Modbus TCP User Manual
V1.0

Update Records

No.	Version	Date	Contents
1	1.0	2025/07/15	New Files

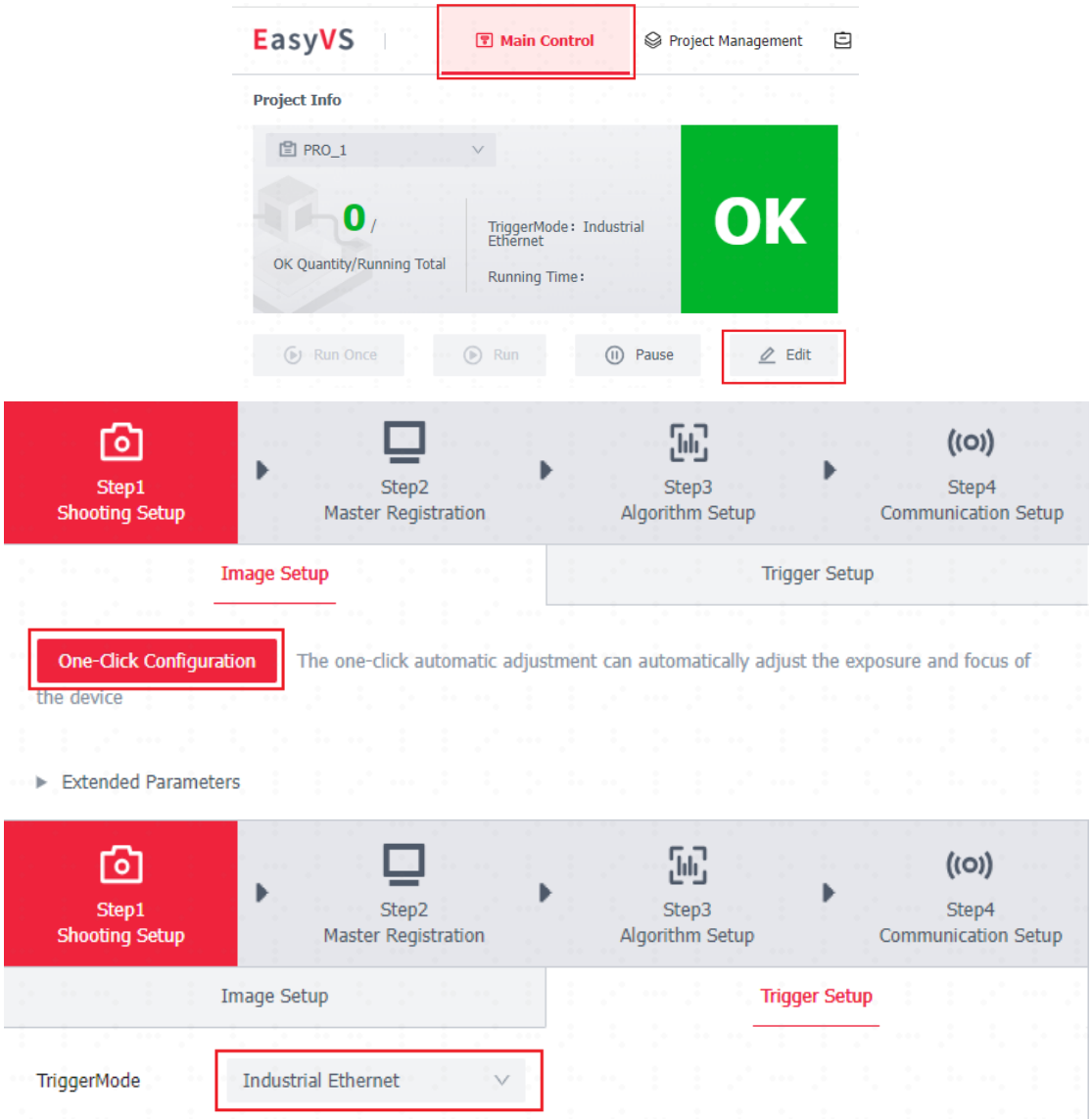
Content

1 Vision Sensor Configuration 1
1.1 Shooting Setup 1
1.2 Master Registration 1
1.3 Algorithm Setup 2
1.4 Communication Setup 4
2 Keyence KV8000 PLC Configuration 6
2.1 PLC Configuration 6
2.1.1 Create New Project 6
2.1.2 Write the PLC program 9
2.2 Communication Test 22
2.2.1 Download program 22
2.2.2 Communication Test 23
3 Rockwell PLC Configuration 25
3.1 PLC Configuration 25
3.1.1 Create New Project 25
3.1.2 Write the PLC program 26
3.2 Communication Test 31
3.2.1 Download program 31
3.2.2 Communication Test 33
4 Mitsubishi FX5U PLC Configuration 35
4.1 PLC Configuration 35
4.1.1 Create New Project 35
4.1.2 Write PLC program 41
4.2 Communication Test 44
4.2.1 Download program 44
4.2.2 Communication Test 46
5 Mitsubishi Q06UDV PLC Configuration 48
5.1 PLC Configuration 48
5.1.1 Create New Project 48
5.1.2 Write the PLC Program 52
5.2 Communication Test 57
5.2.1 Download Program 57
5.2.2 Communication Test 60
6 Siemens 1200 PLC Configuration 62
6.1 PLC Configuration 62
6.1.1 Create New Project 62
6.1.2 Write the PLC Program 64
6.2 Communication Test 76
6.2.1 Download Program 76
6.2.2 Communication Test 77

1 Vision Sensor Configuration

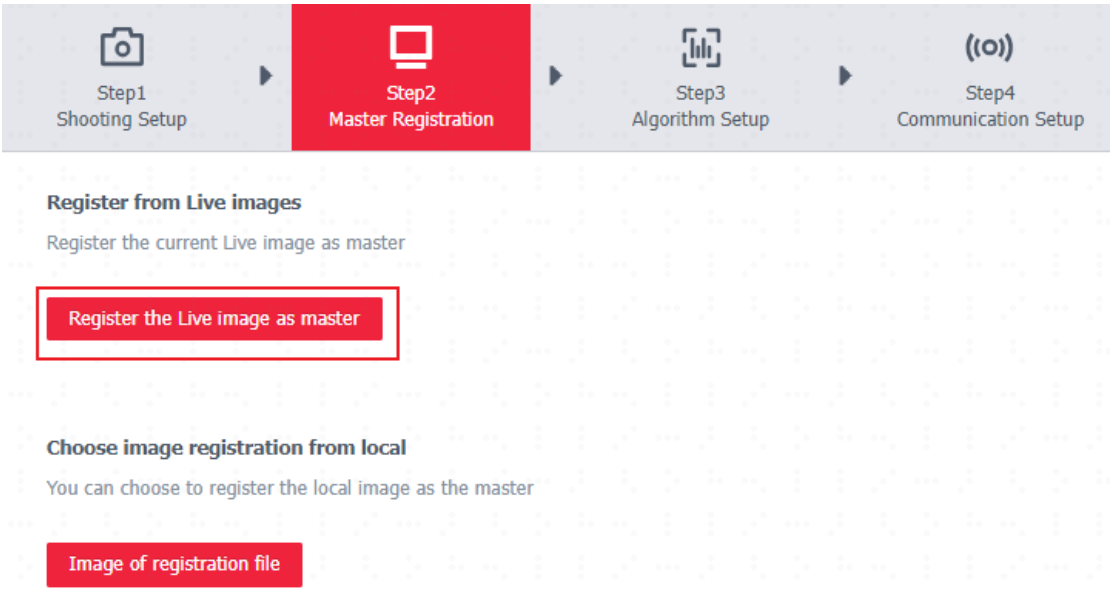
1.1 Shooting Setup

Click [Main Control] → [Edit] to enter the Parameters Interface. In the [Shooting Setup], click [Image Setup] → [One-click Configuration], then wait for the configuration to complete and return. Based on project requirements, select the appropriate Trigger Mode in the [Trigger Setup]. For this case, select [Industrial Ethernet].



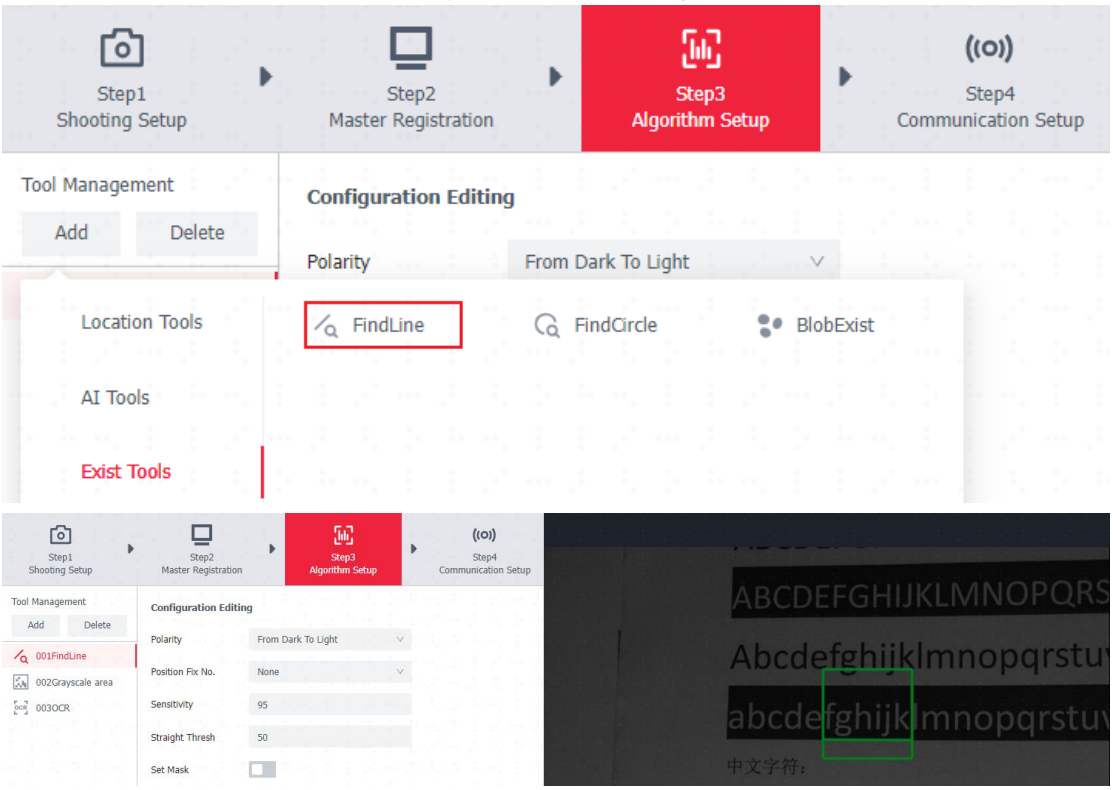
1.2 Master Registration

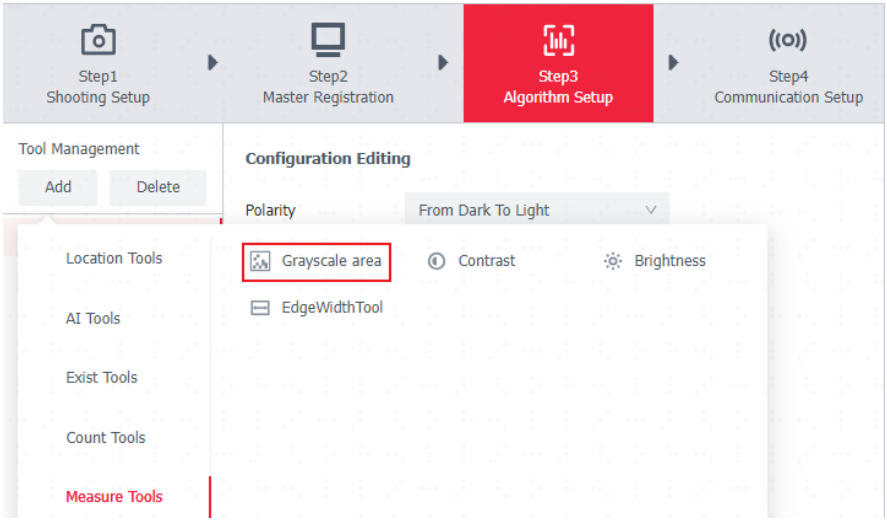
In the Master Registration control part, click [Register the Live Image as Master]

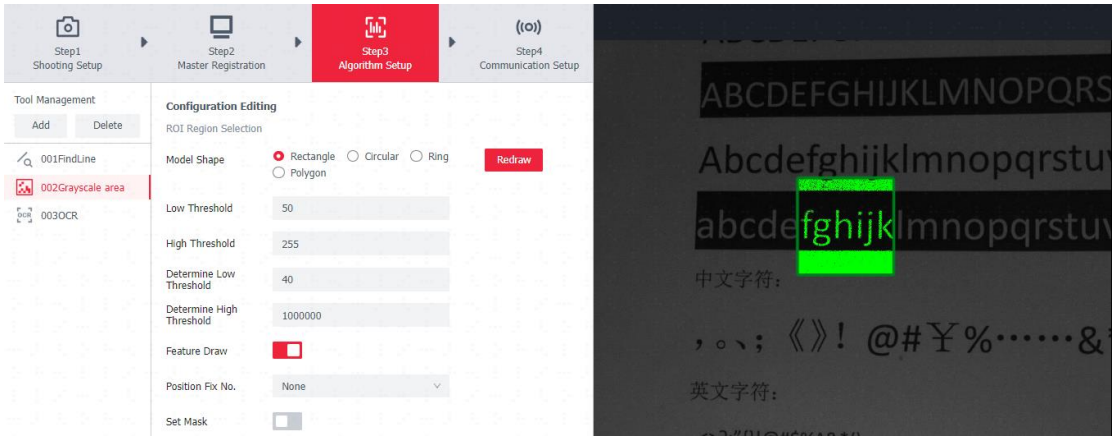


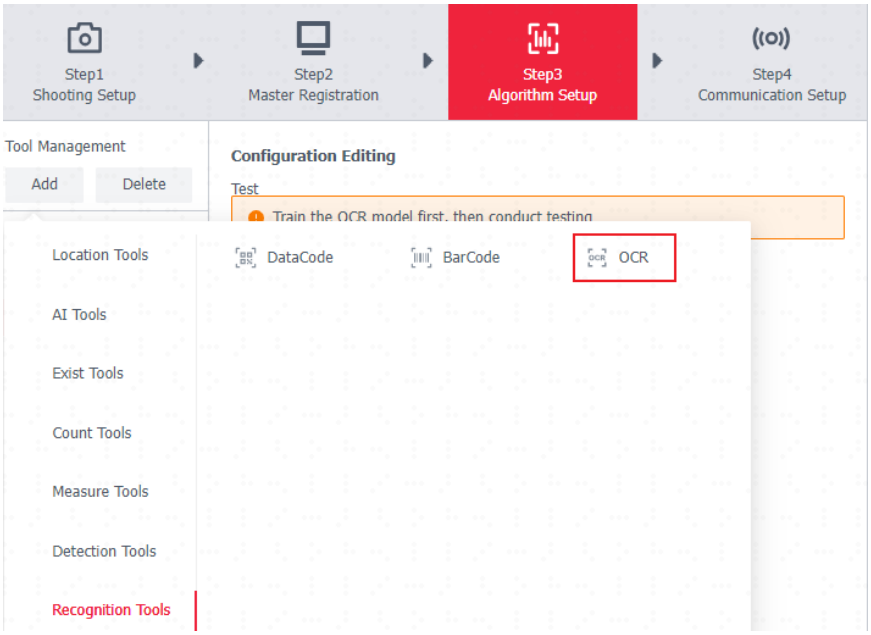
1.3 Algorithm Setup

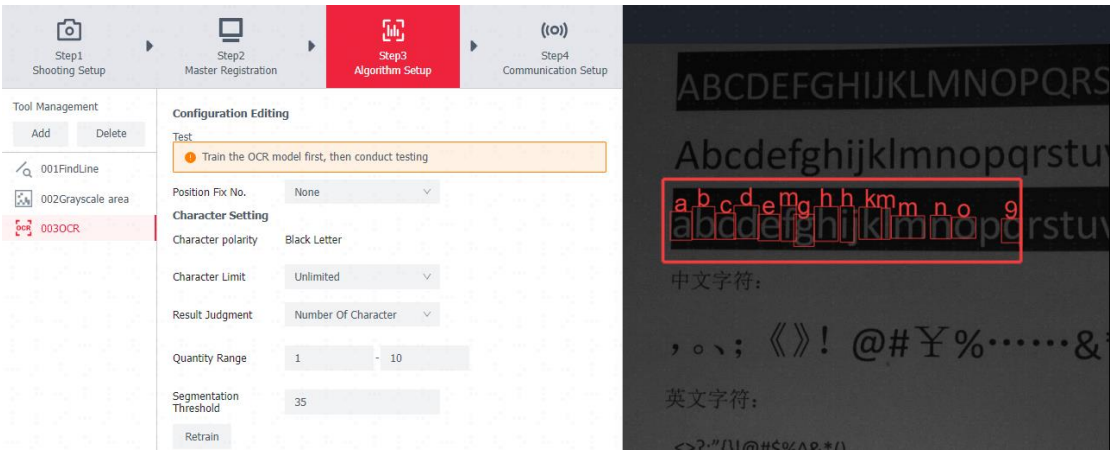
In Algorithm setup, add the required tools for the project. In this case, add the [FindLine], [Grayscale area], and [OCR] tools. Click on [Add] → [Exist Tools] → [FindLine], [Measure Tools] → [Grayscale area], [Recognition Tools] → [OCR].




Step1
Shooting Setup
Step2
Master Registration
Step3
Algorithm Setup
Step4
Communication Setup
Tool Management
Add
Delete
Configuration Editing
Polarity
From Dark To Light
Location Tools
AI Tools
Exist Tools
Count Tools
Measure Tools
Grayscale area
Contrast
Brightness
EdgeWidthTool


Step1
Shooting Setup
Step2
Master Registration
Step3
Algorithm Setup
Step4
Communication Setup
Tool Management
Add
Delete
001FindLine
002Grayscale area
003OCR
Configuration Editing
ROI Region Selection
Model Shape
Rectangle
Circular
Ring
Polygon
Redraw
Low Threshold
50
High Threshold
255
Determine Low Threshold
40
Determine High Threshold
1000000
Feature Draw
Position Fix No.
None
Set Mask
中文字符：
英文字符：

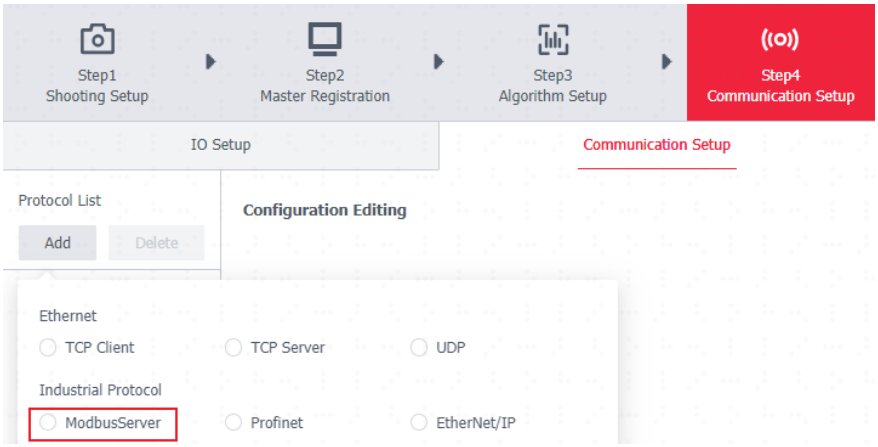

Step1
Shooting Setup
Step2
Master Registration
Step3
Algorithm Setup
Step4
Communication Setup
Tool Management
Add
Delete
Configuration Editing
Test
Train the OCR model first, then conduct testing
Location Tools
AI Tools
Exist Tools
Count Tools
Measure Tools
Detection Tools
Recognition Tools
DataCode
BarCode
OCR

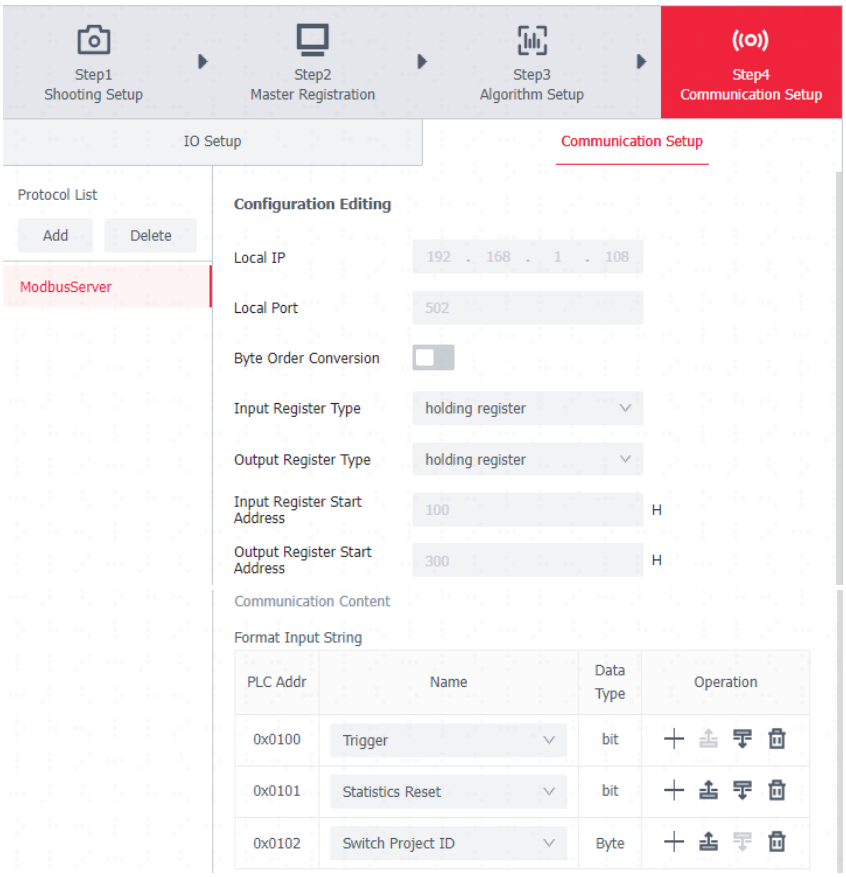


1.4 Communication Setup

In the communication Setup Interface, click [Communication Setup] → [Add] → [ModbusServer]. Add the communication content as required. In this case, the added communication content is shown in the table below.

	PLCAddress	Name	Data Type	Size (bytes)
Format Input String	0x0100	Trigger	bit	1
	0x0101	Statistics Reset	bit	1
	0x0102	Switch Projects ID	Byte	1
Format Output String	0x0300	Running Total	Integer	4
	0x0302	1 LineExist.StartPointX	Float	4
	0x0304	1 LineExist.StartPointX	Float	4
	0x0306	2 GrayscaleArea.Num	Integer	4
	0x0308	3 -1 OCR.String Length	Integer	4
	0x030a	3 -1 OCR.StringContent	string	20





Conditions Output In this case, Select [Unlimited].

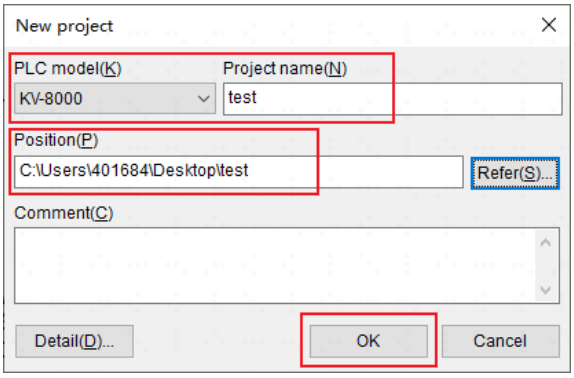
After all configurations are completed, click [Save Project] to end the configuration of the vision sensor.

2 Keyence KV8000 PLC Configuration

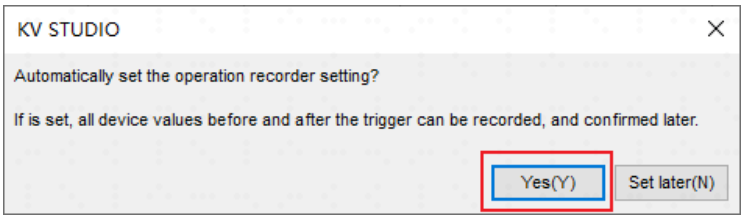
2.1 PLC Configuration

2.1.1 Create New Project

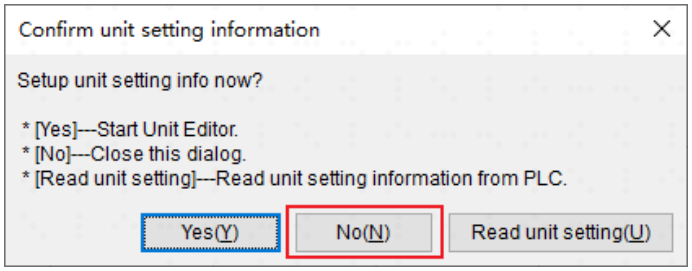
Open KV STUDIO, click New Project, select KV-8000 as the supported model, modify the project name and project save location, and click OK after setting.



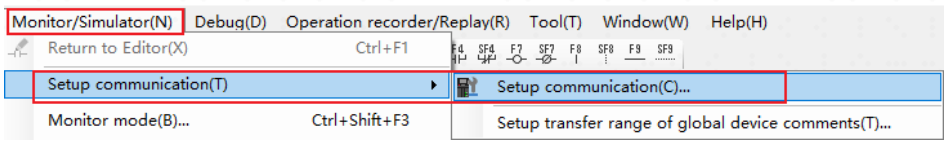
Click [yes].



Click [No].

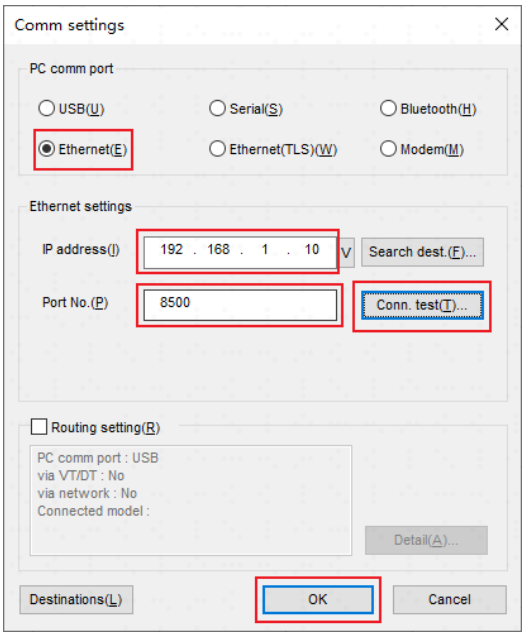


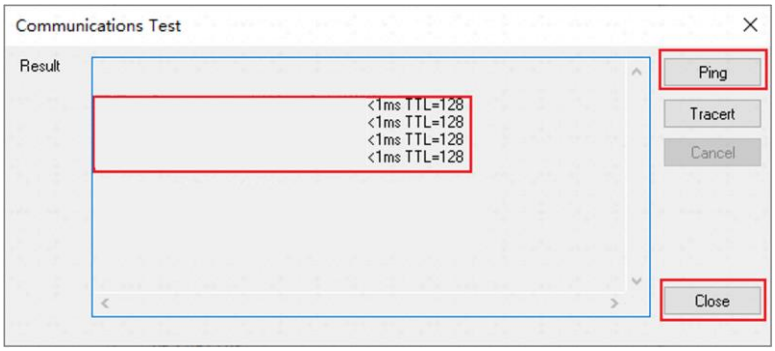
Choose [Monitor/Simulator]→[Setup communication]→[Setup communication].



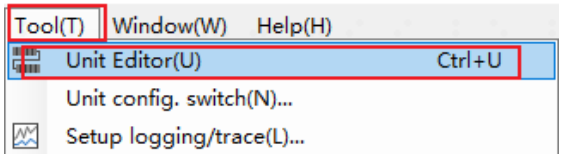
Select the **PC comm port** [Ethernet (E)], enter the PLC's IP address and port number

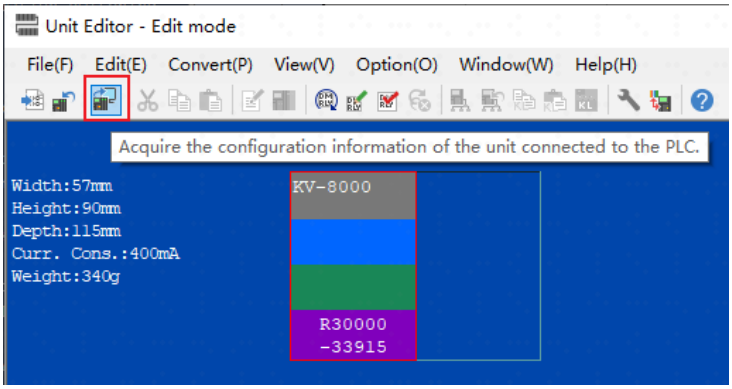
in the Ethernet settings, click [Conn. test] - [Ping], and when the result shows that the network connection status is good, click [Close] - [OK].

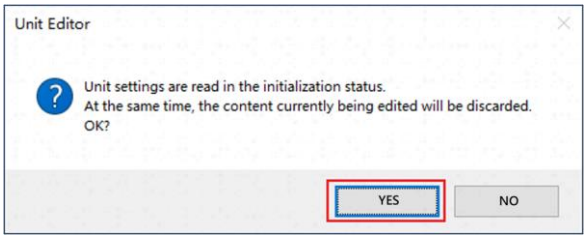




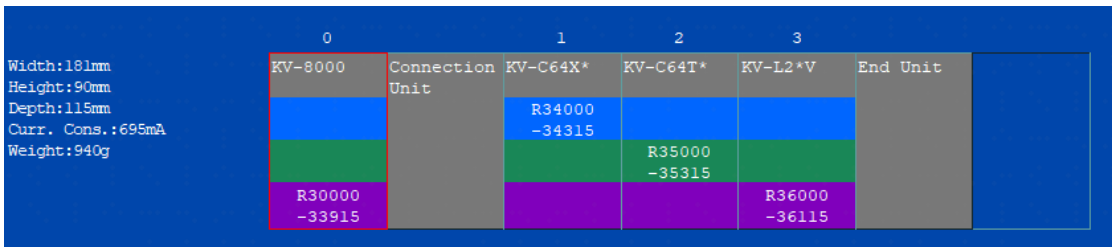
Click [Tool] - [Unit Editor], then click [Acquire the configuration information of the unit connected to the PLC] in the unit editor interface. Select “Yes” in the pop-up interface.



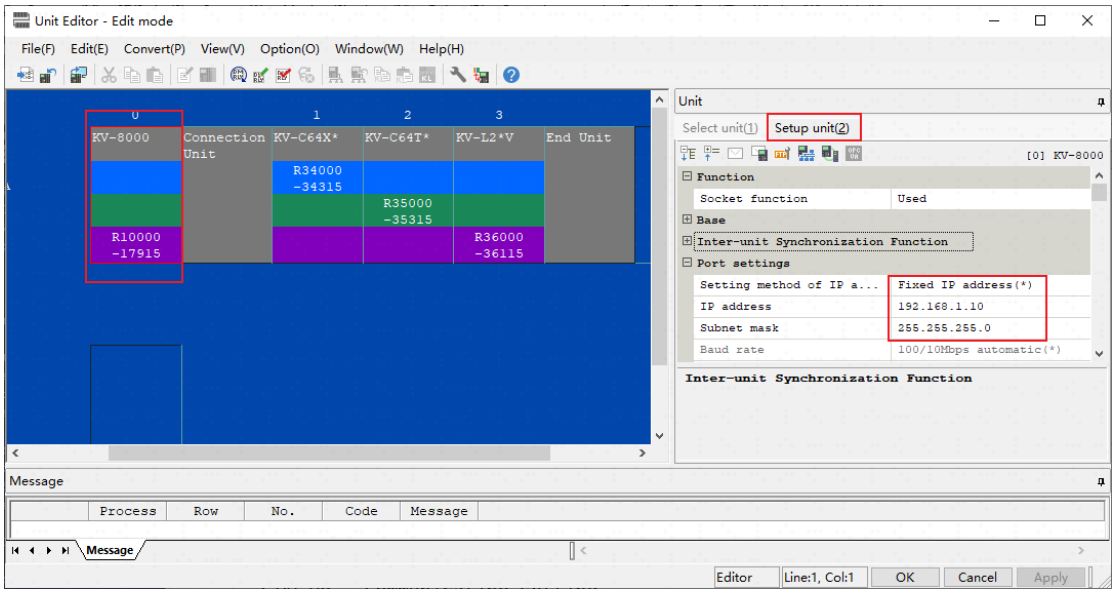




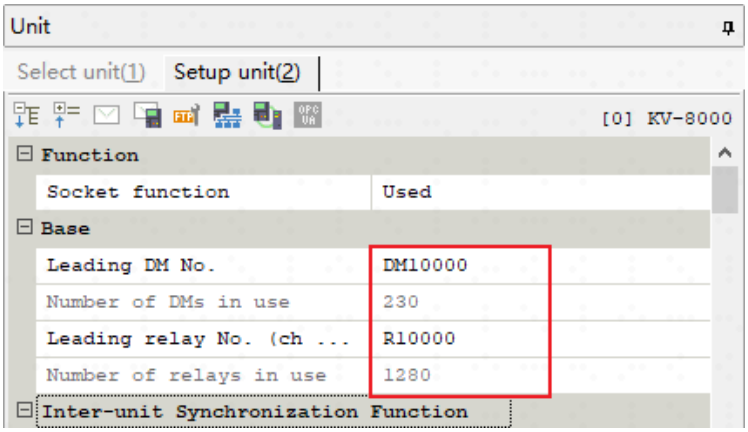
After successful acquisition, the actual PLC modules and information about each module will be displayed in the unit editor. Click OK to exit the unit editor.



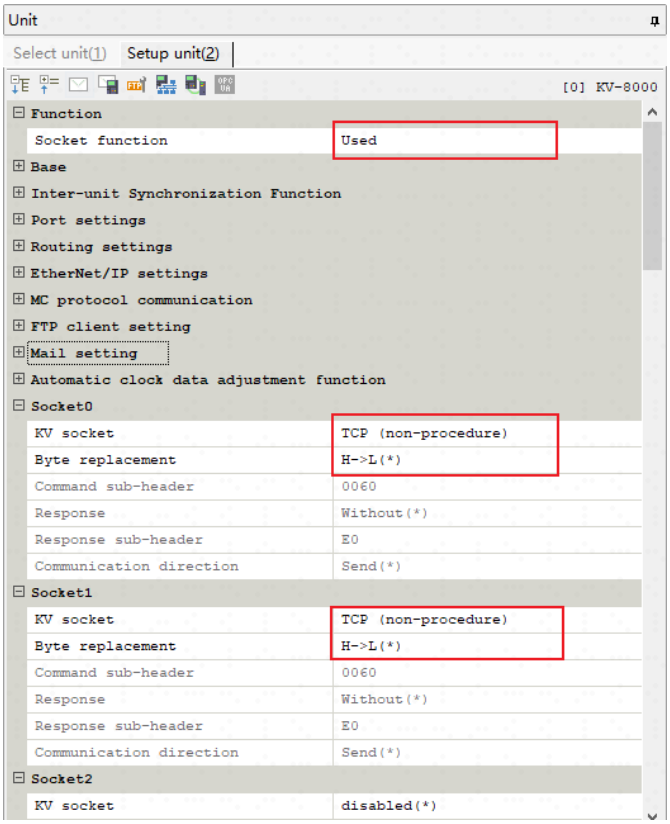
Select the KV-8000 CPU module, click [Setup Unit], and confirm network parameters such as IP address and subnet mask.



Please choose the Socket function to[Used], [Leading relay NO.] Set toR10000.

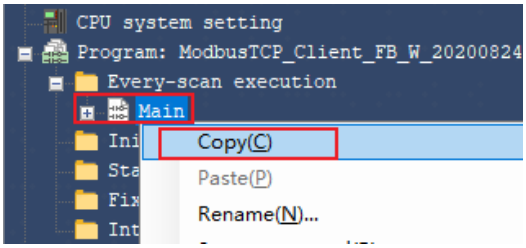


Available socket numbers (socket 0/1 is used in this case), select “TCP (non- procedure)” and “Byte high-to-low exchange (high → low)”.

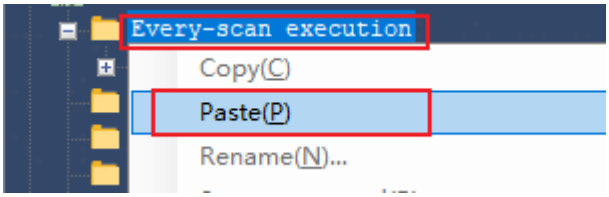


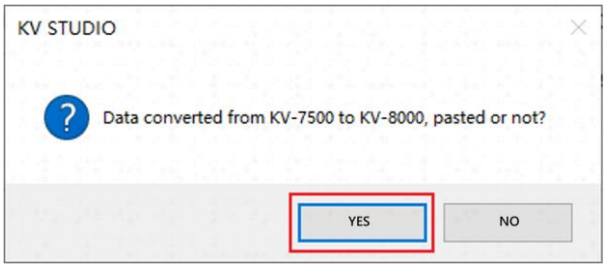
2.1.2 Write the PLC program

Since the Keyence KV-8000 only supports the SocketTCP function and does not have a ModbusTCP instruction library, you need to download the ModbusTCP function block from the Keyence official website. After the download is complete, open the official sample program, go to the left project bar, Copy “Every-scan execution” - “Main” program.

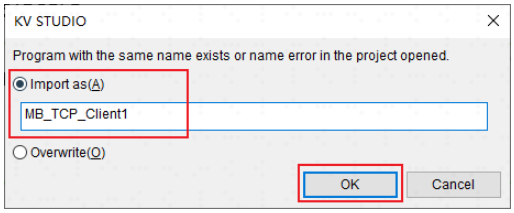


In the current program, select “Every-scan execution” in the left project pane, right-click the menu bar and select ‘Paste’, then select “yes” in the pop-up interface.





A prompt will appear indicating that there is a program with the same name. You will need to modify the program name. In this case, enter MB_TCP_Client1, click OK, and the setup is complete.

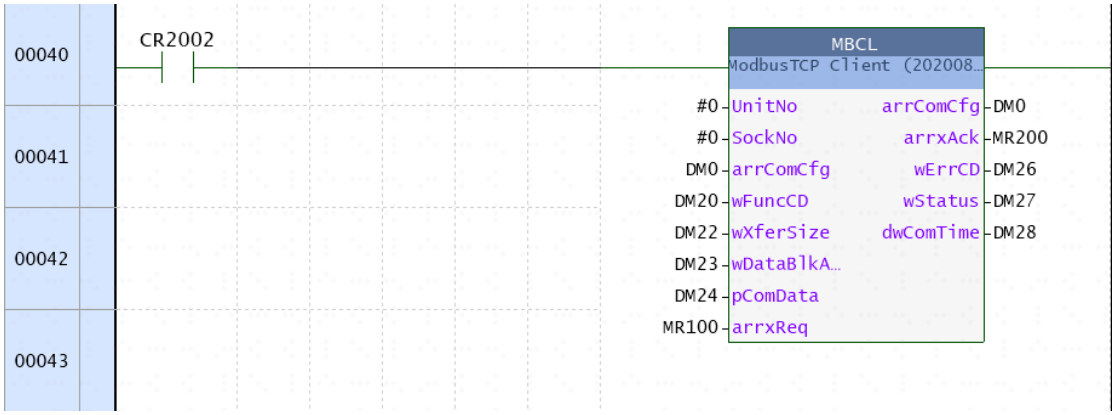


Function block pin configuration：

	Independent variable name	Type	Datatype	Size	Statements	Sample program usage address
1	UnitNo	Unit	UINT	1 word	Unit number(Ethernet Unit)	#0
2	SocketNo	IN	UINT	1 word	0-15	#0
3	arrComCfg	INOUT	UINT	10 words	Communication setting	DM0~DM9
4	wFuncCD	IN	UINT	1 word	Function code	DM20
5	wXferSize	IN	UINT	1 word	Date amount	DM22
6	wDataBlkAddr	IN	UINT	1 word	Date block address	DM23
7	pComData	IN	UDINT	2 words	PLC data address	DM24~DM25
8	arrxReq	IN	UINT	1 word	request relay(4bits)	MR100~MR103 (MR104~115

						reserve)
9	arrxAck	OUT	UINT	1 word	response relay(10bits)	MR200-MR209 (MR210~215 reserve)
10	wErrCD	OUT	UINT	1 word	Error code	DM26
11	wStatus	OUT	UINT	1 word	Connection status	DM27
12	dwComTime	OUT	DINT	1 word	Communication time monitor	DM28-DM29

***Modbus Data storage address EM0~EM123**



ModbusTCP Communication Parameters setting：

Number	Name	Device address	Parameter value	Note
0	PLC port number	DM0	0	0 means automatically setting the port number
1	Server IP address[1]	DM1	192	
2	Server IP address[2]	DM2	168	
3	Server IP address[3]	DM3	1	
4	Server IP address[4]	DM4	108	
5	Server port Number	DM5	502	
6	Server station number	DM6	255	
7	TCP open request timeout [ms]	DM7	1000	Range 0~65535
8	Request sent timeout [ms]	DM8	1000	
9	Receive request timeout [ms]	DM9	1000	

*The original official case program was written in ladder logic language, which was too long to read conveniently. Therefore, part of the program in this case has been modified and written in ST language.

*TCP open request timeout, send request timeout. The receive request timeout was set to 10000 ms in the official case, which was too long and affected the program execution efficiency. In this case, it has been modified to 1000 ms.

```
00007
IF CR2008 THEN
    DM0 := 0;
    DM1 := 192;
    DM2 := 168;
    DM3 := 1;
    DM4 := 108;
    DM5 := 502;
    DM6 := 255;
    DM7 := 1000;
    DM8 := 1000;
    DM9 := 1000;
END_IF;
```

Modbus Communication data：

```
00033
IF CR2008 THEN
    DM20 := 16#0004;
    DM22 := 120;
    DM23 := 768;
END_IF;
```

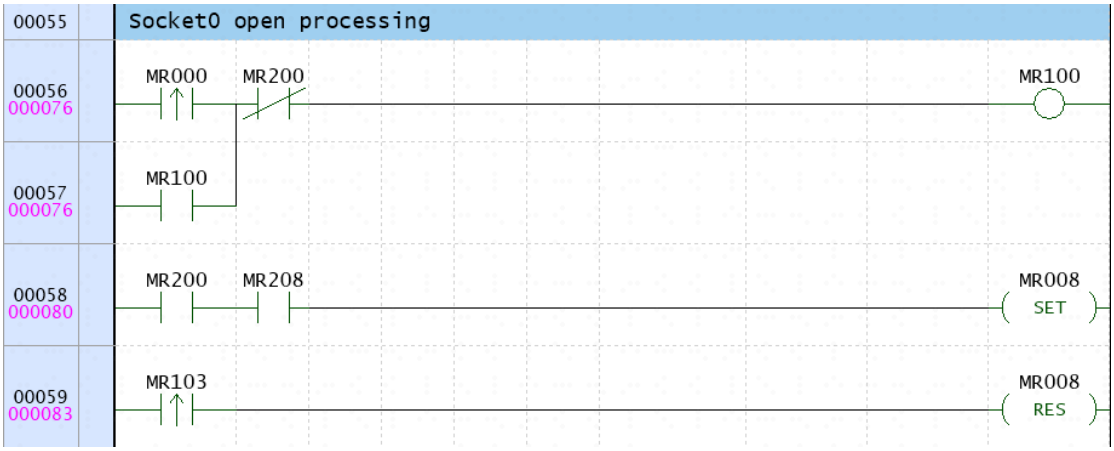
Function block pin	Name	Device address	Sample value	Available value range
wFuncCD	Function code	DM20	$0004	$01/02/03/04/05/06/0F/10
wXferSize	Transfer data size[Word]	DM22	120	1-123
wDataBlkAddr	Data Block address	DM23	768	0-65535

The above Modbus parameters are used to read the result data from the 120 words starting from 0x300 in the Modbus slave input register.

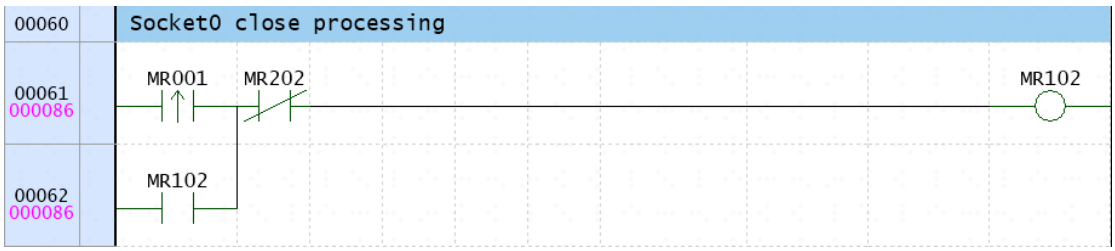
In line 46 of the program, the data pointer DM24 is used to store the read/written data in EM0~EM123, and the data in the data storage area is cleared each time the device is powered on to avoid data interference.

```
00046
IF CR2008 THEN
    ADRSET(DM0,DM24.D);
    FMOV(0,*DM24,124);
END_IF;
```

Starting from line 55 of the program: Trigger MR000 to turn on MR100 (TCP open request socket) and establish a network connection request with the camera.

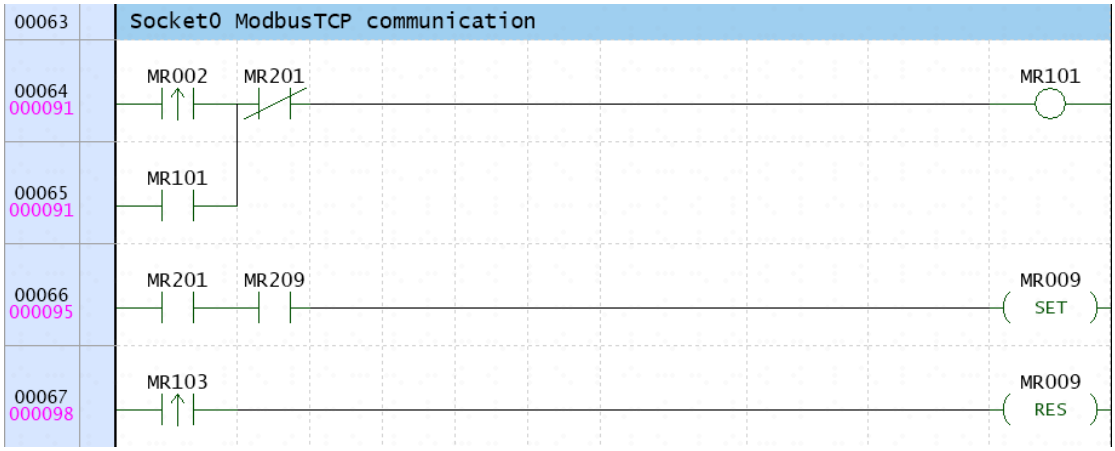


Starting from line 60 of the program: Trigger MR001 to turn on MR102 (TCP close request socket) and close the network connection with the corresponding device.



Function Block pin	Name	Statements
arrxReq	Request Relay	MR100: TCP open request, MR101: Modbus Communication request MR102: TCP Close request, MR103: Clear errors MR104~115 Reserve for the system;
arrxAck	Response Relay	MR200: TCP opening finish, MR201: Modbus Communication finish MR202: TCP Close finish, MR203: Modbus Communication ready MR204-MR207: Reservation MR208: TCP Open Error, MR209: Modbus Communication Error MR210~215 Reserve for the system;

Starting from line 63 of the program: Trigger MR002 to turn on MR101 (Modbus communication request), and read or write the slave station data according to the set communication content: function code + data.



Function Description: When executing a Modbus/TCP communication request via MBCL's FB, first confirm the TCP connection status of both parties. If the connection status is Established, data is sent directly. If the TCP connection has not been established, the TCP open process is executed first. Once the connection is established, Modbus/TCP data is sent.

If a connection has already been established, there is no need to trigger the TCP open request on line 59 of the program again. Each time data is sent, only the Modbus/TCP communication request on line 67 of the program needs to be triggered.

Description of other pins of the function block:

Function Block pin	Name	Device	Statements
wErrCD	Error code	DM26	Range 0~65535
wStatus	Connection status	DM27	0：Close；4：Connected
dwComTime	Communication monitoring time	DM28~DM29	Time elapsed from Modbus/TCP communication request ON to communication completion

Function Block Error codes：

Error codes	Error information	Reasons/Countermeasures
0	No Error	
30	Incorrect IP address or port number settings on the other side	Please confirm the IP address and port number of the other party set in arrComCfg.
31	Local port number is duplicated	Please confirm the local port number setting in arrComCfg.
38	TCP close request was executed without establishing a TCP connection.	Please confirm the timing of the TCP close request trigger.
39	TCP close request was executed before Modbus/TCP communication was completed.	Please confirm the timing of the TCP close request trigger.
40	Modbus/TCP communication timed out.	Please confirm the timeout setting in arrComCfg and the communication status set by the other party.
41	Modbus/TCP communication request failed	Please confirm the communication status set by the other party and whether a TCP connection has been established.
42	TCP open request failed	Please confirm the communication status set by the other party; confirm the communication path (e.g., network cable); check if the PLC socket port number is in conflict.
43	TCP abnormally disconnected	Please confirm the communication status set by the other party; confirm the communication path (e.g., network cable).
200	Local IP address error	Please confirm the IP address setting in the unit editor.
60001~60016	Modbus/TCP server returned an incorrect response	Please confirm the specifications of the Modbus/TCP server (whether the function code and address range are correct); refer to the error codes 01–16 in the other party's device manual.
65532	Data address error (pointer setting error)	Please confirm whether the ADRSET instruction is correct.
65533	Data transfer size error	Please confirm the value of wXferSize; the

		correct range is 1–123.
65534	Function code error.	Please verify the value of wFuncCD. Supported function codes are 01H, 02H, 03H, 04H, 05H, 06H, 0FH, and 10H.
65535	TCP open timeout.	Please verify the communication status set by the other party; verify the communication path (e.g., network cable); and verify the timeout setting value in arrComCfg.

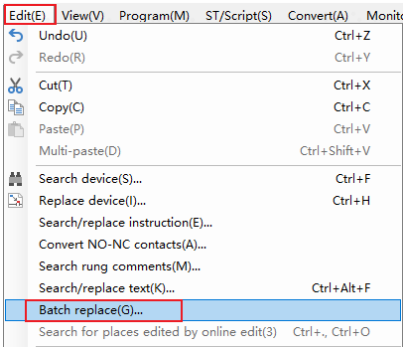
TCP Connection status code：

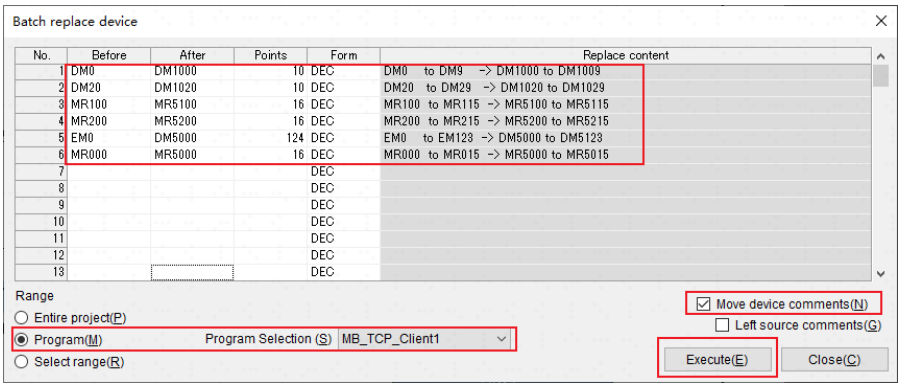
Code	Status	Statements
0	CLOSED	Closed state.
1	LISTEN	Connected state.
2	SYN SENT	Active open state after sending SYN.
3	SYN RCVD	State after the server receives SYN and sends a response.
4	ESTABLISHED	State after the connection is established.
5	CLOSE WAIT	State after receiving FIN and waiting for termination.
6	FIN WAIT1	State after sending FIN after termination.
7	FIN WAIT2	State after both the FIN server and client receive FIN.
8	CLOSING	State after both the server and client receive FIN.
9	LAST ACK	State after receiving FIN from the communication target, terminating, and sending FIN.

Modbus/TCP communication can only be performed when the TCP connection status code is 4, i.e., when the connection has been established.

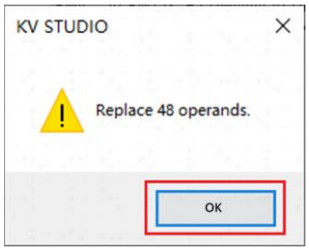
Since the ModbusTCP communication between this case and the vision sensor requires reading result data and sending trigger commands, two official function blocks from Keyence must be used. As a result, the soft component addresses in the official example program will conflict with this case, so they must be replaced with unused soft component addresses.

Procedure: First, import the first official example program into this program. Click the top menu bar - Edit - Batch Replace. In the pop-up window, enter the “Before Replacement,” “After Replacement”, and “Number of Points”. Check the [Move device comments] option in the bottom-right corner, then click [Execute].

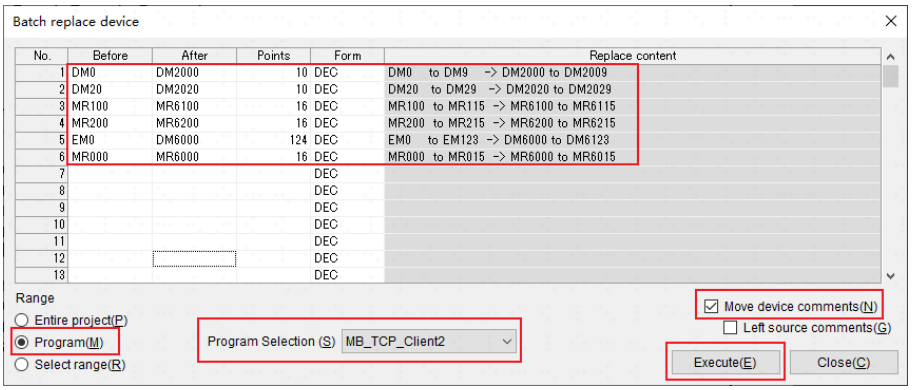




After the replacement is successful, the following window will appear. Click [OK].



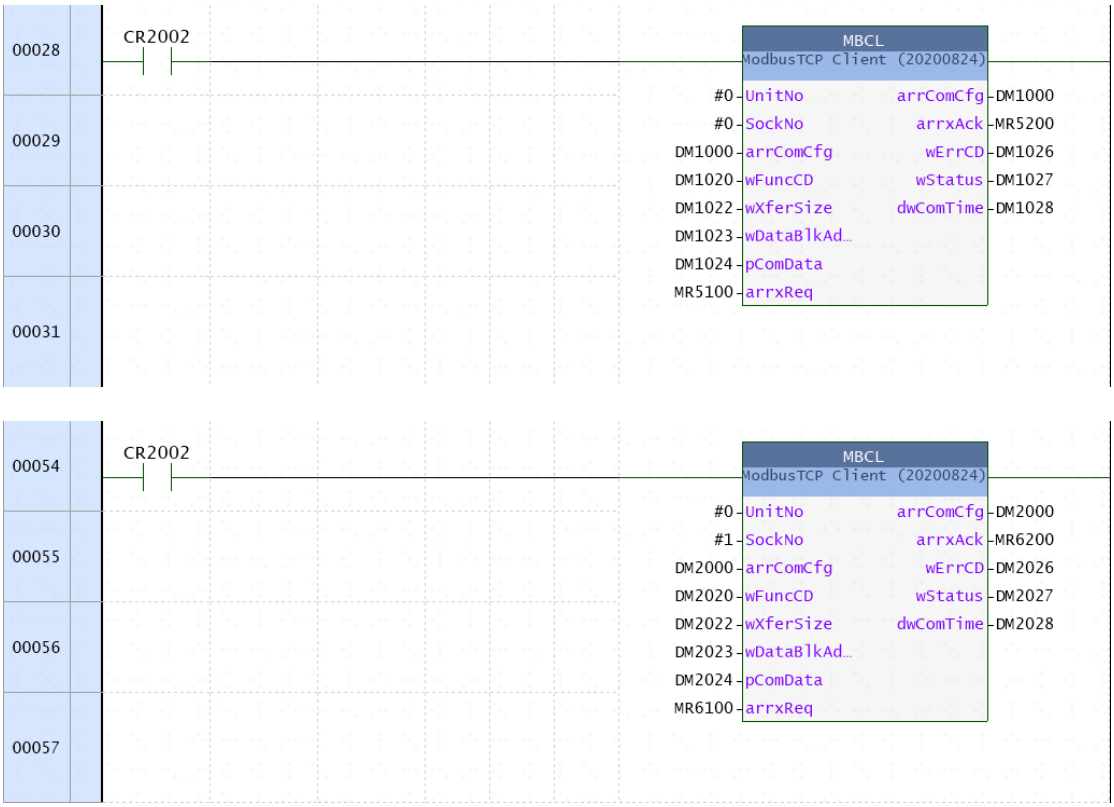
Re-import the official program, replace the soft component address, import the official program twice, and modify the actual network parameters and communication content according to the ModbusTCP address of the vision sensor.



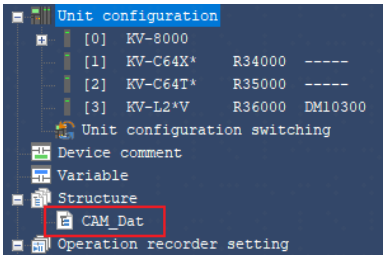
Note:

Modify the SockNo pin of the function block. Each function block needs to be filled with a different socket number. Fill in the 0/1 socket numbers that have been selected in the above steps into the pin.

For the port number pin of the function block, if it is connected to the same device, you can fill in the same soft component address or different soft component addresses with the same parameter values.



Right-click [Structure] in the project tree on the left, select [Build], and create the following data types.

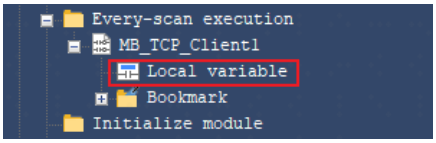


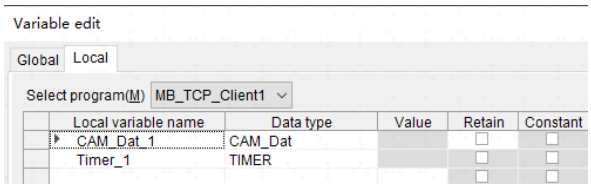
Structure[CAM_Dat]

Structure size: 24 words

Member name	Data type
Trigger	BOOL
StatisticsReset	BOOL
Switch_ProjectID	UINT
RunningTotal	UDINT
LineExist_StartPointX	REAL
LineExist_EndPointX	REAL
GrayscaleArea_Num	UDINT
OCR_StringLength	UDINT
OCR_StringContent	STRING[20]

Create the following variables in the local variables of the main program.





The modified official procedure is as follows:

MB_TCP_Client1

```
00001  ModbusTCP communication sample program (Rev.2.0)
       Supported unit: KV-8000/7500/KV-XLE02/KV-LE21V
       ****************************************
       Supported function code
           0x01    Read Coils
           0x02    Read Discrete Inputs
           0x03    Read Holding Registers
           0x04    Read Input Registers
           0x05    Write Single Coil
           0x06    Write Single Register
           0x0F    Write Multiple Coils
           0x10    Write Multiple Registers
       ****************************************
00002  MR000↑ Open req        Follow communication setting to open TCP socket port
       MR001↑ Close req       Close TCP socket port
       MR002↑ Modbus comm req Follow the setting in D20 to D25 to communicate
       MR003↑ Error clear
       MR007   Open error
       MR007   Modbus comm error
00003  D0~D9   Communicatino setting
       D20     Function code
       D21     Transfer data qty[Word]
       D22     Data Block leading address
       D24,25  PLC data transfer address (specified by ADRSET instruction)
00004  E0~     communicatino data (up to 123 words) (Address for storing is set in D24/25)
```

```
00005  Initial processing
00006
00007  //***************************************************
       //
       IF CR2008 THEN
           DM1000 := 0;
           DM1001 := 192;
           DM1002 := 168;
           DM1003 := 1;
           DM1004 := 108;
           DM1005 := 502;
           DM1006 := 255;
           DM1007 := 1000;
           DM1008 := 1000;
           DM1009 := 1000;
       END_IF;
00034  Function code & transfer setting (Initial value)
00035
       IF CR2008 THEN
           DM1020 := 16#0003;
           DM1022 := 20;
           DM1023 := 768;
       END_IF;
```

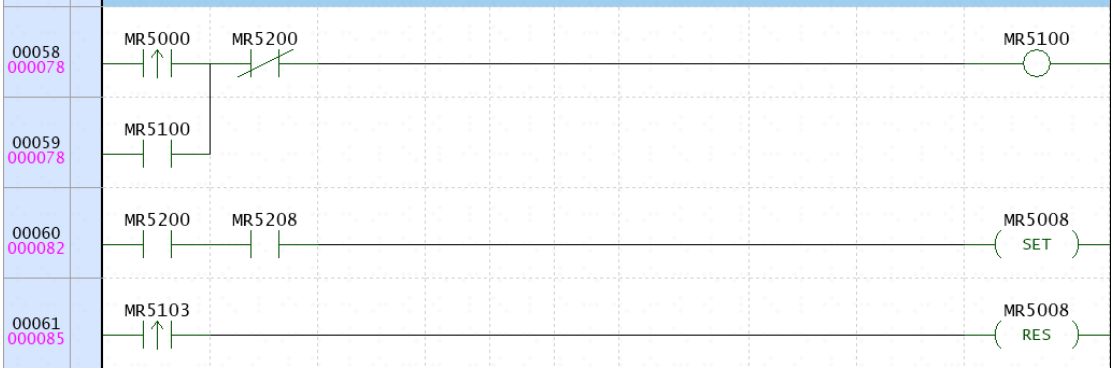
00046 Specify the storage of communication data in EM0 (ADRSET instruction)

00047 Initialize data area

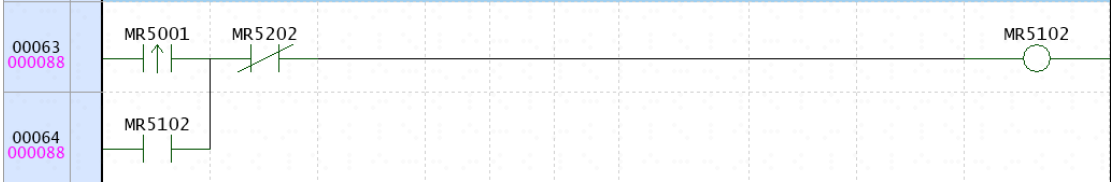
00048

```
IF CR2008 THEN
    ADRSET(DM5000,DM1024.D);
    FMOV(0,*DM1024,124);
END_IF;
```

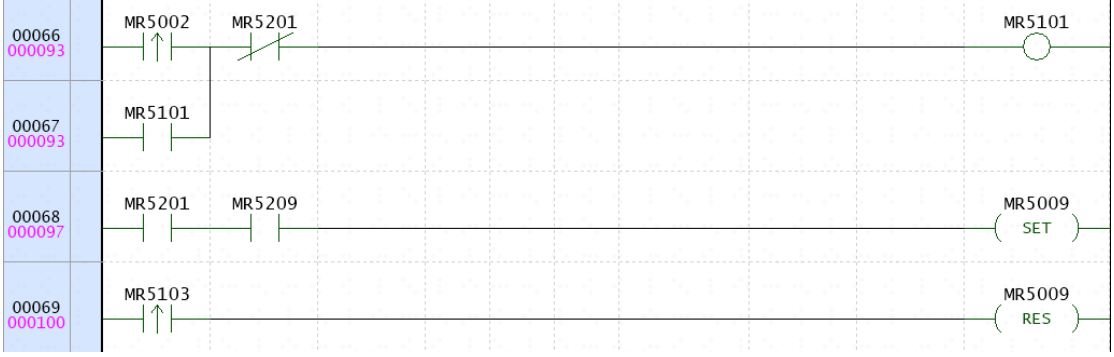
00057 Socket0 open processing



00062 Socket0 close processing

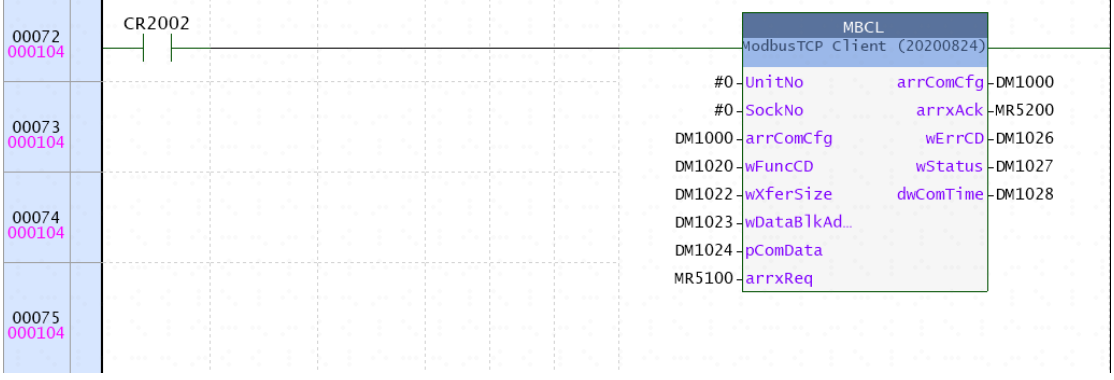


00065 Socket0 ModbusTCP communication



00070 ModbusTCP Client Function block (condition is always on)

00071 Supported unit: KV-8000/7500/KV-XLE02/KV-LE21V



00076 ********

```
00077 //***************************************************
      //
      IF CR2008 THEN
          DM2000 := 0;
          DM2001 := 192;
          DM2002 := 168;
          DM2003 := 1;
          DM2004 := 108;
          DM2005 := 502;
          DM2006 := 255;
          DM2007 := 1000;
          DM2008 := 1000;
          DM2009 := 1000;
      END_IF;
```

00104 Function code & transfer setting (Initial value)

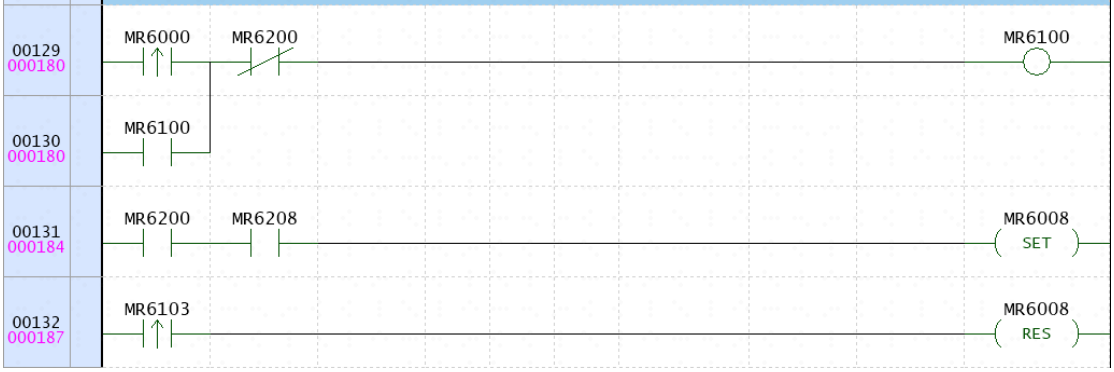
```
00105 //
      IF CR2008 THEN
          DM2020 := 16#0010;
          DM2022 := 3;
          DM2023 := 256;
      END_IF;
```

00117 Specify the storage of communication data in EM0 (ADRSET instruction)

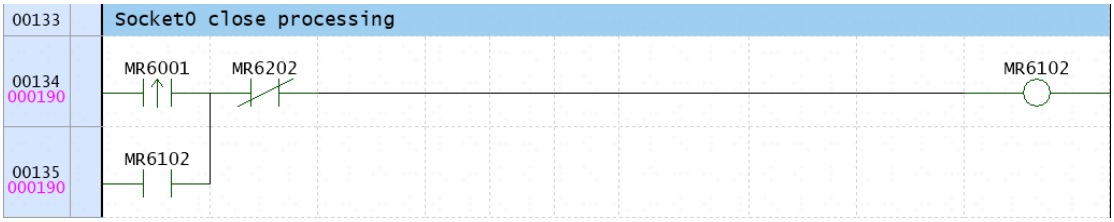
00118 Initialize data area

```
00119 IF CR2008 THEN
          ADRSET(DM6000,DM2024.D);
          FMOV(0,*DM2024,124);
      END_IF;
```

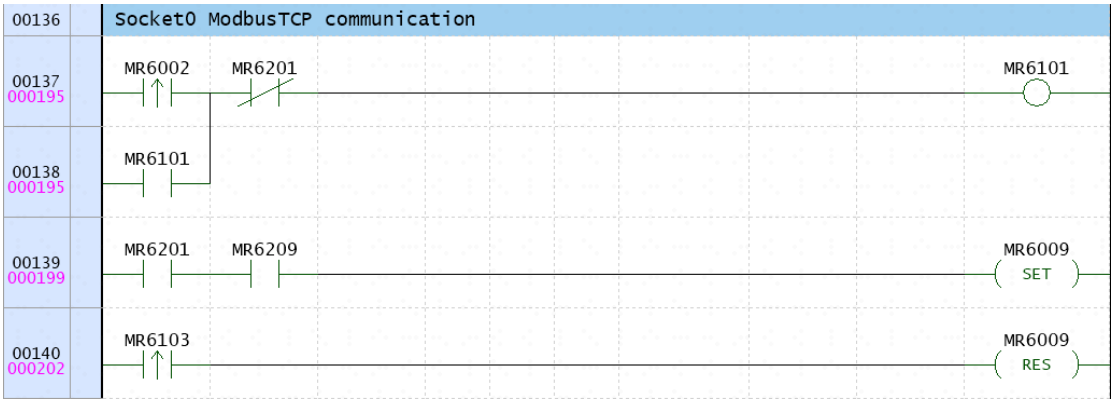
00128 Socket0 open processing

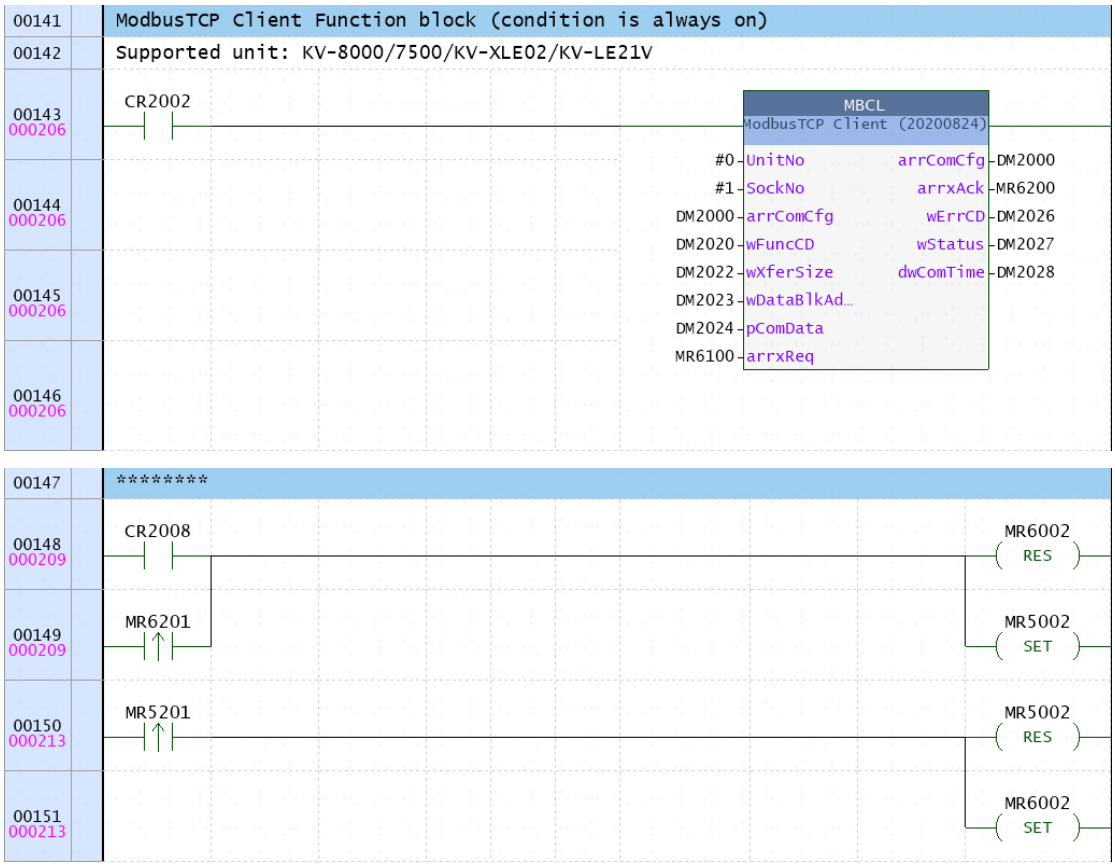


00133 Socket0 close processing



00136 Socket0 ModbusTCP communication





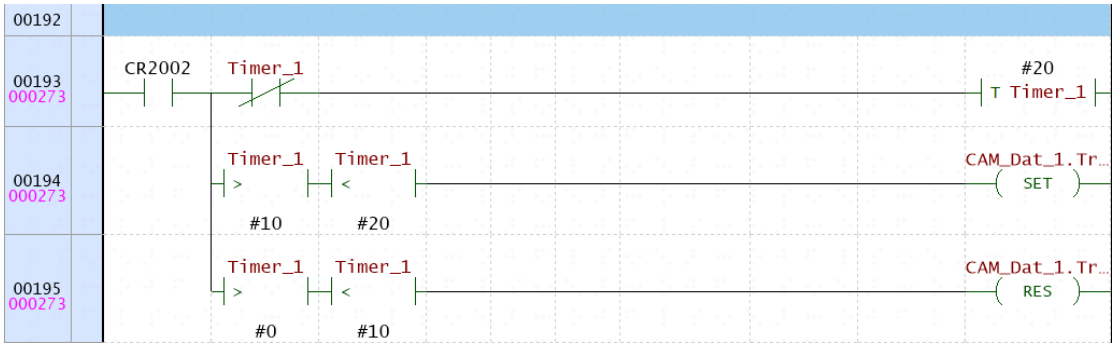
Lines 147-151 of the program: Since the official case only involves communication between a single device and a single function, multiple function blocks must be used to read vision sensor data and send commands without conflicting with each other. Therefore, a polling program must be written to implement the loop read/write function.

00152

```
//
IF CAM_Dat_1.Trigger THEN
    DM6000 := 1;
ELSE
    DM6000 := 0;
END_IF;

IF CAM_Dat_1.StatisticsReset THEN
    DM6001 := 1;
ELSE
    DM6001 := 0;
END_IF;

DM6002 := CAM_Dat_1.Switch_ProjectID;

//
CAM_Dat_1.RunningTotal := DM5000.D;
CAM_Dat_1.LineExist_StartPointX := DM5002.F;
CAM_Dat_1.LineExist_EndPointX := DM5004.F;
CAM_Dat_1.GrayscaleArea_Num := DM5006.D;
CAM_Dat_1.OCR_StringLength := DM5008.D;
CAM_Dat_1.OCR_StringContent := DM5010.T;
```

00153

Line 153 of the program: The [Trigger] and [Statistics Reset] configured in the vision sensor format input string occupy 1 word of space in ModbusTCP communication even though the data type is Bit. Therefore, it is necessary to convert Bit and Word in the program, and then map the received and sent storage areas to newly created local variables.

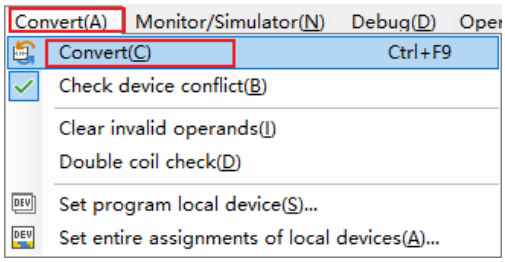


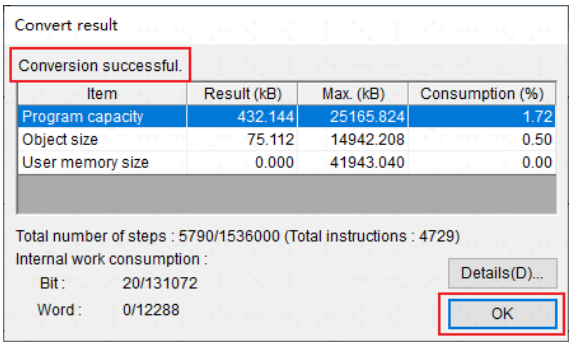
Lines 193-195 of the program: To facilitate testing, a timer instruction is used to periodically send trigger commands to the vision sensor (the vision sensor is configured in single-frame or level mode). When it is necessary to modify the timing interval, the timing interval of the timer instruction can be adjusted as required. In this example, the sensor is triggered to take a photo every 2 seconds. If this functionality is not required, the normally open contact of the CR2002 relay in line 193 of the program can be switched to a normally closed contact to disable this functionality. After modification, the PLC program must be re-downloaded for the changes to take effect.

2.2 Communication Test

2.2.1 Download program

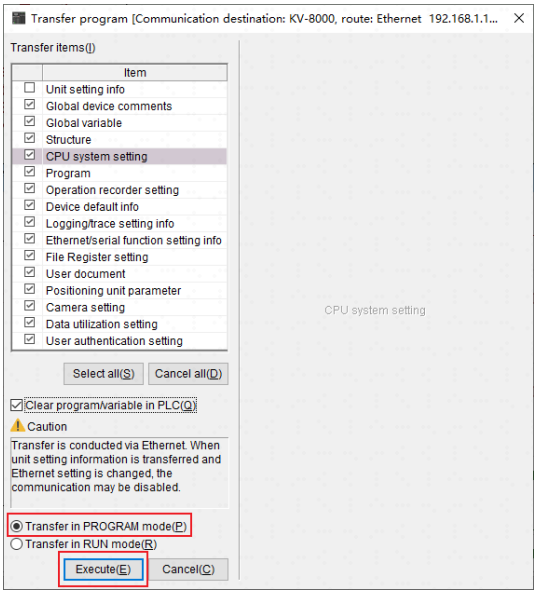
Click [Convert] - [Convert] in the menu bar above to compile the written program. After the conversion is successful, click [OK].



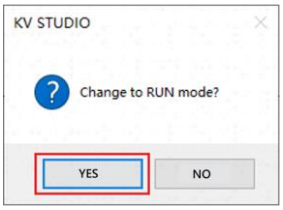


Click PLC Transmission  in the menu bar, select [Transfer in PROGRAM mode] in

the pop-up dialog box, and click [Execute].

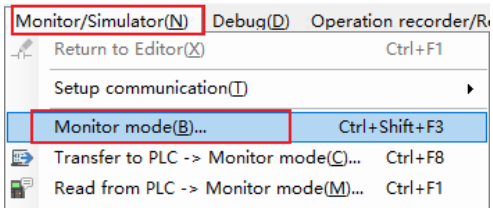


[Change to RUN mode] Choose Yes.

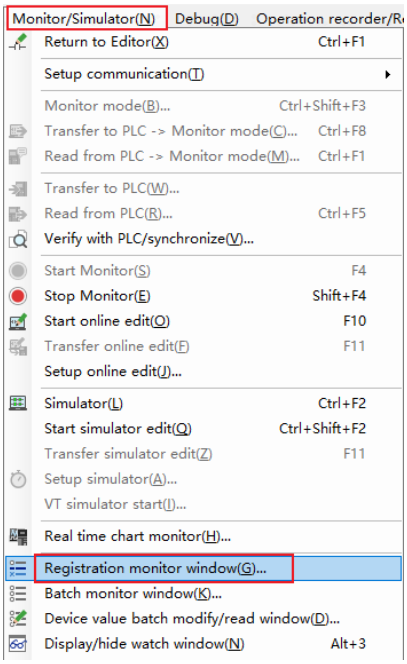


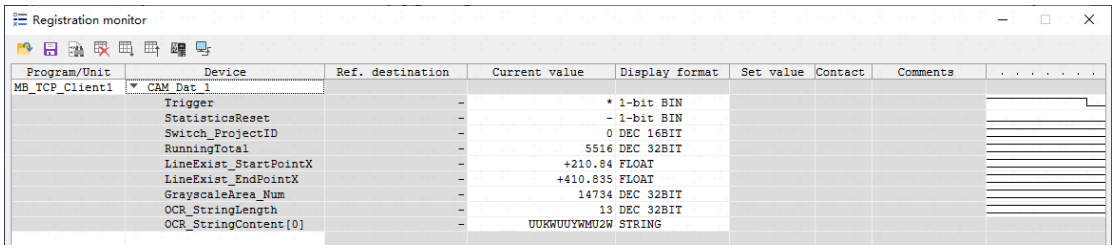
2.2.2 Communication Test

Select [Monitor/Simulator] - [Monitor Mode] from the menu bar at the top, and set the program to online status.



Select [Monitor/Simulator] - [Registration Monitor Window] from the menu bar at the top, and add the following variables in the monitor window.





When manually triggering the vision sensor to take a photo, set the value of the [CAM_Dat_1.Trigger] variable to [TRUE]. The PLC will send a trigger command to the vision sensor, which will then take a photo and send the results back to the PLC. To reset the total count, set the value of the [CAM_Dat_1.Statistics_Reset] variable to [TRUE]. When switching engineering IDs, it is necessary to configure the communication settings for the engineering ID to be switched to the same communication protocol in advance. Otherwise, after the switch, the PLC and the vision sensor will lose connection. After the switch is successful, the vision sensor interface must be manually refreshed to display the engineering ID after the switch.

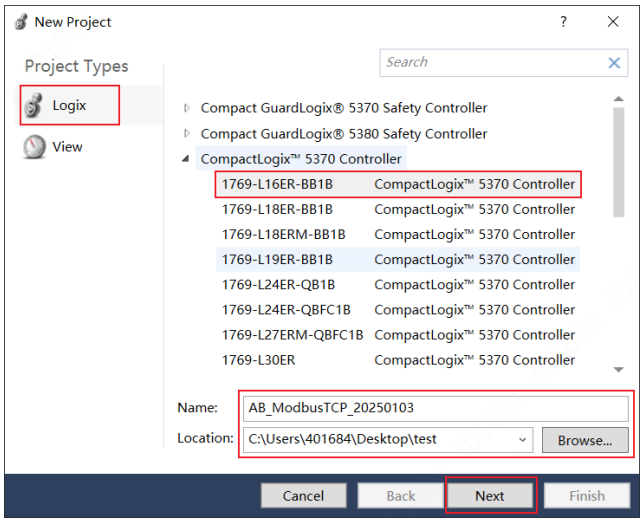
Note: The above PLC configuration and program are only applicable to the current case. Users should modify them according to project requirements when using them.

3 Rockwell PLC Configuration

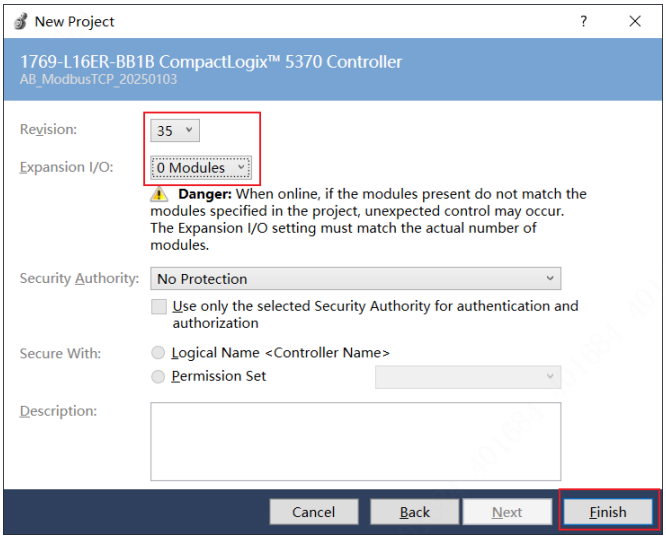
3.1 PLC Configuration

3.1.1 Create New Project

Open the Studio 5000 software, click Project - New, select [Logix] - [CompactLogix 5370 Controller] - [1769-L16ER-BB1B] as the project type, modify the project name and file location, and click Next.

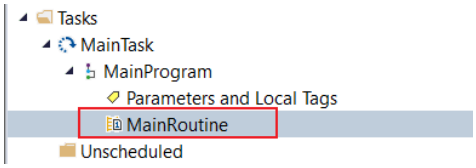


Select the Revision number and number of expansion IO modules based on the PLC used in the project, then click “Finish”.

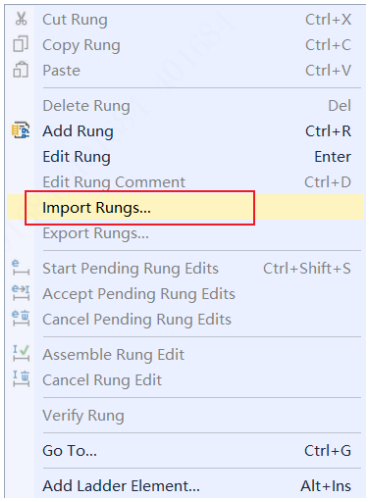


3.1.2 Write the PLC program

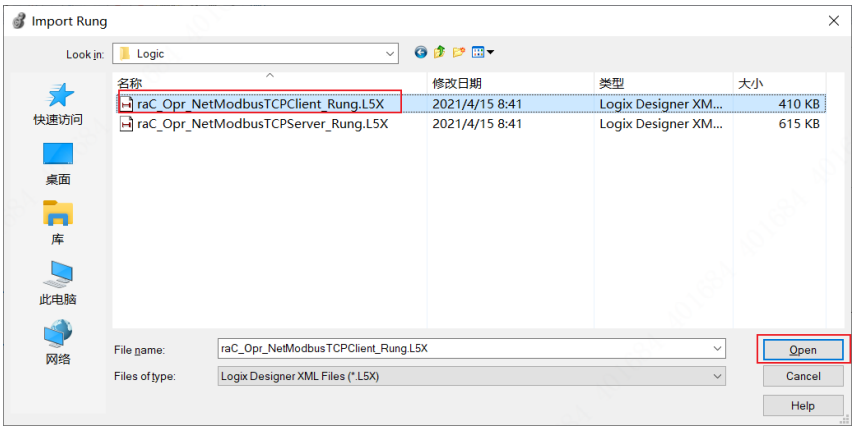
In the project tree, select [Tasks] - [MainTask] - [MainProgram] - [MainRoutine], then double-click to enter the ladder diagram programming interface.



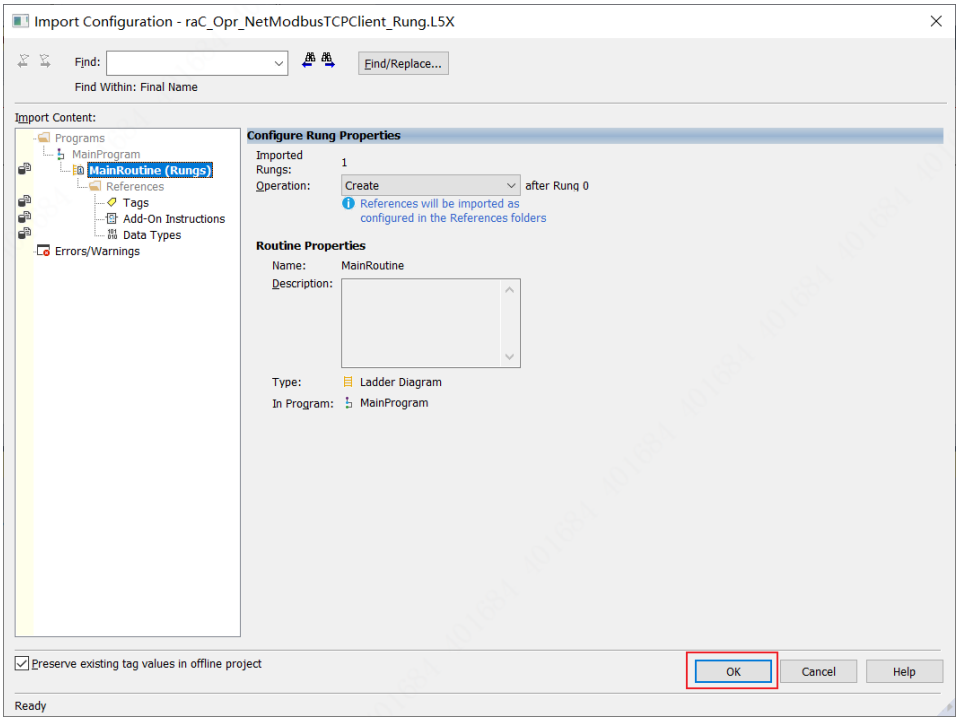
Right-click on the ladder diagram and select [Import Rungs].



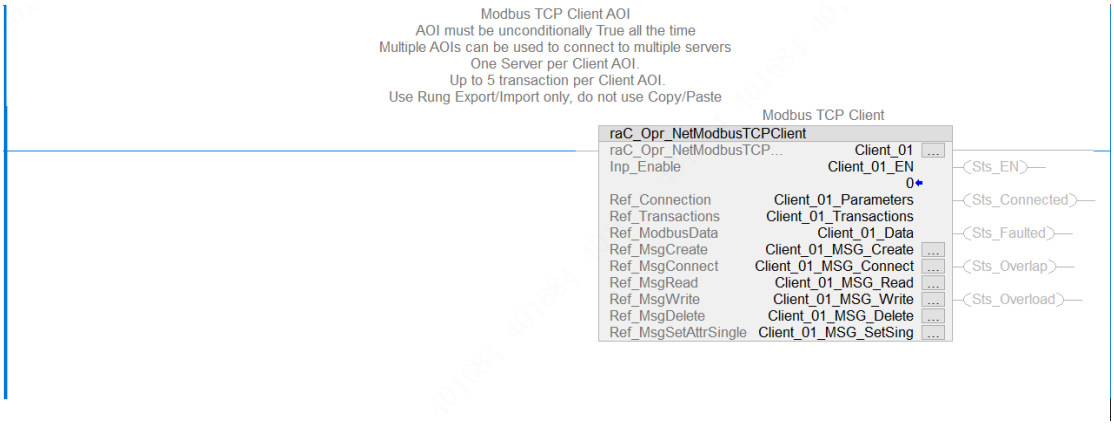
Find the ModbusTCPClient library file provided by Rockwell, select it, and open it.



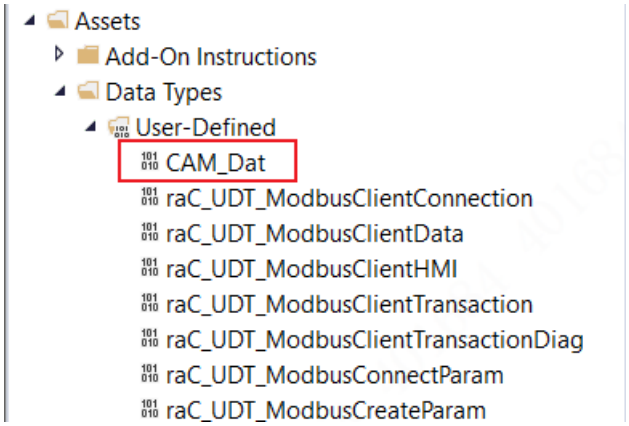
Click OK

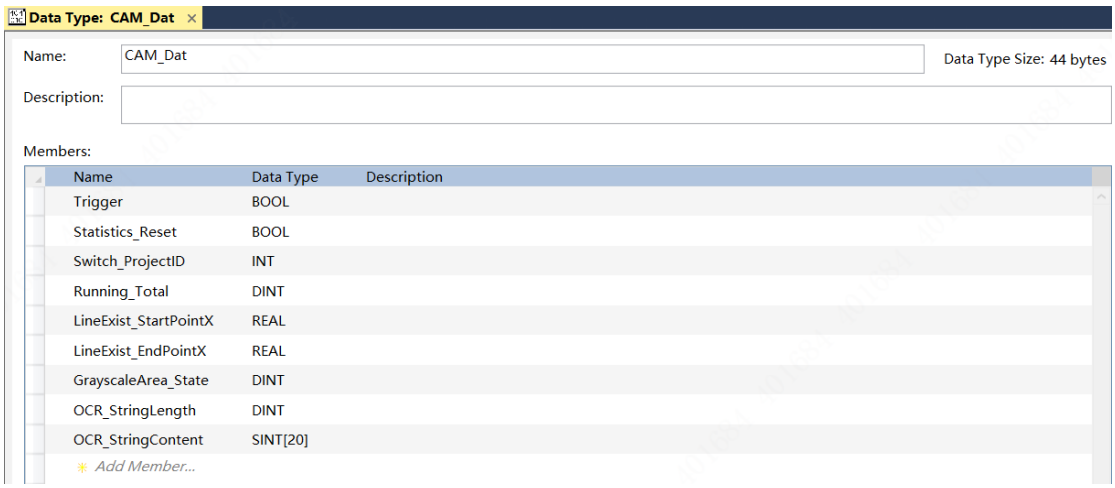


After successful import, the following function blocks will be displayed in the ladder diagram.

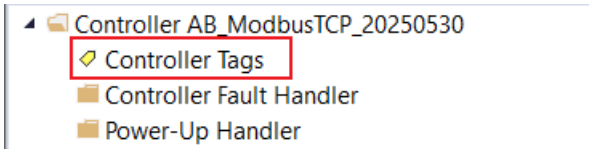


In the project tree, select [Assets] - [Data Types] - [User Defined], and create the following data types.





In the [Controller Tags], add the following variables:

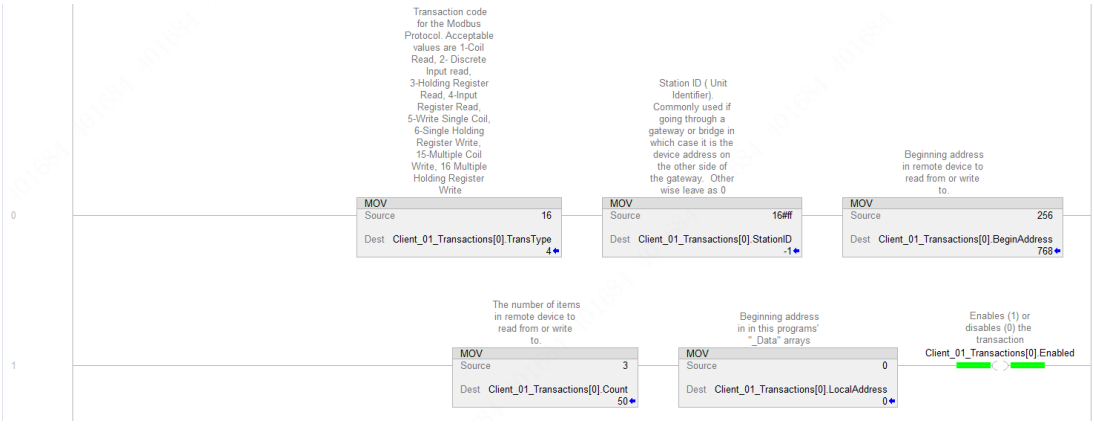


Controller Tags - AB_ModbusTCP_20250530(controller)

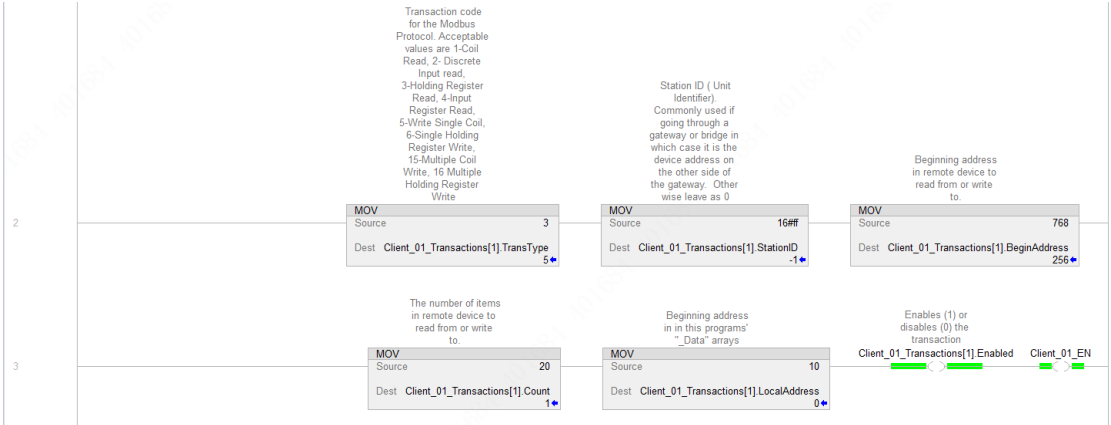
Scope: AB_ModbusTCP_2 Show: All Tags Enter Name Filter...

Name	Alias For	Base Tag	Data Type	Description	External Access	Constant	Style
TIMER_1			TIMER		Read/Write	☐	
ONS_2			BOOL		Read/Write	☐	Decimal
ONS_1			BOOL		Read/Write	☐	Decimal
Local:1:O			AB:Embedded_DiscreteIO:O:0		Read/Write	☐	
Local:1:I			AB:Embedded_DiscreteIO:I:0		Read/Write	☐	
Local:1:C			AB:Embedded_DiscreteIO:C:0		Read/Write	☐	
Client_01_Transactions			raC_UDT_ModbusClientTransaction[5]		Read/Write	☐	
Client_01_Parameters			raC_UDT_ModbusClientConnection		Read/Write	☐	
Client_01_MSG_Write			MESSAGE		Read/Write	☐	
Client_01_MSG_SetSing			MESSAGE		Read/Write	☐	
Client_01_MSG_Read			MESSAGE		Read/Write	☐	
Client_01_MSG_Delete			MESSAGE		Read/Write	☐	
Client_01_MSG_Create			MESSAGE		Read/Write	☐	
Client_01_MSG_Connect			MESSAGE		Read/Write	☐	
Client_01_EN			BOOL		Read/Write	☐	Decimal
Client_01_Data			raC_UDT_ModbusClientData		Read/Write	☐	
Client_01			raC_Opr_NetModbusTCPClient	Modbus TCP Client	Read/Write	☐	
CAM_1			CAM_Dat		Read/Write	☐	
						☐	

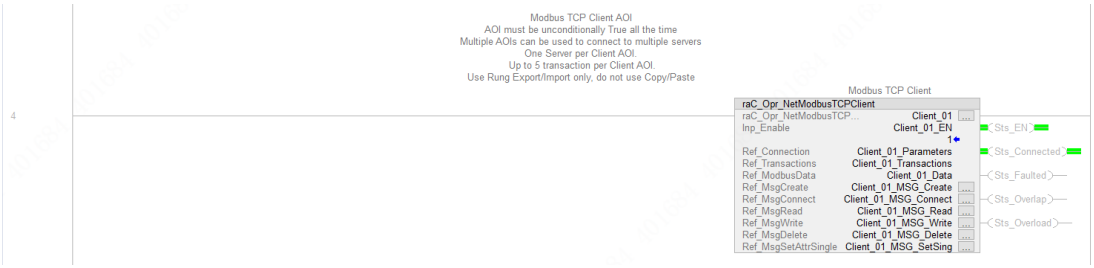
Add the following program to the main program:



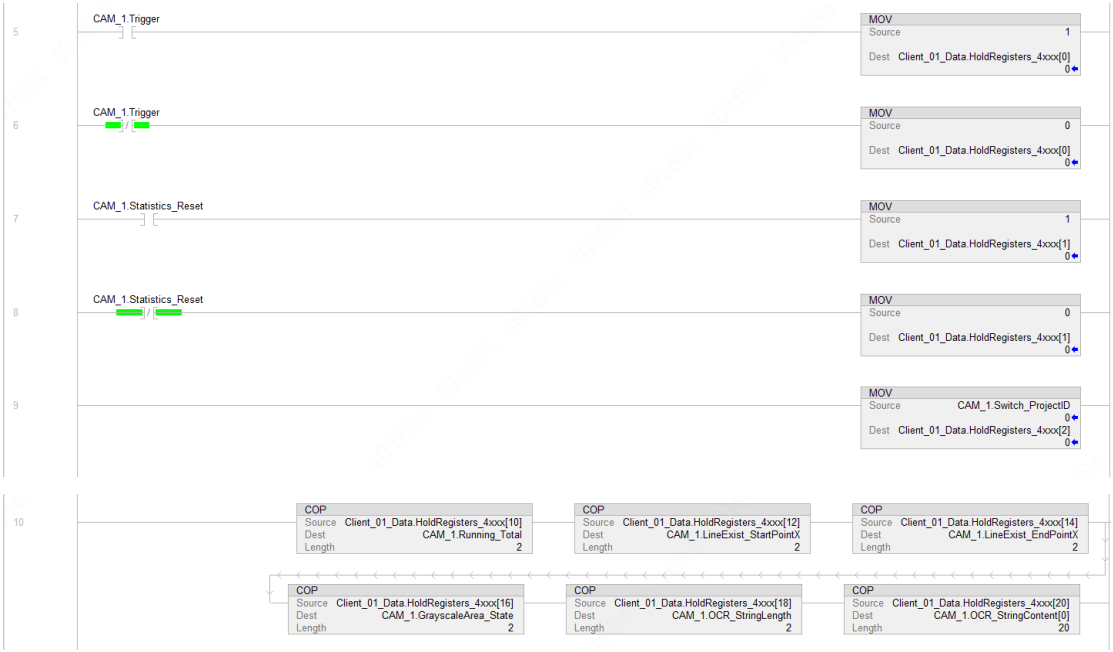
Program lines 0-1: Set the Modbus TCP function code for sending control data to the vision sensor to 16, the slave ID to 16#FF, and the starting address of the vision sensor's hold register to 0x100. which is 256 in decimal, with a write length of 3 bytes. The data to be sent is stored in the address starting from the first integer variable in the PLC controller tag array [Client_01_Data.HoldRegisters_4xxx], and the enable bit is set to 1.



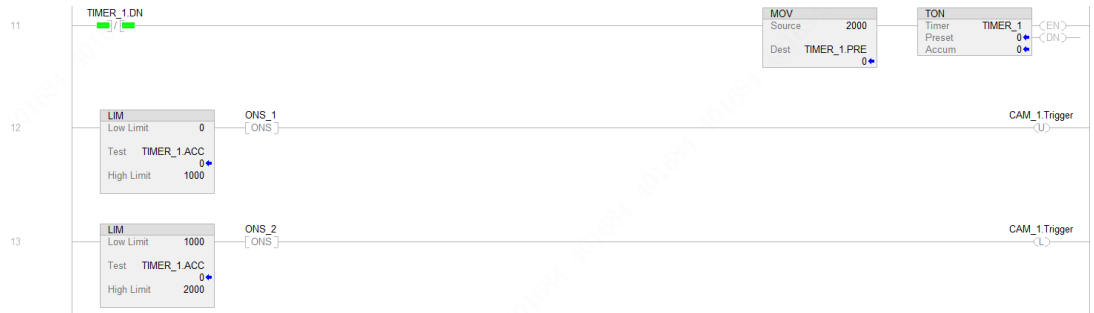
Lines 2-3 of the program: Set the ModbusTCP function code for receiving data from the vision sensor to 3, the slave ID to 16#FF, the starting address for receiving data to 0x300 (decimal representation: 768), and the reception length to 20. The received data is stored in the 10th integer variable of the PLC controller tag [Client_01_Data.HoldRegisters_4xxx] array, and the enable flag is set to 1.



Line 4 of the program: Import the official ModbusTCP function block.



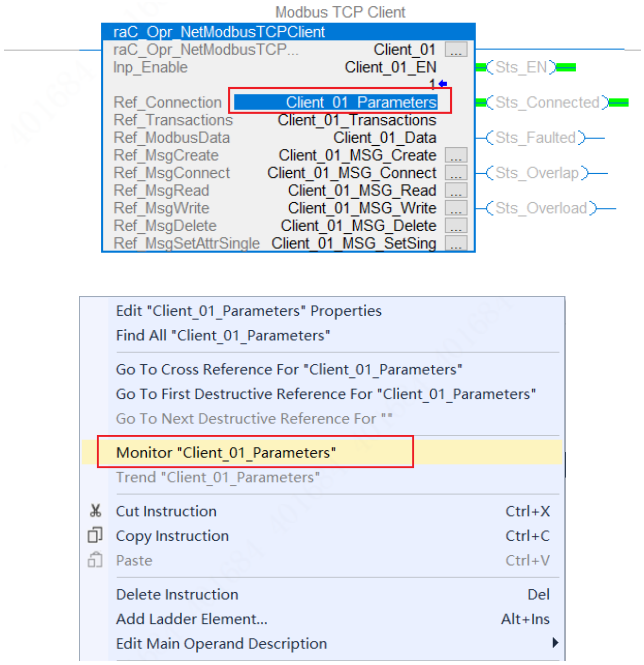
Lines 5-10 of the program: Map the data read and sent by the vision sensor to the global variables created in the previous step.



Lines 11-13 of the program: To facilitate testing, trigger commands are sent at regular

intervals using timer instructions. When you need to modify the timing, you can change the value of the [TIMER_1.PRE] variable in line 11 of the program as required. In this example, the sensor is triggered to take a photo every 2 seconds. If this functionality is not required, modify the normally closed contact [TIMER_1.DN] in line 11 to a normally open contact. After modification, the PLC program must be re-downloaded for the changes to take effect.

In the ModbusTCP communication function block, right-click [Client_01_Parameters], select [Monitor Client_01_Parameters], and enter the monitoring interface of the controller tab.



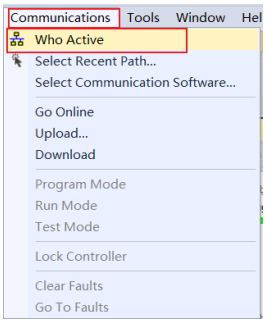
Set [Client_01_Parameters.LocalAddress] to the PLC's IP address, [Client_01_Parameters.DestAddress] to the vision sensor's IP address, and [Client_01_Parameters.DestinationPort] to the vision sensor's port number.

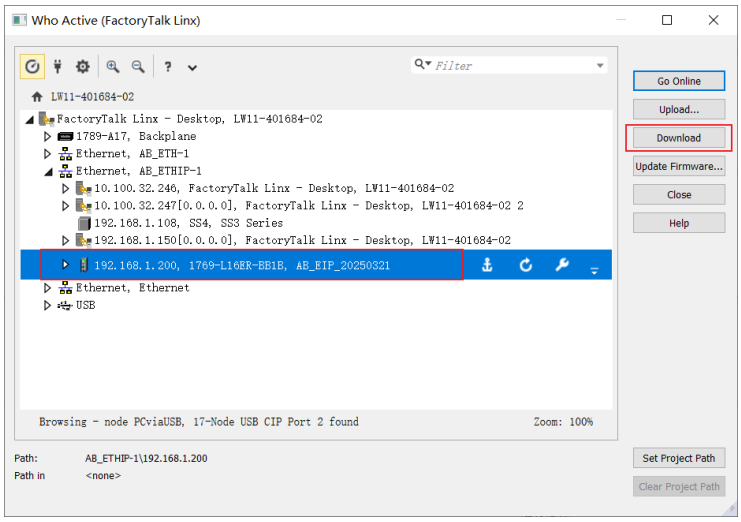
Client_01_Transactions	raC_UDT_ModbusClientTransaction[5]	{...}	{...}	
Client_01_Parameters	raC_UDT_ModbusClientConnection	{...}	{...}	
Client_01_Parameters.LocalSlot	SINT	0		Decimal
Client_01_Parameters.LocalAddress	STR0016	'192.168.1.200'	{...}	
Client_01_Parameters.DestAddress	STR0016	'192.168.1.108'	{...}	
Client_01_Parameters.DestinationPort	DINT	502		Decimal
Client_01_Parameters.MSG_Src	SINT[500]	{...}	{...}	Decimal

3.2 Communication Test

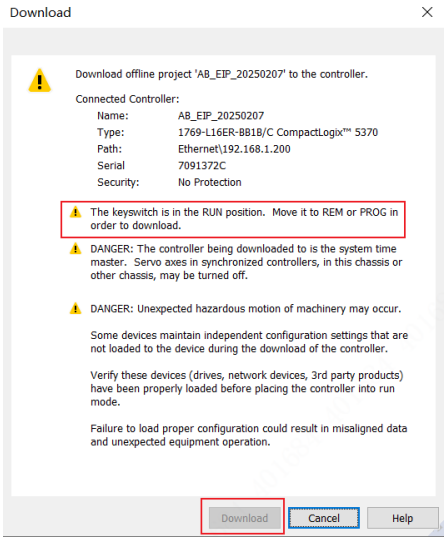
3.2.1 Download program

Select [Communications] - [Who Active], locate the PLC device in the network, and click Download.

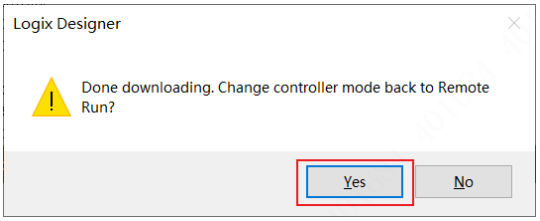




Note: If the key switch on the PLC Device is in the RUN position at this time, the download button cannot be clicked and the following message will be displayed. You need to move the key switch on the PLC body to the REM position before you can click the download button normally.

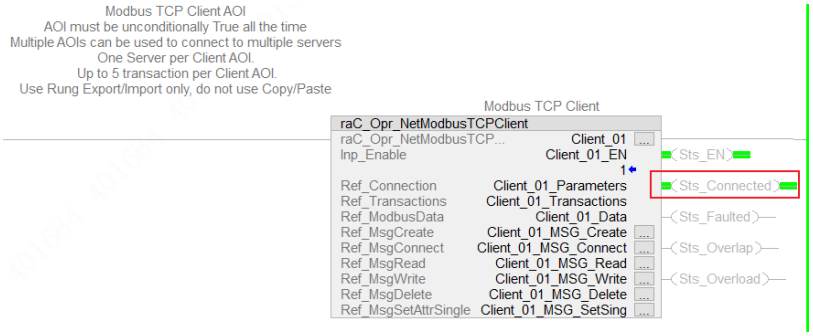


After the download is complete, the following message will appear. Click [Yes] and move the key switch on the PLC body back to the RUN position.

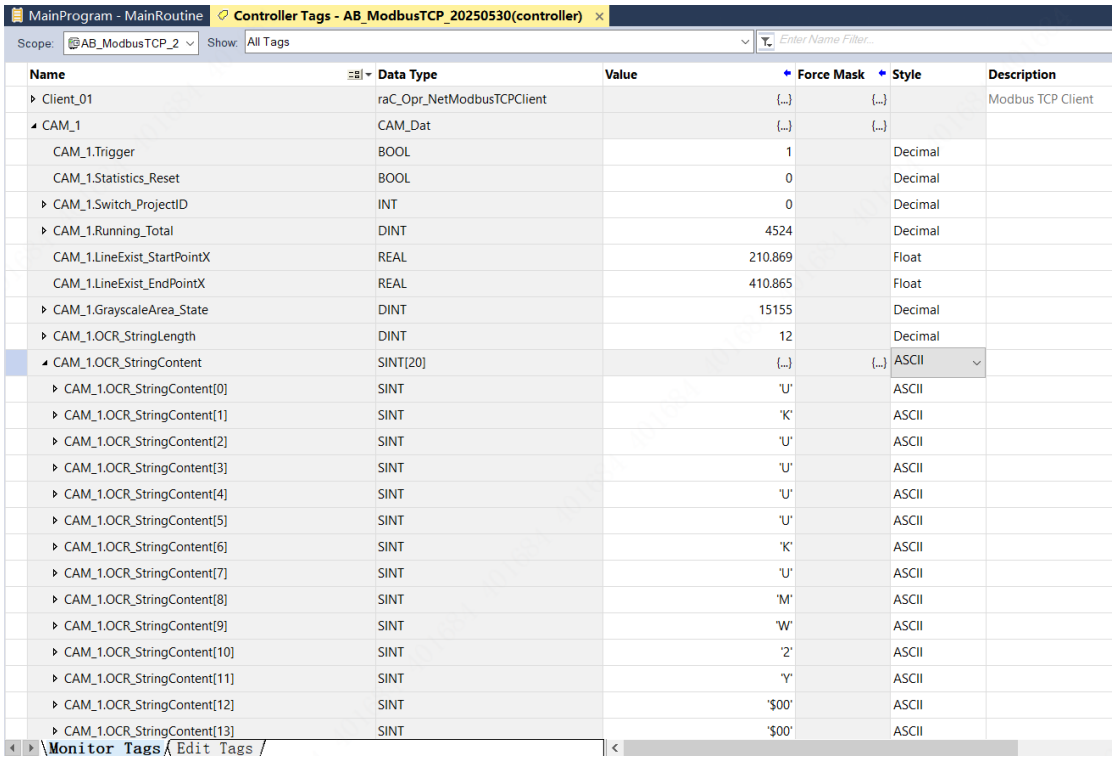


3.2.2 Communication Test

After the download is complete, the software automatically enters online mode. If the [Sts_Connected] pin of the ModbusTCP communication function block is set at this point, it indicates that the PLC has successfully communicated with the vision sensor.



In the [Controller Tags], monitor the values of the following variables:



Name	Data Type	Value	Force Mask	Style	Description
Client_01	raC_Opr_NetModbusTCPClient	{...}	{...}		Modbus TCP Client
CAM_1	CAM_Dat	{...}	{...}		
CAM_1.Trigger	BOOL	1		Decimal	
CAM_1.Statistics_Reset	BOOL	0		Decimal	
CAM_1.Switch_ProjectID	INT	0		Decimal	
CAM_1.Running_Total	DINT	4524		Decimal	
CAM_1.LineExist_StartPointX	REAL	210.869		Float	
CAM_1.LineExist_EndPointX	REAL	410.865		Float	
CAM_1.GrayscaleArea_State	DINT	15155		Decimal	
CAM_1.OCR_StringLength	DINT	12		Decimal	
CAM_1.OCR_StringContent	SINT[20]	{...}	{...}	ASCII	
CAM_1.OCR_StringContent[0]	SINT	'U'		ASCII	
CAM_1.OCR_StringContent[1]	SINT	'K'		ASCII	
CAM_1.OCR_StringContent[2]	SINT	'U'		ASCII	
CAM_1.OCR_StringContent[3]	SINT	'U'		ASCII	
CAM_1.OCR_StringContent[4]	SINT	'U'		ASCII	
CAM_1.OCR_StringContent[5]	SINT	'U'		ASCII	
CAM_1.OCR_StringContent[6]	SINT	'K'		ASCII	
CAM_1.OCR_StringContent[7]	SINT	'U'		ASCII	
CAM_1.OCR_StringContent[8]	SINT	'M'		ASCII	
CAM_1.OCR_StringContent[9]	SINT	'W'		ASCII	
CAM_1.OCR_StringContent[10]	SINT	'2'		ASCII	
CAM_1.OCR_StringContent[11]	SINT	'Y'		ASCII	
CAM_1.OCR_StringContent[12]	SINT	'$00'		ASCII	
CAM_1.OCR_StringContent[13]	SINT	'$00'		ASCII	

When manually triggering the vision sensor to take a photo, set the value of the

[CAM_1.Trigger] variable to 1. When the value of the [CAM_1.Trigger] variable changes from 0 to 1 on the rising edge, the PLC sends a trigger command to the vision sensor, which begins taking photos and outputs the results to the PLC. To reset the total count, set the value of the [CAM_1.Statistics_Reset] variable to 1. When switching engineering IDs, the communication settings for the engineering ID to be switched must be configured to the same communication protocol in advance. Otherwise, the PLC and vision sensor will disconnect after the switch. After the switch is successful, the vision sensor interface must be manually refreshed to display the new engineering ID.

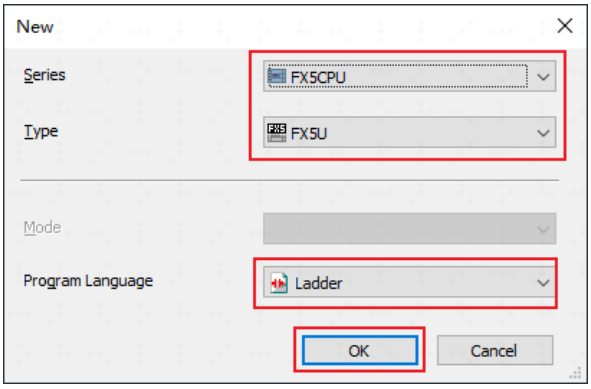
Note: The above PLC configurations and programs are only applicable to the current case. Users should modify them according to project requirements when using them.

4 Mitsubishi FX5U PLC Configuration

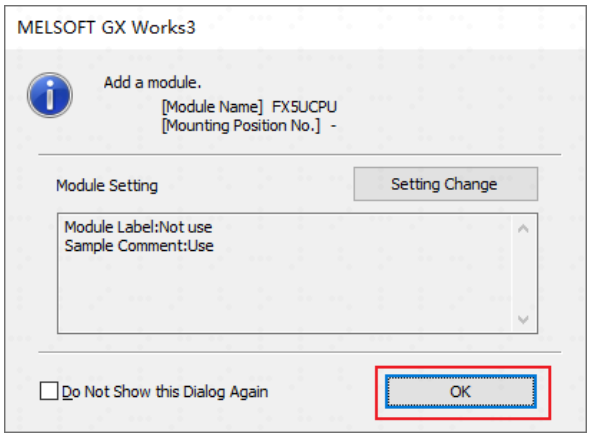
4.1 PLC Configuration

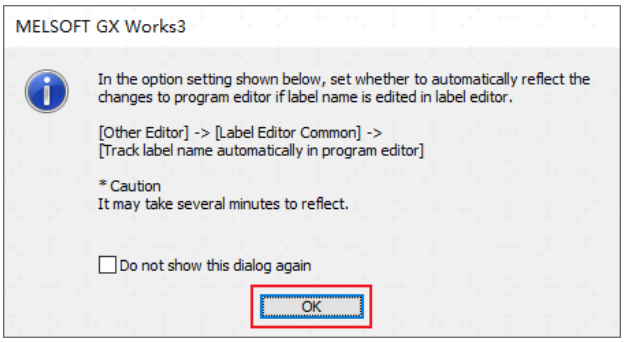
4.1.1 Create New Project

Open Mitsubishi Works3 software, click New Project, select FX5CPU series, select FX5U Type, and click OK.

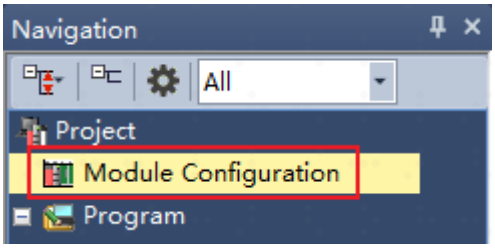


Click OK in the pop-up dialog box.

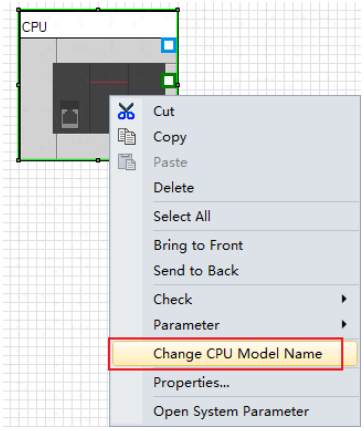




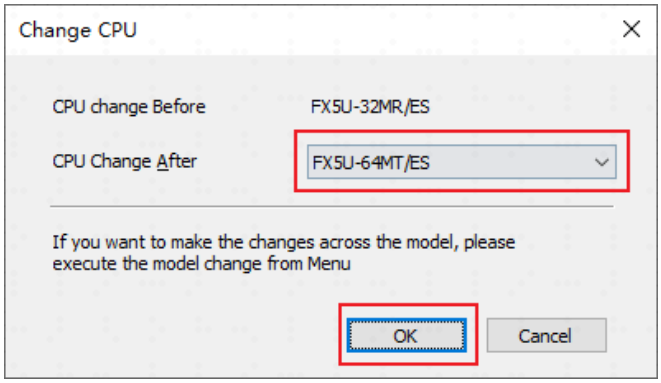
Double-click [Module Configuration] in the left navigation bar.



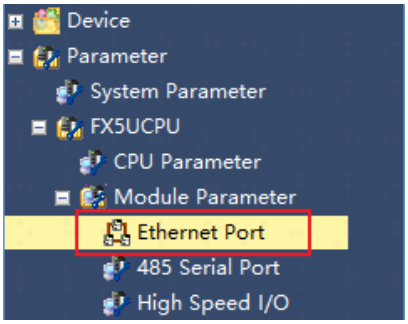
Right-click on the CPU module and click [Change CPU Model Name].



Based on the actual PLC model, [FX5U-64MT/ES] was selected for this case.

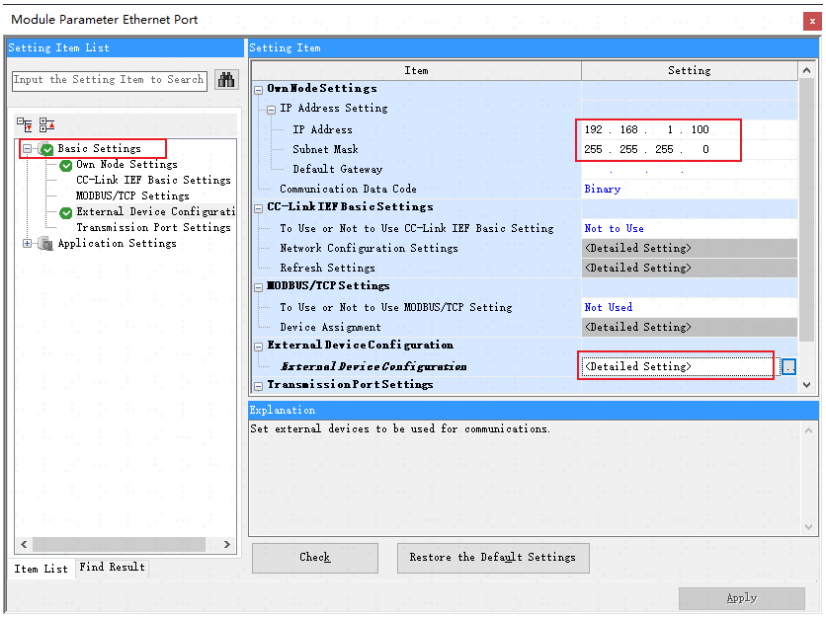


Select [Parameter] - [FX5UCPU] - [Module Parameter] - [Ethernet Port] from the left navigation bar.

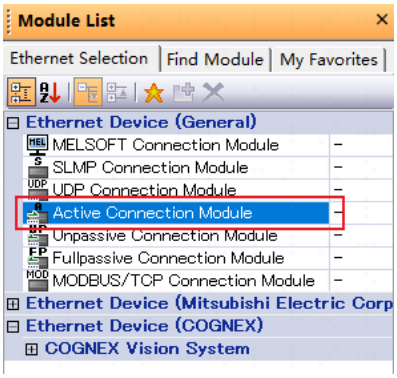


Configure the PLC's IP address, subnet mask, and other information in the basic information section. After setting it up, double-click the object device connection

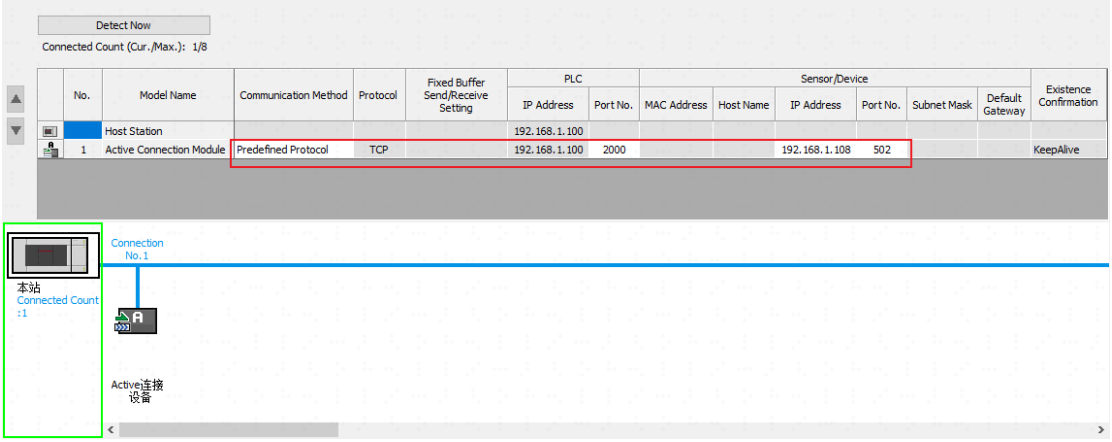
configuration settings.



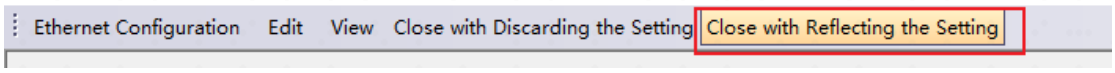
In the Ethernet Devices (General) section of the right navigation bar, select Active Connection Module and drag it to the left interface with the left mouse button.



[Communication Methods] Select the communication protocol, set the [Programmable Controller] port number to 2000 (setting range: 1–5548, 5570–65534; 5549–5569 are already in use by the system, so please do not specify them), and enter the IP address and port number of the [Sensor/Device] vision sensor.

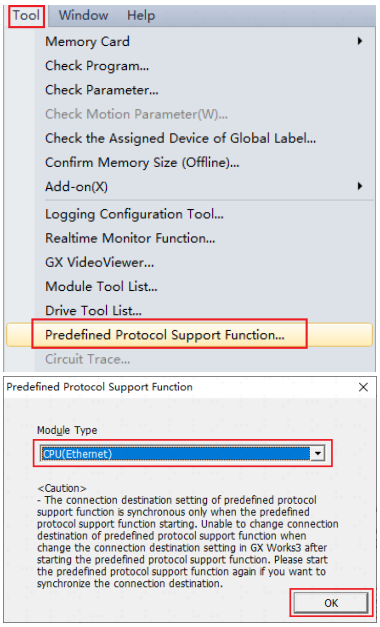


After completing the settings, click [Close with Reflecting the Setting] in the menu bar at the top.

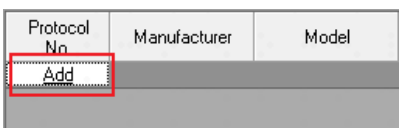


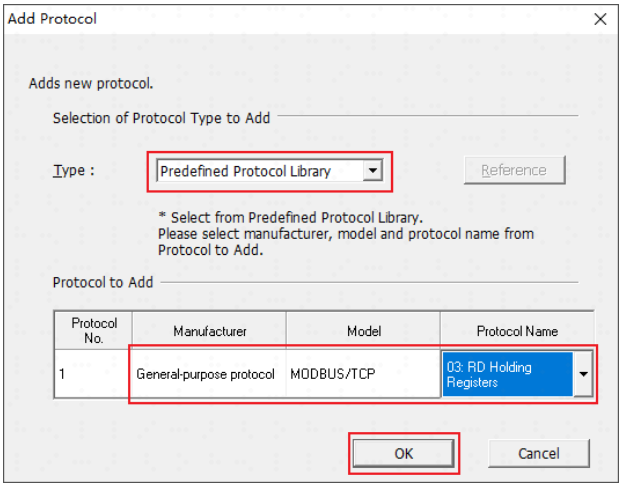
Return to the Ethernet port settings interface, click [Apply] to save the settings and close.

In the menu bar, click [Tool] - [Predefined Protocol Support Function], select [CPU (Ethernet)] in the pop-up dialog box, and click OK.



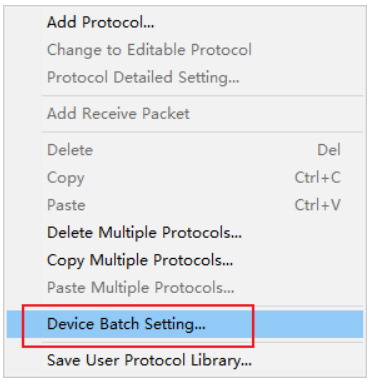
Build new one  , then click Add. Select [Predefined Protocol Library] as the protocol type and Modbus/TCP as the model. Set the protocol name according to the vision sensor register data specifications. You need to set function codes 03/16. First, set function code 03 to read the vision sensor holding register, then repeat the above steps and add function code 16 to write data to the vision sensor holding register.



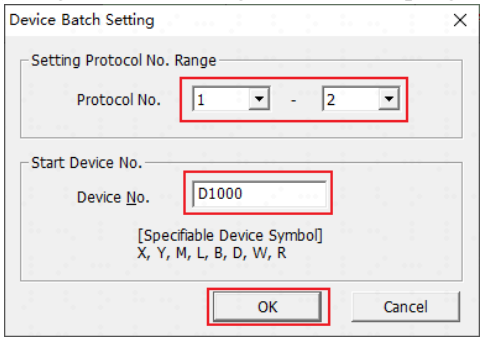


Protocol No.	Manufacturer	Model	Protocol Name	Communication Type	-> Send / <- Receive	Packet Name	Packet Setting
1	General-purpose p	MODBUS/TCP	03: RD Holding Registers	Send&Receive			
					->	Request	Variable Unset
					<-(1)	Normal response	Variable Unset
					<-(2)	Error response	Variable Unset
2	General-purpose p	MODBUS/TCP	16: WR Multi Registers	Send&Receive			
					->	Request	Variable Unset
					<-(1)	Normal response	Variable Unset
					<-(2)	Error response	Variable Unset
Add							

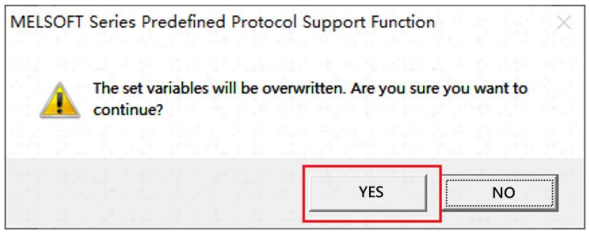
Right-click on the blank space and select [Device Batch setting].



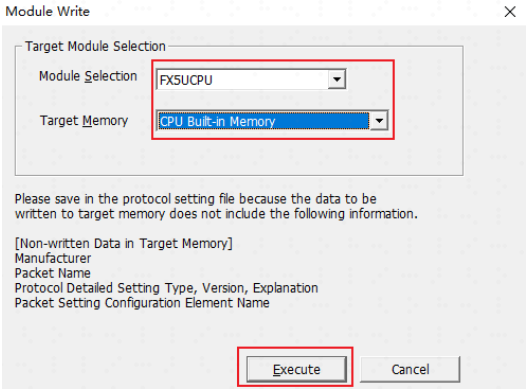
In the Device batch setting, specify the protocol number range as 1-2 and set the starting device number to D1000 (using the idle storage area in the program).



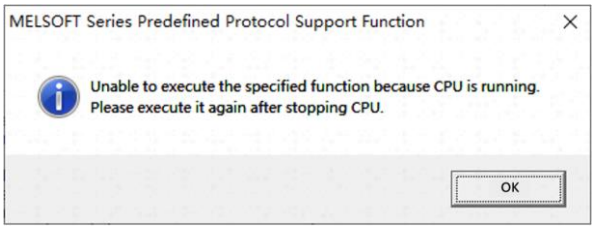
[The set variables will be overwritten. Are you sure you want to continue?] Select [Yes].



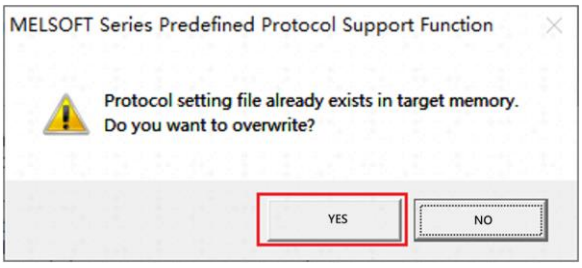
Click [Module Write]  , select [CPU Built-in Memory] for [Target Memory], and click [Execute].



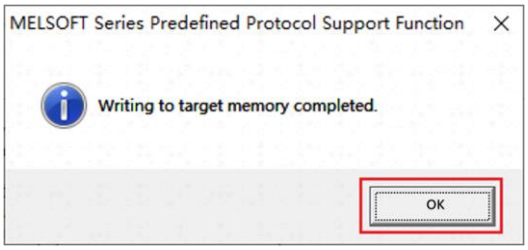
If the message [Unable to execute the specified function because CPU is running. Please execute it again after stopping CPU] pops up, you need to set the CPU to STOP mode and try the module write operation again.



Choose [Yes]

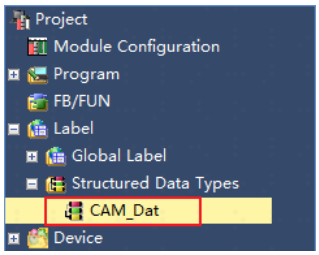


After the message [Write to target memory completed] is displayed, click OK. It is recommended to click Save  to save the configuration to your computer for future use or modification. Close the interface after saving.



4.1.2 Write PLC program

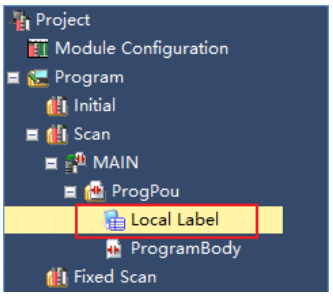
In the project tree, select [Label] - [Structure Data types], right-click and select [New Data], and create the following structure variables.



<Filter> Show Details(Y) Display Setting Check

	Label Name	Data Type		Chinese Simplified/简体中文(Display Target)
1	Trigger	Bit	...	
2	StatisticsReset	Bit	...	
3	SwitchProjectID	Word [Signed]	...	
4	RunningTotal	Double Word [Unsigned]/Bit String [32-bit]	...	
5	LineExist_StartPointX	FLOAT [Single Precision]	...	
6	LineExist_EndPointX	FLOAT [Single Precision]	...	
7	GrayscaleArea_Num	Double Word [Unsigned]/Bit String [32-bit]	...	
8	OCR_StringLength	Double Word [Unsigned]/Bit String [32-bit]	...	
9	OCR_StringContent	String(20)	...	
10			...	

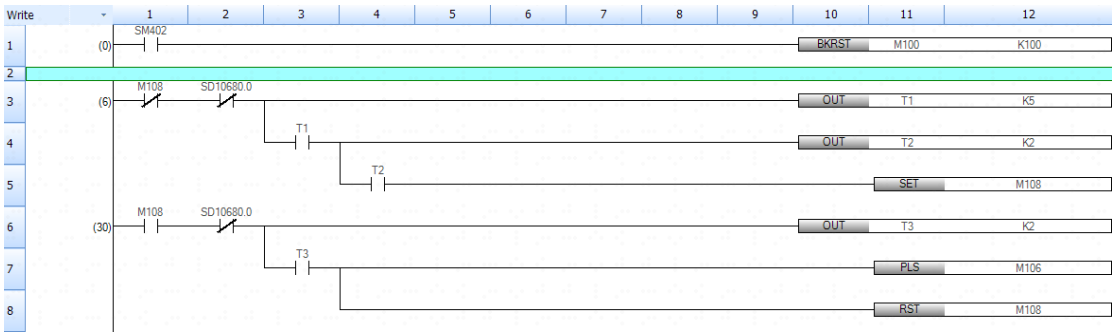
Select [Program] - [Scan] - [MAIN] - [ProgPou] in the project tree, and add the following variables in [Local Label].



<Filter> Show Details(Y) Display Settir

	Label Name	Data Type	
1	CAM_Dat_1	CAM_Dat	...
2	Timer_1	Timer	...
3			...

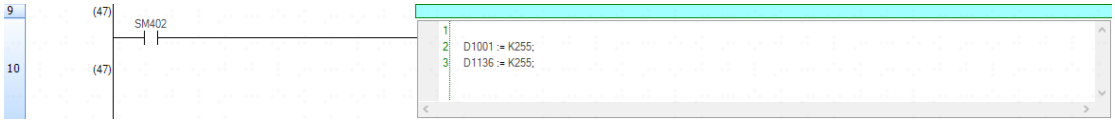
Add the following program to the ladder diagram interface.



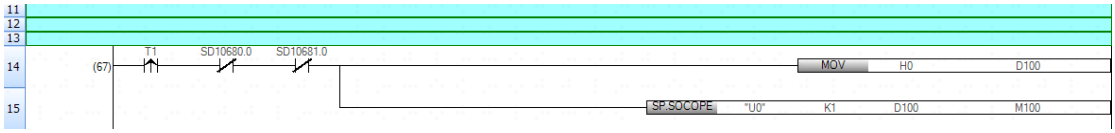
Program Line 1: During the first scan cycle after the PLC powers on, reset the 100 relays starting from M100.

Program lines 3-5: Use auxiliary relay M108 and the normally closed contact of bit 0 of system special register SD10680 to open timer T1. After a 500ms delay, execute the OPEN command to establish the connection and enable timer T2. If the connection channel has not been opened after 200ms, set M108.

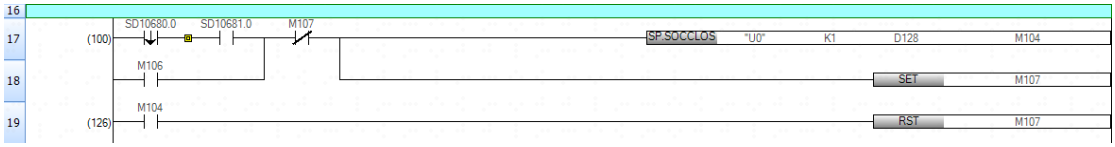
Program lines 6-8: When M108 is set and SD10680.0 is still not connected, timer T3 is activated. After a delay of 200ms, M106 is set for one scan cycle, the CLOSE command is executed to disconnect, and M108 is reset.



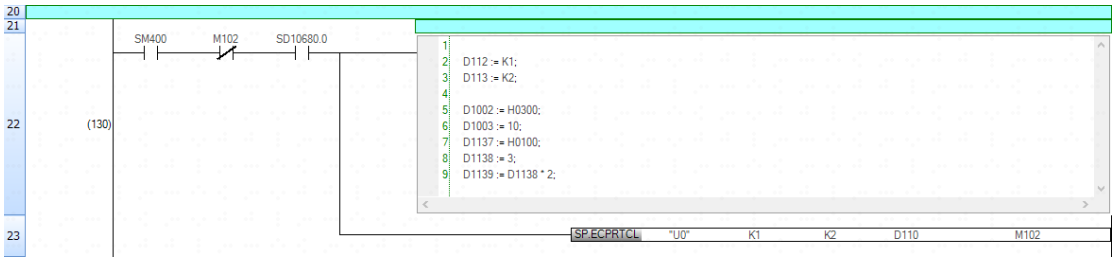
Line 10 of the program: Use the special relay SM402 to assign 255 to registers D1001/D1136 during the first scan cycle of the CPU. These two addresses are used to store the station numbers of the vision sensor slave stations.



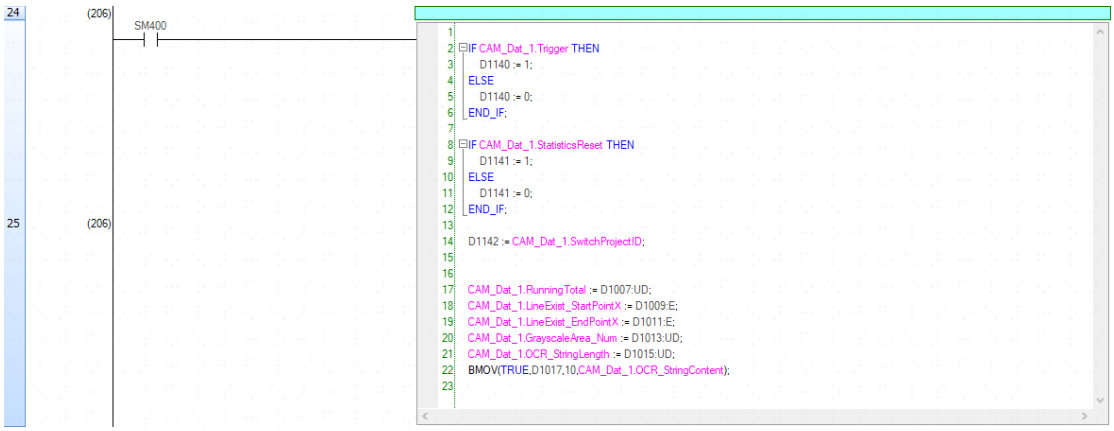
Lines 14-15 of the program: If TCP communication is currently not open and there is no open request, the rising edge of timer T1 will trigger the OPEN instruction to establish a connection with the vision sensor. K1 corresponds to the connection number of the device in the object device connection configuration, and D100 corresponds to some parameters of the OPEN instruction, occupying the next 10 word addresses after D100. If communication parameters are set in the object device connection configuration, assigning a value of 0 to D100 indicates that the pre-set parameters are being called. M100 is the completion bit for the OPEN command, and M101 is the error bit for the OPEN command. Only one scan cycle is triggered. If both M100 and M101 are triggered simultaneously, it indicates an execution error of the command.



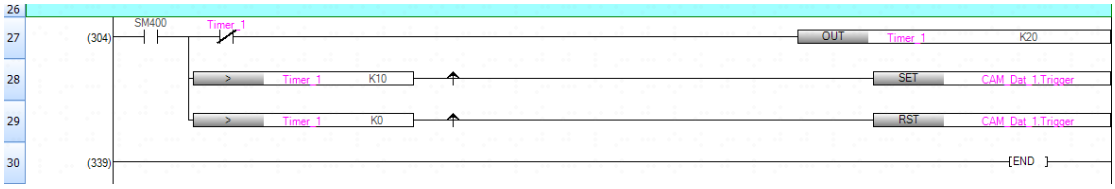
Lines 17-19 of the program: When the connection between devices is suddenly interrupted due to an unstable network cable connection or other reasons, SD10680.0 will change from TRUE to FALSE. The falling edge will trigger the execution of the CLOSE command to close the connection, while setting M107. The normally closed contact of M107 prevents the repeated execution of the CLOSE command. D128 is the starting address of the CLOSE command control data soft element, occupying 2 words. M104 is the completion flag for the CLOSE command, and M105 is the error flag for the CLOSE command, which is only triggered for one scan cycle. When the CLOSE command is completed, M104 will reset M107.



Lines 21-23 of the program: When communication is enabled, the protocol numbers that need to be executed in the communication protocol support function are assigned to the address starting at D112. D1002 stores the starting address 0x300 of the read vision sensor hold register. D1003 stores the number of words to read from the vision sensor's hold register (10 words), D1137 stores the starting address (0x100) for writing to the vision sensor's hold register, D1138 stores the number of bytes for writing to the vision sensor's hold register (3 bytes), and D1139 stores the byte length for writing to the vision sensor's hold register. The ECPRTCL instruction is executed to perform data transmission and reception with the vision sensor. K2 represents the protocol number for executing the communication protocol support function, and D110 stores the starting address of the device for the ECPRTCL instruction control data, with a range from D110 to D127. M102 is the ECPRTCL instruction completion bit, and M103 is the ECPRTCL instruction error bit. Only one scan cycle is triggered. When the ECPRTCL instruction is completed, M102 is set, and its normally closed contact disconnects the ECPRTCL instruction for one scan cycle. After M102 is reset in the next scan cycle, the ECPRTCL instruction continues to be executed.



Line 25 of the program: The [Trigger] and [Statistics Reset] configured in the vision sensor format input string occupy 1 word of space in ModbusTCP communication even though the data type is Bit. Therefore, it is necessary to convert Bit and Word in the program, and after conversion, map the receive and send registers to the newly created local variables.

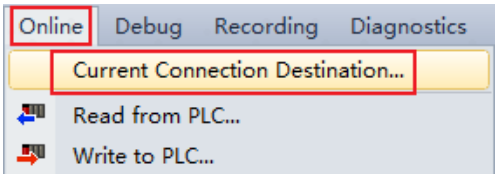


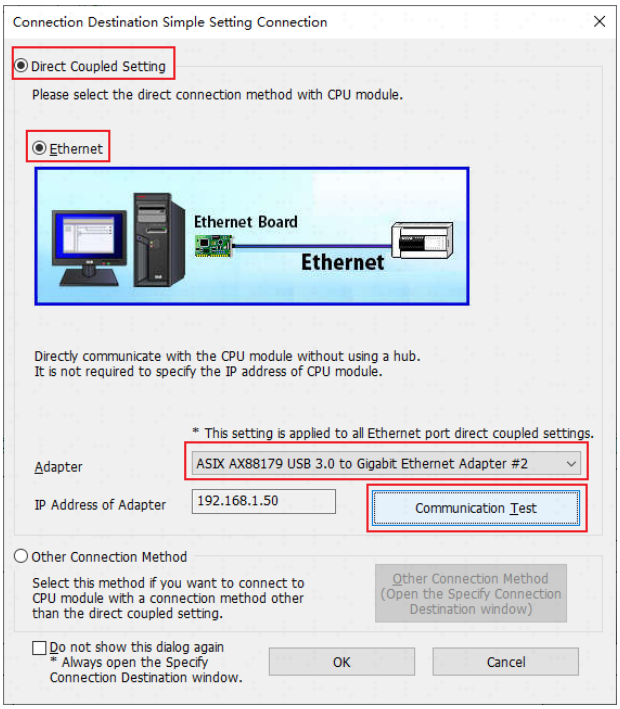
Lines 27-29 of the program: For testing purposes, a timer command is used to send trigger commands to the vision sensor at regular intervals. The timing interval can be modified as needed. In this example, the sensor is triggered to take a photo every 2 seconds. If this function is not required, the normally open contact of the SM400 relay in line 27 of the program can be switched to a normally closed contact to disable this function. After making this modification, the PLC program must be re-downloaded for the changes to take effect.

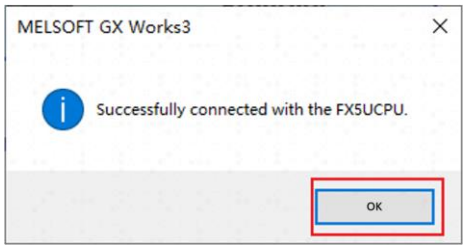
4.2 Communication Test

4.2.1 Download program

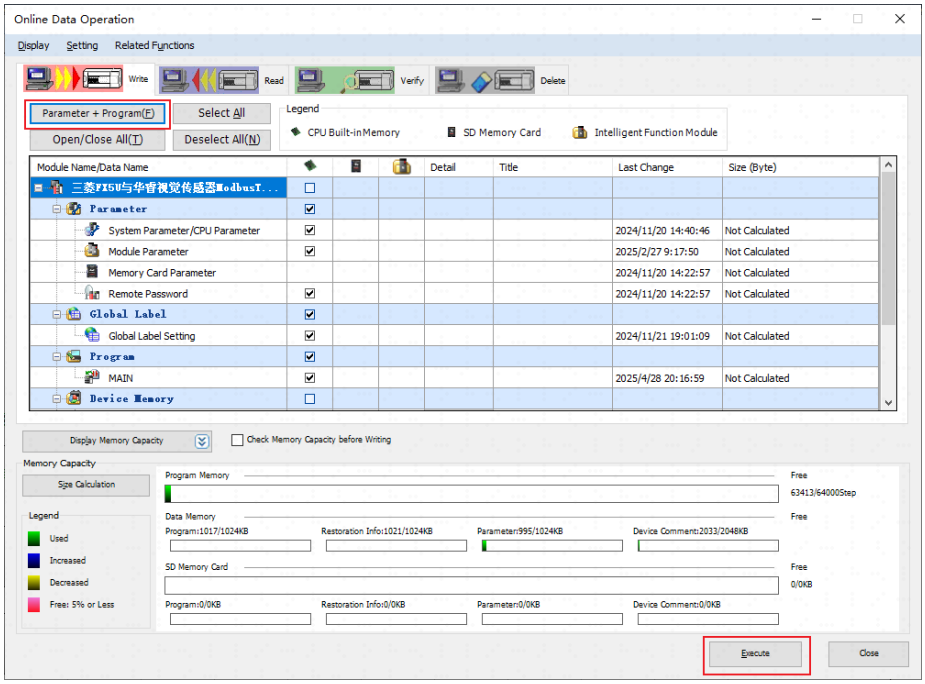
Click [Online] - [Current Connection Destination] in the menu bar at the top. In the pop-up dialog box, select [Direct Coupled Settings] - [Ethernet], select the Ethernet adapter on your computer, click [Communication Test], and when the pop-up message [Successfully connected to FX5UCPU] appears, click OK.



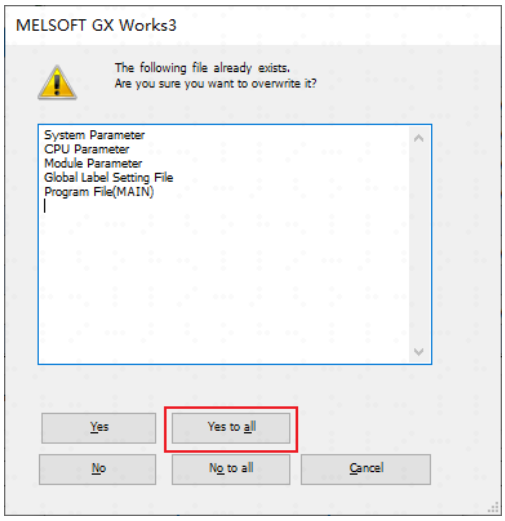




Click the button  above to download the program to the PLC, select [Parameters + Program], and click [Execute].



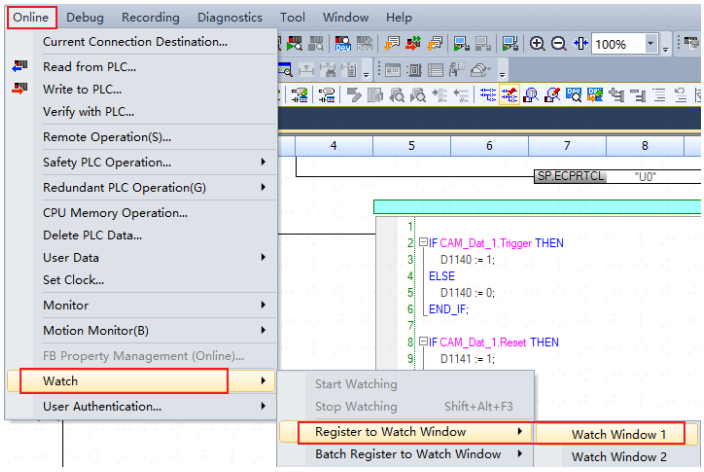
[The following files already exist. Do you want to overwrite them?] Select [Yes to all].



After the download is complete, you need to power off and reboot the PLC.

4.2.2 Communication Test

Click the Monitor Mode button  in the toolbar above to enter monitor mode. Click [Online] - [Watch] - [Register to watch Window] - [Watch Window 1] to enter the monitor window.



Add the following variables to the monitoring window and click  Start Monitoring.

Watch 1[Watching]

ON | OFF | ON/OFF toggle | Update | Start Watching | Stop Watching

Name	Current Value	Display Format	Data Type
ProgPou/CAM_Dat_1		--	CAM_Dat
Trigger	TRUE	BIN	Bit
StatisticsReset	FALSE	BIN	Bit
SwitchProjectID	0	Decimal	Word [Signed]
RunningTotal	8,142	Decimal	Double Word [Unsigned]/Bit String [32-bit]
LineExist_StartPointX	210.826996	--	FLOAT [Single Precision]
LineExist_EndPointX	410.821991	--	FLOAT [Single Precision]
GrayscaleArea_Num	15,957	Decimal	Double Word [Unsigned]/Bit String [32-bit]
OCR_StringLength	9	Decimal	Double Word [Unsigned]/Bit String [32-bit]
OCR_StringContent	UUMUMKUW2	--	String (20)

When manually triggering the vision sensor to take a photo, set the value of the [CAM_Dat_1.Trigger] variable to [TRUE]. The PLC will send a trigger command to the vision sensor, which will then take a photo and send the results back to the PLC. To reset the total count, set the value of the [CAM_Dat_1.StatisticsReset] variable to [TRUE]. When switching engineering IDs, it is necessary to configure the communication settings for the engineering ID to be switched to the same communication protocol in advance. Otherwise, after the switch, the PLC and the vision sensor will lose connection. After the switch is successful, the vision sensor interface must be manually refreshed to display the engineering ID after the switch.

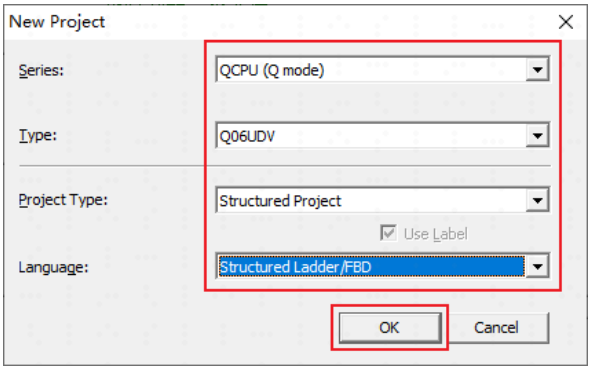
Note: The above PLC configuration and program are only applicable to the current case. Users should modify them according to project requirements when using them.

5 Mitsubishi Q06UDV PLC Configuration

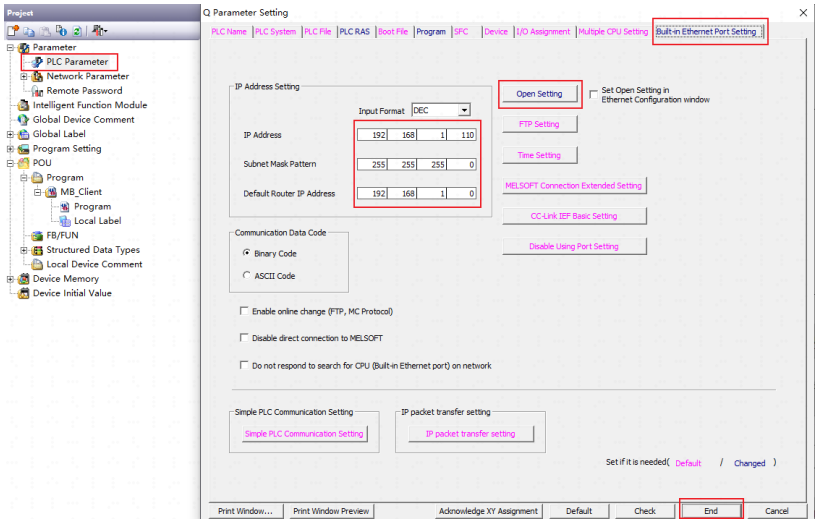
5.1 PLC Configuration

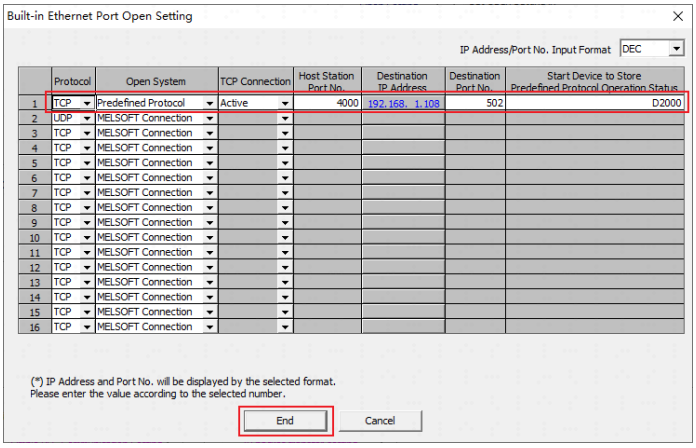
5.1.1 Create New Project

Open Mitsubishi Works2 software, click "New Project", select "QCPU Series", choose camera model "Q06UDV", Project Type select "Structured Project", for Program "Language" select "Structured Ladder/FBD", then click "OK".

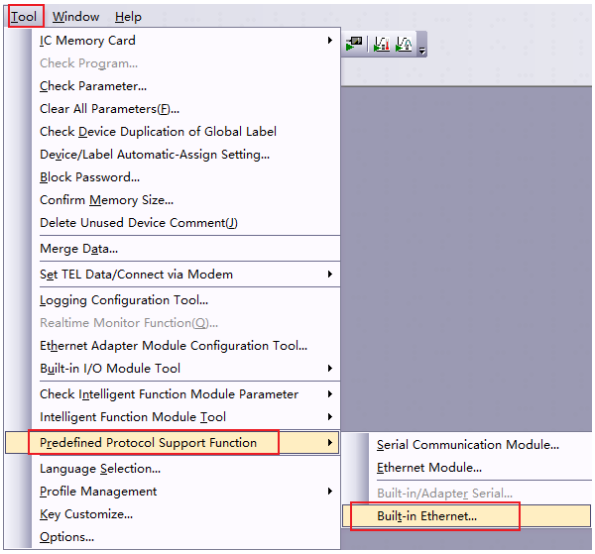


Navigate to left project tree and select [Parameters] → [PLC Parameters] → [Built-in Ethernet Port]. Modify the PLC's IP address, subnet mask, and other parameters. Click [Open Setting], then configure protocol 1 settings as [TCP] → [Communication Protocol] → [Active] → [4000] (The local station port No. settings range is 1025~4999 and 5010~65534. Other port numbers are reserved for system compression capacity and should not be used). Set the IP address and port No. of the vision sensor according to actual usage requirements. Configure [Start Device to Store Predefined Protocol Operation Status] as D2000 (Subsequent 19 words starting from the initial soft element will store the communication protocol running status. Do not reuse this address area elsewhere in the program). Finally, click [End].

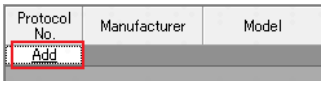


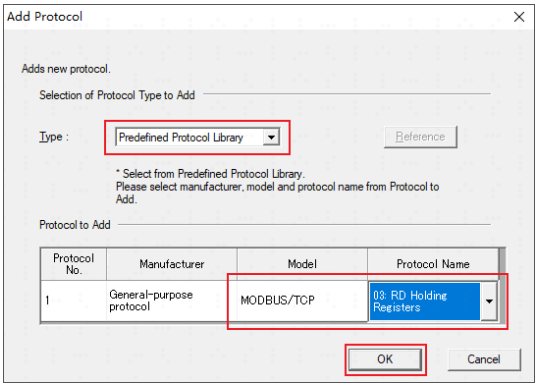


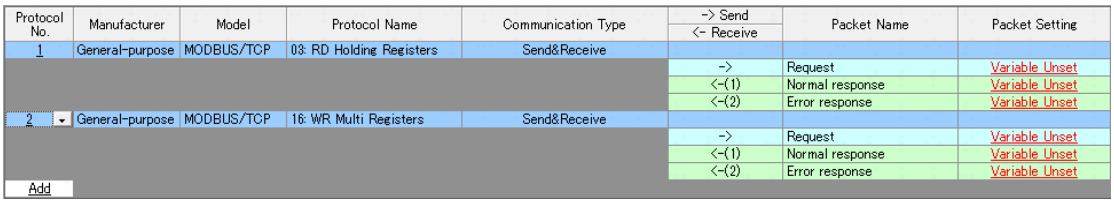
In the upper menu bar, select [Tools] → [Predefined Protocol support Function] → [Built-in Ethernet] to enter the parameters interface.



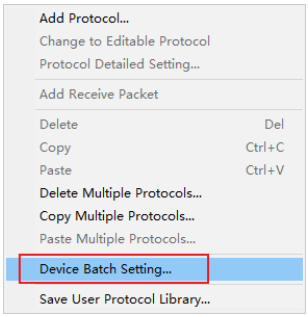
Click "New"  , then click "[Add]". Select [Predefined Protocol Library] for Protocol Type, and choose [MODBUS/TCP] for Model. According to the vision sensor register data specifications, it is required to configure function codes 03/16. First, set function code 03 to read the vision sensor's holding register. After completing the above step, repeat the process and add function code 16 to write data to the vision sensor's holding register.



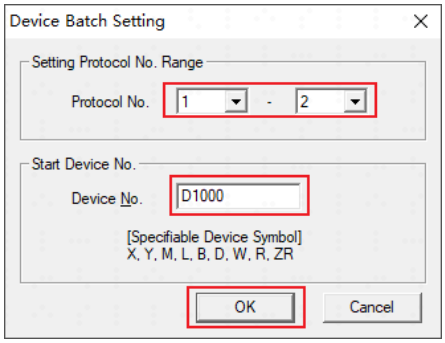




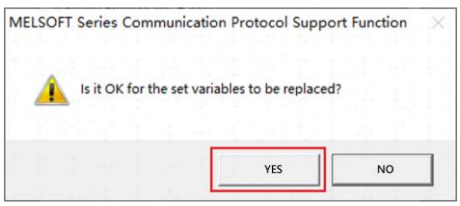
Right-click on the blank space and select [Device Batch Setting].



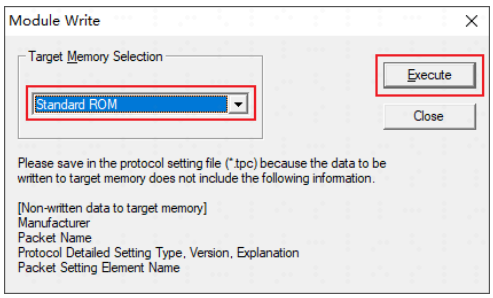
In the Device Batch Setting, designate the setting Protocol No range as 1-2, and set the starting Device No to D1000 (the Storage District in the Usage program).



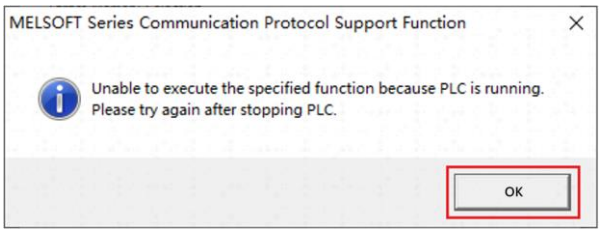
[is it OK for the set variables to be replaced?] Select [Yes].



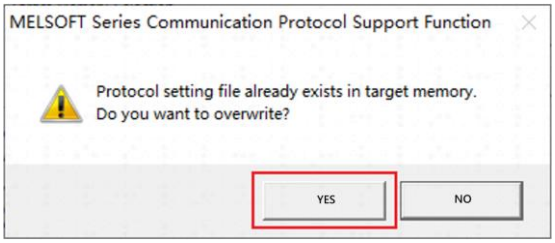
Click [Module Write]  , select [Standard ROM] in the [Target Memory Selection], then click [Execute].



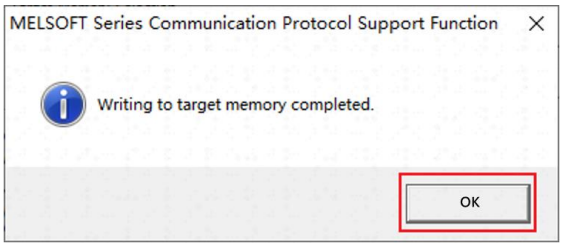
If the following message appears: [Unable to execute the specified function because PLC is running. Please try again after stopping PLC.], then the CPU must be set to STOP mode before performing the Line Module Write Operation again.



Click [Yes].

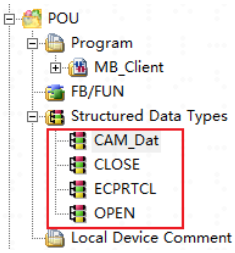


Display [Writing to target memory Completed.] Click OK. Recommended: Click Save  to save the configuration to the computer for future usage or modification. After saving, close the interface.



5.1.2 Write the PLC Program

In the Project Tree Select [POU] → [Structure Data Types], right-click and select [New Data] to create the structure variable as following.



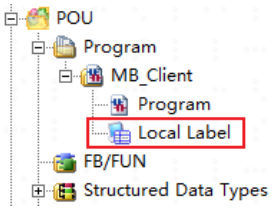
	Label Name	Data Type		Constant	Comment
1	Trigger	Bit	...		
2	StatisticsReset	Bit	...		
3	SwitchProjectID	Word[Signed]	...		
4	RunningTotal	Double Word[Unsigned]/Bit String[32-bit]	...		
5	LineExist_StartPointX	FLOAT (Single Precision)	...		
6	LineExist_EndPointX	FLOAT (Single Precision)	...		
7	GrayscaleArea_Num	Double Word[Unsigned]/Bit String[32-bit]	...		
8	OCR_StringLength	Double Word[Unsigned]/Bit String[32-bit]	...		
9	OCR_StringContent	String(20)	...		
10			...		

	Label Name	Data Type		Constant	Comment
1	S_S1	Word[Signed]	...		
2	S_S2	Word[Signed](0..1)	...		
3	S_D	Bit(0..1)	...		
4	Done	Bit	...		
5	Error	Bit	...		

	Label Name	Data Type		Constant	Comment
1	S_Un	Word[Signed]	...		
2	S_N1	Word[Signed]	...		
3	S_N2	Word[Signed]	...		
4	S_S	Word[Signed](0..17)	...		
5	S_D	Bit(0..1)	...		
6	Done	Bit	...		
7	Error	Bit	...		

	Label Name	Data Type		Constant	Comment
1	S_S1	Word[Signed]	...		
2	S_S2	Word[Signed](0..9)	...		
3	S_D	Bit(0..1)	...		
4	Done	Bit	...		
5	Error	Bit	...		

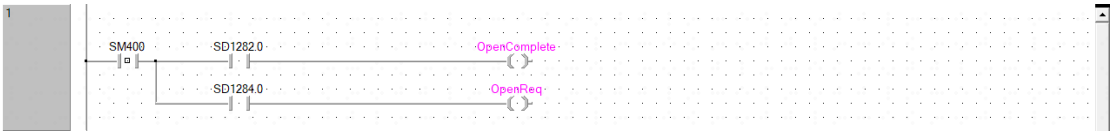
In the project tree select [POU] → [Program] → [Local Label], then add the following variables in Local Tag.



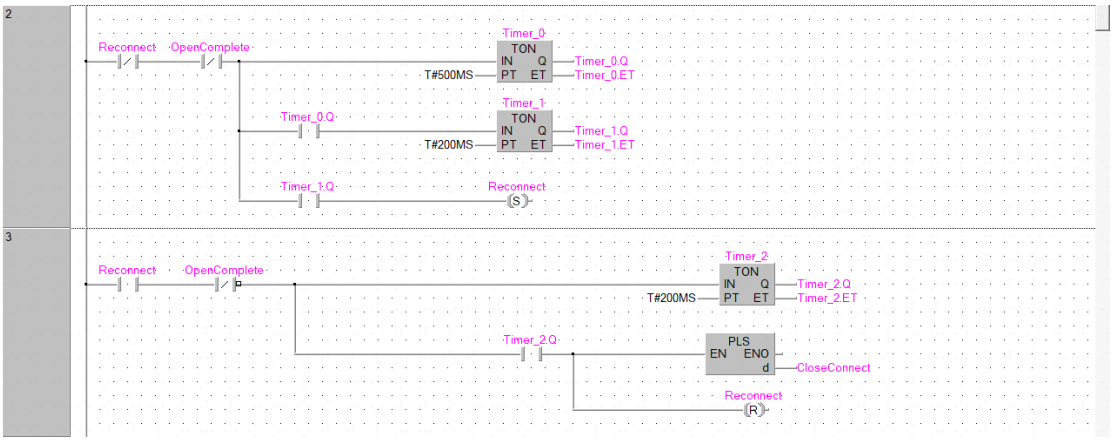
Local Label Setting MB_C...

	Class	Label Name	Data Type		Constant	Device	Address	Comment
1	VAR	Timer_0	TON	...				
2	VAR	Timer_1	TON	...				
3	VAR	Timer_2	TON	...				
4	VAR	OpenComplete	Bit	...				
5	VAR	OpenReq	Bit	...				
6	VAR	Reconnect	Bit	...				
7	VAR	CloseConnect	Bit	...				
8	VAR	CloseStatus	Bit	...				
9	VAR	OPEN	OPEN	...		Detail Setting	Detail Setting	
10	VAR	CLOSE	CLOSE	...		Detail Setting	Detail Setting	
11	VAR	ECPRTCL	ECPRTCL	...		Detail Setting	Detail Setting	
12	VAR	CAM_Dat_1	CAM_Dat	...		Detail Setting	Detail Setting	
13	VAR	Timer_3	TON	...				
14				...				

Double-click on the [Main Program] and Add the following program.

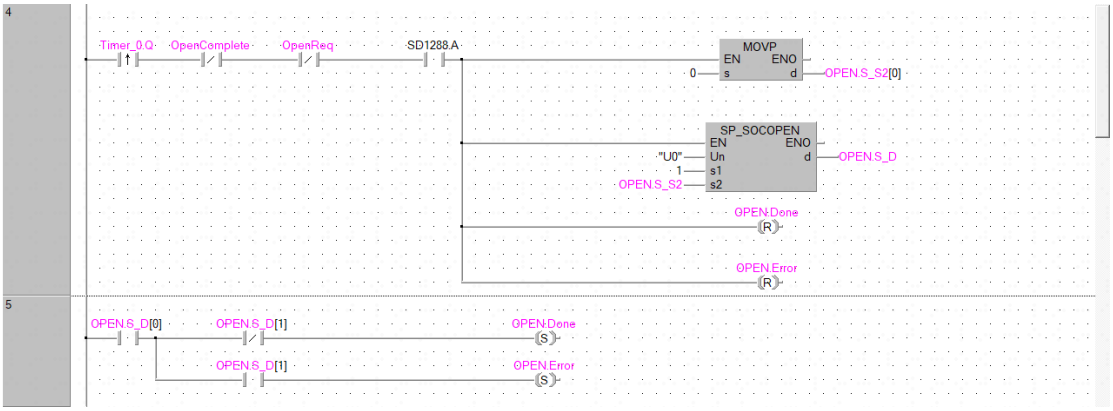


Program No. 1 Line: Map the status of the system's special registers to the newly created local labels in the previous step for convenient reference in the program.

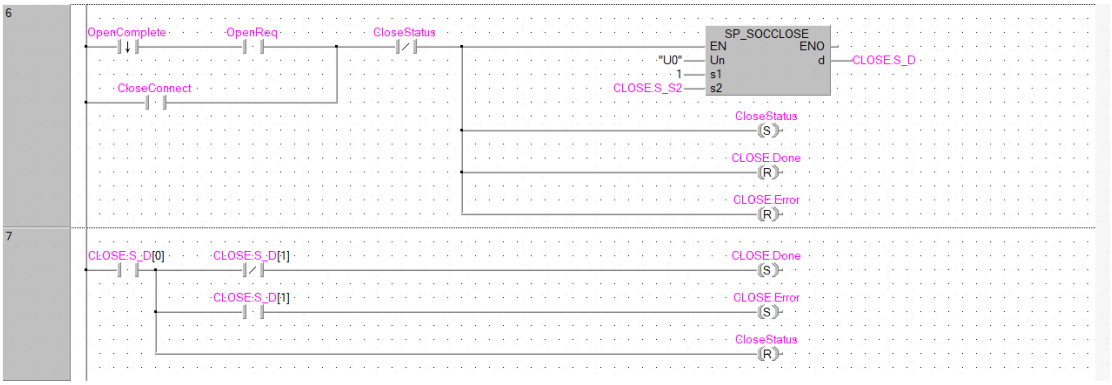


Line 2 of the program: Use Reconnect and the normally-closed contracts of the OpenComplete to open the timer_0. After a delay of 500ms, the OPEN command is executed to establish the connection, and Timer_1 is started. If the connection channel still fails to open after 200ms, the Reconnect relay is set.

Line 3 of the program: When the Reconnect relay is set and communication remains unestablished, Timer_2 is activated. After a delay of 200ms, the CloseConnect relay is set for one scan cycle, the CLOSE command is executed to terminate the connection, and the Reconnect relay is reset.

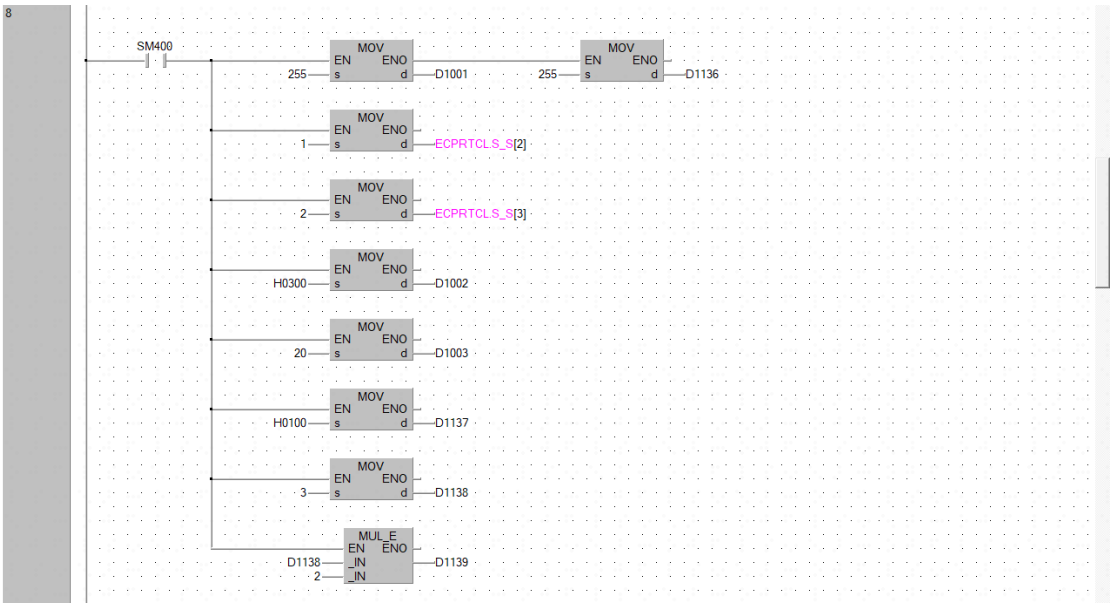


Lines 4-5 of the program: If the TCP communication is currently not OPEN and there is no OPEN request, and the network cable connection is good, the rising edge of the timer Timer_0 will trigger the OPEN command to establish a connection with the camera. The s1 pin in the open command corresponds to the protocol number in the built-in Ethernet port open Settings, and the s2 pin corresponds to some parameters of the open command. Occupying 10 character addresses, if communication parameters are set in the built-in Ethernet port opening Settings, assigning a value of 0 to the first address in s2 indicates the invocation of the set parameters. Open.s_d [0] is the completion bit of the OPEN instruction, and open.s_d [1] is the error bit of the OPEN instruction, which will only trigger one scanning cycle. When both OPEN.S_D[0] and OPEN.S_D[1] are triggered simultaneously, it indicates an instruction execution error.

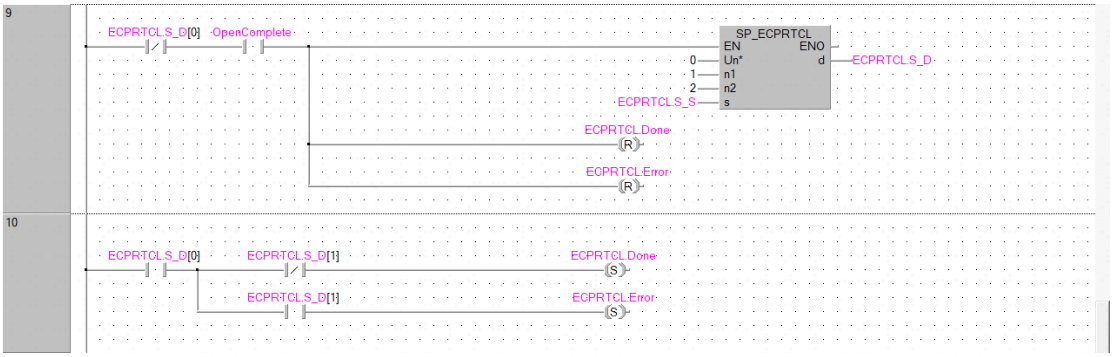


Lines 6-7 of the program: When the connection of a device is suddenly interrupted due to an unstable network cable connection or other reasons, the OpenComplete status will change from TRUE to FALSE. The falling edge will trigger the CLOSE command to perform the connection closure operation, and at the same time, set CloseStatus, the normally-closed contact of CloseStatus prevents the CLOSE instruction from being executed again. The s1 pin in the CLOSE instruction corresponds to the protocol number in the built-in Ethernet port opening Settings, and the s2 pin corresponds to the control data of the CLOSE instruction, occupying a two-word address. The close.s_d [0] is the completion bit of the CLOSE instruction, and the close.s_d [1] is the error bit of the CLOSE instruction, which will only trigger one scan cycle. When both "close.s_d [0]" and "close.s_d [1]" are triggered simultaneously, it indicates an instruction execution error. When the "CLOSE" instruction

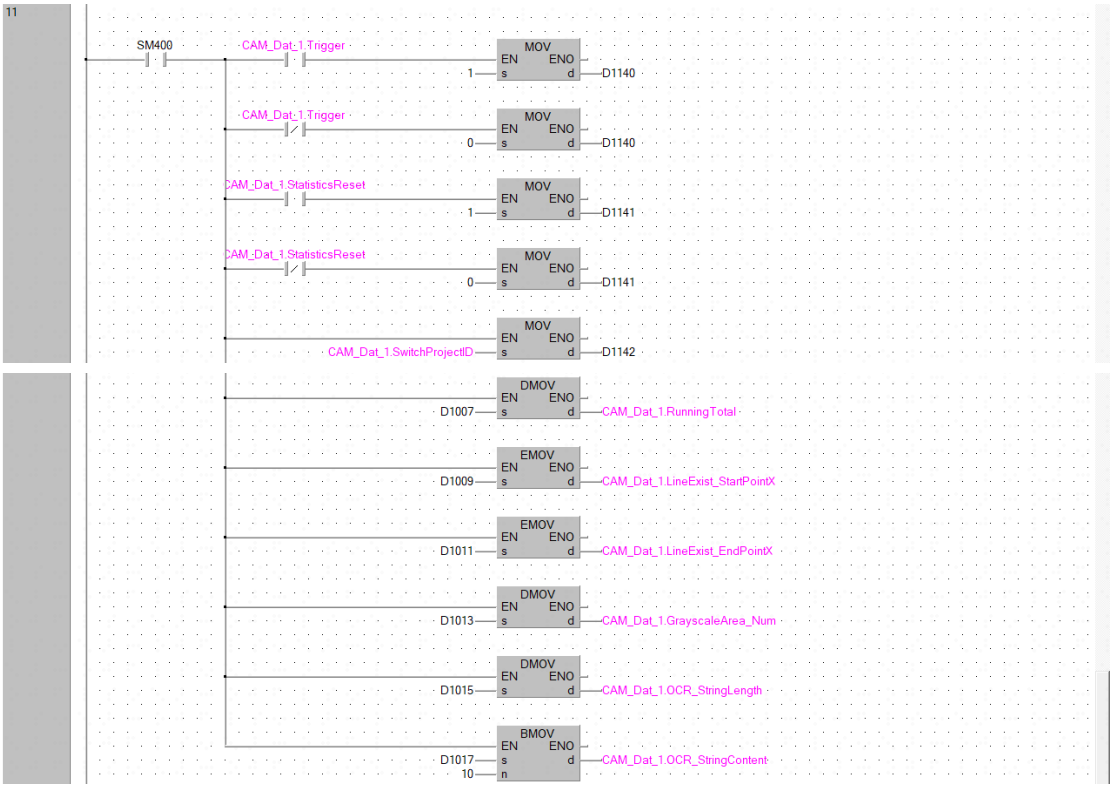
is completed, "close.s_d [0]" triggers the reset of the "CLOSE" instruction execution status.



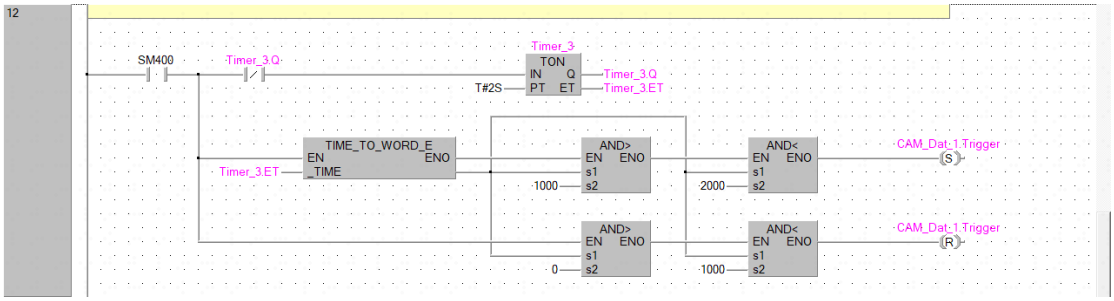
Line 8 of the program: Assign 255 to the registers D1001/D1136. These two addresses are used to store the Modbus slave station number. Assign the protocol number that needs to be executed in the communication protocol support function to ECPRTCL.s_s [2] and ECPRTCL.s_s [3]. D1002 stores the starting address of the read vision sensor hold register at 0x300, D1003 stores the number of read vision sensor hold registers at 20 words, D1137 stores the address of the write vision sensor hold register at 0x100, and D1138 stores the number of write vision sensor hold registers at 3 words. The byte length for storing the write vision sensor hold register in D1139 is 6 bytes.



Lines 9-10 of the program: When communication is on, the ECPRTCL command is executed to send and receive data from the vision sensor. The n1 pin in the ECPRTCL command corresponds to the protocol number in the built-in Ethernet port opening Settings, and the n2 pin represents the number of protocols that support the communication protocol function. The s pin represents the software component that stores the control data of the ECPRTCL instruction. Ecprclt.s_d [0] is the completion bit of the OPEN instruction, and ECprclt.s_d [1] is the error bit of the OPEN instruction, which will only trigger one scan cycle. When both ECPRTCL.S_D[0] and ECPRTCL.S_D[1] are triggered simultaneously, it indicates an instruction execution error.

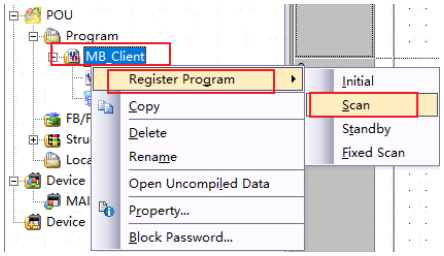


Line 11 of the program: Although the data type of [trigger] and [statistical reset] configured in the formatted input string of the vision sensor is Bit, they still occupy 1Word of space in ModbusTCP communication. Therefore, it is necessary to convert Bit and Word in the program, and after conversion, map the received and sent storage areas to the newly created local variables.



Line 12 of the program: For the convenience of testing, trigger commands are sent to the vision sensor at regular intervals through timer instructions. The timing can be modified as needed. In this case, the sensor takes a photo every 2 seconds. If this function is not required, the normally open contact of the SM400 relay on line 12 of the program can be switched to a normally closed contact to stop this function. The PLC program needs to be re-downloaded after modification for it to take effect.

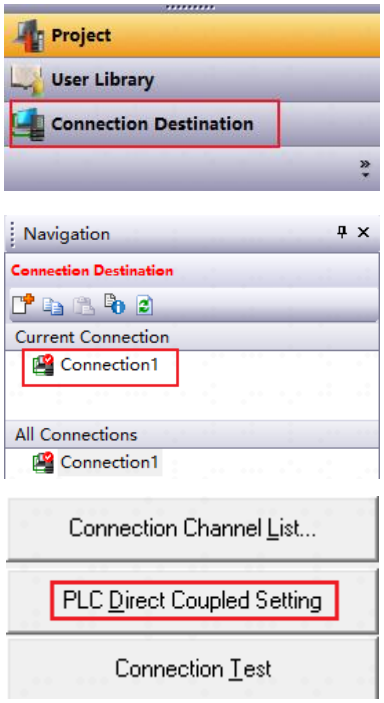
When the program is finished, right-click to edit the program, select [Register Program] -> [Scan], and add the program to the "Scan" program.

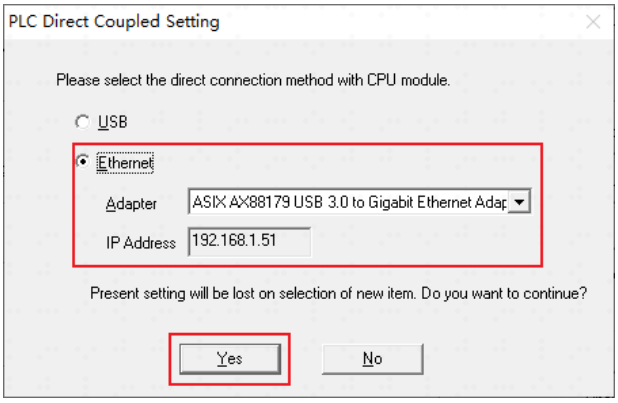


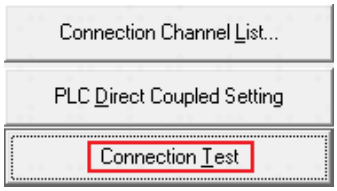
5.2 Communication Test

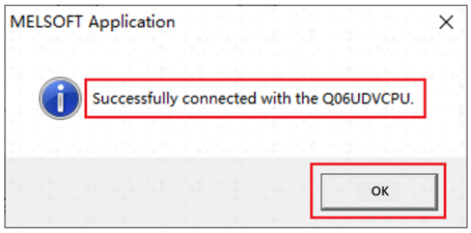
5.2.1 Download Program

Click the [Connection Destination] in the Left Project Tree. In the Current Status Connect Target Medium, Double-click "Connection1" in the current connection target to enter the connection target Settings. Click on the right side [PLC Direct Coupled Settings], then select and check the appropriate connection method and adapter based on the actual connection method with the PLC. Click [Yes], and then click [Connection Test]. Upon successful communication, the message "Successfully connected with the Q06UDV CPU" will be displayed. Click OK to exit the interface.

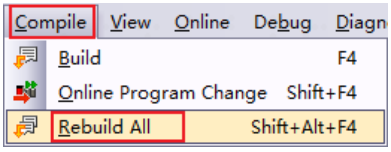




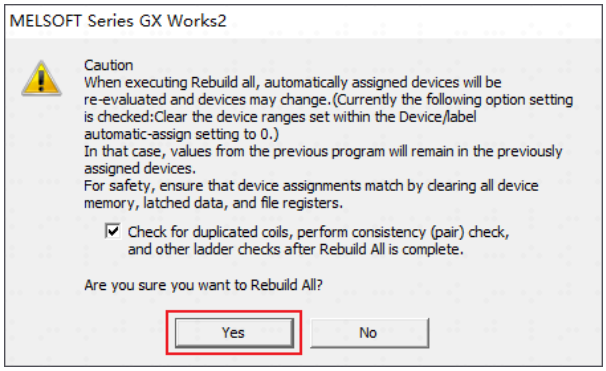




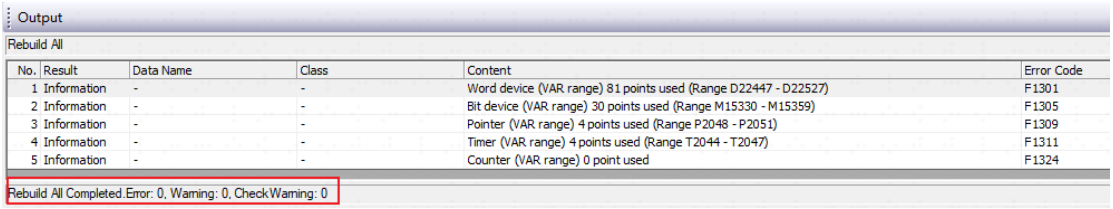
Click the Save button at the top to save the program, then select [Compile] → [Rebuild All].

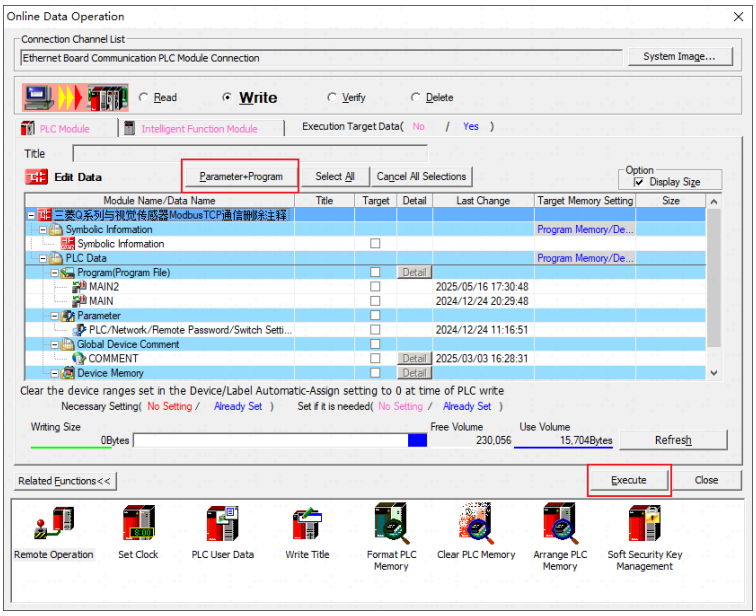


Select [Yes].

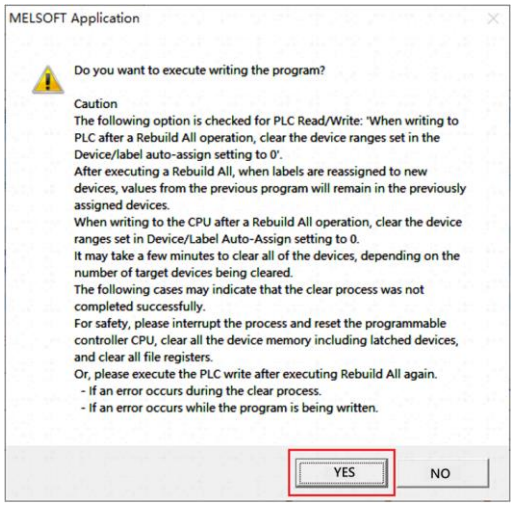


Confirm the Output Result without Error,  Click PLC to Write, And then Select [Parameters + Program], Click [Execute].

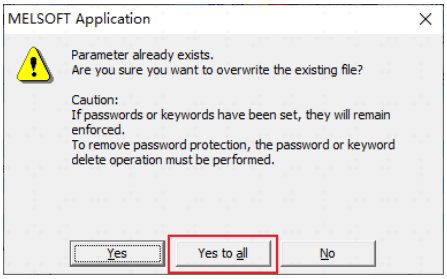




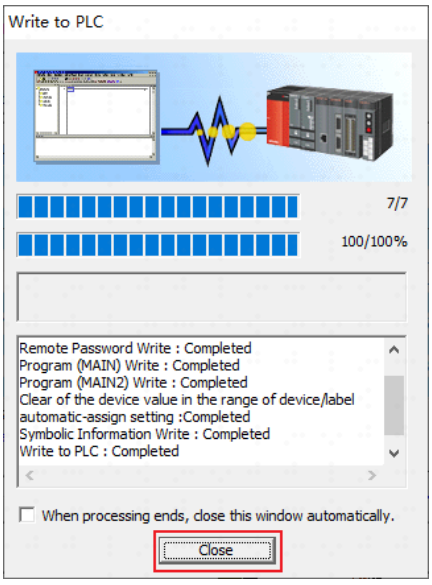
Select [Yes].



Select[Yes to All].



Click [close] when finished.

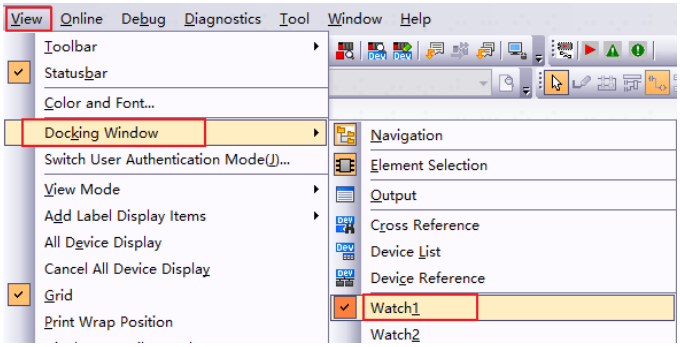


After completion, reset the PLC power.

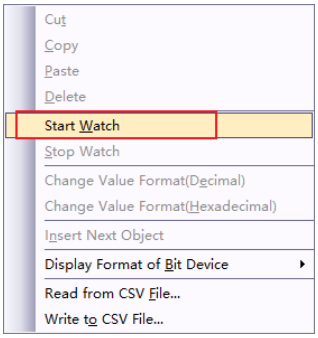
5.2.2 Communication Test

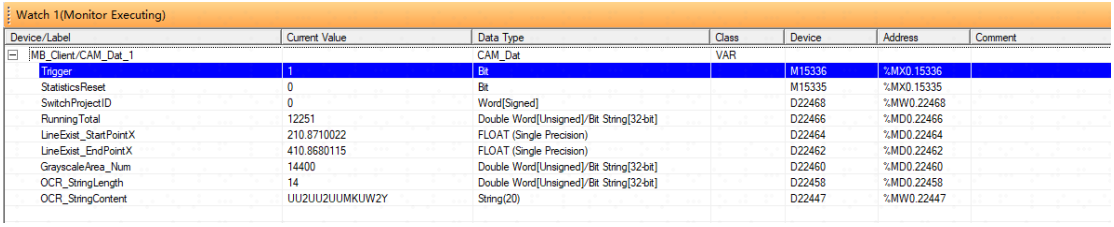
In the upper toolbar, click Spot to Start Monitoring, enter Monitoring Mode, and

select [View] -> [Docking Window] -> [watch 1].



In the monitoring window Add, select Down Variable, then right-click on the Null white space and choose [Start watch].





Device/Label	Current Value	Data Type	Class	Device	Address	Comment
MB_Client/CAM_Dat_1		CAM_Dat	VAR			
Trigger	1	Bit		M15336	%MX0.15336	
StatisticsReset	0	Bit		M15335	%MX0.15335	
SwitchProjectID	0	Word[Signed]		D22468	%MW0.22468	
RunningTotal	12251	Double Word[Unsigned]/Bit String[32-bit]		D22466	%MD0.22466	
LineExist_StartPointX	210.8710022	FLOAT (Single Precision)		D22464	%MD0.22464	
LineExist_EndPointX	410.8680115	FLOAT (Single Precision)		D22462	%MD0.22462	
GrayscaleArea_Num	14400	Double Word[Unsigned]/Bit String[32-bit]		D22460	%MD0.22460	
OCR_StringLength	14	Double Word[Unsigned]/Bit String[32-bit]		D22458	%MD0.22458	
OCR_StringContent	UU2UU2UUMKUW2Y	String(20)		D22447	%MW0.22447	

When the vision sensor is manually triggered to take a photo, modify the value of the [CAM_Dat_1.Trigger] variable to [TRUE]. The PLC will send a trigger command to the vision sensor, which will take a photo and feed back the result data to the PLC. When the total number of runs needs to be reset, Change the value of the [CAM_Dat_1.Reset] variable to [TRUE]. When switching the project ID, it is necessary to configure the communication of the project ID to be switched to the same communication protocol in advance; otherwise, the PLC and the vision sensor will be disconnected after the switch. After the switch is successful, the vision sensor interface needs to be manually refreshed to display the switched project ID.

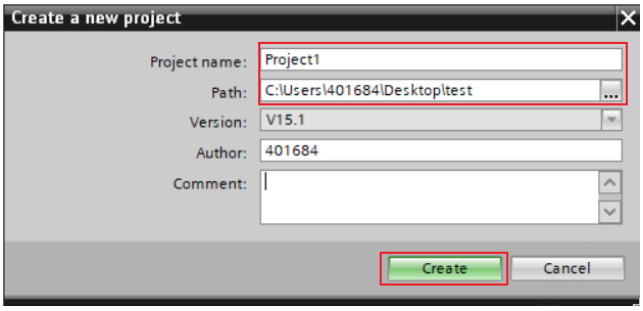
Notes: The configurations and programs in the above PLC are only applicable to the current case. Users are requested to modify them according to the project requirements when using them.

6 Siemens 1200 PLC Configuration

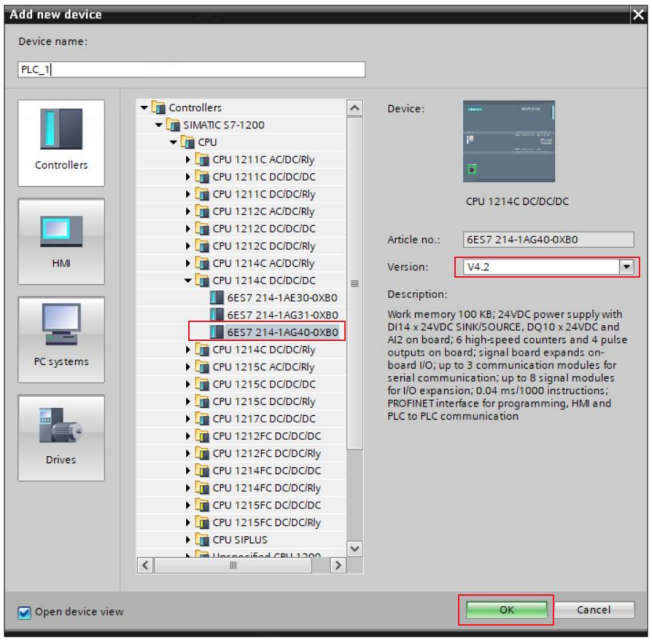
6.1 PLC Configuration

6.1.1 Create New Project

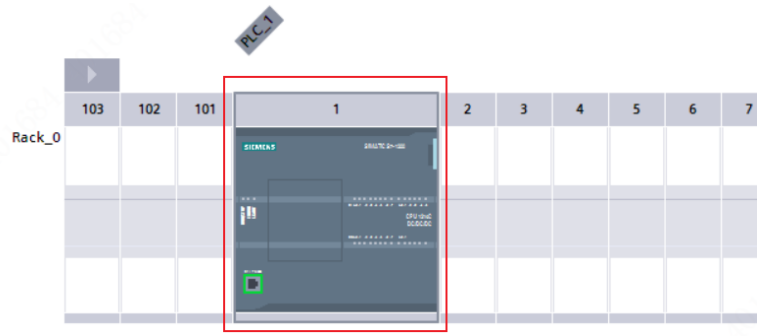
Open TIA Portal V15.1, click the New Button, modify the Project Name and Storage Path, then click [Create]. 



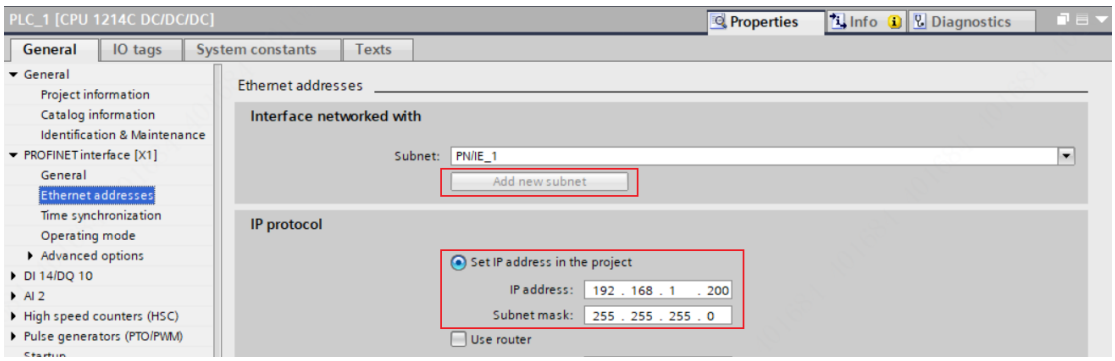
In the project tree on the left, double-click [Add New Device], select the required PLC model and version according to the actual needs. In this case, 1214C DC/DC/DC, version V4.2 is selected, and click [OK].

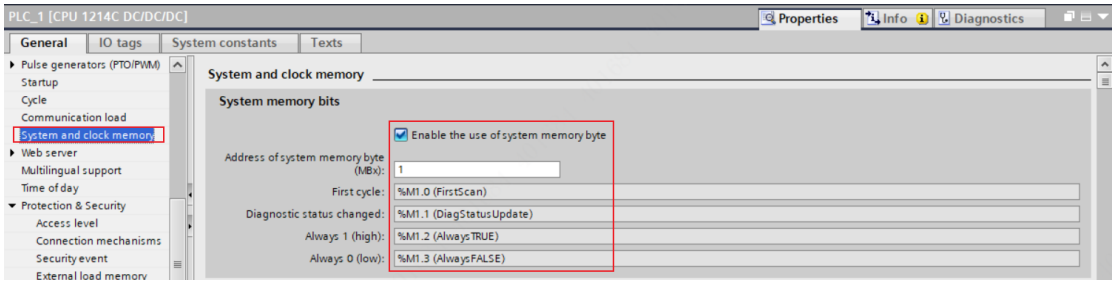


Double-click the CPU Module in the Device View to enter the CPU Attribute Setting Interface.



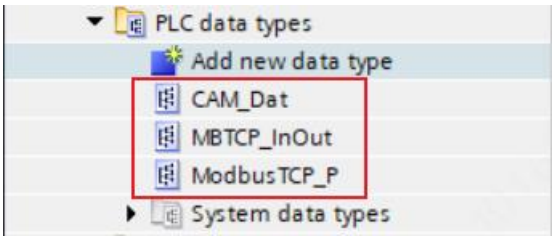
In the [General] , select [PROFINET Interface] → [Ethernet Address], click [Add new Subnet], click [set IP address in the project], and fill in the IP Address and Subnet Mask correctly. To facilitate PLC programming, check [Enable the use of system memory byte].





6.1.2 Write the PLC Program

In the [PLC data types] section of the project tree, add the structure data type shown in the following figure, perform IO mapping corresponding to the formatted input and output strings in the vision sensor, and write the ModbusTCP communication program.



CAM_Dat

	Name	Data type	Default value	Accessible f...	Writa...	Visible in ...	Setpoint	Comment
1	Tigger	Bool	false	☑	☑	☑	☐	
2	StatisticsReset	Bool	false	☑	☑	☑	☐	
3	SwitchProjectId	Byte	16#0	☑	☑	☑	☐	
4	RunningTotal	UDInt	0	☑	☑	☑	☐	
5	LineExist_StartPointX	Real	0.0	☑	☑	☑	☐	
6	LineExist_EndPointX	Real	0.0	☑	☑	☑	☐	
7	GrayscaleArea_Num	UDInt	0	☑	☑	☑	☐	
8	OCR_StringLength	UDInt	0	☑	☑	☑	☐	
9	OCR_StringContent	String	''	☑	☑	☑	☐	

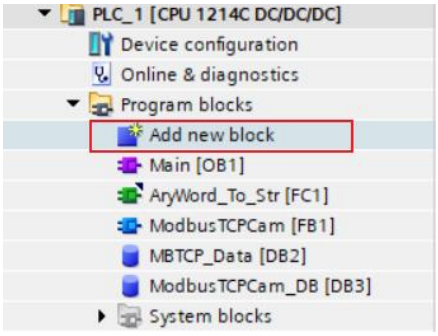
MBTCP_InOut

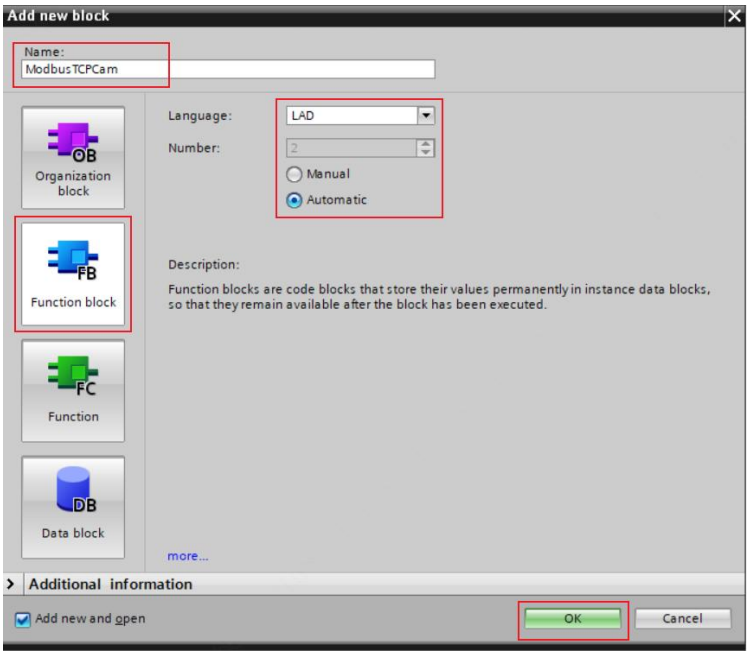
	Name	Data type	Default value	Accessible f...	Writa...	Visible in ...	Setpoint	Comment
1	Input	Array[0..124] of Word		☑	☑	☑	☐	
2	Output	Array[0..122] of Word		☑	☑	☑	☐	
3	MBTCP_Status	Word	16#0	☑	☑	☑	☐	
4	Connected	Bool	false	☑	☑	☑	☐	

ModbusTCP_P

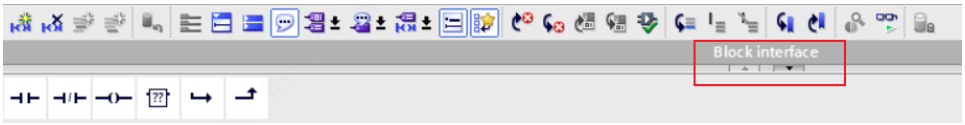
	Name	Data type	Default value	Accessible f...	Writa...	Visible in ...	Setpoint	Comment
1	DISCONNECT	Bool	FALSE	☑	☑	☑	☐	
2	Done	Bool	false	☑	☑	☑	☐	
3	Busy	Bool	false	☑	☑	☑	☐	
4	Error	Bool	false	☑	☑	☑	☐	
5	REQ	Bool	false	☑	☑	☑	☐	
6	Status	Word	16#0	☑	☑	☑	☐	

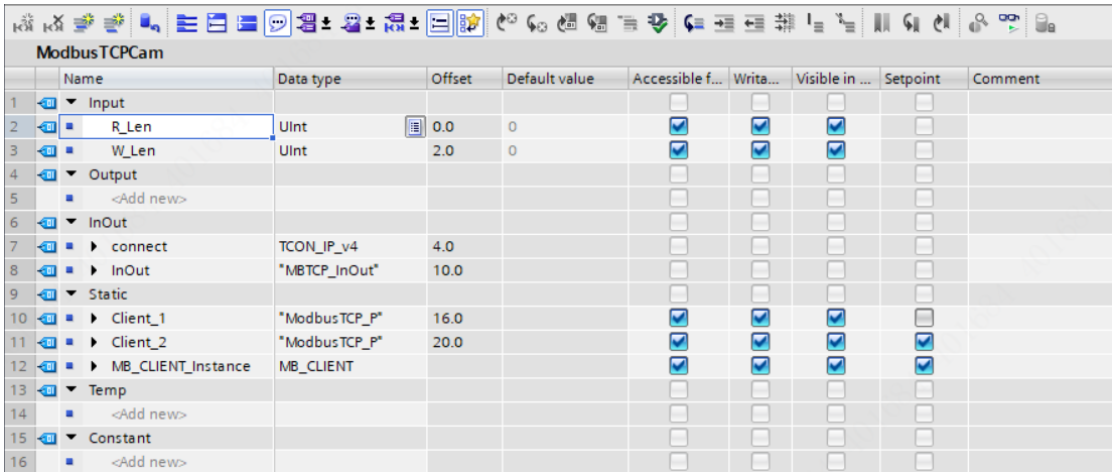
Double-click on the Project Tree 's [Add new Block], select [Function Block], choose [LAD] as the Language, Click OK after changing the name.





Open the newly created function block, click on [Block interface] at the top, and add the following internal variables in the interface.





	Name	Data type	Offset	Default value	Accessible f...	Write...	Visible in ...	Setpoint	Comment
1	Input				☐	☐	☐	☐	
2	R_Len	UInt	0.0	0	☑	☑	☑	☐	
3	W_Len	UInt	2.0	0	☑	☑	☑	☐	
4	Output				☐	☐	☐	☐	
5	<Add new>				☐	☐	☐	☐	
6	InOut				☐	☐	☐	☐	
7	connect	TCON_IP_v4	4.0		☐	☐	☐	☐	
8	InOut	"MBTCP_InOut"	10.0		☐	☐	☐	☐	
9	Static				☐	☐	☐	☐	
10	Client_1	"ModbusTCP_P"	16.0		☑	☑	☑	☐	
11	Client_2	"ModbusTCP_P"	20.0		☑	☑	☑	☑	
12	MB_CLIENT_Instance	MB_CLIENT			☑	☑	☑	☑	
13	Temp				☐	☐	☐	☐	
14	<Add new>				☐	☐	☐	☐	
15	Constant				☐	☐	☐	☐	
16	<Add new>				☐	☐	☐	☐	

In the [connect] variable within the InOut variable, the relevant parameters for

ModbusTCP communication are stored, and their specific meanings are as follows:

InterfaceId: The network port hardware identifier of the client PLC. View the corresponding hardware identifier in [Device Configuration]->[CPU properties]->[System constants].

ID: Connection ID, with a value range of 0 to 4095. This parameter will uniquely determine the connections in the CPU, and each connection must use a unique ID.

ConnectionType: Connection type. The default value is 11 and does not need to be modified. If modified, it will cause communication errors.

ActiveEstablised: 1 indicates an active connection, and 0 indicates a passive connection.

RemoteAddress: The IP address of the connection partner;

RemotePort: The port number of the connection partner;

LocalPort: Local port, 1 to 49151,0 is any port.

6		InOut							
7		connect	TCON_IP_v4	4.0					
8		InterfaceId	HW_ANY	0.0					HW-identifier of IE-interface submodule
9		ID	CONN_OUC	2.0					connection reference / identifier
10		ConnectionType	Byte	4.0					type of connection: 11=TCP/IP, 19=UDP (17=TCP/IP)
11		ActiveEstablished	Bool	5.0					active/passive connection establishment
12		RemoteAddress	IP_V4	6.0					remote IP address (IPv4)
13		ADDR	Array[1..4] of Byte	6.0					IPv4 address
14		ADDR[1]	Byte	6.0					IPv4 address
15		ADDR[2]	Byte	7.0					IPv4 address
16		ADDR[3]	Byte	8.0					IPv4 address
17		ADDR[4]	Byte	9.0					IPv4 address
18		RemotePort	UInt	10.0					remote UDP/TCP port number
19		LocalPort	UInt	12.0					local UDP/TCP port number

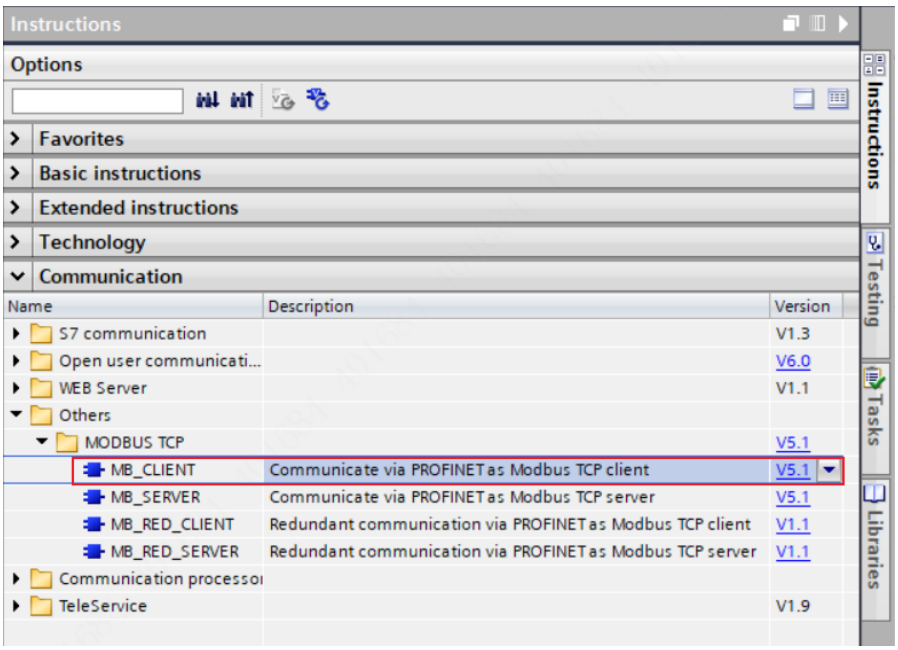
PLC_1 [CPU 1214C DC/DC/DC] Properties Info

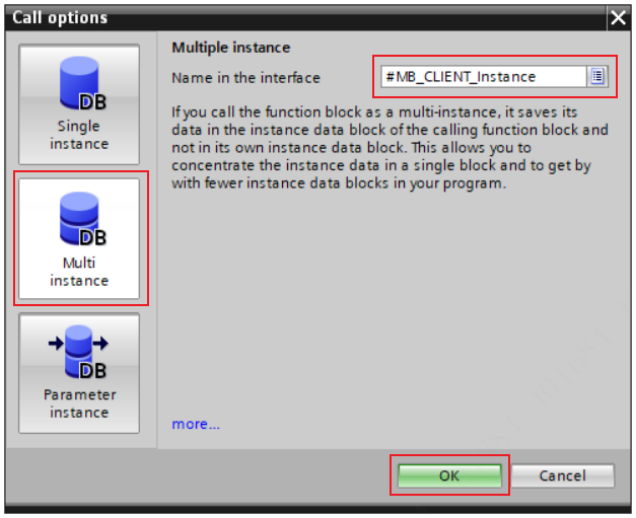
General | IO tags | System constants | Texts

Show hardware system constant

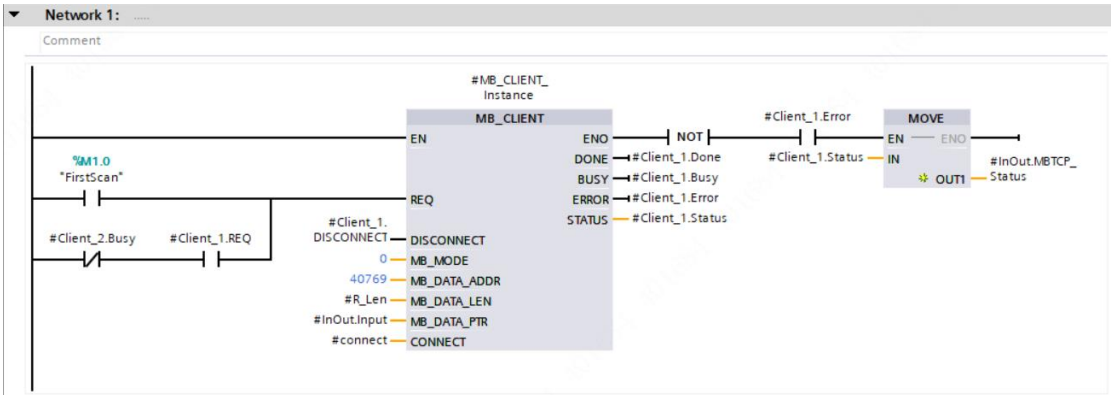
Name	Type	Hardware identi.	Used by	Comment
Local~MC	Hw_SubModule	51	PLC_1	
Local~Common	Hw_SubModule	50	PLC_1	
Local~Device	Hw_Device	32	PLC_1	
Local~Configuration	Hw_SubModule	33	PLC_1	
Local~Exec	Hw_SubModule	52	PLC_1	
Local	Hw_SubModule	49	PLC_1	
Local~PROFINET_接口_1	Hw_Interface	64	PLC_1	
Local~HSC_1	Hw_Hsc	257	PLC_1	
Local~HSC_2	Hw_Hsc	258	PLC_1	

After the variable is created, return to the ladder diagram interface and select it on the right side of the programming interface [Options] → [Communication] → [Others] → [Modbus TCP] → [MB_CLIENT]. Hold down the left mouse button and drag it into the program. In the pop-up interface, select [Multiple Instance], then choose the previously created Static Variable in the [Name in the interface], and click OK.





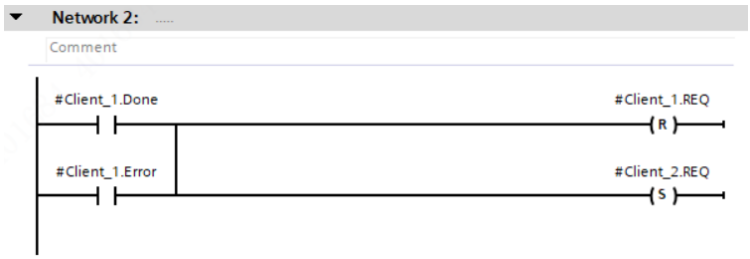
Repeat the above operation, add another MB_CLIENT instruction to respectively achieve the functions of data reception and transmission, and assign values to the corresponding pins of the instruction as shown in the following figure.



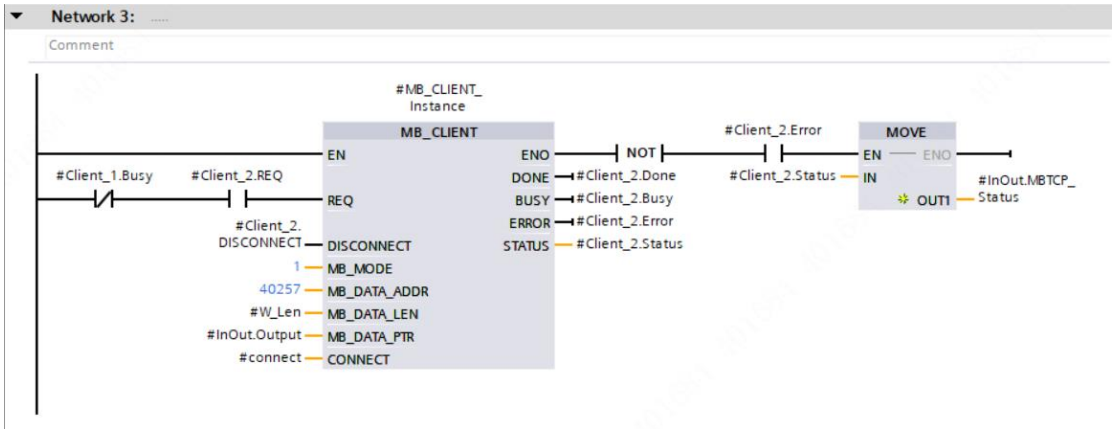
Network 1: When the PLC is turned on for the first scanning cycle or when [Client_1.REQ] is set, the [REQ] pin is set, and the instruction sends a communication request. The [DISCONNECT] pin in the instruction can control the establishment/termination of a connection with the Modbus server. By default, a connection is established with 0. The combination of [MB_MODE], [MB_DATA_ADDR], and [MB_DATA_LEN] defines the ModbusTCP function code used in the current instruction, as well as parameters such as read/write, register address, and data length. In this case, The result data of the vision sensor is stored in the hold register, 03 function code, and a 20-character address starting from 0X300. Therefore, the corresponding combination of parameters [MB_MODE], [MB_DATA_ADDR], and [MB_DATA_LEN] is 0,40769,20. Among them, [MB_DATA_LEN] is connected to the input variable [R_Len], which is controlled by an external pin for convenient modification and debugging.

The following table shows the relationship between the input parameters MB_MODE, MB_DATA_ADDR, MB_DATA_LEN of the "MB_CLIENT" instruction and the associated Modbus function.

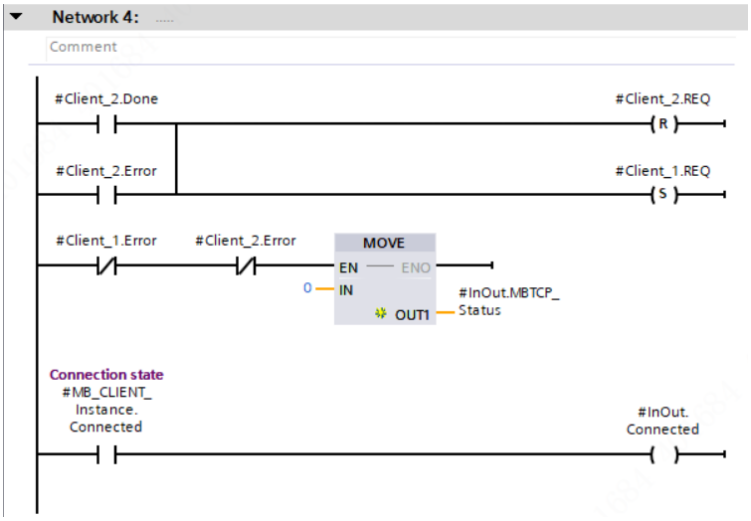
MB_MODE	MB_DATA_ADDR	MB_DATA_LEN	Modbus function	Function and data type
0	1 to 9,999	1 to 2,000	01	Read 1 to 2,000 output bits on the remote address 0 to 9,998
0	10,001 to 19,999	1 to 2,000	02	Read 1 to 2,000 input bits on the remote address 0 to 9,998
0	• 40,001 to 49,999 • 400,001 to 465,535	1 to 125	03	• Read 1 to 125 holding registers on the remote address 0 to 9,998 • Read 1 to 125 holding registers on the remote address 0 to 65,534
0	30,001 to 39,999	1 to 125	04	Read 1 to 125 input words on the remote address 0 to 9,998



Network 2: When the MB_CLIENT instruction in Network 1 completes a bit or reaches error position 1, reset [Client_1.REQ], stop sending communication requests, and simultaneously set [Client_2.REQ].

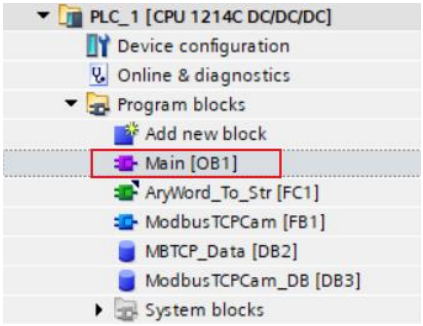


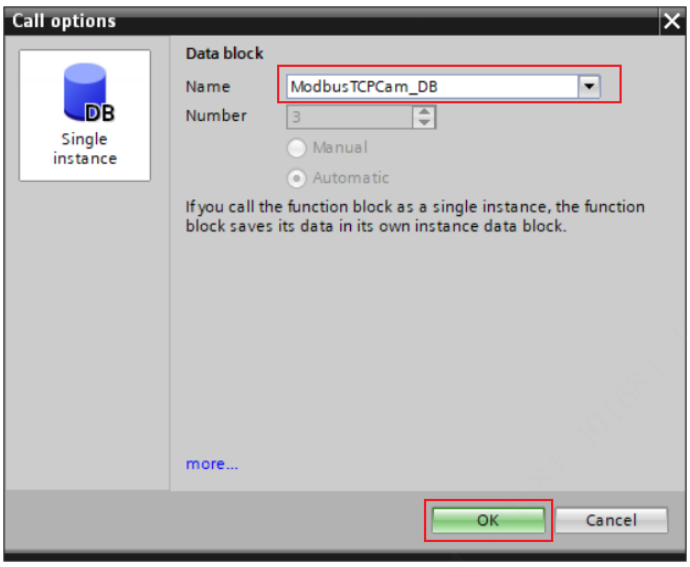
Network 3: When [Client_2.REQ] is set and [Client_2.Busy] is not set, the [REQ] pin is set, and the instruction sends a communication request. The vision sensor input command is saved in the hold register, 16 function code, and a three-character address starting from 0X100. Therefore, the corresponding combination of the parameters MB_MODE, MB_DATA_ADDR, and MB_DATA_LEN is 1,40257,3. Among them, [MB_DATA_LEN] is connected to the input variable [W_Len] and controlled by an external pin, which is convenient for modification and debugging.

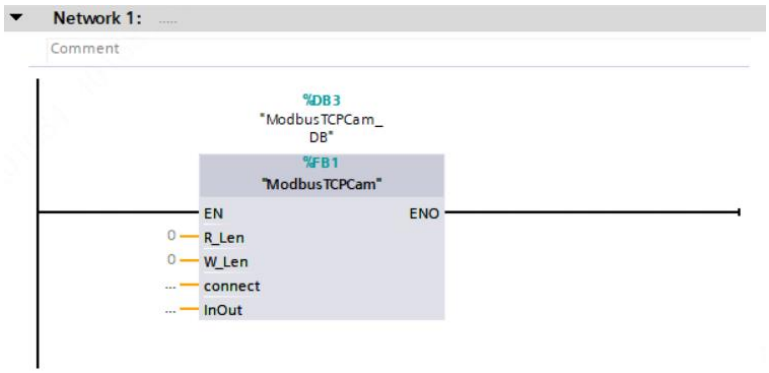


Network 4: When the MB_CLIENT instruction in Network 3 completes a bit or reaches the error position 1, reset [Client_2.REQ], stop sending communication requests, and simultaneously set [Client_1.REQ] to jointly implement the polling operation with Network 2. The current connection status can be obtained through the static variable [Connected] in the MB_CLIENT instruction instance, and the connection status can be mapped to the external InOut variable. It is convenient for debugging personnel to check the connection status.

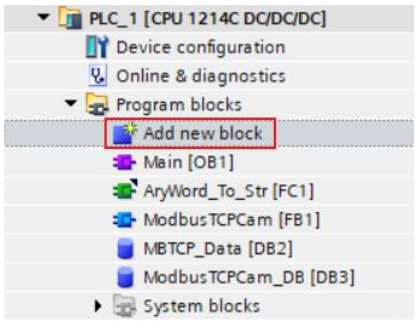
Double-click the [Main] main program in the project tree to enter the main program interface. Drag the newly created FB function block in the previous step into the main program with the left mouse button. Change the name of the instance DB block as needed and click OK.

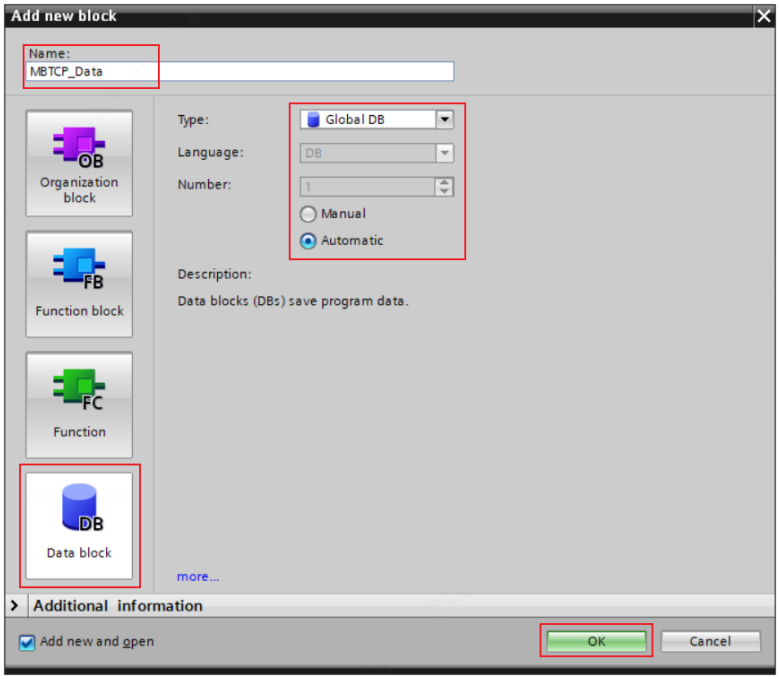


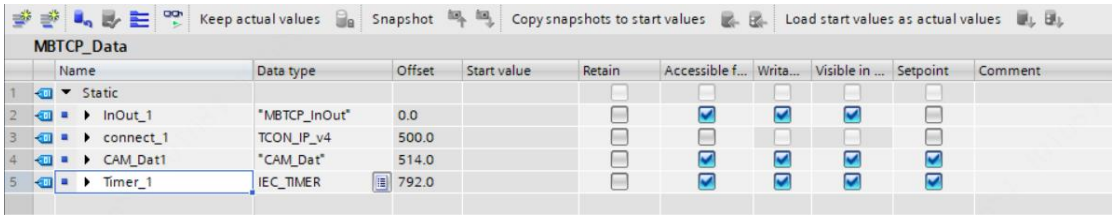




Click [Add New Block], select [Data Block] After changing the name and confirming, add the following variables to the data block.

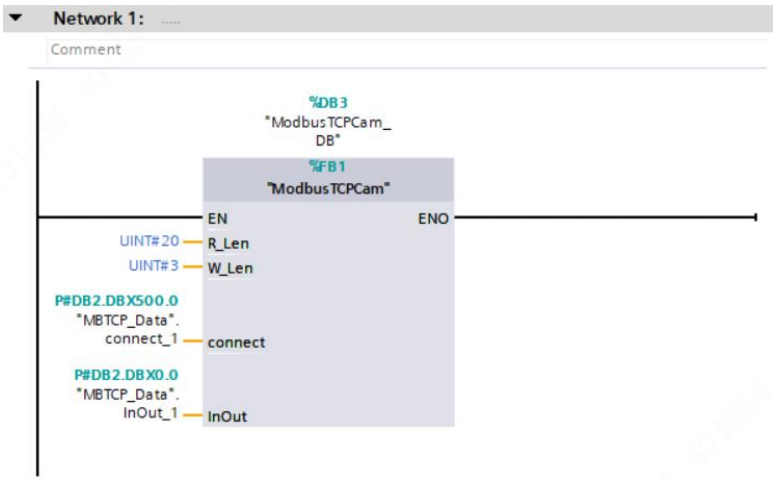






Among them, [connect_1] contains information related to ModbusTCP communication. The starting value is modified according to the communication Settings of the actual connected vision sensor. In this case, the starting value is modified as shown in the following figure.

To add the variables in the data block in the previous step, drag them to the pins of the function block in the main program. Here, the [R_Len] pin represents the number of registers needed to read the result data from the vision sensor, and fill in 20. The [W_Len] pin represents the length of the registers needed to write the command to the vision sensor, and fill in 3. (Note: Although the data type of [trigger] and [statistical reset] configured in the vision sensor formatting input string is Bit, they still occupy 1Word of space in ModbusTCP communication. Similarly, although the data type of [Switch project ID] is Byte, it still occupies 1Word of space.)



At Network 2 of the main program, insert the SCL Network to convert the byte sets sent and received by ModbusTCP communication into data types corresponding to the formatted input and output of the vision sensor communication configuration.

Lines 1-6 of the program: When ["MBTCP_Data".CAM_Dat1.Trigger] is set, the PLC sends a 1 to the vision sensor hold register 0x100 via ModbusTCP, and the vision sensor takes a photo; otherwise, it will send a zero.

Lines 8-12 of the program: When ["MBTCP_Data".CAM_Dat1.StatisticsReset] is set, the PLC sends 1 to the vision sensor hold register 0x101 via ModbusTCP, and the [Running Total] in the vision sensor is reset; otherwise, zero is sent.

Line 14 of the program:

Modify the value of ["MBTCP_Data".CAM_Dat1.SwitchProjectId], and the PLC sends the corresponding value to the holding register 0x102 of the vision sensor via ModbusTCP. The [project ID] in the vision sensor will change. If the new project being switched is not configured with ModbusTCP communication, Then the communication will be interrupted. Please be cautious when switching.

```
16  //
17  //
18 "MBTCP_Data".CAM_Dat1.RunningTotal := DWORD_TO_UDINT(IN := (SHL(IN := WORD_TO_DWORD(IN := "MBTCP_Data".InOut_1.Input[1]), N := 16)
19                                       OR WORD_TO_DWORD(IN := "MBTCP_Data".InOut_1.Input[0])));
20
21  //
22 "MBTCP_Data".CAM_Dat1.LineExist_StartPointX := DWORD_TO_REAL(IN := (SHL(IN := WORD_TO_DWORD(IN := "MBTCP_Data".InOut_1.Input[3]), N := 16)
23                                       OR WORD_TO_DWORD(IN := "MBTCP_Data".InOut_1.Input[2])));
24
25  //
26 "MBTCP_Data".CAM_Dat1.LineExist_EndPointX := DWORD_TO_REAL(IN := (SHL(IN := WORD_TO_DWORD(IN := "MBTCP_Data".InOut_1.Input[5]), N := 16)
27                                       OR WORD_TO_DWORD(IN := "MBTCP_Data".InOut_1.Input[4])));
28
29  //
30 "MBTCP_Data".CAM_Dat1.GrayscaleArea_Num := DWORD_TO_UDINT(IN := (SHL(IN := WORD_TO_DWORD(IN := "MBTCP_Data".InOut_1.Input[7]), N := 16)
31                                       OR WORD_TO_DWORD(IN := "MBTCP_Data".InOut_1.Input[6])));
32
33  //
34 "MBTCP_Data".CAM_Dat1.OCR_StringLength := DWORD_TO_UDINT(IN := (SHL(IN := WORD_TO_DWORD(IN := "MBTCP_Data".InOut_1.Input[9]), N := 16)
35                                       OR WORD_TO_DWORD(IN := "MBTCP_Data".InOut_1.Input[8])));
36
```

Lines 18-19 of the program: Since the results received by the PLC from the vision sensor are stored in word array, but the [Running Total] in the vision sensor is of UDINT(32-bit unsigned double integer) type, it is necessary to convert the two characters to this data type. Therefore, the merging and conversion are carried out through the steps of "Word→DWord→ high character left shift 16 bits → high and low character OR operation →DWord to UDInt".

Lines 22-23 of the program: The conversion method of [LineExist.StartPointX] is similar to the previous step, with the only difference being that the DWORD is converted to the Real type at the end.

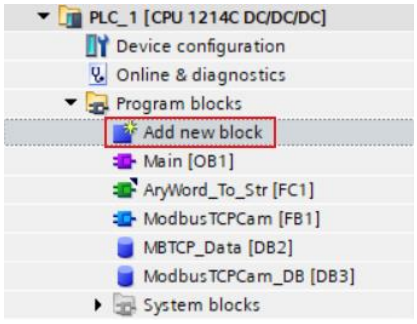
Lines 26-27 of the program: The same as above.

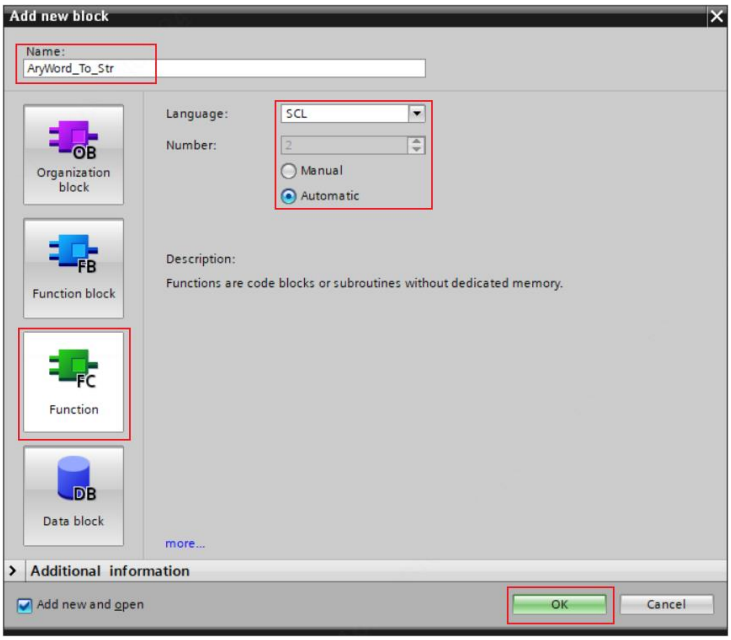
Lines 30-31 of the program: The conversion method is the same as [Running Total].

Lines 34-35 of the program: The conversion method is the same as that of [Running Total].

For the string-type data in the result data of the vision sensor, since there are no relevant instructions for converting word array to strings in the Siemens PLC, to facilitate program writing and reading, this function is achieved by writing a word array to string conversion function.

Double-click on [Add New Block] in the project tree, select [Function], choose [SCL] for the programming language, automatically number, modify the function name, and click [OK].





In the opened function, click on [Block Interface] to add the following variables.

AryWord_To_Str

	Name	Data type	Offset	Default value	Comment
1	Input				
2	Ary_Word	Variant			
3	pChars	Int			
4	Cnt	Int			
5	Output				
6	Strg	String			
7	InOut				
8	<Add new>				
9	Temp				
10	i	Int	0.0		
11	Temp_Word	Array[0..123] of Word	2.0		
12	Temp_Byte	Array[0..255] of Byte	250.0		
13	Constant				
14	<Add new>				
15	Return				
16	AryWord_To_Str	Void			

Add the following program to the function to implement the operation of converting Chars to string.

```
IF TypeOf(#Ary_Word) = TypeOf(#Temp_Word) THEN
    VariantGet(SRC := #Ary_Word,
               DST => #Temp_Word);
END_IF;

FOR #i := 0 TO (#Cnt - 1) DO
    IF (#i MOD 2) = 0 THEN
        #Temp_Byte[#i] := WORD_TO_BYTE(IN := #Temp_Word[#pChars + (#i / 2)]);
    ELSE
        #Temp_Byte[#i] := WORD_TO_BYTE(IN := SHR_WORD(IN := #Temp_Word[#pChars + (#i / 2)], N := 8));
    END_IF;
END_FOR;

Chars_TO_Strg(Chars:=#Temp_Byte,
              pChars:=0,
              Cnt:=INT_TO_UINT(IN := #Cnt),
              Strg=>#Strg);
```

Lines 1-4 of the program: Determine whether the data types of [Ary_Word] and [Temp_Word] are the same. If they are the same, pass the data of [Ary_Word] into [Temp_Word].

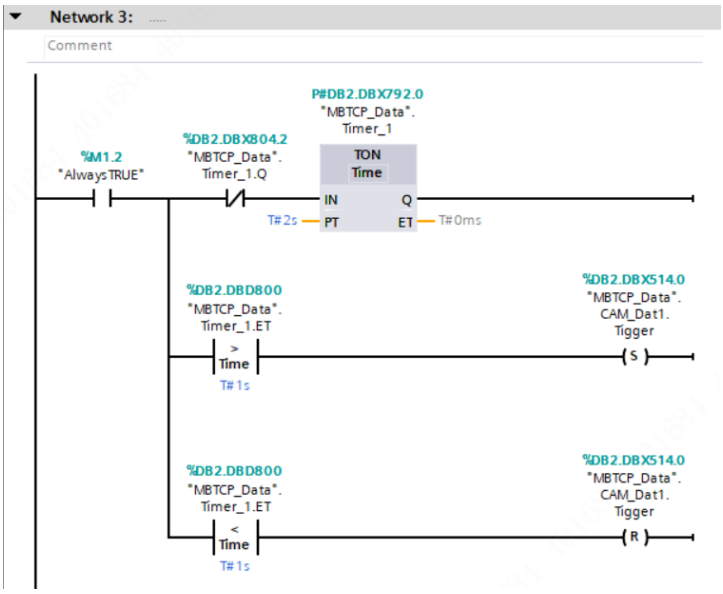
Lines 6-12 of the program use a FOR loop to split the words in the word array into high and low bytes and pass them into the intermediate variable of the byte array. During the process of splitting the high and low bytes, take the remainder of 2 from the independent variable i of the FOR loop. If the remainder is 0, it indicates that the current byte is low. By converting the data type from words to bytes, the low byte data can be extracted and passed into the intermediate variable of the byte array. If the remainder is not 0, it indicates that the current byte is high. The data needs to be shifted 8 bits to the right first, and then the data type needs to be converted from word to byte before the high byte data can be extracted and passed into the intermediate variable of the byte array.

Lines 14-17 of the program: Convert the byte array to a string through the [Chars_TO_Strg] instruction.

After the function is written, drag the written function into the SCL segment of the main program from the left project tree and fill in the corresponding pins.

```
"AryWord_To_Str"(Ary_Word:="MBTCP_Data".InOut_1.Input,
                 pChars:=10,
                 Cnt:=20,
                 Strg=>"MBTCP_Data".CAM_Dat1.OCR_StringContent);
```

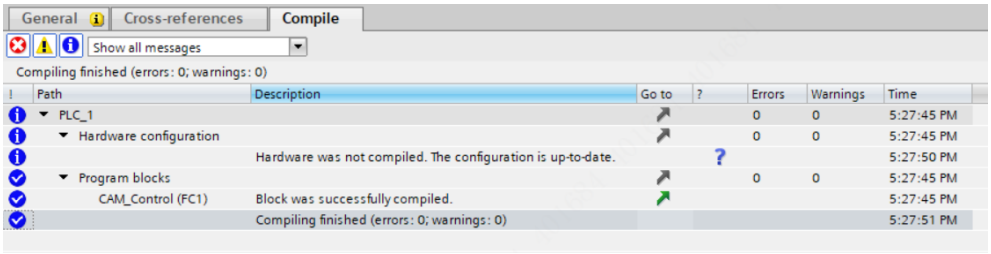
For the convenience of testing, the following program is added at Network 3 of the main program to achieve timed triggering. The timing can be modified by modifying the timer pins and judgment conditions. If not in use, changing [AlwaysTRUE] to a normally closed contact can block this function.

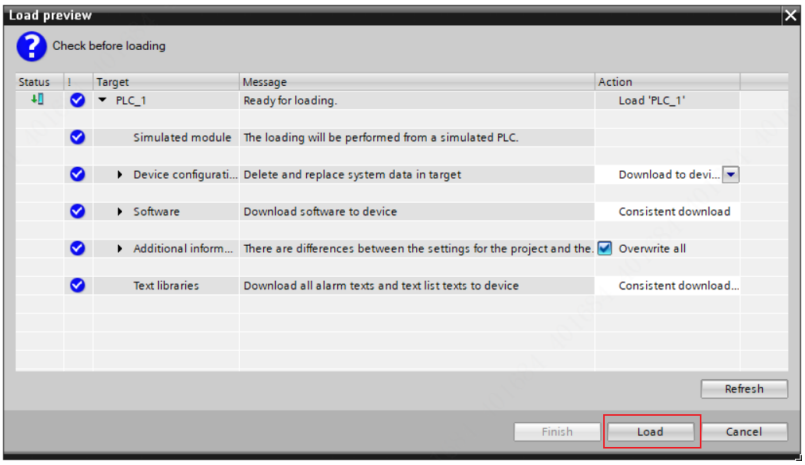


6.2 Communication Test

6.2.1 Download Program

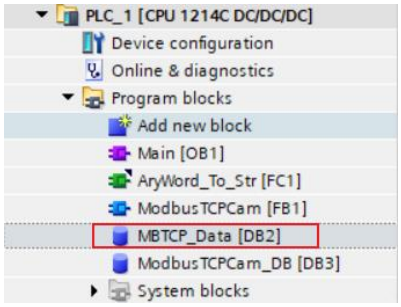
Select the PLC device in the project tree and click the Compile button, perform a full compilation of all programs. When the compilation is displayed as complete without errors, it indicates the compilation has passed. Click the Download button, select Reinitialize, and then click [Load].

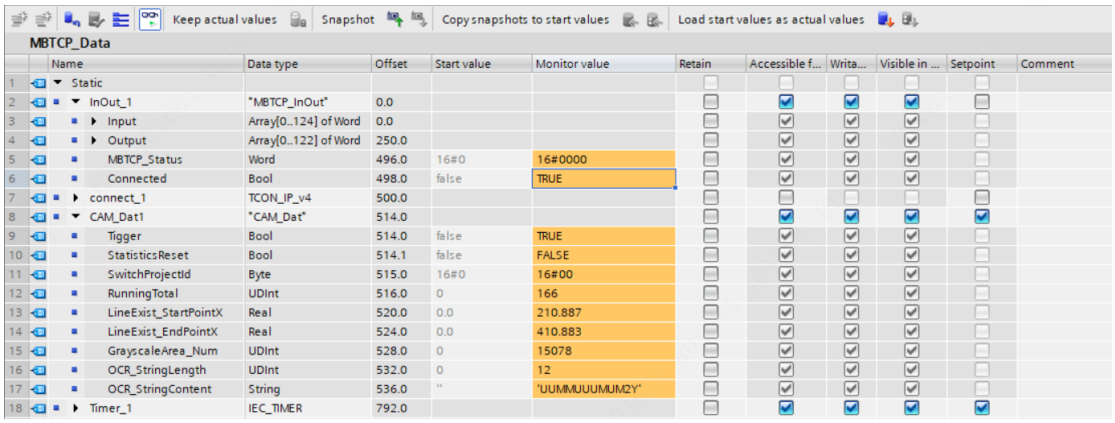




6.2.2 Communication Test

In the Project Tree, double-click the [MBTCP _Data] Data block to open it. Then, click the [All Monitoring] button  to monitor the variable numbers in the Data block.





[InOut_1.Connected] display shows [TRUE], indicating a successful connection between the PLC and the vision sensor, while [FALSE] indicates a failed connection. At this point, it is necessary to check whether the connection Settings of the PLC are configured correctly as per the above instructions. If they are correct, check whether the communication Settings of the vision sensor are correct.

To send instructions to the vision sensor, simply double-click the monitoring value of the sent instruction to modify it.

Note: When the vision sensor is connected to the PLC, if the PLC is powered off or placed in STOP mode and then restarted, the vision sensor and the PLC can be connected normally, but data transmission/reception will fail. At this time, the communication configuration of the vision sensor needs to be reconfigured or the vision sensor needs to be restarted to restore it.