Modbus RTU Analog Input 8CH

From Waveshare Wiki
Jump to: navigation, search

Overview

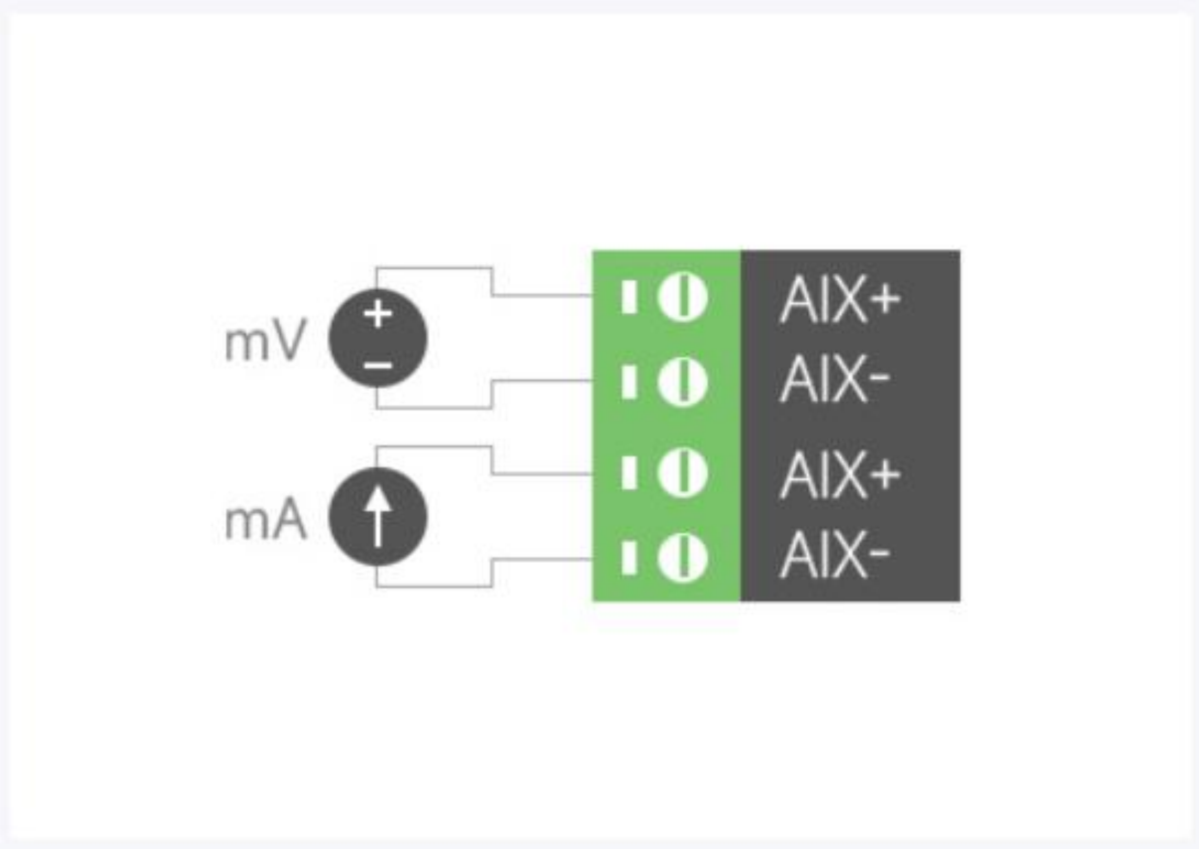
Modbus RTU Analog Input 8CH

(https://www.waveshare.com/modbus-rtu-analog-input-8ch.htm)

RS485

Hardware Description

- Each channel can be individually configured for its range, making it more convenient for users.
  "AIN+" is the positive input, and "AIN-" is the negative input. The module supports both differential and single-ended input. When used as a single-ended input, "AIN-" is connected to the ground.



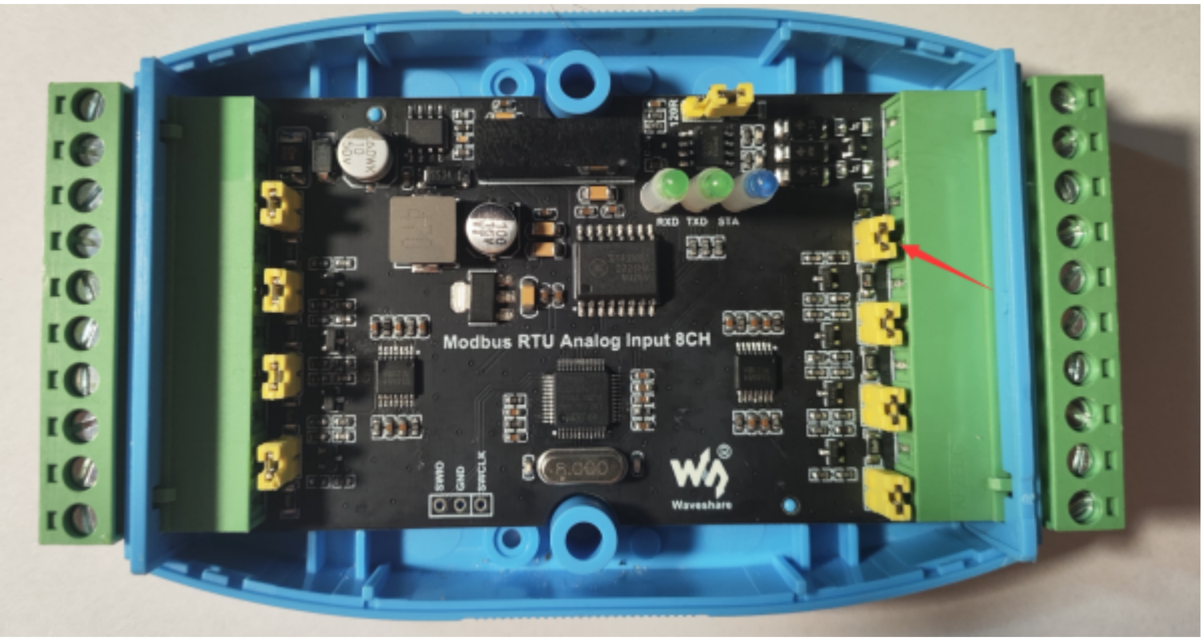
Supports voltage and current simultaneous acquisition

(/wiki/

File:Modbus-RTU-AI-8CH-06-1.jpg)

Note: When inputting the different powers, it is important to connect the ground wire to establish a common ground. Otherwise, the collected data may be inaccurate.

- Opening the device case, you can see jumpers are near the device terminals, corresponding to the eight channels AI1~AI8. You need to select the jumper mode based on the measurement signal; otherwise, the measurement data will be inaccurate.
    - When measuring voltage signals, the jumper wires for the corresponding channels should be disconnected.
    - When measuring current signals, the jumper wires for the corresponding channels should be connected.



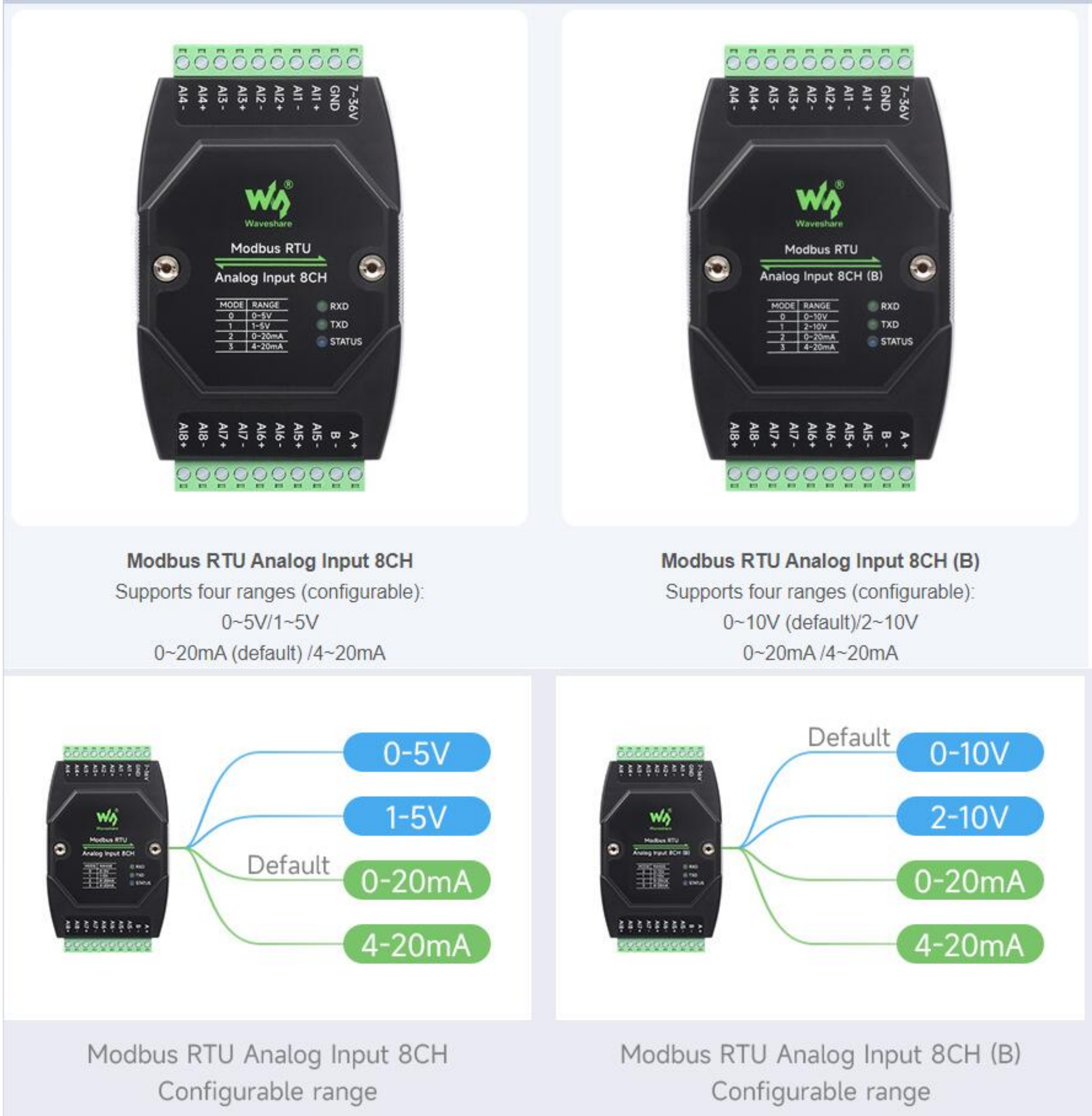
(/wiki/

File:Modbus_RTU_Analog_Input_8CH-Overview.png)

- Modbus RTU Analog Input 8CH defaults to current mode with the jumper wire connected. Modbus RTU Analog Input 8CH (B) defaults to voltage mode with the jumper wires disconnected.

Version Comparision

- Currently, there are two versions of the analog input series, one defaults to current input, and the other defaults to voltage input.



Modbus RTU Analog Input 8CH
Supports four ranges (configurable):
0~5V/1~5V
0~20mA (default) /4~20mA

Modbus RTU Analog Input 8CH (B)
Supports four ranges (configurable):
0~10V (default)/2~10V
0~20mA /4~20mA

Modbus RTU Analog Input 8CH
Configurable range

Modbus RTU Analog Input 8CH (B)
Configurable range

(/wiki/File:Modbus-RTU-Analog-Input-AB-Compare-01.jpg)

Version	Modbus RTU Analog Input 8CH	Modbus RTU Analog Input 8CH (B)
Default mode	8-ch current mode, 0~20mA	8-ch voltage mode, 0~10V
Measurement range	0~5V/1~5V 0~20mA/4~20mA	0~10V/2~10V 0~20mA/4~20mA
Resolution	12-bit	12-bit
Current sampling resistance	249Ω	499Ω
Operational amplifier ratio	32.4/49.9	10/32.4
Channel	8-AI	8-AI

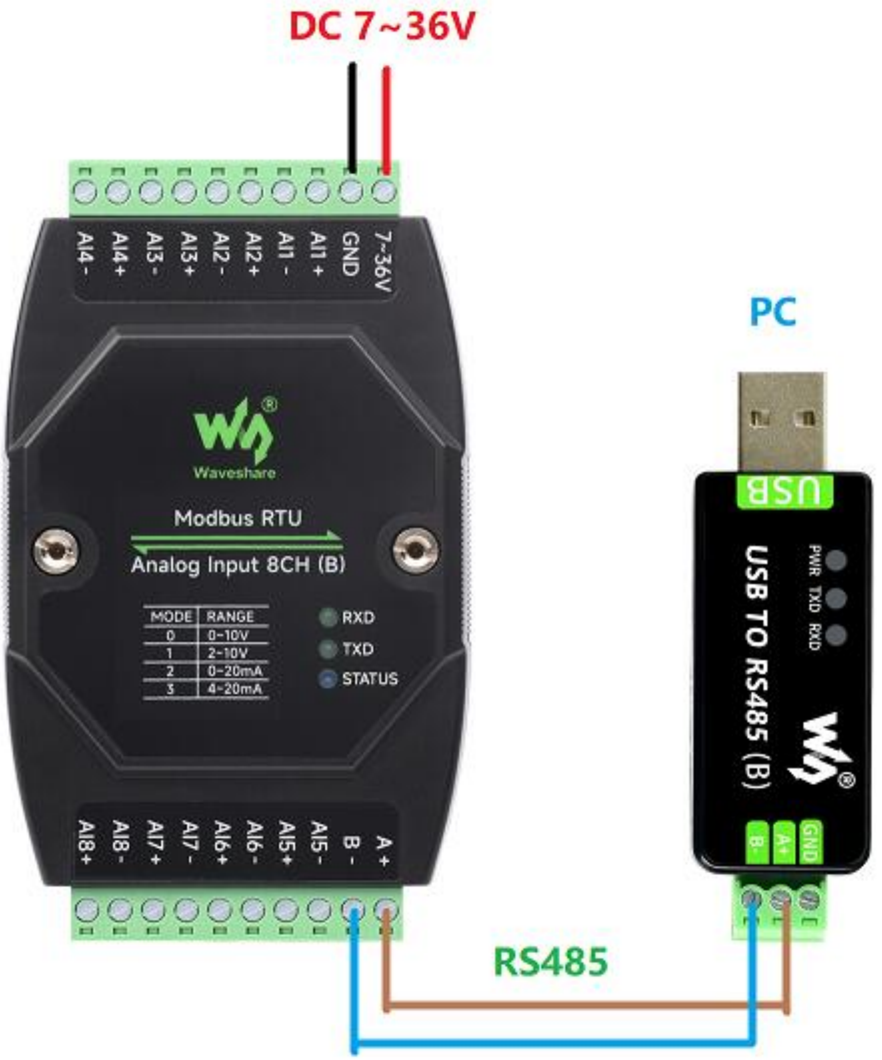
- Each version has five range modes from 0 to 4.

Mode	Modbus RTU Analog Input 8CH	Modbus RTU Analog Input 8CH (B)
0	0~5V voltage mode	0~10V voltage mode
1	1~5V voltage mode	2~10V voltage mode
2	0~20mA current mode	0~20mA current mode
3	4~20mA current mode	4~20mA current mode
4	4096-scale code mode	4096-scale code mode

- The scale code is the data collected by the AD converter and needs to undergo a linear transformation to obtain voltage or current data. The conversion formula is as follows.
  - Voltage = Scale Code * 3300/4095/Operational Amplifier Ratio
  - Current = Voltage/Sampling Resistor

Hardware Connection

- Connect the USB TO 485 to the target boards via cables, A-A and B-B connected as shown below:



(/

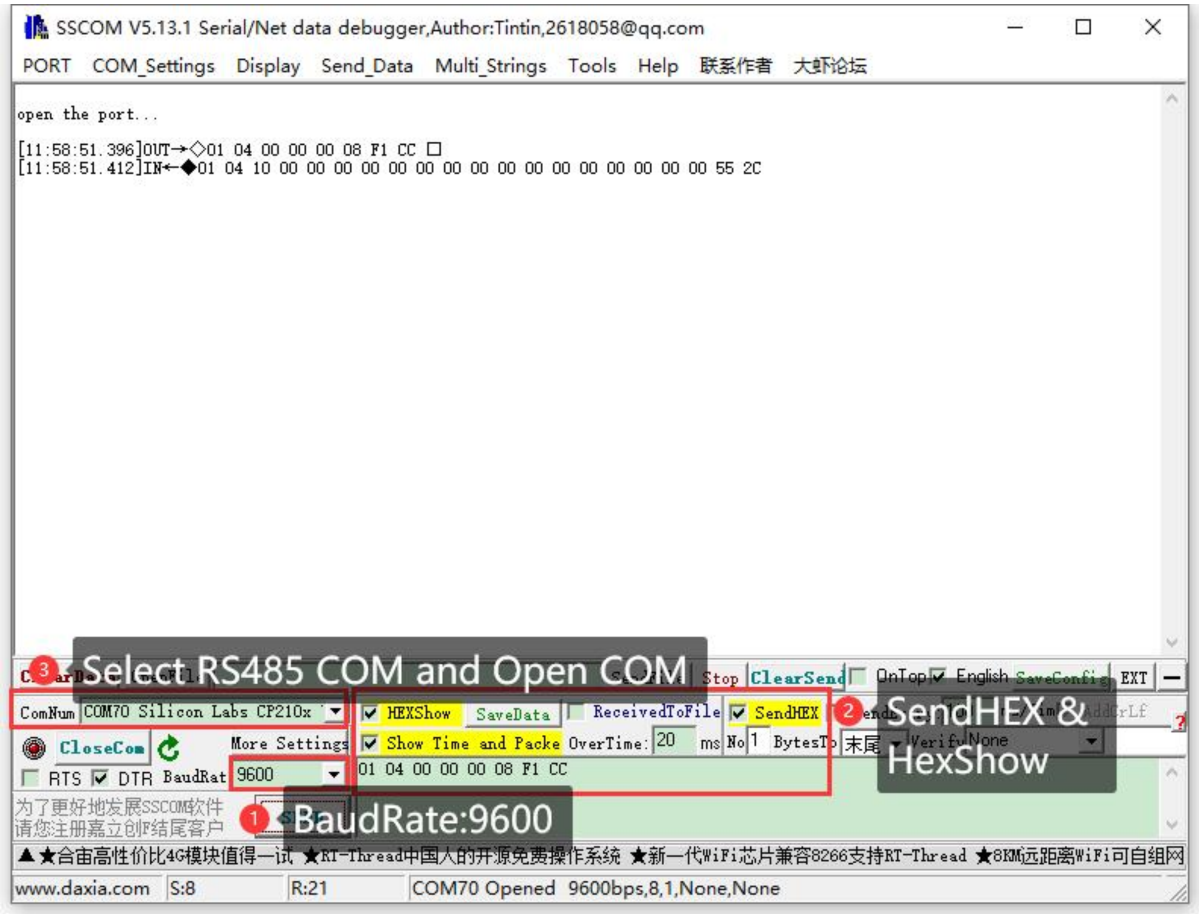
wiki/File:Modbus_RTU_Analog_Input_8CH-03.jpg)

Software Test

SSCOM Serial Port Debugging Assistant

- Download SSCOM serial port debugging assistant (https://files.waveshare.com/wiki/common/Sscom5.13.1.zip) and open it on the computer. Open the corresponding port number, set the baud rate as 9600, and select hex to send and receive.
  Send the following command, and it will return the 8-channel analog input data normally.

```
01 04 00 00 00 08 F1 CC
```



```
open the port...

[11:58:51.396]OUT→◇01 04 00 00 00 08 F1 CC □
[11:58:51.412]IN←◆01 04 10 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 55 2C
```

ComNum COM70 Silicon Labs CP210x

☑ HEXShow

☑ Show Time and Packe OverTime: 20 ms No 1 BytesTo 末尾 Verify None

☑ SendHEX

□ RTS ☑ DTR BaudRat 9600

```
01 04 00 00 00 08 F1 CC
```

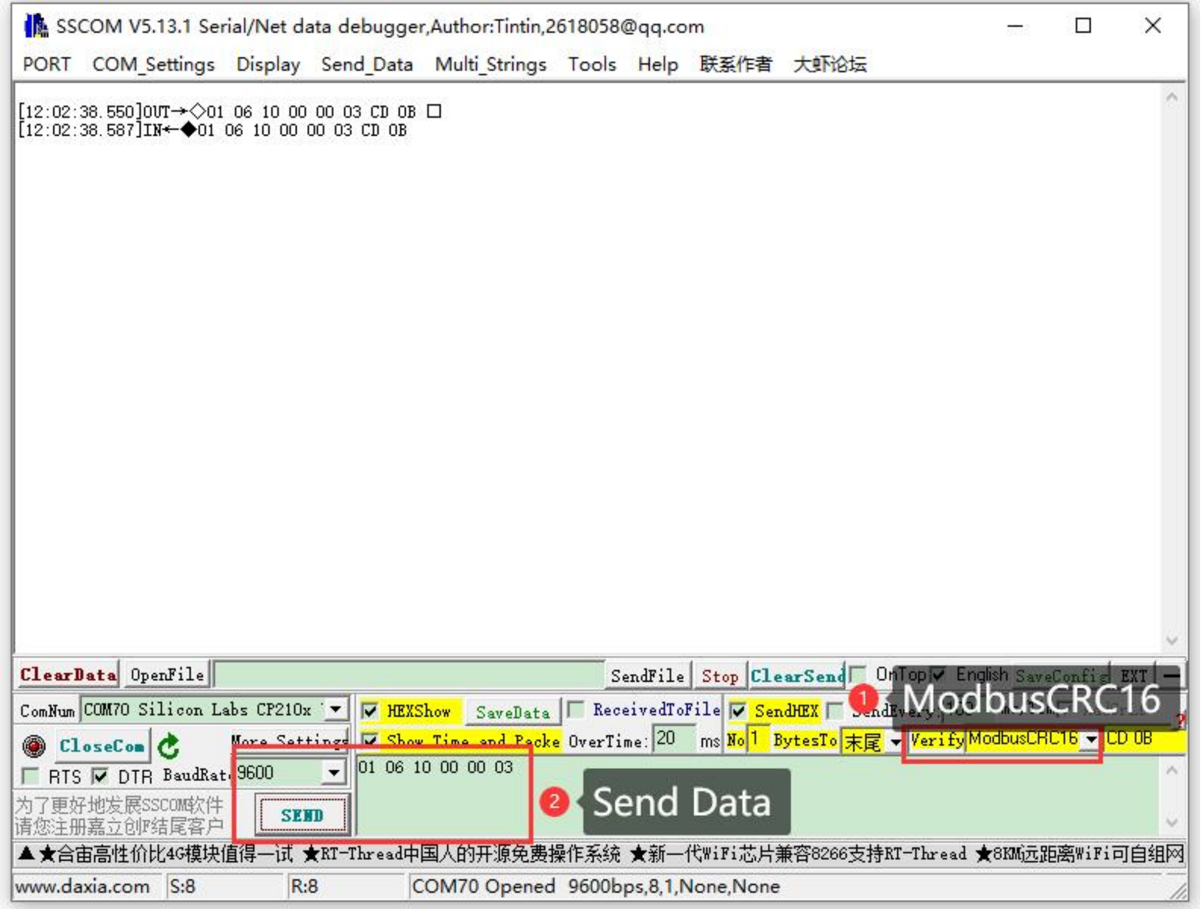
www.daxia.com S:8 R:21 COM70 Opened 9600bps,8,1,None,None

(/wiki/

File:Modbus-RTU-Analog-Input-SSCOM-test-01.jpg)

- If you need to send other commands, choose SendHEX. For checksum validation, select ModbusCRC16. After entering the first six bytes of the command, clicking SEND will automatically add the CRC check code. For example, send the following command, you can set channel 1 to 4-20mA current input mode.

```
01 06 10 00 00 03
```



```
[12:02:38.550]OUT→◇01 06 10 00 00 03 CD 0B □
[12:02:38.587]IN←◆01 06 10 00 00 03 CD 0B
```

ComNum COM70 Silicon Labs CP210x

BaudRate 9600

☑ HEXShow ☑ SendHEX

OverTime: 20 ms

Verify ModbusCRC16 CD 0B

01 06 10 00 00 03

www.daxia.com S:8 R:8 COM70 Opened 9600bps,8,1,None,None

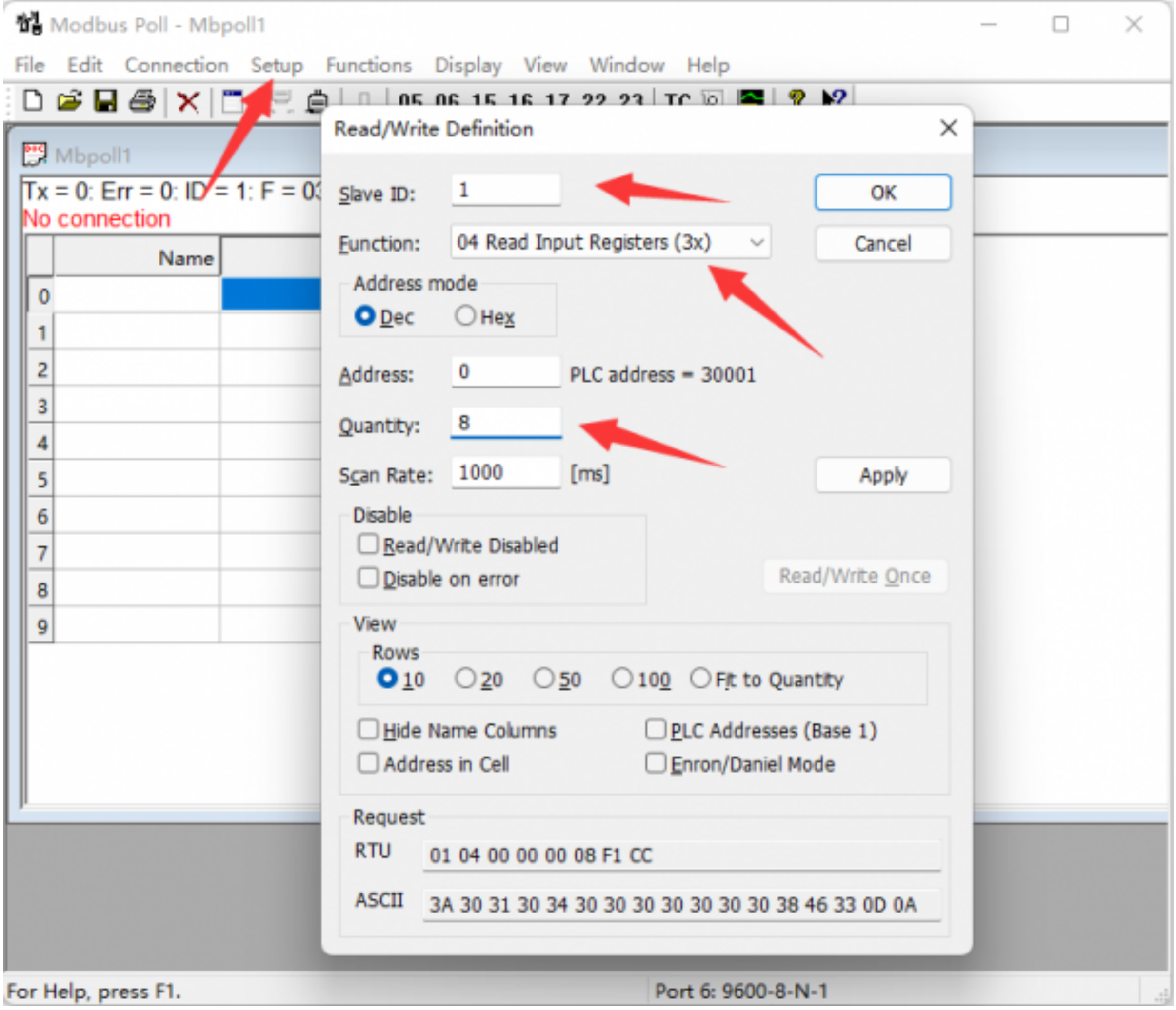
(/wiki/

File:Modbus-RTU-Analog-Input-SSCOM-test-02.jpg)

- For more control commands, you can refer to the development protocol.

Modbus Poll Software

- It is not convenient to use the SSCOM software for observing the data, you can select Modbus Poll software (https://www.modbustools.com/download.html) to read the data. Download and install the Modbus Poll software.
- Open the software, select Setup -> Read/Write Definition. Select the actual device address for Slave ID, 04 Read Input Registers (3x) for Function, and 8 channels for Quantity, and click "OK" to confirm.



Modbus Poll - Mbpoll1
File Edit Connection Setup Functions Display View Window Help
05 06 15 16 17 22 23 TC
Mbpoll1

Tx = 0: Err = 0: ID = 1: F = 03

No connection

	Name
0	
1	
2	
3	
4	
5	
6	
7	
8	
9	

Read/Write Definition

Slave ID: 1

Function: 04 Read Input Registers (3x)

Address mode: ● Dec ○ Hex

Address: 0 PLC address = 30001

Quantity: 8

Scan Rate: 1000 [ms]

Disable

□ Read/Write Disabled

□ Disable on error

View

Rows: ● 10 ○ 20 ○ 50 ○ 100 ○ Fit to Quantity

□ Hide Name Columns □ PLC Addresses (Base 1)

□ Address in Cell □ Enron/Daniel Mode

Request

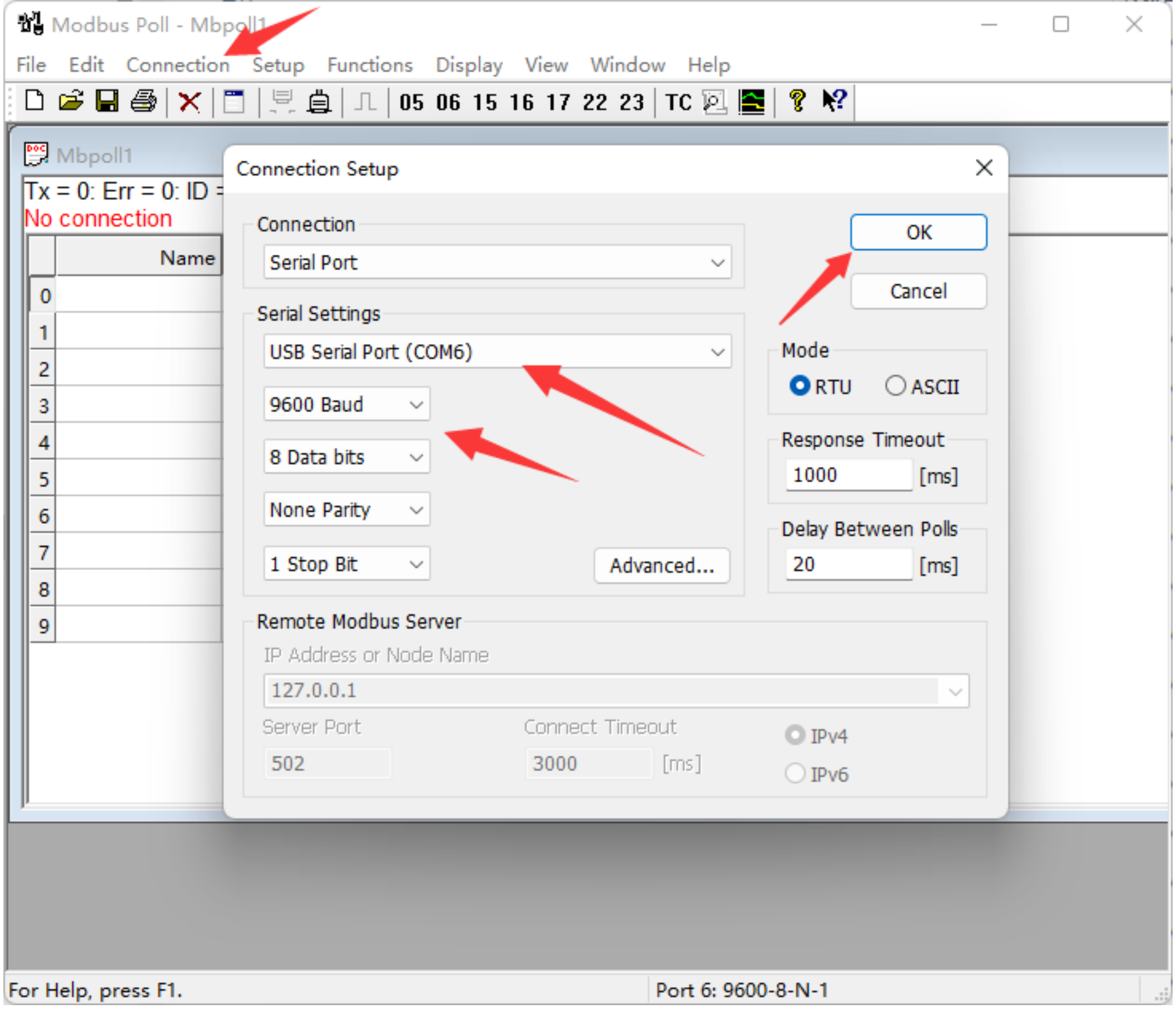
RTU: 01 04 00 00 00 08 F1 CC

ASCII: 3A 30 31 30 34 30 30 30 30 30 30 38 46 33 0D 0A

For Help, press F1. Port 6: 9600-8-N-1

(/wiki/ File:Modbus_Poll-1.png)

- Select Connection->Connect..., choose the corresponding serial port, set the baud rate to 9600, and select 8 Data bits and None Parity. Click OK to connect.



Tx = 0: Err = 0: ID =

No connection

Connection Setup

Connection: Serial Port

Serial Settings: USB Serial Port (COM6), 9600 Baud, 8 Data bits, None Parity, 1 Stop Bit

Mode: ● RTU ○ ASCII

Response Timeout: 1000 [ms]

Delay Between Polls: 20 [ms]

Remote Modbus Server

IP Address or Node Name: 127.0.0.1

Server Port: 502

Connect Timeout: 3000 [ms]

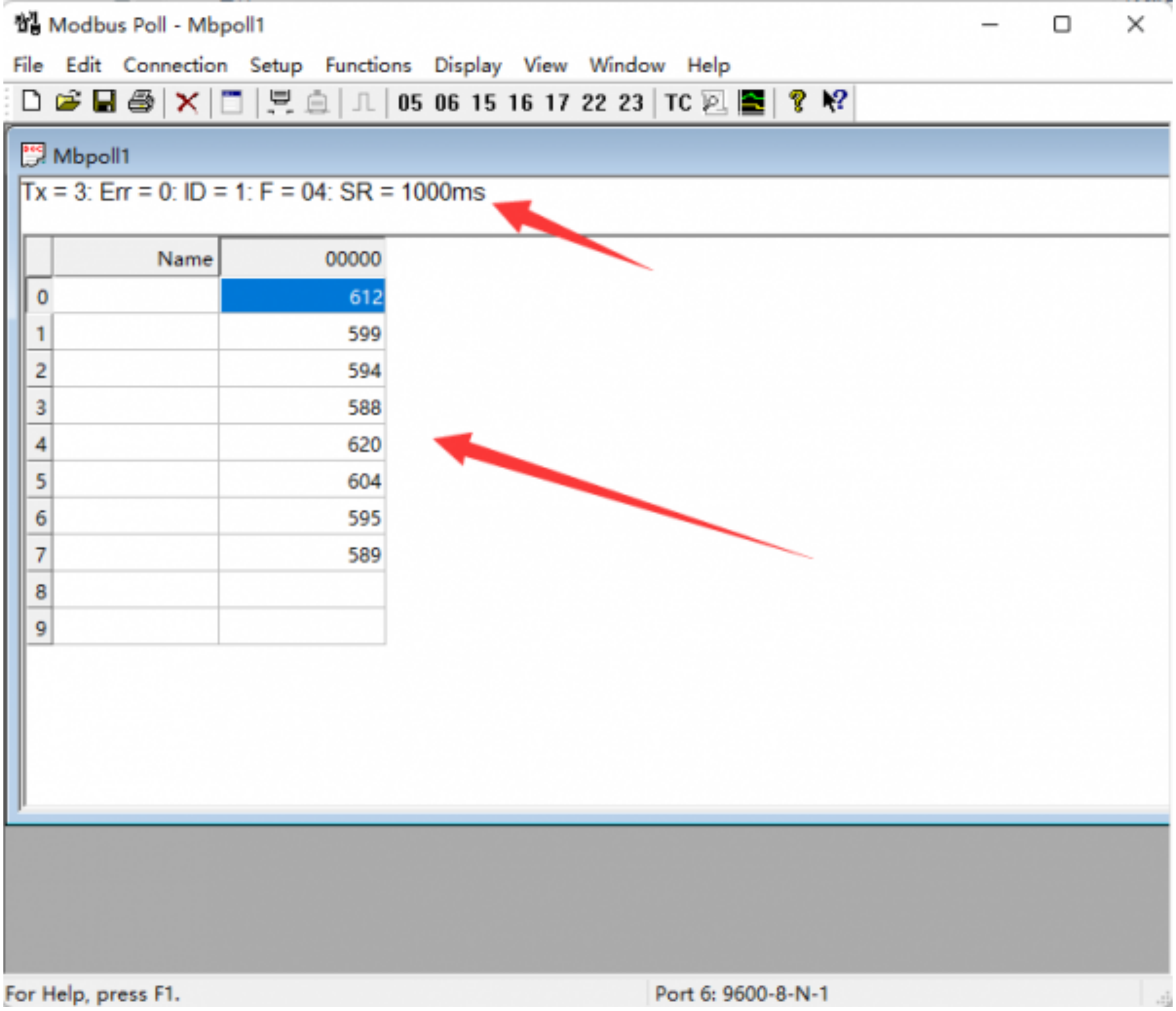
● IPv4 ○ IPv6

For Help, press F1.

Port 6: 9600-8-N-1

(/wiki/File:Modbus-RTU-Analog-Input-3.png)

- After successful connection, it can display the analog input data for channels 1-8.


Modbus Poll - Mbpoll1
File Edit Connection Setup Functions Display View Window Help
05 06 15 16 17 22 23 TC
Mbpoll1

Tx = 3: Err = 0: ID = 1: F = 04: SR = 1000ms

	Name	00000
0		612
1		599
2		594
3		588
4		620
5		604
6		595
7		589
8		
9		

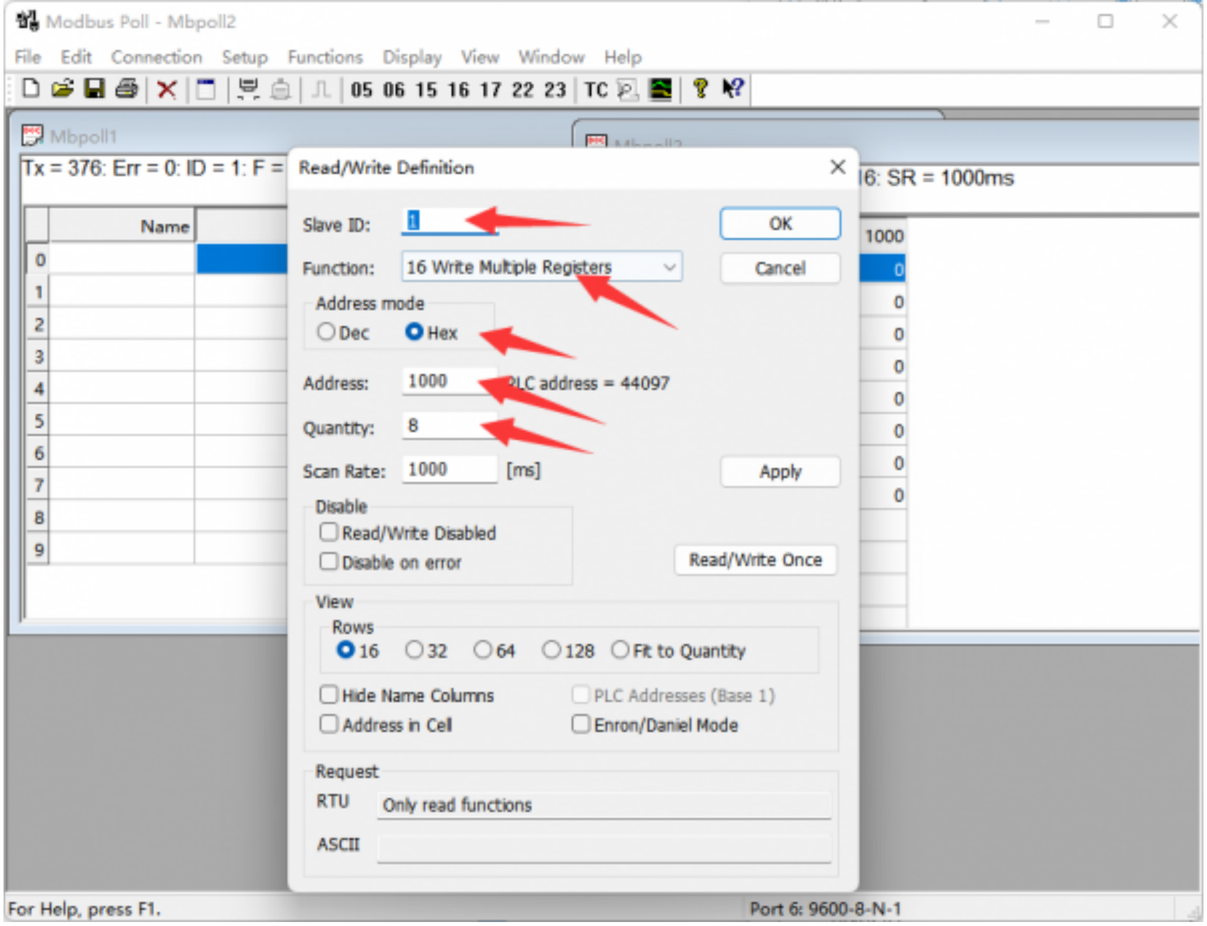
For Help, press F1.

Port 6: 9600-8-N-1

(/wiki/

File:Modbus_Poll-3.png)

- Modbus RTU Analog Input 8CH (A) displays the current by default, and the unit is uA. Modbus RTU Analog Input 8CH (B) displays the voltage by default, and the unit is mV.
- Choose File->New to create a new window, select Setup->Read/Write Definition. Select the actual device address for Slave ID, 16 Write Multiple Registers for Function, Hex for Address Mode, 1000 for Address, and 8 channels for Quantity, and then click "OK" to confirm.



Tx = 376: Err = 0: ID = 1: F =

16: SR = 1000ms

	Name	1000
0		0
1		0
2		0
3		0
4		0
5		0
6		0
7		0
8		
9		

Read/Write Definition

Slave ID: 1

Function: 16 Write Multiple Registers

Address mode: ○ Dec ● Hex

Address: 1000 PLC address = 44097

Quantity: 8

Scan Rate: 1000 [ms]

Disable

□ Read/Write Disabled

□ Disable on error

View

Rows ● 16 ○ 32 ○ 64 ○ 128 ○ Fit to Quantity

□ Hide Name Columns □ PLC Addresses (Base 1)

□ Address in Cell □ Enron/Daniel Mode

Request

RTU Only read functions

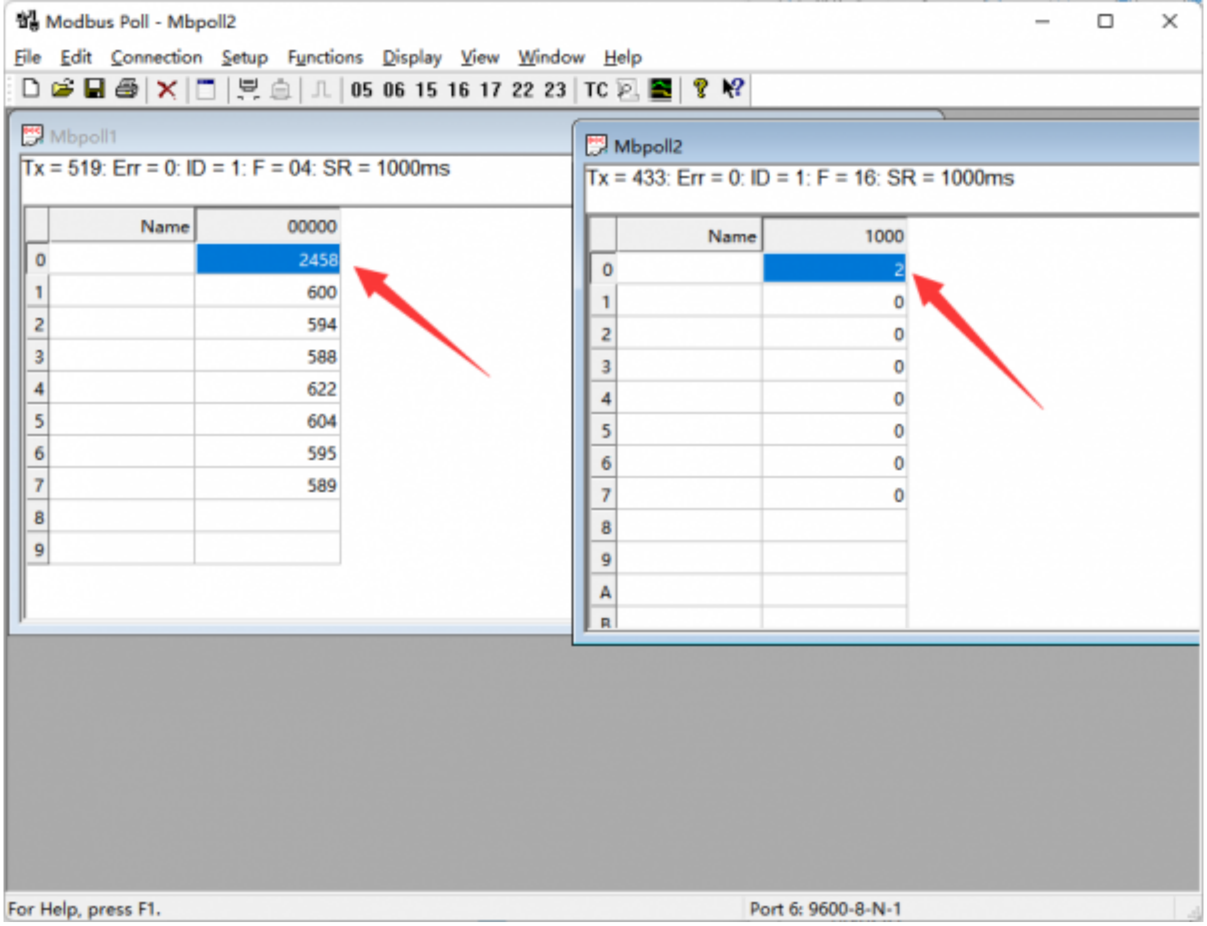
ASCII

(/wiki/

File:Modbus_Poll-4.png)

- The new window 2 can set up the measuring modes for different channels. For example, you can set the channel 1 mode as 2, that is, 0~20mA current mode. And channel 1 of the window 1 will display the current.

Note: The internal jumper wires should be modified when changing the current and voltage mode, otherwise, the measurement data will not be accurate.


Modbus Poll - Mbpoll2
File Edit Connection Setup Functions Display View Window Help
05 06 15 16 17 22 23 TC
Mbpoll1
Mbpoll2

Tx = 519: Err = 0: ID = 1: F = 04: SR = 1000ms

	Name	00000
0		2458
1		600
2		594
3		588
4		622
5		604
6		595
7		589
8		
9		

Tx = 433: Err = 0: ID = 1: F = 16: SR = 1000ms

	Name	1000
0		2
1		0
2		0
3		0
4		0
5		0
6		0
7		0
8		
9		
A		
B		

For Help, press F1. Port 6: 9600-8-N-1

(/wiki/
File:Modbus_Poll-5.png)

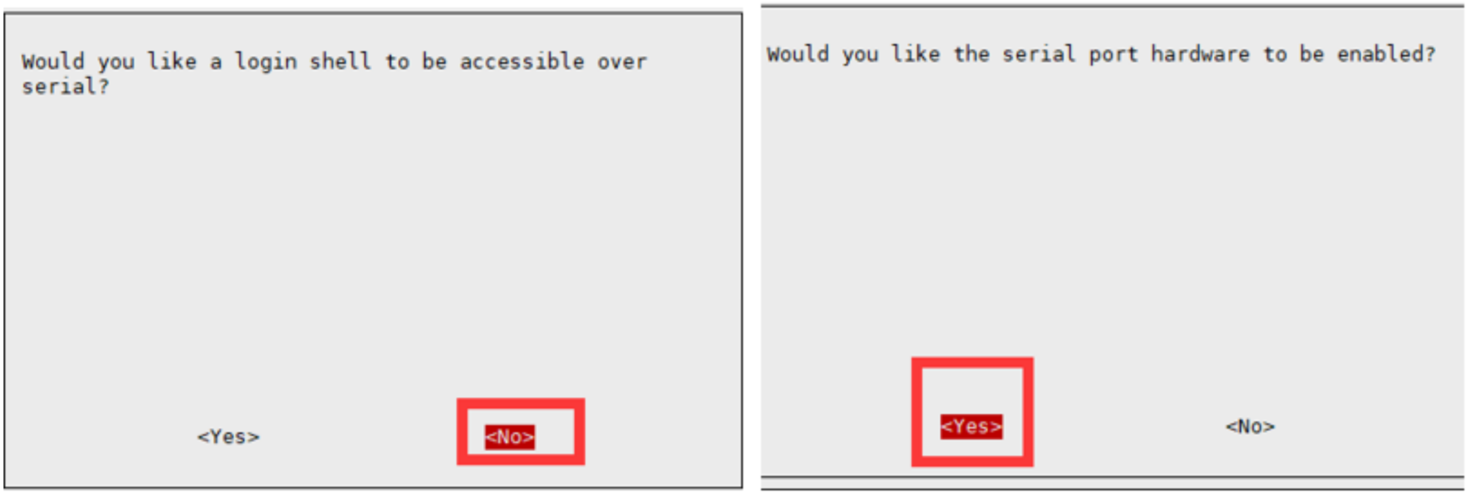
Demo Test

Note: RS485 can not be directly connected to the serial port of the Raspberry Pi, otherwise it may burn the device, you need to add 485 level conversion. For Raspberry Pi, it is recommended to work with the RS485 CAN HAT module. For NUCLEO-F103RB and Arduino, it is recommended to work with the RS485 CAN Shield module.

Raspberry Pi

Open the Raspberry Pi terminal and enter the following command to enter the configuration interface

```
sudo raspi-config
Select Interfacing Options -> Serial, disable shell access, and enable the hardw
are serial port
```



Would you like a login shell to be accessible over serial?

Would you like the serial port hardware to be enabled?

(/wiki/File:L76X_GPS_Module_rpi_serial.png)

Then restart Raspberry Pi:

```
sudo reboot
```

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

```
enable_uart=1
```

For Raspberry Pi 3B users, the serial port is used for Bluetooth and needs to be commented out:

```
#dtoverlay=pi3-miniuart-bt
```

Then restart Raspberry Pi:

```
sudo reboot
```

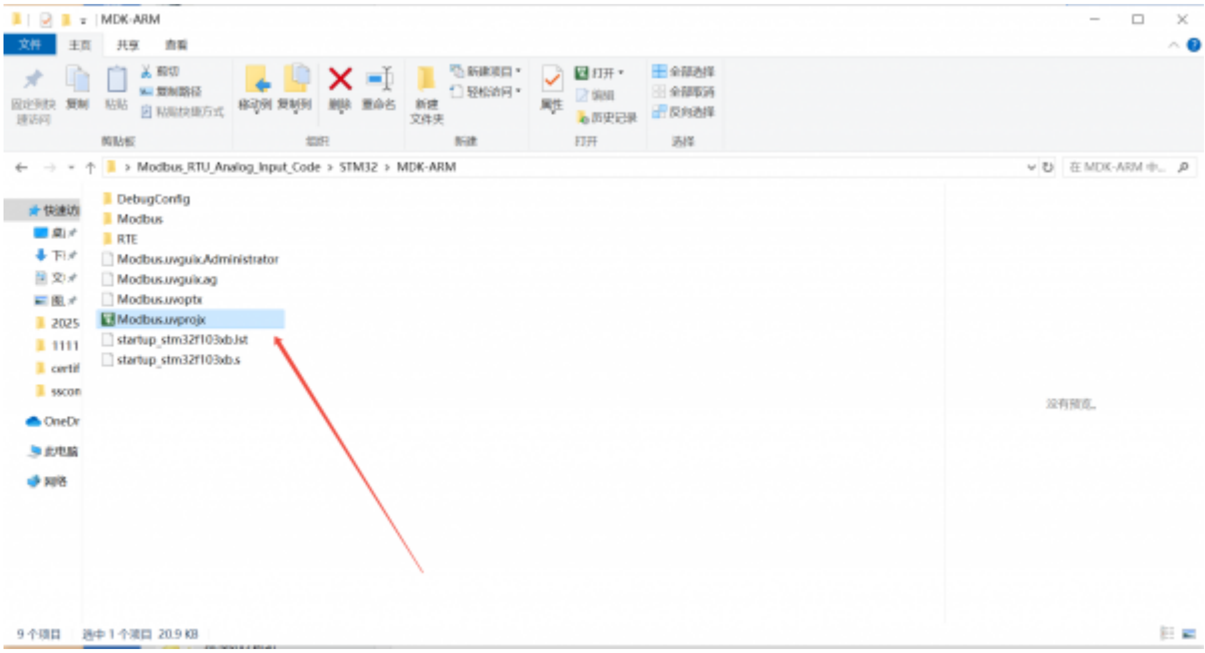
Insert the RS485 CAN HAT into the Raspberry Pi, and connect the Modbus RTU Relay module to the RS485 CAN HAT through A and B.
If you are using other 485 devices, make sure to connect A-A, B-B.< br/> Run the following commands to run the demo:

```
sudo apt-get install unzip
sudo apt-get install python3-pip
pip install modbus_tk
wget https://files.waveshare.com/wiki/Modbus-RTU-Analog-Input-8CH/Modbus_RTU_Ana
log_Input_Code.zip
unzip Modbus_RTU_Analog_Input_Code.zip
cd Modbus_RTU_Analog_Input_Code/Python3
sudo python3 modbus.py
```

STM32

Note: The STM32 demo is based on the NUCLEO-F103RB and RS485 CAN Shield module.

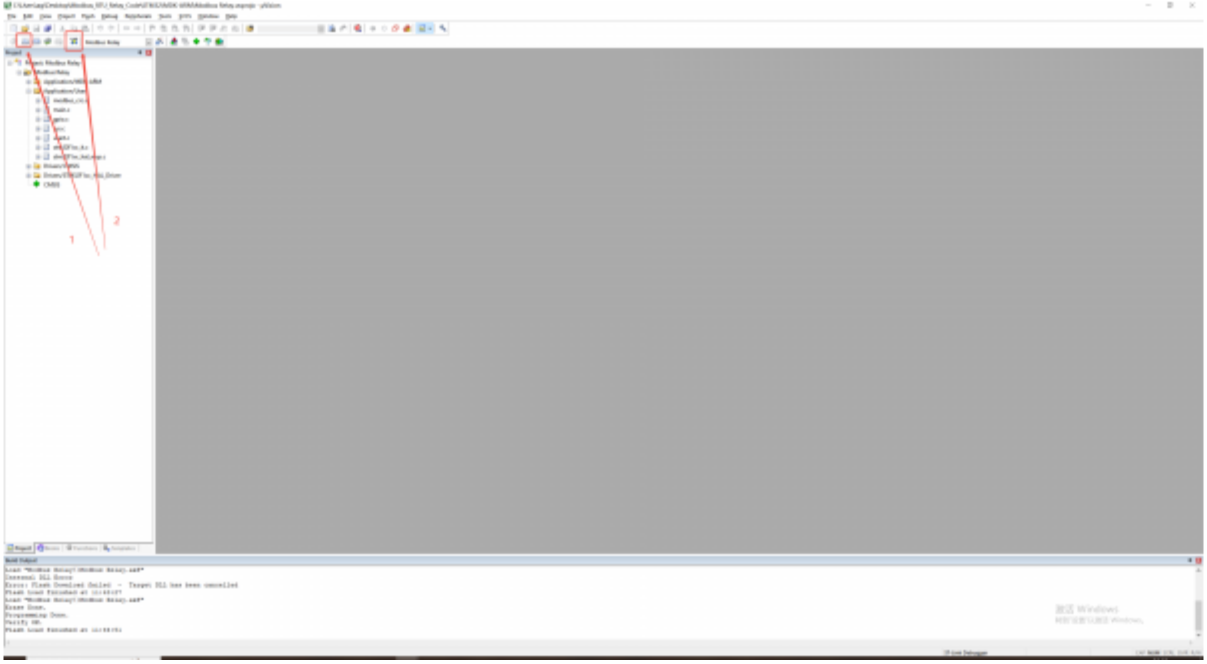
1. Download Demo (https://files.waveshare.com/wiki/Modbus-RTU-Analog-Input-8CH/Modbus_RTU_Analog_Input_Code.zip), find the STM32 project file Modbus.uvprojx in the path Modbus_RTU_Analog_Input_Code\STM32\MDK-ARM, and double-click to open the STM32 project file. Note that you should ensure Keil5 software is installed on your computer before using it.



(/wiki/

File:600px-Modbus-RTU-Relay-stm32-6.png)

2. Connect the STM32 to a computer via the STM32 download and debug probe. Compile and download the program to the development board.



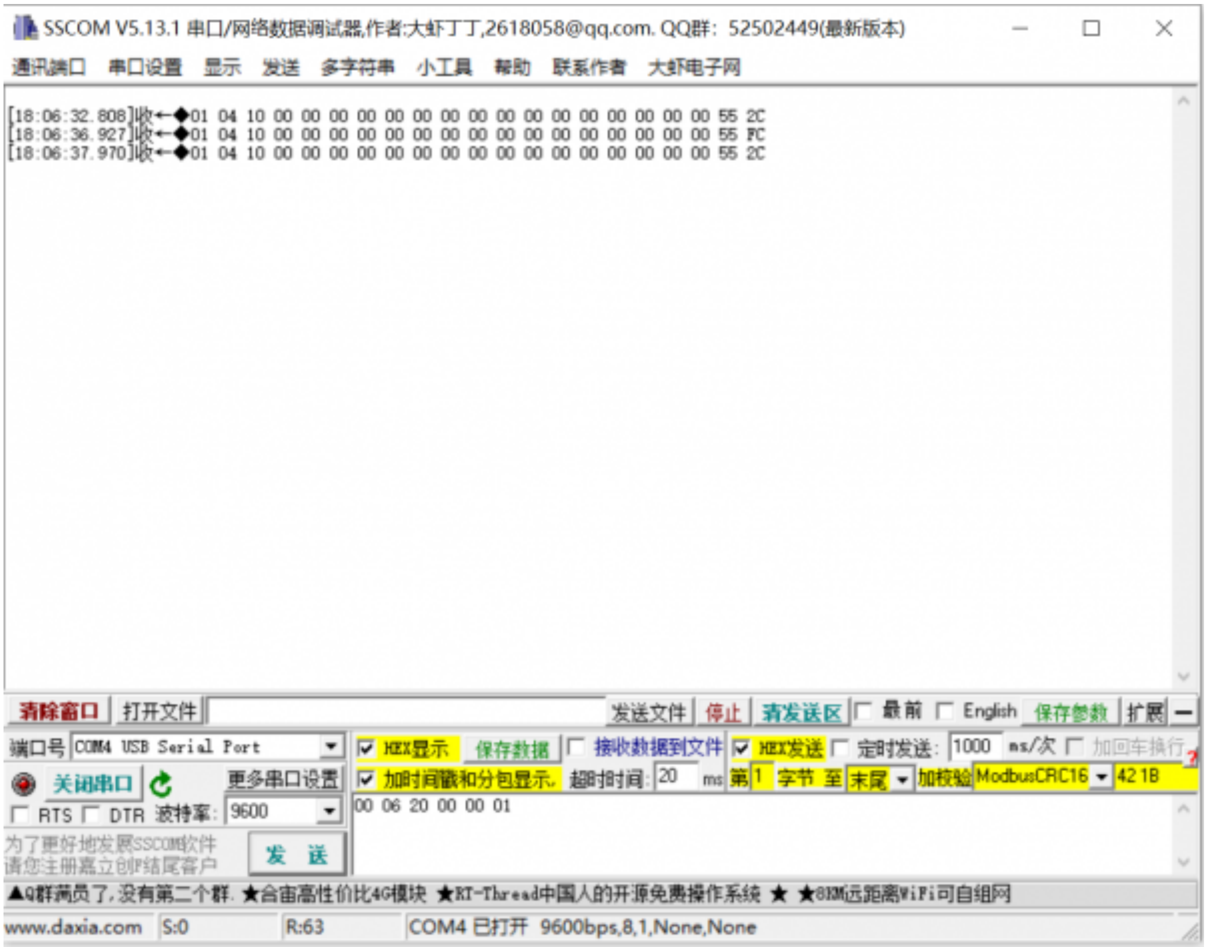
```
Load "Modbus Relay\\Modbus Relay.axf"
Internal DLL Error
Error: Flash Download failed  -  Target DLL has been cancelled
Flash Load finished at [illegible]
Load "Modbus Relay\\Modbus Relay.axf"
Erase Done.
Programming Done.
Verify OK.
Flash Load finished at [illegible]
```

(/wiki/

File:600px-Modbus-RTU-Relay-stm32-2.png)

3. Install the RS485 CAN Shield module on the STM32. Connect the RS485_A on the RS485 CAN Shield module to the RS485_A on the Modbus RTU Analog Input 8CH via a wire, and connect the RS485_B on the RS485 CAN Shield module to the RS485_B on the Modbus RTU Analog Input 8CH via a wire. Then power on the Modbus RTU Analog Input 8CH and the STM32 sequentially.

4. After the program runs normally, you can observe through the serial port assistant that the device prints the collected results



SSCOM V5.13.1 串口/网络数据调试器,作者:大虾丁丁,2618058@qq.com. QQ群：52502449(最新版本)

```
[18:06:32.808]收←◆01 04 10 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 55 2C
[18:06:36.927]收←◆01 04 10 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 55 FC
[18:06:37.970]收←◆01 04 10 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 55 2C
```

端口号 COM4 USB Serial Port

☑ HEX显示 ☑ 加时间戳和分包显示, 超时时间: 20 ms 第1 字节 至 末尾 加校验 ModbusCRC16 42 1B

☑ HEX发送 □ 定时发送: 1000 ms/次

波特率: 9600

```
00 06 20 00 00 01
```

www.daxia.com S:0 R:63 COM4 已打开 9600bps,8,1,None,None

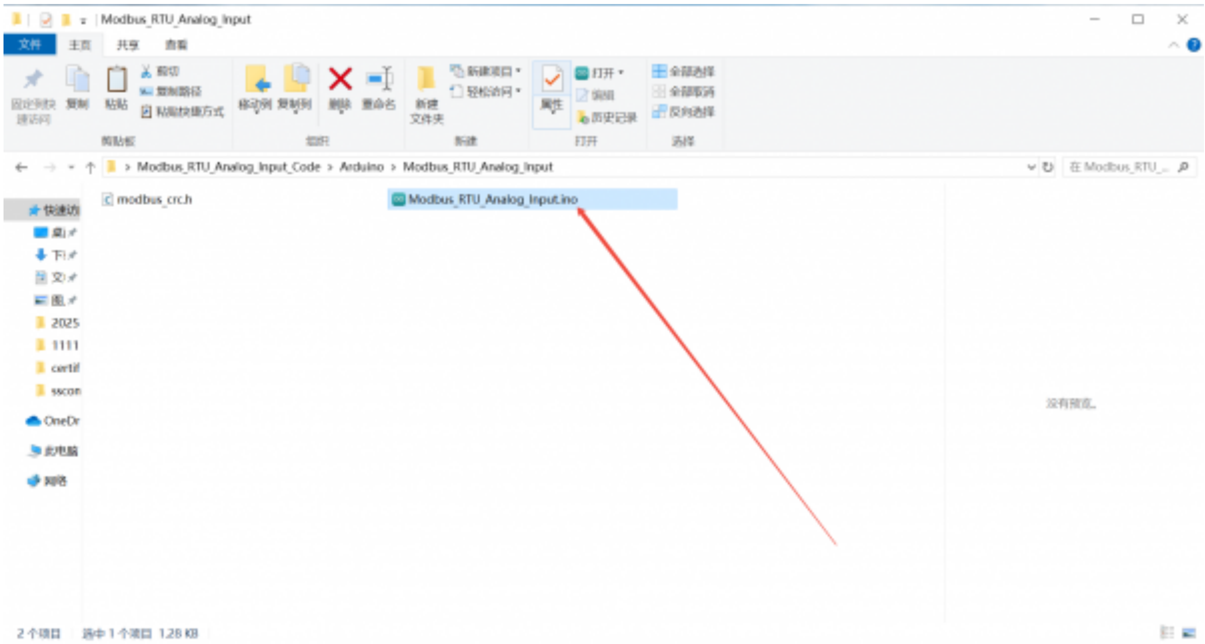
(/wiki/

File:600px-Modbus-RTU-Relay-stm32-7.png)

Arduino

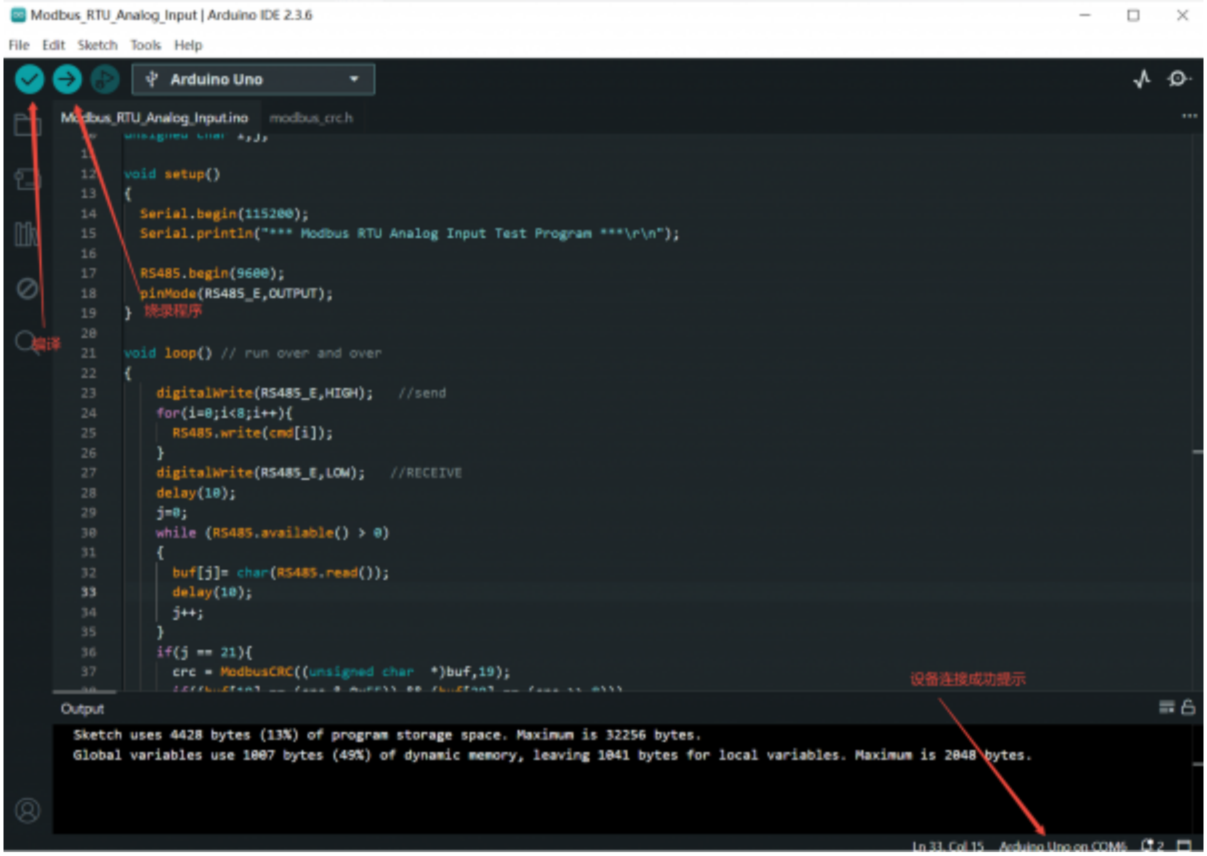
Note: The Arduino demo is based on the UNO PLUS and RS485 CAN Shield module.

1. Download Demo (https://files.waveshare.com/wiki/Modbus-RTU-Analog-Input-8CH/Modbus_RTU_Analog_Input_Code.zip), find the Arduino project file Modbus_RTU_Analog_Input.ino in the path Modbus_RTU_Analog_Input_Code\Arduino\Modbus_RTU_Analog_Input, and double-click to open the Arduino project file. Note that you should ensure Arduino IDE software is installed on your computer before using it.



(/wiki/File:600px-Modbus-RTU-Relay-arduino-6.png)

2. Connect the Arduino to the computer via a USB cable. In the Arduino IDE software, select the Arduino board model under Tools->Board. Choose the COM port that the Arduino is connected to under Tools->Port.
3. After seeing the prompt to connect to the computer in the lower right corner, click to compile and flash the program, and wait for the flashing to complete.



Modbus_RTU_Analog_Input | Arduino IDE 2.3.6
File Edit Sketch Tools Help
Arduino Uno
Modbus_RTU_Analog_Input.ino
modbus_crc.h
烧录程序
编译
设备连接成功提示
Output
Ln 33, Col 15 Arduino Uno on COM6

```
unsigned char i,j;

void setup()
{
  Serial.begin(115200);
  Serial.println("*** Modbus RTU Analog Input Test Program ***\r\n");

  RS485.begin(9600);
  pinMode(RS485_E,OUTPUT);
}

void loop() // run over and over
{
    digitalWrite(RS485_E,HIGH);   //send
    for(i=0;i<8;i++){
      RS485.write(cmd[i]);
    }
    digitalWrite(RS485_E,LOW);   //RECEIVE
    delay(10);
    j=0;
    while (RS485.available() > 0)
    {
      buf[j]= char(RS485.read());
      delay(10);
      j++;
    }
    if(j == 21){
      crc = ModbusCRC((unsigned char  *)buf,19);
```

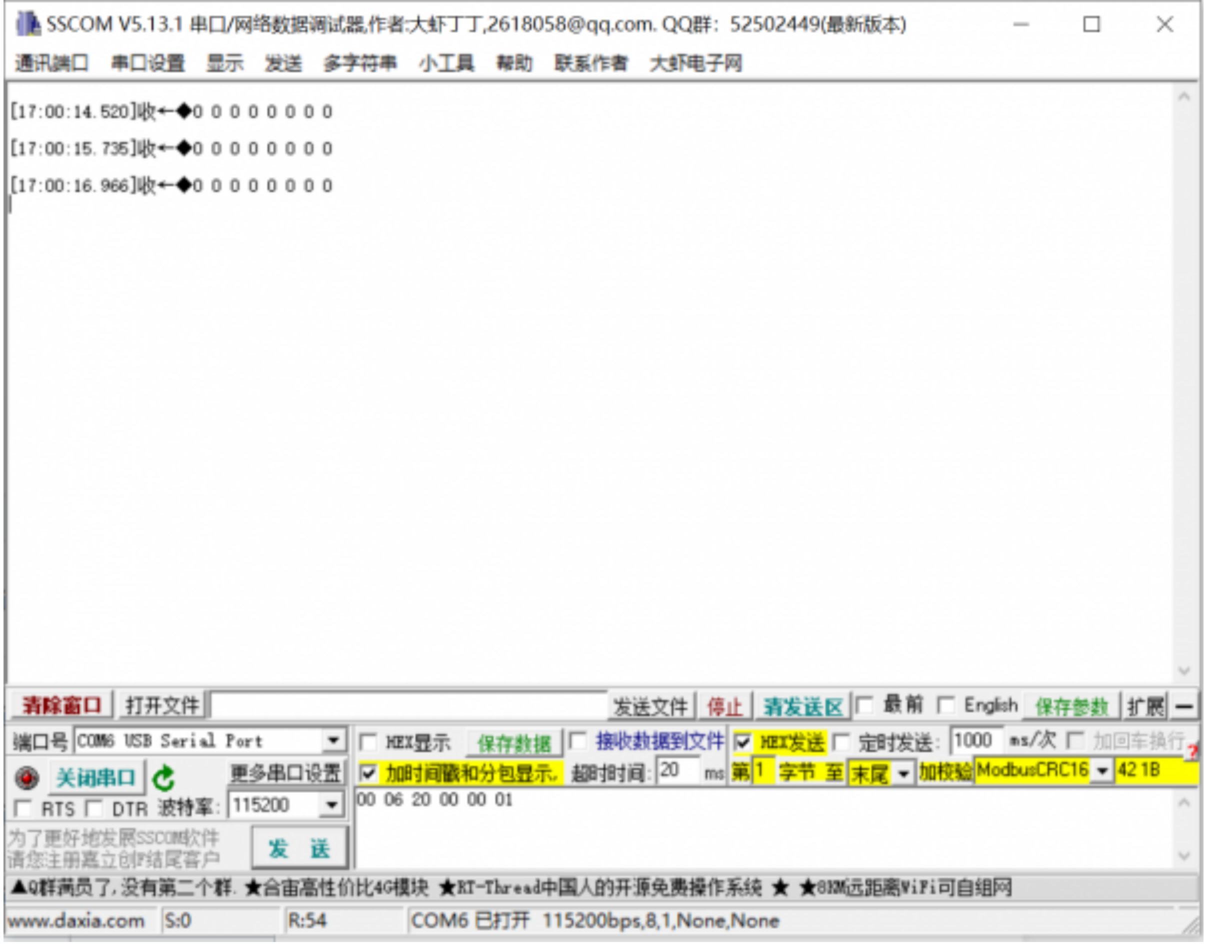
```
Sketch uses 4428 bytes (13%) of program storage space. Maximum is 32256 bytes.
Global variables use 1007 bytes (49%) of dynamic memory, leaving 1041 bytes for local variables. Maximum is 2048 bytes.
```

(/wiki/

File:600px-Modbus-RTU-Relay-arduino-7.png)

4. Install the RS485 CAN Shield module on the Arduino. Connect the RS485_A on the RS485 CAN Shield module to the RS485_A on the Modbus RTU Analog Input 8CH via a wire, and connect the RS485_B on the RS485 CAN Shield module to the RS485_B on the Modbus RTU Analog Input 8CH via a wire. Then power on the Modbus RTU Analog Input 8CH and the Arduino sequentially.

5. After the program runs normally, you can observe through the serial port assistant that the device prints the collected results


SSCOM V5.13.1 串口/网络数据调试器,作者:大虾丁丁,2618058@qq.com. QQ群：52502449(最新版本)
通讯端口 串口设置 显示 发送 多字符串 小工具 帮助 联系作者 大虾电子网
清除窗口 打开文件 发送文件 停止 清发送区 □ 最前 □ English 保存参数 扩展 —
关闭串口 更多串口设置 保存数据 发 送
为了更好地发展SSCOM软件
请您注册嘉立创F结尾客户
▲Q群满员了,没有第二个群. ★合宙高性价比4G模块 ★RT-Thread中国人的开源免费操作系统 ★ ★8KM远距离WiFi可自组网

```
[17:00:14.520]收←◆0 0 0 0 0 0 0 0
[17:00:15.735]收←◆0 0 0 0 0 0 0 0
[17:00:16.966]收←◆0 0 0 0 0 0 0 0
```

端口号 COM6 USB Serial Port

□ RTS □ DTR 波特率: 115200

□ HEX显示 □ 接收数据到文件 ☑ HEX发送 □ 定时发送: 1000 ms/次 □ 加回车换行

☑ 加时间戳和分包显示, 超时时间: 20 ms 第 1 字节 至 末尾 加校验 ModbusCRC16 42 1B

```
00 06 20 00 00 01
```

www.daxia.com S:0 R:54 COM6 已打开 115200bps,8,1,None,None

(/wiki/

File:600px-Modbus-RTU-Relay-arduino-8.png)

Development Protocol V2

Function Code Introduction

Function Code	Description
03	Read holding register
04	Read input register
06	Write single holding register
10	Write single input register

Register Address Introduction

Address (HEX)	Address storage content	Register value	Permission	Modbus Function Code
3x0000 …… 3x0007	Channels 1~8 input data	Read values as unsigned hexadecimal	Read	0x04
4x1000 …… 4x1007	Channels 1~8 data types	0x0000~0x0004 five ranges	Read/Write	0x03,0x06,0x10
4x2000	UART Parameter	The high eight bits indicate the parity mode: 0x00~0x02 The low eight bits indicate the baud rate mode: 0x00~0x07	Read/Write	0x03, 0x06
4x4000	Device Address	Directly store Modbus address Device address: 0x0001-0x00FF	Read/Write	0x03, 0x06
4x8000	Software Version	Converting to decimal and then shifting the decimal point two places to the left will represent the software version 0x0064 = 100 = V1.00	Read	0x03

Operation Command Introduction

Read Analog Input Command

Send code: 01 04 00 00 00 08 F1 CC

Field	Description	Note
01	Device Address	0x00 indicates the broadcast address, 0x01-0xFF indicates the device address
04	04 command	Read input register
00 00	Register Start Address	0x0000 - 0x0007 correspond to 1~8 input channels
00 08	Register Number	The number of the registers to be read, which must not exceed the maximum number of the channels
F1 CC	CRC16	The CRC16 checksum of the first 6 bytes of data

Return code: 01 04 10 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 55 2C

Field	Description	Note
01	Device Address	0x00 indicates the broadcast address, 0x01-0xFF indicates the device address
04	04 command	Read input register
10	Byte Number	Data length
00 00 …… 00 00	Register data	Indicates the values of analog inputs from channels 0 - 7 An unsigned 16-bit identifier for a channel, with the higher bits first and the lower bits last The data range is determined by the output data type
55 2C	CRC16	The CRC16 checksum of the first 6 bytes of data

For example: [Address 1 device]

```
Read 1–8 channels : 01 04 00 00 00 08 F1 CC
Read 1 channel : 01 04 00 00 00 01 31 CA
Read 2 channel : 01 04 00 01 00 01 60 0A
Read 3–5 channels : 01 04 00 02 00 03 11 CB
```

Read Channel Data Type Command

Send code: 01 03 10 00 00 08 40 CC

Field	Description	Note
01	Device Address	0x00 indicates the broadcast address, 0x01-0xFF indicates the device address
03	03 Command	Read holding register
10 00	Register Start Address	0x1000 - 0x1007 correspond to 1~8 input channels
00 08	Register Number	The number of the registers to be read, which must not exceed the maximum number of the channels
40 CC	CRC16	The CRC16 checksum of the first 6 bytes of data

Return code: 01 03 10 00 02 00 02 00 02 00 02 00 02 00 02 00 02 00 02 09 C3

Field	Description	Note
01	Device Address	0x00 indicates the broadcast address, 0x01-0xFF indicates the device address
03	03 Command	Read holding register
10	Byte Number	The number of all bytes of the returned status information

00 02
……
00 02
	Data Type	Indicates data types of 0-7 channels, 0x0000~0x0004 represents five ranges
		0x0000: Range 0~5V, output range 0~5000 or 0~10000, unit mV;
		0x0001: Range 1~5V, output range 1000~5000 or 2~10V, output range 2000~10000, unit mV;
		0x0002: Range 0~20mA, output range 0~20000, unit uA;
		0x0003: Range 4~20mA, output range 4000~20000, unit uA;
		0x0004: Direct output of numerical code, output range 0~4096, requires linear conversion to obtain actual measured voltage and current;

09 C3	CRC16	The CRC16 checksum of the first 6 bytes of data

For example: [Address 1 device]

```
Read data types for channels 1-8 : 01 03 10 00 00 08 40 CC
Read data type for channel 1 : 01 03 10 00 00 01 80 CA
Read data type for channel 2 : 01 03 10 01 00 01 D1 0A
Read data type for channels 3-5 : 01 03 10 02 00 03 A0 CB
```

Set Single-channel Data Type Command

Send code: 01 06 10 00 00 03 CD 0B

Field	Description	Note
01	Device Address	0x00 indicates the broadcast address, 0x01-0xFF indicates the device address
06	06 Command	Write single register
10 00	Register Start Address	0x1000 - 0x1007 correspond to data types of 1~8 input channels

00 03	Channel Data Type	Channel data types, 0x0000~0x0004 represents five ranges
		0x0000: Range 0~5V, output range 0~5000 or 0~10000, unit mV;
		0x0001: Range 1~5V, output range 1000~5000 or 2~10V, output range 2000~10000, unit mV;
		0x0002: Range 0~20mA, output range 0~20000, unit uA;
		0x0003: Range 4~20mA, output range 4000~20000, unit uA;
		0x0004: Direct output of numerical code, output range 0~4096, requires linear conversion to obtain actual measured voltage and current;

CD 0B	CRC16	The CRC16 checksum of the first 6 bytes of data

Return code: 01 06 10 00 00 03 CD 0B

Field	Description	Note
01	Device Address	0x00 indicates the broadcast address, 0x01-0xFF indicates the device address
06	06 Command	Write single register
10 00	Channel Data Type Address	0x1000 - 0x1007 correspond to data types of 1~8 input channels
00 03	Channel Data Type	Channel data type, 0x0000~0x0004 represents five ranges
CD 0B	CRC16	The CRC16 checksum of the first 6 bytes of data

For example: [Address 1 device]

```
Set data type to 0~20mA for channel 1 : 01 06 10 00 00 02 0C CB
Read data type 4~20mA for channel 2: 01 06 10 01 00 03 9C CB
```

Set Multi-channel Data Type Command

Send code: 01 10 10 00 00 08 10 00 03 00 03 00 03 00 03 00 03 00 03 00 03 00 03 91 2B

Field	Description	Note
01	Device Address	0x00 indicates the broadcast address, 0x01-0xFF indicates the device address
10	10 Command	Write multiple registers
10 00	Register Start Address	0x1000 - 0x1007 correspond to data types of 1~8 input channels
00 08	Register Number	Set register number, which must not exceed the maximum number of the channels
10	Byte Number	Set the number of bytes to be output

00 03 ...... 00 03	Command	Corresponding to data types of 0-7 channels, 0x0000~0x0004 represents five ranges
		0x0000: Range 0~5V, output range 0~5000 or 0~10000, unit mV;
		0x0001: Range 1~5V, output range 1000~5000 or 2~10V, output range 2000~10000, unit mV;
		0x0002: Range 0~20mA, output range 0~20000, unit uA;
		0x0003: Range 4~20mA, output range 4000~20000, unit uA;
		0x0004: Direct output of numerical code, output range 0~4096, requires linear conversion to obtain actual measured voltage and current;

91 2B	CRC16	The CRC16 checksum of the first 6 bytes of data

Return code: 01 10 10 00 00 08 C5 0F

Field	Description	Note
01	Device Address	0x00 indicates the broadcast address, 0x01-0xFF indicates the device address
10	10 Command	Write multiple registers
10 00	Register Start Address	0x1000 - 0x1007 correspond to data types of 1~8 input channels
00 08	Register Number	Set register number, which must not exceed the maximum number of the channels
C5 0F	CRC16	The CRC16 checksum of the first 6 bytes of data

For example: [Address 1 device]

```
Read data type 4~20mA for channels 1-8: 01 10 10 00 00 08 10 00 03 00 03 00 03 0
0 03 00 03 00 03 00 03 00 03 91 2B
Read data type for channels 3–5 : 01 10 10 02 00 03 06 00 01 00 01 00 01 BE 4A
```

Set Baudrate Command

Send code: 00 06 20 00 00 05 43 D8

Field	Description	Note
00	Device Address	0x00 indicates the broadcast address, 0x01-0xFF indicates the device address
06	06 command	Set the baud rate and device address
20 00	Command Register	0x2000: set the baud rate; 0x4000: set the device address
00	Parity Method	0x00: no parity, 0x01: even parity; 0x02: odd parity
05	Baud Rate Value	Correspondence of baud rate values 0x00: 4800 0x01: 9600 0x02: 19200 0x03: 38400 0x04: 57600 0x05: 115200 0x06: 128000 0x07: 256000
43 D8	CRC16	The CRC16 checksum of the first 6 bytes of data

Return code: 00 06 20 00 00 05 43 D8

Field	Description	Note
00	Device Address	0x00 indicates the broadcast address, 0x01-0xFF indicates the device address
06	06 command	Set the baud rate and device address
20 00	Command Register	0x2000: set the baud rate; 0x4000: set the device address
00	Parity Method	0x00: no parity, 0x01: odd parity; 0x02: even parity
05	Baud Rate	Correspondence of baud rate values 0x00: 4800 0x01: 9600 0x02: 19200 0x03: 38400 0x04: 57600 0x05: 115200 0x06: 128000 0x07: 256000
43 D8	CRC16	The CRC16 checksum of the first 6 bytes of data

For example: [Address 1 device]

```
Set the baud rate as 4800: 00 06 20 00 00 00 83 DB
Set the baud rate as 9600: 00 06 20 00 00 01 42 1B
Set the baud rate as 115200: 00 06 20 00 00 05 43 D8
```

Set Device Address Command

Send code: 00 06 40 00 00 01 5C 1B

Field	Description	Note
00	Device Address	0x00 indicates the broadcast address, 0x01-0xFF indicates the device address
06	06 command	Set the baud rate and device address
40 00	Command Register	0x2000: set the baud rate; 0x4000: set the device address
00 01	Device Address	Set the device address, 0x0001-0x00FF
5C 1B	CRC16	The CRC16 checksum of the first 6 bytes of data

Return code: 00 06 40 00 00 01 5C 1B

Field	Description	Note
00	Device Address	0x00 indicates the broadcast address, 0x01-0xFF indicates the device address
06	06 command	Set the baud rate and device address
40 00	Command Register	0x2000: set the baud rate; 0x4000: set the device address
00 01	Device Address	Set the device address, 0x0001-0x00FF
5C 1B	CRC16	The CRC16 checksum of the first 6 bytes of data

For example: [Address 1 device]

```
Set the device address as 0x01: 00 06 40 00 00 01 5C 1B
Set the device address as 0x02: 00 06 40 00 00 02 1C 1A
Set the device address as 0x03: 00 06 40 00 00 03 DD DA
```

Read Device Address Command

Send code: 00 03 40 00 00 01 90 1B

Field	Description	Note
00	Device Address	0x00 indicates the broadcast address, 0x01-0xFF indicates the device address
03	03 Command	Read the device address
40 00	Command register	0x4000: read the device address, 0x8000: read software version
00 01	Byte Number	Fixed 0x0001
90 1B	CRC16	The CRC16 checksum of the first 6 bytes of data

Return code: 01 03 02 00 01 79 84

Field	Description	Note
00	Device Address	0x00 indicates the broadcast address, 0x01-0xFF indicates the device address
03	03 Command	Read the software version and device address
02	Byte Number	The number of bytes returned
00 01	Device Address	Set the device address, 0x0001-0x00FF
79 84	CRC16	The CRC16 checksum of the first 6 bytes of data

For example: [Address 2 device]

```
Send: 00 03 40 00 00 01 90 1B
Return : 02 03 02 00 02 7D 85     // Address 0x02
```

Read Software Version Command

Send code: 00 03 80 00 00 01 AC 1B

Field	Description	Note
01	Device Address	0x00 indicates the broadcast address, 0x01-0xFF indicates the device address
03	03 Command	Read the software version and device address
80 00	Command register	0x4000: read the device address, 0x8000: read software version
00 01	Byte Number	Fixed 0x0001
8F CA	CRC16	The CRC16 checksum of the first 6 bytes of data

Return code: 01 03 02 00 64 B9 AF

Field	Description	Note
01	Device Address	0x00 indicates the broadcast address, 0x01-0xFF indicates the device address
03	03 Command	Read the software version and device address
02	Byte Number	The number of bytes returned
00 64	Software Version	Converting to decimal and then shifting the decimal point two places to the left will represent the software version 0x0064 = 100 = V1.00
B9 AF	CRC16	The CRC16 checksum of the first 6 bytes of data

For example:

```
Send: 00 03 80 00 00 01 AC 1B
Return: 01 03 02 00 64 B9 AF       //0x0064 = 100 =V1.00
```

Exception Function Code

When the received command is incorrect or the device is abnormal, an exception response will be returned in the following format:

Return: 01 85 03 02 91

Field	Description	Note
01	Device Address	0x00 indicates the broadcast address, 0x01-0xFF indicates the device address
85	Exception Function Code	Exception function code = Request function code + 0x80
03	Byte Number	Exception Code
02 91	CRC16	The CRC16 checksum of the first 6 bytes of data

An exception code is a single-byte value that indicates the type of error. Several commonly used exception codes defined by the Modbus protocol:

Exception Code	Name	Description
0x01	Illegal Function	The requested function code is not supported
0x02	Illegal Data Address	The requested data address is incorrect
0x03	Illegal Data Value	The requested data value or operation cannot be executed
0x04	Server Failure	Server equipment failure
0x05	Response	The request has been received and is being processed
0x06	Device Busy	The device is currently busy and cannot perform the requested operation

Resources

Demo

- Demo (https://files.waveshare.com/wiki/Modbus-RTU-Analog-Input-8CH/Modbus_RTU_Analog_Input_Code.zip)

Softwares

- Sscom serial port debugging assistant (https://files.waveshare.com/wiki/common/Sscom5.13.1.zip)
- Modbus Poll software (https://www.modbustools.com/download.html)
- SecureCRT software (https://files.waveshare.com/wiki/common/SecureCRT.7z)

Related Resources

- Modbus Protocol Specification (https://www.waveshare.com/wiki/Modbus_Protocol_Specification)
- Modbus Series BootLoader Description (https://www.waveshare.com/wiki/Modbus_Series_BootLoader_Description)

FAQ

Question:I sent a command to control the relay, but it didn't respond. What could be the issue??

Answer:

The command must be sent in hex format with a CRC checksum.

1、If the module does not respond to the command, verify that the baud rate and device ID are correct. You may also try Restoring the factory settings (https://www.waveshare.com/wiki/Templ

ate:Modbus_Series_Products:_Factory_Reset_and_Firmware_Upgrade_Guide).
2、If the above steps do not resolve the issue, please submit a ticket (https://service.waveshare.com/) to contact the Waveshare technical support team.

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/)

Retrieved from "https://www.waveshare.com/w/index.php?title=Modbus_RTU_Analog_Input_8CH&oldid=102116 (https://www.waveshare.com/w/index.php?title=Modbus_RTU_Analog_Input_8CH&oldid=102116)"