ODOT-S7MPI / ODOT-S7MPI Pro

User manual

V2.0.1

2021.06.16

ODOT-S7MPI / ODOT-S7MPI Pro

Odot Automation System Co., Ltd.

2017-03

Version information

The following changes have been made to the document:

Date	Version	Modification	Author
20170308	V1.0.0	Release version	CCL
20191101	V2.0.0	New features: 32 Ethernet connections.	CCL
20210616	V2.0.1	Updated features: 16 Ethernet connections; The PRO version supports MODBUS RTU; The PRO version supports TCP client function;	CCL

Disclaimer of Warranties

Product Usage

NOTE
• **When installing, operating, and maintaining the equipment, do not exceed any of the ratings specified in the electrical characteristics;** • **When installing, operating, and maintaining the equipment, do not exceed any of the ratings specified in the environmental characteristics. Do not use the product in the following places: places with dust, oil fumes, conductive dust, corrosive gases, and flammable gases; Do not expose to high temperatures, condensation, wind and rain; Vibration and shock will also cause damage to the product;** **Failure to follow the instructions may render the protection provided by the device null and may result in minor bodily injury or damage to the device.**

Disclaimer of Warranties

The Company shall not be liable for any damage or malfunction of the equipment caused by:

1. Transportation damage: equipment damage caused by improper transportation or packaging;

2. Natural factors: damage caused by lightning strikes, voltage fluctuations, water ingress or natural disasters (such as fires, floods, etc.);

3. Improper use: damage caused by overload, non-standard operation, unauthorized

modification or use of unqualified accessories;

4. Unauthorized maintenance: equipment failure caused by unauthorized maintenance or alteration;

5. Other non-product reasons: damage caused by other reasons that have nothing to do with the equipment itself.

Repair services

1. For the damage caused by the above reasons, the company will charge the repair fee according to the actual situation.

2. Outside the warranty period, the company provides paid maintenance services, and the cost is charged according to the maintenance situation.

Assumption of Risk

The company shall not be liable for casualties, property damage or other related losses caused by the use of the equipment. All risks are borne by the user.

Catalog

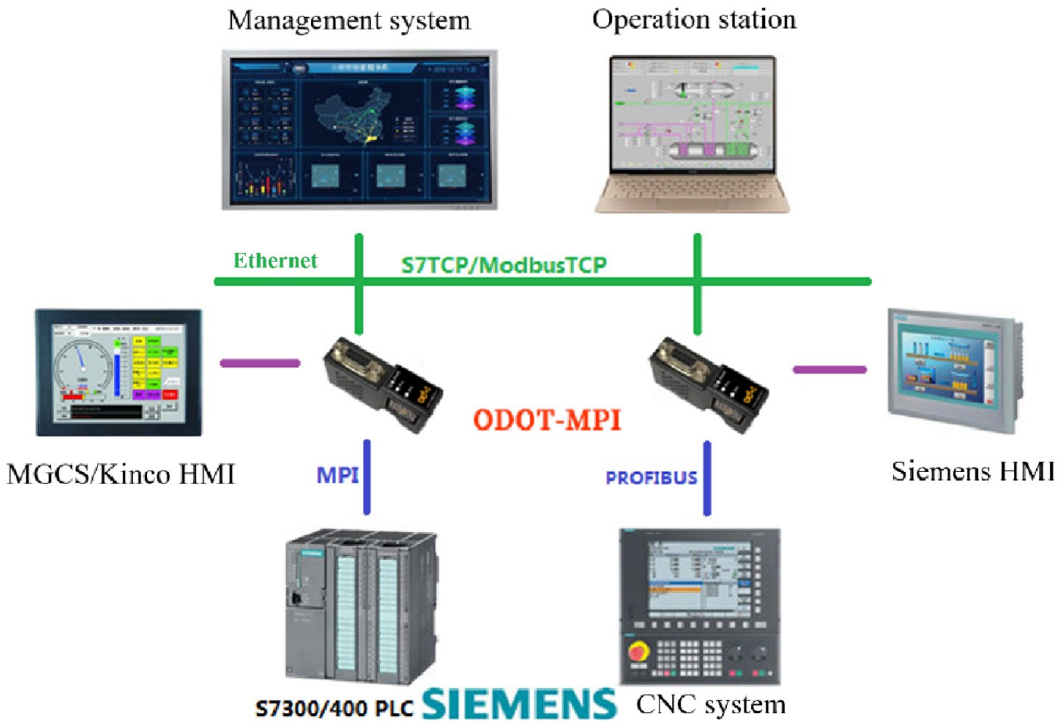
1 Product Introduction

Used for Siemens S7300/400 series PLC Ethernet communication, replacing the communication functions of Siemens CP343 (Ethernet) and CP341 (Modbus).

1.1 Application Scenarios

The industrial communication bridge (hereinafter referred to as the bridge) mainly provides hardware support for the industrial equipment networking project, helping the upper terminal to collect the data of the underlying industrial equipment (mainly PLC, but not limited to PLC equipment) through Ethernet, and then statistics and analysis of these data.

The bridge is dedicated to Siemens S7300/400 series PLCs and supports Siemens 840D and other CNC systems.



1.2 Product Selection

The bridge is divided into basic and professional versions, and the models are ODOT-S7MPI and ODOT-S7MPI Pro (divided into pass-through and bridged types).

The Pro version not only includes all the functions of the basic version, but also adds the MODBUS master-slave function of the expansion port X2 (bridged type support only) and the S7TCP and Modbus TCP client functions of the Ethernet port.

The features of the Basic and Pro versions are as follows:

Function description		ODOT-S7MPI (Basic Version).	ODOT-S7MPI PRO Pass-through (Pro).	ODOT-S7MPI PRO Bridge (Pro)
Ethernet port function	Programming and debugging (STEP7, TIA software)	√	√	√
	Monitor software	√	√	√
	Modbus TCP server	√	√	√
	S7TCP protocol	√	√	√
	Open TCP protocol	√	√	√
	S7TCP Client functions	×	√	√
	Modbus TCP Client functions	×	√	√
Expand the X2 port function	HMI interface	√	√	×
	Modbus master function	×	×	√
	Modbus slave function	×	×	√

1.3 Technical parameters

Basic parameters	Product Name	Industrial communication bridges
	Product Model	ODOT-S7MPI/ ODOT-S7MPI Pro
	Description	Siemens S7-300/400, 840D Ethernet communication
Product appearance	Housing Color	Industrial black
	Communication indicator	Pwr/Bus
	Ethernet indicator	Link/Active
	Reset button	Reset
	Size (L*W*H)	65*30*17mm
	Weight	60g
	Installation method	The communication port is pluggable
Power supply	Power supply mode	PLC communication port directly draws power/supplies DIP2 externally
	Voltage	24VDC/100mA
Communication port Com1	Interface Type	DSUB9F （RS485）
	Transfer Rate	9.6/19.2/187.5K/500K/1.5M BPS
	Communication	MPI/PROFIBUS

	Protocol	
	Supported devices	Siemens S7300/400 PLC
Communication port Com2	Interface type	DSUB9M （RS485）
	Transfer rate	9.6/19.2/187.5K/500K/1.5M BPS
	Communication protocol	MPI/PROFIBUS
	Supported devices	Siemens, MCGS, Weilun, Proface, Kinco and other human-machines
Ethernet communication port	Interface type	RJ45
	Transfer rate	10/100M
	Communication protocol	S7TCP/Modbus TCP
	Number of TCP connections	16
Host software	Programming software	STEP7/TIA
	configuration software	WinCC, IFIX, INTOUCH, LABVIEW
	OPC software	Kepware OPC, Matricon OPC
Parameter configuration	Parameter tool	NET-LINK
	WEB browser	Default IP：192.168.1.188
Working environment	Temperature	-20~85℃
	Humidity	95% (no condensation)
Certification	Electromagnetic compatibility	2014/30/EU
	CE	YES

2 Function Introduction

2.1 Function 1 - programming debugging

The bridge supports programming and debugging of PLC programs, as detailed in Chapter 3: Programming Software Communication.

2.2 Function 2 - Configuration software monitoring

The bridge supports mainstream configuration software to monitor PLC data through the S7TCP protocol or MODBUS TCP protocol, as detailed in Chapter 4: Configuration Software Communication.

2.3 Function 3: MODBUS TCP protocol

The Ethernet port of the bridge supports the MODBUS TCP protocol, which can be used by configuration software or MES systems to access PLC data, as detailed in Chapter 5: Modbus TCP Communication.

2.4 Function 4 - Open TCP protocol

The Ethernet port of the bridge supports the Open TCP protocol, which can be used by OPC software or communication software in high-level languages to access PLC data, as detailed in Chapter 6: Open TCP Protocol Specification.

2.5 Function 4 - TCP client function

The bridge (only supported by ODOT-S7MPI Pro) supports data exchange between PLCs and PLCs and between PLCs and DCS systems through S7TCP or ModbusTCP protocols, as detailed in Chapter 7: Data Exchange.

2.6 Function 5 - Modbus RTU communication

The expansion port X2 of the bridge (only supported by the ODOT-S7MPI Pro bridge type) supports Modbus RTU communication function:

Modbus RTU master function: access MODBUS slave devices (e.g., MODBUS meters) through Modbus RTU protocol;

Modbus RTU slave function: MODBUS master device (e.g., DCS system) can access PLC data through the MODBUS RTU protocol;

See Chapter 8: MODBUS RTU Communication for details.

3 Hardware installation and commissioning

3.1 Introduction to communication interfaces

1. Bridge X1 communication port: DSUB9 male, connected to Siemens PLC communication port;

2. Bridge X2 communication port: DSUB9 female, communication port connected to touch screen;

3. Ethernet communication port: RJ45 socket, network port connected to the computer or LAN network;

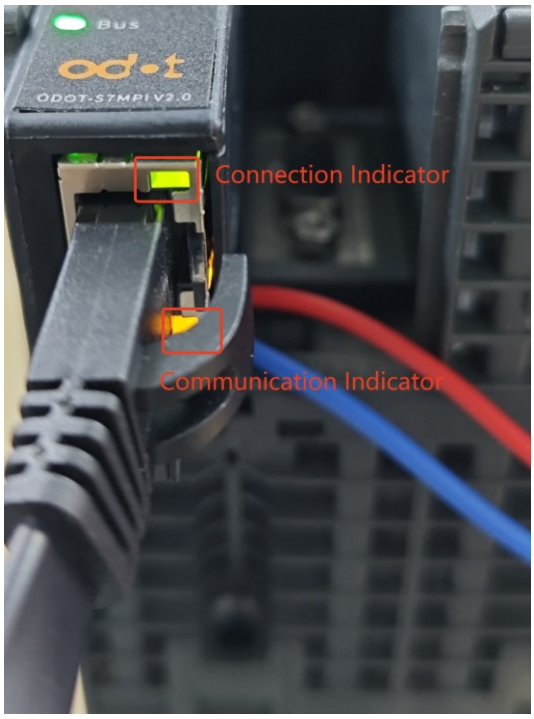
4. Power supply interface: DC 24V, if there is no power supply or insufficient power supply at the PLC communication port, provide a backup power interface;

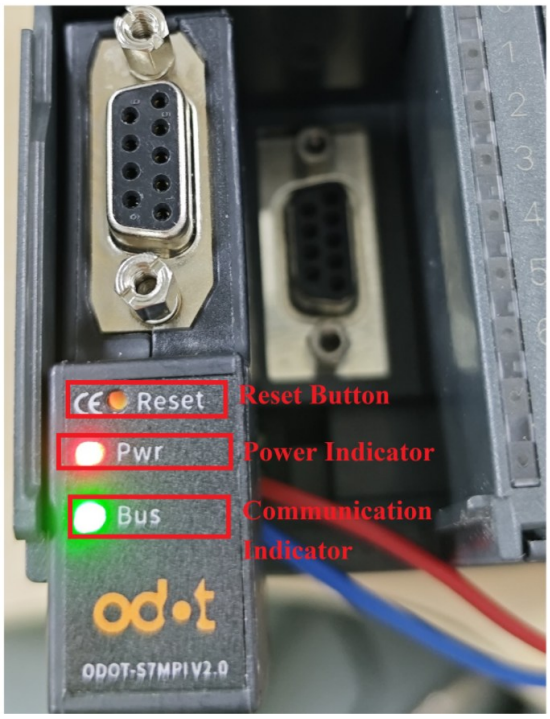
3.1 Indicator introduction

1. The position of the bridge reset button;

2. Serial port communication status light position;

3. The position of the network port communication status light;





3.3 Installation

1. The user can plug the bridge directly into the MPI or DP communication port of the PLC;

2. The bridge is directly powered by the Siemens PLC (except for some Siemens CNC systems) communication port, without the need to connect to an external 24V power supply;

3. If the touch screen is originally inserted on the communication port of the PLC, the touch screen can be unplugged first, and then the touch screen is inserted on the X2 expansion port of the bridge after the bridge is plugged in.

4. After tightening the mounting bolts on both sides, connect the network port of the bridge to the network port or LAN network of the computer through the network cable.

3.4 Debugging

1. After the device is powered on, the power indicator light (PWR) of the bridge remains on;

2. Bridge communication indicator (Bus), status:

Bus communication light status	Function description	Solution
Flashes at intervals of 1 second	The bridge does not establish a normal connection with the PLC	1. It is in the process of baud rate adaptation. Wait about 3 seconds; 2. Check whether the bridge communication protocol mode is compatible with the device through the parameter tool; 3. Siemens S7200PLC detect whether the communication of the PLC has been modified to a free-port communication module; 4. The DP port of Siemens S7300 and S7400 series PLCs is not open by default, and the communication port parameters need to be configured; 5. Check whether the communication port of the PLC or bridge is damaged.

ON	The bridge has established a normal connection with the PLC and there is currently no communication request	
Fast flash	The module communicates normally, and the indicator light flashes according to the data transmission and reception speed	

3. After the network port connection light (Link) of the bridge is plugged into the network cable, the green light is always on; the communication light flashes green;

4. Configure and diagnose the parameters of the module through the WEB page or parameter setting tool.

4 Parameter settings

The bridge can be parameterized by logging in to the WEB page or the parameter setting tool.

In order to meet the user's product export needs, the bridge's WEB web interface supports Chinese and English languages;

In order to make it easier for users to manage field devices in practical applications, the bridge opens the [Device Name] parameter, and users can name the field devices connected to the bridge.

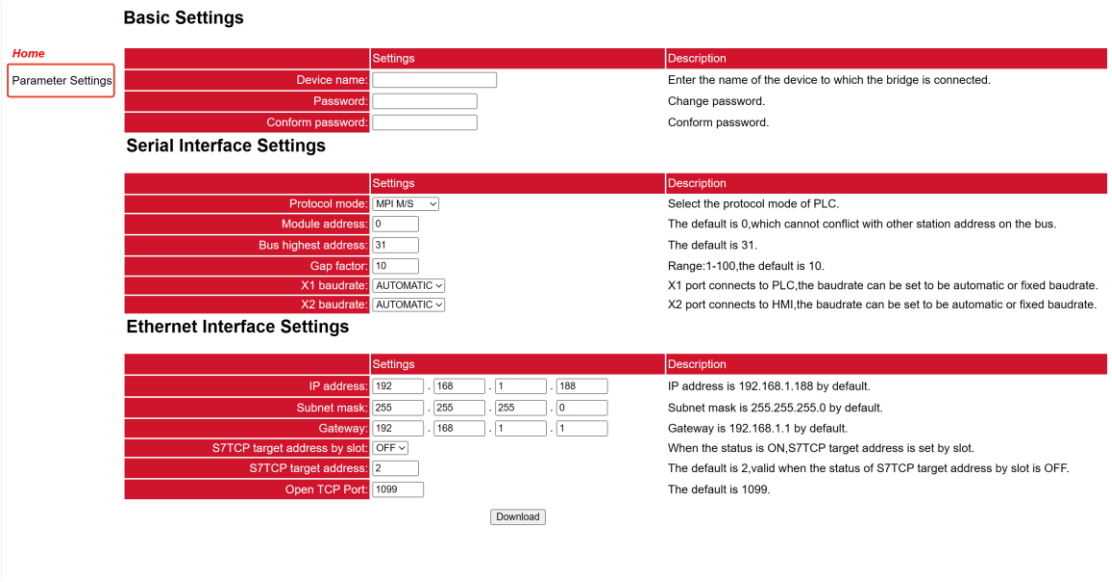
4.1 Set the parameters through the WEB page

Before setting the bridge parameters through the web page, it needs to set the IP address of the computer to the same network segment as the IP address of the bridge (the IP address at the time of factory is 192.168.1.188).

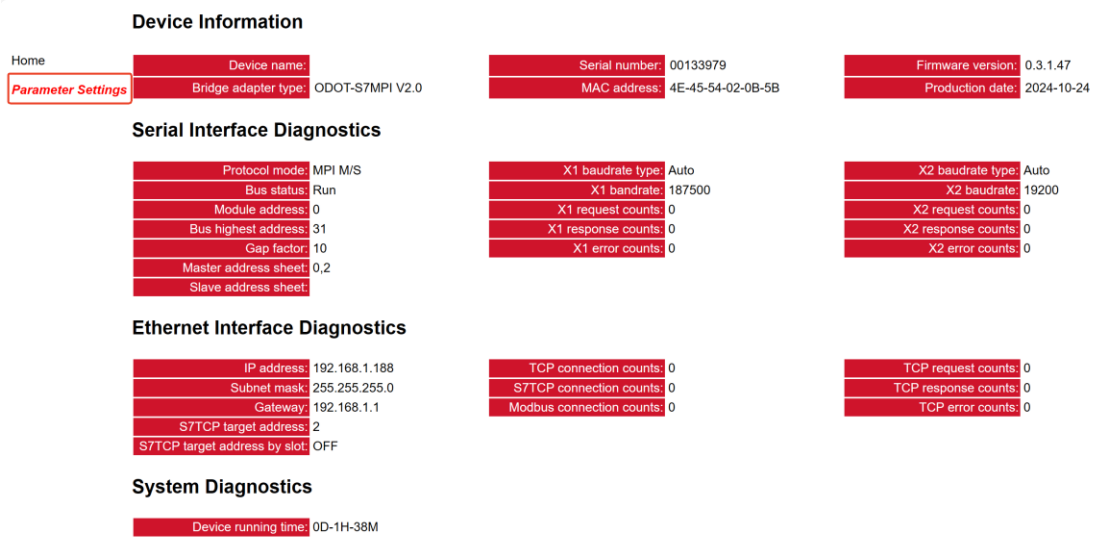
After setting the IP address of the computer correctly, open the browser (take IE browser as an example), enter the IP address of the bridge in the address bar of the browser: 192.168.1.188 (take the IP at the factory as an example), click the enter key, and it can

open the login interface of the bridge:

Select the display interface in English **through** Language, and enter the correct login password, and click the Login button to open the bridge's homepage:



It can see the operation of the bridge on the homepage, click the parameter settings in the left navigation bar to enter the parameter settings page:



The parameters are described as follows:

Device Name: it can name the field equipment connected by the bridge, for example: No. 1 air compressor, or not set;

Password and Confirmation password: To change the login password, you must ensure that the characters entered twice are the same, if it is inconsistent or not set, the login password is the default login password: admin;

Protocol mode: It can be set as MPI master-slave station (when the bridge is plugged into the MPI port of the PLC) or PROFIBUS (when the bridge is plugged into the DP port of the PLC);

Module address: the station address of the bridge is set, which does not conflict with the station address of other devices on the bus;

Bus highest address: Set the highest station address of the bus to identify the device, it is recommended to default;

Gap factor: Recommended default;

X1 baudrate: it can be set to automatic, automatic status: the bridge automatically recognizes the baud rate of PLC communication, and can also manually set the baud rate according to the actual PLC baud rate;

X2 baudrate: it can be set to automatic, automatic status: the bridge automatically recognizes the baud rate of the touch screen communication, and can also be manually set according to the baud rate of the actual touch screen, this parameter is only

meaningful for the bridge type;

IP address: set the IP address of the bridge;

Subnet mask: Set the subnet mask of the bridge;

Gateway: The gateway for setting the bridge;

S7TCP target address by slot: Enabled state: S7TCP destination address is determined by slot number.

S7TCP target address: This parameter works only when the S7TCP destination address slot number is determined to be closed.

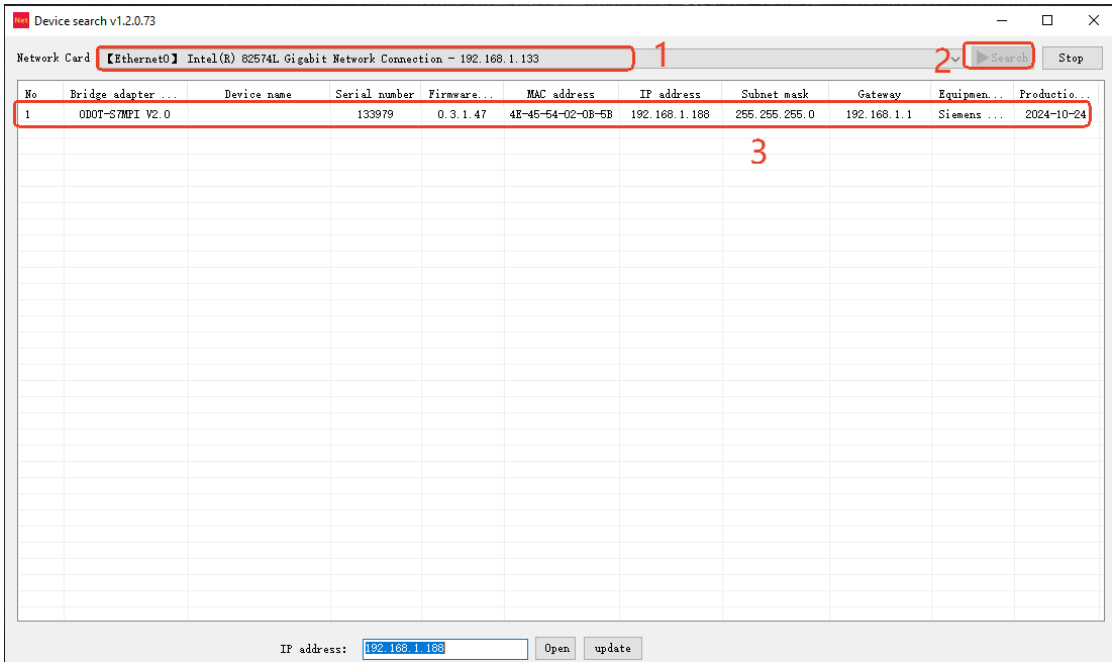
Open TCP port: default is 1099, it is recommended to default;

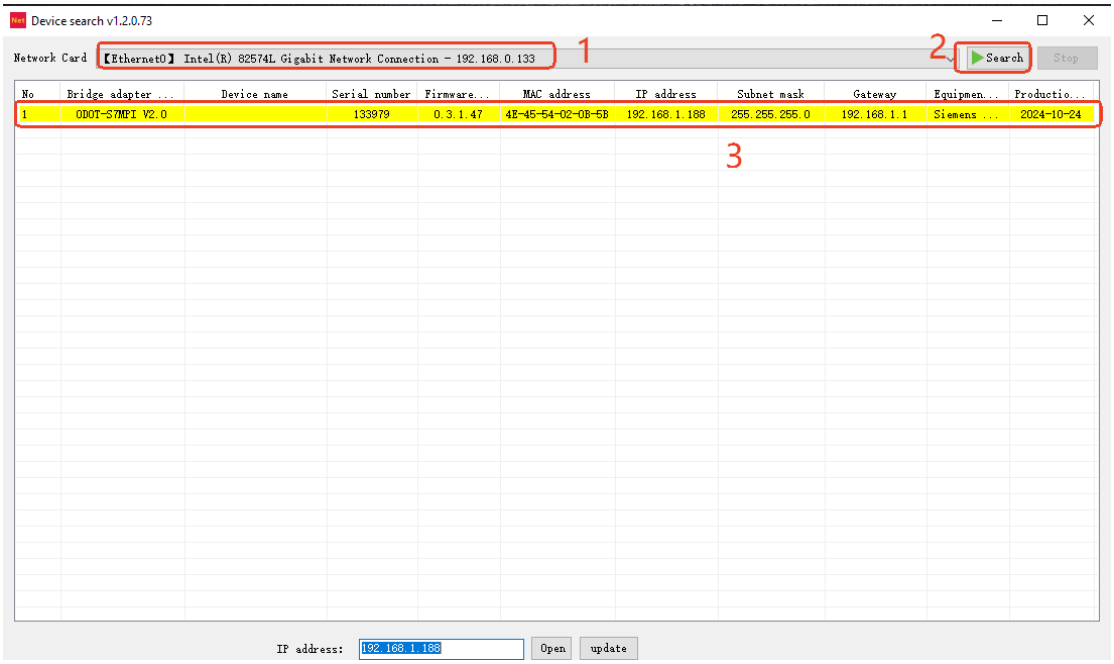
After modifying the parameters, click the download button.

4.2 Set parameters through the NET-Link

4.2.1 Search for devices

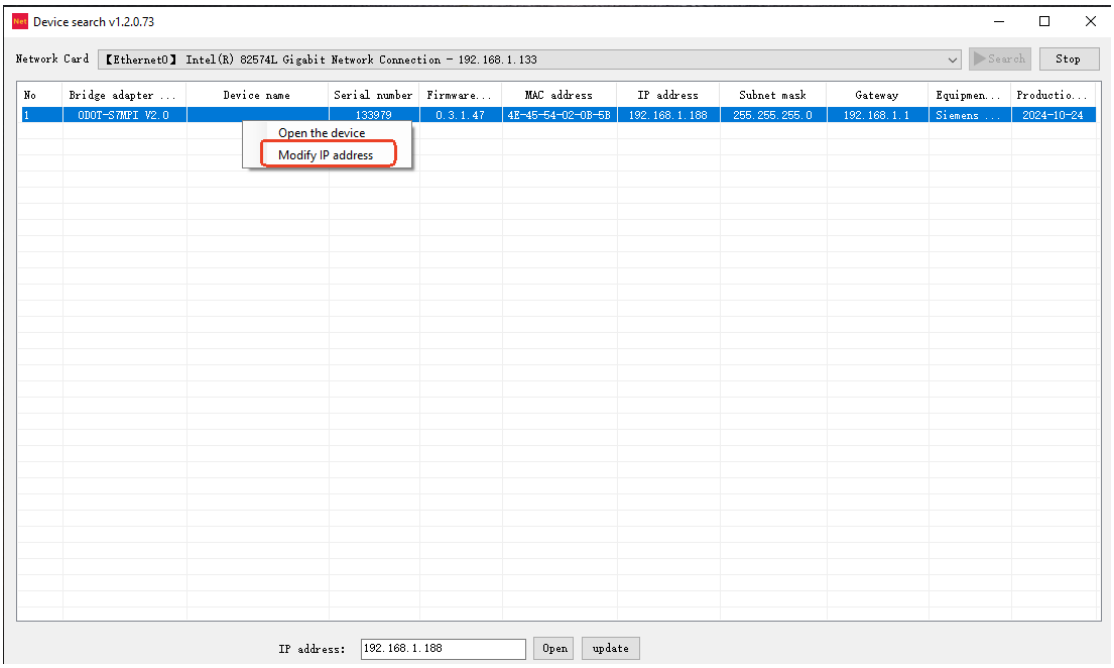
Open the parameter configuration tool NET-Link, select the network card connected to the computer and the bridge, and click the search button to search for the bridge. Cross-CIDR search is supported, and the cross-CIDR segment search module is displayed in yellow.

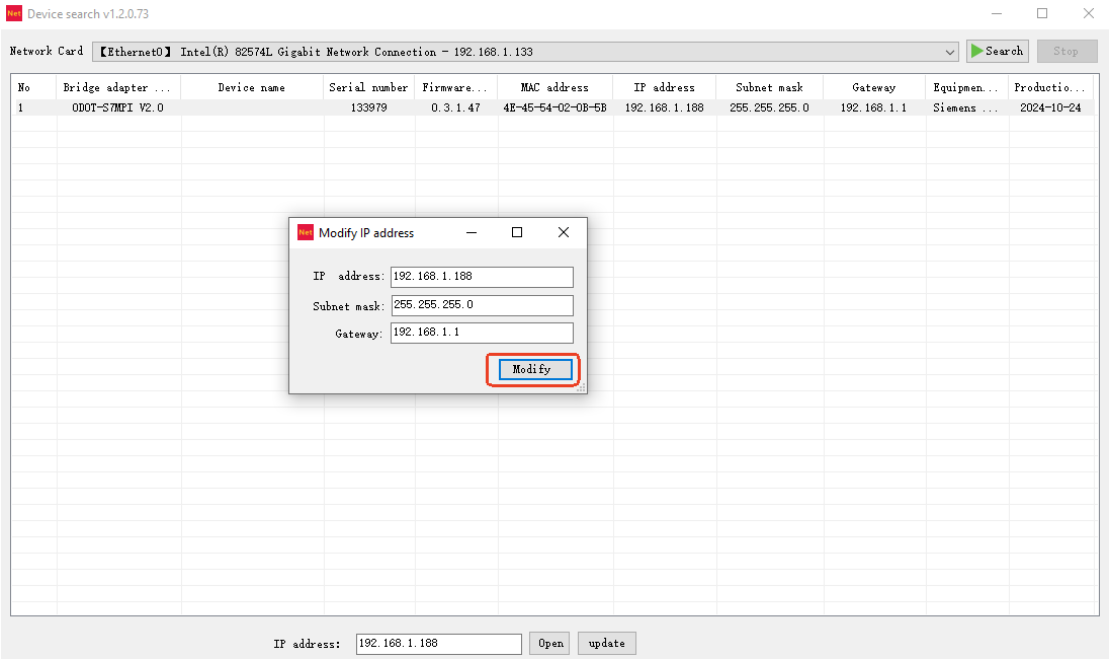




4.2.2 Modify the IP address

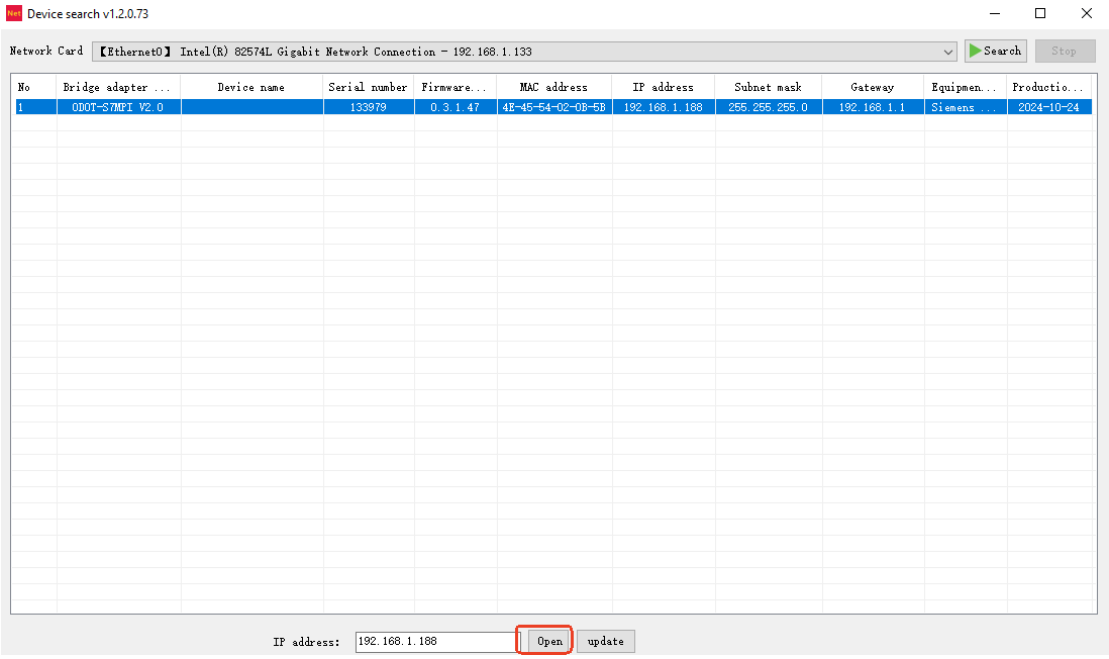
If want to modify the IP address, subnet mask, and gateway parameters of the bridge, you can select the bridge, right-click the mouse, select Modify IP address, enter the IP address, subnet mask, and gateway want to modify in the pop-up dialog box, and click the Modify button.





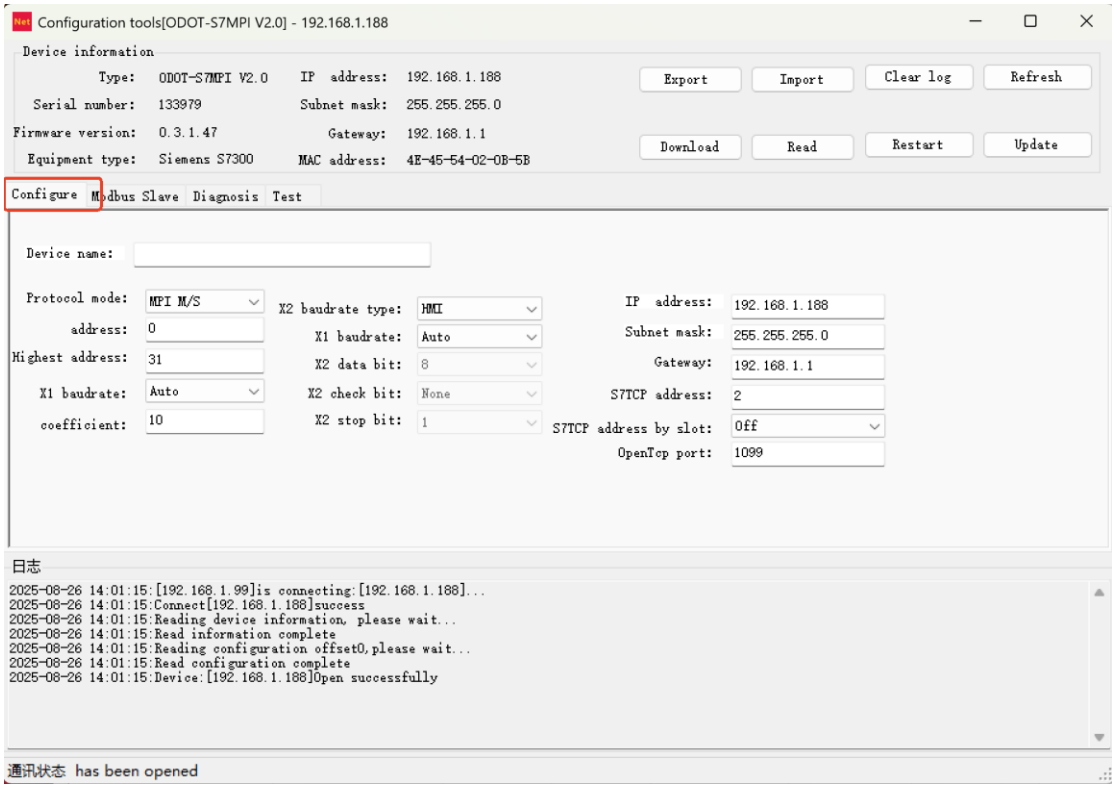
4.2.3 Open device

Select the bridge, right-click the mouse, and select Open Device to enter the bridge's parameter settings, diagnosis, and test pages.



4.2.4 Configure interface

Open the communication interface as follows:



The parameters are described as follows:

Device Name: it can name the field equipment connected by the bridge, for example: No. 1 air compressor, or not set;

Password and Confirmation password: To change the login password, you must ensure that the characters entered twice are the same, if it is inconsistent or not set, the login password is the default login password: admin;

Protocol mode: It can be set as MPI master-slave station (when the bridge is plugged into the MPI port of the PLC) or PROFIBUS (when the bridge is plugged into the DP port of the PLC);

Module address: the station address of the bridge is set, which does not conflict with the station address of other devices on the bus;

Bus highest address: Set the highest station address of the bus to identify the device, it is recommended to default;

Gap factor: Recommended default;

X1 baudrate: it can be set to automatic, automatic status: the bridge automatically recognizes the baud rate of PLC communication, and can also manually set the baud rate according to the actual PLC baud rate;

X2 baudrate: it can be set to automatic, automatic status: the bridge automatically recognizes the baud rate of the touch screen communication, and can also be manually set according to the baud rate of the actual touch screen, this parameter is only meaningful for the bridge type;

IP address: set the IP address of the bridge;

Subnet mask: Set the subnet mask of the bridge;

Gateway: The gateway for setting the bridge;

S7TCP target address by slot: Enabled state: S7TCP destination address is determined by slot number.

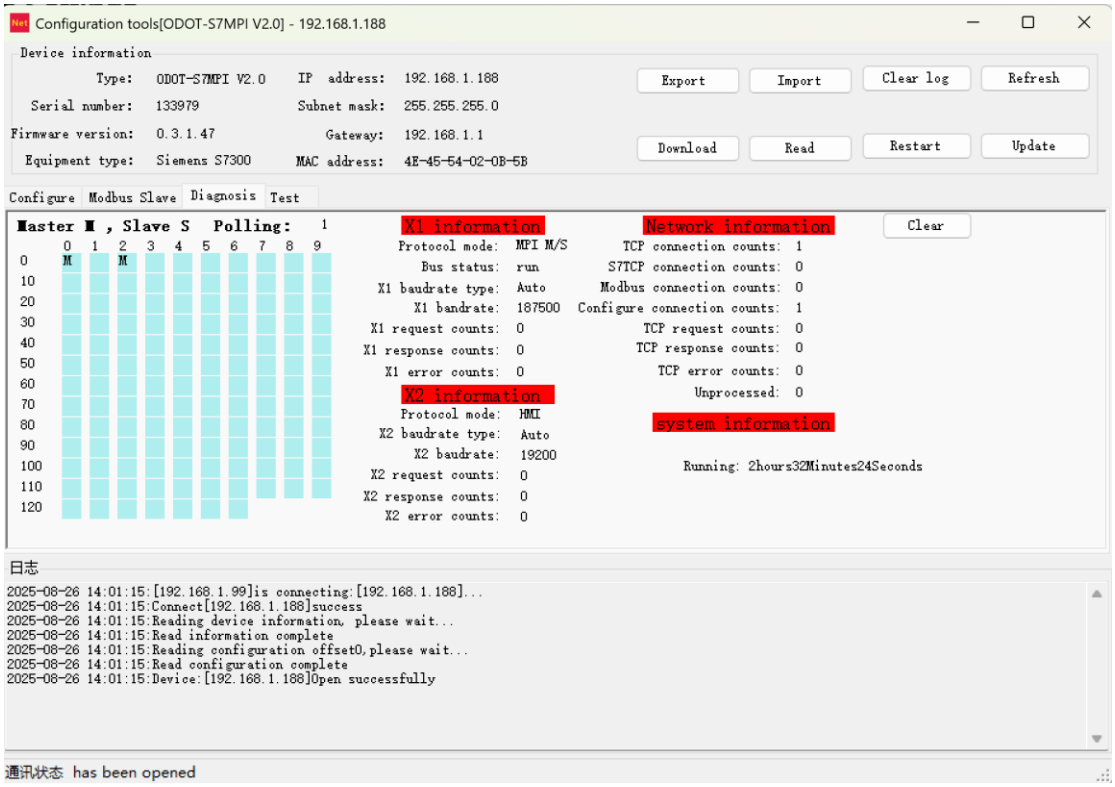
S7TCP target address: This parameter works only when the S7TCP destination address slot number is determined to be closed.

Open TCP port: default is 1099, it is recommended to default;

After modifying the parameters, click the download button.

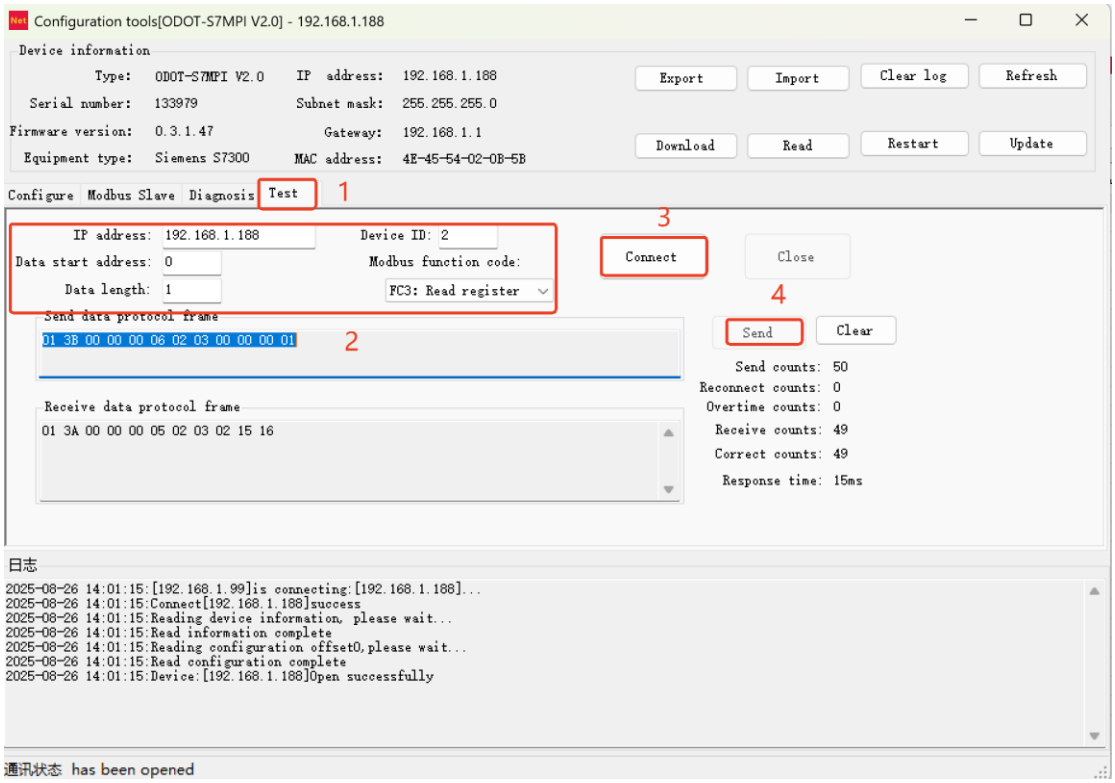
4.2.5 Diagnosis Interface

Select the Diagnosis page to open the diagnostics screen to view the operation of the bridge:



4.2.6 Test interface

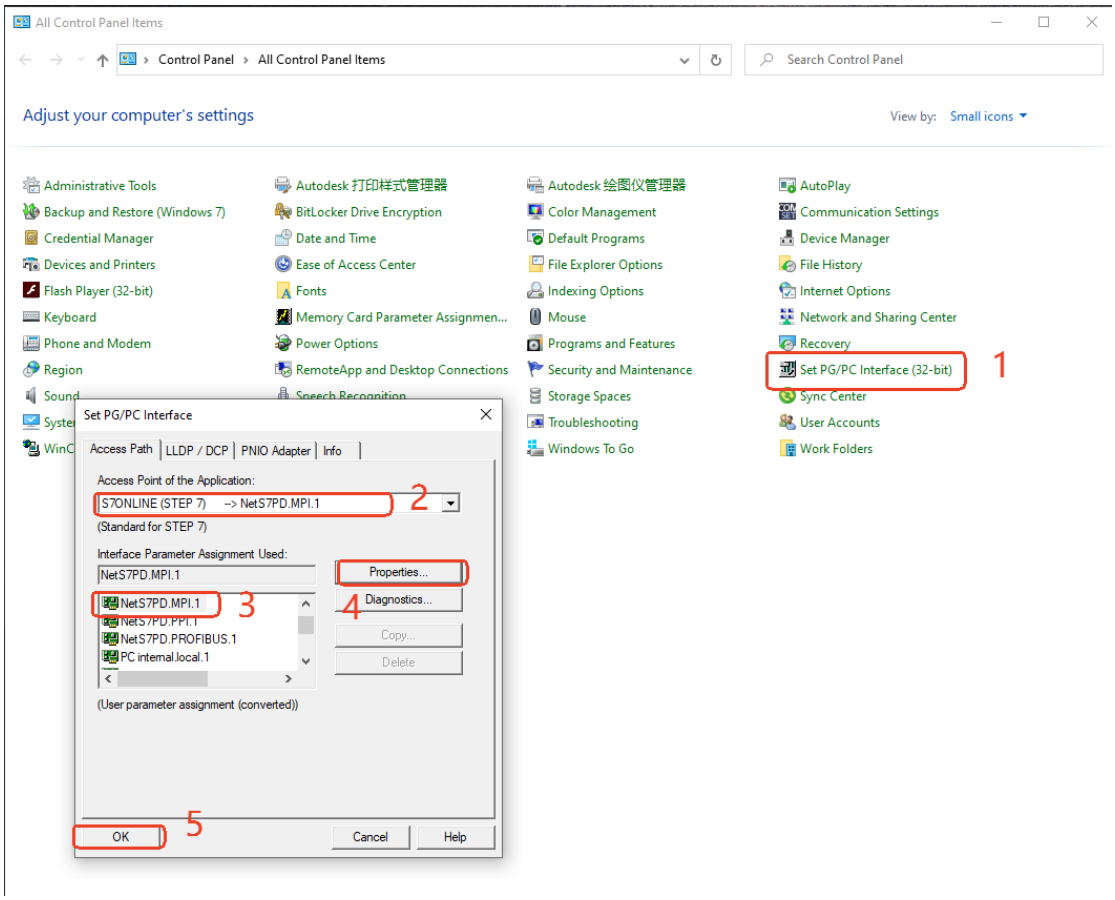
Select the test options page, open the test interface, set the communication parameters, and click the connect ----send button to test the Modbus communication between the bridge and the PLC.


Configuration tools[ODOT-S7MPI V2.0] - 192.168.1.188
Device information
Type: ODOT-S7MPI V2.0
IP address: 192.168.1.188
Serial number: 133979
Subnet mask: 255.255.255.0
Firmware version: 0.3.1.47
Gateway: 192.168.1.1
Equipment type: Siemens S7300
MAC address: 4E-45-54-02-0B-5B
Export
Import
Clear log
Refresh
Download
Read
Restart
Update
Configure
Modbus Slave
Diagnosis
Test
1
IP address: 192.168.1.188
Device ID: 2
Data start address: 0
Modbus function code:
Data length: 1
FC3: Read register
3
Connect
Close
4
Send data protocol frame
01 3B 00 00 00 06 02 03 00 00 00 01
2
Send
Clear
Send counts: 50
Reconnect counts: 0
Overtime counts: 0
Receive counts: 49
Correct counts: 49
Response time: 15ms
Receive data protocol frame
01 3A 00 00 00 05 02 03 02 15 16
日志
2025-08-26 14:01:15:[192.168.1.99]is connecting:[192.168.1.188]...
2025-08-26 14:01:15:Connect[192.168.1.188]success
2025-08-26 14:01:15:Reading device information, please wait...
2025-08-26 14:01:15:Read information complete
2025-08-26 14:01:15:Reading configuration offset0,please wait...
2025-08-26 14:01:15:Read configuration complete
2025-08-26 14:01:15:Device:[192.168.1.188]Open successfully
通讯状态 has been opened

5 Programming Software Communication

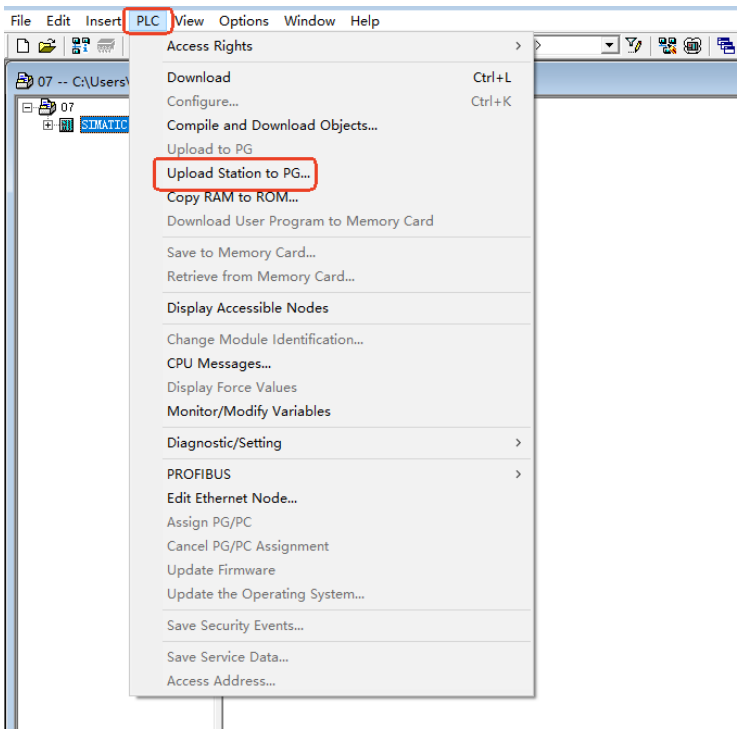
5.1 Set the PG/PC interface

Open the PG/PC interface in the control panel, select the application access point S7ONLINE (STEP7) to point to the programming driver, click the property button, enter the IP address of the bridge in the pop-up dialog box, and click the OK button.

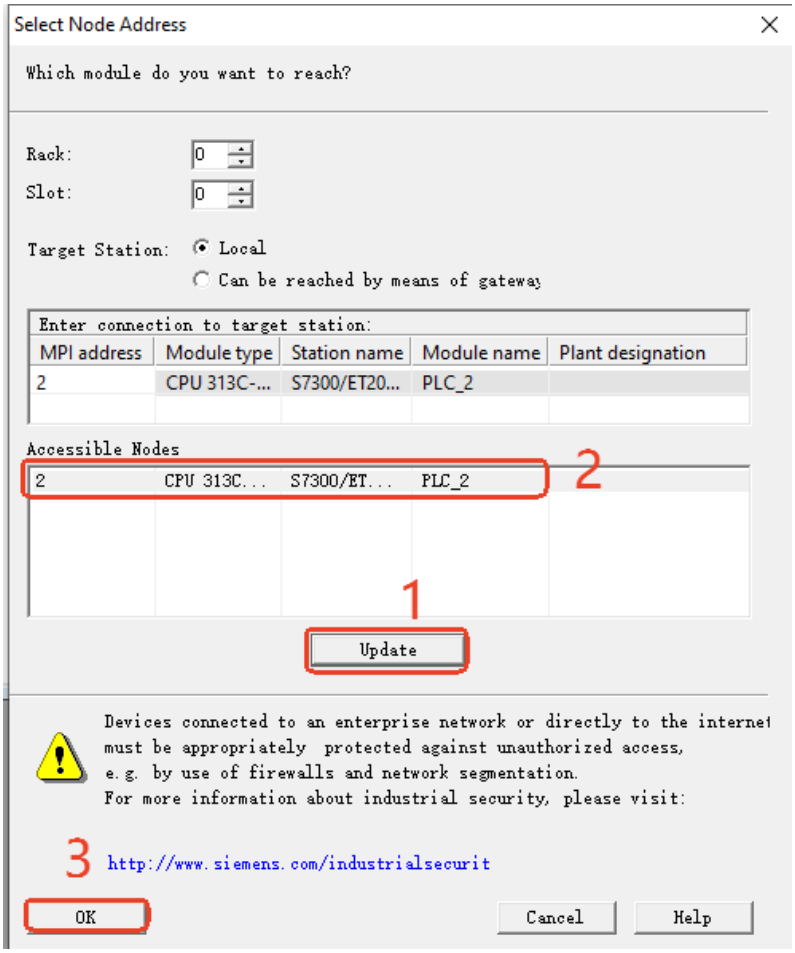


5.2 Step7 Programming

Open Siemens programming software Step 7, create a new project, select PLC in the menu bar - upload the site to PG;

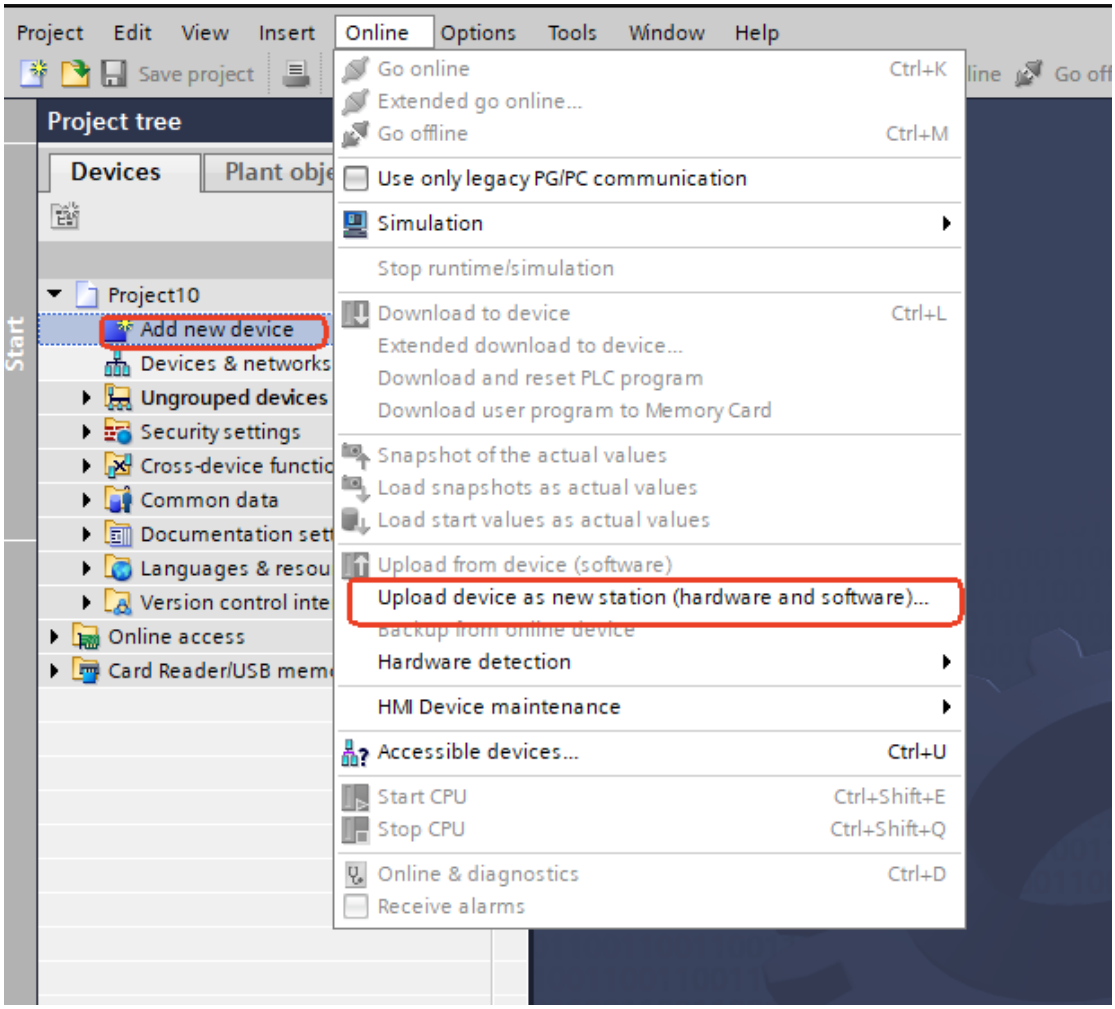


In the pop-up dialog box, click the view button, it can search for the online PLC device, after selecting the device, click the OK button to upload the PLC program.

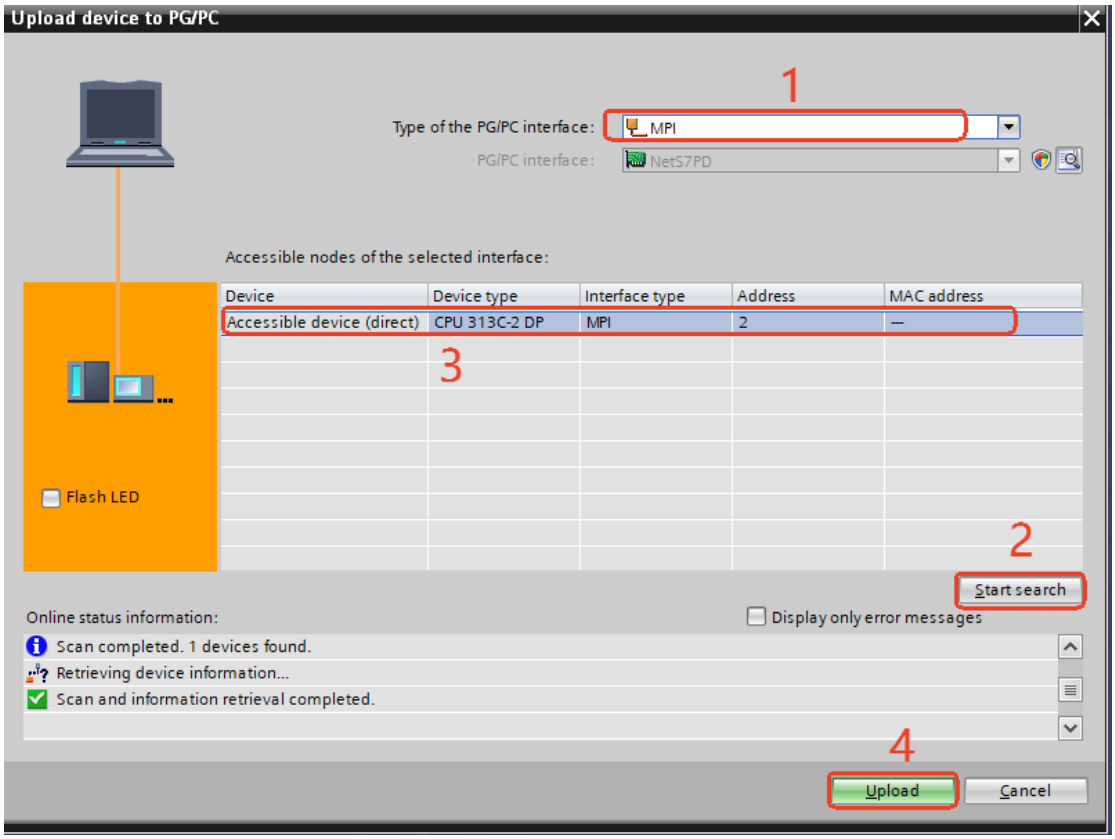


5.3 TIA Portal Programming

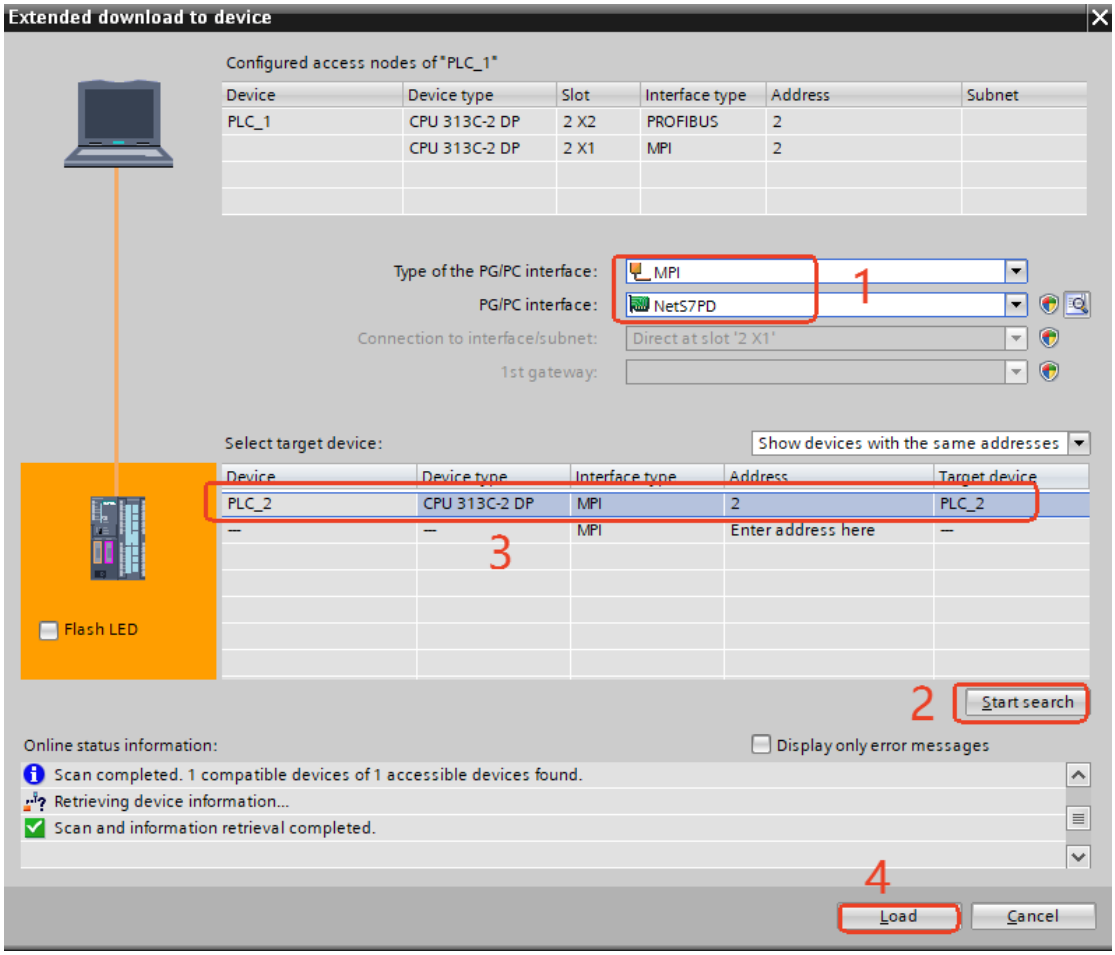
Open the TIA software, create a new project, click Online in the menu bar - upload the device as a new station (hardware and software);



In the pop-up dialog box, select MPI as the type of PG/PC interface; PG/PC interface selection programming driver; Click the start search button to search for the online PLC device, select the device, click the upload from device button to upload the PLC program;



Click the download button in the menu bar, and in the pop-up dialog box, select MPI as the type of PG/PC interface. PG/PC interface selection programming driver; Click the Start Search button to search for online PLC devices, select the device, and click the Go to Online button to download the PLC program.


Extended download to device
Configured access nodes of "PLC_1"
Device
Device type
Slot
Interface type
Address
Subnet
PLC_1
CPU 313C-2 DP
2 X2
PROFIBUS
2
CPU 313C-2 DP
2 X1
MPI
2
Type of the PG/PC interface:
MPI
1
PG/PC interface:
NetS7PD
Connection to interface/subnet:
Direct at slot '2 X1'
1st gateway:
Select target device:
Show devices with the same addresses
Device
Device type
Interface type
Address
Target device
PLC_2
CPU 313C-2 DP
MPI
2
PLC_2

MPI
Enter address here

3
Flash LED
2
Start search
Online status information:
Display only error messages
Scan completed. 1 compatible devices of 1 accessible devices found.
Retrieving device information...
Scan and information retrieval completed.
4
Load
Cancel

6 Modbus TCP Communication

6.1 Modbus Mapping Table

PLC internal register address and MODBUS address correspondence table:

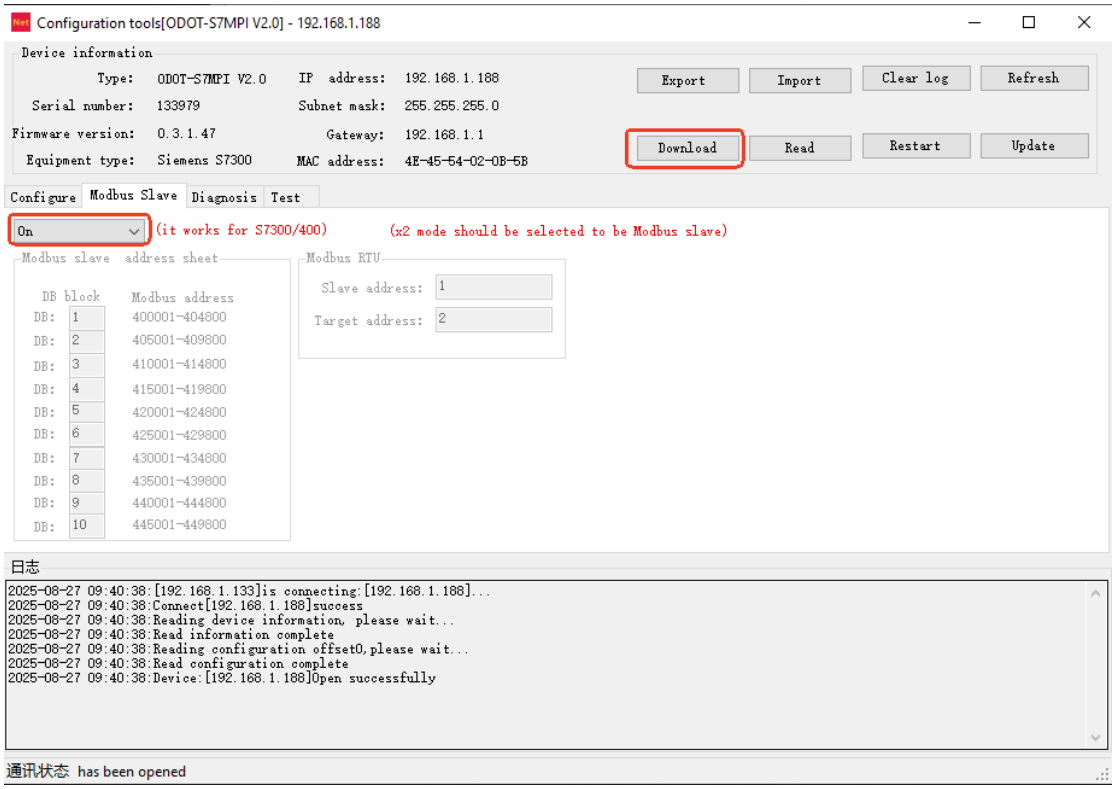
Modbus Address	PLC Interior Register Address	Data Type	Calculation Formula	MODBUS function co	Maximum number of instructions
000001~	Q0.0~	Bit	Qm.n = 000001 + m*8 + n ①	FC1 (Read the coil) FC5 (Write coil)	S7-200：119 S7-300：784
050001~	M0.0~		Mm.n = 050001 + m*8 + n		FC5：1
100001~	I0.0~		Im.n = 100001 + m*8 + n	FC2 (Read Input)	S7-200：119 S7-300：784
400001~	DBx.DBW0~	Word	DB1.DBWm = 400001 + m/2, m is an even number ②	FC3 (Read Multiplication Register) FC16 (write multiplication register) FC6 (Write a single multiplication register)	FC3：111 FC16：111 FC6：1
450001~	MW0~		MWm= 450001 + m/2, m is an even number		

Notes:

①: For example, Q1.5 corresponds to the Modbus address: Q1.5=000001+8*1+5 =000014;

②: For example, DB1.DBW100, its corresponding Modbus address is: DB1.DBW100=400001+100/2=400051;

By default, 400001-450000 in the Modbus address area corresponds to the DB1 data block of Siemens 300PLC, and you can also freely specify multiple DB data blocks to modify through the NET-Link tool:



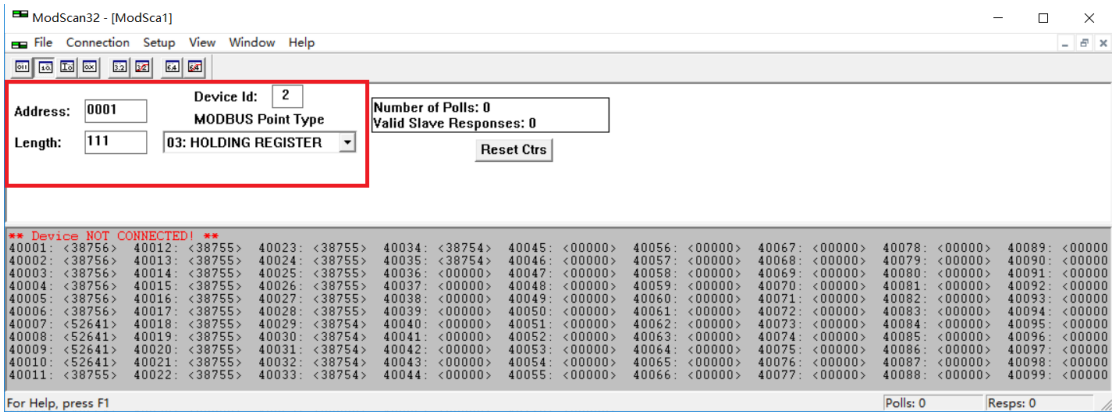
Open the Modbus slave mapping table and specify the correspondence between multiple DB block numbers and Modbus address areas.

By default, for example, DB2.DBW100, the corresponding Modbus address is: DB2.DBW100=405001+100/2=405051. It can modify the DB block number according to needs.

6.2 Modbus Communication Test

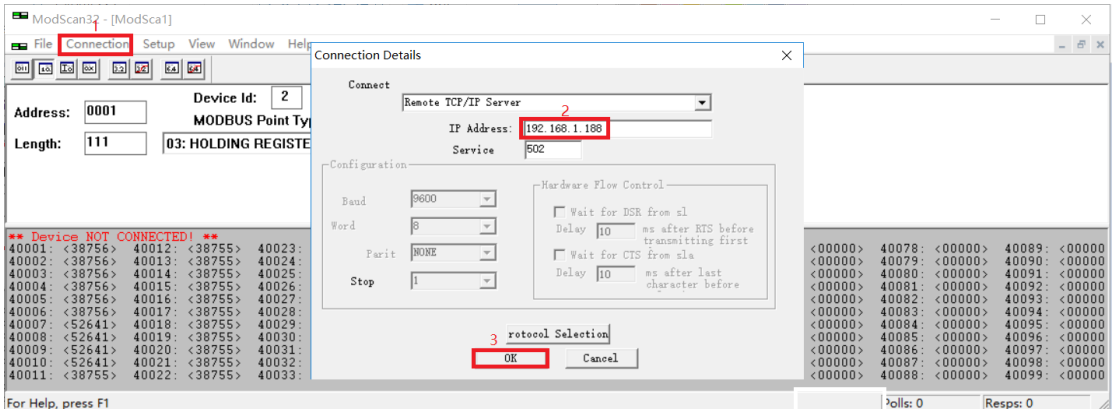
Tested with the ModScan32 tool

1. Open the software and set the data to be tested, for example: read the 111 words at the beginning of the 400001, and set it as follows:

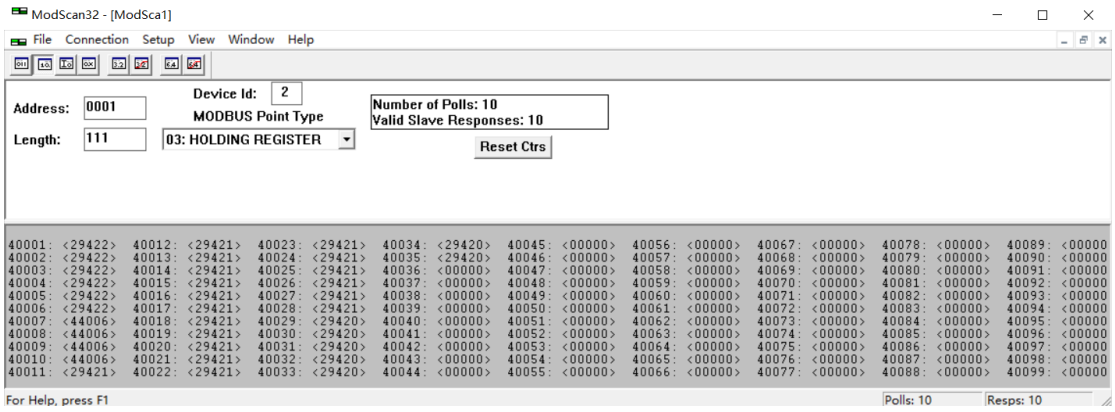


2. Select Connection--Connect in the menu bar, enter the IP address of the bridge in the

IP Address in the dialog box, and click the OK button.



3. The normal picture of the test is as follows:



7 OpenTCP Protocol Specification

OpenTCP protocol Ethernet communication packets consist of a fixed 8-byte header, an 8-byte extended header, and optional up to 200 bytes of user data, which follows this structure whether sending or receiving packets. The table is as follows:

Section	Byte	Parameter	Type	Description
8 bytes header	0	msg.rx	byte	Receiver Identification ID
	1	msg.tx	byte	Sender Identification ID
	2	msg.ln	byte	Expand the total length of the packet header and user data
	3	msg.nr	byte	Message ID
	4	msg.a	byte	Response number
	5	msg.f	byte	Error number
	6	msg.b	byte	Command number
	7	msg.e	byte	Extension number
8 Bytes of extended headers	8	msg.device_adr	byte	Remote (PLC) station
	9	msg.data_area	byte	Data area
	10,11	msg.data_adr	word	Data address
	12	msg.data_idx	byte	Data index number
	13	msg.data_cnt	byte	The number of bytes of
	14	msg.data_type	byte	Data type
	15	msg.function	byte	Function number
User data	16~215	msg.d[0~199]	byte array	User data up to 200 bytes

Thereinto:

For the client (computer), the identification ID is 0xFF (decimal number 255), and the identification ID for the server (bridge) is 0x03 (decimal number 3); Therefore:

The client sends a data command frame to the server: msg.rx=0x03, msg.tx=0xFF;

The server sends a data response frame to the client: msg.rx=0xFF, msg.tx=0x03;

The client should check the msg.rx and msg.tx of the received packet to determine whether it is the response packet of the bridge.

The total length of the extended header and user data area msg.ln is the sum of the bytes of the extended header and user data, therefore:

The client sends the read data command frame to the server: msg.ln=0x08; No user data;

The client sends the write data command frame to the server: msg.ln=0x08 + byte length of the data to be written;

The server sends the read data response frame to the client: msg.ln=0x08+return data byte length;

The server sends a write data response frame to the client: msg.ln=0x08; No user data;

The client should judge the integrity of the packet based on the msg.ln of the received packet.

Packet IDs msg.nr identify the corresponding information for each pair of sent/received packets. In order to receive the correct reply packet, the client should automatically increase the msg.nr by 1 before each message is sent, and then judge whether the msg.nr of the received packet is consistent with the msg.nr of the sent packet.

The response number msg.a is 0x00 in the client sent packet, and the command number msg.b should be sent by the server. When the client receives the message data, it should judge whether the receiving message msg.a is equal to the sending message's msg.b, and then process the data if it is consistent.

The error number msg.f is 0x00 in the client sent message, and if msg.f=0x00 indicates that the client's request is correctly processed by the server. The client should check the msg.f of the received packet, and if it is not 0, it should retry or check the send command.

The command number msg.b is the specified command code (described below) in the client sent packet and 0x00 in the server sent packet.

The extension msg.e is always 0x00;

The definition of the 8-byte expansion header is described in detail in the subsequent command message of the document.

The length of the user data area is 0 when the client sends the read data command, that is, there is no user data area. Store the data to be written when the client sends the write data command. Store the read data when the server sends a read data response frame; When the server sends a write data response frame, the length is 0, that is, there is no user data area.

7.1 Read DB block data

Note: For S7-200, the V region corresponds to the DB1 data block;

The client sends a command to read data:

	Byte	Parameter	Type	Description
8 bytes header	0	msg.rx	byte	0x03
	1	msg.tx	byte	0xFF
	2	msg.ln	byte	0x08
	3	msg.nr	byte	Client given
	4	msg.a	byte	0x00
	5	msg.f	byte	0x00
	6	msg.b	byte	0x31 (read and write DB blocks)
	7	msg.e	byte	0x00
8 bytes of extended headers	8	msg.device_adr	byte	Remote (PLC) station address 0-31
	9	msg.data_area	byte	Read the higher 8-bit value of the start byte address, = start address/256
	10,11	msg.data_adr	word	DB block number, 0~65534; The V area of S7-200 is 0x0001 (DB1)
	12	msg.data_idx	byte	Read the lower 8 bits of the start byte address, = start address %256
	13	msg.data_cnt	byte	The maximum number of bytes of data to be read is 200
	14	msg.data_type	byte	0x05 (bytes)
	15	msg.function	byte	0x01 (Read Data)

The server sends a read data response frame:

	Byte	Parameter	Type	Description
8 bytes header	0	msg.rx	byte	0xFF
	1	msg.tx	byte	0x03
	2	msg.ln	byte	0x08 + byte count of data read
	3	msg.nr	byte	Consistent with client specifications
	4	msg.a	byte	0x31 (Read and write DB blocks)
	5	msg.f	byte	0x00 (Non-0 indicates error)
	6	msg.b	byte	0x00
	7	msg.e	byte	0x00
8 bytes	8	msg.device_adr	byte	Remote (PLC) station addresses 0-

of extended headers				31
	9	msg.data_area	byte	Read the high 8-bit value of the starting byte address, = start address / 256
	10,11	msg.data_adr	word	DB block number, 0~65534; The V region of S7-200 is 0x0001 (DB1)
	12	msg.data_idx	byte	Read the lower 8-bit value of the start byte address, = start address %256
	13	msg.data_cnt	byte	The number of bytes of data read is less than or equal to 200
	14	msg.data_type	byte	0x05 (byte)
	15	msg.function	byte	0x01 (Read Data)
User data (up to 200 bytes)	16~ 16+ (byte of data read - 1)	msg.d[0~(Read data bytes - 1)]	byte array	Read data

For example, a client reads DB1 of S7-300 (station address 2). DBB100~DBB119 has a total of 20 bytes.

Client sends (hexadecimal):

03	FF	08	01	00	00	31	00	02	00	00	01	64	14	05	01

Server sends (hexadecimal):

FF	03	1C	01	31	00	00	00	02	00	00	01	64	14	05	01
00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
00	00	00	00												

The green data is a total of 20 bytes of DB1.DBB100~DBB119 read.

The red data is the starting address DB1.DBB100 (0x0064);

7.2 Write DB block data

Note: For S7-200, the V region corresponds to the DB1 data block;

The client sends a write data command:

	Byte	Parameter	Type	Description
8 bytes header	0	msg.rx	byte	0x03
	1	msg.tx	byte	0xFF
	2	msg.ln	byte	0x08 + number of bytes of written data
	3	msg.nr	byte	Client given
	4	msg.a	byte	0x00

	5	msg.f	byte	0x00
	6	msg.b	byte	0x31 (read and write DB blocks)
	7	msg.e	byte	0x00
8 bytes of extended headers	8	msg.device_adr	byte	Remote (PLC) station address 0-31
	9	msg.data_area	byte	Write the higher 8-bit value of the start byte address, = start address/256
	10,11	msg.data_adr	word	DB block number, 0~65534; The V area of S7-200 is 0x0001 (DB1)
	12	msg.data_idx	byte	Write the lower 8-bit value of the starting byte address, = %256 of the start address
	13	msg.data_cnt	byte	The maximum number of bytes of data to be written is 200
	14	msg.data_type	byte	0x05 (bytes)
	15	msg.function	byte	0x02 (write data)
User data (up to 200 bytes)	16~ 16+ (Bytes of data written - 1)	msg.d[0~(Number of bytes written to data-1)]	byte array	data written

The server sends a write data response frame:

	Byte	Parameter	Type	Description
8 bytes header	0	msg.rx	byte	0xFF
	1	msg.tx	byte	0x03
	2	msg.ln	byte	0x08
	3	msg.nr	byte	Consistent with the client given
	4	msg.a	byte	0x31 (read and write DB blocks)
	5	msg.f	byte	0x00 (non-0 means there is an error)
	6	msg.b	byte	0x00
	7	msg.e	byte	0x00
8 bytes of extended headers	8	msg.device_adr	byte	Remote (PLC) station address 0-31
	9	msg.data_area	byte	Write the higher 8-bit value of the start byte address, = start

				address/256
	10,11	msg.data_adr	word	DB block number, 0~65534; The V area of S7-200 is 0x0001 (DB1)
	12	msg.data_idx	byte	Write the lower 8-bit value of the starting byte address, = %256 of the start address
	13	msg.data_cnt	byte	The number of bytes of data that has been written is less than or equal to 200
	14	msg.data_type	byte	0x05 (bytes)
	15	msg.function	byte	0x02 (write data)

For example, a client writes data 0x01020304 to DB1.DBD1000 of S7-300 (station address 2), which is 4 bytes

Client send (hexadecimal):

03	FF	0C	01	00	00	31	00	02	03	00	01	E8	04	05	02
01	02	03	04												

Server send (hexadecimal):

FF	03	08	01	31	00	00	00	02	03	00	01	E8	04	05	02

The green data is the written DB1.DBD1000 with a total of 4 bytes of data.

The red data is the starting address DB1.DBD1000 (0x03E8);

7.3 Read the data in the M area

The client sends a command to read data:

	字节	参数	类型	注释
8 bytes header	0	msg.rx	byte	0x03
	1	msg.tx	byte	0xFF
	2	msg.ln	byte	0x08
	3	msg.nr	byte	Client given
	4	msg.a	byte	0x00
	5	msg.f	byte	0x00
	6	msg.b	byte	0x33 (read/write M-zone)
	7	msg.e	byte	0x00
8 bytes of extended headers	8	msg.device_adr	byte	Remote (PLC) station address 0-31
	9	msg.data_area	byte	Useless, 0x00
	10,11	msg.data_adr	word	Starting address in M area, 0~65534 [10] = Starting address/256 [11] = From address %256

	12	msg.data_idx	byte	Useless, 0x00
	13	msg.data_cnt	byte	The maximum number of bytes of data to be read is 200
	14	msg.data_type	byte	0x05 (bytes)
	15	msg.function	byte	0x01 (Read Data)

The server sends a read data response frame:

	Byte	Parameter	Type	Exegesis
8 bytes header	0	msg.rx	byte	0xFF
	1	msg.tx	byte	0x03
	2	msg.ln	byte	0x08 + number of bytes of data read
	3	msg.nr	byte	Consistent with the client given
	4	msg.a	byte	0x33 (read/write M-zone)
	5	msg.f	byte	0x00 (non-0 means there is an error)
	6	msg.b	byte	0x00
	7	msg.e	byte	0x00
8 bytes of extended headers	8	msg.device_adr	byte	Remote (PLC) station address 0-31
	9	msg.data_area	byte	Useless, 0x00
	10,11	msg.data_adr	word	Starting address in M area, 0~65534 [10] = Starting address/256 [11] = From address %256
	12	msg.data_idx	byte	Useless, 0x00
	13	msg.data_cnt	byte	The number of bytes of data that has been read is less than or equal to 200
	14	msg.data_type	byte	0x05 (bytes)
	15	msg.function	byte	0x01 (Read Data)
User data (up to 200 bytes)	16~16+ (Bytes Read - 1)	msg.d[0~(Number of bytes read - 1)]	byte array	Read data

For example, the client reads the MB10~MB15 of S7-300 (station address 2), which is 6 bytes

Client sends (hexadecimal):

03	FF	08	01	00	00	33	00	02	00	00	0A	00	06	05	01

Server send (hexadecimal):

FF	03	0E	01	33	00	00	00	02	00	00	0A	00	06	05	01
00	00	00	00	00	00										

The green data is a total of 6 bytes of MB10~MB15 read data.

The red data is the starting address MB10 (0x000A);

7.4 Write M area data

The client sends a write data command:

	Byte	Parameter	Type	Exegesis
8 byte header	0	msg.rx	byte	0x03
	1	msg.tx	byte	0xFF
	2	msg.ln	byte	0x08 + number of bytes of written data
	3	msg.nr	byte	Client given
	4	msg.a	byte	0x00
	5	msg.f	byte	0x00
	6	msg.b	byte	0x33 (read/write M-zone)
	7	msg.e	byte	0x00
8 bytes of extended headers	8	msg.device_adr	byte	Remote (PLC) station address 0-31
	9	msg.data_area	byte	Useless, 0x00
	10,11	msg.data_adr	word	Starting address in M area, 0~65534 [10] = Starting address/256 [11] = From address %256
	12	msg.data_idx	byte	Useless, 0x00
	13	msg.data_cnt	byte	The maximum number of bytes of data to be written is 200
	14	msg.data_type	byte	0x05 (bytes)
	15	msg.function	byte	0x02 (write data)
User data (up to 200 bytes)	16~ 16+ (Bytes of data written - 1)	msg.d [0~(Number of bytes written to data-1)]	byte array	data written

The server sends a write data response frame:

	Byte	Parameter	Type	Exegesis
8 bytes	0	msg.rx	byte	0xFF

header	1	msg.tx	byte	0x03
	2	msg.ln	byte	0x08
	3	msg.nr	byte	Consistent with the client given
	4	msg.a	byte	0x33 (read/write M-zone)
	5	msg.f	byte	0x00 (non-0 means there is an error)
	6	msg.b	byte	0x00
	7	msg.e	byte	0x00
8 bytes of extended headers	8	msg.device_adr	byte	Remote (PLC) station address 0-31
	9	msg.data_area	byte	Useless, 0x00
	10,11	msg.data_adr	word	Starting address in M area, 0~65534 [10] = Starting address/256 [11] = From address %256
	12	msg.data_idx	byte	Useless, 0x00
	13	msg.data_cnt	byte	The number of bytes of data that has been written is less than or equal to 200
	14	msg.data_type	byte	0x05 (bytes)
	15	msg.function	byte	0x02 (write data)

For example, a client writes data 0x0102 to MW20 of S7-300 (station address 2), which is 2 bytes

Client send (hexadecimal):

03	FF	0A	01	00	00	33	00	02	00	00	14	00	02	05	02
01	02														

Server send (hexadecimal):

FF	03	08	01	33	00	00	00	02	00	00	14	00	02	05	02

The green data is a total of 2 bytes of MW20 data written.

The red data is the starting address MW20 (0x0014);

7.5 Read the data in the I and Q zones (input/output signals).

The client sends a command to read data:

	Byte	Parameter	Type	Exegesis
8 bytes header	0	msg.rx	byte	0x03
	1	msg.tx	byte	0xFF
	2	msg.ln	byte	0x08
	3	msg.nr	byte	Client given
	4	msg.a	byte	0x00

	5	msg.f	byte	0x00
	6	msg.b	byte	0x34 (Read/Write I, Q Zones)
	7	msg.e	byte	0x00
8 bytes of extended headers	8	msg.device_adr	byte	Remote (PLC) station address 0-31
	9	msg.data_area	byte	Data area 0x00: Zone I 0x01: Area Q
	10,11	msg.data_adr	word	The starting address of the I, Q area, 0~65534 [10] = Starting address/256 [11] = From address %256
	12	msg.data_idx	byte	Useless, 0x00
	13	msg.data_cnt	byte	The maximum number of bytes of data to be read is 200
	14	msg.data_type	byte	0x05 (bytes)
	15	msg.function	byte	0x01 (Read Data)

The server sends a read data response frame:

	Byte	Parameter	Type	Exegesis
8 bytes header	0	msg.rx	byte	0xFF
	1	msg.tx	byte	0x03
	2	msg.ln	byte	0x08 + number of bytes of data read
	3	msg.nr	byte	Consistent with the client given
	4	msg.a	byte	0x34 (Read/Write I, Q Zones)
	5	msg.f	byte	0x00 (non-0 means there is an error)
	6	msg.b	byte	0x00
	7	msg.e	byte	0x00
8 bytes of extended headers	8	msg.device_adr	byte	Remote (PLC) station address 0-31
	9	msg.data_area	byte	Data area 0x00: Zone I 0x01: Area Q
	10,11	msg.data_adr	word	The starting address of the I, Q area, 0~65534 [10] = Starting address/256 [11] = From address %256

	12	msg.data_idx	byte	Useless, 0x00
	13	msg.data_cnt	byte	The number of bytes of data that has been read is less than or equal to 200
	14	msg.data_type	byte	0x05 (bytes)
	15	msg.function	byte	0x01 (Read Data)
User data (up to 200 bytes)	16~ 16+ (Bytes Read -1)	msg.d[0~(Number of bytes read - 1)]	byte array	Read data

Example 1: The IB0 of S7-300 (station address 2) is read by the client and has a total of 1 byte Client send (hexadecimal):

03	FF	08	01	00	00	34	00	02	00	00	00	00	01	05	01

Server send (hexadecimal):

FF	03	09	01	34	00	00	00	02	00	00	00	00	01	05	01	00

The green data is a total of 1 byte of IB0 data read.

The red data is the starting address IB0 (0x0000);

Example 2: The client reads the QB1~QB2 of the S7-300 (station address 3) with a total of 2 bytes Client send (hexadecimal):

03	FF	08	01	00	00	34	00	03	01	00	01	00	02	05	01

Server send (hexadecimal):

FF	03	0A	01	34	00	00	00	03	01	00	01	00	02	05	01	00	00

The green data is a total of 2 bytes of QB1~QB2 data read;

The red data is the starting address QB1 (0x0001);

7.6 Write I and Q zone (input/output signal) data

The client sends a write data command:

	Byte	Parameter	Type	Exegesis
8 bytes header	0	msg.rx	byte	0x03
	1	msg.tx	byte	0xFF
	2	msg.ln	byte	0x08 + number of bytes of written data
	3	msg.nr	byte	Client given
	4	msg.a	byte	0x00
	5	msg.f	byte	0x00
	6	msg.b	byte	0x34 (Read/Write I, Q Zones)

	7	msg.e	byte	0x00
8 bytes of extended headers	8	msg.device_adr	byte	Remote (PLC) station address 0-31
	9	msg.data_area	byte	Data area 0x00: Zone I 0x01: Area Q
	10,11	msg.data_adr	word	The starting address of the I, Q area, 0~65534 [10] = Starting address/256 [11] = From address %256
	12	msg.data_idx	byte	Useless, 0x00
	13	msg.data_cnt	byte	The maximum number of bytes of data to be written is 200
	14	msg.data_type	byte	0x05 (bytes)
	15	msg.function	byte	0x02 (write data)
User data (up to 200 bytes)	16~ 16+ (Bytes of data written - 1)	msg.d[0~(Number of bytes written to data-1)]	byte array	data written

The server sends a write data response frame:

	Byte	Parameter	Type	Exegesis
8 bytes header	0	msg.rx	byte	0xFF
	1	msg.tx	byte	0x03
	2	msg.ln	byte	0x08
	3	msg.nr	byte	Consistent with the client given
	4	msg.a	byte	0x34 (Read/Write I, Q Zones)
	5	msg.f	byte	0x00 (non-0 means there is an error)
	6	msg.b	byte	0x00
	7	msg.e	byte	0x00
8 bytes of extended headers	8	msg.device_adr	byte	Remote (PLC) station address 0-31
	9	msg.data_area	byte	Data area 0x00: Zone I 0x01: Area Q
	10,11	msg.data_adr	word	The starting address of the I, Q area, 0~65534 [10] = Starting

				address/256 [11] = From address %256
	12	msg.data_idx	byte	Useless, 0x00
	13	msg.data_cnt	byte	The number of bytes of data that has been written is less than or equal to 200
	14	msg.data_type	byte	0x05 (bytes)
	15	msg.function	byte	0x02 (write data)

For example, a client writes data 0xFF to QB0 of S7-300 (station address 2), a total of 1 byte Client send (hexadecimal):

03	FF	09	01	00	00	34	00	02	01	00	00	00	01	05	02
FF															

Server send (hexadecimal):

FF	03	08	01	34	00	00	00	02	01	00	00	00	01	05	02

The green data is a total of 1 byte of written QB0 data.

The red data is the starting address QB0 (0x0000);

7.7 Read the bit values of DB, M, I, Q

Note: The OpenTCP protocol only supports reading one bit.

The client sends a read command:

	Byte	Parameter	Type	Exegesis
8 bytes header	0	msg.rx	byte	0x03
	1	msg.tx	byte	0xFF
	2	msg.ln	byte	0x08
	3	msg.nr	byte	Client given
	4	msg.a	byte	0x00
	5	msg.f	byte	0x00
	6	msg.b	byte	Consistent with byte operation definitions
	7	msg.e	byte	0x00
8 bytes of extended headers	8	msg.device_adr	byte	Remote (PLC) station address 0-31
	9	msg.data_area	byte	Consistent with byte operation definitions
	10,11	msg.data_adr	word	Consistent with byte operation definitions
	12	msg.data_idx	byte	Consistent with byte operation definitions

	13	msg.data_cnt	byte	Useless = 0x00
	14	msg.data_type	byte	High four-digit value: bit offset 0-7 Lower 4 digits: = 4 (bits)
	15	msg.function	byte	0x01 (Read Data)

The server sends a read response frame:

	Byte	Parameter	Type	Exegesis
8 bytes header	0	msg.rx	byte	0xFF
	1	msg.tx	byte	0x03
	2	msg.ln	byte	0x09
	3	msg.nr	byte	Consistent with the client given
	4	msg.a	byte	Consistent with byte operation definitions
	5	msg.f	byte	0x00 (non-0 means there is an error)
	6	msg.b	byte	0x00
	7	msg.e	byte	0x00
8 bytes of extended headers	8	msg.device_adr	byte	Remote (PLC) station address 0-31
	9	msg.data_area	byte	Consistent with byte operation definitions
	10,11	msg.data_adr	word	Consistent with byte operation definitions
	12	msg.data_idx	byte	Consistent with byte operation definitions
	13	msg.data_cnt	byte	Useless = 0x00
	14	msg.data_type	byte	High four-digit value: bit offset 0-7 Lower 4 digits: = 4 (bits)
	15	msg.function	byte	0x01 (Read Data)
User Data (1 byte)	16	msg.d[0]	byte	Read bit values 0x00:OFF 0x01：ON

For example, the client reads the bit value of Q0.5 of S7-300 (station address 2). Client send (hexadecimal):

03	FF	08	01	00	00	34	00	02	01	00	00	00	00	54	01

Server send (hexadecimal):

FF	03	09	01	34	00	00	00	02	01	00	00	00	00	54	01
00															

The green data is the bit value of the read Q0.5, i.e., OFF;

The red data is the starting address QB0 (0x0000), and the higher 4 bits (=5) of the 0x54 are the bit offset.

7.8 Write the bitvalues of DB, M, I, Q

Note: The OpenTCP protocol only supports writing to one bit (the input I area cannot be written, depending on the external signal).

The client sends a write command:

	Byte	Parameter	Type	Exegesis
8 bytes header	0	msg.rx	byte	0x03
	1	msg.tx	byte	0xFF
	2	msg.ln	byte	0x09
	3	msg.nr	byte	Client given
	4	msg.a	byte	0x00
	5	msg.f	byte	0x00
	6	msg.b	byte	Consistent with byte operation definitions
	7	msg.e	byte	0x00
8 bytes of extended headers	8	msg.device_adr	byte	Remote (PLC) station address 0-31
	9	msg.data_area	byte	Consistent with byte operation definitions
	10,11	msg.data_adr	word	Consistent with byte operation definitions
	12	msg.data_idx	byte	Consistent with byte operation definitions
	13	msg.data_cnt	byte	Useless = 0x00
	14	msg.data_type	byte	High four-digit value: bit offset 0-7 Lower 4 digits: = 4 (bits)
	15	msg.function	byte	0x02 (write data)
User Data (1 byte)	16	msg.d[0]	byte	Bit values written to the 0x00：OFF 0x01：ON

The server sends a write bit response frame:

	Byte	Parameter	Type	Description
8 bytes header	0	msg.rx	byte	0xFF
	1	msg.tx	byte	0x03
	2	msg.ln	byte	0x08
	3	msg.nr	byte	Consistent with the client given

	4	msg.a	byte	Consistent with byte operation definitions
	5	msg.f	byte	0x00 (non-0 means there is an error)
	6	msg.b	byte	0x00
	7	msg.e	byte	0x00
8 bytes of extended headers	8	msg.device_adr	byte	Remote (PLC) station address 0-31
	9	msg.data_area	byte	Consistent with byte operation definitions
	10,11	msg.data_adr	word	Consistent with byte operation definitions
	12	msg.data_idx	byte	Consistent with byte operation definitions
	13	msg.data_cnt	byte	Useless = 0x00
	14	msg.data_type	byte	High four-digit value: bit offset 0-7 Lower 4 digits: = 4 (bits)
	15	msg.function	byte	0x02 (write data)

For example, the client is set to Q0.5 in S7-300 (station address 2). Client send (hexadecimal):

03	FF	09	01	00	00	34	00	02	01	00	00	00	00	54	02	01

Server send (hexadecimal):

FF	03	08	01	34	00	00	00	02	01	00	00	00	00	54	02

The green data is the bit value of the written Q0.5, i.e., ON;

The red data is the starting address QB0 (0x0000), and the higher 4 bits (=5) of the 0x54 are the bit offset.

7.9 Error number

0x00: no error;

0xA1~0xAC: P LC busy or answering error (S7 bus communication error);

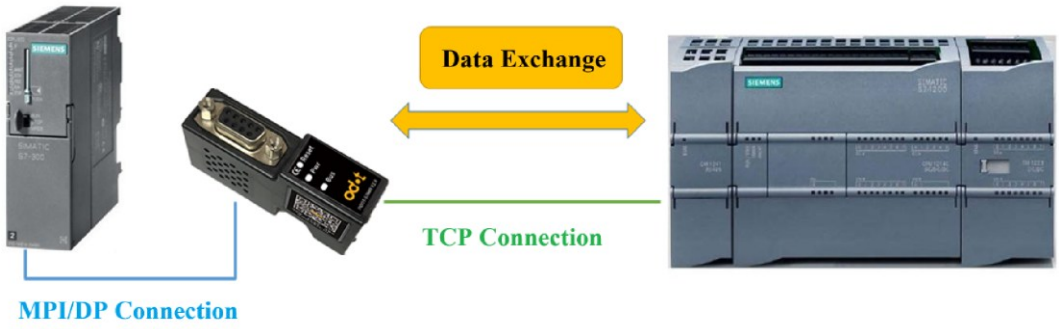
0x88~0x8E: P LC illegal address access (read and write addresses do not exist in the PLC);

Usually the error number for accessing an illegal address is 0x8C.

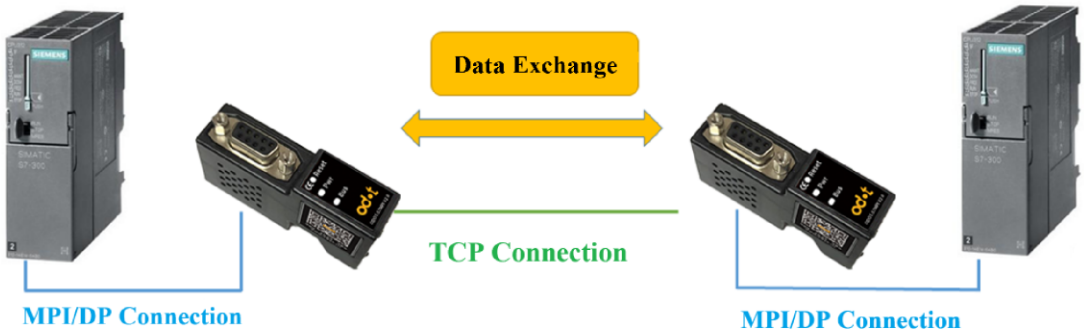
8 Data exchange

The bridge (only supported by ODOT-S7MPI Pro) supports guest functions (supports S7TCP protocol and ModbusTCP protocol), which can realize data exchange between PLCs, such as S7300 and S71200/1500, S7300 and S7300/S7200, or S7300 and Modbus TCP slave devices, etc., There is no need to write read and write programs in the PLC.

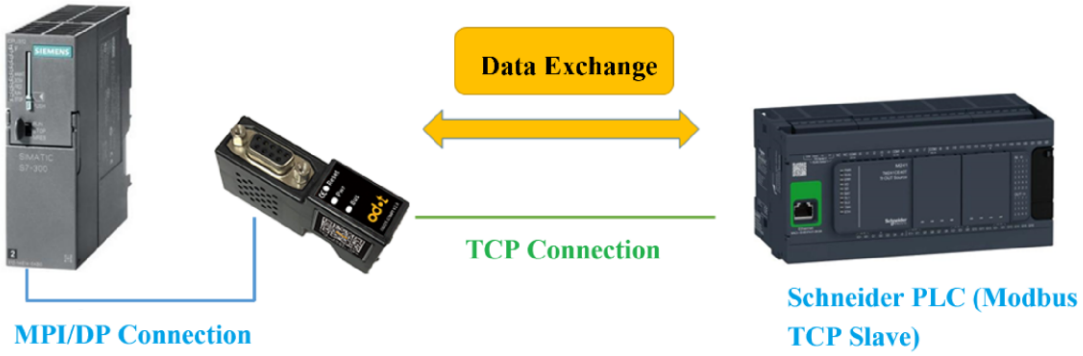
8.1 Data exchange between S7300 and S71200/1500



8.2 Data exchange between S7300 and S7300/S7200



8.3 Data exchange between the S7300 and the Modbus TCP slave device



Odot Automation System Co., Ltd.

Add: No. 261 Feiyun Avenue Comprehensive free Trade Zone 204 workshop, Mianyang, Sichuan, China.

Tel: +86-0816-2538289

Zip Code: 621000

Email:sales@odotautomation.com

Web: www.odotautomation.com