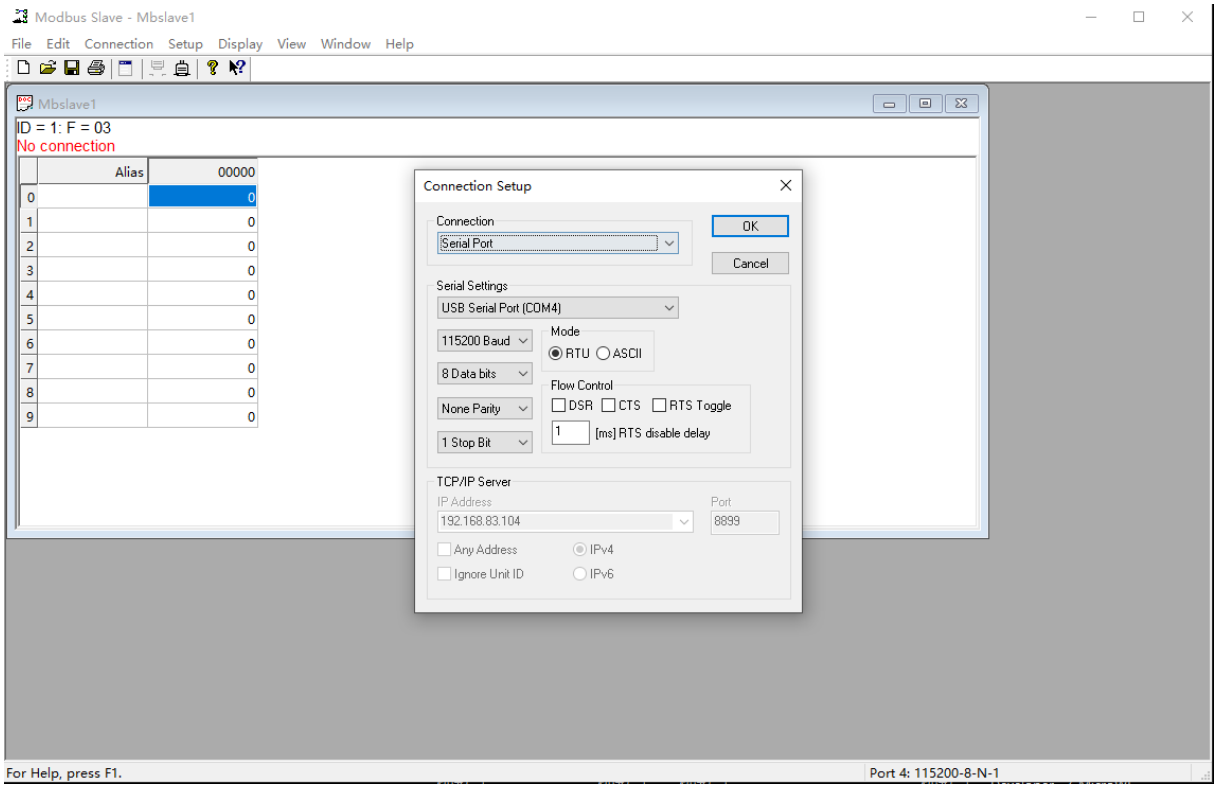
Set server address and reboot(Our MQTT test server: 112.124.43.15, port 1883), it also support HTTP/WebSocket communication.



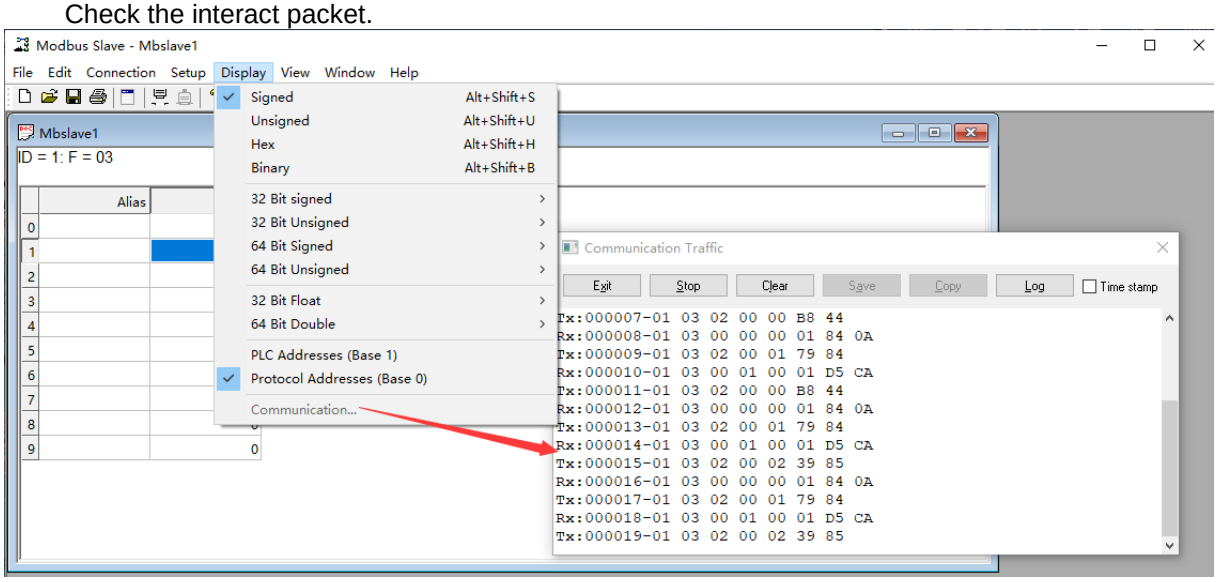
Open UART tools, the device will send out query command as setting, each command interval is 1 second, each command group interval is 10 seconds.



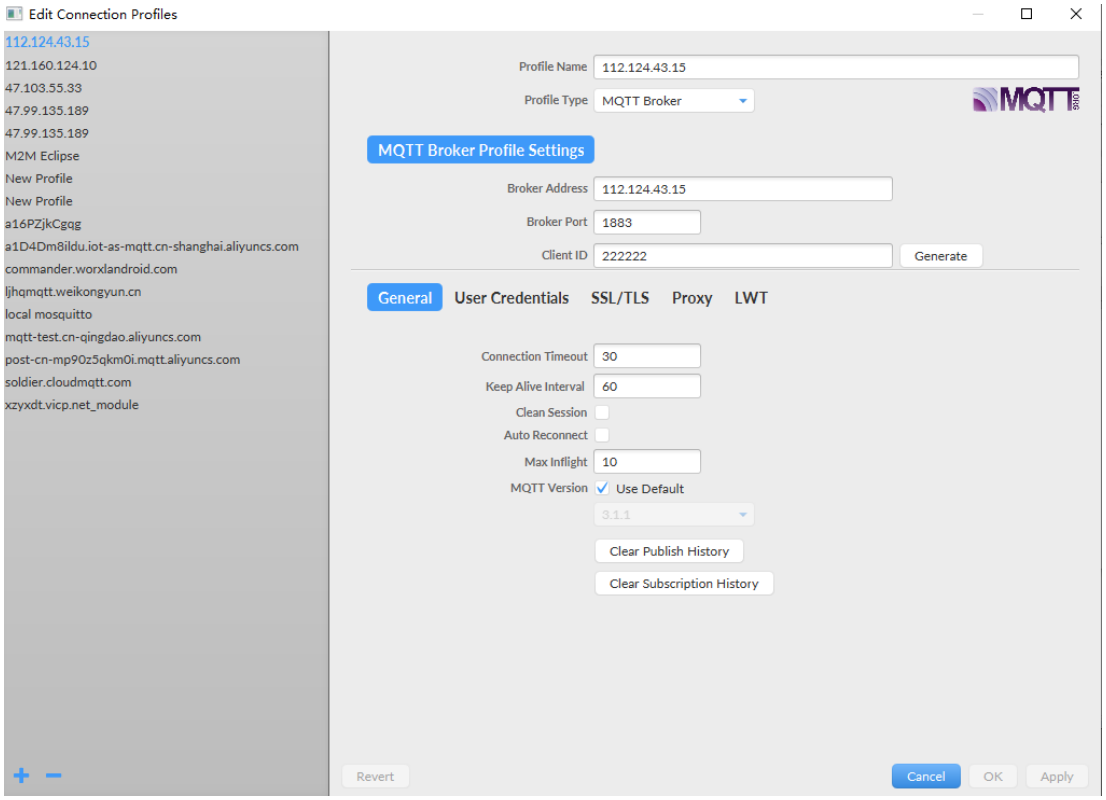
Open Modbus Slave to immulate modbus slave device.



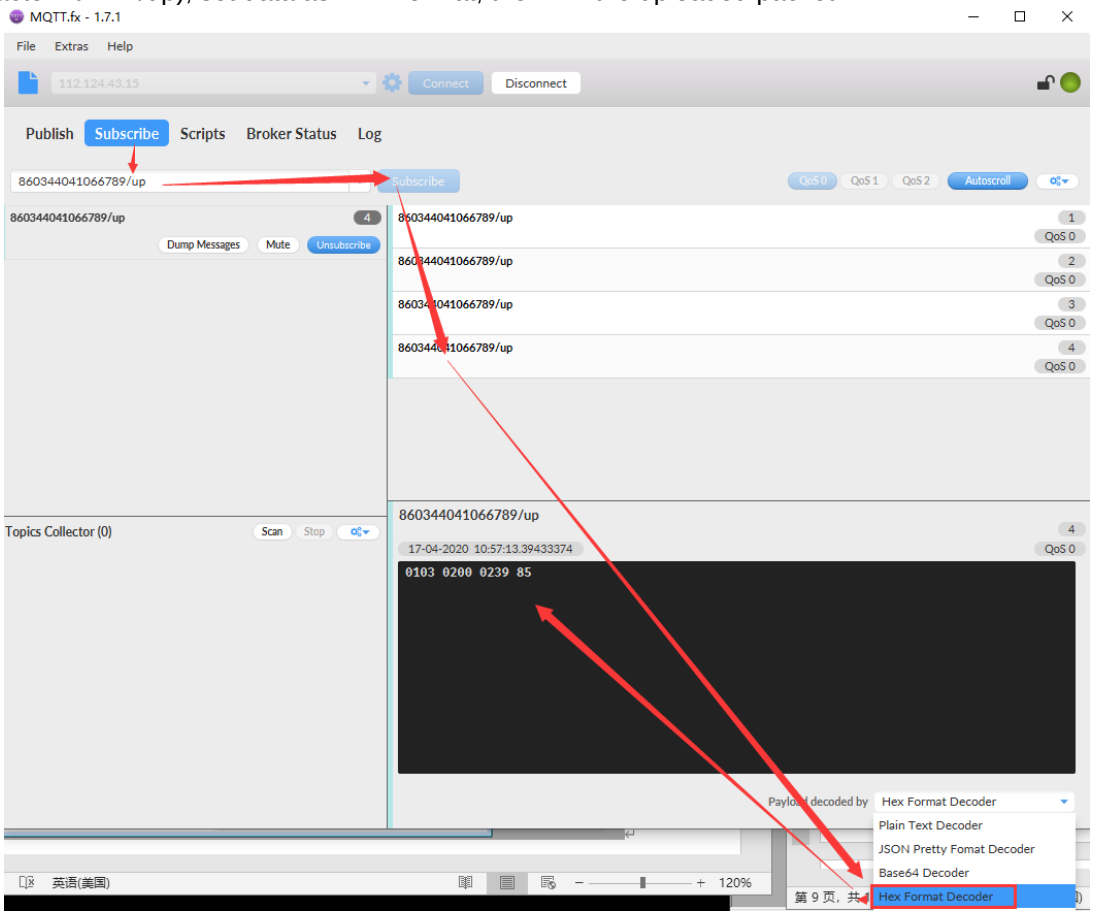
Check the interact packet.



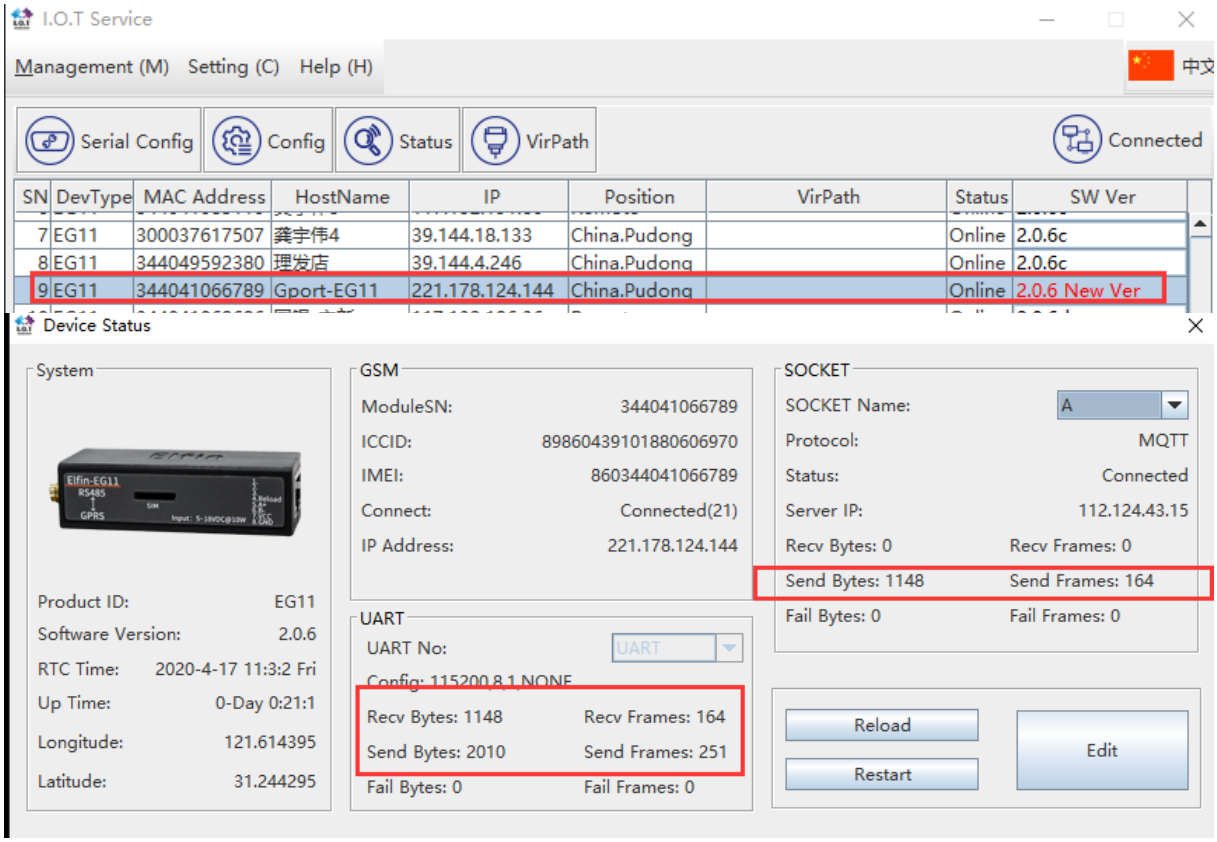
Open MQTT.FX tools, connect to our test MQTT server.



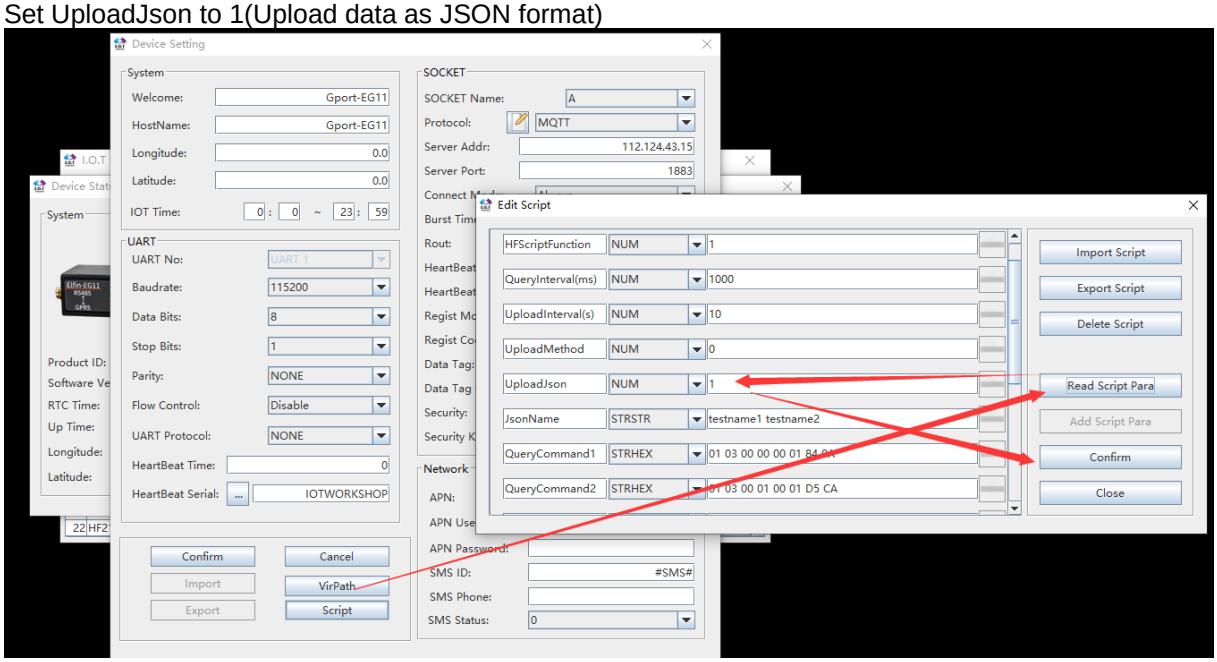
Subscribe the device publish topic(860344041066789/up, device side config use pattern character %IMEI/up), set data as HEX format, then will the uploaded packet



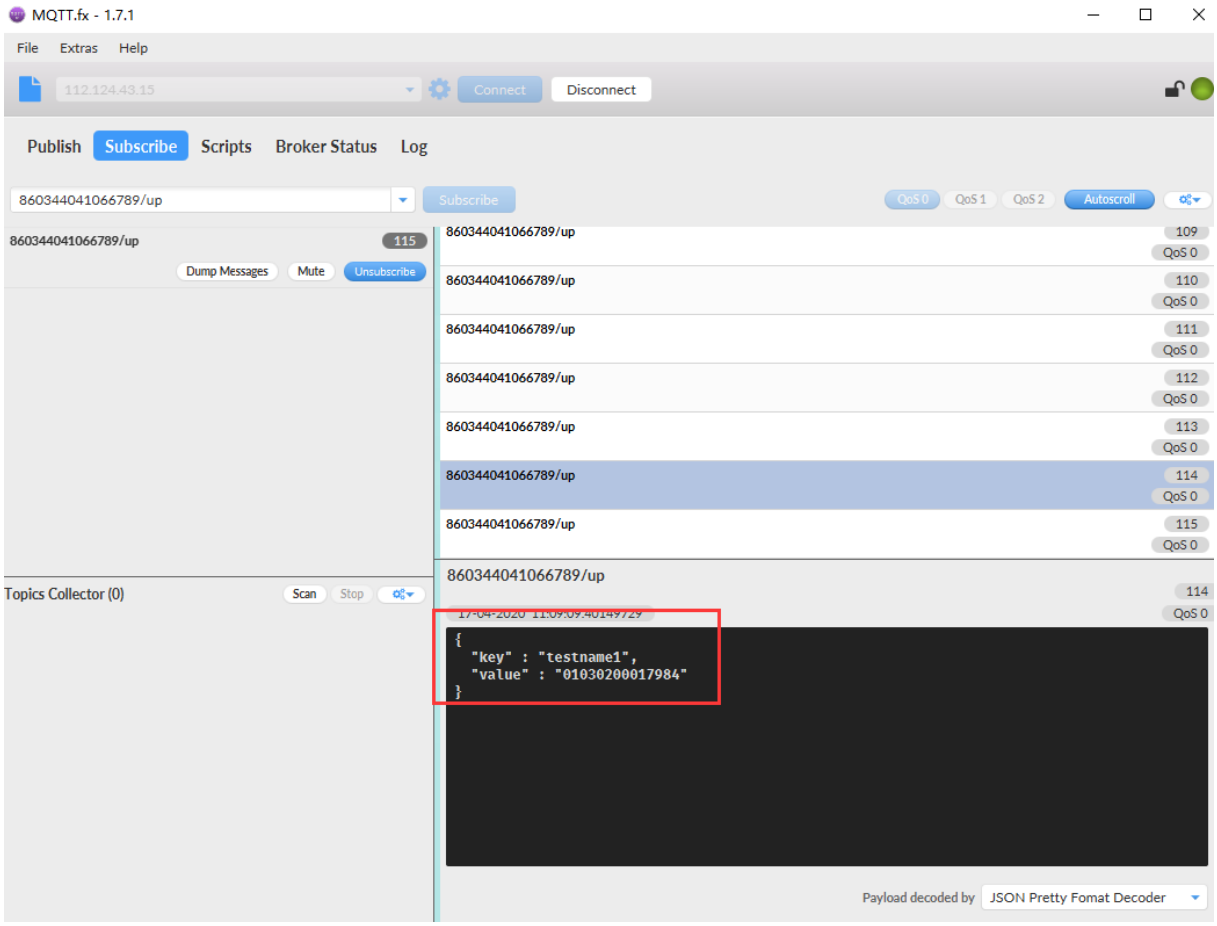
Bound device in IOTService(See details in guide doc). Check device status ,there are Packets count in count.



Set UploadJson to 1(Upload data as JSON format)



Check MQTT.FX received data as following.



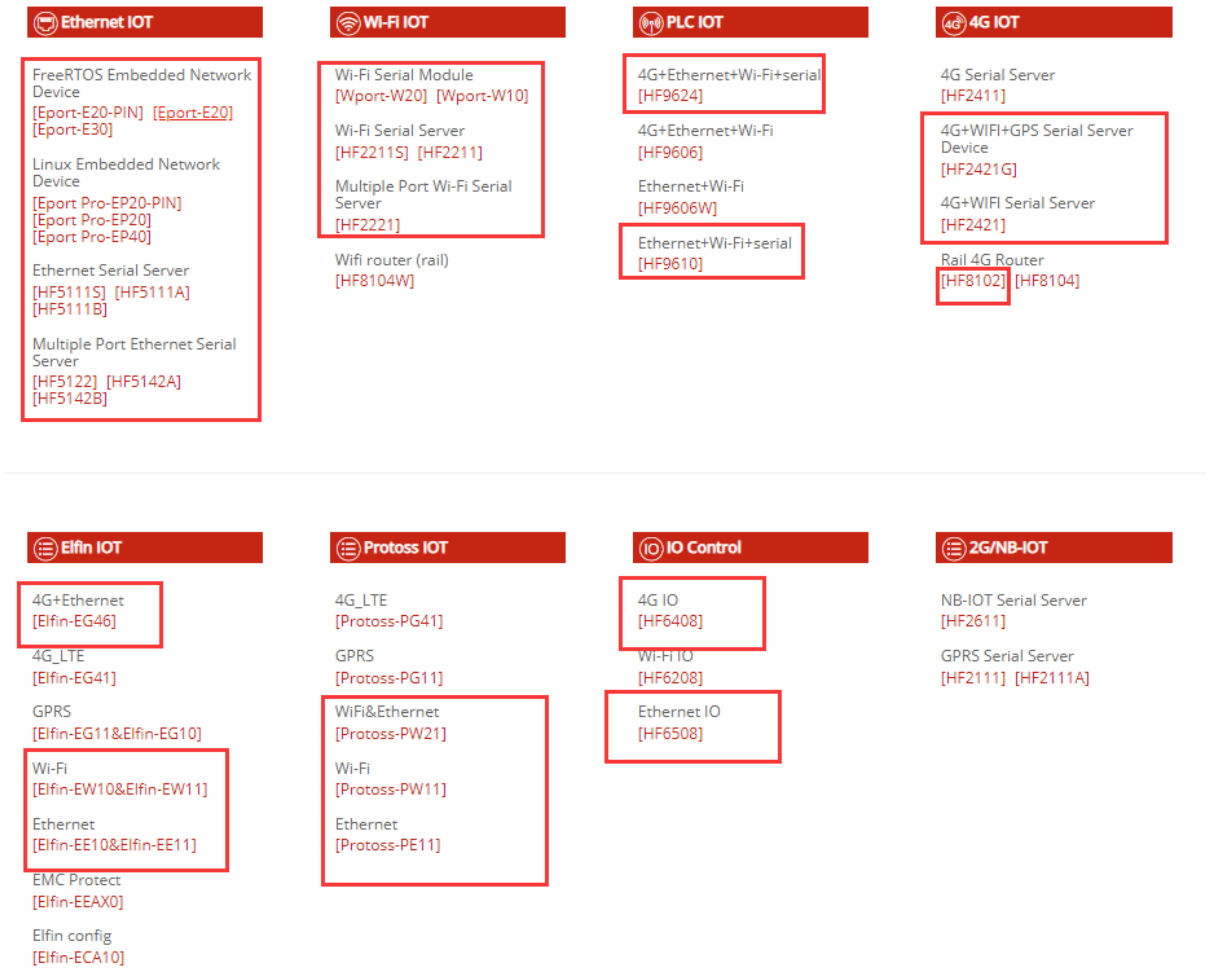
Note:

1, This function use HIS script to cofnig device parameters, but actually HIS Script can do more than this such as modify the received UART packet. See more in following link(Except EG11, EG10, HF2111A, PG11, these products does not support this HIS full function as the following link)

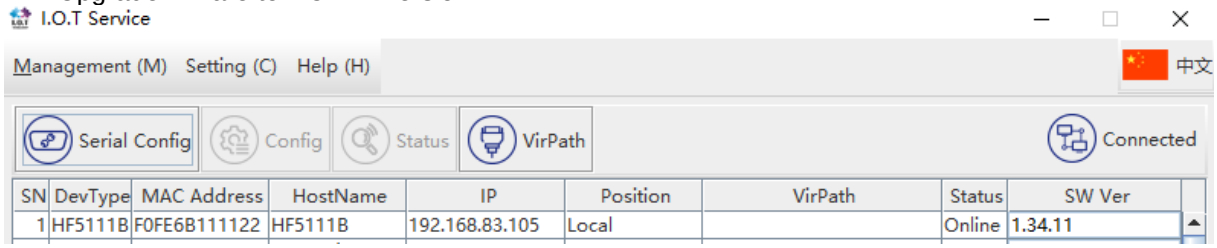
http://www.hi-flying.com/download-center-1/application-notes-1/download-item-his-script

2.2. Ethernet and Wi-Fi Products Test

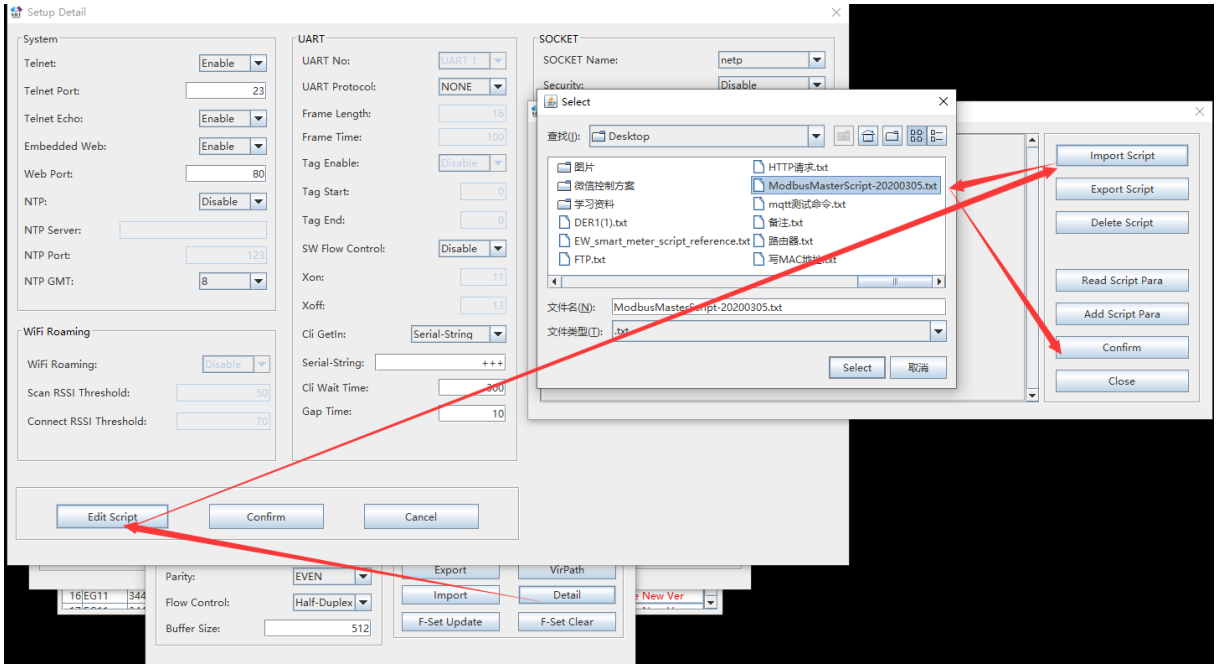
Ethernet and Wi-Fi products use IOTService tools to config via network, the detailed products supports this function is as following, this case take HF5111B for example.



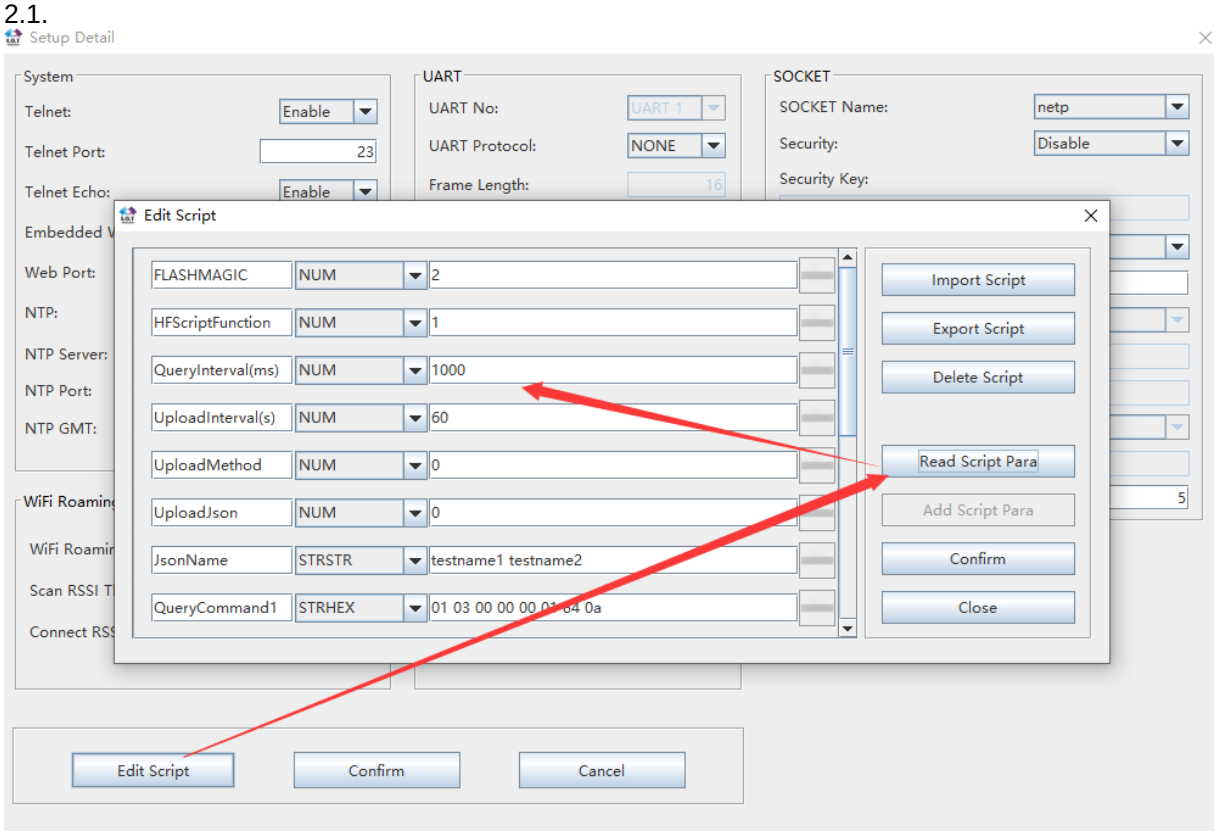
Upgrade firmare to 1.34.11 version.



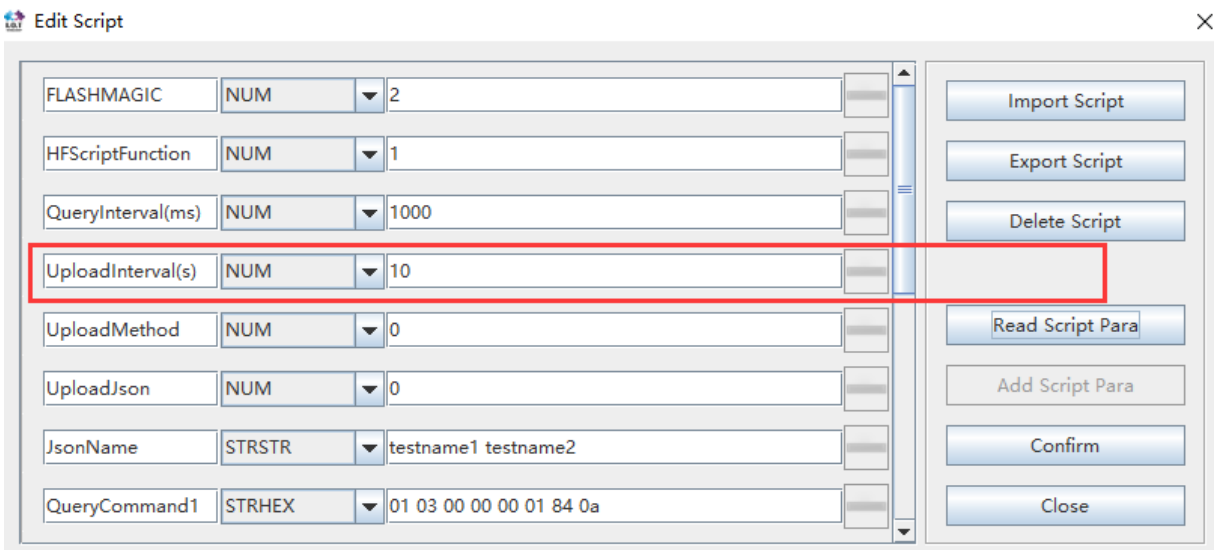
Load config Script file(ModbusMasterScript).



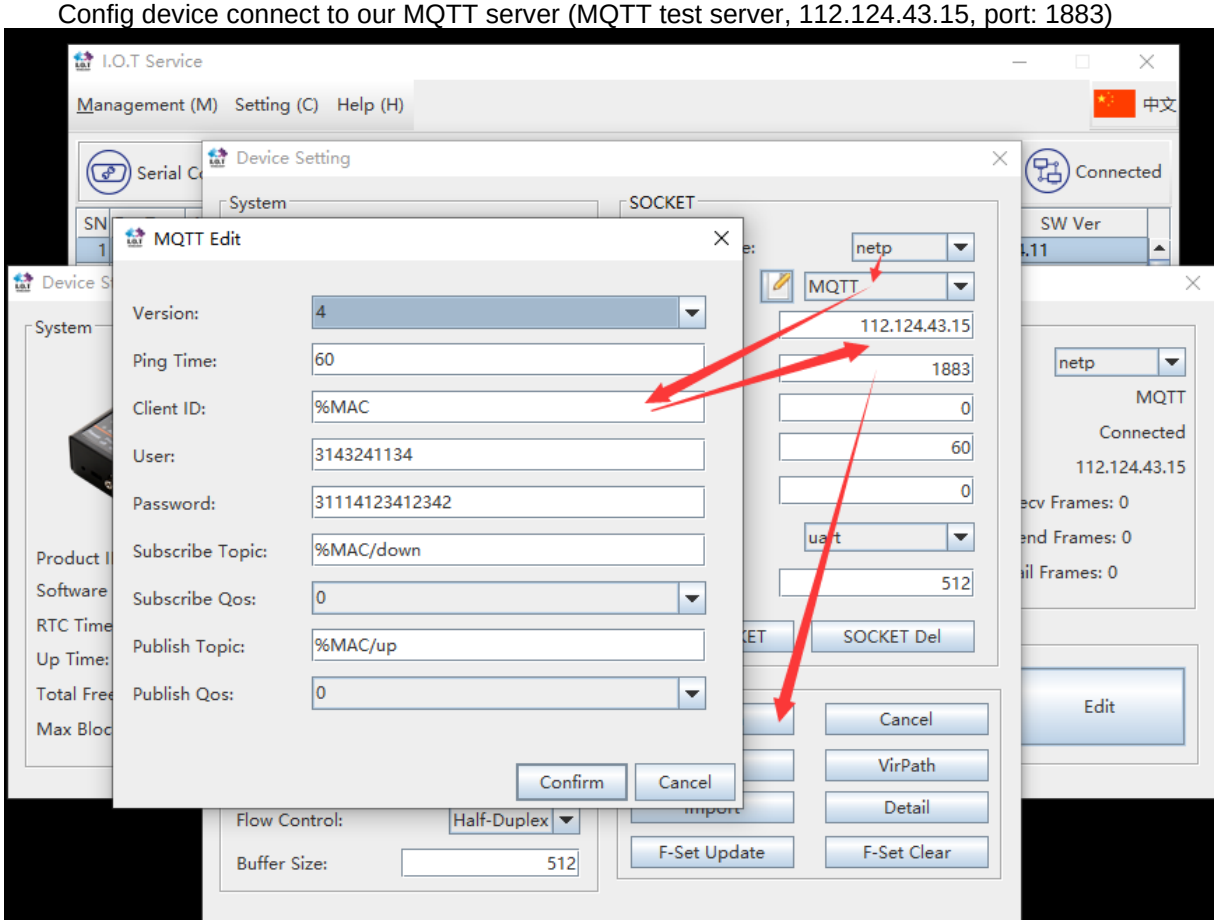
Read Scirpt Para. The detailed parameter function is the same as previous described. in chapter 2.1.



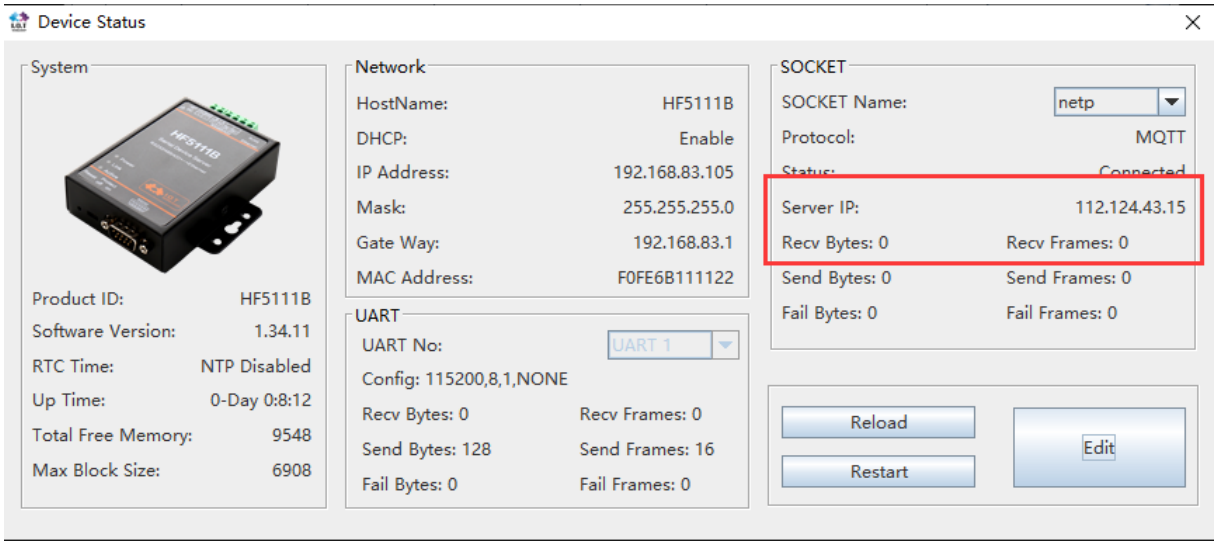
Set UploadInterval to 10 for fast query.



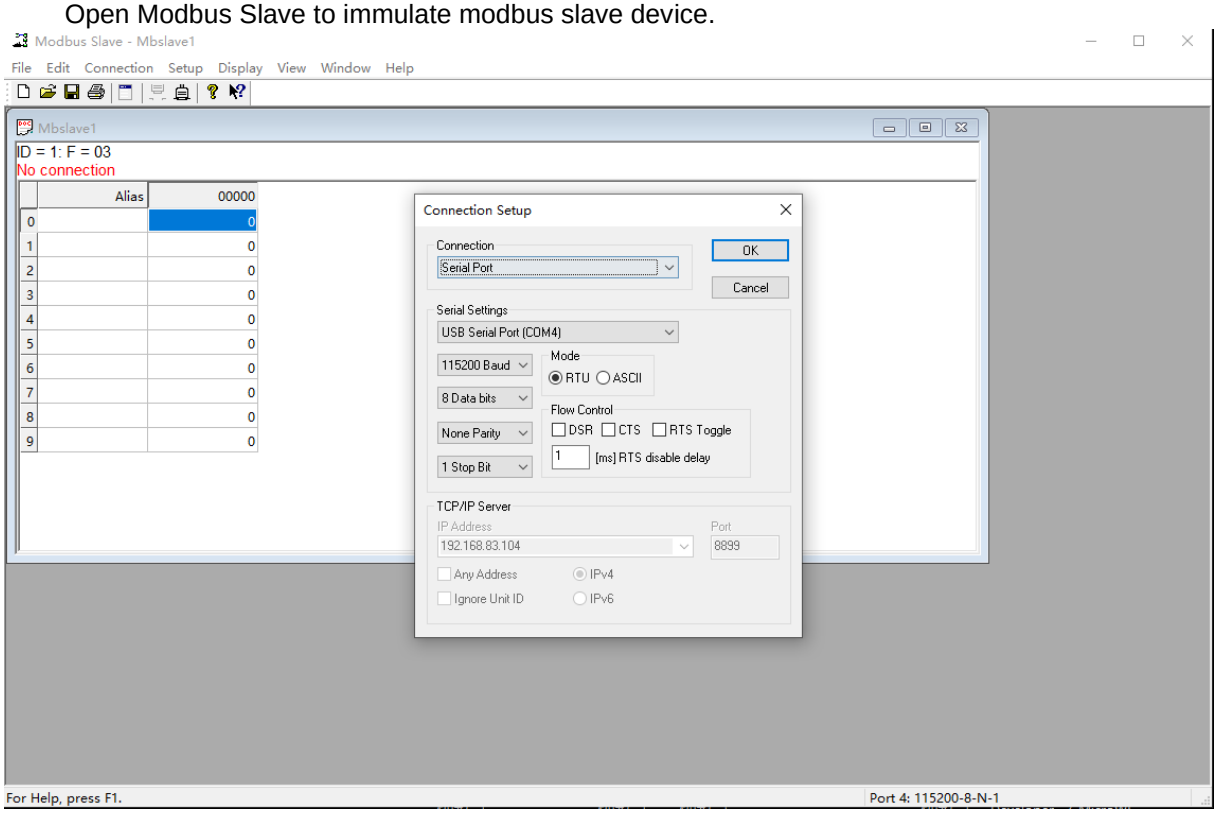
Config device connect to our MQTT server (MQTT test server, 112.124.43.15, port: 1883)



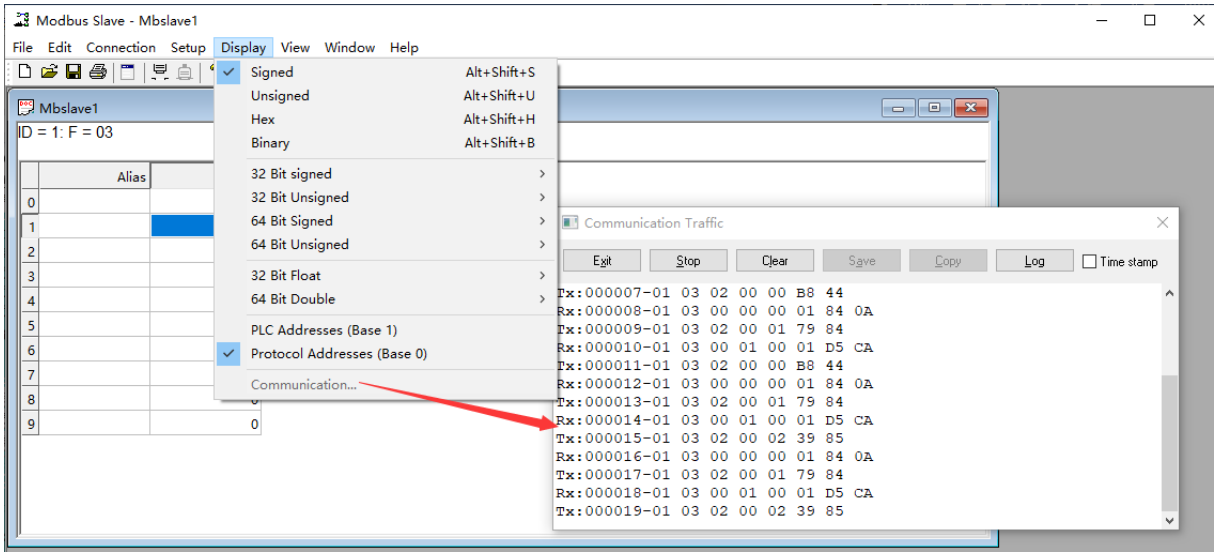
Check device Status that shows it connect to the server.



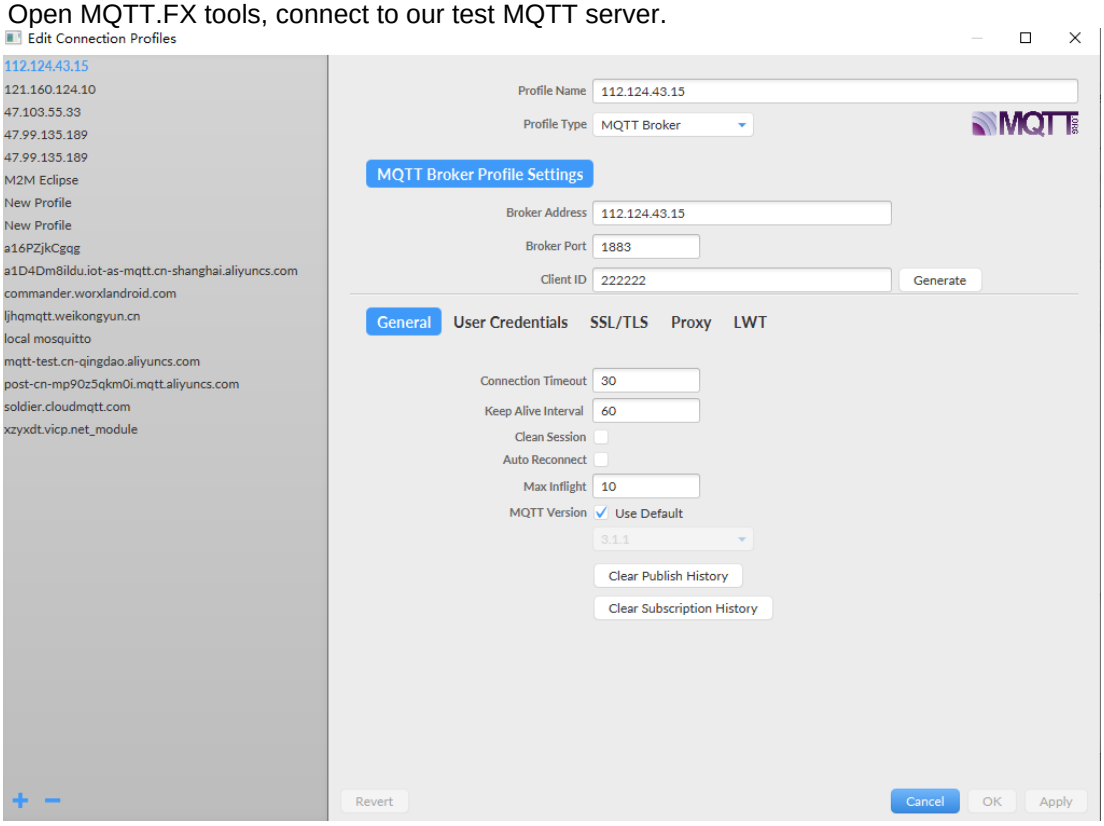
Open Modbus Slave to immulate modbus slave device.



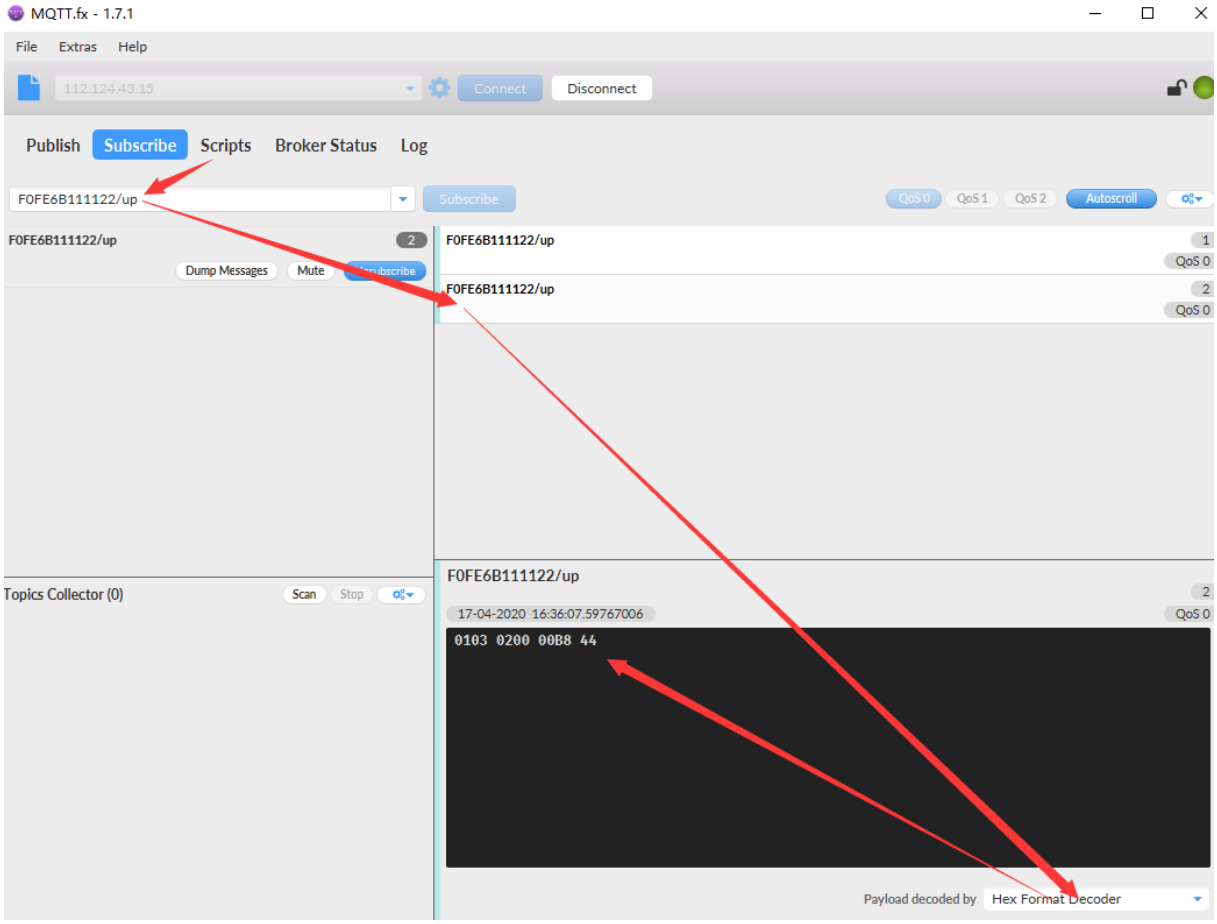
Check the interact packet.



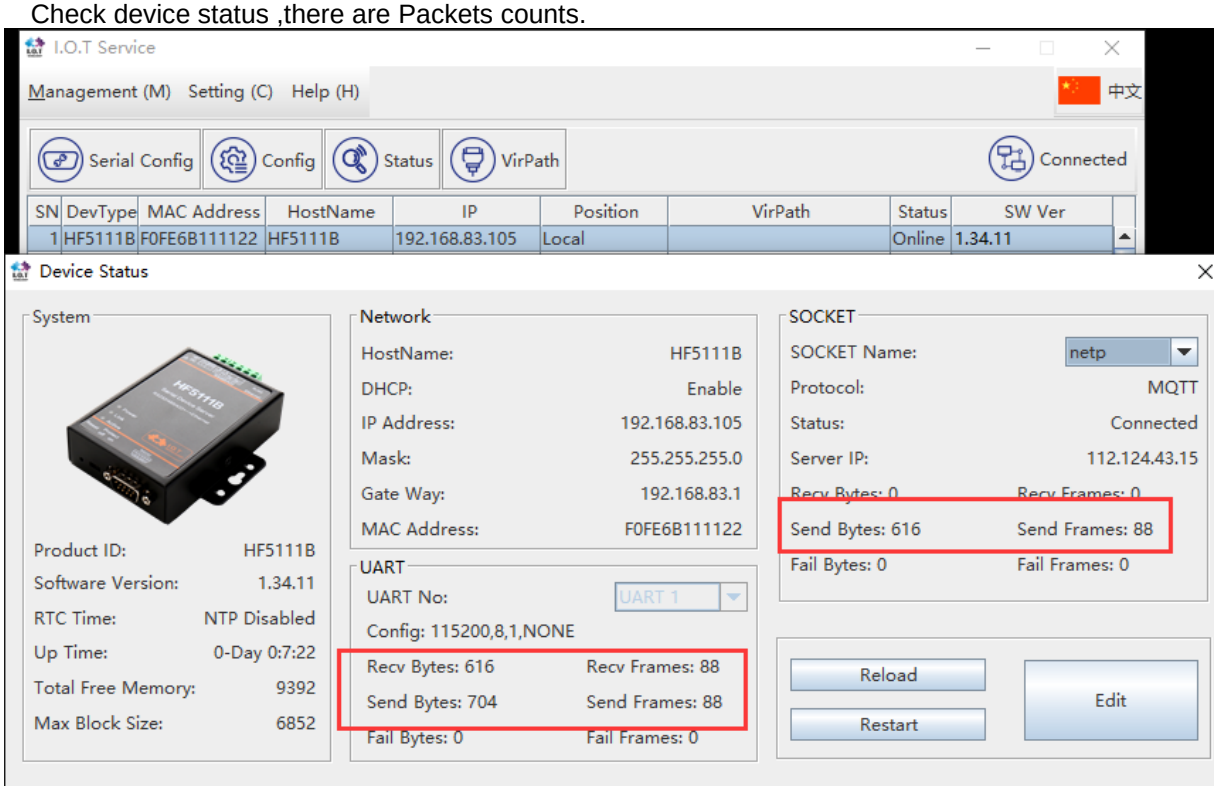
Open MQTT.FX tools, connect to our test MQTT server.



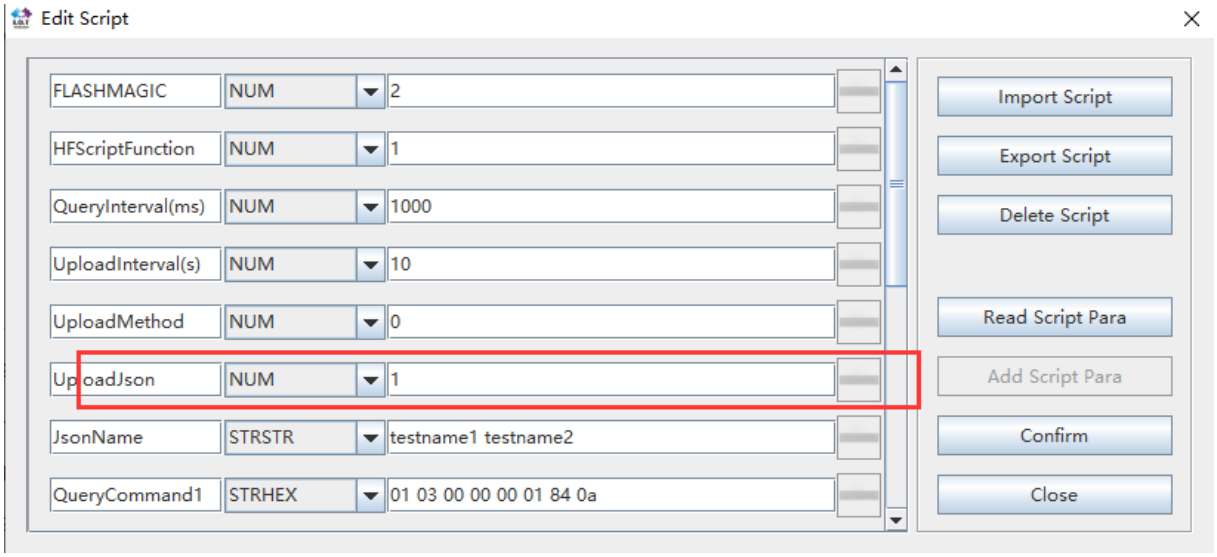
Subscribe the device publish topic(860344041066789/up, device side config use pattern character %IMEI/up), set data as HEX format, then will the uploaded packet



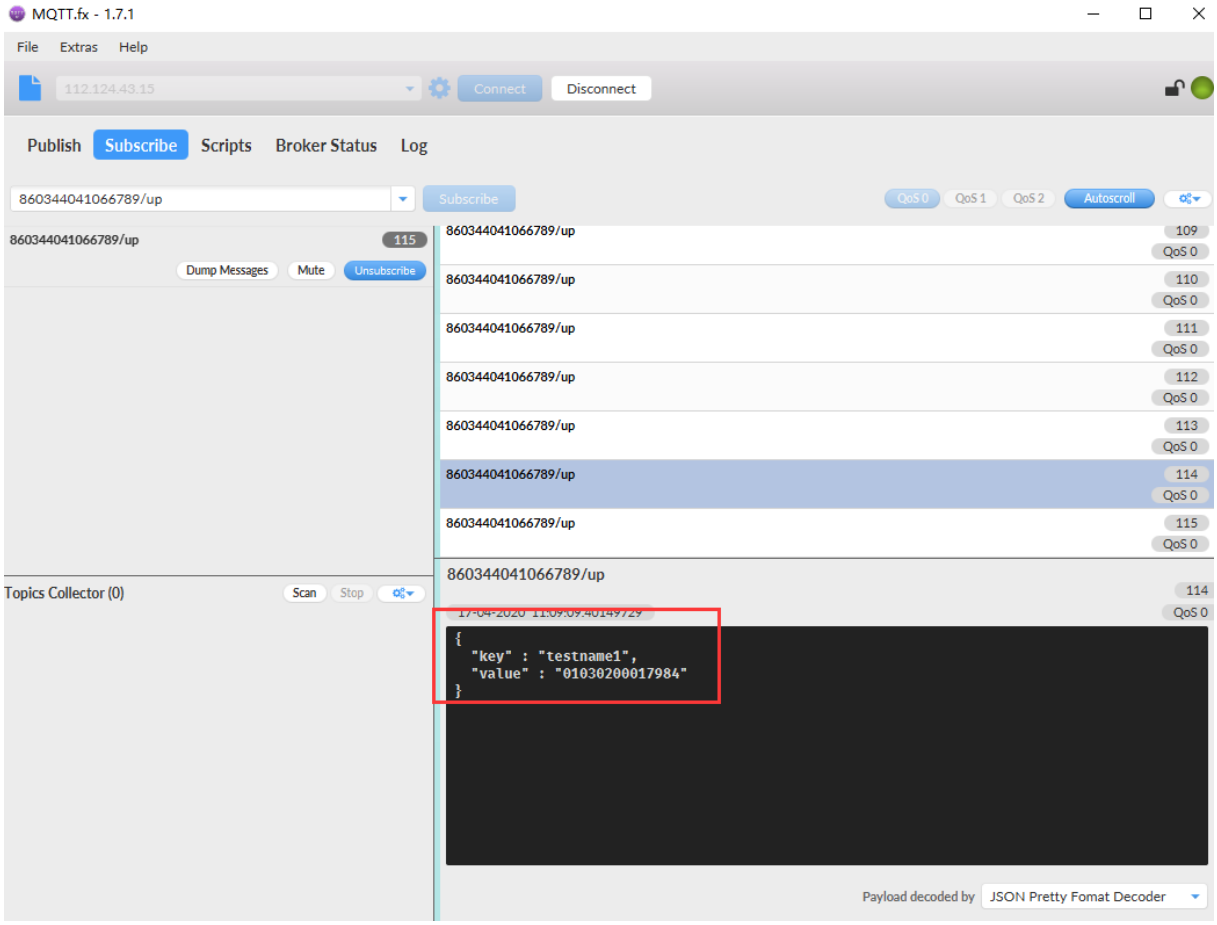
Check device status ,there are Packets counts.



Set UploadJson to 1(Upload data as JSON format)



Check MQTT.FX received data as following.



Note:

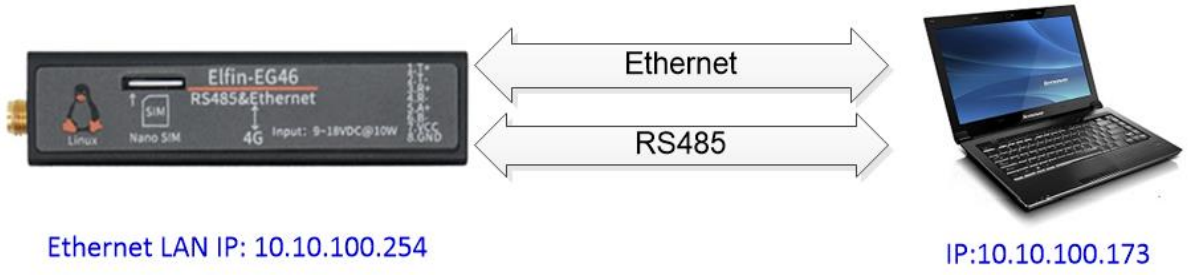
1, This function use HIS script to cofnig device parameters, but actually HIS Script can do more than this such as modify the received UART packet. See more in following link)

http://www.hi-flying.com/download-center-1/application-notes-1/download-item-his-script

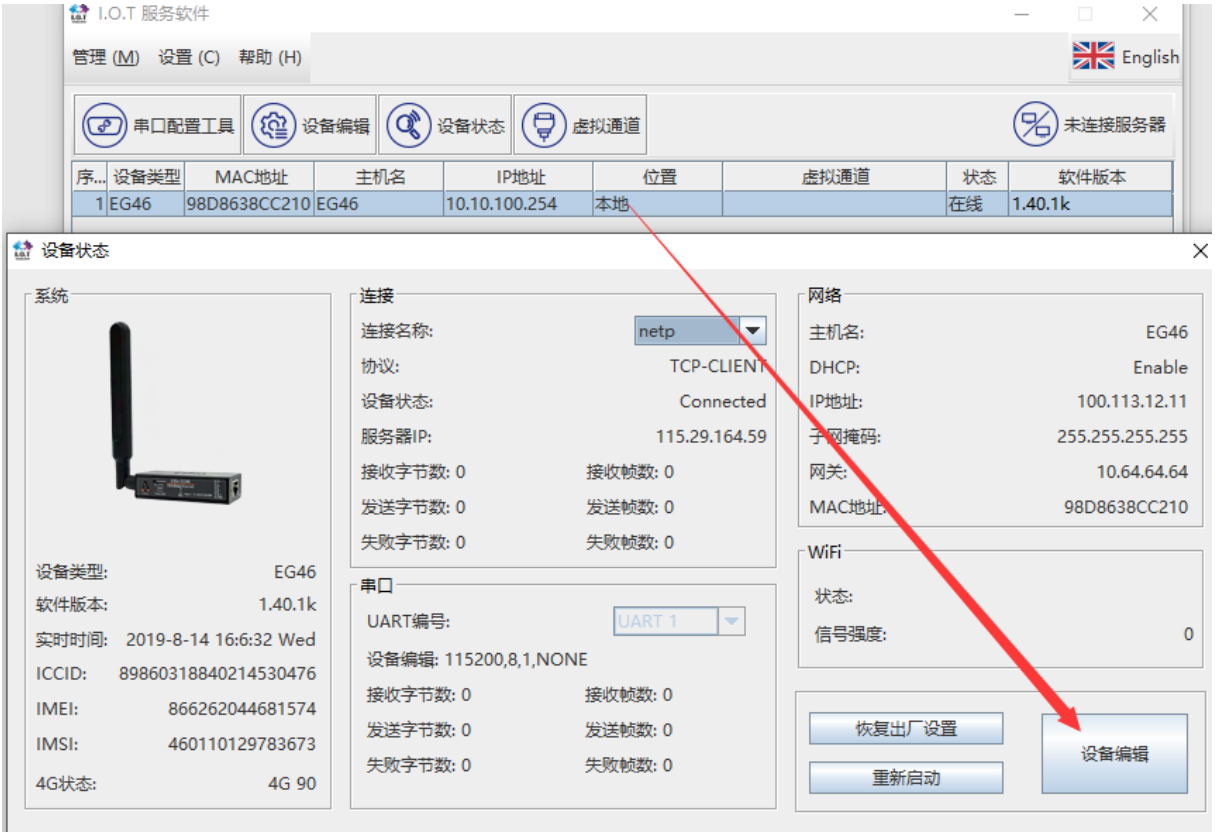
3. COMPLETE MODBUS MASTGER TEST

3.1. MODBUS MASTER POLLING FUNCTION

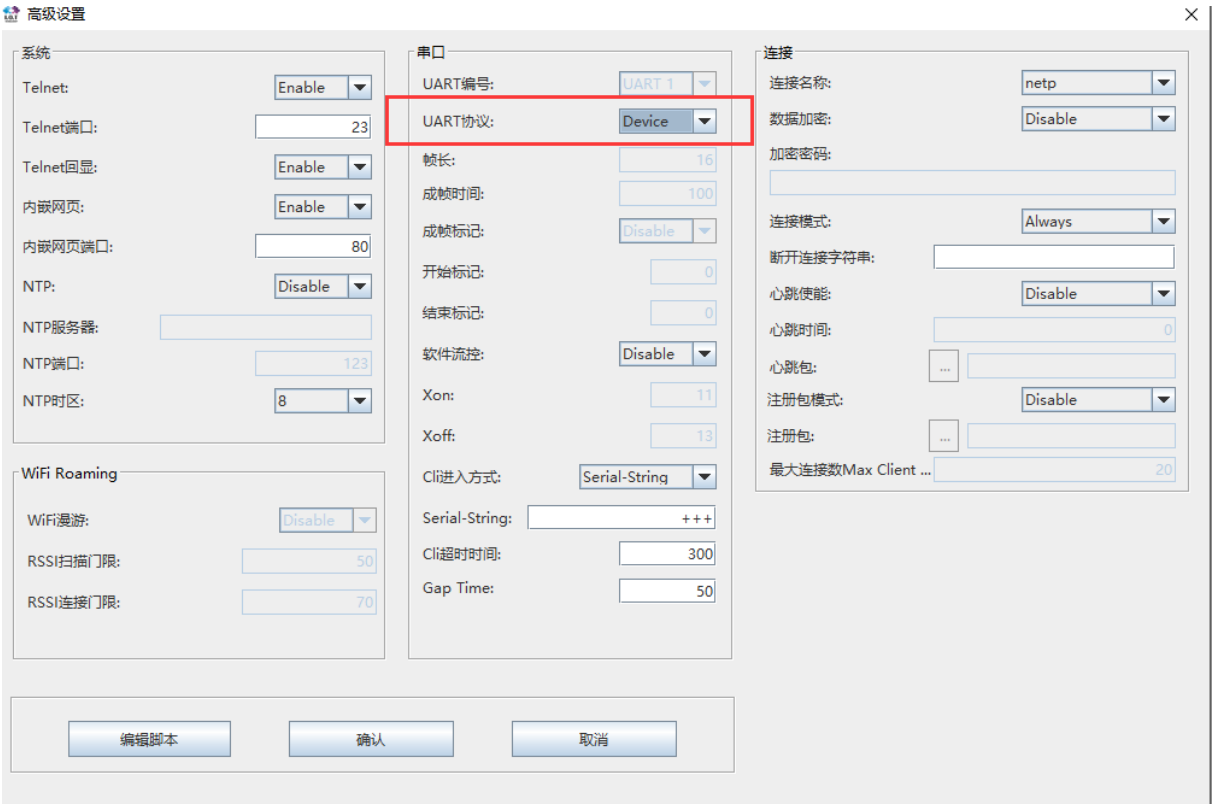
PC network cable and serial cable connection EG46



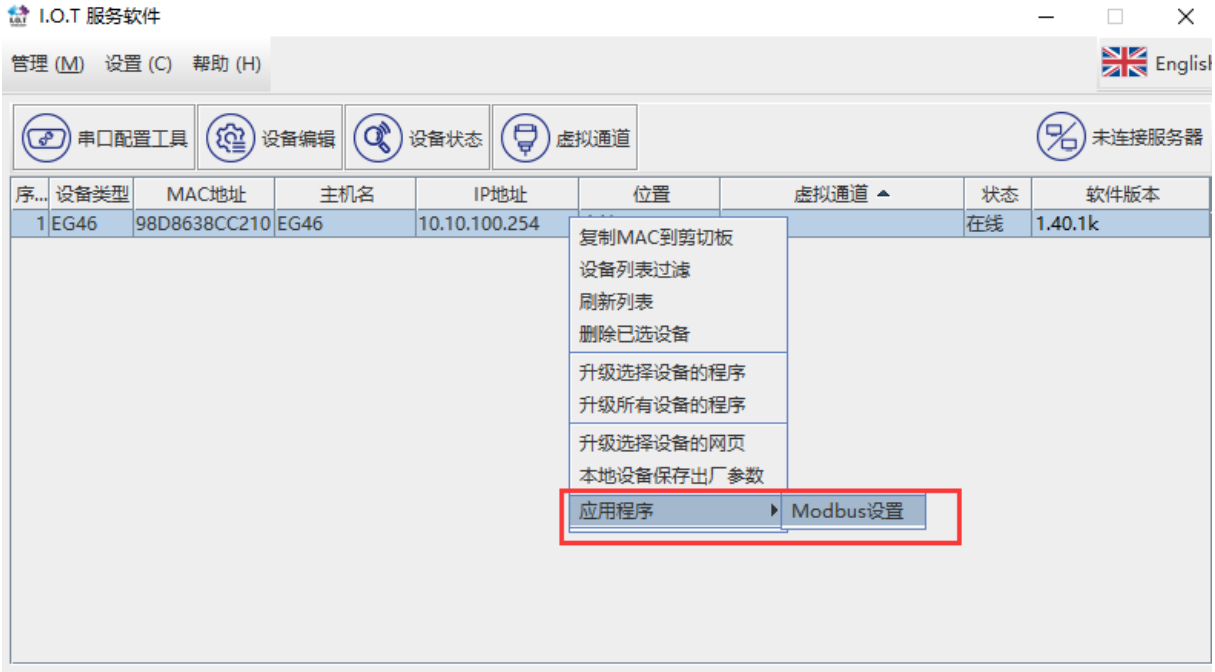
Open the IOTService Tool and click Edit.



Modify the serial port protocol to device (Modbus master function, the serial port cannot use the cli command in this configuration state).

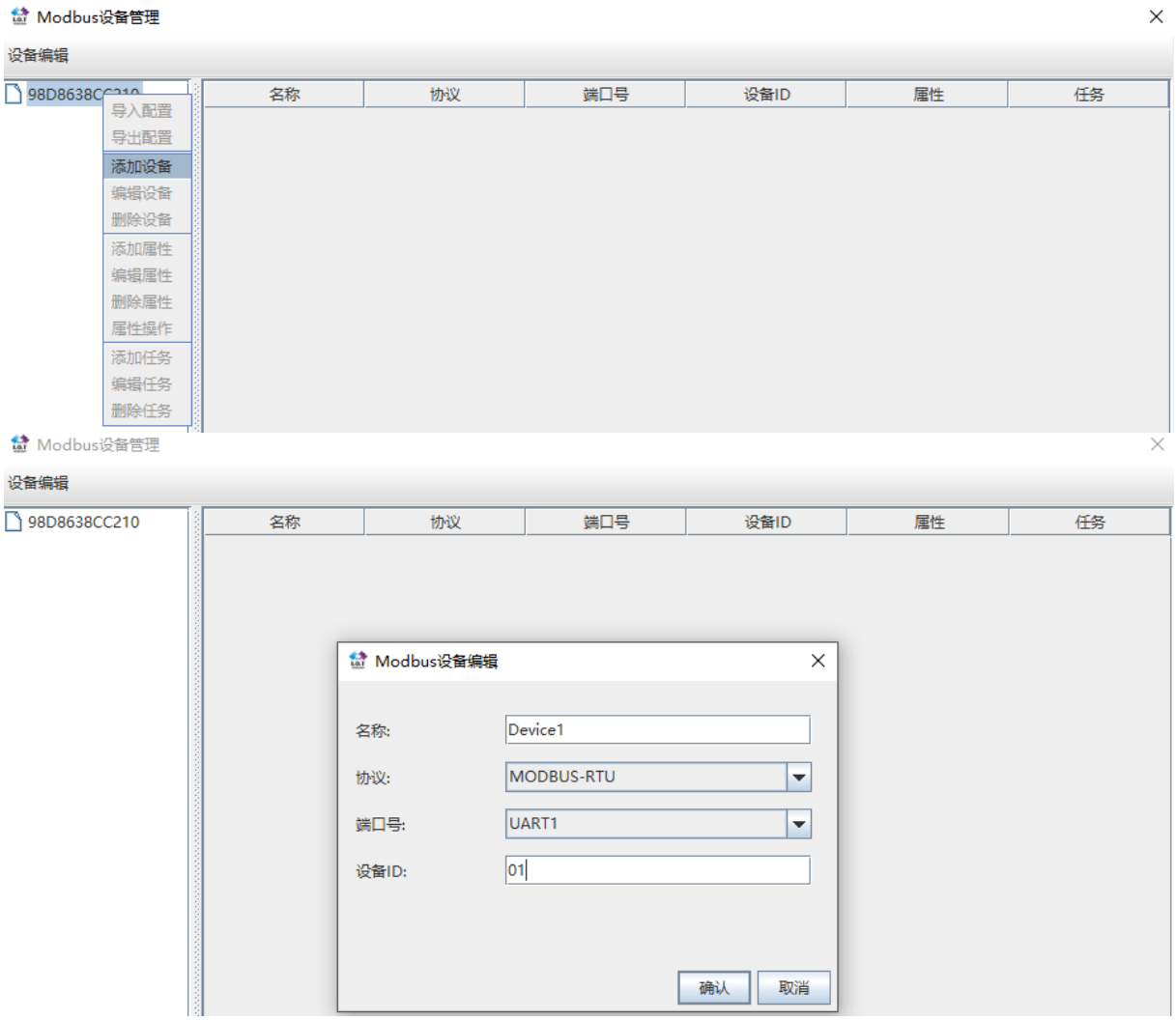


Right click on the device information and click on the Modbus settings page.



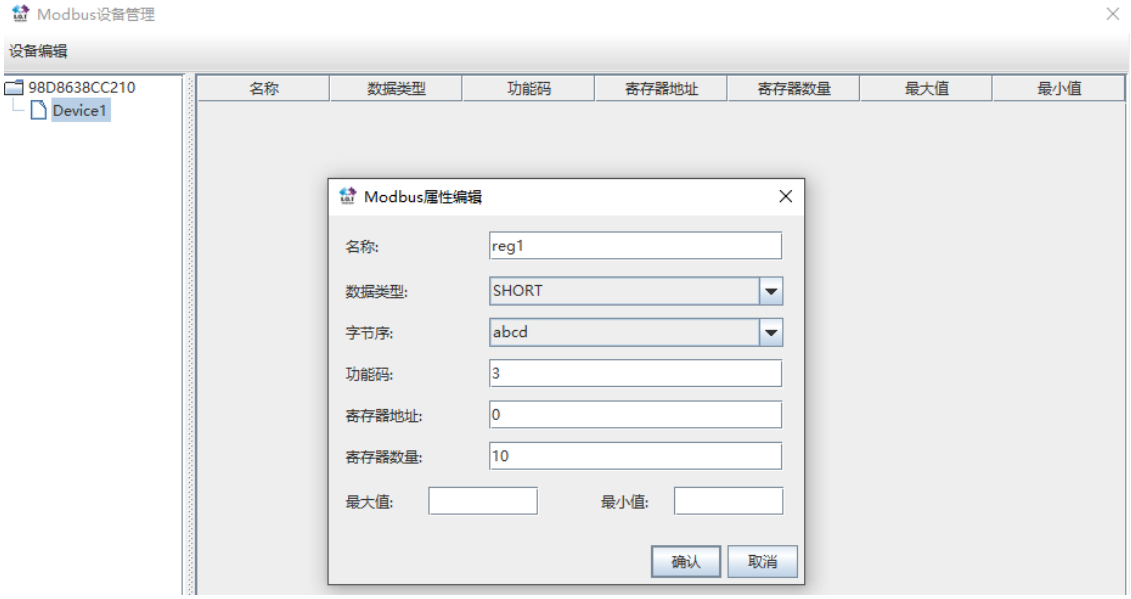
Right click on EG46 and select Add Device.

- Name: Modbus slave name
- Protocol: Modbus slave device protocol type
- Port number: Serial port number (EG46 only 1 serial port, fill in UART1)
- Device ID: Modbus slave address

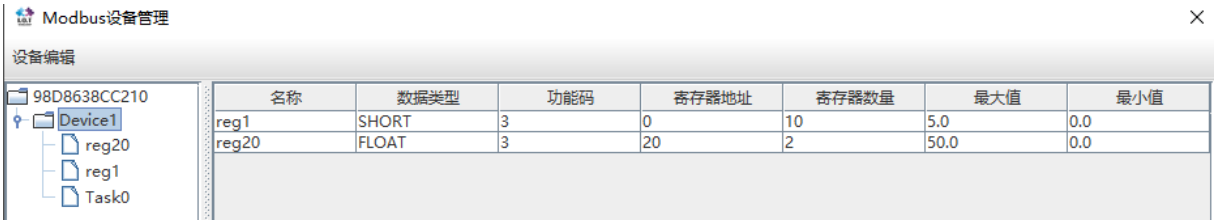


Right-click the created Device device and select Add Attribute.

- Name: The name used when data is reported. When polling multiple registers, the data with no name is automatically reported according to the regXX name.
- Data type: short, bool or float, affecting data reporting.
  - bool: The report is false or true.
  - short: The data is short, and the length of 2 bytes is used as a register value.
  - float: The data is in floating point number and the length of 4 bytes is used as a register value.
- Endian: The byte order of the data, whether the high order is in the front or the low order is in the front, and so on.
  - abcd：big endian mode
  - badc：big endian mode and swap byte order.
  - dcba：little endian mode
  - cdab：little endian mode and swap byte order.
- Function code: The function code of modbus polling.
- Register address: modbus polling register start address
- Number of registers: The number of modbus polling registers.
- Maximum: The maximum data allowed under the float data type.
- Minimum value: The minimum value of data allowed under the float data type.

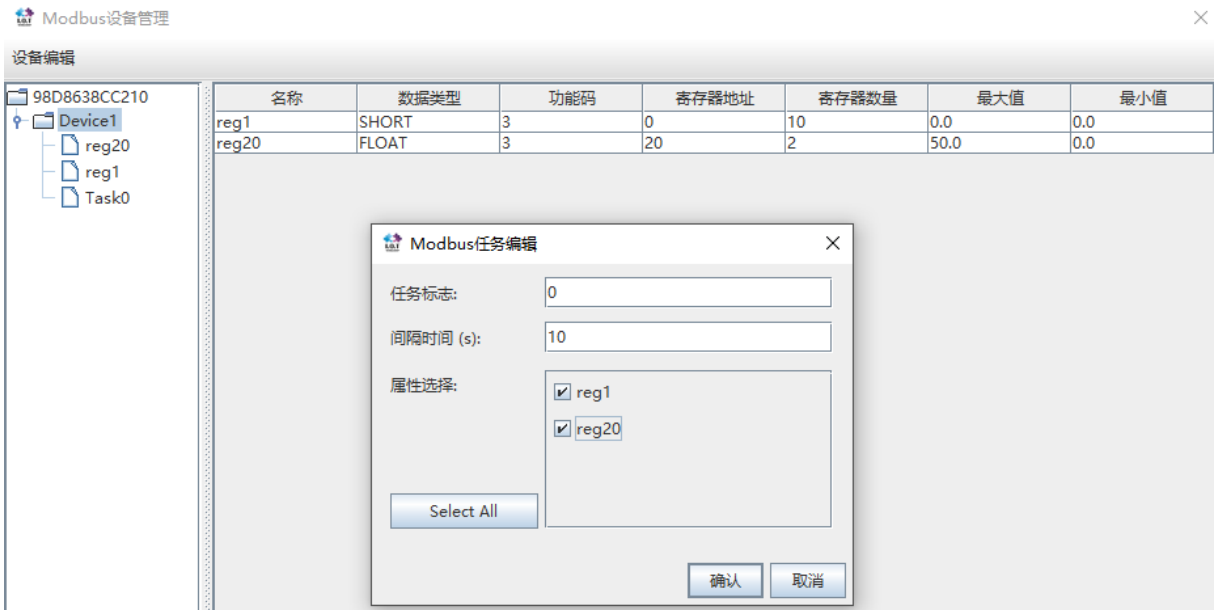


Two query instructions are created as follows, reg20 is of type float.



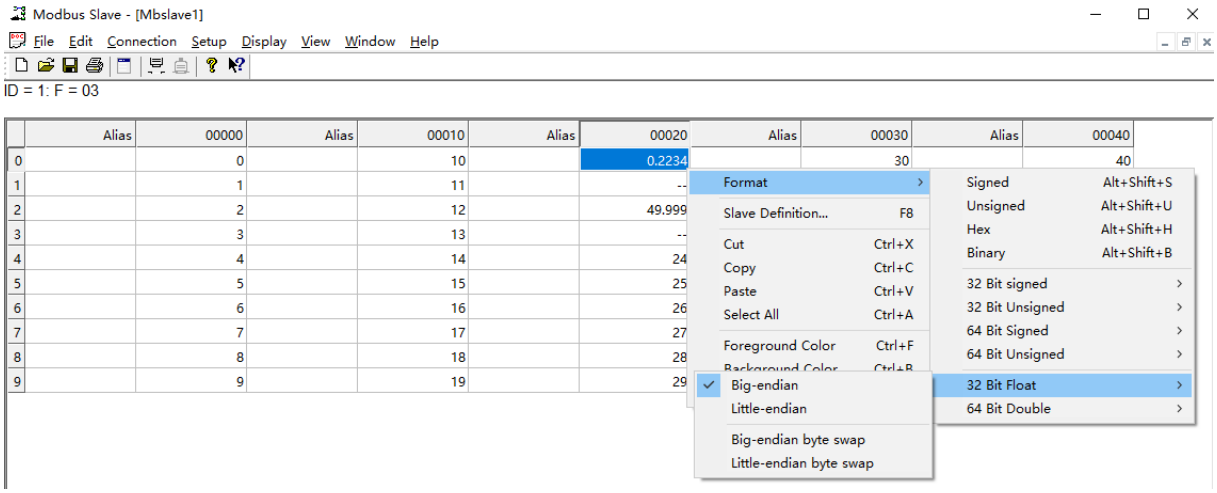
Create a task, set the collection time and collection range. After the task is created, you need to restart the device to take effect.

- Task tag: Task id name
- Interval: A complete polling cycle.
- Attribute selection: The name of the register participating in the polling.
-

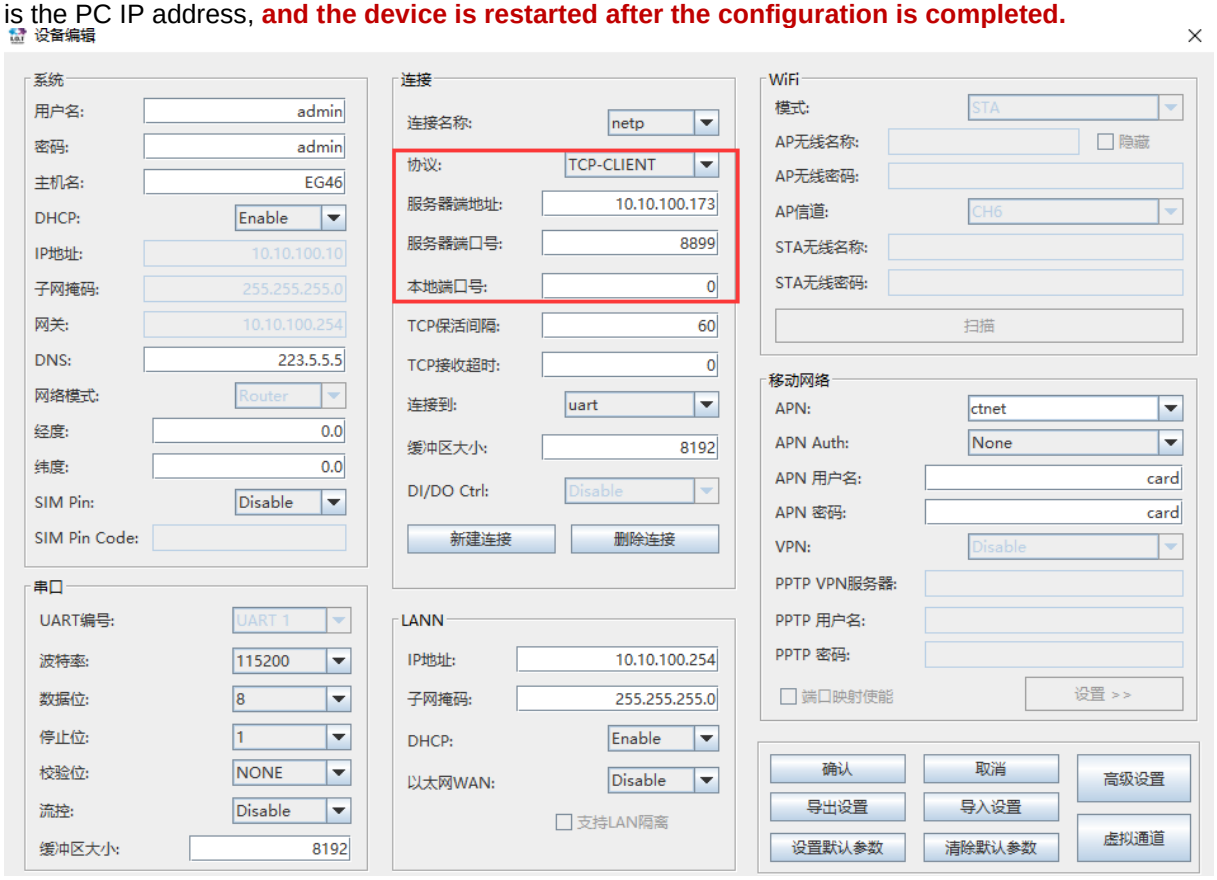


The Modbus slave software is opened on the PC side, and the Modbus device is simulated. As shown in the figure below, the address is set to 1, the function code is 3, the register value is set as

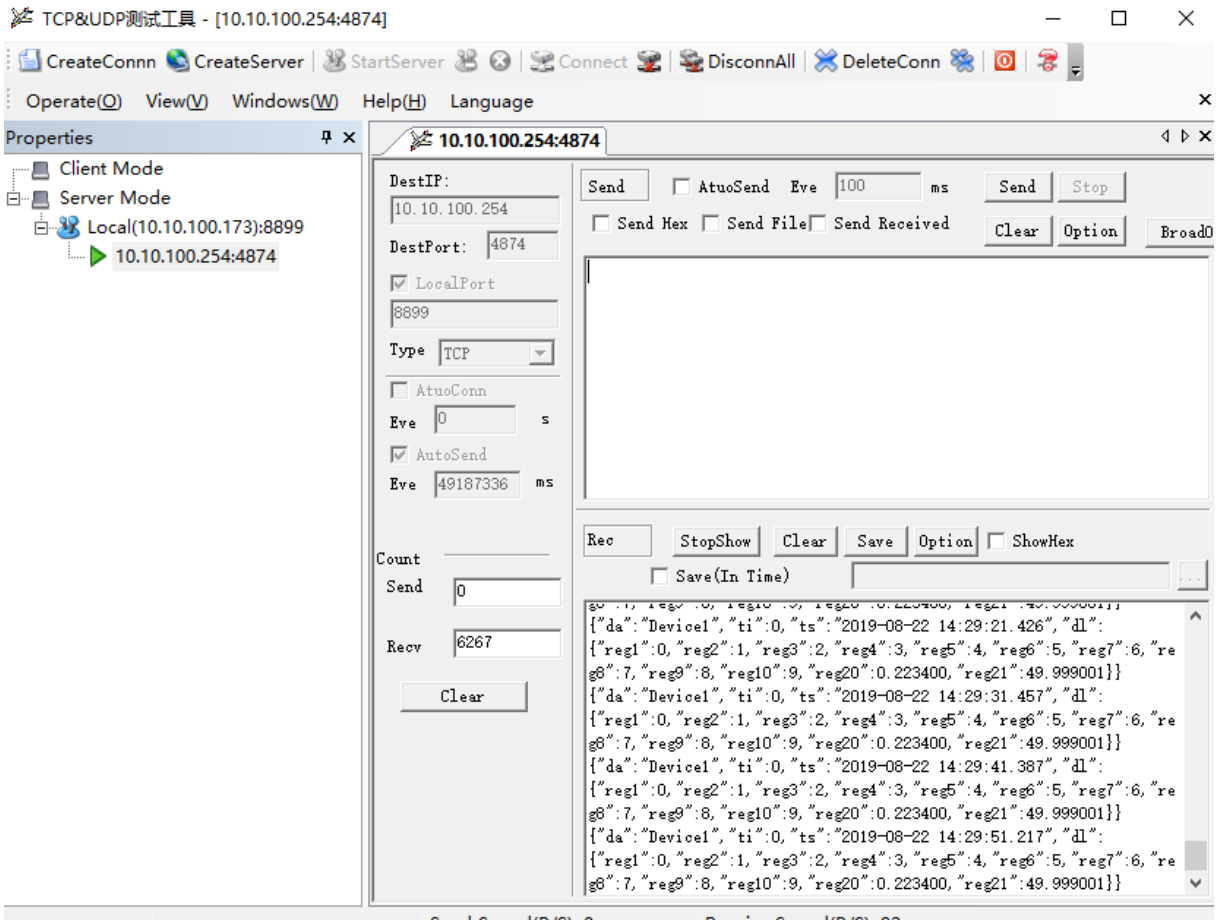
shown below, and the data of the address 20 and 22 is configured as the big end floating point number.



After the above configuration, the serial port is set up. After that, you can set the data report. You can use TCP, HTTP, MQTT, etc. The following example uses TCP Client to connect to the PC to view the data reported after polling. The aircraft software establishes TCP Server, port 8899, 10.10.100.173 is the PC IP address, **and the device is restarted after the configuration is completed.**



From the PC side, you can see the message sent after the device polls.



The interactive process is described as follows:

According to the configuration of the serial port in the above example, the product automatically initiates polling data every 10 seconds, and a single poll initiates two instructions.

Instruction 1: modbus address 1, function code 3, register start address 0, length 10.
Device Polling Command: 01 03 00 00 00 0A C5 CD
Modbus Slave reply: 01 03 14 00 00 00 00 01 00 02 00 03 00 04 00 05 00 06 00 07 00 08 00 09 CD 51

Instruction 2: modbus address 1, function code 3, register start address 20, length 4.
Device polling command: 01 03 00 14 00 04 04 0D
Modbus Slave reply: 01 03 08 3E 64 C2 F8 42 47 FE FA E7 F1

For the received data, the data is parsed according to the floating point number or the short type in the configuration, and the configuration information is reported and reported.

```
{
    "da": "Device1",            //设备名称
    "ti": 0,                    //任务标记
    "ts": "2019-08-22 14:14:05.409",    //时间戳
    "dl": {
        "reg1": 0,                      //寄存器名称，short 类型。
        "reg2": 1,
        "reg3": 2,
        "reg4": 3,
        "reg5": 4,
        "reg6": 5,
        "reg7": 6,
```

```
            "reg8": 7,
            "reg9": 8,
            "reg10": 9,
            "reg20": 0.223400,              //寄存器名称，浮点数类型。
            "reg21": 49.999001
        }
}
```

Remark:

1、 Support for creating multiple modbus slave devices.
2、 Support several data points.
3、 Support batch import and export configuration information, directly import parameters, and complete parameter configuration with one click.