iManifold

INSTRUCTION

iManifold Robot Control System

Modbus Communication Usage Document

Copyright

Statement

Shenzhen iManifold Robot Technology Ltd. reserves the right to modify the product or its features without prior notice.

Shenzhen iManifold Robot Technology Ltd. does not assume the responsibility for direct or indirect injury or damage caused by improper use of the product.

Contact us

Shenzhen iManifold Robot Technology Ltd.

Address: 410, Building B, Phase II, Huafeng International Robot Industrial Park, Bao'an District, Shenzhen

iManifold Technology (Jiangyin) Ltd.

Address : Room 102, Building 60, Tianan Digital City, No. 55 Changshan Avenue, Jiangyin City, Wuxi City, Jiangsu Province

Contents

1 Modbus introduction to communication protocol

This function provides a modbus_tcp communication method for industrial robots, providing interfaces for system configuration parameters, variable resources, motion control instructions, and robot work status.

Through this function, modbus_tcp communication can be performed on the robot system, thereby achieving real-time monitoring and collection of the robot's operating status, multi device interconnection and collaborative operation, remote operation control, and maintenance needs.

Partial features:

1. Efficiency in remote mode

2. Update cycle:

1. The input variable is updated every 100ms

2. The variable part in hold is updated 100 times every 100ms, in a loop (all variables are updated once every 1000ms)

3. The coil variable is updated every 100ms

4. The discrete variable is updated every 100ms

5. Use the trigger command to update automatically through the signal slot

3. For detailed address, please use the address calculation tool

2 Register address table

Modbus supports four types of registers

Register type	Object type	Access type	Contents
Input_Register Status register	16-Bit	Read only	The I/O system provides this type of data
Coli Coil digital input	1-Bit	Reading and writing	Changing this type of data through an application
Holding_Registe Command Register	16-Bit	Reading and writing	Changing this type of data through an application
Discrete_Input Digital input	1-Bit	Read only	The I/O system provides this type of data

2.1 Input_Register

Status Register: Read Only

2.1.1 System status

Name	Data type	Address	Notes
System status	16-bit integer	0x0000	0: Standby mode; 1: Enable; 2: In progress; 3: Pause; 2560: Warning; 2561: Servo warning; 2562: Warning during execution; 2563: Warning during pause; 2564: Alarm;

2.1.2 Servo status

Name	Data type	Address	Notes
Servo status	16-bit integer	0x0001	0: Servo off; 1: Servo open; 2: Upper servo alarm; 3: Driver error;

2.1.3 Working mode

Name	Data type	Address	Notes
Working mode	16-bit integer	0x0002	0: Remote teaching mode; 1: Remote playback mode;

2.1.4 Current posture number

Name	Data type	Address	Notes
Posture number	16-bit integer	0x0003	Current posture number

2.1.5 Current external axis number

Name	Data type	Address	Notes
External axis number	16-bit integer	0x0004	

2.1.6 Playback speed

Name	Data type	Address	Notes
Playback speed	16-bit integer	0x0005	Percentage

2.1.7 Teaching speed

Name	Data type	Address	Notes
Teaching speed	16-bit integer	0x0006	Percentage

2.1.8 Current joint position

Name	Data type	Address	Notes
Axis1	64-bit double	0x1000~0x1003	
Axis2	64-bit double	0x1004~0x1007	
Axis3	64-bit double	0x1008~0x100B	
Axis4	64-bit double	0x100C~0x100F	
Axis5	64-bit double	0x1010~0x1013	
Axis6	64-bit double	0x1014~0x1017	
Axis7	64-bit double	0x1018~0x101B	
Axis8	64-bit double	0x101C~0x101F	
Axis9	64-bit double	0x1020~0x1023	
Axis10	64-bit double	0x1024~0x1027	
Axis11	64-bit double	0x1028~0x102B	
Axis12	64-bit double	0x102C~0x102F	
Axis13	64-bit double	0x1030~0x1033	
Axis14	64-bit double	0x1034~0x1037	
Axis15	64-bit double	0x1038~0x103B	
Axis16	64-bit double	0x103C~0x103F	

2.1.9 Current Cartesian position

Name	Data type	Address	Notes
x	64-bit double	0x1050~0x1053	
y	64-bit double	0x1054~0x1057	
z	64-bit double	0x1058~0x105B	
r	64-bit double	0x105C~0x105F	
p	64-bit double	0x1060~0x1063	
y	64-bit double	0x1064~0x1067	

2.1.10 Current workpiece coordinate system 1 position

Name	Data type	Address	Notes
x	64-bit double	0x1070~0x1073	
y	64-bit double	0x1074~0x1077	
z	64-bit double	0x1078~0x107B	
r	64-bit double	0x107C~0x107F	
p	64-bit double	0x1080~0x1083	
y	64-bit double	0x1084~0x1087	

2.1.11 Current workpiece coordinate system 2 position

Name	Data type	Address	Notes
x	64-bit double	0x1090~0x1093	
y	64-bit double	0x1094~0x1097	
z	64-bit double	0x1098~0x109B	
r	64-bit double	0x109C~0x109F	
p	64-bit double	0x10A0~0x10A3	
y	64-bit double	0x10A4~0x10A7	

2.1.12 Current world coordinate system position

Name	Data type	Address	Notes
x	64-bit double	0x10B0~0x10B3	
y	64-bit double	0x10B4~0x10B7	
z	64-bit double	0x10B8~0x10BB	
r	64-bit double	0x10BC~0x10BF	
p	64-bit double	0x10C0~0x10C3	
y	64-bit double	0x10C4~0x10C7	

2.1.13 Current flange coordinate system position

Name	Data type	Address	Notes
x	64-bit double	0x10D0~0x10D3	
y	64-bit double	0x10D4~0x10D7	
z	64-bit double	0x10D8~0x10DB	
r	64-bit double	0x10DC~0x10DF	
p	64-bit double	0x10E0~0x10E3	
y	64-bit double	0x10E4~0x10E7	

2.1.14 Alarm message

Name	Data type	Address	Notes
Alarm message	128-byte string	0x1100~0x113F	Save alarm information in byte stream

2.1.15 Current axis

Name	Data type	Address	Notes
tcs_id	16-bit integer	0x1140	
wcs_id	16-bit integer	0x1141	
pcs1_id	16-bit integer	0x1142	
pcs2_id	16-bit integer	0x1143	
Current axis	16-bit integer	0x1144	0: Joint coordinate system; 1: Cartesian coordinate system; 2: Workpiece coordinate system 1; 3: Workpiece coordinate system 2; 4: World coordinate system; 5: Tool coordinate system; 6: Expand joint coordinate system;

			7: Undefined

2.1.16 Current program

Name	Data type	Address	Notes
Current project name	128-byte string	0x1150~0x118F	Save alarm information in byte stream
Current project name	128-byte string	0x1190~0x11CF	Save alarm information in byte stream
Current number of running program lines	32-bit integer	0x11D0~0x11D1	
Current program execution mode	16-bit integer	0x11D2	0: Once; 1: Step forward; 2: Cycle;

2.1.17 Global numerical variable monitoring

Name	Data type	Address	Notes
Variable01	64-bit double	0x11E0~0x11E3	
Variable02	64-bit double	0x11E4~0x11E7	
Variable03	64-bit double	0x11E8~0x11EB	
Variable04	64-bit double	0x11EC~0x11EF	
Variable05	64-bit double	0x11F0~0x11F3	
Variable06	64-bit double	0x11F4~0x11F7	
Variable07	64-bit double	0x11F8~0x11FB	
Variable08	64-bit double	0x11FC~0x11FF	
Variable09	64-bit double	0x1200~0x1203	
Variable10	64-bit double	0x1204~0x1207	

2.1.18 Global string variable monitoring

Name	Data type	Address	Notes
Variable01	128byte string	0x1210~0x124F	
Variable02	128byte string	0x1250~0x128F	
Variable03	128byte string	0x1290~0x12CF	
Variable04	128byte string	0x12D0~0x130F	
Variable05	128byte string	0x1310~0x134F	
Variable06	128byte string	0x1350~0x138F	
Variable07	128byte string	0x1390~0x13CF	
Variable08	128byte string	0x13D0~0x140F	
Variable09	128byte string	0x1410~0x144F	
Variable10	128byte string	0x1450~0x148F	

2.1.19 Analog input

Name	Data type	Address	Notes
AI0	64-bit double	0x2000~0x2003	
AI1	64-bit double	0x2004~0x2007	
.....	.....	.....	
AI49	64-bit double	0x20C4~0x20C7	

2.1.20 Analog output

Name	Data type	Address	Notes
AO0	64-bit double	0x2100~0x2103	
AO1	64-bit double	0x2104~0x2107	
.....	.....	.....	
AO49	64-bit double	0x21C4~0x21C7	

2.2 Holding_Register

Command register (bold table of command words, the rest are parameters)

2.2.1 Servo enable

Name	Data type	Address	Notes
Servo Enable	16-bit integer	0x0000	The rising edge is effective, and it automatically resets after taking effect; If it is not currently enabled, the signal triggers the servo to enable it. If it is currently enabled, the signal triggers the servo to disable it

2.2.2 Clear alarm

Name	Data type	Address	Notes
clear alarm	16-bit integer	0x0001	The rising edge is effective, and it automatically resets after taking effect; Attempt to clear the alarm message after the signal is triggered

2.2.3 Set playback speed

Name	Data type	Address	Notes
Playback speed	16-bit integer	0x0002	Percentage ratio, value changes have immediate effects

2.2.4 Set teaching speed

Name	Data type	Address	Notes
Teaching speed	16-bit integer	0x0003	Percentage ratio, value changes have immediate

			effects

2.2.5 Switch working mode

Name	Data type	Address	Notes
Working mode	16-bit integer	0x0004	0: Remote jog; 1: Remote playback Value changes have immediate effects

2.2.6 Executing programs

Name	Data type	Address	Notes
Execution project name	128-byte string	0x1000~0x103F	Save alarm information in byte stream
Execution project name	128-byte string	0x1040~0x107F	Save alarm information in byte stream
Number of executed program lines	32-bit integer	0x1080~0x1081	
Execute program mode	16-bit integer	0x1082	0: Once; 1: Step forward; 2: Cycle;
Execute program	16-bit integer	0x1083	The rising edge is effective, and it automatically resets after taking effect; Pause the program after the signal is triggered
Pause program	16-bit integer	0x1084	The rising edge is effective, and it automatically resets after taking effect; Pause the program after the signal is triggered
Continue the program	16-bit integer	0x1085	The rising edge is effective, and it automatically resets after taking effect; Pause the program after the signal is

			triggered
Pause program	16-bit integer	0x1086	The rising edge is effective, and it automatically resets after taking effect; Pause the program after the signal is triggered
resumption of execution	16-bit integer	0x1087	The rising edge is effective, and it automatically resets after taking effect; Pause the program after the signal is triggered

2.2.7 Moving to point

Name	Data type	Address	Notes
Coordinate System Type	16-bit integer	0x10A0	0: Joint coordinate system; 1: Cartesian coordinate system; 2: Workpiece coordinate system 1; 3: Workpiece coordinate system 2; 4: World coordinate system; 5: Tool coordinate system; 6: Expand joint coordinate system; 7: Undefined
pcs_id	16-bit integer	0x10A1	
wcs_id	16-bit integer	0x10A2	
tcs_id	16-bit integer	0x10A3	
Joint position (16-double)	64-bit double*16	0x10A4~0x10E3	Storage structure reference 2.1.1

Cartesian position (6-double)	64-bit double*6	0x10E4~0x10FB	Storage structure reference 2.1.2
Attitude number	16-bit integer	0x10FC	
External axis number	16-bit integer	0x10FD	
Moving coordinate system	16-bit integer	0x10FE	0: Joint; 1: Cartesian
Execute motion to point	16-bit integer	0x10FF	0: Stop moving; 1: Start exercising Automatic stop motion after 500ms

2.2.8 Setting coordinate system

Name	Data type	Address	Notes
tcs_id	16-bit integer	0x1110	Value changes have immediate effects
wcs_id	16-bit integer	0x1111	Value changes have immediate effects
pcs1_id	16-bit integer	0x1112	Value changes have immediate effects
pcs2_id	16-bit integer	0x1113	Value changes have immediate effects
Current coordinate system	16-bit integer	0x1114	0: Joint coordinate system; 1: Cartesian coordinate system; 2: Workpiece coordinate system 1; 3: Workpiece coordinate system 2; 4: World coordinate system; 5: Tool coordinate system; 6: Expand joint coordinate system; 7: Undefined Value changes have immediate effects

2.2.9 Jogging robot

Name	Data type	Address	Notes
Sports mode	16-bit integer	0x1120	0: Continuous motion; 1: Inch movement
Axis 1 move	16-bit integer	0x1121	0: Stop moving; 1: Positive movement; 2: Reverse motion Automatic stop motion after 500ms
Axis 2 move	16-bit integer	0x1122	0: Stop moving; 1: Positive movement; 2: Reverse motion Automatic stop motion after 500ms
.....	.....	.....	.....
Axis 16 move	16-bit integer	0x1130	0: Stop moving; 1: Positive movement; 2: Reverse motion Automatic stop motion after 500ms

2.2.10 Numbers for output

Name	Data type	Address	Notes
Group number	16-bit integer	0x1140	A total of 50 groups, 1. 50
Port number	16-bit integer	0x1141	16 per group, 1-16
Target value	16-bit integer	0x1142	Value changes take effect immediately

2.2.11 Analog output

Name	Data type	Address	Notes
Number	16-bit integer	0x1150	A total of 50 groups, 1. 50. The changes will take immediate effect
Target value	64-bit double	0x1151-0x1154	Value

2.2.12 Writing numerical variables

Name	Data type	Address	Notes
Number	16-bit integer	0x1160	A total of 1000 groups, 1. 1000. The changes will take immediate effect
Target value	64-bit double	0x1161-0x1164	Value

2.2.13 String variable inhalation operation

Name	Data type	Address	Notes
Number	16-bit integer	0x1170	A total of 1000 groups, 1. 1000. The changes will take immediate effect
Target value	128 byte array	0x1171-0x11B0	Value change

2.2.14 Global numerical variable monitoring

Name	Data type	Address	Notes
Number 01	16-bit integer	0x11C0	
Number 02	16-bit integer	0x11C1	
Number 03	16-bit integer	0x11C2	
Number 04	16-bit integer	0x11C3	
Number 05	16-bit integer	0x11C4	
Number 06	16-bit integer	0x11C5	
Number 07	16-bit integer	0x11C6	
Number 08	16-bit integer	0x11C7	
Number 09	16-bit integer	0x11C8	
Number 10	16-bit integer	0x11C9	

2.2.15 Global character variable monitoring

Name	Data type	Address	Notes
Number 01	16-bit integer	0x11D0	
Number 02	16-bit integer	0x11D1	
Number 03	16-bit integer	0x11D2	
Number 04	16-bit integer	0x11D3	
Number 05	16-bit integer	0x11D4	
Number 06	16-bit integer	0x11D5	
Number 07	16-bit integer	0x11D6	
Number 08	16-bit integer	0x11D7	
Number 09	16-bit integer	0x11D8	
Number 10	16-bit integer	0x11D9	

2.3 Discrete_Input

2.3.1 Digital Input

Name	Data type	Address	Notes
DI0	1-bit	0x0000	
DI1	1-bit	0x0001	
.....	.....	.....	
DI799	1-bit	0x031F	

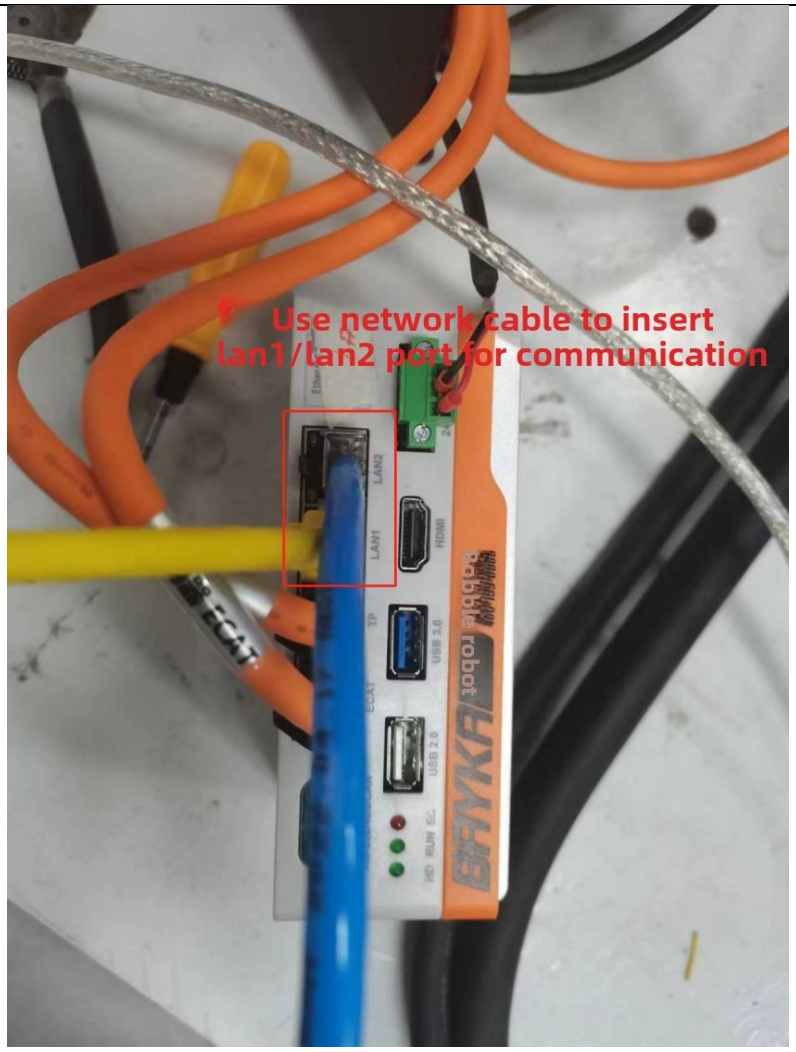
2.3.2 Digital output

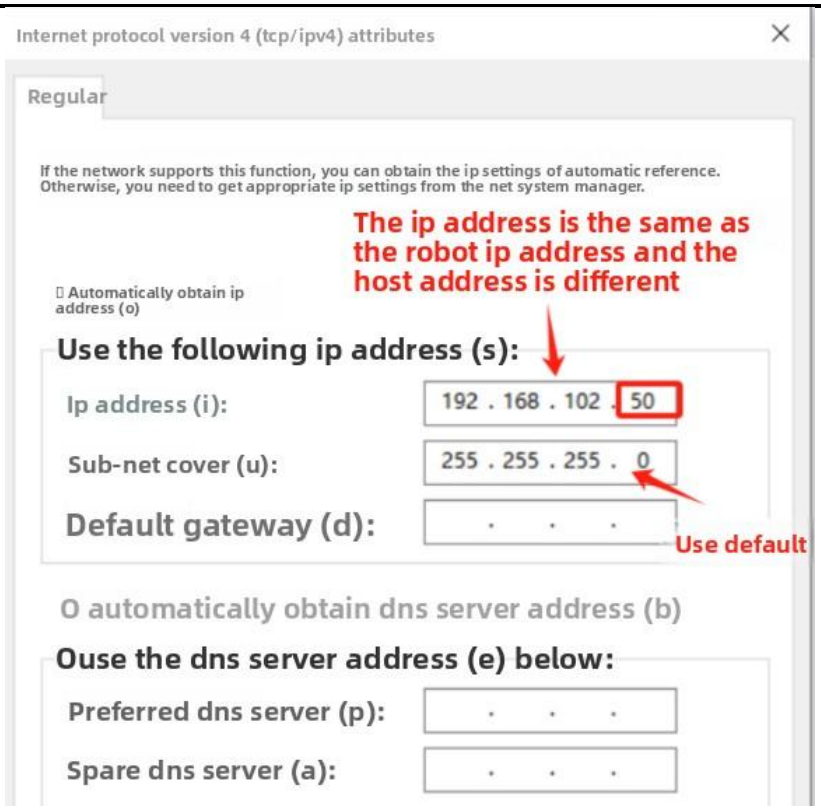
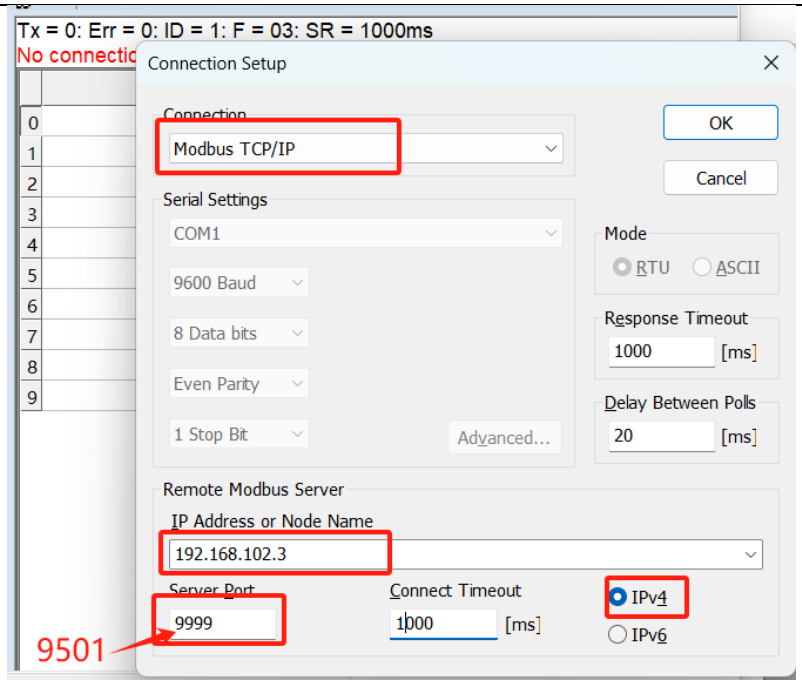
Name	Data type	Address	Notes
DO0	1-bit	0x1000	
DO1	1-bit	0x1001	
.....	.....	.....	
DO799	1-bit	0x131F	

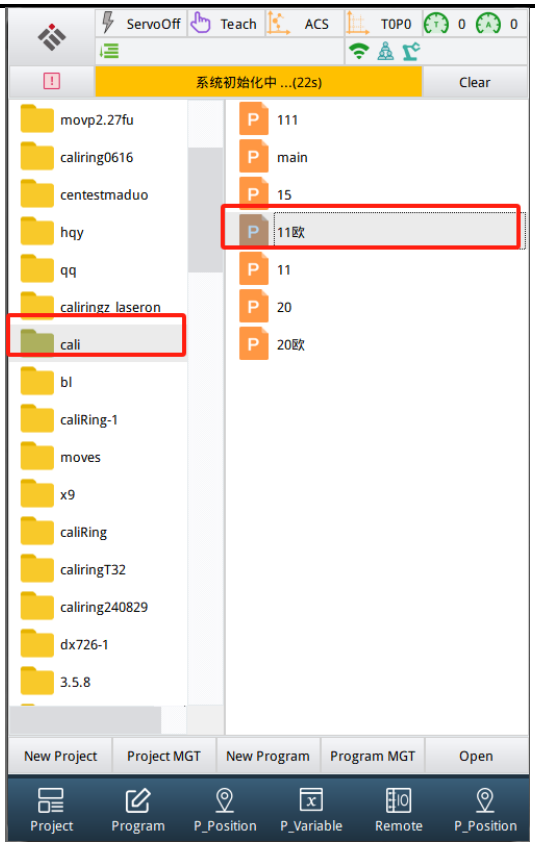
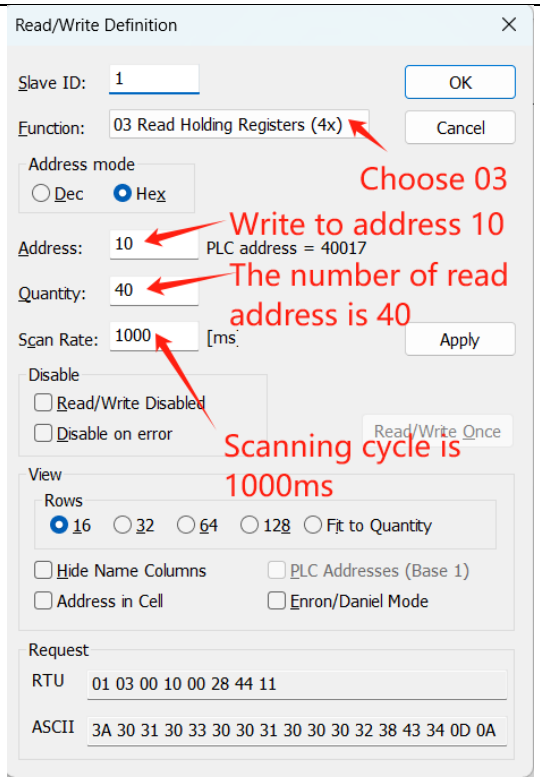
3 Communication instance

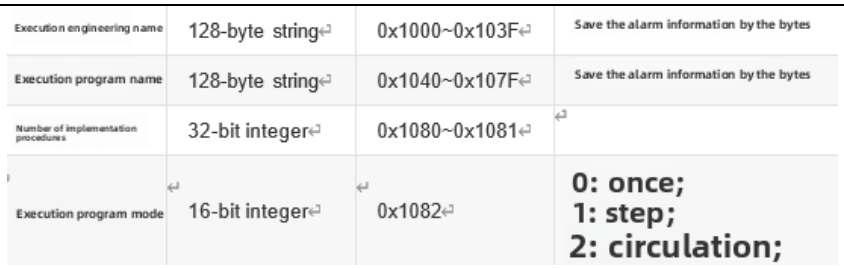
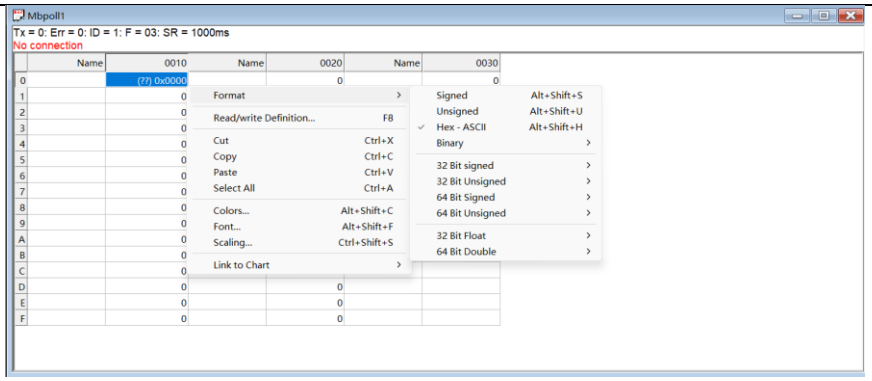
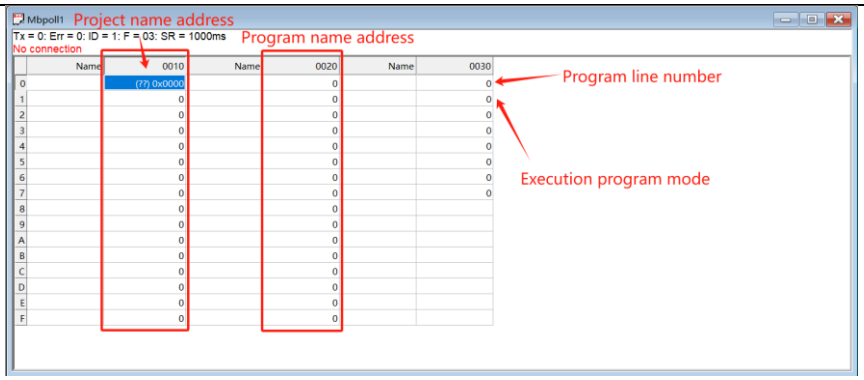
Modbus has numerous clients, and in this document, the Modbus poll tool is used to demonstrate how to communicate with robot systems through Moubus. Modbus communication only occurs when the robot is in remote mode.

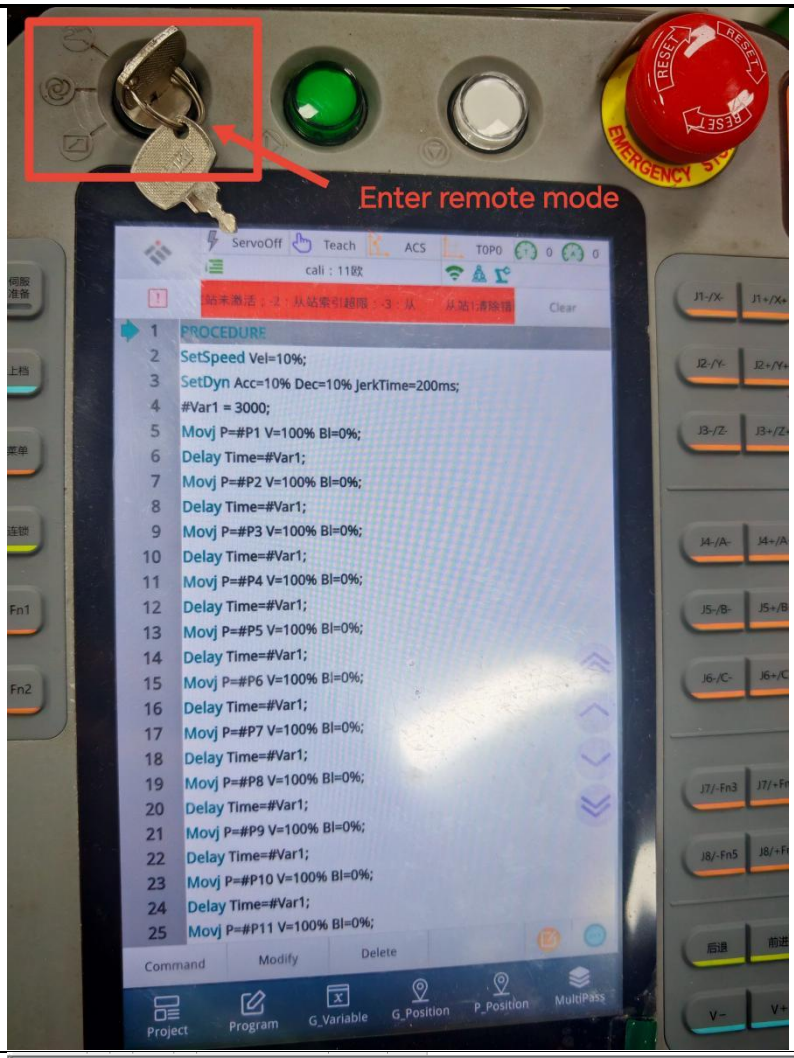
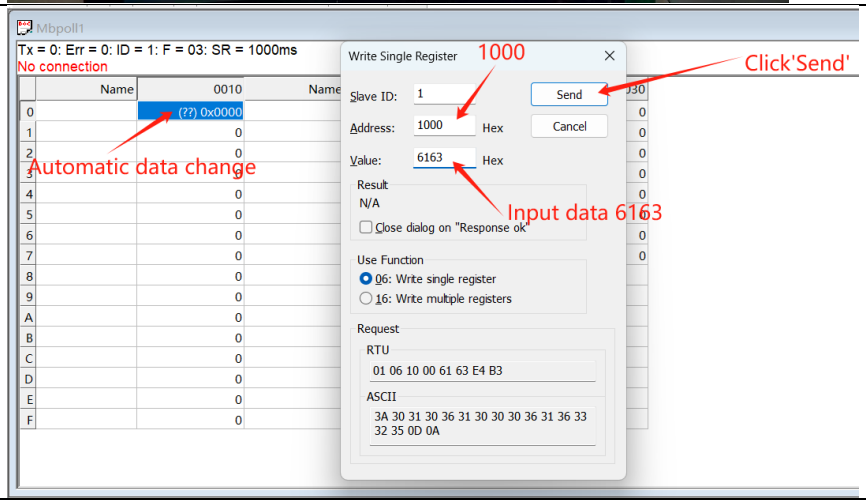
3.1 Operation instance

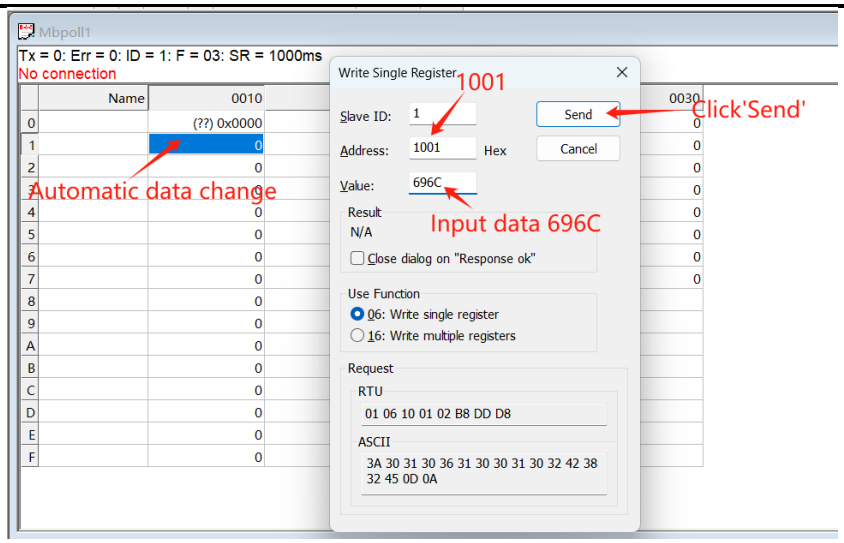
Project	Operation instance
Connection controller	 Connect Lan1 or Lan2 to the computer via Ethernet cable

Setting network	 Need to keep the computer IP consistent with the controller IP
Connections setting up	 After opening Modbus Poll, open the connection interface using Modbus TCP/IP protocol; The IP address should be the same as the robot's IP address; The server port is 9501; The protocol version is IPv4; Other defaults.

Project instance	 Demo: Start a project named Cali and a program named 11 ohms using Modbus poll. The execution mode of the program is multiple loops, starting from line 11 of the program
Set read/write definitions	 According to the different function codes used, it is necessary to find the corresponding function address table for correspondence; The default address format is decimal, which must be changed to

	hexadecimal, otherwise the data address will be incorrect;
Menu address switching	 According to the functional address table, find the execution project name, execution program name, number of execution program lines, and execution mode address. Different addresses require different data types to be input, and data types need to be converted. Need to use a conversion tool to convert to hexadecimal. Conversion tool（ http://www.kjson.com/transform/ox2str/ ）
Format selection	 Select addresses 1000 and 1040, and change the address format to "hexadecimal ASCII"
Data address	 According to the address table, find the line count of 0x1080 for executing the program and the address of 0x1082 for executing the program mode to set

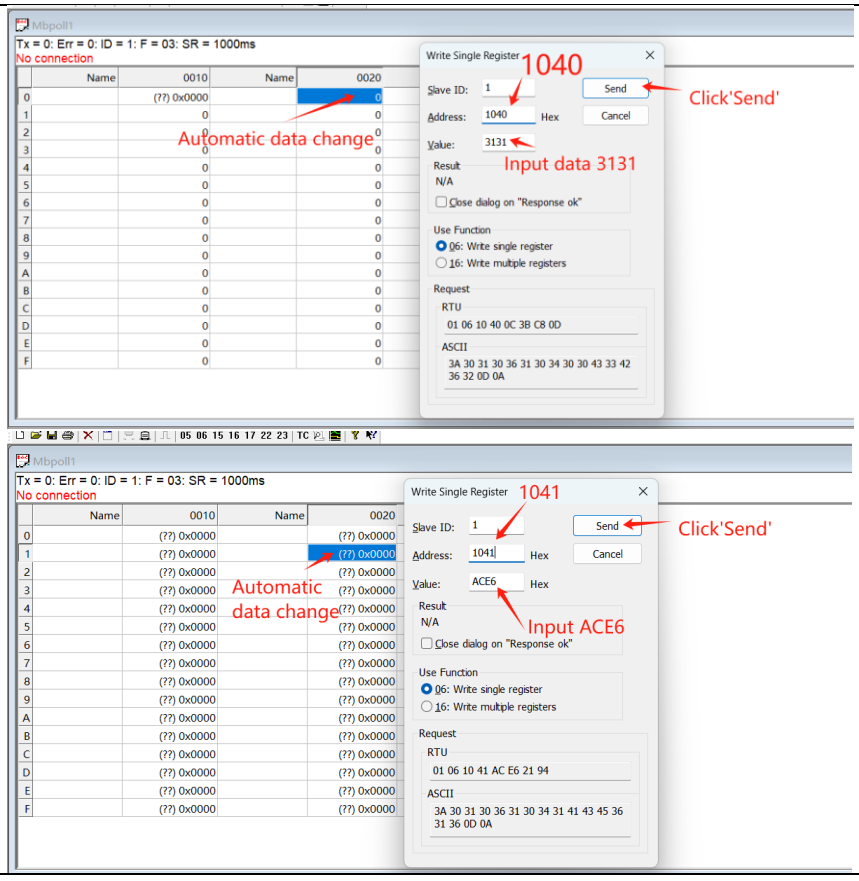
Switch mode	
Write project name	

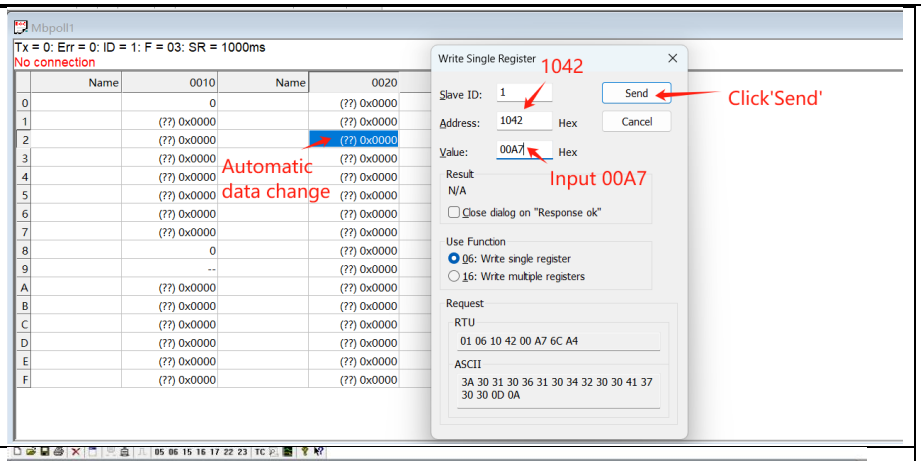
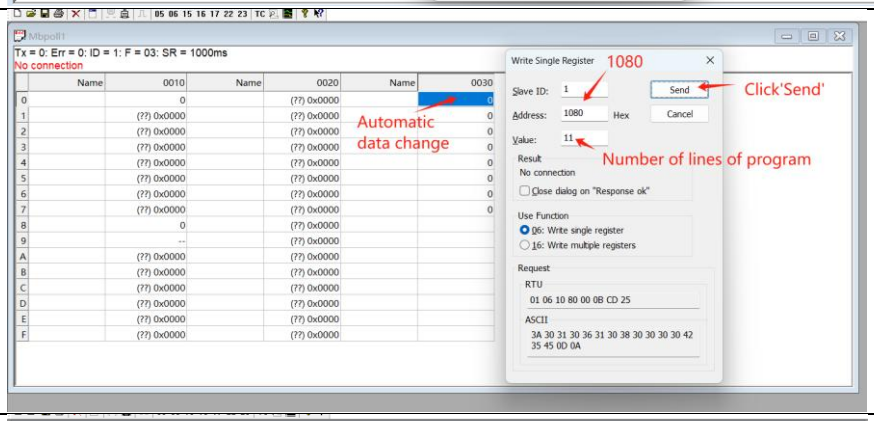
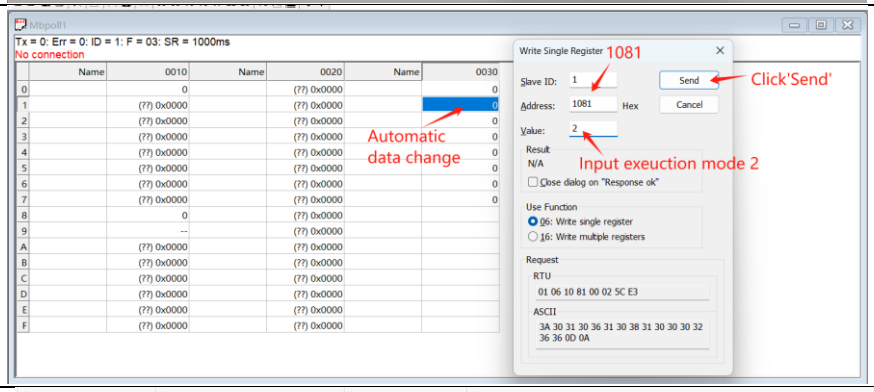


The data transmission format for this feature uses little_dedian_stwap, and attention should be paid to byte order. Convert the project name 'cali' to hexadecimal code using a conversion tool.

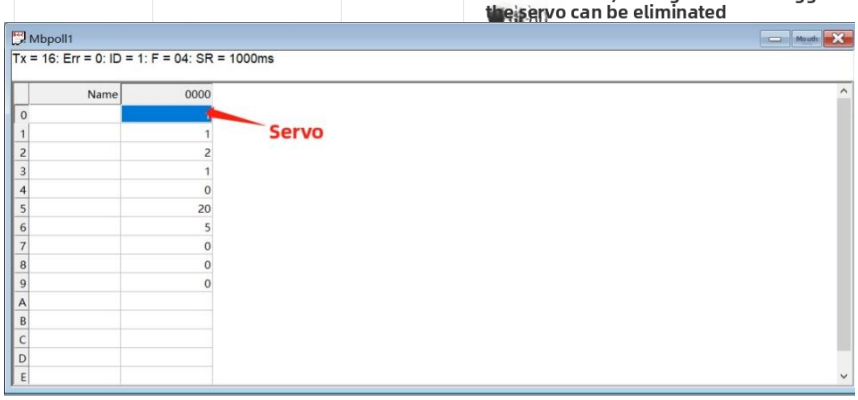
Conversion tool (http://www.kjson.com/transform/ox2str/)

Write program name

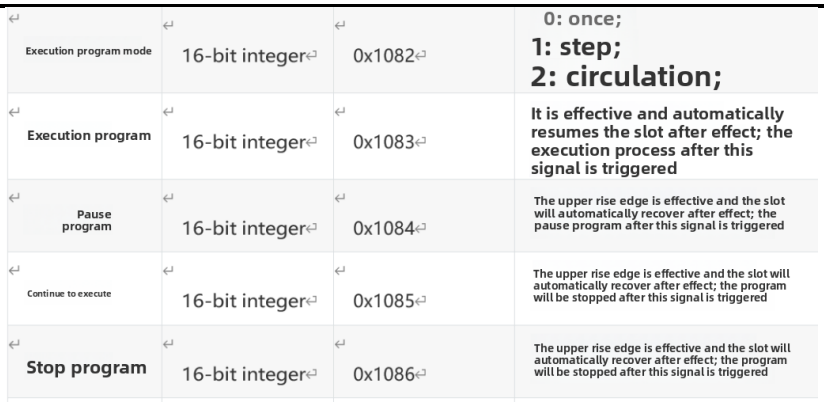
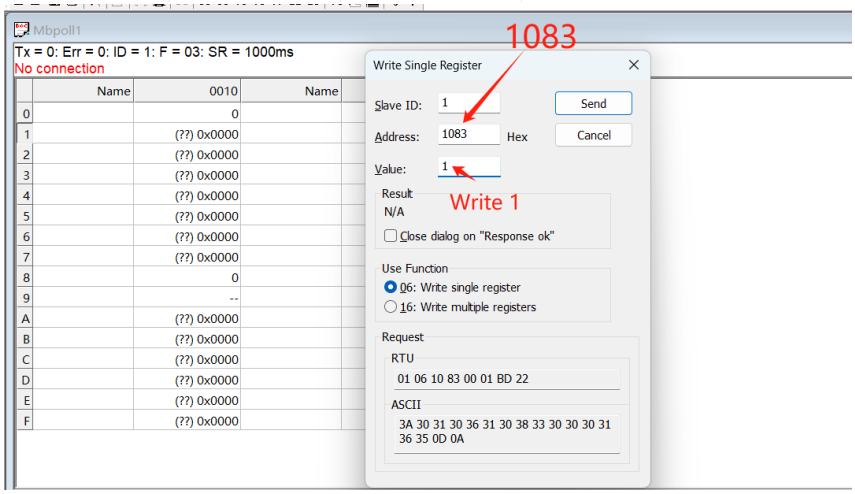


<table>
<tr><td></td><td> 
</td></tr>
<tr><td>Write the number of lines to execute the program</td><td> 
</td></tr>
<tr><td>Write and execute program mode</td><td> 
</td></tr>
<tr><td>Execute servo</td><td>

Name	Data type	Address	Note
Servo enable	16-bit integer	0x0000	The upper rise edge is effective and will automatically reslot after effect; □ if the current failure is not available, the servo function will be triggered. if the current energy has been used, the signal will be triggered and the servo can be eliminated



According to the address table of the Holding Register command register function, find the upper servo address for remote upper servo</td></tr>
</table>

Execute program	  According to the address table, find the address 0x1083, write the value 1, and the program will execute

4 Common problems

4.1 Conversion of data types

Modbus poll does not support strings, ACSII code needs to be converted to hexadecimal.

Chinese characters and Chinese symbols require the use of string to hexadecimal conversion tools.

Example: "Circle" → "e59c88"

Conversion tool (http://www.kjson.com/transform/ox2str/)

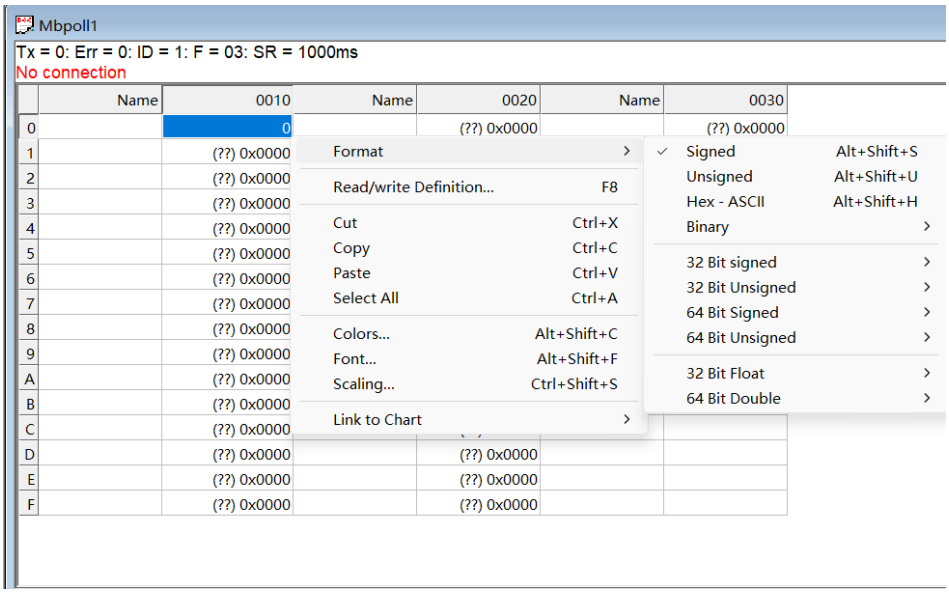
ACSII code requires the use of ASCII to hexadecimal conversion tools.

Example: "Cali" → "63616C69"

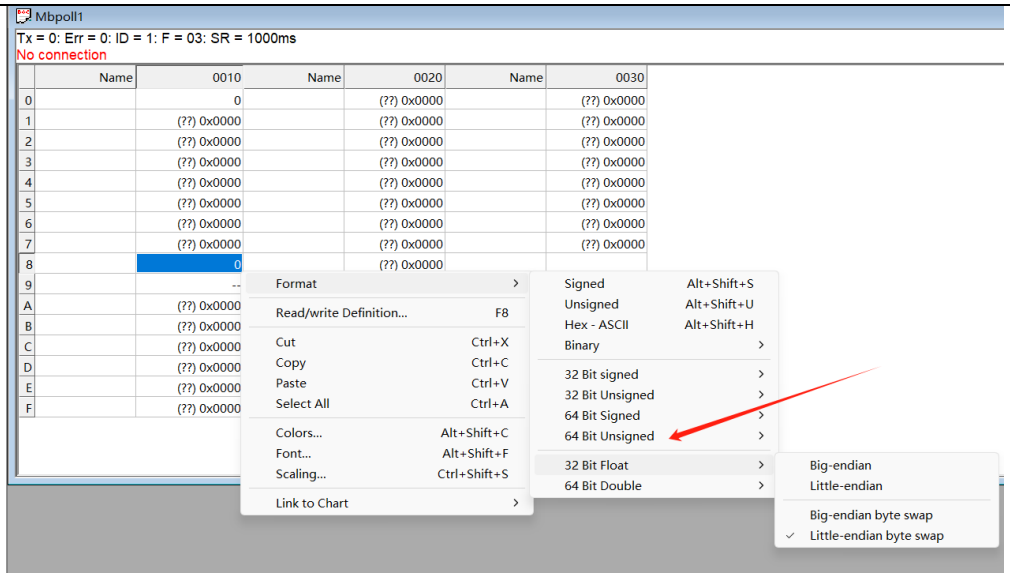
4.2 Data Format

The data transmission format for this feature uses little_dedian_stwap. The data types of the menu are 16 bit integer, 64 bit double, 32-bit string, and 128 byte string. When viewing data, it is important to pay attention to the data types.

Modbus Poll defaults to "16 bit integer (signed)".



If the data type is "64 bit float", the format needs to be set to "64 bit floating-point number" → "small byte swap" to view the data.



4.3 Input of numerical values

The address value input should be based on the remarks column of the function code. For the rising edge effective command word, only "1" can be input. When inputting string data to the address, the format needs to be changed to "Hexadecimal-ASCII", and then double-click the value to be modified. (When using the "Write to a single register" function, the default input is decimal and cannot write hexadecimal data.) When inputting hexadecimal codes, note that 4 characters form a group and the byte order should be paid attention to. When the data exceeds 4 digits, multiple addresses need to be input.

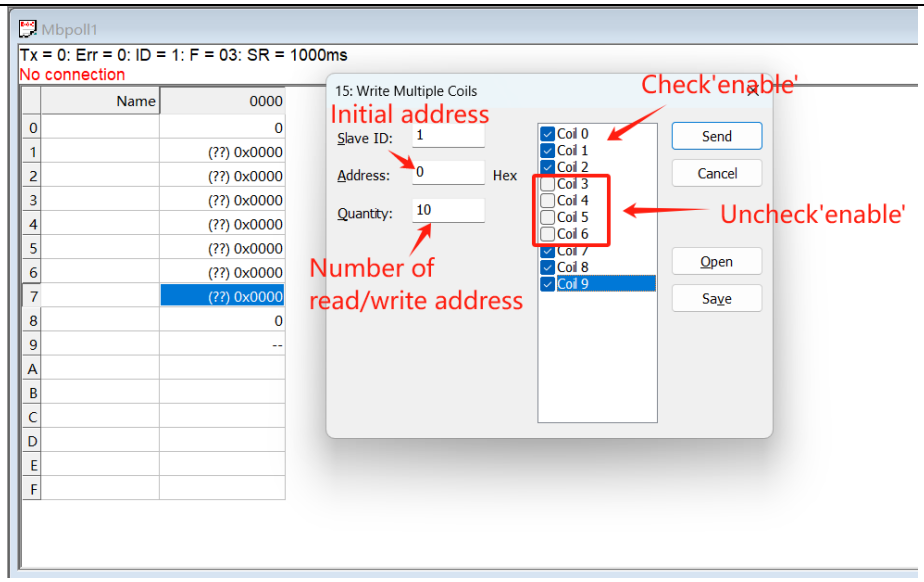
For example, "Circle" → ".e59c88", when inputting the address, it should be written as "0x9ce5" "0x0088".

4.4 Definition of Read/Write

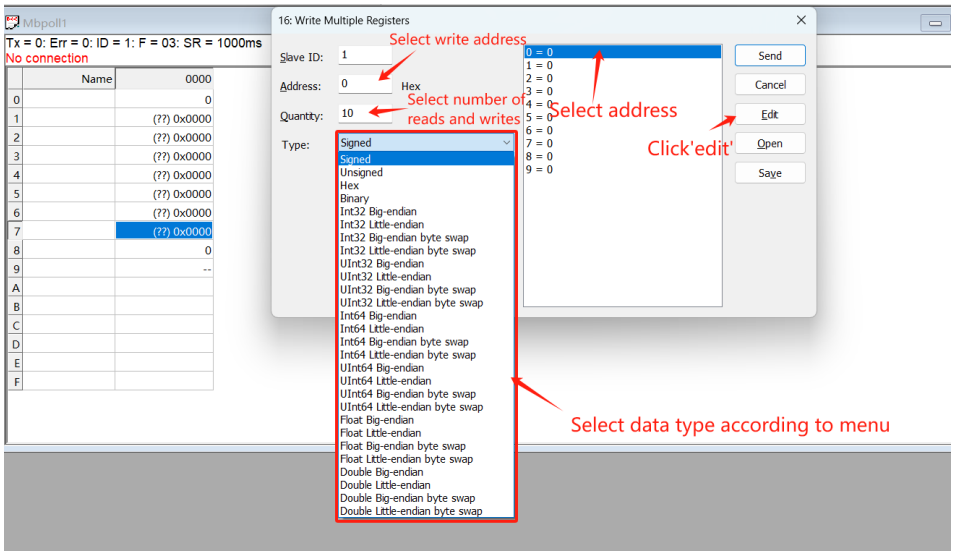
In the definition of read/write, the slave station is fixed at 1; There are a total of 4 function codes: 01, 02, 03, 04; The default address format is decimal, which needs to be changed to hexadecimal, otherwise the data address will be incorrect; Default scanning rate is 1000ms

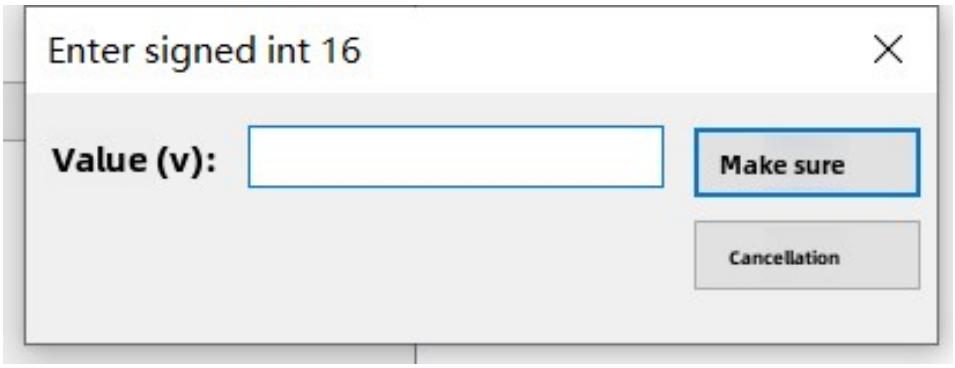
4.5 Writing to multiple coils/registers

Modbus supports multiple coil/register inputs, for example: enabling multiple coils



Example: Writing multiple registers





4.6 Modbus poll error codes

No connection	The normal unconnected state can be eliminated by connecting when the normal COM port is not occupied.
Timeout	All commands issued by software without a response from the slave device will display Timeout. There are many possibilities for the slave device not to reply, such as: 1. Connection configuration error, the host's baud rate, Slave ID and other information do not correspond to the slave device, and the slave will not respond. 2. The line is abnormal, and there is an abnormality in the communication line before the computer follows the device, and it is also unable to receive a reply normally. 3. The slave device does not respond to abnormal parsing, which can be found in the detailed explanation of the Modbus protocol.
Write error/Read error	Using a USB to 485 conversion tool, swapping the A and B wires of 485 may result in this situation. If a USB to 232 or TTL converter is used, short circuiting Tx and Rx will result in this situation. When connecting with a network cable, disconnecting the cable can cause this situation. During the sending process, similar errors may occur when data conflicts are received on the bus.
Checksum Error	The CRC check returned by the slave device is incorrect. There is interference on the communication bus. The checksum, data bits, and stop bits in the connection configuration are configured incorrectly.
Insufficient bytes received	If the number of received bytes is

	incomplete, it may be due to the circuit or some reason that the length of the returned instruction does not match the theoretical length of the returned instruction, which will result in some errors.
Illegal Function	Function code exception, usually when the accessed slave device does not have an operable function code, it will return the 01 exception code that does not exist for this function code. When the software receives this instruction, it will report this error.
Illegal Data Address	Address exception, usually when the accessed slave device does not have the register/coil address to be read, it will return an 02 exception code indicating that this address does not exist. When the software receives this instruction, it will report this error.
Illegal Data Value	Data exception, usually refers to the current data to be read/written. If the slave device does not allow the operation of data at this address, it will return a 03 exception code indicating that the data cannot be operated. When the software receives this instruction, it will report this error.
Slave Device Busy	Data exception, usually refers to the current data to be read/written. If the slave device does not allow the operation of data at this address, it will return a 03 exception code indicating that the data cannot be operated. When the software receives this instruction, it will report this error.
Response Error	Response exception is usually reported when the slave device replies with some unrecognized commands. Another issue is whether it is a bug in the software. Generally, the broadcast command (0 address) cannot read data and can only broadcast write. However, this software can be set to read with broadcast commands. When this is set, this "Response Error" exception will appear.