

# VT-DTU500C

## User Manual



Figure 1 VT-DTU500C

**CONTENT**

1. OVERVIEW .....	3
2. FEATURES .....	4
3. TECHNICAL PARAMETERS .....	5
4. INSTRUCTION .....	7
4.1 HARDWARE SPECIFICATION .....	7
5. CONFIGURATION .....	9
5.1. Serial Port AT Command Configuration .....	9
5.2. Firmware/Configuration file mode .....	16
6. PRODUCT FUNCTIONS .....	19
6.1. Communication test .....	19
6.2. MODBUS RTU to JSON Conversion Test .....	27
7. AT COMMAND .....	31
7.1. Login and Configuration .....	32
7.2. Serial Port Parameters .....	33
7.3. Network Parameters .....	34
7.4. Registration package and heartbeat package .....	34
7.5. Remote management function .....	34
7.6. MQTT Parameter .....	35
8. AFTER- SERVICE AND TECHNICAL SUPPORT .....	37

## 1. Overview

VT-DTU500C supports 2G GPRS mode. It converts RS485 to 4G, with CAT1 4G transmission rates up to 5Mbps upstream and 10Mbps downstream. It also supports RS485 to 4G conversion and features Rail mounting for easier installation. Compared to traditional wall-mounted 4G DTUs, its width is reduced to approximately 1/4, resulting in a smaller footprint. It uses a terminal block power input, supporting a wide voltage range of 9-24V. The shell is made of high-temperature-resistant, flame-retardant alloy plastic, meeting industrial fire protection requirements.

VT-DTU500C not only has the function of registering messages, heartbeat packets, but also the relatively new MQTT, Modbus RTU to JSON to connect to the cloud server. It has the features of high-speed transmission, low latency, support for new technologies and so on.

VT-DTU500C supports device configuration, firmware upgrades, and MQTT/JSON configuration via the serial port. Furthermore, a large number of distributed devices can be remotely and centrally managed through a server, enabling remote configuration, viewing, and program upgrades. In conjunction with the public cloud, web-based device management, web-based data viewing, and remote control are also possible.

Collected data can be uploaded in JSON format, enabling automated data collection. Data collection supports Modbus RTU, 645 instrument version 97, 645 instrument version 07, and various non-standard RS485 protocols. Vircom allows users to configure the uploaded data format and JSON keywords. Uploads support MQTT, HTTP POST, HTTP GET, transparent transmission protocols, and various non-standard network protocols.

Supports edge computing functions, including data overrun alarms, data panning and scaling, data change uploads, device offline alarms, device independent data collection, and automatic device connection. This function is usually used in conjunction with the JSON function.

A specially designed watchdog circuit ensures long-term stable operation of the 4G module.

The product supports an industrial temperature range of  $-40^{\circ}\text{C}$  to  $85^{\circ}\text{C}$  and has passed electromagnetic compatibility tests, including those for static electricity.

### Applications:

1. Data collection in the Industrial Internet and industrial automation sectors.
2. New energy, solar power, wind power generation, and power data collection and monitoring.
3. Access control and security.
4. Collection and monitoring of hydrological, meteorological, and environmental data.
5. Intelligent transportation and vehicle-mounted data collection.
6. Smart agriculture, smart greenhouses, and smart animal husbandry.

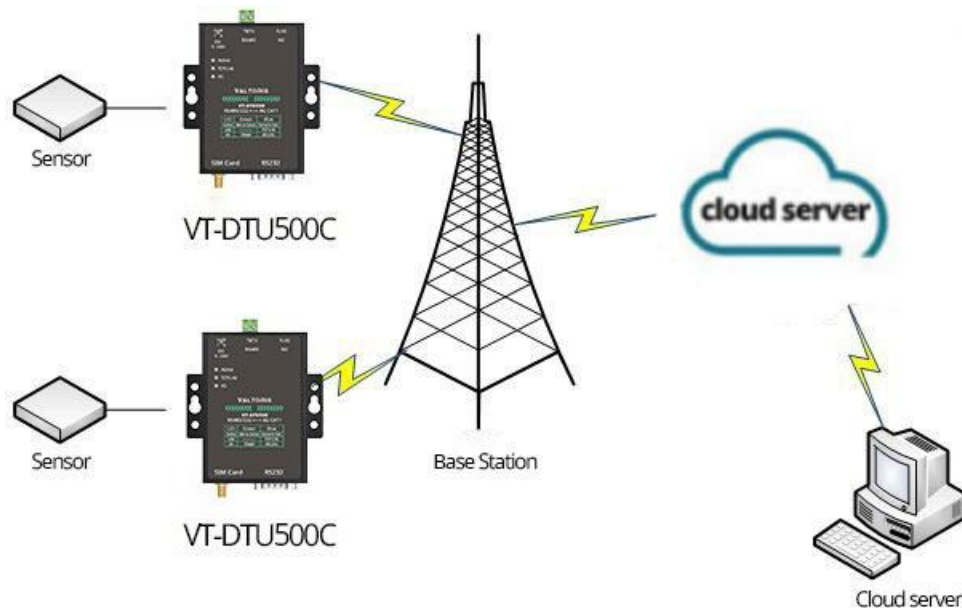


Figure 2. Application Environment

## 2. Features

- 1 Support 3 modes, TD-LTE/ FDD-LTE/ GSM, including Unicom, 4G, 2G, mobile 4G, 2G and telecom 4G.

- 2 Support TCP client, UDP mode.
- 3 Serial port support 300~921600 baud rate, support 5~8 data bits, support no check, odd check, even check, support 1~2 stop bit.
- 4 Support serial port (RS232/485) To 4G.
- 5 Support serial port transparent transmission, Modbus RTU to Modbus TCP, MQTT protocol.
- 6 Support serial port AT command configuration, and VIRCUM software to view some parameters.
- 7 Support for serial port configuration of MQTT parameters.
- 8 Support DTL-645/Modbus RTU automatic collection and conversion to cloud platform JSON format.
- 9 The VT-DTU500C firmware can be updated on the device through the serial port, and the firmware can be updated on the server side through the Vircom software.
- 10 Support server side remote device management, device configuration, device upgrade.

### 3. Technical parameters

Table 1 VT-DTU500 Technical parameter

Specifications	
<b>Support Mode</b>	4G CAT1 support 3 modes: B1/B3/B5/B8@FDD LTE B34/B38/B39/B40/B41@TDD-LTE B3/B8@GSM including Unicom 4G, 2G, Mobile 4G, 2G and telecome 4G Internet.
<b>Transmission rate</b>	LTE: Max 10Mbps (downlink) /Max 5 Mbps (uplink) GPRS: 85.6Kbps (downlink) /Max85.6Kbps (uplink)
<b>SIM card</b>	Voltage: 3V, 1.8V; Mini SIMCard (2FF), Size: 25mm × 15mm

<b>Antenna</b>	50Ω/SMA Rubber stick antenna or suction cup antenna is optional
<b>Interface</b>	RS232/RS485
<b>Serial port data</b>	Baud rate: 300~921600bps; Digit bit: 5~8 bits; Stop bits: 1~2 bits; Check bits: none, even, odd.
<b>Power supply</b>	Q2.1 socket, Can be customized for power terminal input.
<b>Power supply:</b>	DC 9V~24V
<b>Working current</b>	dial/4G When communication 50mA@12V, free 25mA@12V
<b>Operating temperature:</b>	-40~85°C
<b>Storage temperature</b>	-45~165°C
<b>Humidity range</b>	5~95% relative humidity
<b>Device size</b>	Length×Width×Height=9.4cm×6.5cm×2.5cm



4G Signal light	Power supply signal	<p>The blue flashing indicates that 4G is dialing. The dialing starts 15 seconds after the system is powered on. Generally, it can be dialed within 10 seconds.</p> <p>Blue is always bright, indicating 4G connectivity</p>
-----------------	---------------------	--



Figure 4 interface picture

Figure 4 shows the upper end interface:

1. Input power supply: interface Q2.1 socket, input voltage DC+9V~+24VDC, power need above 3W. default adapter is 12V. Can be customized for power terminal input.
2. RS485 interface: RS485 signal input, Be careful not to connect the power.
3. RJ45 interface: Leave it to later to extend the Ethernet interface.

It is currently invalid.

Figure 5 shows the lower ports:



Figure 5 lower ports picture

4. Antenna: VT-DTU500C antenna interface using 50Ω/SMA (female head), external antenna must be suitable for 4G working band antenna. ZLAN can provide glue stick or suction cup antenna, the suction cup can be sucked to the metal housing of the chassis (the default suction cup antenna lead is 1.5 meters long)
5. SIM Card installed: When installing the SIM card, ensure that the device is not powered on. Using a pen tip and a screwdriver, push the SIM card slot out and push the SIM metal face down into the slot.
6. DB9: RS232 signal input.

## 5. Configuration

The device can configure parameters through a serial port and can also be configured remotely by installing configuration software on a remote server after connecting to it.

### 5.1 Serial port AT instruction configuration

Connect the USB to RS232 line to the serial port of VT-DTU500C, power on VT-DTU500C, open Vircom (hereinafter referred to as configuration tool), and enter the main interface of the configuration tool as shown in Figure 6.

Click Device Management to select serial port search, as shown in Figure 7, and the serial port parameter selection interface appears, as shown in Figure 8. Select the serial port number, here is COM15, and the baud rate is 115200. 115200 is the factory default setting, and if the user previously set VT-DTU500C to other baud rate, it can also be searched.

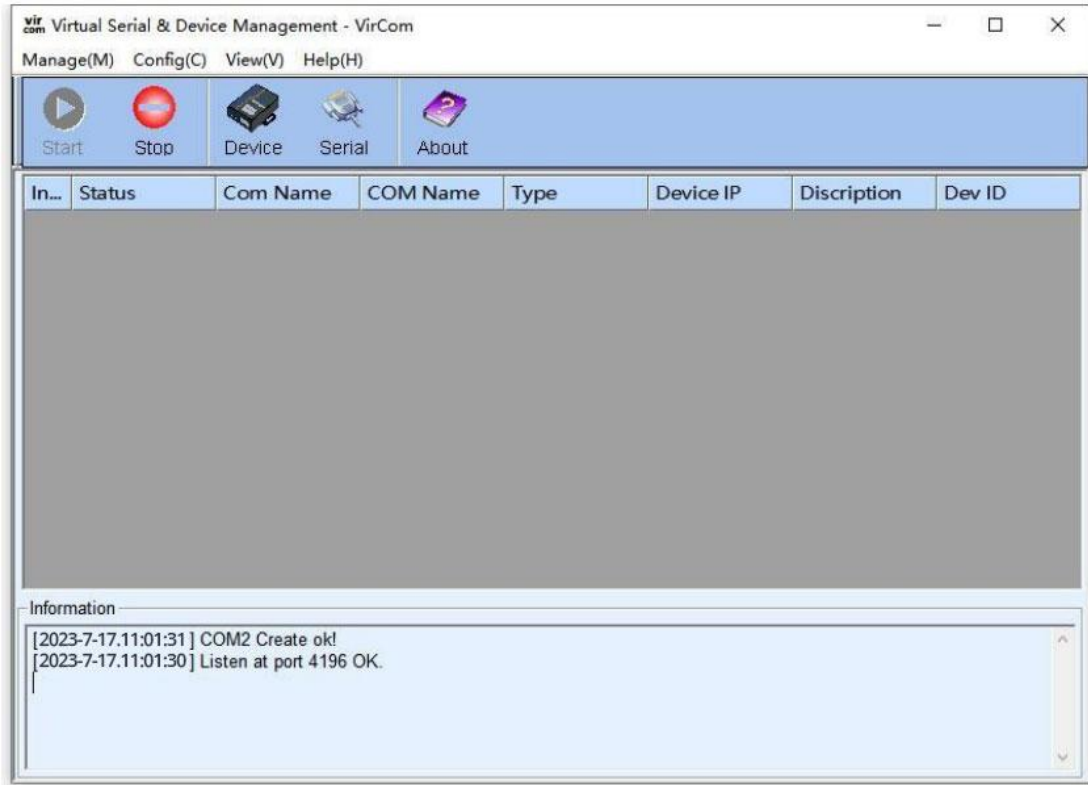


Figure 6 Vircom Main Interface

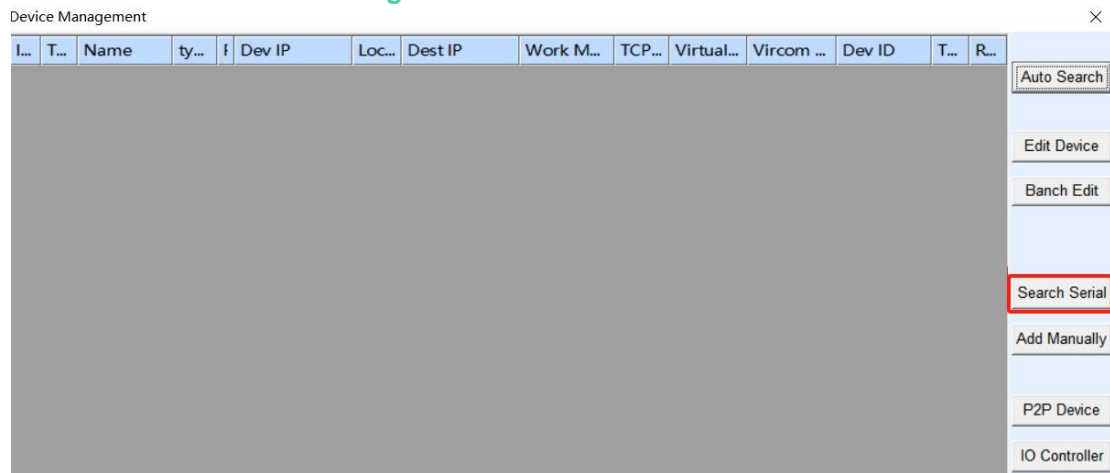


Figure 7 Serial port search interface

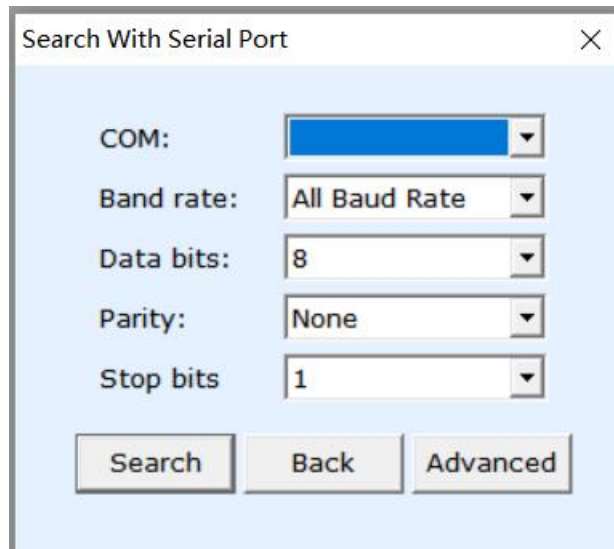


Figure 8 Serial port parameter Settings

After power on, wait for 15 seconds, that is, after the dial light starts flashing, select and click the "Search" serial port. At this time, the configuration tool will attempt to communicate with the device. If successful, it will enter the ConfTool interface. See Figure 9 below:

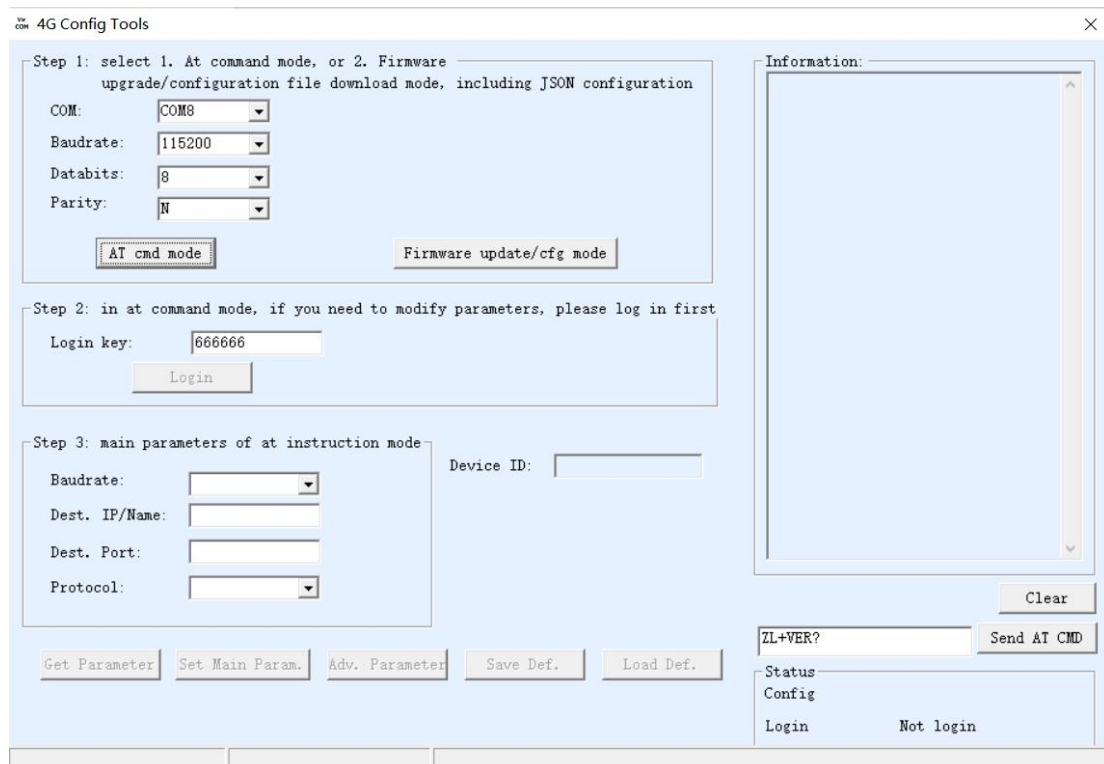
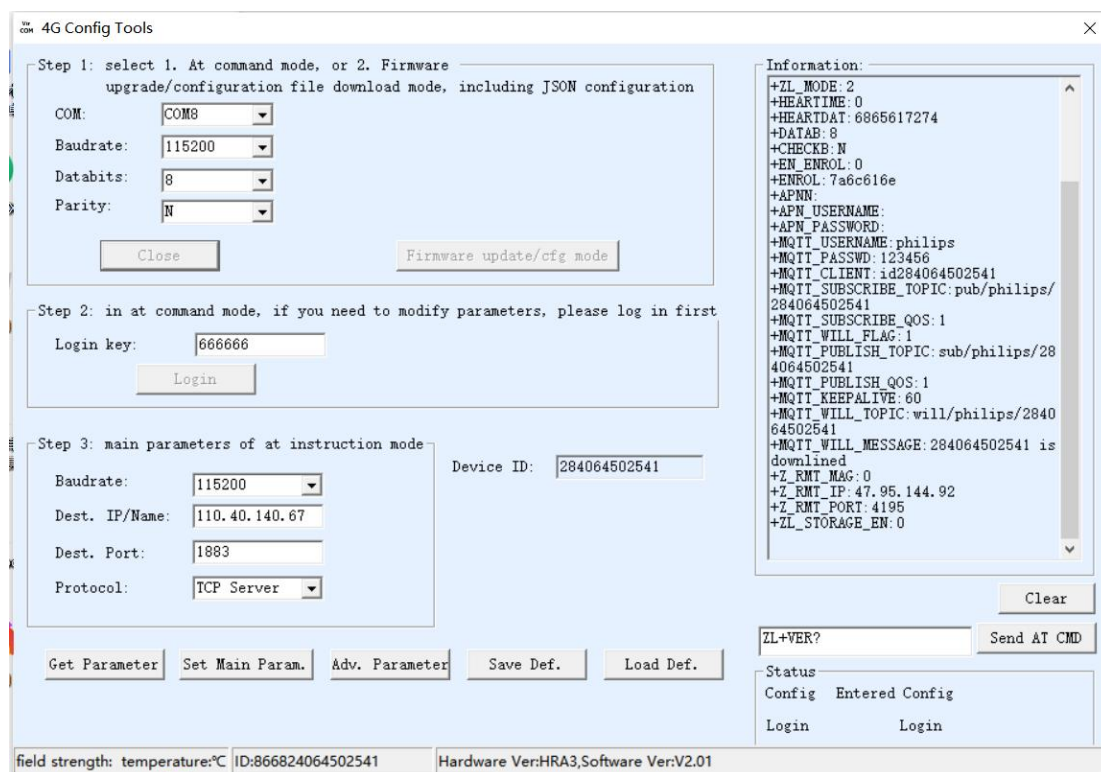


Figure 9 ConfTool interface

Click to enter the AT command mode, and the configuration tool will attempt to communicate with the device. After successful communication, the return information of AT command will be displayed on the right side, and the configuration mode will be shown as having entered the configuration mode, as shown in Chart 10 below:



**Figure 10 Display the interface for entering configuration mode**

The default login password is 666666. Before clicking "Login", all parameters are read-only and cannot be set or modified. Click the "Login Button"

It can be seen that after logging in, the LOGIN status changes to "Logged in", and the message "+LOGIN OK" appears on the right, as shown in Figure 11.

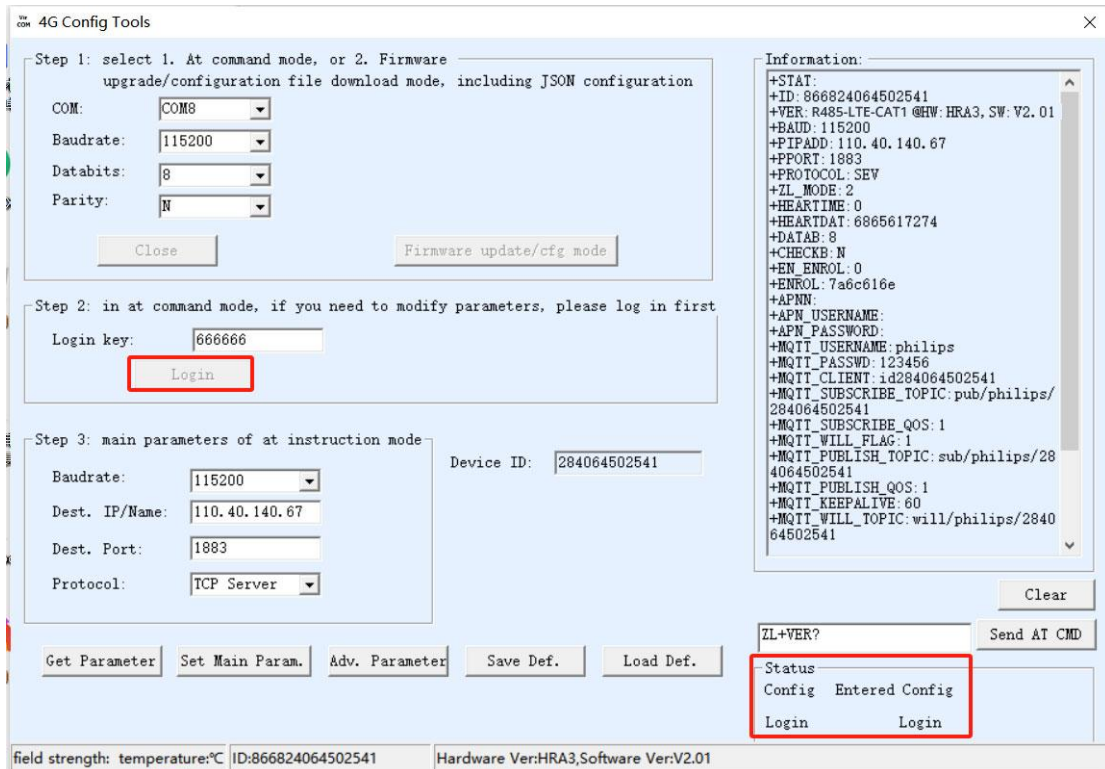


Figure 11 Login interface

The main parameters of the AT command mode include baud rate, destination IP, destination port and protocol. The protocol supports either TCP or UDP protocols. After modifying the corresponding parameters, click "Set Parameters" to set the new parameters to the device. At the same time, the device will return the successfully set parameters, as shown in Figure 12.

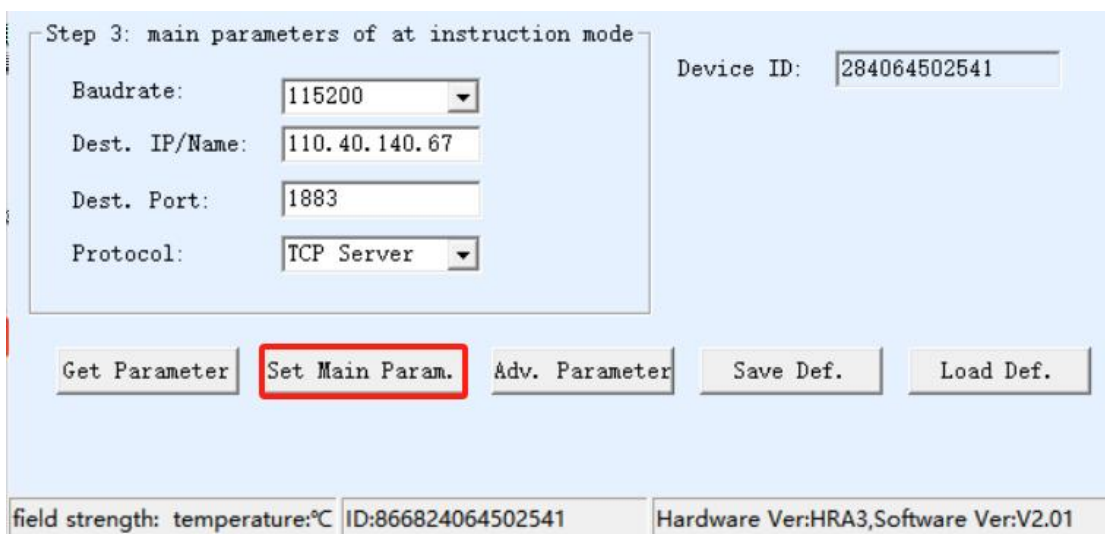


Figure 12 Setting parameters

The "Get Parameters" button can be used to obtain the parameters of the current device. Parameters are obtained by sending AT instructions. The data returned by the AT instructions is listed on the right. For the AT command, you can refer to other chapters of this article. Since "Get Parameters" will be automatically executed once after a successful "Open", there is generally no need to click the "Get Parameters" button.

Click "Advanced Parameters". The advanced parameters box is shown in Figure 13. The commonly used parameters are:

1. Heart rate interval: A heart rate packet with an interval of 15 seconds can be set.
2. Heartbeat content: Set the content of the heartbeat pack.
3. Serial port data bit
4. Serial port check bit
5. Enable the registration package: Whether to enable the registration package.
6. Registration package content: The content of the registration package sent after connecting to the server.
7. APN: The name of the access point of APN.
8. APN username
9. APN password
10. MQTT parameters: Used to set the parameters for accessing the MQTT server
11. Remote device management: It is used for devices with remote management functions to connect to remote servers

After selecting the parameters, click the "Take Effect Advanced Parameters" button, and observe the information bar on the right to check whether the setting information returned by the device is consistent with the information filled in, as shown in Figure 14.

Figure 13 Advanced Parameters

Figure 14 sets advanced parameters to return information

## 5.2 Firmware/Configuration file mode

After entering the ConfTool interface, click the firmware/configuration file mode button as shown in Figure 15 to jump to the firmware/configuration file interface as shown in Figure 16. First, create a local configuration web page root directory to store the configuration file. Click on the MQTT configuration to input the information for connecting to the MQTT server. After the Settings are completed, click Save the MQTT configuration as shown in Figure 17. Click on the JSON configuration to perform the JSON up-and-down configuration. Save the JSON configuration as shown in Figure 18. Click the download button, and the configuration software will download all the files in the directory to the device. After the download is successful, a transfer completion interface will pop up, and the device will automatically restart, as shown in Figure 19.

The screenshot displays the '4G Config Tools' application window. It is divided into several sections:

- Step 1:** 'select 1. At command mode, or 2. Firmware upgrade/configuration file download mode, including JSON configuration'. It contains dropdown menus for COM (COM8), Baudrate (115200), Databits (8), and Parity (N). Two buttons are present: 'AT cmd mode' and 'Firmware update/cfg mode', with the latter highlighted by a red box.
- Step 2:** 'in at command mode, if you need to modify parameters, please log in first'. It includes a 'Login key' input field with '066666' and a 'Login' button.
- Step 3:** 'main parameters of at instruction mode'. It features input fields for Baudrate, Dest. IP/Name, Dest. Port, and Protocol, along with a 'Device ID' field.
- Information:** A large vertical text area on the right side of the interface.
- Buttons:** A row of buttons at the bottom includes 'Get Parameter', 'Set Main Param.', 'Adv. Parameter', 'Save Def.', and 'Load Def.'.
- Status:** A section at the bottom right shows 'ZL+VER?' with a 'Send AT CMD' button and a 'Status' area with 'Config' and 'Login' indicators, where 'Login' is currently 'Not login'.

Figure 15 Configuration interface

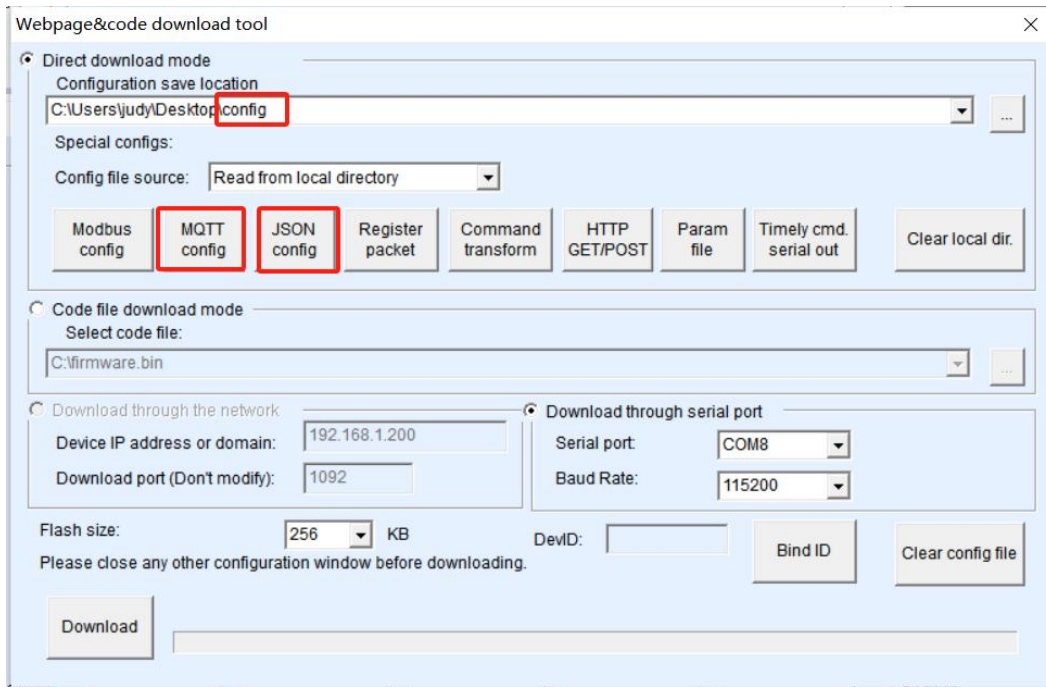


Figure 16 Firmware/configuration file interface

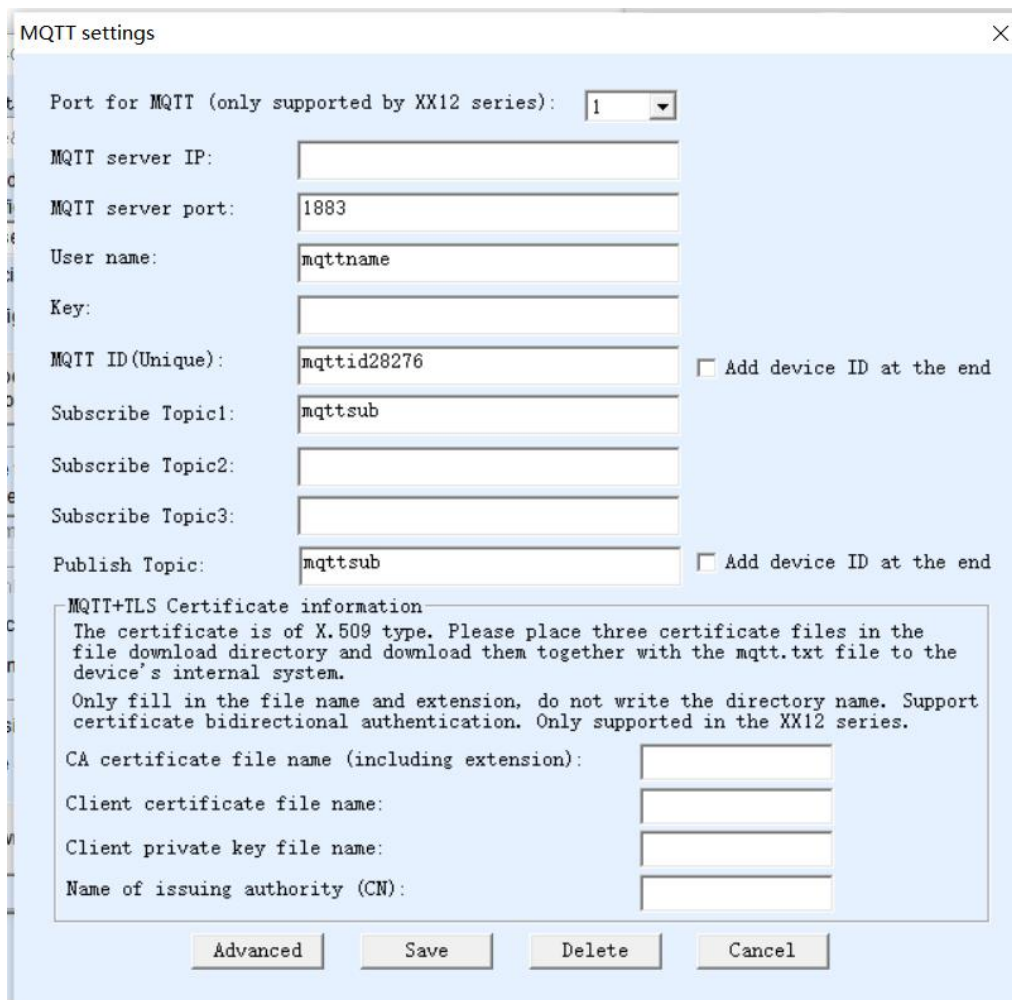


Figure 17 MQTT configuration interface

JSON To Modbus RTU Settings

Config and options

Select port (only supported by XX12 series):   Time sharing collection for each port

Time zone:   The keyword name is Unicode encoding

- Data transmit interval to server:  (ms, range: 100 - 31718940, max 8.8hours, 0 is no send)  
 Enable short link, when time come start link, then wait  ms for establish TCP connection  
 Then send data, then after 1s close connection.  Upload according to NTP time.
- Select the cloud platform to access:
- The Uplayer Protocol of JSON:   
 GET/POST URL(not include the ahead "http://")   
 When selecting the GET protocol, the [#JSON\_NAME] input in the URL will be replaced with the corresponding value of JSON\_NAME. Design JSON\_NAME by 'JSON upload' below.  
 The Variable Name of the POST(Like name={ }). No need for pure json):
- Add prefix to upload data(e.g. 01 02):  Data format:   
 Select ASCII data format, and the [#JSON\_NAME] appearing in the prefix will be replaced with the corresponding value of JSON\_NAME. JSON\_NAME designde by 'JSON upload' below.  
  
 Register packet (sent when connecting to server):
- After  times of upload, serial send data:  Condition(Def. empty):   
 Design timing send serial command table(support transparent transmission when NO JSON):
- Add or Remove Modbus Registers:
- Click to save JSON settings and display the results:
- Export/Import config file.

Figure 18 JSON configuration interface

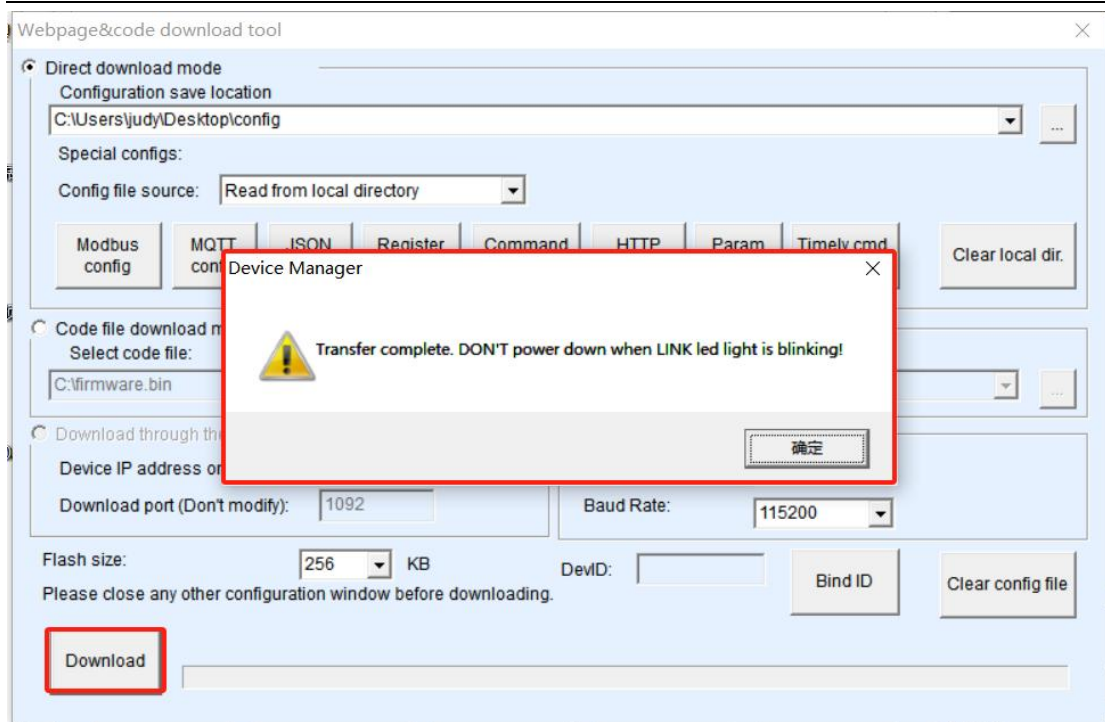


Figure 19 Download interface

## 6. Product Function

### 6.1 Communication test

#### 6.1.1 Server Transparent Transmission Test

Suppose there is the following networking structure as shown in the figure below, configured to connect to the \*\*\* port of the server \*\*\*.\*\*\*.\*\*\*.\*\*\*. Please configure it through the methods in the "Serial Port Configuration" section. After the configuration is completed and the power is restored, it takes 20 to 40 seconds to connect to the server.



Figure 20 Network connection structure diagram

We run the TCP tool SocketDlgTest on the server.

The screenshot shows the server-side tool interface for the VT-DTU500C. It features several configuration panels and a central log window.

- Communication settings:** Work mode is set to "TCP Server". Local port is 8888. Dest IP is 192.168.1.200. Dest port is 1001. Group IP is 230.90.76.1. An "Open" button is present.
- Receive settings:** Includes checkboxes for "Receive as Hex" and "Receive to file", and a "Clear window" button.
- Send settings:** Includes checkboxes for "Send as Hex (format 01 02)" and "Send every 100 ms", and a "Send receive mode" dropdown set to "What rec". A "Modify send-file" button is also present.
- Receive area:** Shows a "Receive" section with a "Receive buffer size" of 2000 Bytes. Below it is a "Send window" with "Send", "Stop", and "Clear Info." buttons.
- Log window:** Displays the following information:
 

```

11:36:18.837 :TCP listen at port 8888 OK!
11:36:15.448 :TCP client connected to www.p2p-zlan.com:4195!
11:36:15.417 :TCP client connecting www.p2p-zlan.com(47.95.144.92):4195.
11:36:15.324 :TCP client socket create OK!
11:36:12.290 :UDP socket closed!

```
- Bottom status bar:** Shows "Local IP: 192.168.1.118", "Advance" button, and "Count and checksum TXD: 0 0 RXD: 0 0" with a "Reset cnt" button.

Figure 21 Server-side tools

As shown in the figure, select the local port as 4196 (note that if you are running the Vircom tool, you need to change the port), and then click the "Open" button. When The device is connected to the server, it will display "The NO..." is accepted!" The information.

Now connect the device's serial port to a USB-to-485 serial port cable, and open the serial port debugging tool, as well as the correct COM port.

Now, when data is sent through the serial port, the server side will reply with the corresponding data. Similarly, when the device receives the message reply from the server and outputs it through the serial port, the serial port tool will receive the same data here. This demonstrates the two-way communication from the serial port to 4G network, as shown in Figure 22 below:

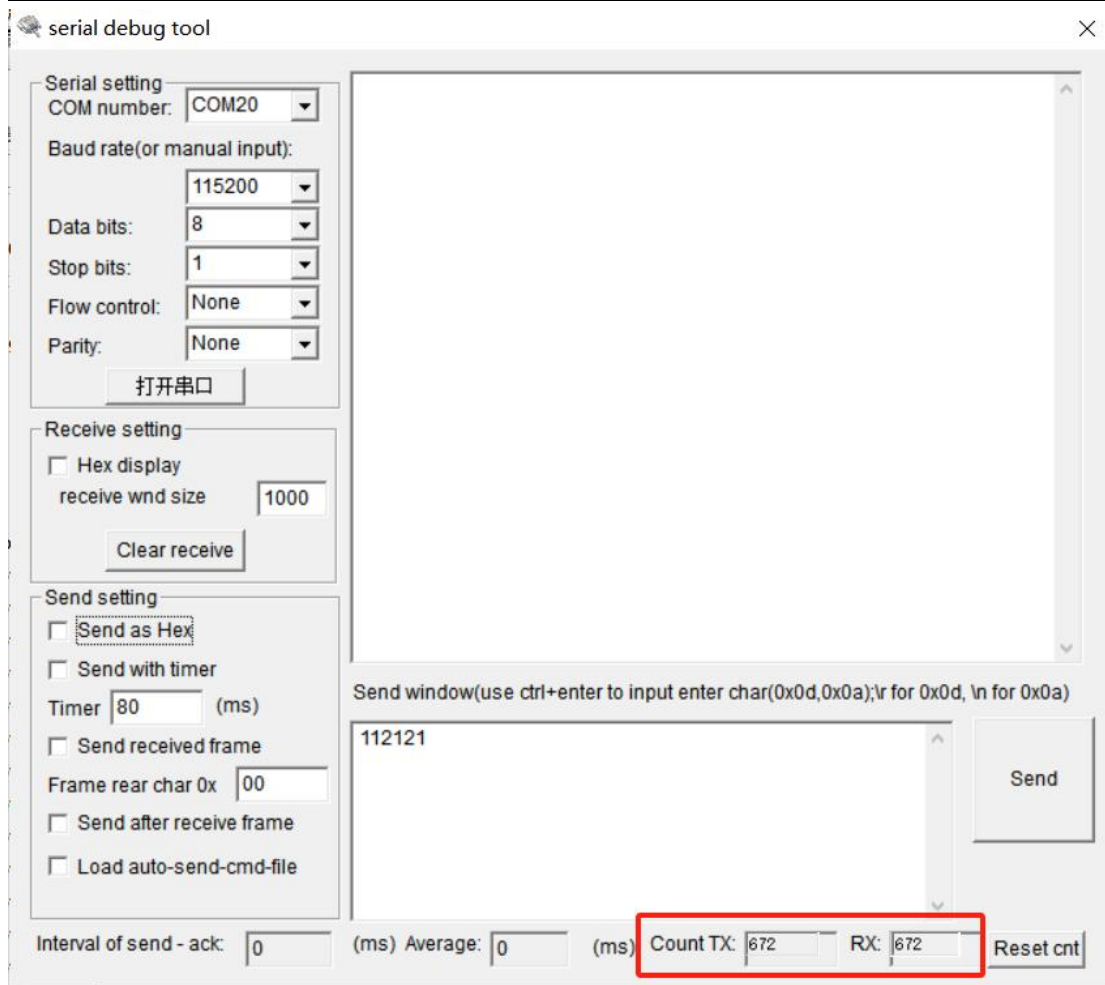


Figure 22 Serial port debugging tool for the device end

### 6.1.2 Modbus Protocol Conversion Test

The configuration parameters are basically the same as the no-protocol pass-through test; you only need to change the transformation protocol to the Modbus protocol. The serial port Modbus RTU protocol can be converted to the network Modbus TCP protocol, and the network Modbus TCP protocol can be converted to the serial port Modbus RTU protocol.

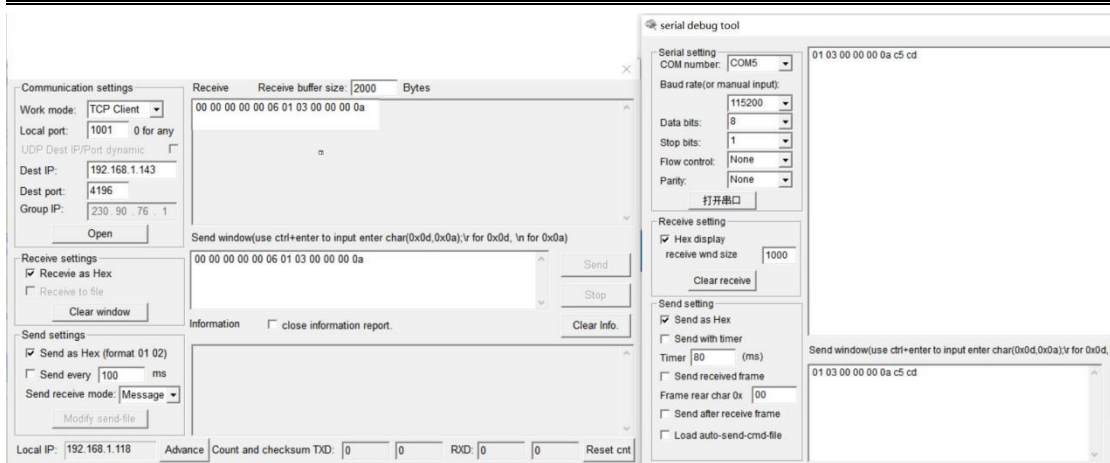


Figure 23 Modbus Protocol conversion test

### 6.1.3 MQTT Protocol Testing

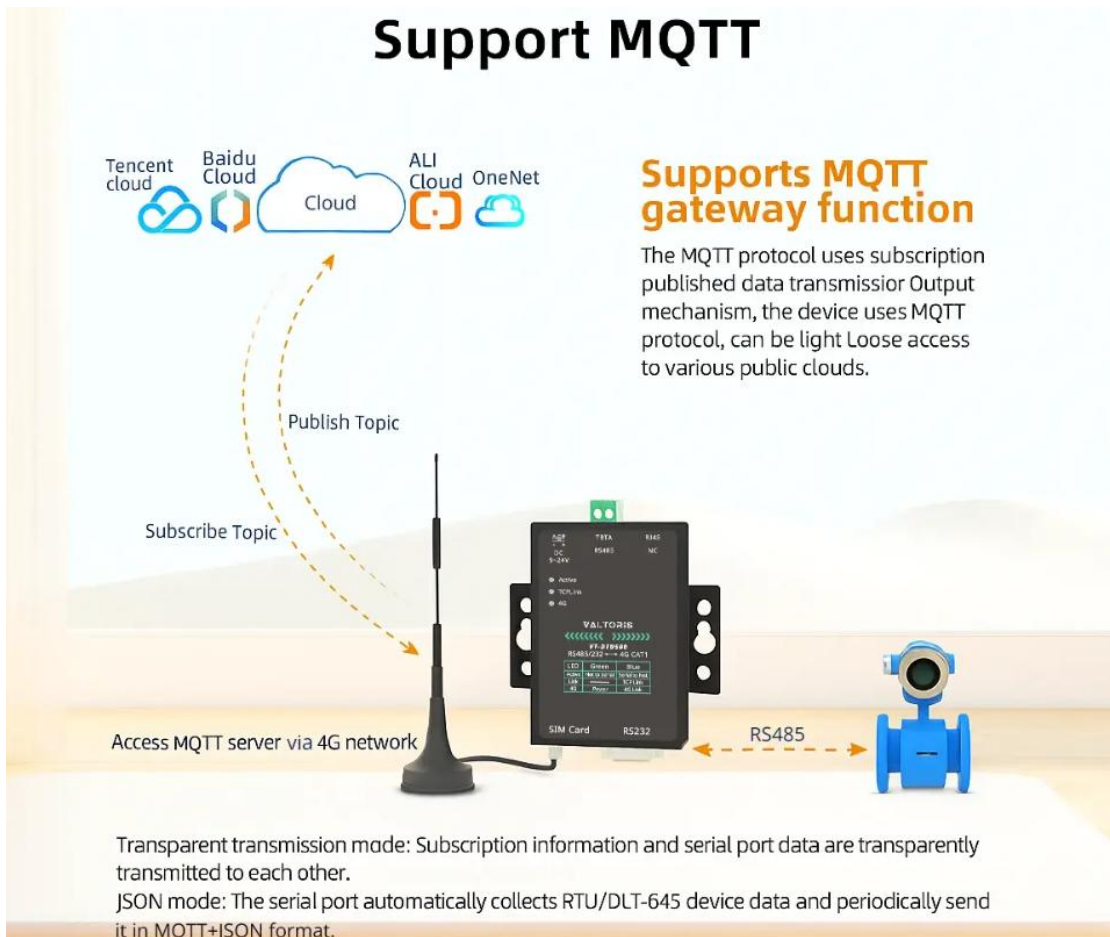


Figure 24 Schematic diagram of MQTT

This test is for connecting to Alibaba Cloud. Create a subscription topic named "test" and a publishing topic named "1" on Alibaba Cloud, as shown in Figure 25. According to the configuration instructions in step 5, first fill in the IP and port configuration of the MQTT server, save the parameters, and the parameter filling is

shown in Figure 26. Then, on the advanced parameter page, fill in the MQTT ID, username, password, as well as the subscription and publication topic and the retention time. The parameter filling is shown in Figure 27. Please note to select the working mode as MQTT mode.

Topic	Operation permission
/a1WSVHIXkDh/\${deviceName}/user/test	Subscribe -
/a1WSVHIXkDh/\${deviceName}/user/1	Publish -

Figure 25 Adds a theme to Alibaba Cloud

4G Config Tools

Step 1: select 1. At command mode, or 2. Firmware upgrade/configuration file download mode, including JSON configuration

COM: COM8  
 Baudrate: 115200  
 Databits: 8  
 Parity: N

Close Firmware update/cfg mode

Step 2: in at command mode, if you need to modify parameters, please log in first

Login key: 666666  
 Login

Step 3: main parameters of at instruction mode

Baudrate: 115200  
 Device ID: 284064502541  
 Dest. IP/Name: iot-as-mqtt.cn  
 Dest. Port: 1883  
 Protocol: TCP Client

Get Parameter Set Main Param. Adv. Parameter Save Def. Load Def.

Information:

```
+STAT:
+ID: 866824064502541
+VER: R485-LTE-CAT1 @HW: HRA3, SW: V2.01
+BAUD: 115200
+PIPAD:
+PPORT: 1883
+PROTOCOL: SEV
+ZL_MODE: 2
+HEARTIME: 0
+HEARTDAT: 6865617274
+DATAB: 8
+CHECKB: N
+EN_ENROL: 0
+ENROL: 7a6c616e
+APNN:
+APN_USERNAME:
+APN_PASSWORD:
+MQTT_USERNAME: mqttname
+MQTT_PASSWD:
+MQTT_CLIENT: mqttid28446
+MQTT_SUBSCRIBE_TOPIC: mqttsub
+MQTT_SUBSCRIBE_QOS: 1
+MQTT_WILL_FLAG: 0
+MQTT_PUBLISH_TOPIC: mqttsub
+MQTT_PUBLISH_QOS: 1
+MQTT_KEEPAIVE: 60
+MQTT_WILL_TOPIC:
+MQTT_WILL_MESSAGE:
+Z_RMT_MAG: 0
+Z_RMT_IP: 47.95.144.92
```

ZL+VER? Send AT CMD

Status  
 Config Entered Config  
 Login Login

field strength: temperature:°C ID:866824064502541 Hardware Ver:HRA3,Software Ver:V2.01

Figure 26 Alibaba Cloud IP and Port

Advanced Parameters

Work Parameters

Work Type: MQTT

DNS Server IP:

Heart Beat Interval: Disable

Heart Beat Content:  ASCII

Serial Data Bits: 8

Serial Parity: N

Stop Bits:

Login Key:

Enable Register Pkt: Disable

Register Pkt Content:  ASCII

APN:

APN UserName:

APN Key:

Enable P2P: Disable

No Data Restart: 1500 Min(0 disable)

Enable Off-line Storage

MQTT Parameters

MQTT version: V3.1.1

User Name: 112121@a1WSVHIXKDh

Key: 041BD699CB041300ADD336E96

Client ID: thod-hmacsha1,timestamp-

Subscribe Topic: XkDh/112121/user/test

Subscribe QOS: 1

Publish Topic: WsVHIXkDk112121/user/1

Publish QOS: 1

Keep Alive Time: 60

Enable Will: 0

Last-will Topic:

Last-will Message:

Remote Device Manage

Enable Remote Device Manage

Server IP/DNS:

Server TCP Port:

Set Cancel Get Default

Figure 27 Configuration of Alibaba Cloud MQTT

After the Settings are completed, open the Alibaba Cloud Device Management interface and enter the log service page to view the information sent on the device, as shown in Figure 28. Data is sent through the serial port of the device. A message (" MTEST ") is sent to the MQTT server of Alibaba Cloud via the subject of 1. Alibaba Cloud receives the data as shown in Figure 29. The Alibaba Cloud server then sends a message (" ALI\_send ") to the serial port of the device via the test subject as shown in Figure 30. This completes the MQTT sending and receiving test.

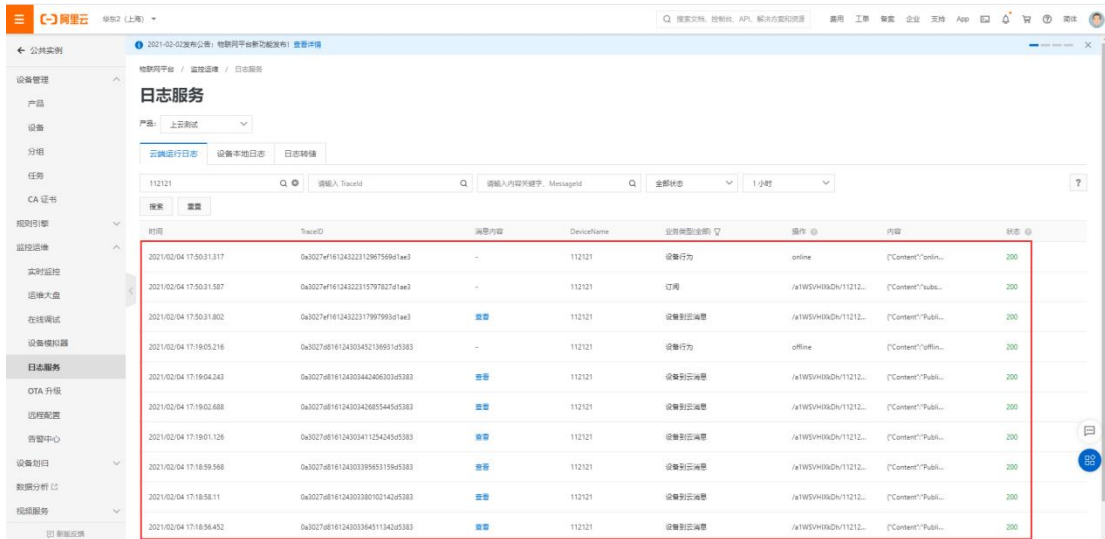


Figure 28 Alibaba Cloud Log Service

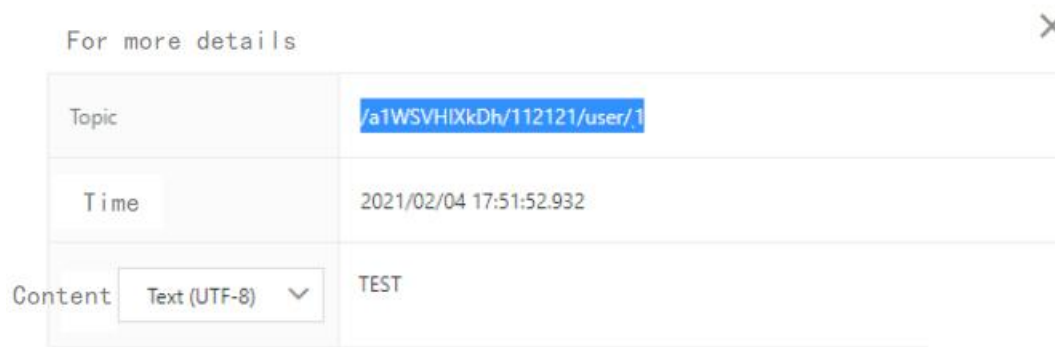


Figure 29: Alibaba Cloud receives serial port data

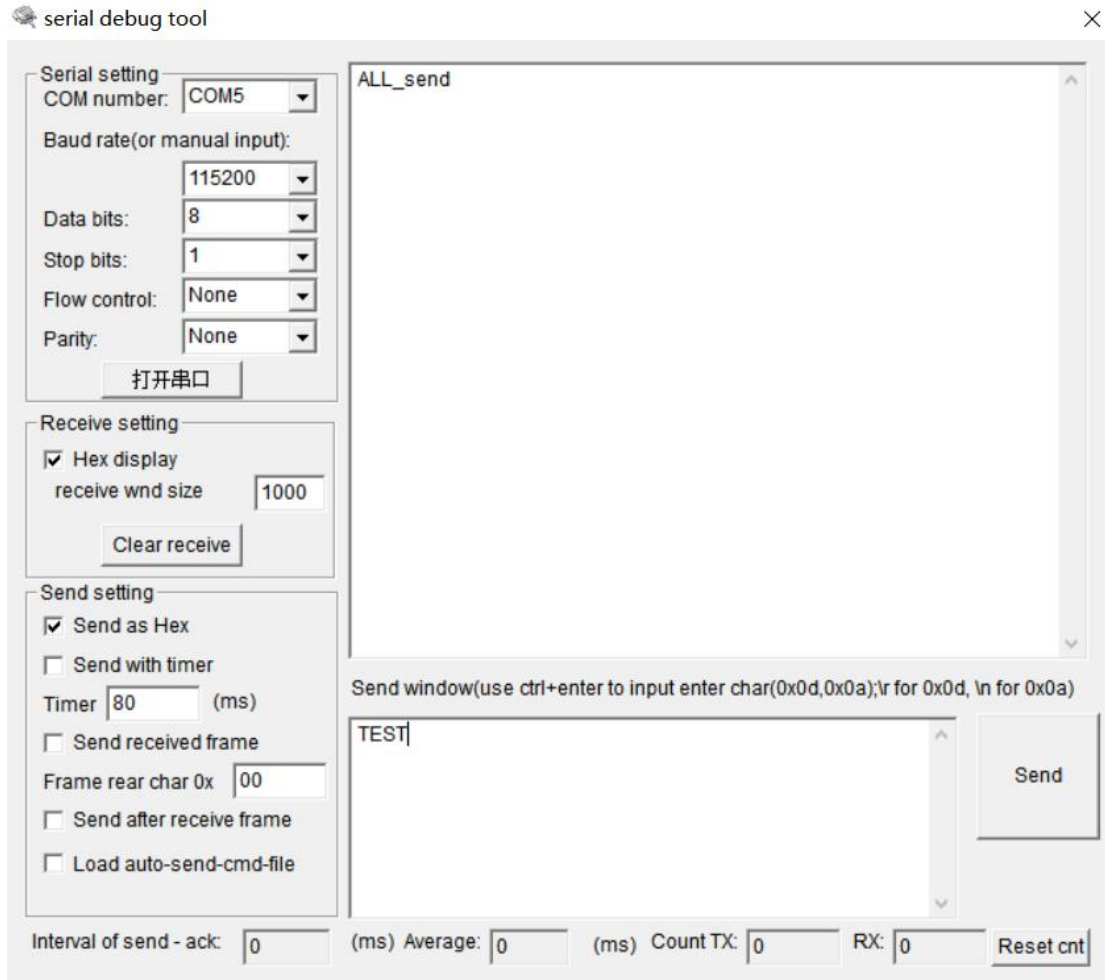
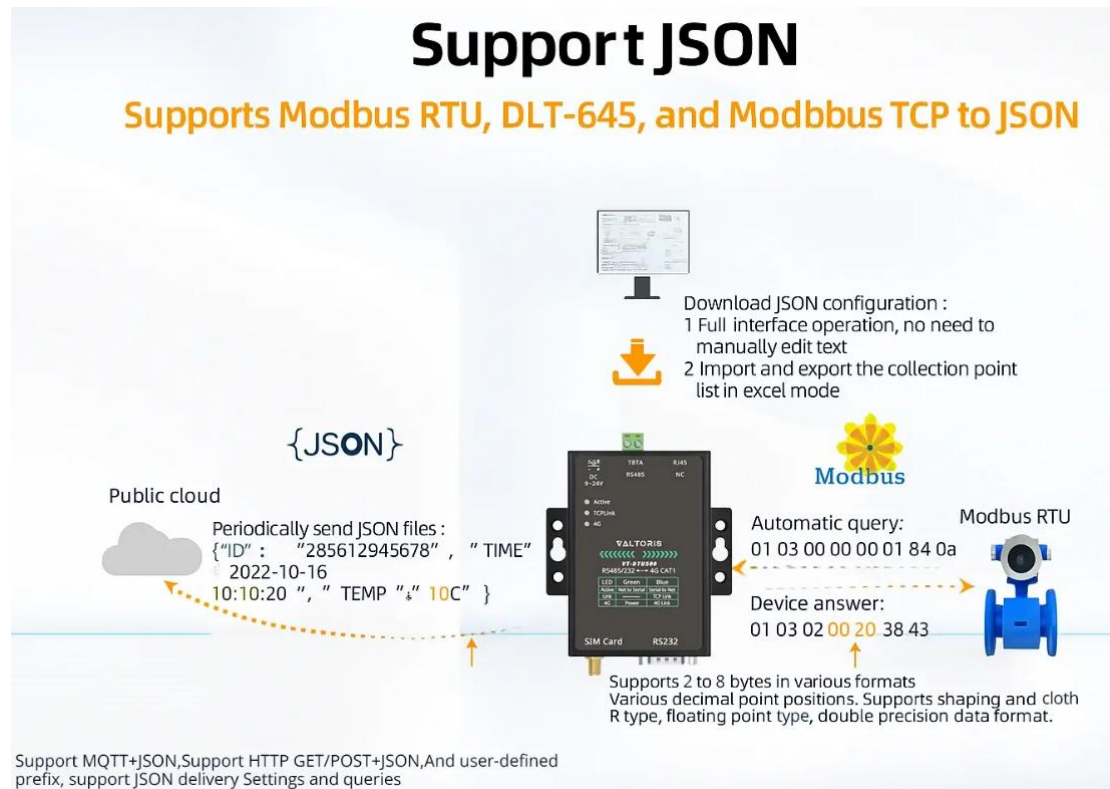


Figure 30 shows the serial port receiving data from Alibaba Cloud

**6.2 MODBUS RTU to JSON Conversion Test**



**Figure 31: Schematic diagram of JSON**

**6.2.1 Configure JSON for Posting**

Through the above section: Modbus protocol conversion test, a simple JSON upload template is configured. The configuration process is shown in Figures 32, 33, 34, and 35 below. Data from some MODBUS nodes is collected and converted into JSON format for upload.

JSON To Modbus RTU Settings

Config and options

Select port (only supported by XX12 series):   Time sharing collection for each port

Time zone:   The keyword name is Unicode encoding

- Data transmit interval to server:  (ms, range: 100 - 31718940, max 8.8hours, 0 is no send)
   
 Enable short link, when time come start link, then wait  ms for establish TCP connection
   
Then send data, then after is close connection.  Upload according to NTP time.
- Select the cloud platform to access:
- The Uplayer Protocol of JSON: 
  
GET/POST URL(not include the ahead "http://") 
  
When selecting the GET protocol, the [#JSON\_NAME] input in the URL will be replaced with the corresponding value of JSON\_NAME. Design JSON\_NAME by 'JSON upload' below.
   
The Variable Name of the POST(Like name={ }). No need for pure json:
- Add prefix to upload data(e.g. 01 02):  Data format: 
  
Select ASCII data format, and the [#JSON\_NAME] appearing in the prefix will be replaced with the corresponding value of JSON\_NAME. JSON\_NAME designde by 'JSON upload' below.
   
  
Register packet (sent when connecting to server):
- After  times of upload, serial send data:  Condition(Def. empty): 
  
Design timing send serial command table(support transparent transmission when NO JSON):
- Add or Remove Modbus Registers:
- Click to save JSON settings and display the results:
- Export/Import config file.

Figure 32 Configuration JSON upload

Add JSON Node

Following is the 1. th design of register. It has been added:

JSON node data type:  Object data(Default value, including this node and later ones with { }, need Input JSON keyword)  
 Array data(including data by [ ], without JSON keyword)

Corresponding JSON Keyword: 1 Data source: Modbus RTU Other Data source: Current Time Format: 2025-08-07 14:37:54 Fixed String:  No quotation

**Modbus RTU Settings**

- Slave Address: 1 - IP: 0 . 0 . 0 . 0  
 - Modbus Function Code: 3 - Port: 502  
 - Register Address: 1

**645/698 Protocol**

- 645/698 Version: 97 Version - Read FE numbers: 0  
 - Device ID(6B): 000000000001 - Write FE numbers: 0  
 - Data type: 9410 - 698 Data type: Total positiv  
 - Keep invalid 0  - 698 Client Addr(CA): 0

1. Data length: 2 Bytes. 4 Bytes order: Big Endian (AI) (big-endin 4 bytes: Data ABCD, low address store 2 bytes AB)  
 2. Decimal point places: 0 digit. After get as intenger left shift the decimal point.  
 3. Enable shift and scale:  ubtract integer: 0 then divid float: 1 Register is float   
 4. Data format: Unsigned int Bool value at postion bit: 1  
 5. Add unit name to rear:   
 6. Add quotation to data:   
 7. The Period between two RTU cmd: 200 (ms) minimum 10. 100ms for 9600bps, and 500ms for 2400bps.  
 If timeout wait more: 0 (ms), before send next command. Set 0 to disable this function.  
 8. Transmit data to server when data changes:   
 9. If RS485 device offline, set special value:  Special value type: Special vs , special value: 0 .Set data to 1 if online:   
 10. Enable overrun alarm:  , minimum normal value: 0 maximum normal value: 0

Embedded JSON Related

Design and View

Exit Design

Figure 33 Configure the keywords for collection, register addresses, and collection intervals

Add JSON Node

Following is the 21. th design of register. It has been added:

JSON node data type:  Object data(Default value, including this node and later ones with { }, need Input JSON keyword)  
 Array data(including data by [ ], without JSON keyword)

Corresponding JSON Keyword: Data source: Modbus RTU Other Data source: Current Time Format: 2025-08-07 14:38:59 Fixed String:  No quotation

**Modbus RTU Settings**

- Slave Address: 20 - IP: 0 . 0 . 0 . 0  
 - Modbus Function Code: 3 - Port: 502  
 - Register Address: 21

**645/698 Protocol**

- 645/698 Version: 97 Version - Read FE numbers: 0  
 - Device ID(6B): 000000000001 - Write FE numbers: 0  
 - Data type: 9410 - 698 Data type: Total positiv  
 - Keep invalid 0  - 698 Client Addr(CA): 0

1. Data length: 2 Bytes. 4 Bytes order: Big Endian (AI) (big-endin 4 bytes: Data ABCD, low address store 2 bytes AB)  
 2. Decimal point places: 0 digit. After get as intenger left shift the decimal point.  
 3. Enable shift and scale:  ubtract integer: 0 then divid float: 1 Register is float   
 4. Data format: Unsigned int Bool value at postion bit: 1  
 5. Add unit name to rear:   
 6. Add quotation to data:   
 7. The Period between two RTU cmd: 200 (ms) minimum 10. 100ms for 9600bps, and 500ms for 2400bps.  
 If timeout wait more: 0 (ms), before send next command. Set 0 to disable this function.  
 8. Transmit data to server when data changes:   
 9. If RS485 device offline, set special value:  Special value type: Special vs , special value: 0 .Set data to 1 if online:   
 10. Enable overrun alarm:  , minimum normal value: 0 maximum normal value: 0

Embedded JSON Related

Design and View

Exit Design

Figure 34 Save and exit after the configuration is completed

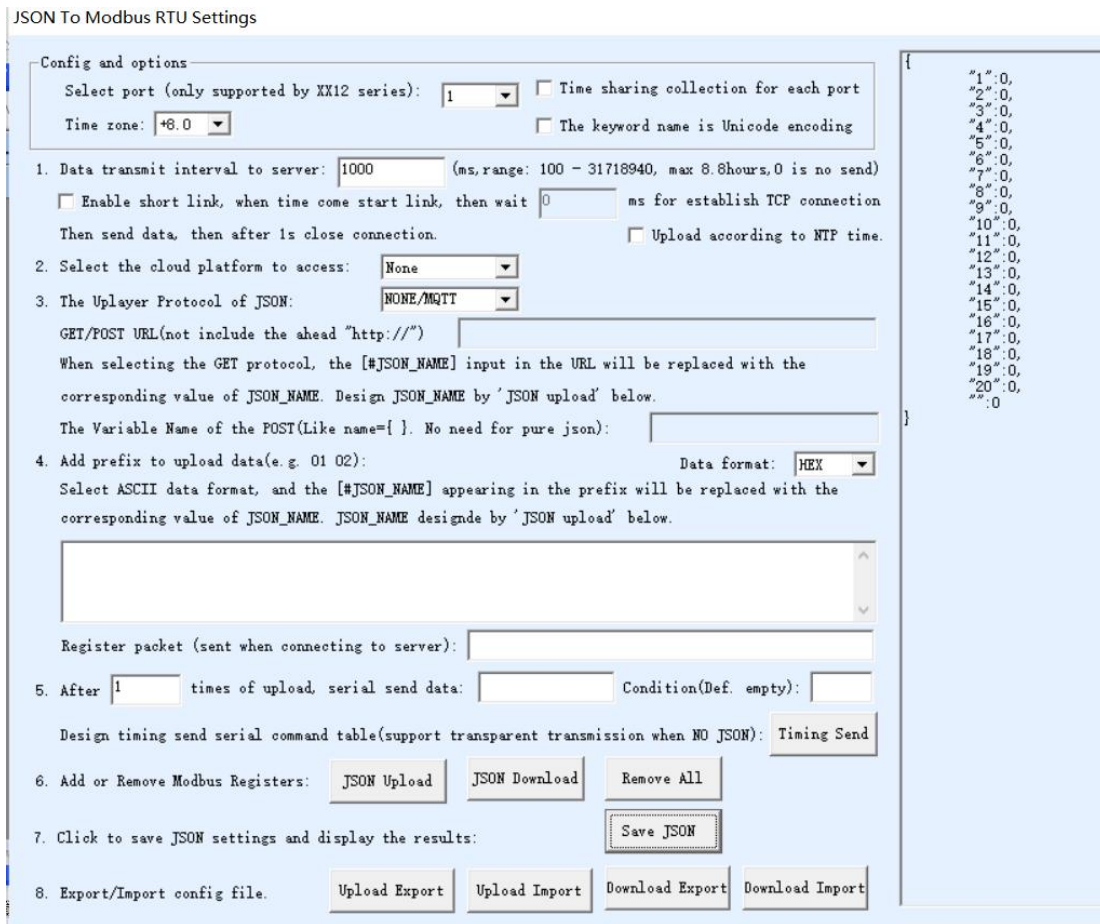


Figure 35 saves the JSON Settings and views the preview of the JSON format

## 6.2.2 Configure the MODBUS RTU simulation device

Simulate the MODEBUS Slave device through the Modbus Slave software, connect the device to the computer via the serial port cable, and open the connection of Modbus Slave. The configuration of Modbus Slave is shown in Figure 36.

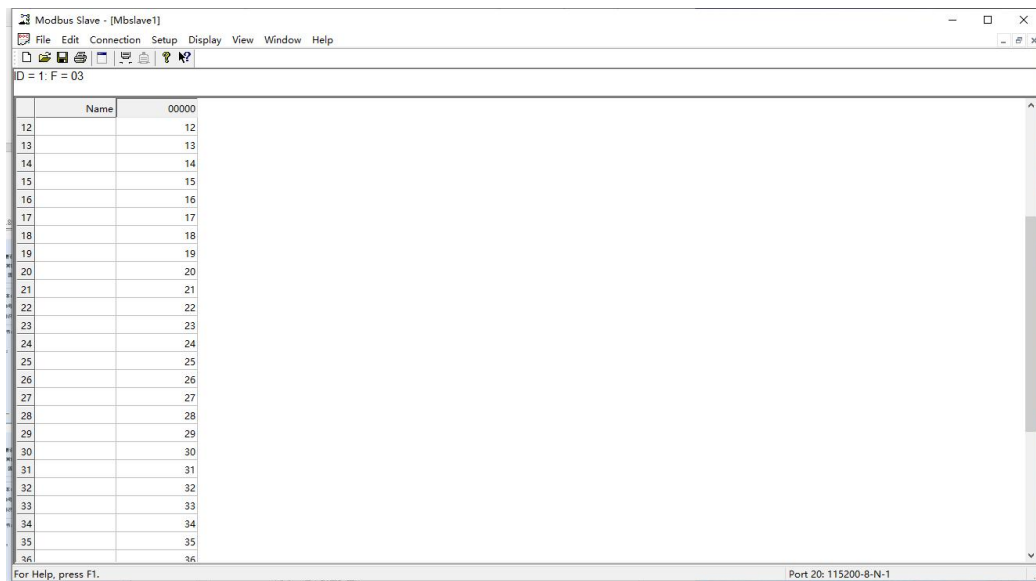


Figure 36: Modbus Slave fills in the simulation data

### 6.2.3 View the JSON posted above

By using the Alibaba Cloud log service to view the uploaded JSON data, it can be observed that the collected data is consistent with the data configured by Modbus Slave. This completes the simple MODBUS to JSON conversion test.

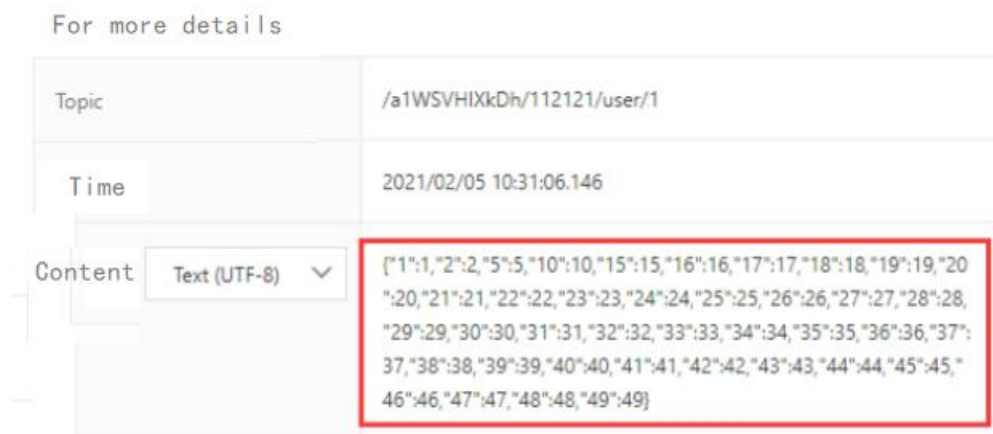


Figure 37 shows the serial port receiving data from Alibaba Cloud

## 7. AT Instruction

### 7.1 Login and configuration

#### 7.1.1 Enter configuration mode

Instruction: REQUEST CFG MODE

Function: Enter the configuration mode. Send this command during the startup phase of the device. After the device is started up, the device can enter the configuration mode.

Return: CFG MODE\r.

#### 7.1.2 Login

Instruction: ZL+LOGIN=666666\r\n

Function: Login. Modifying device parameters can only be successful when logged in.

Return: +LOGIN:OK\r\n(Success)or+LOGIN:NG\r\n(Failure)

#### 7.1.3 Obtain the device status

Instruction: ZL+STAT?\r\n

Function: Query the signal strength, temperature and voltage of the device

Return: +STAT: Intensity, temperature, voltage \r\n

## **7.2 Serial Port Parameters**

### **7.2.1. Obtain serial port parameters**

Instruction: ZL+BAUD?\r\n

Function: Obtain baud rate

Return: +BAUD:n\r\n, n Indicate the specific baud rate

### **7.2.2. Obtain the check bit**

Instruction: ZL+CHECKB?\r\n

Function: Obtain the check bit

Return: +CHECKB:N/O/E\r\n

N: No verification

O: Even parity

E: Odd parity

### **7.2.3. Obtain data bits**

Instruction: ZL+DATAB?\r\n

Function: Obtain data bits

Return: +DATAB:5/6/7/8\r\n

### **7.2.4. Set serial port parameters**

Instruction: ZL+BAUD=n\r\n

Function: Set the check bit

Return: +BAUD:n\r\n

### **7.2.5. Set the check bit**

Instruction: ZL+CHECKB= N/O/E \r\n

Function: Set the check bit

Return: +CHECKB:N/O/E\r\n

N: No verification

O:Even check

E:Odd parity

### **7.2.6. Set data bits**

Instruction: ZL+DATAB=5/6/7/8\r\n

Function: Set data bits

Return: +DATAB:5/6/7/8\r\n

## **7.3 Network Parameters**

### **7.3.1. Obtain the destination IP or domain name**

Instruction: ZL+PIPADD?\r\n

Function: Obtain the destination IP or domain name

Return: +PIPADD=ip\r\n

### **7.3.2. Obtain the destination port**

Instruction: ZL+PPORT?\r\n

Function: Obtain the destination port

Return: +PPORT=n\r\n

### **7.3.3. Obtain the working mode of the equipment**

Instruction: ZL+PROTOCOL?\r\n

Function: Obtain the working mode of the equipment

Return: +PROTOCOL=TCP/UDP\r\n

### **7.3.4. Obtain the IP address of the DNS server**

Instruction: ZL+PDNS?\r\n

Function: Obtain the IP address of the DNS server

Return: +PDNS=ip\r\n

### **7.3.5. Set the destination IP or domain name**

Instruction: ZL+PIPADD=ip\r\n

Function: Set the destination IP or domain name

Return: +PIPADD=ip\r\n

### **7.3.6. Set the destination port**

Instruction: ZL+PPORT=n\r\n

Function: Set the destination port

Return: +PPORT:n\r\n

### **7.3.7 Set the working mode**

Instruction: ZL+PROTOCOL=TCP/UDP \r\n

Function: Set the working mode

Return: +PROTOCOL=TCP/UDP\r\n

### **7.3.7. Set the IP of the DNS server**

Instruction: ZL+PDNS=ip\r\n

Function: Set the IP address of the DNS server

Return: +PDNS=ip\r\n

## **7.4 Registration package and heartbeat package**

### **7.4.1. Query the contents of the registration package**

Instruction: ZL+ENROL?\r\n

Query the content of the registration package (default registration package is in hexadecimal)

Return: +ENROL:1234567890\r\n

### **7.4.2. Whether the registration package is enabled**

Instruction: ZL+EN\_ENROL?\r\n

Query whether the registration package is enabled (1 enabled, 0 disabled)

Return: +EN\_ENROL:1\r\n

### **7.4.3. Set the contents of the registration package**

Instruction: ZL+ENROL=123456\r\n

Set the content of the registration package (the default registration package is in hexadecimal). The actual registration package is 0X12, 0X34, and 0X56

Return: +ENROL:123456\r\n

### **7.4.4. Enable/Disable Registration Package**

Instruction: ZL+EN\_ENROL=1\r\n

Enable/Disable registration package, where 1 indicates enable and 0 indicates disable

Return: +EN\_ENROL:1\r\n

## **7.5 Remote management function**

### **7.5.1 Query the remote management function**

Instruction: ZL+Z\_RMT\_MAG?\r\n

Check whether the remote management function is enabled. 1 indicates enabled and

0 indicates disabled

Return: + ZL+Z\_RMT\_MAG:1\r\n

### 7.5.2 Enable the remote management function

Instruction: ZL+Z\_RMT\_MAG=1\r\n

Enable/Disable remote management function: 1 indicates enabled, 0 indicates disabled

Return: + ZL+Z\_RMT\_MAG:1\r\n

### 7.5.3 Query the IP address of the remote management server

Instruction: ZL+Z\_RMT\_IP ?\r\n

Query the IP address of the remote management server \*\*\*\*\*

Return: + ZL+Z\_RMT\_IP =\*\*\*\*\*\r\n

### 7.5.4 Query the port of the remote management server

Instruction: ZL+ Z\_RMT\_PORT ?\r\n

Query the remote management server port \*\*\*\*

Return: + ZL+ Z\_RMT\_PORT =\*\*\*\*\r\n

### 7.5.5 Set the IP address of the remote management server

Instruction: ZL+Z\_RMT\_IP =\*\*\*\*\*\r\n

Set the IP address of the remote management server

Return: + ZL+Z\_RMT\_IP:\*\*\*\*\*\r\n

### 7.5.6 Set the port of the remote management server

Instruction: ZL+ Z\_RMT\_PORT =\*\*\*\*\r\n

Set the port of the remote management server

Return: + ZL+ Z\_RMT\_PORT:\*\*\*\* \r\n

## 7.6 MQTT Parameter

### 7.6.1 Set the MQTT username

Instruction: ZL+ MQTT\_USERNAME =\*\*\*\*\r\n

Set the MQTT username

Return: + ZL+ MQTT\_USERNAME:\*\*\*\* \r\n

### 7.6.2 Set the MQTT password

Instruction: ZL+ MQTT\_PASSWD =\*\*\*\*\r\n

Set the MQTT password

Return: + ZL+ MQTT\_PASSWD:\*\*\*\* \r\n

### 7.6.3 Set the MQTT client ID

Instruction: ZL+ MQTT\_CLIENT =\*\*\*\*\r\n

Set the MQTT client ID

Return: + ZL+ MQTT\_CLIENT:\*\*\*\* \r\n

#### **7.6.4 Set the MQTT publishing theme**

Instruction: ZL+ MQTT\_PUBLISH\_TOPIC =\*\*\*\*\r\n

Set the MQTT publishing theme

Return: + ZL+ MQTT\_PUBLISH\_TOPIC:\*\*\*\* \r\n

#### **7.6.5 Set up MQTT subscription topics**

Instruction: ZL+ MQTT\_SUBSCRIBE\_TOPIC =\*\*\*\*\r\n

Set up MQTT subscription topics

Return: + ZL+ MQTT\_SUBSCRIBE\_TOPIC:\*\*\*\* \r\n

#### **7.6.6 Set the quality of MQTT subscriptions**

Instruction: ZL+ MQTT\_SUBSCRIBE\_QOS =1\0\r\n

Set the quality of MQTT subscriptions 1\0

Return: + ZL+ MQTT\_SUBSCRIBE\_QOS: 1\0\r\n

#### **7.6.7 Set the quality of MQTT publishing**

Instruction: ZL+ MQTT\_PUBLISH\_QOS =1\0\r\n

Set the MQTT publication quality to 1/0

Return: + ZL+ MQTT\_PUBLISH\_QOS: 1\0\r\n

#### **7.6.8 Set the MQTT live retention time**

Instruction: ZL+MQTT\_KEEPALIVE =\*\*\*\*\r\n

Set the MQTT live retention time

Return: + ZL+ MQTT\_KEEPALIVE:\*\*\*\* \r\n

#### **7.6.9 Set the MQTT last wish topic**

Instruction: ZL+ MQTT\_WILL\_TOPIC =\*\*\*\*\r\n

Set the MQTT last wish topic

Return: + ZL+ MQTT\_WILL\_TOPIC:\*\*\*\* \r\n

#### **7.6.10 Set the MQTT last wish message**

Instruction: ZL+ MQTT\_WILL\_MESSAGE =\*\*\*\*\r\n

Set the MQTT last wish message

Return: + ZL+ MQTT\_WILL\_MESSAGE:\*\*\*\* \r\n

---

---

## 8. After-service and Technical Support

Phone/WhatsApp: +86 18321985506

Web: [www.valtoris.com](http://www.valtoris.com)

Email: [support@valtoris.com](mailto:support@valtoris.com)