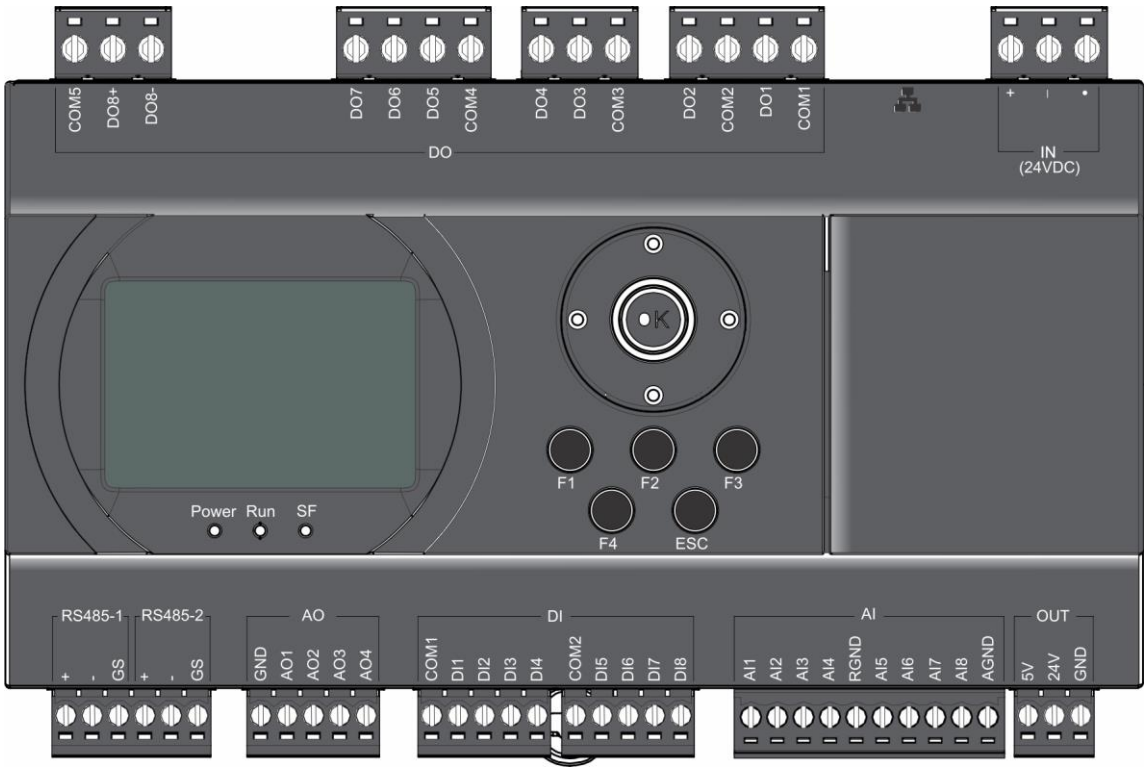
SystemePLC Studio

SystemePLC Studio Software

User Manual

for Smart relay SystemePLC SR and PLC SystemePLC S172



systeme.ru

Système electric

Contents

Contents II
Versions VIII
1 Product introduction ..... - 9 -
1.1 Overview ..... - 10 -
1.2 SM172 controller ..... - 10 -
1.2.1 Component description ..... - 10 -
1.2.2 Technical specifications ..... - 12 -
1.2.3 External wiring diagram ..... - 13 -
1.3 SR2 controller ..... - 15 -
1.3.1 Technical specifications ..... - 15 -
1.3.2 Component description and external wiring ..... - 17 -
1.4 SR1 controller ..... - 24 -
1.4.1 Technical specifications ..... - 24 -
1.4.2 Component description and external wiring ..... - 26 -
1.5 Interface definitions ..... - 32 -
1.6 Expansion modules ..... - 32 -
1.6.1 SM172EAM0800 ..... - 33 -
1.6.2 SM172EDM0800 ..... - 34 -
1.6.3 SM172EDM0800P7 ..... - 35 -
1.6.4 SM172EDM0810 ..... - 36 -
1.6.5 SM172EMIO1000 ..... - 37 -
1.6.6 SM172EDM1600 ..... - 38 -
1.6.7 SM172EDM2800 ..... - 39 -
1.6.8 SM172EMIO2800 ..... - 40 -
1.7 Working environment ..... - 41 -
1.8 System architecture ..... - 42 -
2 Installing ..... - 43 -
2.1 Installation precautions ..... - 44 -
2.2 Installation dimensions ..... - 44 -
2.3 Installation method ..... - 46 -
2.4 Grounding and wiring ..... - 48 -
3 Configuration of a simple project ..... - 49 -
3.1 SystemePLC Studio and PLC hardware connection ..... - 50 -
3.2 PC system requirements for SystemePLC Studio installation ..... - 51 -
3.3 New project ..... - 51 -
3.4 Hardware configuration ..... - 54 -
3.5 Programming ..... - 56 -
3.6 SystemePLC Studio communication with PLC ..... - 59 -
3.7 Program compilation and download ..... - 62 -
3.8 Monitoring ..... - 65 -
3.9 Global variables ..... - 67 -

3.10 Use of branchers ..... - 68 -
3.11 Use of FDO ..... - 69 -
3.12 Program working status ..... - 71 -
3.13 Task ..... - 71 -
3.13.1 Task configuration ..... - 71 -
3.13.2 Task configuration principle ..... - 73 -
3.14 Expansion module status ..... - 74 -
3.15 Upload ..... - 76 -
3.15.1 Project upload ..... - 76 -
3.15.2 Bin files upload ..... - 77 -
4 Communication configuration ..... - 78 -
4.1 Modbus RTU communication ..... - 79 -
4.1.1 Modbus RTU master communication configuration ..... - 79 -
4.1.2 Modbus RTU slave communication configuration ..... - 84 -
4.2 Modbus TCP communication ..... - 86 -
4.2.1 Modbus TCP master communication configuration ..... - 86 -
4.2.2 Modbus TCP slave communication configuration ..... - 89 -
4.3 Modbus Diagnosis ..... - 90 -
4.4 Commissioning ..... - 92 -
4.5 FBD block pin inversion function ..... - 96 -
4.6 FBD connection break function ..... - 97 -
4.7 Modbus customer editor ..... - 98 -
4.8 Display ..... - 100 -
4.8.1 Overview ..... - 100 -
4.8.2 Page introduction ..... - 100 -
4.8.3 Toolbar ..... - 101 -
4.8.4 HMI properties ..... - 102 -
4.8.5 Applications of static ..... - 103 -
4.8.6 Applications of string list ..... - 105 -
4.8.7 Edit box ..... - 107 -
4.8.8 Image ..... - 112 -
4.8.9 Animation ..... - 114 -
4.8.10 Buttons ..... - 116 -
5 Instruction description ..... - 120 -
5.1 Basic type conversion ..... - 121 -
5.2 Basic type conversion BCD conversion ..... - 121 -
5.3 TIM conversion ..... - 121 -
5.3.1 DATE AND TIME TO TIME OF DAY ..... - 121 -
5.3.2 DATE AND TIME TO DATE ..... - 121 -
5.4 Numerical ..... - 122 -
5.4.1 ABS ..... - 122 -
5.4.2 SQRT ..... - 122 -
5.4.3 LN ..... - 122 -
5.4.4 LOG ..... - 122 -
5.4.5 EXP ..... - 123 -
5.4.6 SIN ..... - 123 -

5.4.7 COS - 123 -
5.4.8 TAN - 123 -
5.4.9 ASIN - 124 -
5.4.10 ACOS - 124 -
5.4.11 ATAN - 124 -
5.5 Airthmetic - 125 -
5.5.1 ADD - 125 -
5.5.2 MUL - 125 -
5.5.3 SUB - 125 -
5.5.4 DIV - 126 -
5.5.5 MOD - 126 -
5.5.6 EXPT - 126 -
5.5.7 MOVE - 126 -
5.6 Time - 127 -
5.6.1 ADD_TIME - 127 -
5.6.2 ADD_TOD_TIME - 127 -
5.6.3 MULTIME - 127 -
5.6.4 SUB_TIME - 128 -
5.6.5 SUB_DATE_DATE - 128 -
5.6.6 SUB_TOD_TIME - 128 -
5.6.7 SUB_DT_TIME - 128 -
5.6.8 SUB_DT_DT - 129 -
5.6.9 DIVTIME - 129 -
5.7 Bit Shift - 129 -
5.7.1 SHL - 129 -
5.7.2 SHR - 130 -
5.7.3 ROR - 130 -
5.7.4 ROL - 130 -
5.8 Bit wise - 131 -
5.8.1 AND - 131 -
5.8.2 OR - 131 -
5.8.3 XOR - 131 -
5.8.4 NOT - 132 -
5.9 Selection - 132 -
5.9.1 SEL - 132 -
5.9.2 MAX - 133 -
5.9.3 MIN - 133 -
5.9.4 LIMIT - 133 -
5.9.5 MUX - 134 -
5.10 Comparision - 134 -
5.10.1 GT - 134 -
5.10.2 GE - 135 -
5.10.3 EQ - 135 -
5.10.4 LT - 135 -
5.10.5 LE - 135 -
5.10.6 NE - 136 -
5.11 Character string - 136 -
5.11.1 CONCAT - 136 -

5.11.2 CONCAT DATE TOD - 136 -
5.11.3 DELETE - 137 -
5.11.4 FIND - 137 -
5.11.5 LEFT - 137 -
5.11.6 LEN - 138 -
5.11.7 MID - 138 -
5.11.8 REPLACE - 138 -
5.11.9 RIGHT - 139 -
5.12 Standard function blocks - 139 -
5.12.1 SR - 139 -
5.12.2 RS - 140 -
5.12.3 R_TRIG - 140 -
5.12.4 F_TRIG - 140 -
5.12.5 CTU - 141 -
5.12.6 CTU_LINT - 141 -
5.12.7 CTU_UDINT - 142 -
5.12.8 CTU_ULINT - 142 -
5.12.9 CTD - 142 -
5.12.10 CTD_LINT - 143 -
5.12.11 CTD_UDINT - 143 -
5.12.12 CTD_ULINT - 144 -
5.12.13 CTUD - 144 -
5.12.14 CTUD_LINT - 145 -
5.12.15 CTUD_UDINT - 146 -
5.12.16 CTUD_ULINT - 146 -
5.12.17 TP - 147 -
5.12.18 TON - 148 -
5.12.19 TOF - 148 -
5.13 Smart function blocks - 149 -
5.13.1 GET_TICKS - 149 -
5.13.2 GET_RTC - 149 -
5.13.3 SET_RTC - 150 -
5.13.4 POPUP_PAGE - 150 -
5.13.5 GEN_RAND - 151 -
5.13.6 PIDAdvanced - 151 -
5.13.7 PIDAutoTuning - 156 -
5.13.8 advPID - 159 -
5.14 Bit conversion functions - 166 -
5.14.1 BitToByte - 166 -
5.14.2 BitToWord - 166 -
5.14.3 ByteToBit - 168 -
5.14.4 WordToBit - 168 -
5.14.5 WordToDWord - 170 -
5.14.6 DWordToLWord - 170 -
5.14.7 DWordToWord - 170 -
5.14.8 LWordToDWord - 171 -
5.15 Special functions - 172 -
5.15.1 Hours_Counter - 172 -

5.15.2 Threshold_Trigger ..... - 173 -
5.15.3 UpDown_Counter ..... - 174 -
5.15.4 Analog_ Amplifier ..... - 176 -
5.15.5 Analog_Comparator ..... - 176 -
5.15.6 Analog_DifferentialTrigger ..... - 178 -
5.15.7 Analog_Filter ..... - 180 -
5.15.8 Analog_MUX ..... - 181 -
5.15.9 Analog_Ramp ..... - 182 -
5.15.10 Analog_ThresholdTrigger ..... - 183 -
5.15.11 Analog_Watchdog ..... - 184 -
5.15.12 AverageValue ..... - 186 -
5.15.13 Mathematicinstruction ..... - 188 -
5.15.14 MaxMin ..... - 190 -
5.15.15 PWM ..... - 192 -
5.15.16 Latching_Relay ..... - 194 -
5.15.17 Pulse_Relay ..... - 195 -
5.15.18 Shift_Register ..... - 196 -
5.15.19 Asynchronous_PulseGenerator ..... - 197 -
5.15.20 EdgeTriggered_WipingRelay ..... - 198 -
5.15.21 MultipleFunctionSwitch ..... - 199 -
5.15.22 Off_Delay ..... - 201 -
5.15.23 On_Delay ..... - 202 -
5.15.24 Onoff_Delay ..... - 202 -
5.15.25 Random_Generator ..... - 203 -
5.15.26 Retentive_OnDelay ..... - 204 -
5.15.27 StairwayLightingSwitch ..... - 205 -
5.15.28 Stopwatch ..... - 206 -
5.15.29 Weekly_Timer ..... - 207 -
5.15.30 Wiping_Relay ..... - 209 -
5.15.31 Yearly_Timer ..... - 210 -

6 LD program ..... - 213 -

6.1 Characteristics of Programming Elements ..... - 214 -
6.1.1 Contact Elements ..... - 215 -
6.1.2 Coil Elements ..... - 215 -
6.1.3 Elements Introduction ..... - 215 -
6.1.4 Function Block Elements ..... - 217 -
6.2 Creating LD Main Programs and Subprograms ..... - 218 -
6.3 Adding Contact Call Points, Adding Inverted Contacts, or Modifying Inverted Contacts ..... - 220 -
6.4 Branches ..... - 220 -
6.5 Adding Instructions ..... - 222 -
6.6 Quick Call of Points on Instructions ..... - 222 -
6.7 Deleting Contacts or Instructions ..... - 223 -
6.8 Crossover Points ..... - 223 -
6.9 Example of a Chasing Light Program ..... - 224 -
6.10 Display of LD Program Execution Status ..... - 225 -

7 Fault ..... - 227 -

8 Appendix .....- 230 -
8.1 Checking the IP address of the PLC .....- 231 -
8.2 Modifying the PLC IP and gateway .....- 231 -
8.3 PLC run/stop.....- 232 -
8.4 PLC clock.....- 233 -
8.5 Checking the PLC baud rate .....- 234 -
8.6 Firmware upgrades.....- 234 -
8.7 PLC info .....- 234 -
8.8 LCD.....- 234 -
8.9 Update firmware from USB flash.....- 235 -
8.10 Update firmware from USB type-C cable .....- 235 -
8.11 Update firmware from Ethernet cable.....- 236 -
8.12 Expansion bus characteristics.....- 238 -
8.13 IO Mapping .....- 239 -
8.14 Ordering information.....- 242 -

Versions

Release date	Versions	Revision
08/2026	V1.10	•Add 3.15.Upload •Add advPID •Add 8.13 IO Mapping
06/2026	V1.00	•Add 6.1.3 Elements Introduction
05/2026	V0.90	•Add 4.3 Modbus Diagnosis
12/2025	V0.80	•Add LD program, See LD program
12/2025	V0.71	•Modify the Solid State Relay output
11/2025	V0.70	•Add Product specifications 1.3.2.4 ZR2PP11P7, 1.3.2.5 ZR2PP11BD, 1.6.1 SM172EAM0800, 1.6.3 SM172EDM0800P7, 1.6.4 SM172EDM0810, 1.6.6 SM172EDM1600, 1.6.7SM172EDM2800. •Add 3.2 PC system requirements for SystemePLC Studio installation •Add 3.12 Program working status •Add 3.13 Task •Add 3.14 Expansion module status •Update 6 Fault •Add 7.9 Update Firmware from USB flash •Add 7.10 Update Firmware from USB Type-C cable •Add 7.11 Update Firmware from Ethernet cable •Add 7.12 Expansion Bus Characteristics
8/2025	V0.60	•Add 4.8 FBD block pin inversion function •Add 4.9 FBD connection break function •Add 4.11 Display •Add 6 Fault
5/2025	V0.50	•Add Product specifications •Add Commissioning, refer to section 4.4 Commissioning for details •Add a description of global variables, please refer to section 4.5 Global Variables for details •Add branching instructions, refer to section 4.6 Use of Branchers for details •Add instructions to explain, please refer to Chapter 5 Instruction Descriptionfor details •Add screen menu instructions, refer to Chapter 6 Appendix for details
9/2024	V0.30	•First edition

First section

Product introduction

1.1 Overview

1.2 SM172 controller

1.3 SR2 controller

1.4 SR1 controller

1.5 Interface definitions

1.6 Expansion modules

1.7 Working environment

1.8 System architecture

1.1 Overview

There are currently 22 models of Smart Relay series PLC products:

Model	
SM172 controller	SM172PS11BDT, SM172PS11BDR, SM172PS11BDM.
SR2 controller	ZR2PB11P7, ZR2PA11BD, ZR2PP11BD2A, ZR2PP11P7, ZR2PP11BD.
SR1 controller	ZR1PA00P7, ZR1PB00P7, ZR1PB00BD, ZR1PA00BD, ZR1PP00BD2A, ZR1PP00BD4A.
Extension module	SM172EAM0800, SM172EDM0800,. SM172EDM0800P7, SM172EDM0810, SM172EMIO1000, SM172EDM1600, SM172EDM2800, SM172EMIO2800.

1.2 SM172 controller

1.2.1 **Component description**

The appearance diagrams of SM172PS11BDR, SM172PS11BDT, and SM172PS11BDM are as follows:

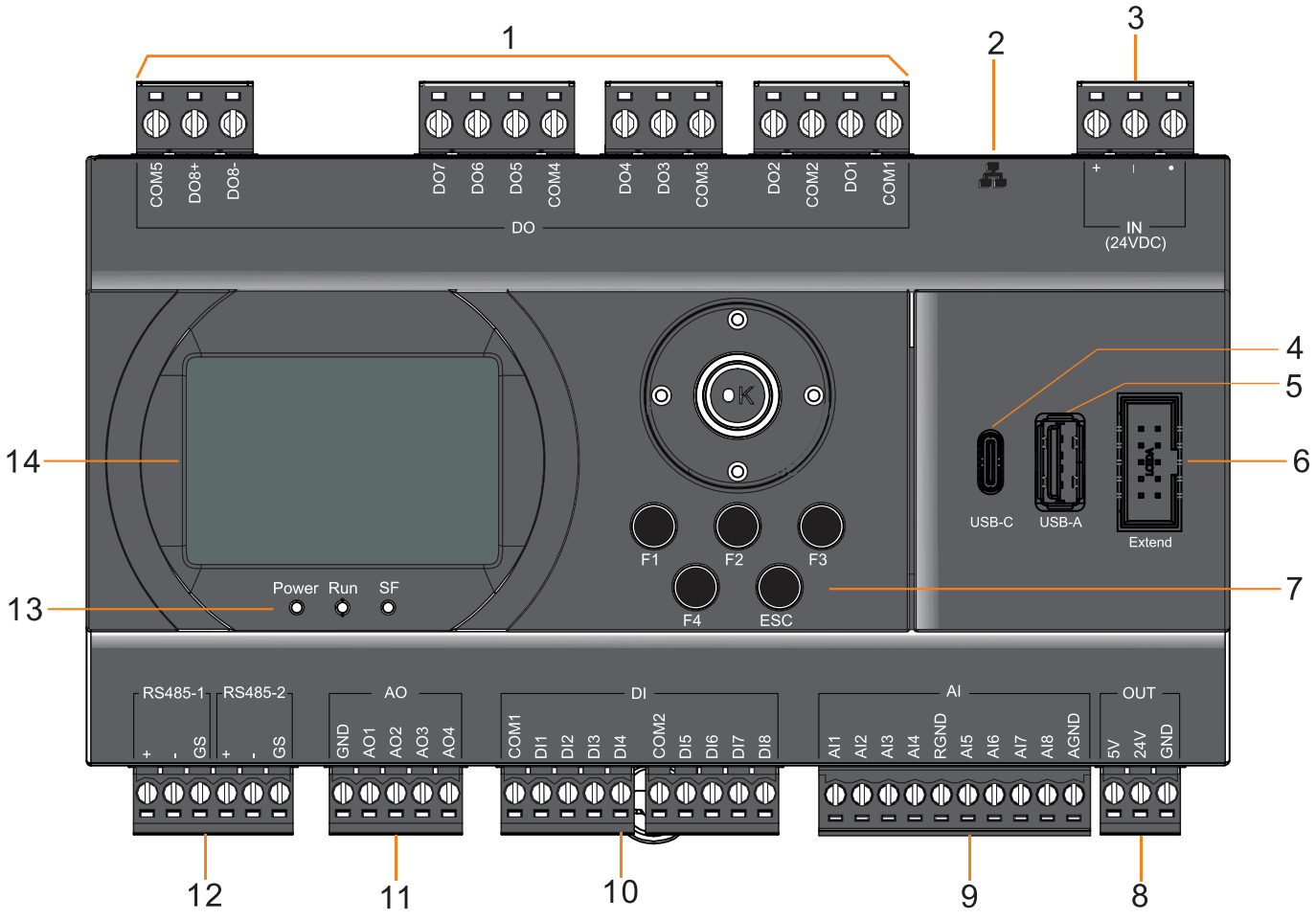


Table 1-1. Interface Description of SM172PS11BDT, SM172PS11BDR, SM172PS11BDM

No.	Interface	Description
1.	Digital output （DO）	**SM172PS11BDR:** 8 channel outputs (3A electromagnetic relay output).
		SM172PS11BDT: 8 channel outputs (0.5A transistor drain output). DO1~DO2 are fast outputs(FDO)
		SM172PS11BDM: 6 channel 3A electromagnetic relay outputs, 2 channel 0.3A/240VAC solid-state relay outputs.
2.	Ethernet interface	1 channel, support MODBUS TCP master-slave function, ntp/sntp time synchronisation service, DHCP/bootp automatic IP acquisition function, webserver website service.
3.	Power source interface	24V DC power supply input
4.	Type C interface	For connecting to a computer and communicating with programming software.
5.	Type A	Used to read and write USB flash drive files.
6.	Expansion Module Interface	Supports connecting 7 Extension modules, The expansion bus supports extension, with the following constraints: only one expansion bus is allowed to be extended, and the length of the extension cable shall not exceed 1 meter. Otherwise, the modules behind the extension cable cannot be correctly detected. Hot-swapping is not supported for this interface.
7.	KEY	10 buttons, including 6 buttons for up, down, left, right, ok, and esc to control the screen menu, and F1, F2, F3, and F4 user-defined function codes.
8.	Power supply output	Provide external 5V and 24V power supply interfaces with a power of 1W to supply power to active sensors.
9.	Analog input (AI) Resolution of 16 bits	**8 channel analogue inputs** Voltage signal: 0-10VDC Current signal: 4-20mA/0-20mA Resistance signal: 0-30kΩ, support NTC_10K (3950) or PT1000 PT100.
10.	Digital Input (DI)	**8 channel inputs** Active 24VDC input Drain/Source, with isolation. The last 4 channels support pulse counting mode (high speed), 100kHz.
11.	Analog Output (AO) Resolution of 16 bits	**4 channel analogue outputs.** Voltage signal: 0-10VDC Current signal: 4-20mA
12.	RS485 interface	2 channels, baud rate support up to 115200bps, support MODBUS RTU master/slave.
13.	Panel indicator light	POWER: Power indicator, green, ON=with power, OFF=without power.
		Run: Run indicator, which is a two-color indicator. Red, ON = user logic stopped, flash =file transfer in progress, logic suspended. Green, ON = user logic is running.
		SF: Alarm indicator, yellow, ON = Abnormal alarm.
14.	LCD display screen	Display of predefined menu interface configuration parameters. Display of user-defined interfaces can display logical data and design logical data.

1.2.2 Technical specifications

Table 1-2 SM172PS11BDT, SM172PS11BDR, SM172PS11BDM Complete Machine Specifications

Function	Description
Power supply voltage	Typical 24VDC, Range: 24VDC±15%
Power dissipation in W	10W
Operating environment	-20 ~ 60°C, 10% ~ 90% relative humidity, no condensation.
Storage environment	-40~70°C; 10% ~ 90% relative humidity, no condensation.
Protection class	IP20
Wiring terminal	Plug-in screw terminals for one 1.5 mm^2 wire
Memory (unit)	Flash：2MB (execute (1M BIOS + 1M Application)) SRAM：516KB (read-write(256KB BIOS + 256KB Application + 4KB retain variables)) NOR Flash：8MB (File System (2M BIOS + 2M Application + 4KB retain variables + other) PSRAM：8MB (read-write (BIOS))
Analogue Input/Analogue Output Accuracy	Voltage signal: 0-10VDC, error ±1% of full scale. Current signal: 4-20mA/0-20mA, error ±1% of full scale. Resistor signal: supports NTC10K. Allowable error: （-15℃~55℃：±0.2℃. -25℃~15℃ and 55℃-70℃：±0.4℃. -40℃~-25℃ and 70℃~110℃：±1℃）. PT1000, Allowable error：-100℃ ~400℃：0.3℃. -200℃ ~-100℃ and 400℃ ~600℃：0.5℃ . 600℃ ~850℃：1℃ PT100, Allowable error：-100℃~400℃：3℃. -200℃ ~-100℃ and 400℃ ~600℃ ：5℃. 600℃ ~850℃：10℃
Performance	Control logic program cycle time: less than 50ms, operation cycle can be configured.
RTC real time clock	Accuracy (@±25°C): ±2 sec/day Power down hold 14 days @ ±25°C
Install	35mm DIN rail mounting
Housing	Plastic material PC+ABS
Number of IO	8AI, 8DI, 4 AO, 8DO

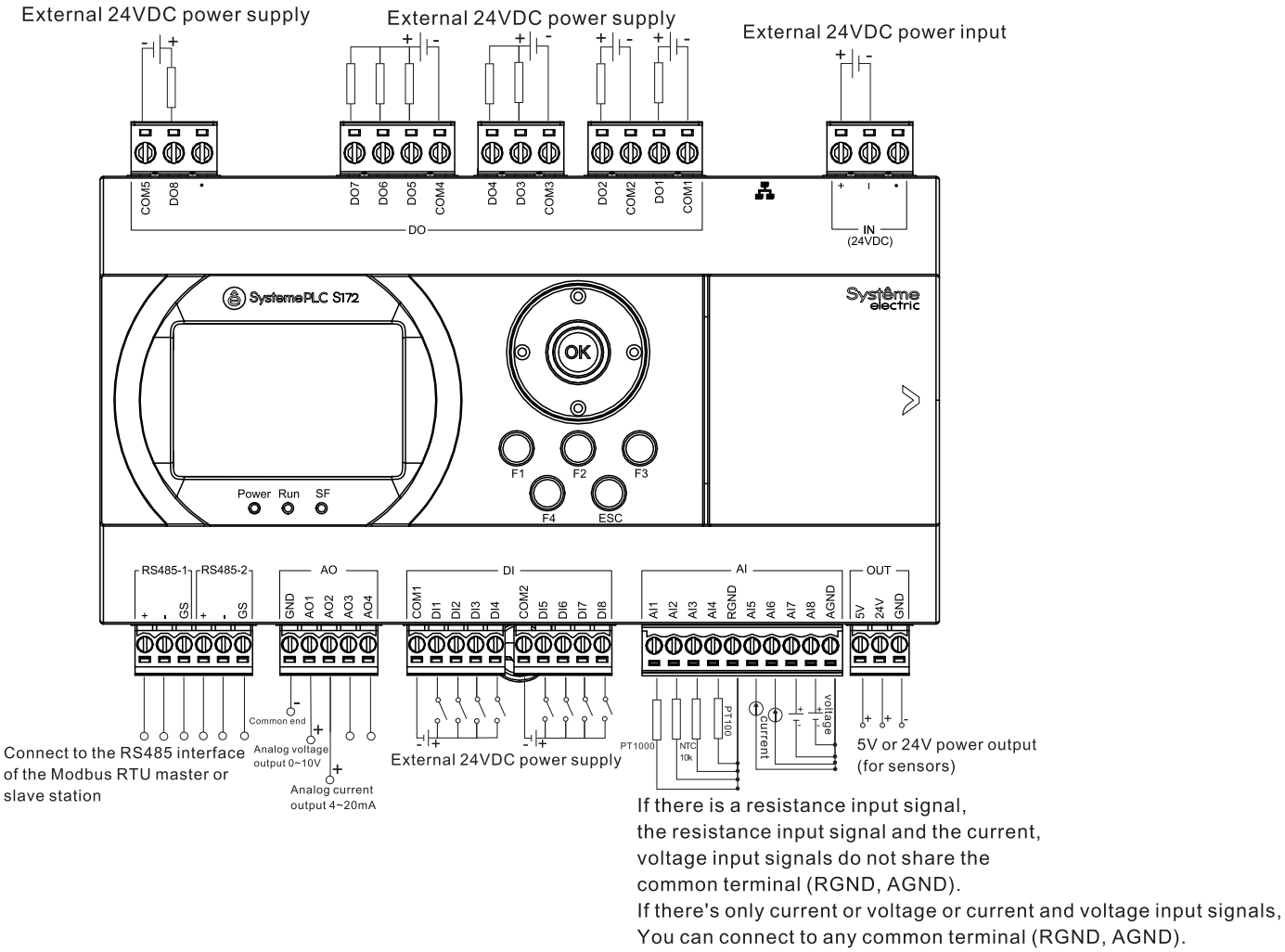
Table 1-3 SM172PS11BDR, SM172PS11BDT, SM172PS11BDM System Functional Specifications

Function	Describe
Maintenance	Equipment maintainability: supports remote and local firmware and control logic updates.
Fault Alarm	The device malfunction light is on, and the module is offline with an alarm message.
Control the operation of logic files	Support parsing control logic files according to control logic parsing specifications.
Support standard instructions	Support standard instructions in IEC61131-3.
Local closed-loop control	The controller should be able to independently execute programs locally without relying on the upper computer for operation.

Data storage	Controller setup parameters, network configuration, power down hold data stored to flash.
Time synchronization	Support configuring NTP/SNTP for automatic time synchronization service.
MODBUS master-slave station function	Support MODBUS TCP/RTU master-slave communication function.
User defined interface	Supports user-defined interfaces that can be used to display and enter data.
Parameter Menu	A menu interface with configuration parameters on the user interface.
Webserver	Support web services, able to access web pages on devices through IP, modify parameters through web pages, and view and modify user-defined data through web pages.

1.2.3 External wiring diagram

The SM172PS11BDT wiring diagram is shown below:

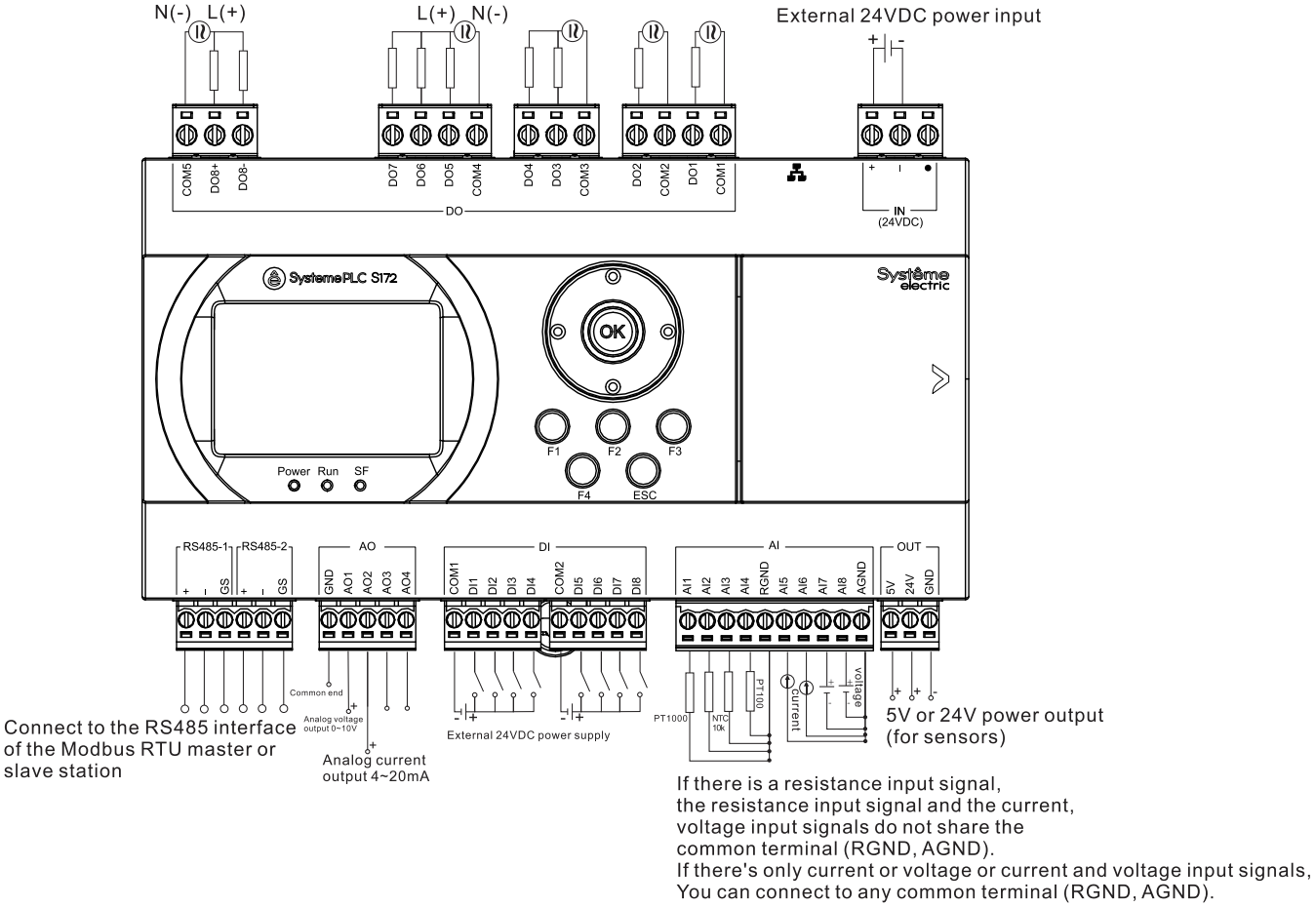


Attention

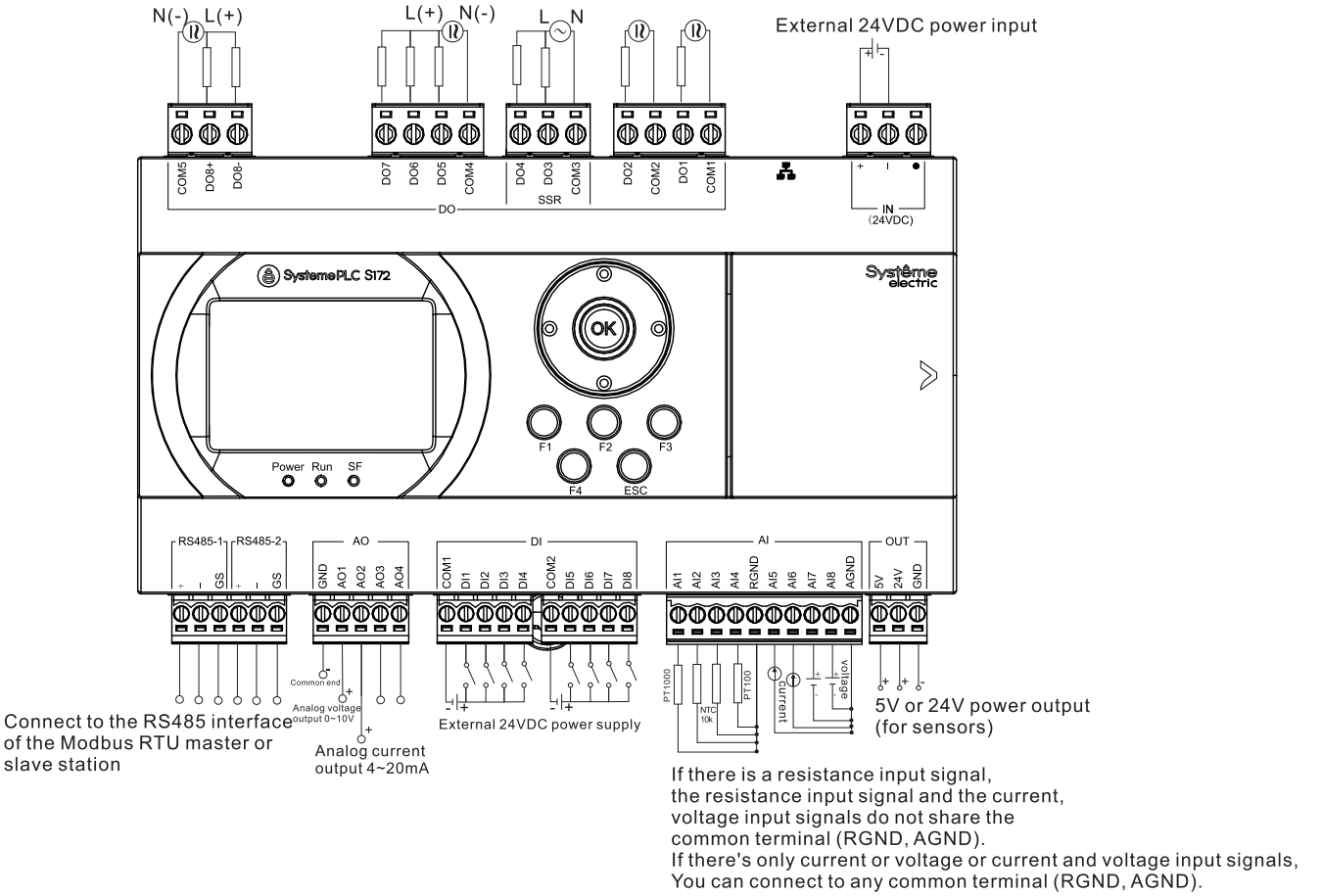
1) Do not install or wire the controller and related equipment under live electrical conditions, as incorrect operation may result in mechanical destruction, personal injury or even death. Before installing and removing any electrical equipment, make sure the power supply to that equipment is disconnected.

2) Please strictly follow the wiring diagram above, otherwise it may cause equipment damage or serious personal injury.

The SM172PS11BDR wiring diagram is shown below:



The SM172PS11BDM wiring diagram is shown below:



1.3 SR2 controller

1.3.1 **Technical specifications**

Table 1-1 SR2 controller complete machine specifications

No.	Function	Description	Model
1	Power supply voltage	Typical 24VDC, Range: 24VDC±15%	ZR2PA11BD ZR2PP11BD2A ZR2PP11BD
		Typical 220VAC, 50Hz, Range: 220VAC±15%	ZR2PB11P7 ZR2PP11P7
2	Power dissipation in W	ZR2PB11P7：4.2W ZR2PA11BD：4W ZR2PP11BD2A：3.2W ZR2PP11P7：6.7W ZR2PP11BD：6.1W	All of SR2
3	Operating environment	-20 ~ 60°C, 10% ~ 90% relative humidity, no condensation.	All of SR2
4	Storage environment	-40~70°C; 10% ~ 90% relative humidity, no condensation.	All of SR2
5	Protection class	IP20	All of SR2
6	Wiring terminal	Plug-in screw terminals for one 1.5 mm^2 wire	All of SR2
7	Memory	Flash：2MB (execute (1M BIOS + 1M Application)) SRAM：516KB (read-write(256KB BIOS + 256KB Application + 4KB retain variables)) NOR Flash：8MB (File System (2M BIOS + 2M Application + 4KB retain variables + other)	All of SR2
8	Communications bus	Extended Bus Protocol	All of SR2
9	Analogue Input/Analogue Output Accuracy	Voltage signal: 0-10VDC, error ±1% of full scale. Current signal: 4-20mA/0-20mA, error ±1% of full scale. Resistor signal: supports NTC10K. Allowable error: （-15℃~55℃：±0.2℃. -25℃~15℃ and 55℃-70℃：±0.4℃. -40℃~-25℃ and 70℃~110℃：±1℃）. PT1000, Allowable error: -100℃ ~400℃: 0.3℃. -200℃ ~-100℃ and 400℃ ~600℃: 0.5℃ . 600℃ ~850℃: 1℃ PT100, Allowable error: -100℃~400℃: 3℃. -200℃ ~-100℃ and 400℃ ~600℃ : 5℃. 600℃ ~850℃: 10℃	All of SR2
10	Performance	Control logic program cycle time: less than 50ms。	All of SR2
11	RTC real time clock	Accuracy (@±25°C): ±2 sec/day Power down hold 14 days @ ±25°C	All of SR2
12	Install	35mm DIN rail mounting	All of SR2
13	Housing	Plastic material PC+ABS	All of SR2

Table 1-2 SR2 Controller System Functional Specifications

No.	Function	Description
1	Maintenance	Equipment maintainability: supports remote and local firmware and control logic updates.
2	Fault Alarm	The device malfunction light is on, and the module is offline with an alarm message.
3	Control the operation of logic files	Support parsing control logic files according to control logic parsing specifications。
4	Instructions	Support standard instructions in IEC61131-3.
5	Local closed-loop control	The controller should be able to independently execute programs locally without relying on the upper computer for operation.
6	Data storage	Controller setup parameters, network configuration, power down hold data stored to flash.
7	Time synchronization	Support configuring NTP/SNTP for automatic time synchronization service.
8	MODBUS master-slave station	Support MODBUS TCP/RTU master-slave communication function.
9	User interface	Supports user-defined interfaces that can be used to display and input data.
10	Parameter Menu	A menu interface with configuration parameters on the user interface.

1.3.2 Component description and external wiring

1.3.2.1 ZR2PB11P7

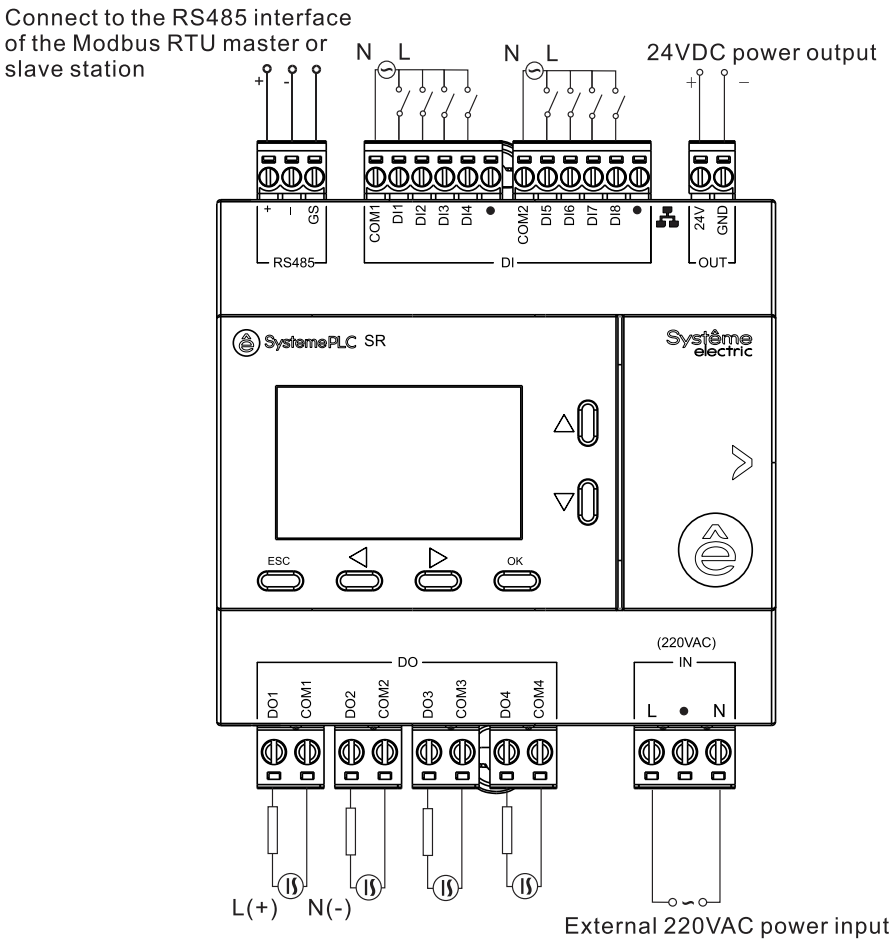


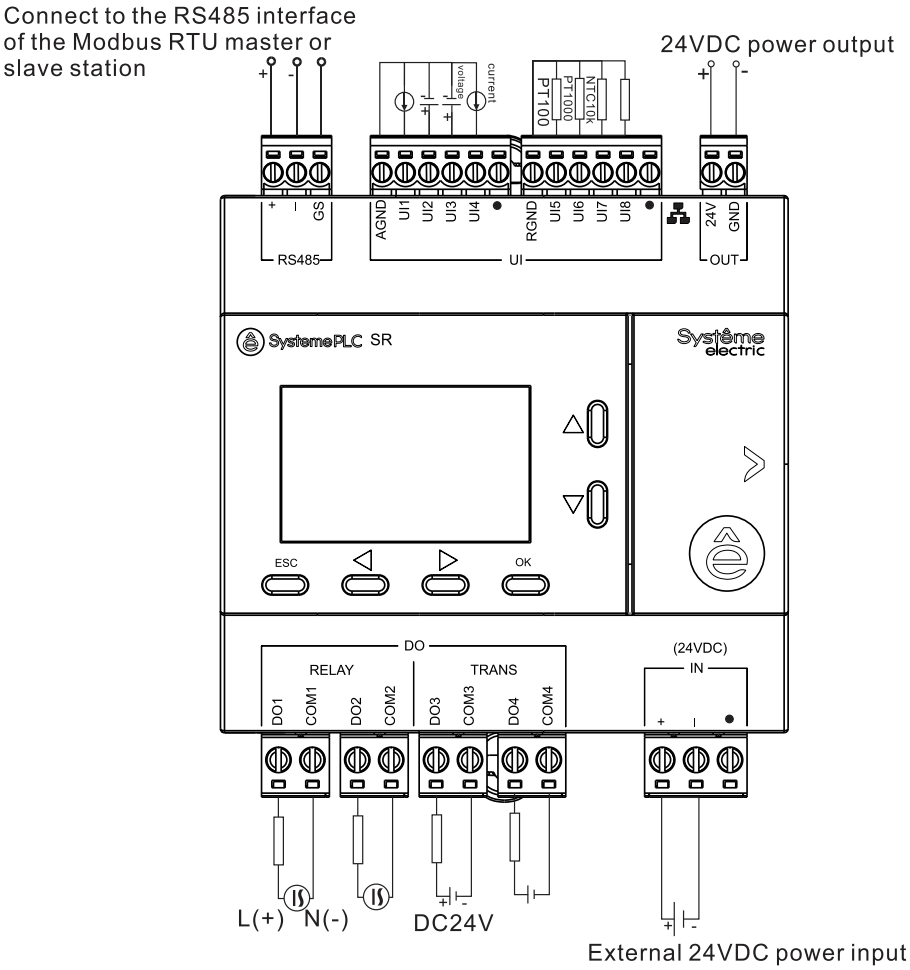
Table 1-3. Interface Description of ZR2PB11P7

No.	Interface	Description
1	RS485 interface	1 channel, baud rate support up to 115200bps, Configure MODBUS RTU master/slave stations through parameter.
2	Digital Input (DI)	8 channel inputs: Active 220VAC input with isolation.
3	Ethernet interface	1 channel, support MODBUS TCP master-slave function, ntp/sntp time synchronisation service, DHCP/bootp automatic IP acquisition function, webserver website service
4	Power supply output	Provide external 24V power supply interfaces with a power of 15W to supply power to active sensors.
5	USB-C interface	Connect the computer and programming software for communication
6	Extension Module Interface	Supports connecting 7 Extension modules, The expansion bus supports extension, with the following constraints: only one expansion bus is allowed to be extended, and the length of the extension cable shall not exceed 1 meter. Otherwise, the modules behind the extension cable cannot be correctly detected. Hot-swapping is not supported for this interface.
7	Power source interface	220VAC power supply
8	Digital output (DO)	4-channel 3A electromagnetic relay output

No.	Interface	Description
9	KEY	6 buttons: up, down, left, right, ok, esc to control the screen menu.
10	LCD display screen	Display of predefined menu interface configuration parameters. Display of user-defined interfaces can display logical data and design logical data.

1.3.2.2 ZR2PA11BD

When the UI serves as the AI



When the UI serves as the DI

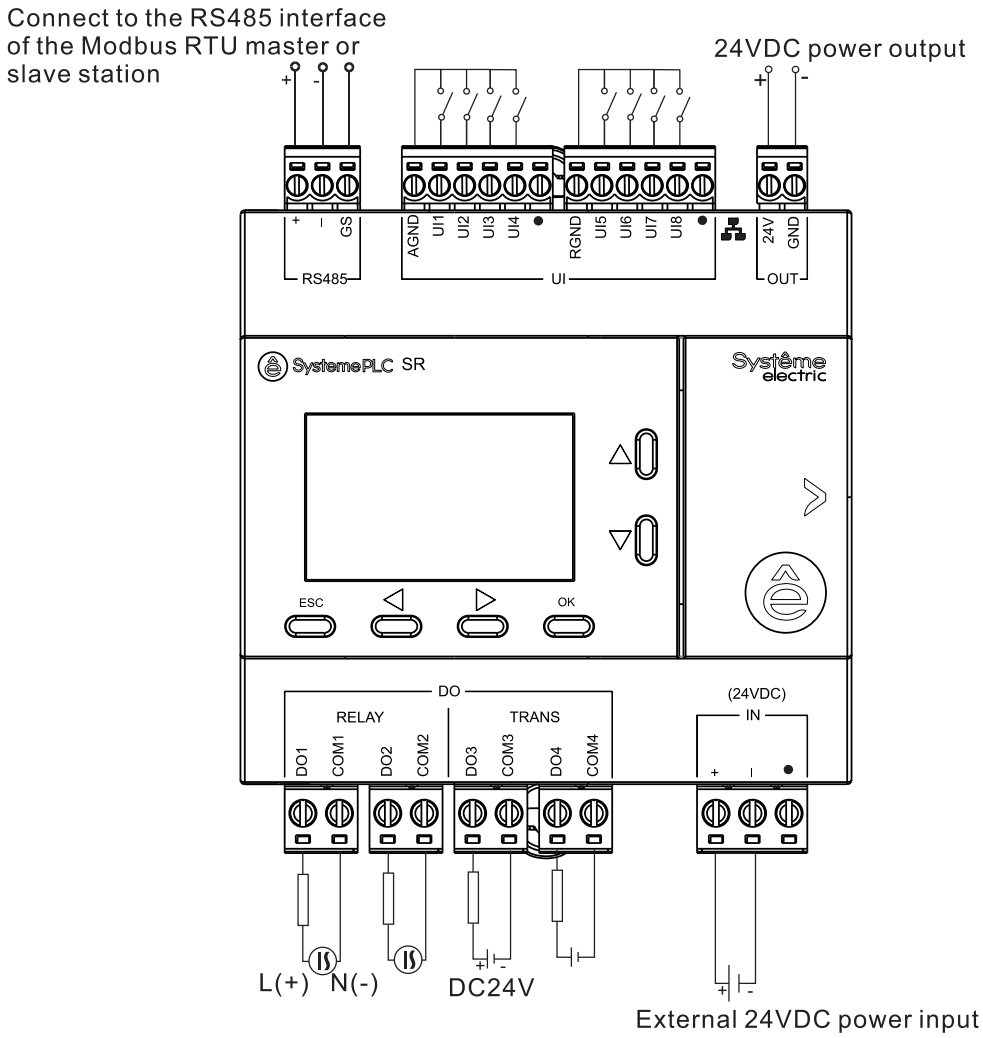


Table 1-4. Interface Description of ZR2PA11BD

No.	Interface	Description
1	RS485 interface	1 channel, baud rate support up to 115200bps, Configure MODBUS RTU master/slave stations through parameter.
2	Universal input (UI)	8-channel universal input (accuracy to be determined) Voltage signal: 0-10VDC. Current signal: 4-20mA/0-20mA. Resistance signal: Supports NTC10K, PT1000, PT100, Resistance value 0-30kΩ. Digital input: Passive dry contact input.
3	Ethernet interface	1 channel, support MODBUS TCP master-slave function, ntp/sntp time synchronisation service, DHCP/bootp automatic IP acquisition function webserver website service.
4	Power supply output	Provide external 24V power supply interfaces with a power of 1W to supply power to active sensors.
5	USB-C interface	Connect the computer and programming software for communication.
6	Extension Module Interface	Supports connecting 7 Extension modules, The expansion bus supports extension, with the following constraints: only one expansion bus is allowed to be extended, and the length of the extension cable shall not exceed 1 meter. Otherwise, the modules behind the extension cable cannot be correctly detected. Hot-swapping is not supported for this interface.

No.	Interface	Description
7	Power source interface	24VDC power supply
8	Digital output (DO)	2-channel 3A electromagnetic relay output 2-channel0.5A transistor 24VDC drain output
9	KEY	6 buttons: up, down, left, right, ok, esc to control the screen menu.
10	LCD display screen	Display of predefined menu interface configuration parameters. Display of user-defined interfaces can display logical data and design logical data.

1.3.2.3 ZR2PP11BD2A

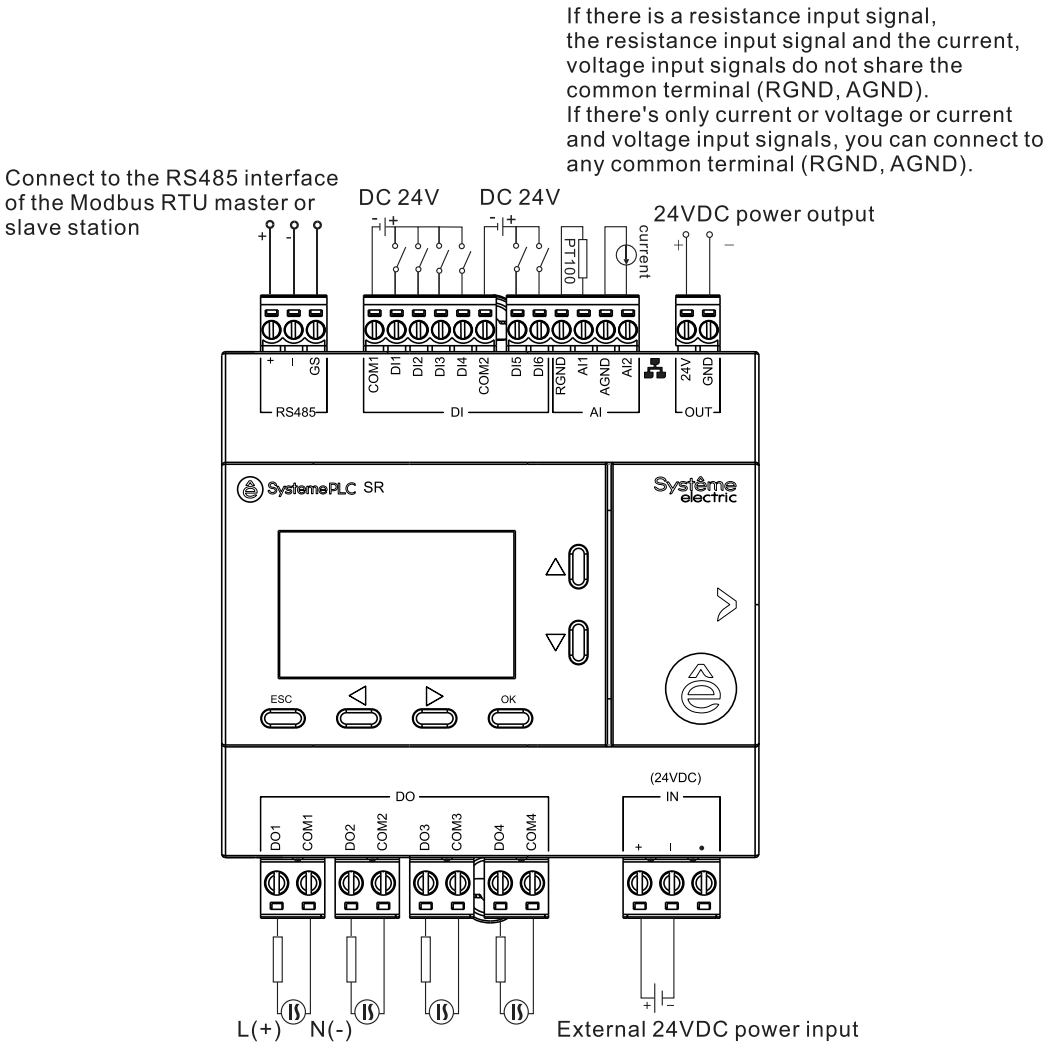
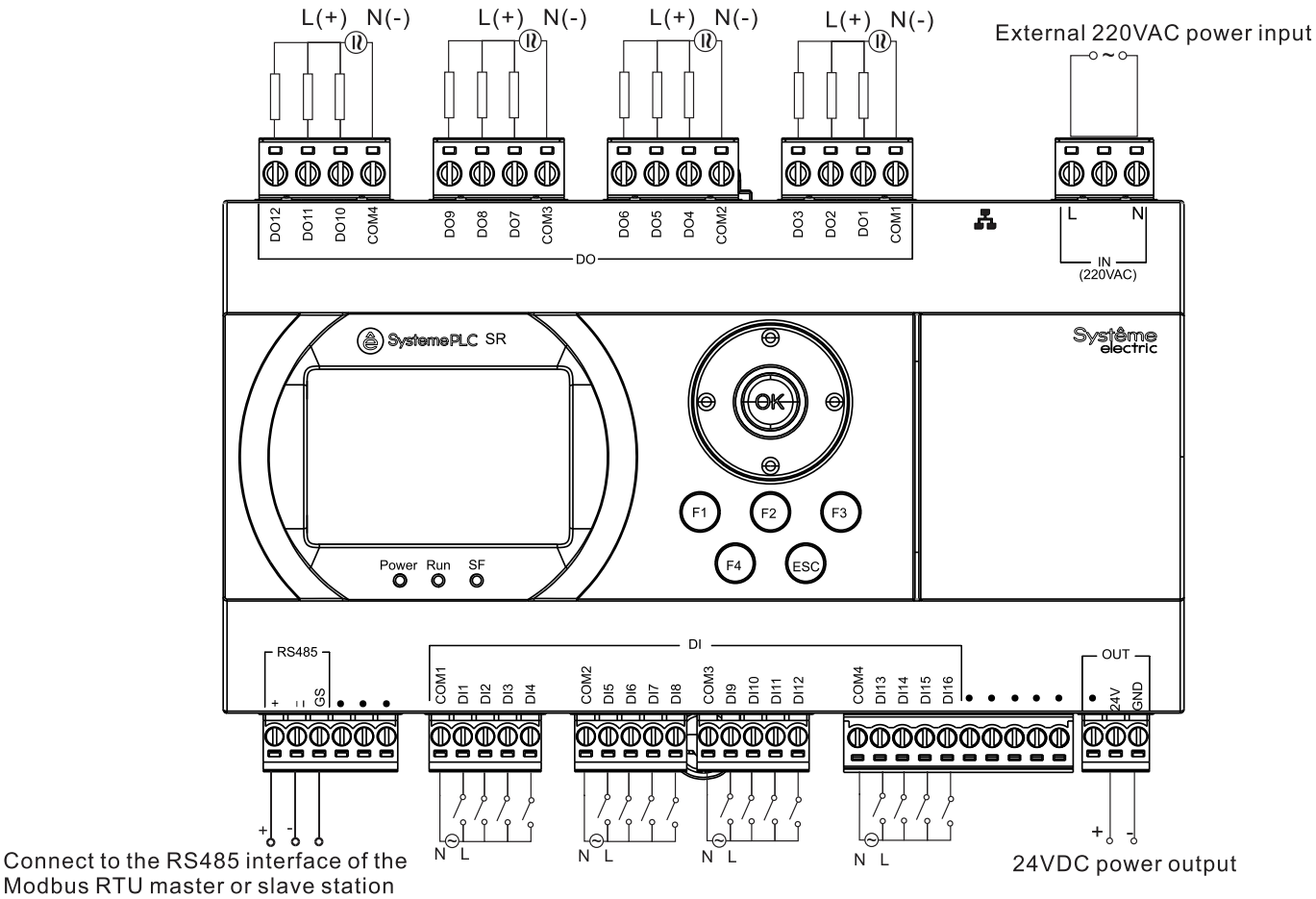


Table 1-5. Interface Description of ZR2PP11BD2A

No.	Interface	Description
1	RS485 interface	1 channel, baud rate support up to 115200bps, Configure MODBUS RTU master/slave stations through parameter.
2	Digital Input (DI)	6 channels, active 24VDC source input with isolation.
3	Analog input (AI)	2-channel analog inputs Voltage signal: 0-10VDC Current signal: 4-20mA/0-20mA Resistance signal: 0-30kΩ, support NTC_10K (3950) or PT1000 PT100.

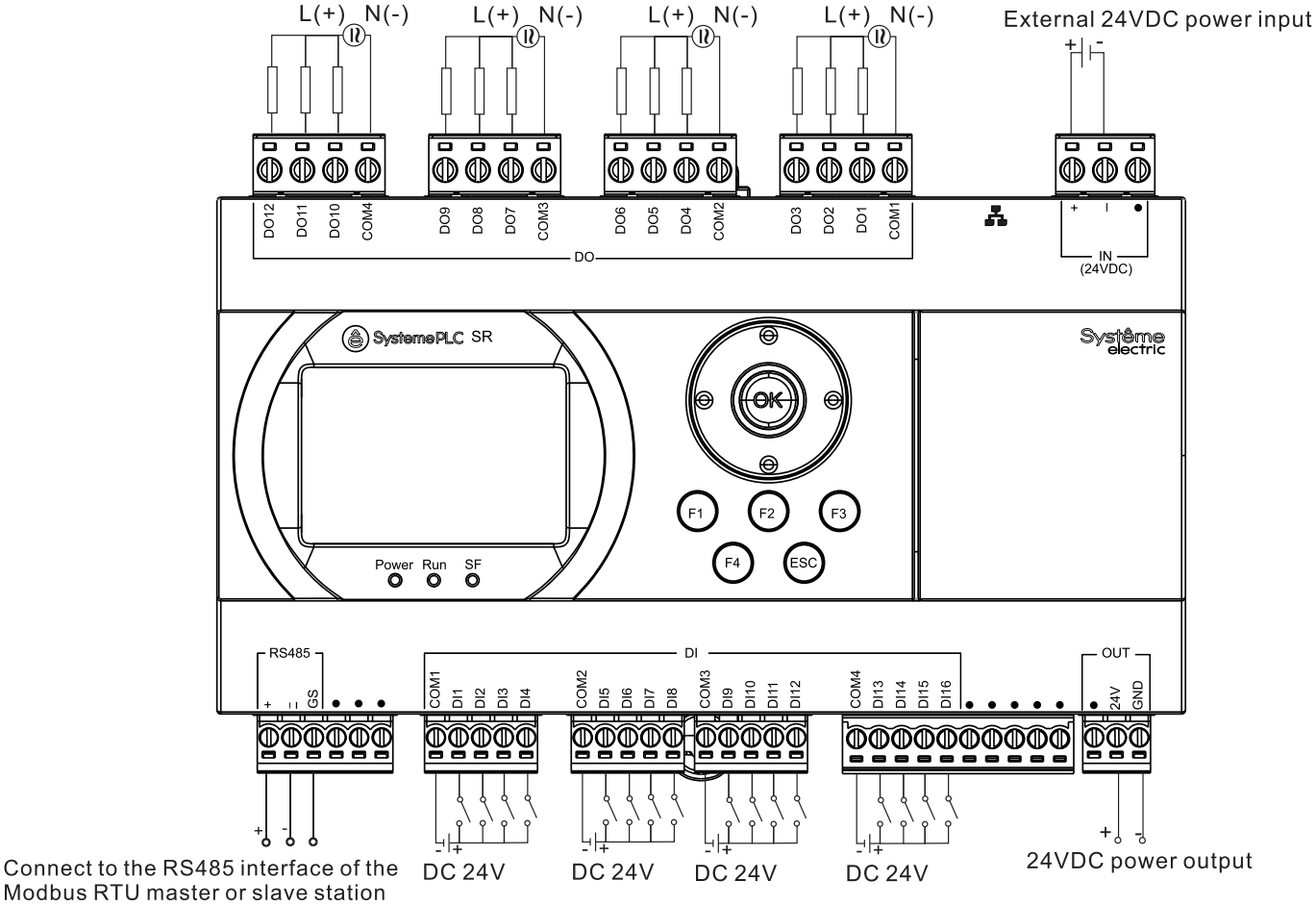
No.	Interface	Description
4	Ethernet interface	1 channel, support MODBUS TCP master-slave function, ntp/sntp time synchronisation service, DHCP/bootp automatic IP acquisition function, webserver website service.
5	Power supply output	Provide external 24V power supply interfaces with a power of 1W to supply power to active sensors.
6	USB-C interface	Connect the computer and programming software for communication
7	Extension Module Interface	Supports connecting 7 Extension modules, The expansion bus supports extension, with the following constraints: only one expansion bus is allowed to be extended, and the length of the extension cable shall not exceed 1 meter. Otherwise, the modules behind the extension cable cannot be correctly detected. Hot-swapping is not supported for this interface.
8	Power source interface	24VDC
9	Digital output (DO)	4-channel 3A electromagnetic relay output
10	KEY	6 buttons: up, down, left, right, ok, esc to control the screen menu.
11	LCD display screen	Display of predefined menu interface configuration parameters. Display of user-defined interfaces can display logical data and design logical data.

1.3.2.4 ZR2PP11P7



No.	Interface	Description
1	Digital output （DO）	12 channel outputs (3A electromagnetic relay output).
2	Ethernet interface	1 channel, support MODBUS TCP master-slave function, ntp/sntp time synchronisation service, DHCP/bootp automatic IP acquisition function, webserver website service.
3	Power source interface	220 VAC power supply input
4	Type C interface	For connecting to a computer and communicating with programming software.
5	Type A	Used to read and write USB flash drive files.
6	Expansion Module Interface	Supports connecting 7 Extension modules, The expansion bus supports extension, with the following constraints: only one expansion bus is allowed to be extended, and the length of the extension cable shall not exceed 1 meter. Otherwise, the modules behind the extension cable cannot be correctly detected. Hot-swapping is not supported for this interface.
7	Power supply output	Provide external 24V power supply interfaces with a power of 15W to supply power to active sensors.
8	Digital Input (DI)	16 channel inputs: :Active 220VAC input with isolation.
9	KEY	10 buttons, including 6 buttons for up, down, left, right, ok, and esc to control the screen menu, and F1, F2, F3, and F4 user-defined function codes.
10	RS485 interface	1 channels, baud rate support up to 115200bps, support MODBUS RTU master/slave.
10	Panel indicator light	**POWER:** Power indicator, green, ON=with power, OFF=without power.
		Run: Run indicator, which is a two-color indicator. Red, ON = user logic stopped, flash =file transfer in progress, logic suspended. Green, ON = user logic is running.
		SF: Alarm indicator, yellow, ON = Abnormal alarm.
11	LCD display screen	Display of predefined menu interface configuration parameters. Display of user-defined interfaces can display logical data and design logical data.

1.3.2.5 ZR2PP11BD



No.	Interface	Description
1	Digital output （DO）	12 channel outputs (3A electromagnetic relay output).
2	Ethernet interface	1 channel, support MODBUS TCP master-slave function, ntp/sntp time synchronisation service, DHCP/bootp automatic IP acquisition function, webserver website service.
3	Power source interface	24 DC power supply input
4	Type C interface	For connecting to a computer and communicating with programming software.
5	Type A	Used to read and write USB flash drive files.
6	Expansion Module Interface	Supports connecting 7 Extension modules, The expansion bus supports extension, with the following constraints: only one expansion bus is allowed to be extended, and the length of the extension cable shall not exceed 1 meter. Otherwise, the modules behind the extension cable cannot be correctly detected. Hot-swapping is not supported for this interface.
7	Power supply output	Provide external 5V and 24V power supply interfaces with a power of 1W to supply power to active sensors.
8	Digital Input (DI)	16 channel inputs: Active 24VDC input. Source, with isolation.
9	KEY	10 buttons, including 6 buttons for up, down, left, right, ok, and esc to control the

		screen menu, and F1, F2, F3, and F4 user-defined function codes.
10	RS485 interface	1 channels, baud rate support up to 115200bps, support MODBUS RTU master/slave.
11	Panel indicator light	**POWER:** Power indicator, green, ON=with power, OFF=without power.
		Run: Run indicator, which is a two-color indicator. Red, ON = user logic stopped, flash =file transfer in progress, logic suspended. Green, ON = user logic is running.
		SF: Alarm indicator, yellow, ON = Abnormal alarm.
12	LCD display screen	Display of predefined menu interface configuration parameters. Display of user-defined interfaces can display logical data and design logical data.

1.4 SR1 controller

1.4.1 **Technical specifications**

Table1-6 ZR1PB00P7, ZR1PA00P7, ZR1PB00BD, ZR1PA00BD, ZR1PP00BD2A, ZR1PP00BD4A technical specifications

No.	Function	Description	Model
1	Power supply voltage	Typical 24VDC, Range: 24VDC±15%	ZR1PA00BD ZR1PB00BD ZR1PP00BD2A ZR1PP00BD4A
		Typical 220VAC , 50Hz, Range: 220VAC±15%	ZR1PB00P7 ZR1PA00P7
2	Power dissipation in W	ZR1PA00BD：5.3W ZR1PA00P7：6.3W ZR1PB00BD：3W ZR1PB00P7：4W ZR1PP00BD2A：3W ZR1PP00BD4A：6W	All of SR1
3	Operating environment	-20 ~ 60°C, 10% ~ 90% relative humidity, no condensation	All of SR1
4	Storage environment	-40~70°C; 10% ~ 90% relative humidity, no condensation.	All of SR1
5	Protection class	IP20	All of SR1
6	Wiring terminal	Plug-in screw terminals for one 1.5 mm^2 wire	All of SR1
7	Memory	Flash：1MB (execute (512KB BIOS + 512KB Application)) SRAM：192KB (read-write(96KB BIOS + 92KB Application + 4KB retain variables)) NOR Flash：4MB (File System (1M BIOS + 1M Application + 4KB retain variables + other)	All of SR1
8	Performance	Control logic program cycle time: less than 50ms.	All of SR1

12	RTC real time clock	Accuracy (@±25°C): ±2 sec/day Power down hold 14 days @ ±25°C	All of SR1
13	Install	35mm DIN rail mounting	All of SR1
14	Housing	Plastic material PC+ABS.	All of SR1
15	Analogue Input/Analogue Output Accuracy	Voltage signal: 0-10VDC, error ±1% of full scale. Current signal: 4-20mA/0-20mA, error ±1% of full scale. Resistor signal: supports NTC10K. Allowable error: （-15 ℃~55℃:±0.2℃. -25℃~15℃and 55℃-70℃: ±0.4℃. -40℃~-25℃ and 70℃~110℃: ±1℃）. PT1000, Allowable error: -100℃ ~400℃: 0.3℃. -200℃ ~-100℃ and 400℃ ~600℃: 0.5℃ . 600℃ ~850℃: 1℃ PT100, Allowable error: -100℃~400℃: 3℃. -200℃ ~-100℃ and 400℃ ~600℃ : 5℃. 600℃ ~850℃ : 10℃	ZR1PP00BD2A ZR1PP00BD4A

Table 1-7 ZR1PB00P7, ZR1PA00P7, ZR1PB00BD, ZR1PA00BD, ZR1PP00BD2A, ZR1PP00BD4A System Functional Specifications.

No.	Function	Description
1	Maintenance	Equipment maintainability: supports remote and local firmware and control logic updates.
2	Fault Alarm	The device malfunction light is on, and the module is offline with an alarm message.
3	Control the operation of logic files	Support parsing control logic files according to control logic parsing specifications.
4	Support standard instructions	Support standard instructions in IEC61131-3.
5	Local closed-loop control	The controller should be able to independently execute programs locally without relying on the upper computer for operation.
6	Data storage	Controller setup parameters, network configuration, power down hold data stored to flash.
7	MODBUS master-slave function	Support MODBUS RTU master-slave communication function.

1.4.2 Component description and external wiring

1.4.2.1 ZR1PA00P7

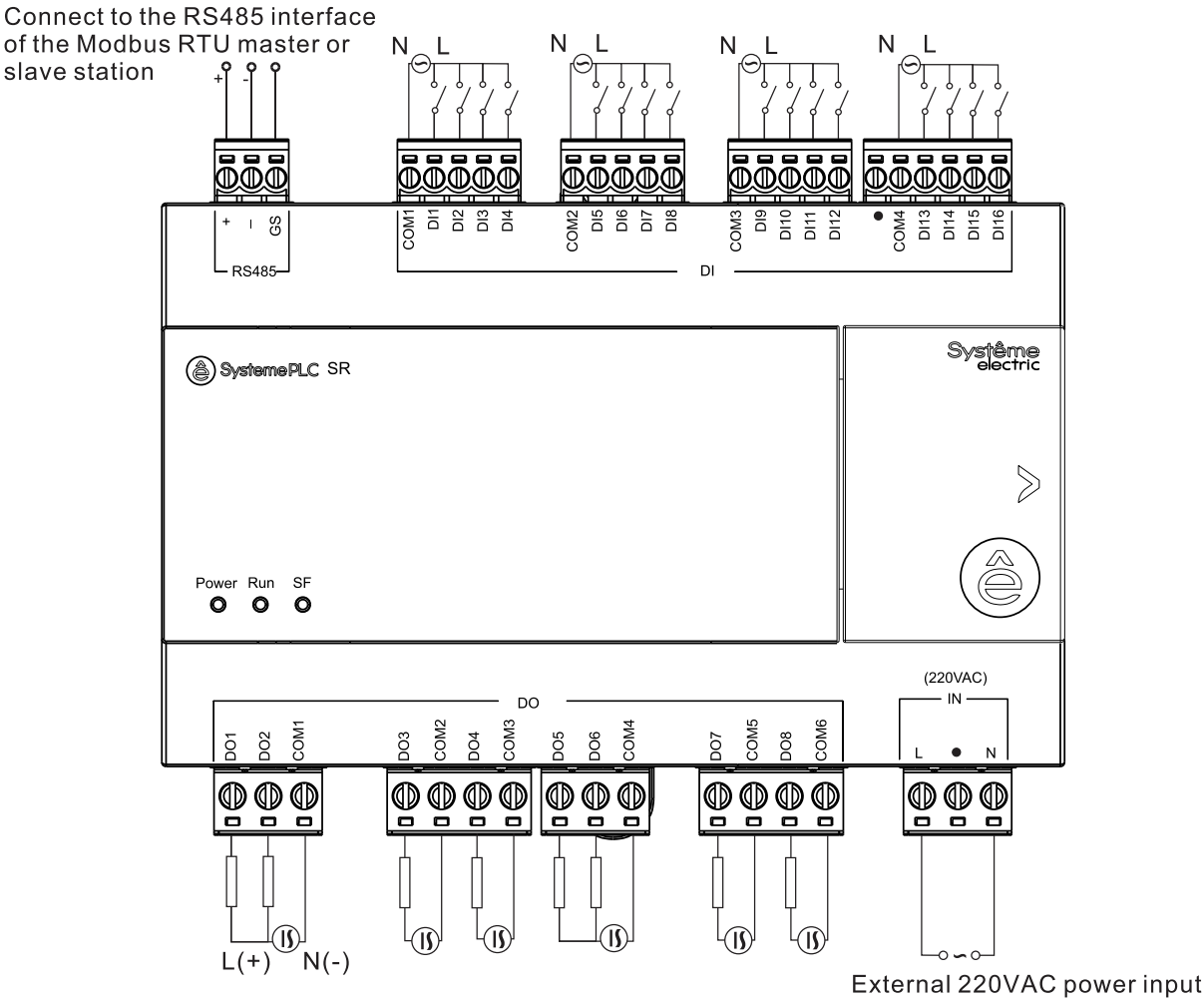


Table 1-8 ZR1PA00P7 Component Description

No.	Interface	Description
1	RS485 interface	1 channel, baud rate support up to 115200bps, Configure MODBUS RTU master/slave stations through parameter.
2	Digital Input (DI)	16 channel inputs: Active 220VAC input with isolation.
3	RUN-STOP switch	Control program to stop/run
4	USB-C interface	Connect the computer and programming software for communication
5	Power interface	220VAC power supply
6	Digital output (DO)	8-channel 3A electromagnetic relay output
7	Indicator light	**POWER:** Power indicator light, green, ON=with power, OFF= without power.
		Run: Run indicator light, which is a two-color indicator. Red, ON = user logic stopped, flash =file transfer in progress, logic suspended. Green, ON = user logic is running.
		SF: Alarm indicator light, yellow, ON=abnormal alarm.

1.4.2.2 ZR1PB00P7

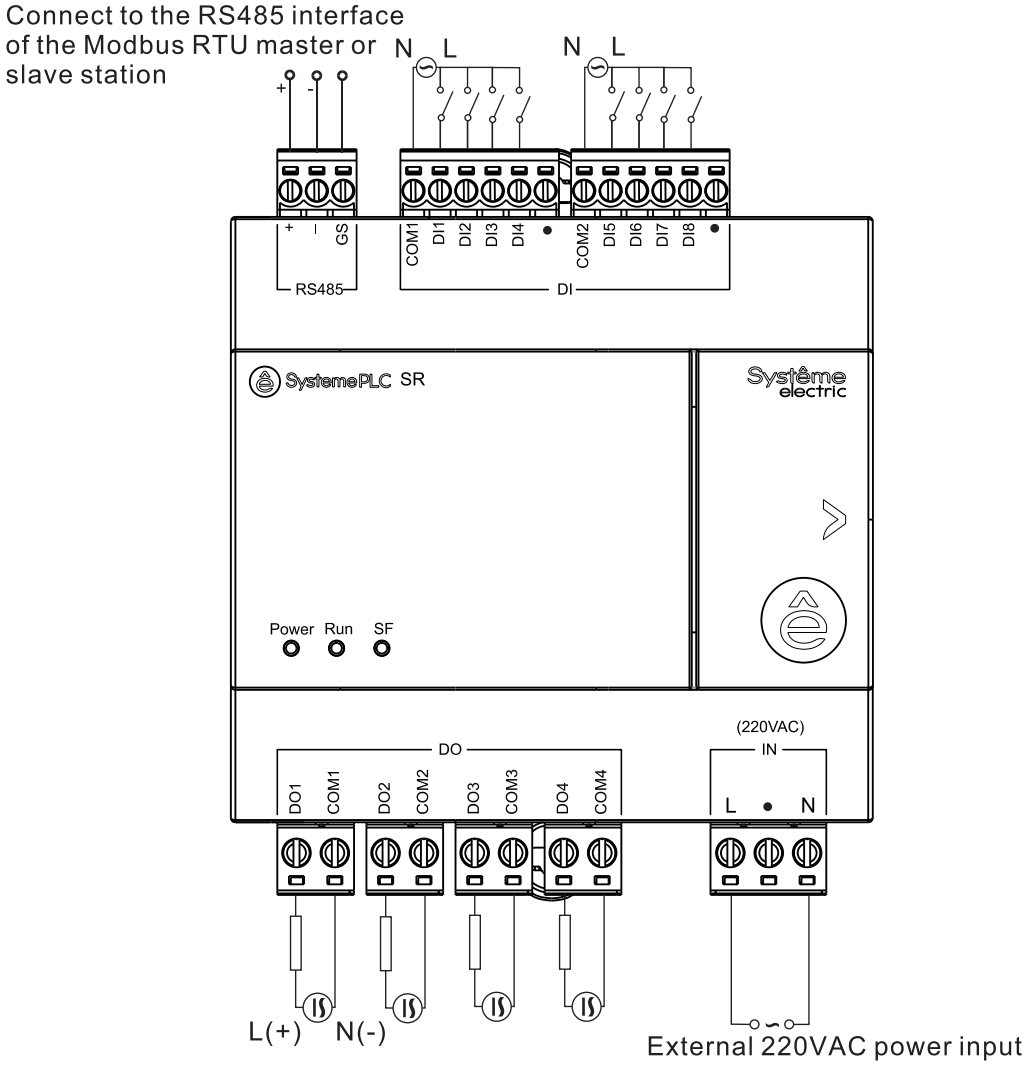


Table 1-9 ZR1PB00P7 Component Description

No.	Interface	Description
1	RS485 interface	1 channel, baud rate support up to 115200bps, Configure MODBUS RTU master/slave stations through parameter.
2	Digital Input (DI)	8 channel inputs: Active 220VAC input with isolation.
3	RUN-STOP switch	Control program to stop/run
4	USB-C interface	Connect the computer and programming software for communication
5	Power interface	220VAC power supply
6	Digital output (DO)	4-channel 3A electromagnetic relay output
7	Indicator light	**POWER:** Power indicator light, green, ON=with power, OFF= without power.
		Run: Run indicator light, which is a two-color indicator. Red, ON = user logic stopped, flash =file transfer in progress, logic suspended. Green, ON = user logic is running.
		SF: Alarm indicator light, yellow, ON=abnormal alarm.

1.4.2.3 ZR1PB00BD

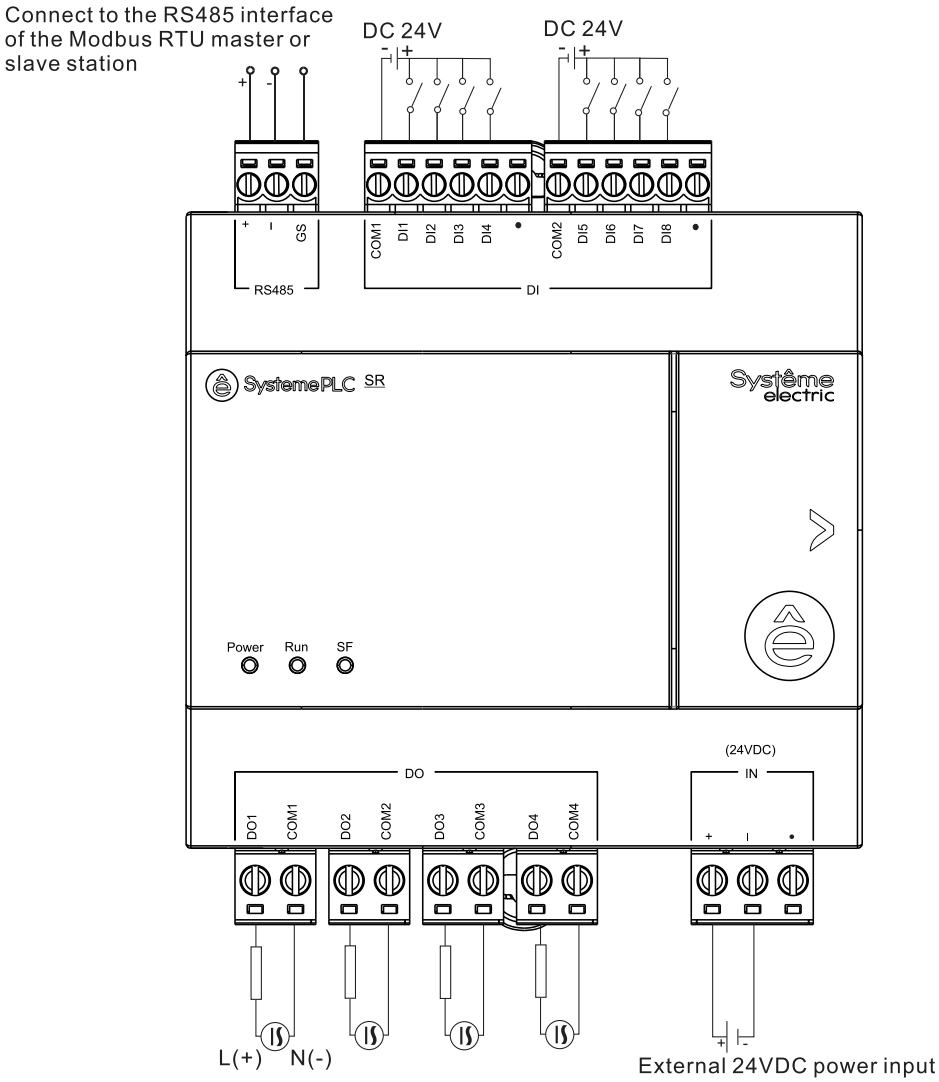


Table 1-10 ZR1PB00BD Component Description

No.	Interface	Description
1	RS485 interface	1 channel, baud rate support up to 115200bps, Configure MODBUS RTU master/slave stations through parameter.
2	Digital Input (DI)	8 channels, active 24VDC source input, with isolation.
3	RUN-STOP switch	Control program to stop/run
4	USB-C interface	Connect the computer and programming software for communication
5	Power source interface	24VDC power supply
6	Digital output (DO)	4-channel 3A electromagnetic relay output
7	Indicator light	**POWER:** Power indicator light, green, ON=with power, OFF= without power.
		Run: Run indicator light, which is a two-color indicator. Red, ON = user logic stopped, flash =file transfer in progress, logic suspended. Green, ON = user logic is running.
		SF: Alarm indicator light, yellow, ON=abnormal alarm.

1.4.2.4 ZR1PA00BD

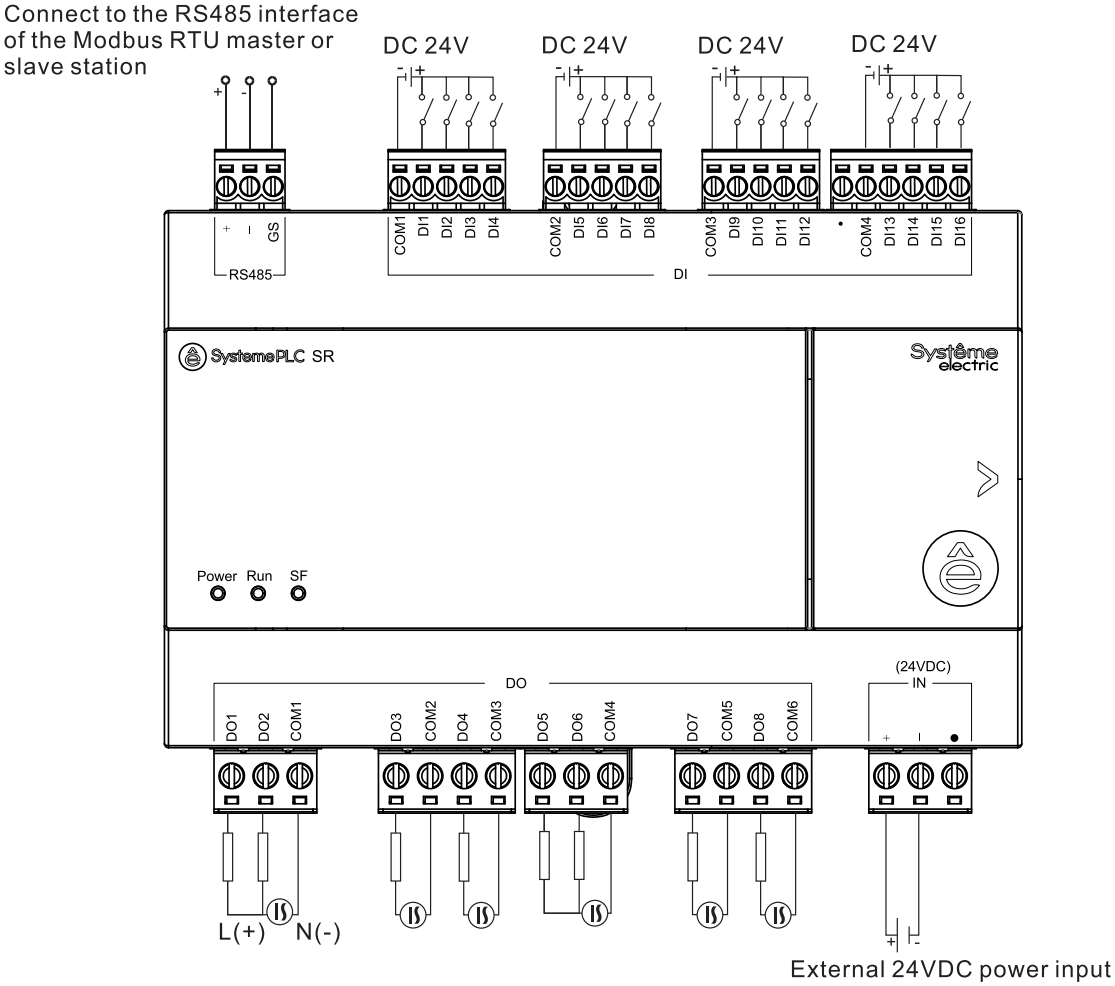


Table 1-11 ZR1PA00BD Component Description

No.	Interface	Description
1	RS485 interface	1 channel, baud rate support up to 115200bps, Configure MODBUS RTU master/slave stations through parameter.
2	digital input(DI)	16 channel DI, active 24VDC source input, with isolation
3	RUN-STOP switch	Control program to stop/run
4	USB-C interface	Connect the computer and programming software for communication
5	Power source interface	24VDC power supply
6	Digital Output (DO)	8-channel 3A electromagnetic relay output
7	Indicator light	**POWER:** Power indicator light, green, ON=with power, OFF= without power.
		Run: Run indicator light, which is a two-color indicator. Red, ON = user logic stopped, flash =file transfer in progress, logic suspended. Green, ON = user logic is running.
		SF: Alarm indicator light, yellow, ON=abnormal alarm.

1.4.2.5 ZR1PP00BD2A

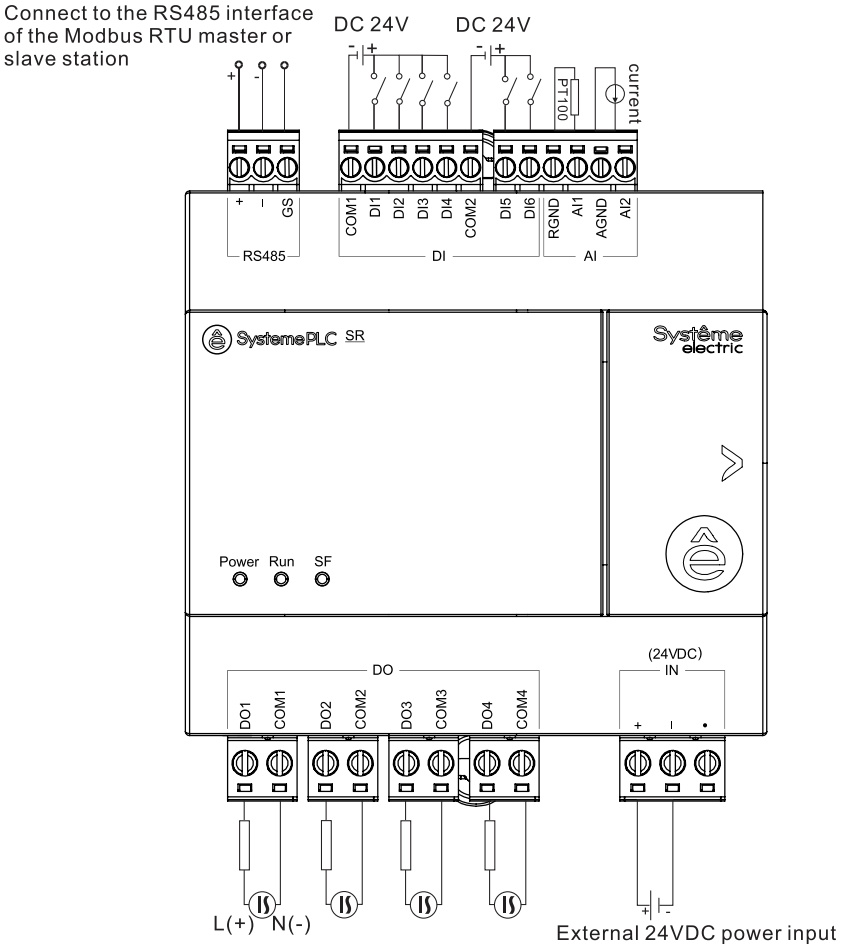


Table 1-12 ZR1PP00BD2A Component Description

No.	Interface	Description
1	RS485 interface	1 channel, baud rate support up to 115200bps, Configure MODBUS RTU master/slave stations through parameter.
2	Digital Input (DI)	6-channel active 24VDC source input with isolation
3	Analog input (AI)	2-channel analog input Voltage signal: 0-10VDC Current signal: 4-20mA/0-20mA Resistance signal: 0-30kΩ, support NTC_10K (3950) or PT1000 PT100.
4	RUN-STOP switch	Control program to stop/run
5	USB-C interface	Connect the computer and programming software for communication
6	Power source interface	24VDC power supply
7	Digital output (DO)	4-channel 3A electromagnetic relay output
8	Indicator light	**POWER:** Power indicator light, green, ON=with power, OFF= without power.
		Run: Run indicator light, which is a two-color indicator. Red, ON = user logic stopped, flash =file transfer in progress, logic suspended. Green, ON = user logic is running.

		SF: Alarm indicator light, yellow, ON=abnormal alarm.

1.4.2.6 ZR1PP00BD4A

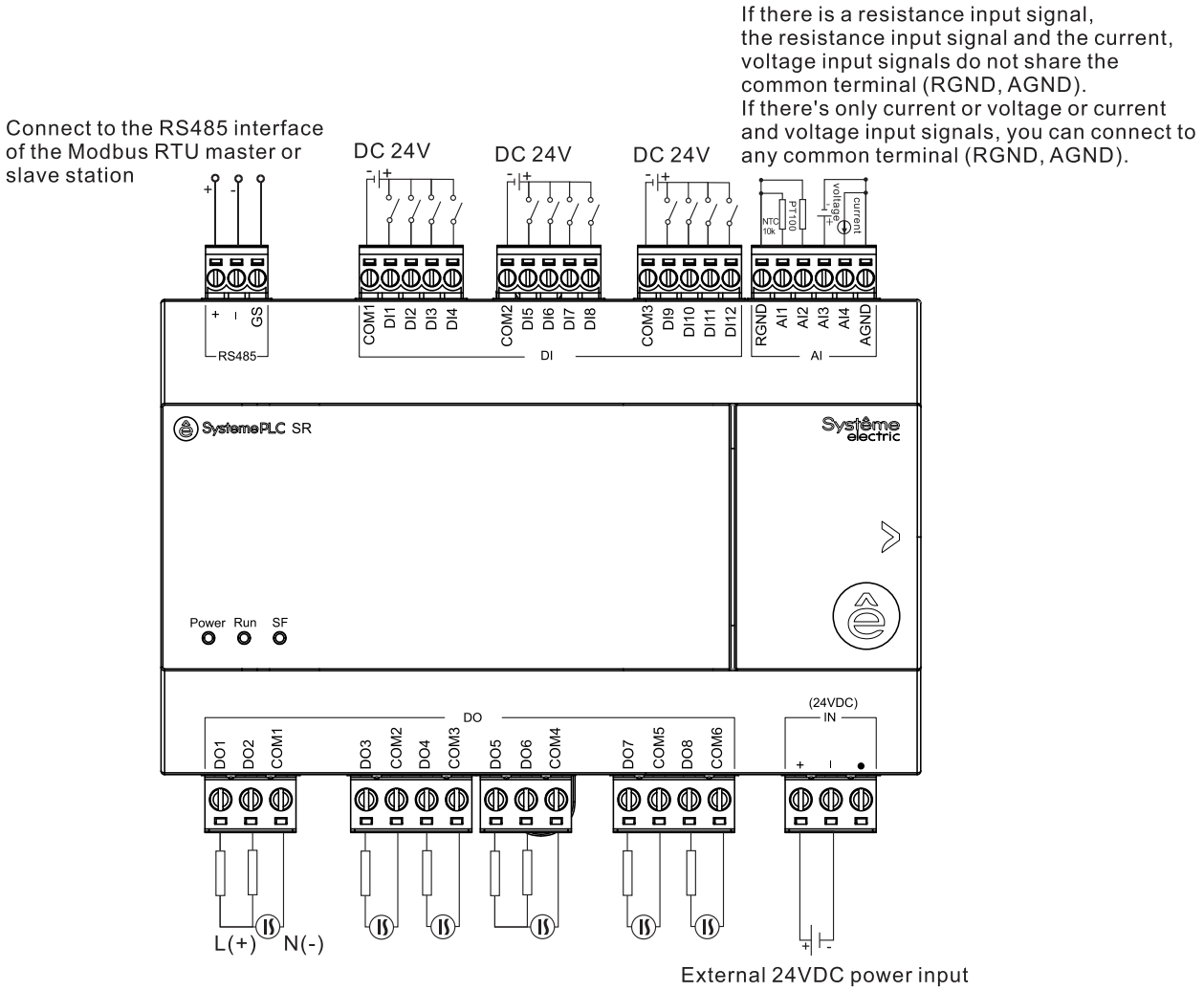


Table 1-13 ZR1PP00BD4A Component Description

No.	Interface	Description
1	RS485 interface	1 channel, baud rate support up to 115200bps, Configure MODBUS RTU master/slave stations through parameter.
2	Digital Input (DI)	12-channel active 24VDC source input with isolation
3	Analog input (AI)	4-channel analog input Voltage signal: 0-10VDC Current signal: 4-20mA/0-20mA Resistance signal: 0-30kΩ, support NTC_10K (3950) or PT1000 PT100.
4	RUN-STOP switch	Control program to stop/run
5	USB-C interface	Connect the computer and programming software for communication
6	Power source interface	24VDC power supply
7	Digital Output (DO)	8-channel 3A electromagnetic relay output

8	Panel indicator light	**POWER:** Power indicator light, green, ON=with power, OFF= without power.
		Run: Run indicator light, which is a two-color indicator. Red, ON = user logic stopped, flash =file transfer in progress, logic suspended. Green, ON = user logic is running.
		SF: Alarm indicator light, yellow, ON=abnormal alarm.

1.5 Interface definitions

Table 1-4 Ethernet Interface Signal Definitions

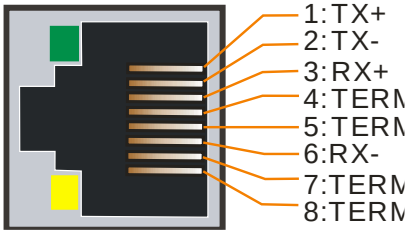
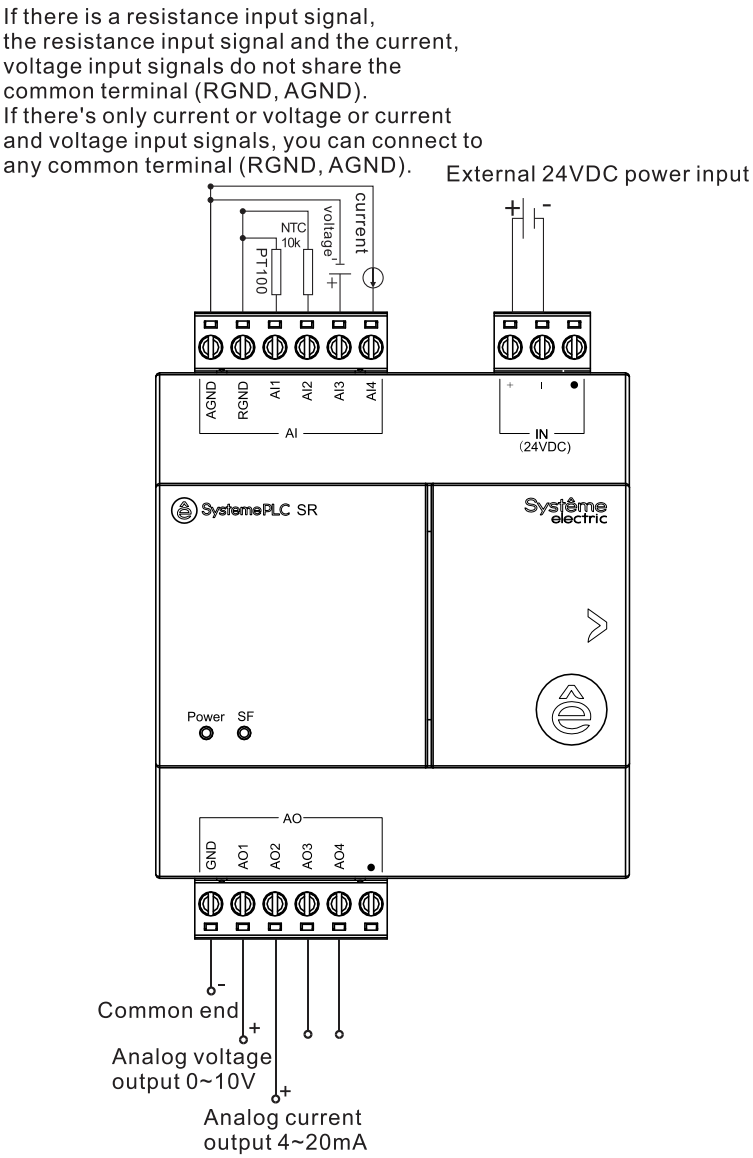
RJ45 network port	No.	signal	Signal Definition
	1	TX+	Data sent +
	2	TX-	Data sent -
	3	RX+	Data reception+
	4	TERM	--
	5	TERM	--
	6	RX-	Data reception -
	7	TERM	--
	8	TERM	--
	Connector Shells	PE	Chassis grounding
Green Indicator	As an indication of EtherNET connection status		
Yellow Indicator	As EtherNET speed status indication		

Table 1-5 RS485 Terminal Definitions

RS485 port	Signal	Signal Definition
	+	RS485 signal A
	-	RS485 signal B
	GS	RS485 signal ground

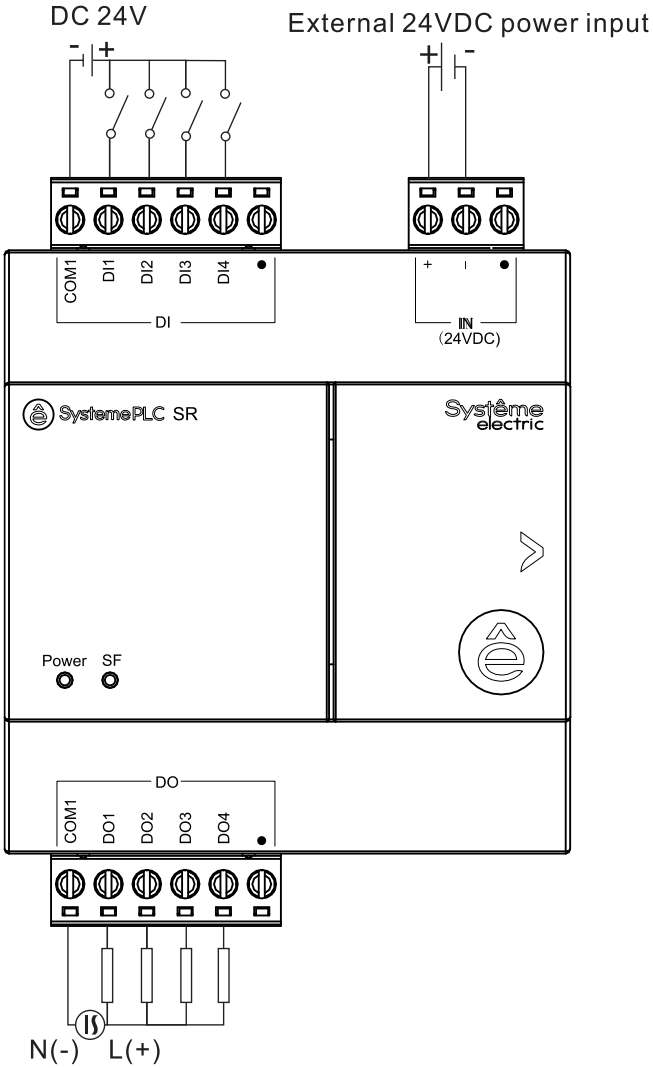
1.6 Expansion modules

1.6.1 **SM172EAM0800**



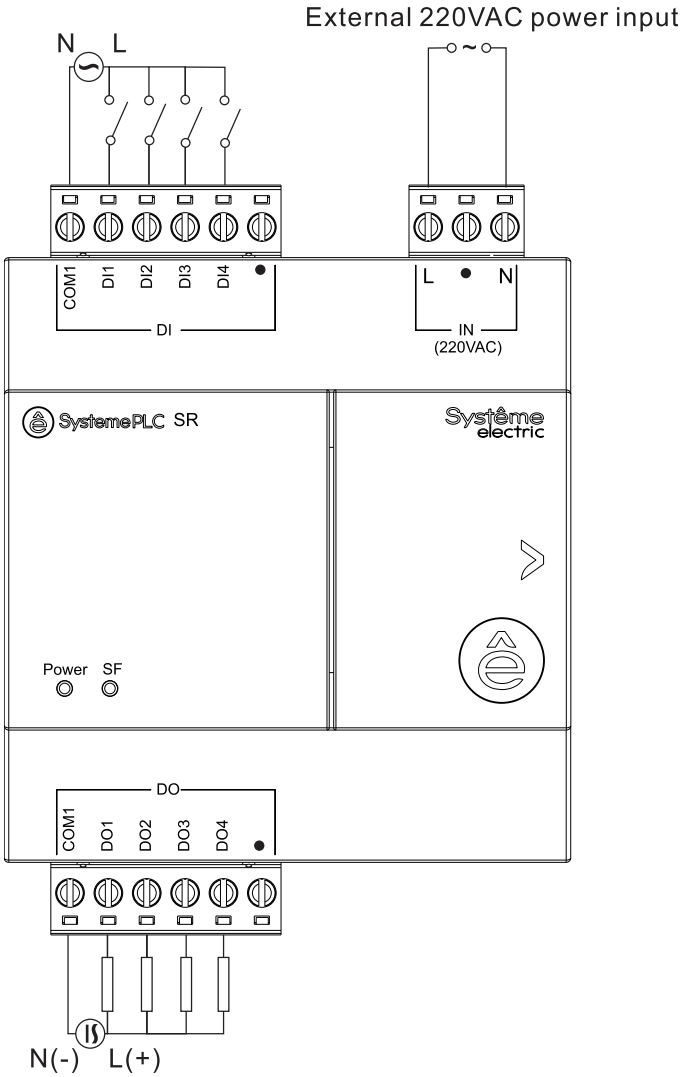
No.	Interface	Description
1	Analog input (AI) Resolution of 16 bits	**4 channel analogue inputs** Voltage signal: 0-10VDC Current signal: 4-20mA/0-20mA Resistance signal: 0-30kΩ, support NTC_10K (3950) or PT1000 PT100.
2	power interface	24V DC power input
3	Expansion module interface	Expansion module interface, Hot-swapping is not supported.
4	Analog Output (AO)	**4 channel analogue output** Voltage signal: 0-10VDC Current signal: 4-20mA
5	Panel indicator light	**POWER**: Bus Power Indicator, green, ON = Module bus power is turned on; OFF = Module bus power is turned off. **SF:** Alarm indicator, yellow, ON = Abnormal alarm.
6	Power dissipation in W	2W

1.6.2 **SM172EDM0800**



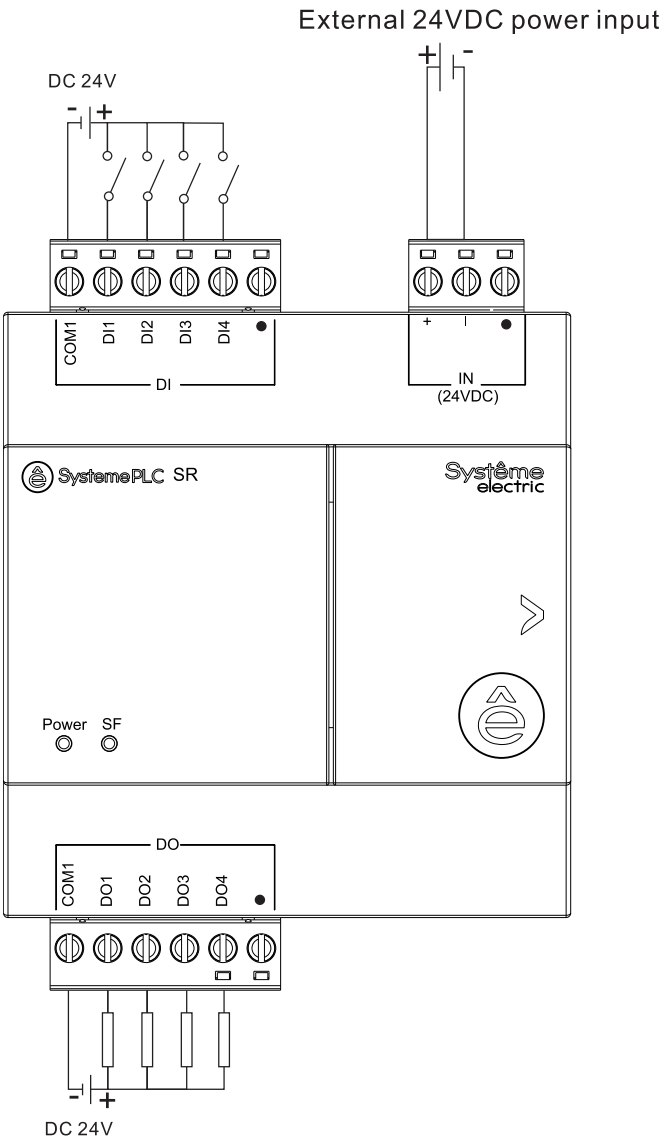
No.	Interface	Description
1	Digital Input (DI)	4 channel active 24V inputs, source type connection, isolated
2	power interface	24V DC power input
3	Expansion module interface	Expansion module interface, Hot-swapping is not supported.
4	Digital Output (DO)	4-channel 3A electromagnetic relay output.
5	Panel indicator light	**POWER**: Bus Power Indicator, green, ON = Module bus power is turned on; OFF = Module bus power is turned off. **SF:** Alarm indicator, yellow, ON = Abnormal alarm.
6	Power dissipation in W	1W

1.6.3 **SM172EDM0800P7**



No.	Interface	Description
1	Digital Input (DI)	4 channel inputs: Active 220VAC input with isolation.
2	power interface	220VAC power supply
3	Expansion module interface	Expansion module interface, Hot-swapping is not supported.
4	Digital Output (DO)	4-channel 3A electromagnetic relay output.
5	Panel indicator light	**POWER**: Bus Power Indicator, green, ON = Module bus power is turned on; OFF = Module bus power is turned off. **SF:** Alarm indicator, yellow, ON = Abnormal alarm.
6	Power dissipation in W	2.3W

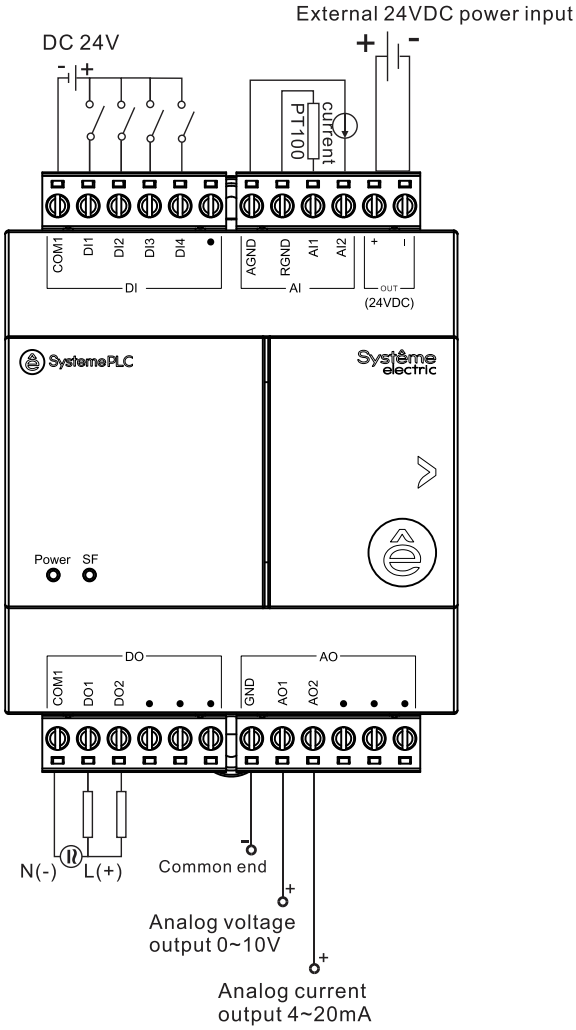
1.6.4 **SM172EDM0810**



No.	Interface	Description
1	Digital Input (DI)	4 channel active 24V inputs, source type connection, isolated
2	power interface	24V DC power input
3	Expansion module interface	Expansion module interface, Hot-swapping is not supported.
4	Digital Output (DO)	4-channel 0.5A transistor 24VDC drain output.
5	Panel indicator light	**POWER**: Bus Power Indicator, green, ON = Module bus power is turned on; OFF = Module bus power is turned off. **SF:** Alarm indicator, yellow, ON = Abnormal alarm.
6	Power dissipation in W	0.4W

1.6.5 **SM172EMIO1000**

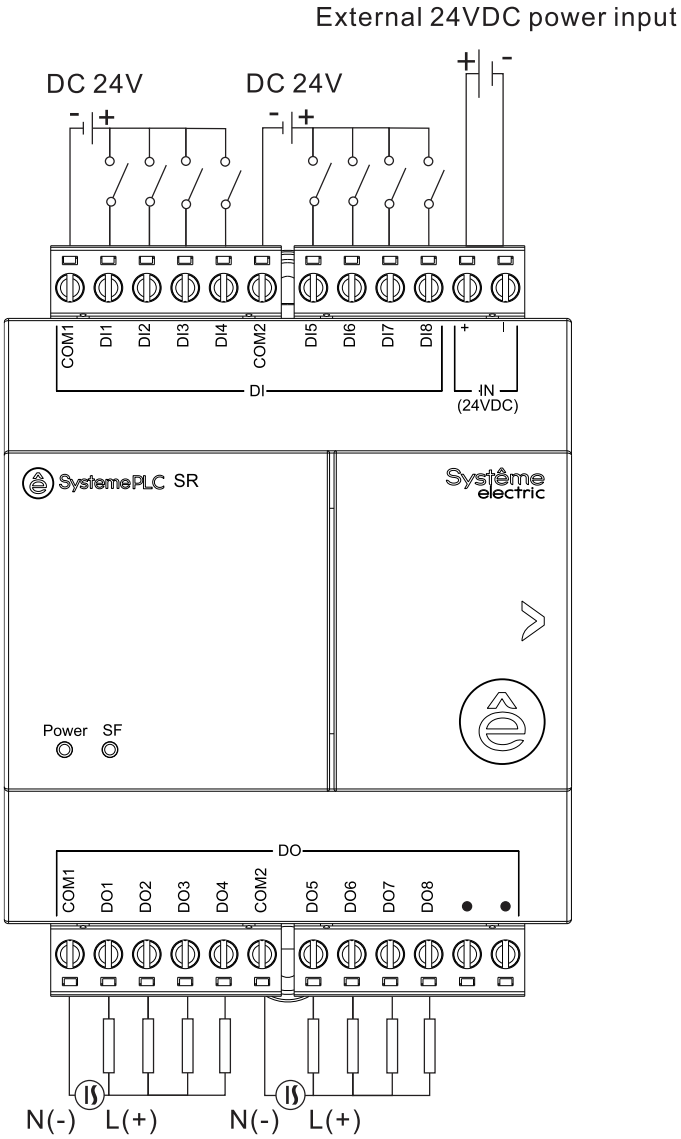
If there is a resistance input signal,
the resistance input signal and the current,
voltage input signals do not share the
common terminal (RGND, AGND).
If there's only current or voltage or current
and voltage input signals, you can connect to
any common terminal (RGND, AGND).



No.	Interface	Description
1	Digital Input (DI)	4 channel active 24V inputs, source type connection, isolated
2	power interface	24V DC power input
3	Analog input (AI) Resolution of 16 bits	**4 channel analogue inputs** Voltage signal: 0-10VDC Current signal: 4-20mA/0-20mA Resistance signal: 0-30kΩ, support NTC_10K (3950) or PT1000 PT100.
4	Expansion module interface	Expansion module interface, Hot-swapping is not supported.
5	Analog Output (AO)	**2 channel analogue output** Voltage signal: 0-10VDC Current signal: 4-20mA
6	Digital Output (DO)	2-channel 3A electromagnetic relay output.

No.	Interface	Description
7	Panel indicator light	**POWER**: Bus Power Indicator, green, ON = Module bus power is turned on; OFF = Module bus power is turned off. **SF:** Alarm indicator, yellow, ON = Abnormal alarm.
8	Power dissipation in W	2W

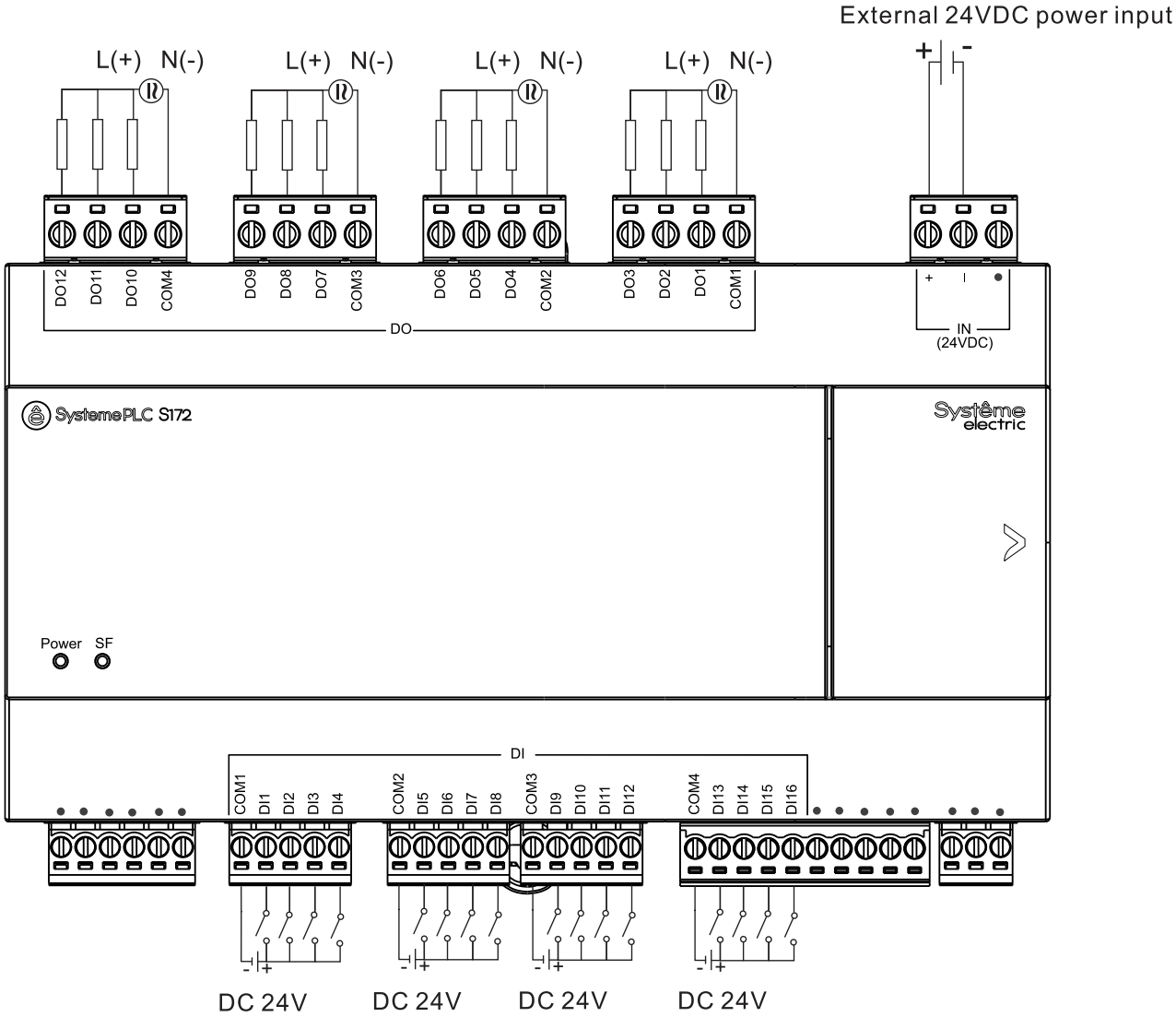
1.6.6 **SM172EDM1600**



No.	Interface	Description
1	Digital Input (DI)	8 channel active 24V inputs, source type connection, isolated
2	power interface	24V DC power input
3	Expansion module interface	Expansion module interface, Hot-swapping is not supported.
4	Digital Output (DO)	8-channel 3A electromagnetic relay output.
5	Panel indicator light	**POWER**: Bus Power Indicator, green, ON = Module bus power is turned on; OFF = Module bus power is turned off. **SF:** Alarm indicator, yellow, ON = Abnormal alarm.

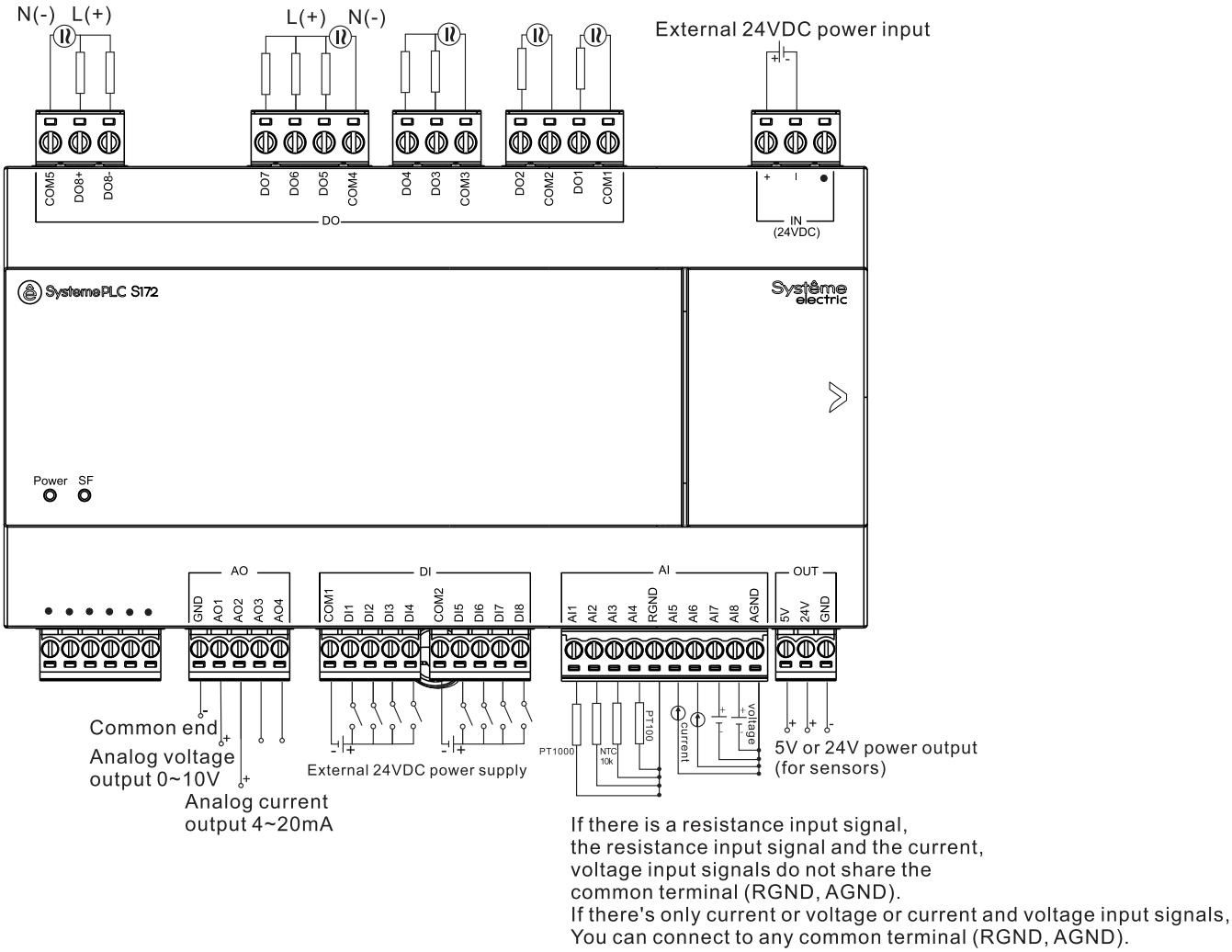
No.	Interface	Description
6	Power dissipation in W	1.4W

1.6.7 **SM172EDM2800**



No.	Interface	Description
1	Digital Output (DO)	12-channel 3A electromagnetic relay output.
2	power interface	24V DC power input
3	Expansion module interface	Expansion module interface, Hot-swapping is not supported.
4	Digital Input (DI)	16 channel active 24V inputs, source type connection, isolated
5	Panel indicator light	**POWER**: Bus Power Indicator, green, ON = Module bus power is turned on; OFF = Module bus power is turned off. **SF:** Alarm indicator, yellow, ON = Abnormal alarm.
6	Power dissipation in W	2.085W

1.6.8 **SM172EMIO2800**



No.	Interface	Description
1.	Digital Output (DO)	8-channel 3A electromagnetic relay output.
2.	power interface	24V DC power input
3.	Expansion module interface	Expansion module interface, Hot-swapping is not supported.
4.	Power supply output	Provide external 5V and 24V power supply interfaces with a power of 1W to supply power to active sensors.
5.	Analog input (AI) Resolution of 16 bits	**8 channel analogue inputs** Voltage signal: 0-10VDC Current signal: 4-20mA/0-20mA Resistance signal: 0-30kΩ, support NTC_10K (3950) or PT1000 PT100
6.	Digital Input (DI)	8 channel active 24V inputs, source type connection, isolated
7.	Analog Output (AO)	**4 channel analogue output** Voltage signal: 0-10VDC Current signal: 4-20mA
8.	Panel indicator light	**POWER**: Bus Power Indicator, green, ON = Module bus power is turned on; OFF = Module bus power is turned off. **SF:** Alarm indicator, yellow, ON = Abnormal alarm.

9.	Power dissipation in W	6W

Attention

1) Do not install or wire the controller and related equipment under live electrical conditions, as incorrect operation may result in mechanical destruction, personal injury or even death. Before installing and removing any electrical equipment, make sure the power supply to that equipment is disconnected.

2) Please strictly follow the wiring diagram above, otherwise it may cause equipment damage or serious personal injury.

1.7 Working environment

Table 1-7 All Smart Relay Master and Expansion Modules Comply with the Electrical Specifications Listed in the Table

Environmental conditions - transport and storage	
Temperature	-40℃~+70℃(-40℉~158℉)
Atmospheric pressure	1080 hPa~660 hPa (corresponding to altitudes of -1000m~+3500m)
Relative Humidity	10%~95%, Non-condensing
Fall	1m, 10 times, transport packaging
Environmental conditions - work	
Temp	-20℃~60℃(-4℉~140℉) **Note:** If the operating temperature is greater than the maximum temperature, a fan or air conditioner must be installed in the direction of the heat sink.
Atmospheric pressure	1080 hPa~795 hPa (corresponding to altitudes of -1000m~+2000m)
Relative humidity	10%~95% RH, non condensing
Harsh Environment Pollutant concentration	Low salt spray, humidity, dust and mist environment. SO_2<0.5ppm, Relative humidity<60%, non condensing. H_2S<0.1ppm, Relative humidity<60%, non condensing.
EMC - Immunity	
Comply with GB/T15969.2 and IEC61131.2	
Environmental testing	
High temperature operation IEC60068-2 Low temperature operation IEC60068-2	60℃(140℉), duration: 24h, temperature change rate: ≤ 1℃(33.8℉)/min, room temperature recovery time: ≥ 1h. -20℃(-4℉), duration: 24h, temperature change rate: ≤ 1℃(33.8℉)/min, room temperature recovery time: ≥ 1h.
High temperature start-up IEC60068-2 Low temperature start-up IEC60068-2	60℃, 2 hours, after the temperature is stabilised, no continuous power on 3 times. -20℃, 2 hours, after the temperature stabilisation, not continuously on 3 times.

High and low temperature cycle operation IEC60068-2	-20℃~60℃, residence time of 3 hours, temperature rate of 1℃/min, 5 cycles, room temperature recovery time of 1 hour.
High temperature storage IEC60068-2 Low temperature storage IEC60068-2	70℃, storage time: 72h, temperature change rate: ≤ 1℃/min, room temperature recovery time: ≥ 1h. -40℃, storage time: 72h, temperature change rate: ≤ 1℃/min, room temperature recovery time: ≥ 1h.
Thermal Shock IEC60068-2	-40℃~70℃, residence time 3h, temperature change time: 2min, 3 cycles.
Constant high humidity IEC60068-2	Storage temperature: 40°C. Storage time: 48h. Storage humidity: 93%.
Environmental Chamber IEC60068-2	25℃~55℃, humidity: 95%, single cycle: 24 hours, 2 cycles.
Sine vibration (bare metal) IEC60068-2-6	5Hz～8.4Hz, 3.5mm, 8.4Hz～150Hz, 1g, X/Y/Z triaxial, 10 cycles/axis
Mechanical shock (bare metal) IEC60068-2-27	150m/s^2, 11ms, ±Six directions of X/Y/Z, 3 times/direction, 18 times in total.

1.8 System architecture

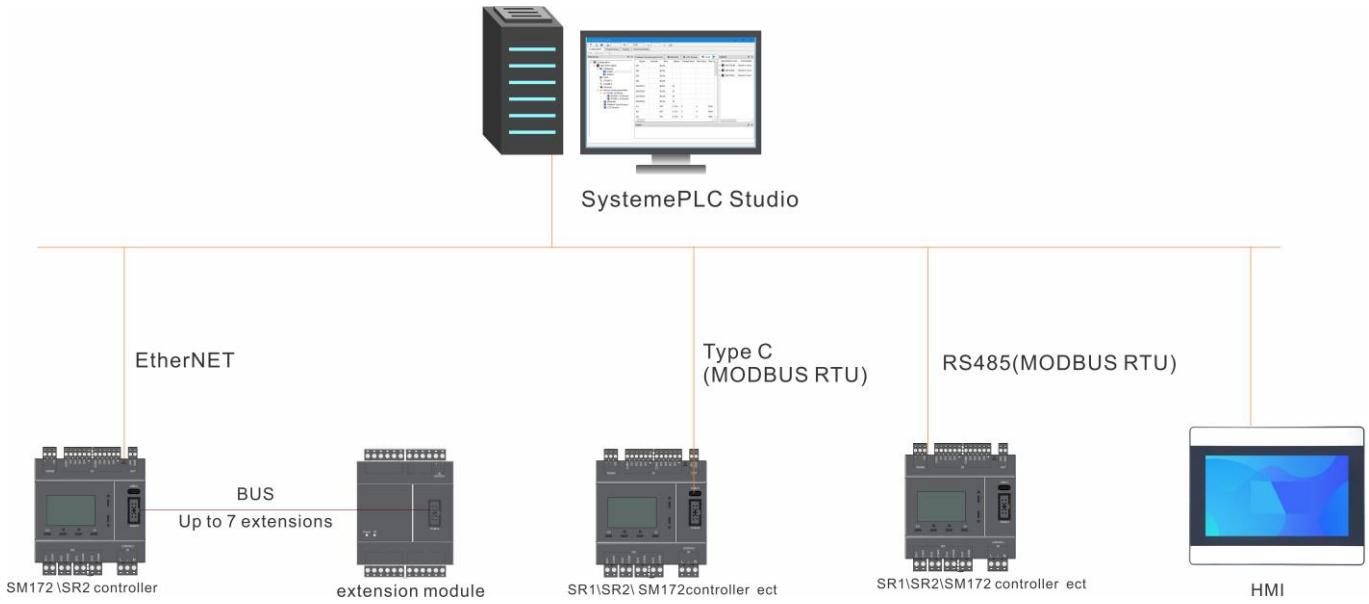


Figure 1-1 Network Architecture Diagram

Second section

Installing

2.1 Installation precautions

2.2 Installation dimensions

2.3 Installation method

2.4 Grounding and wiring

2.1 Installation precautions

Follow the guidelines to install the field controller:

• To reduce vibration and impact damage, please transport the controller in the original packaging.

• Confirm that all components are shipped with the controller.

• Do not drop the controller on the ground or subject it to physical impact.

1) Isolate Smart Relay Series Controller from the heating device, high voltage and electronic noise

When installing equipment and components, separate equipment with high voltage and high electronic noise from Smart relay series controller low-voltage electronic equipment.

When arranging Smart Relay series controller on the back panel of the control cabinet, electronic devices should be arranged in the area with low temperature in the control cabinet. Long term operation in high temperature environment will shorten the service life of electronic devices.

The backplane wiring of the control cabinet shall be considered, and the AC power supply line, high energy DC signal line with high switching frequency, low-voltage signal line and communication cable shall not be designed in the same slot as far as possible.

2) Leave proper space for heat dissipation and wiring

Smart relay series controller is designed with natural ventilation for heat dissipation, and at least 30mm space must be reserved above and below the module for normal heat dissipation. The distance between the front panel and the back panel shall be at least 80mm.

Attention

1) When installed vertically, the maximum allowable ambient temperature should be 10 ℃ lower than when installed horizontally, and the CPU should be installed below all expansion modules.

2) Before connecting AI to external sensors, it is recommended to use a multimeter to measure the sensor voltage and confirm that the specifications are met before connecting to the controller to avoid burning out the controller hardware.

3) Try to keep the wire as short as possible and ensure that the wire thickness meets the current requirements. The suitable wire thickness for terminal blocks is 2mm² to 0.3mm² (14AWG to 22AWG), and using shielded cables can achieve the best anti electronic noise characteristics. Generally, grounding the shielding layer can achieve the best effect.

3) Power budget

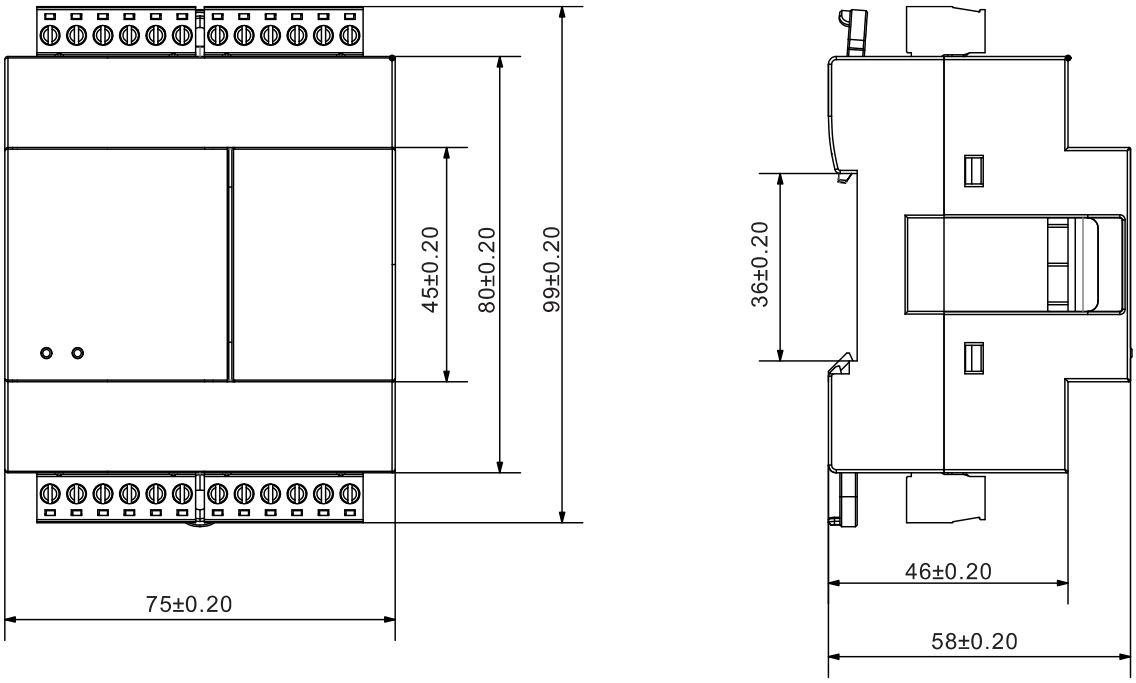
All Smart Relay series controllers have an internal power supply that provides 5V and 24V DC power to external sensors.

Special attention must be paid to the system configuration to ensure that the 5V and 24V power supplies provided by the controller can meet the needs of the sensor you have selected. If your configuration requirements exceed the power supply capacity of the controller, an external power supply will be required to power the sensor.

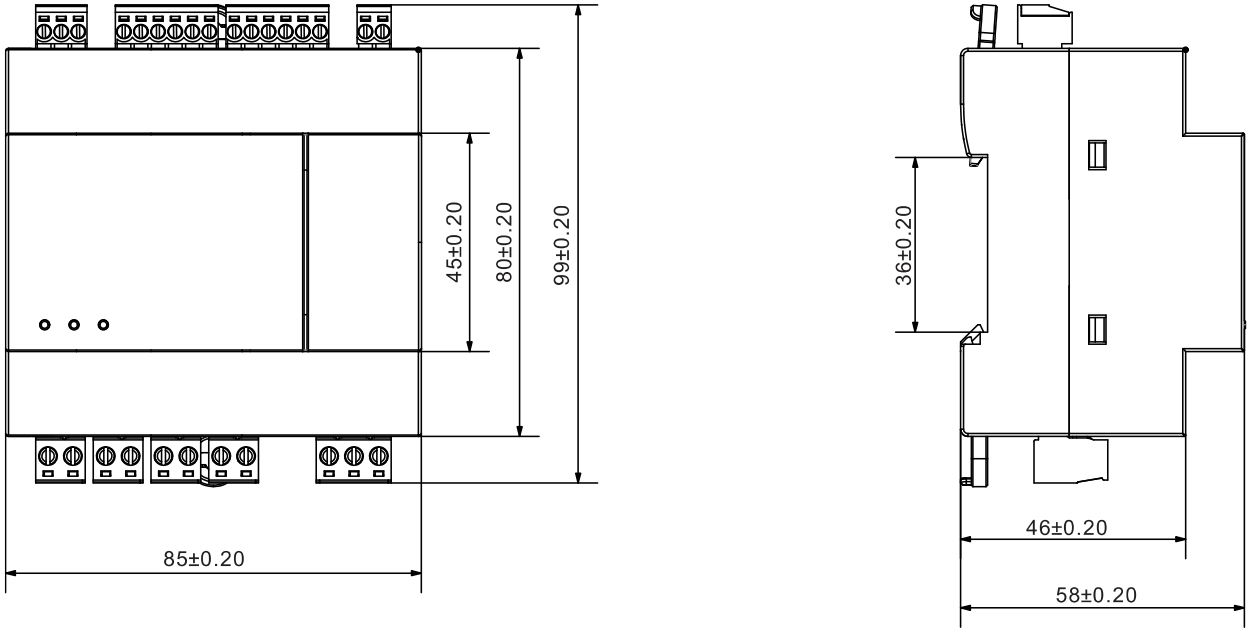
The 24VDC sensor power supply of the Smart Relay series controller should not be supplied to the same point simultaneously with any external power supply.

2.2 Installation dimensions

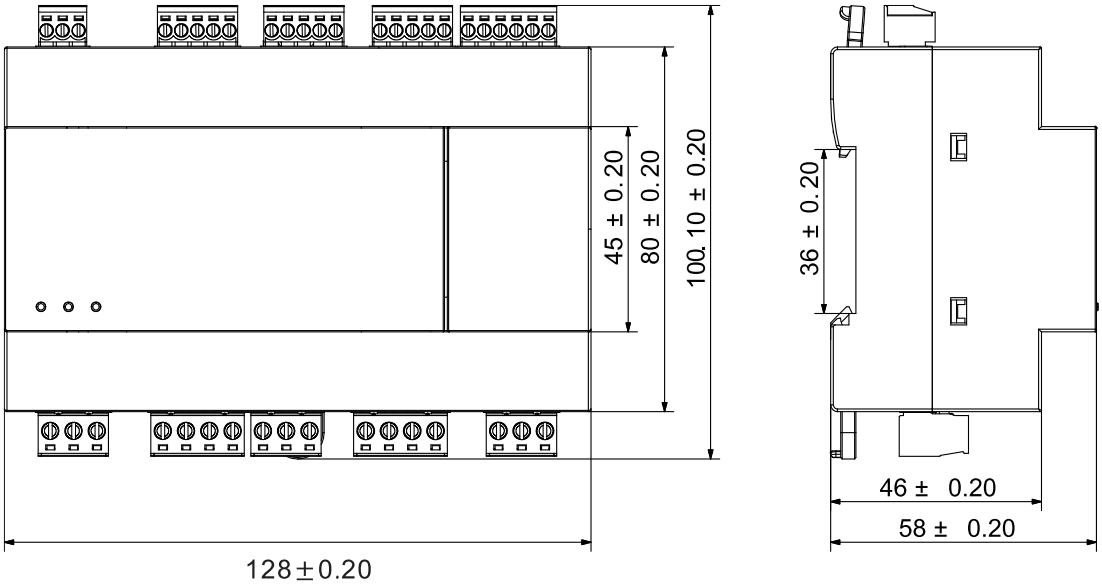
Dimensions of SM172EMIO1000, SM172EDM1600, SM172EDM0800, SM172EDM0810, SM172EDM0800P7, SM172EAM0800 (Unit: mm):



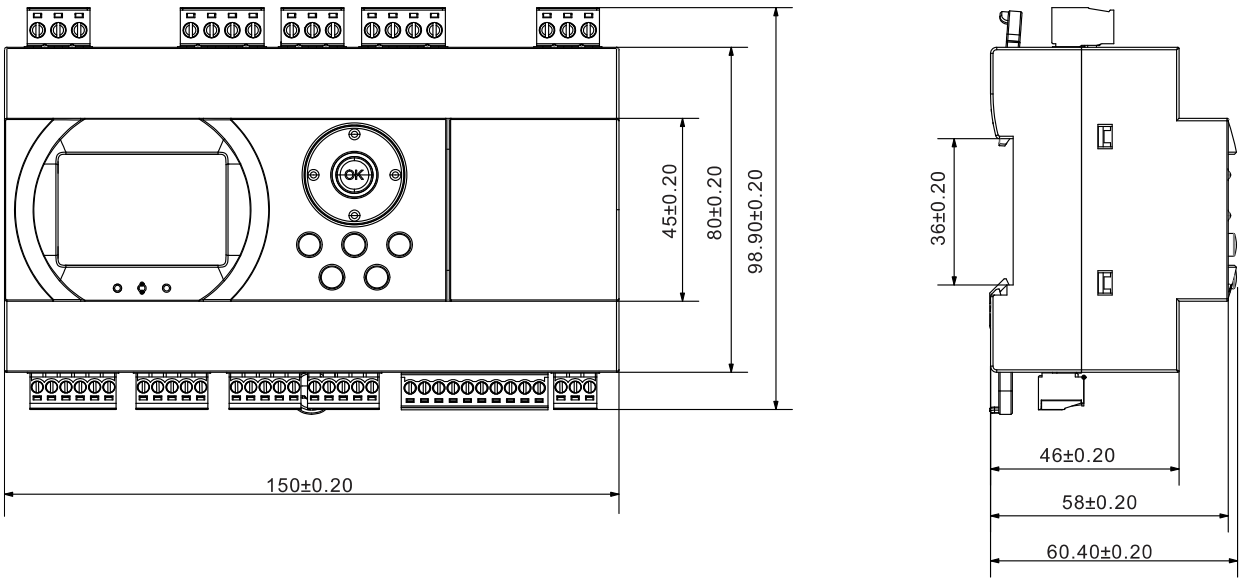
Dimensions of ZR1PB00P7, ZR1PB00BD, ZR1PP00BD2A, ZR2PA11BD, ZR2PP11BD2A (Unit: mm):



Dimensions of ZR1PA00P7, ZR1PA00BD, ZR1PP00BD4A (Unit: mm):



Dimensions of ZR2PP11P7, ZR2PP11BD, ZR2PB11P7, SM172PS11BDR, SM172PS11BDM, SM172PS11BDT, SM172EMIO2800, SM172EDM2800 (Unit: mm):



2.3 Installation method

When installing and disassembling PLC and its related equipment, appropriate safety measures must be taken in advance and its power supply must be confirmed to be cut off.

When replacing or installing the PLC, it is necessary to ensure that the correct or equivalent module is used. In addition to using the same module, it is also necessary to ensure that the installation direction and position are correct. If the incorrect module is installed, the PLC program may produce the wrong function.

Warning

● Installing or disassembling PLCs and related equipment under live conditions may result in electric shock or equipment misoperation. Failure to cut off all power sources during the installation and disassembly of controllers and related equipment may cause death or serious personal injury and equipment damage.

	● If the same modules are not used to replace the PLC in the same direction and sequence, it may result in death or serious personal injury and equipment damage.

1) Installation method

PLC can be installed on the back panel of the control cabinet or on standard DIN rails. It can be installed horizontally or vertically, and the CPU and power module should always be installed on the left or bottom during installation.

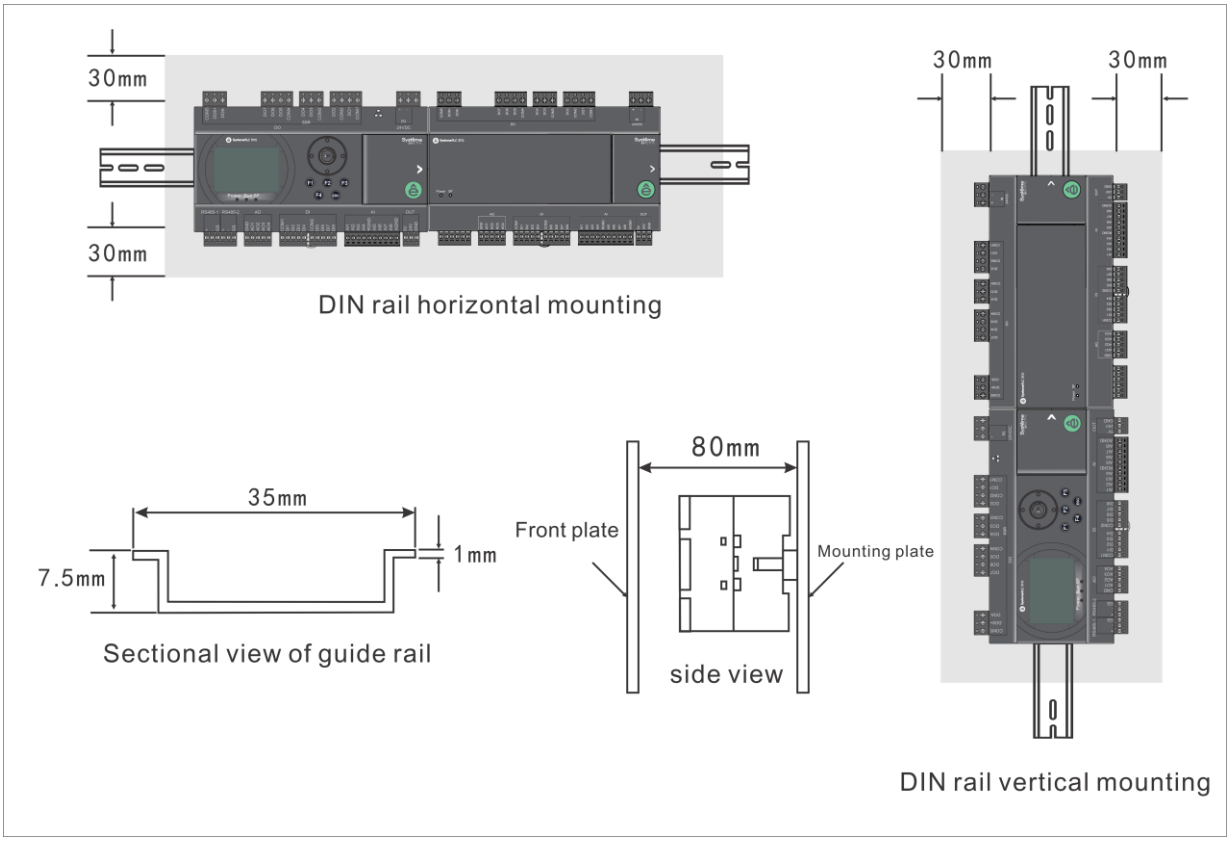


Figure 2-3 Installation Diagram

2) Installation and removal

Please install or remove the PLC according to the following methods.

● DIN Rail Mounting

1) Mount the rail on the back panel and keep the spacing at 80mm.

2) Open the DIN clamp at the bottom of the module and secure the back of the module to the DIN rail using the DIN rail clamp.

3) If an expansion module is used, connect the flat cable of the expansion module to the expansion port on the front cover.

4) Rotate the module to fit closely against the DIN rail and close the DIN clamp.

5) Carefully check that the DIN clamp on the module is securely fixed to the DIN rail.

⚠	**Attention** When the PLC is used in an environment with significant vibrations or is installed vertically, DIN rail stop blocks should be used. If the system is in a high-vibration environment, a backplane installation method can provide a higher level of vibration protection.

● Remove CPU or expansion module

1) Remove the power supply of PLC.

2) Remove all wiring and cables from the module.

3) If any other expansion module is connected to the module you removed, open the upper cover and unplug the expansion flat cable of the adjacent module.

4) Remove mounting screws or open the DIN clip.

2.4 Grounding and wiring

Grounding and Wiring Guidelines for PLCs

Proper grounding and wiring are critical for all electrical equipment, ensuring optimal system performance and providing better electronic noise protection.

Before grounding and wiring, ensure that the power supply to the equipment has been disconnected, and that the power supply to related equipment has also been disconnected.

When wiring PLCs and related equipment, ensure that all applicable electrical coding rules are followed. Install and operate all equipment in accordance with all applicable national or regional standards. Contact authorities in your area to determine which standards meet your specific needs.

Warning

Grounding or wiring under live conditions may cause death or serious personal injury and equipment damage.

The grounding and wiring of PLC systems must consider safety factors, otherwise it may cause equipment misoperation. Therefore, you should comply with all safety regulations to avoid personal injury and equipment damage.

Warning

Controlling equipment may cause misoperation of the devices it controls. This kind of misoperation may lead to death or serious personal injury and equipment damage. Therefore, the system must have emergency stop function independent of PLC, electromechanical interlock or other redundant safety facilities.

Third section

Configuration of a simple project

3.1 SystemePLC Studio and PLC hardware connection

3.2 PC system requirements for SystemePLC Studio installation

3.3 New project

3.4 Hardware configuration

3.5 Programming

3.6 SystemePLC Studio communication with PLC

3.7 Program compilation and download

3.8 Monitoring

3.9 Global variables

3.10 Use of branchers

3.11 Use of FDO

3.12 Program Working Status

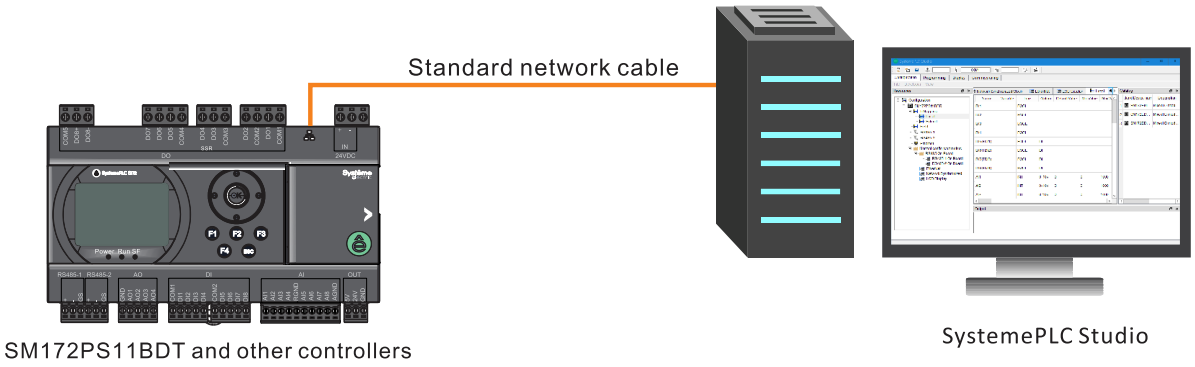
3.13 Task

3.14 Expansion module status

3.1 SystemePLC Studio and PLC hardware connection

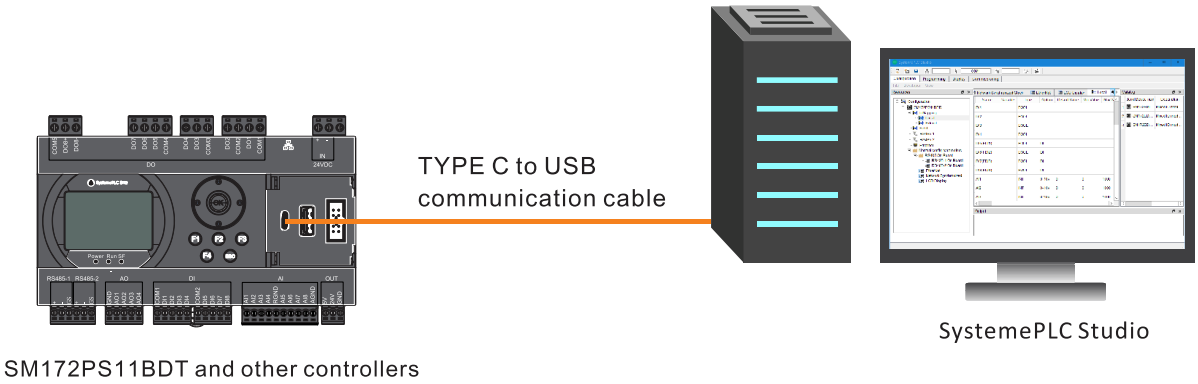
1. Connected to the controller through the Ethernet interface

Connection of the programming device PC to the controller via standard network cable (RJ45 communication port, MODBUS TCP protocol).



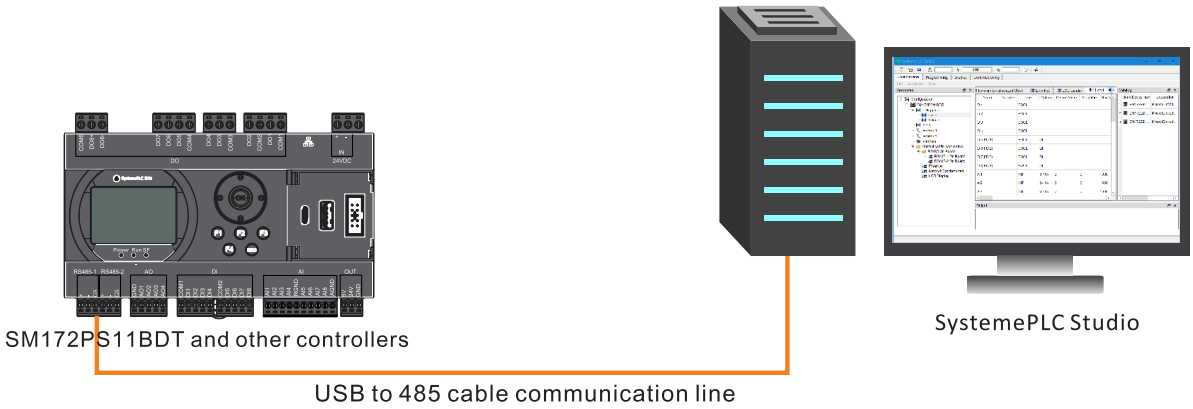
2. Connected to the controller through the TYPE C port

Connection of the programming device PC to the controller using a TYPE C communication cable (TYPE C communication port, MODBUS RTU protocol).



3. Through the RS485 port connected to the controller

Connect the programming device PC to the controller using a USB to 485 cable (RS485 communication port, MODBUS RTU protocol).



Attention

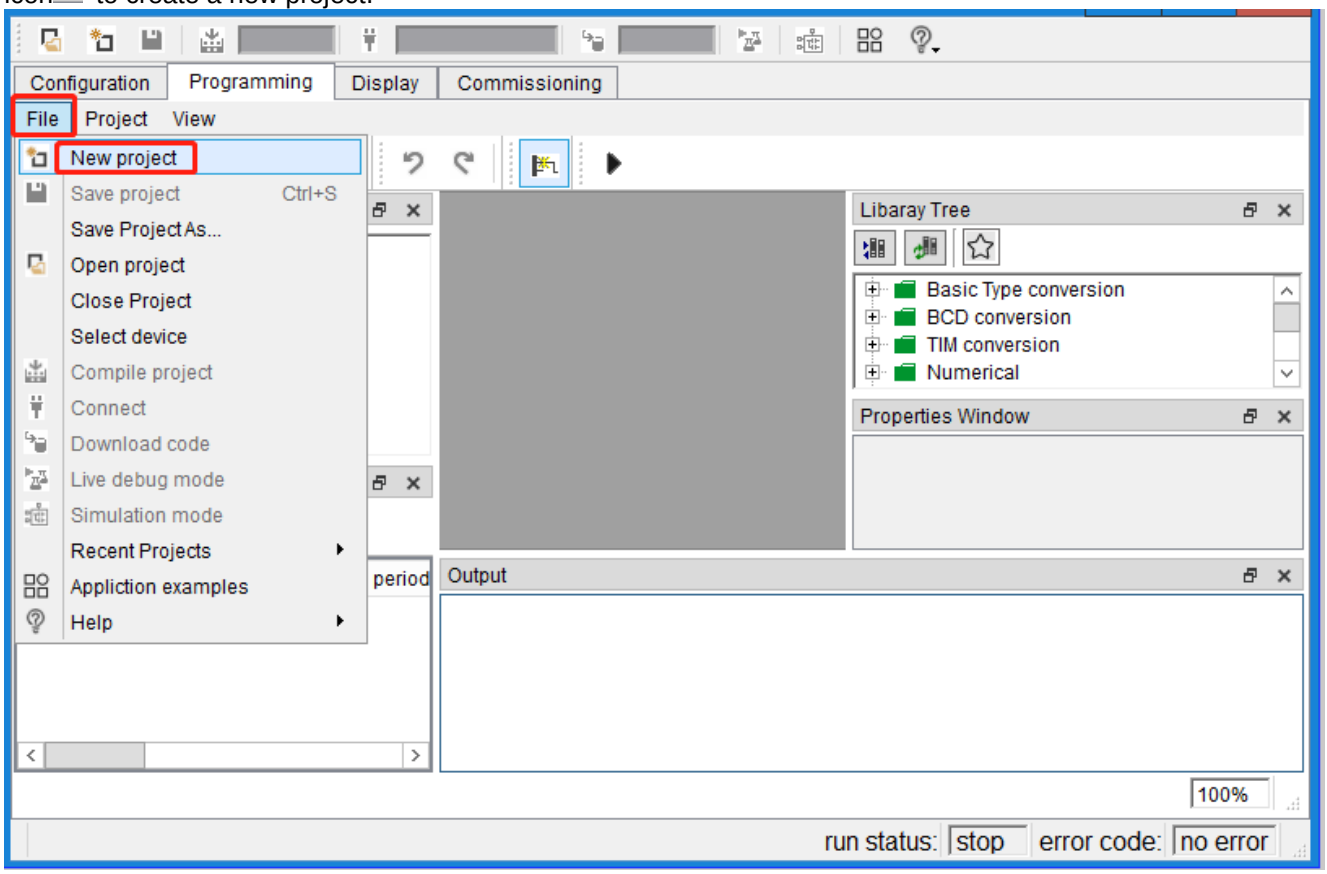
When using the SM172EMIO2800 hybrid module, the controller and hybrid module must be powered on at the same time, or the controller must be powered on before the module, otherwise the module may not be found!

3.2 PC system requirements for SystemePLC Studio installation

- **Operating System:** Windows 10 64-bit / Windows 11 64-bit.
- **Processor:** 2 GHz or faster 64-bit compatible processor (dual-core or multi-core).
- **RAM:** 4 GB or more.
- **Storage Space:** 10 GB or more.
- **Display Resolution:** 1920×1080 (1080p) or higher is recommended.

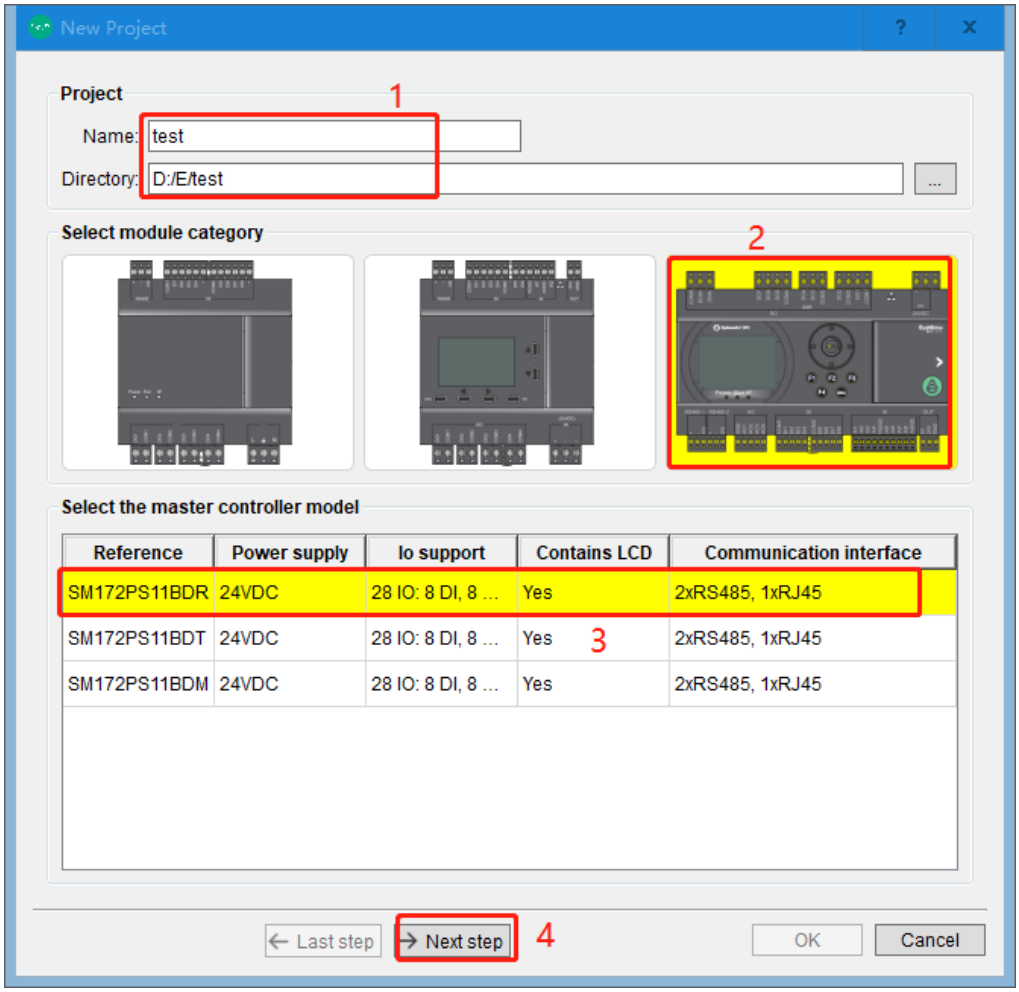
3.3 New project

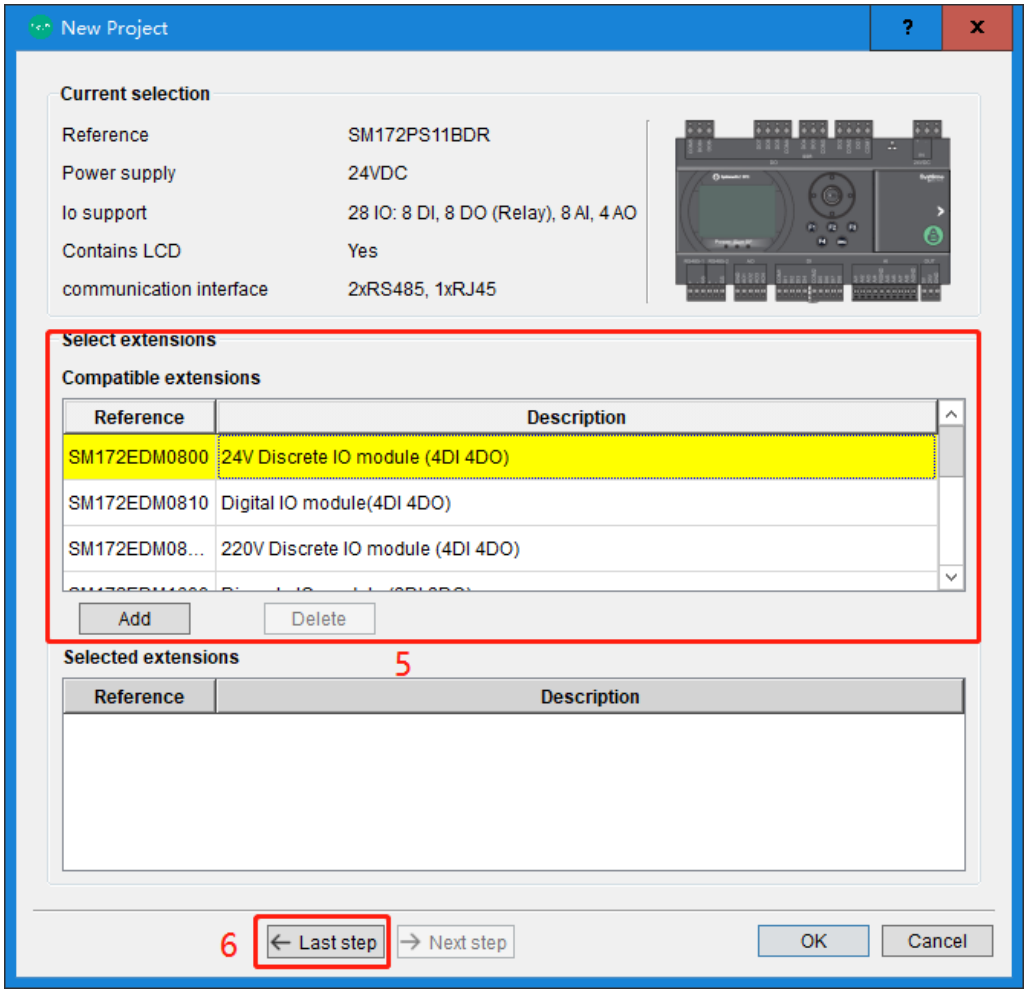
1. The starting interface of SystemePLC Studio is as follows. Create a new project under File or click the New project icon to create a new project.



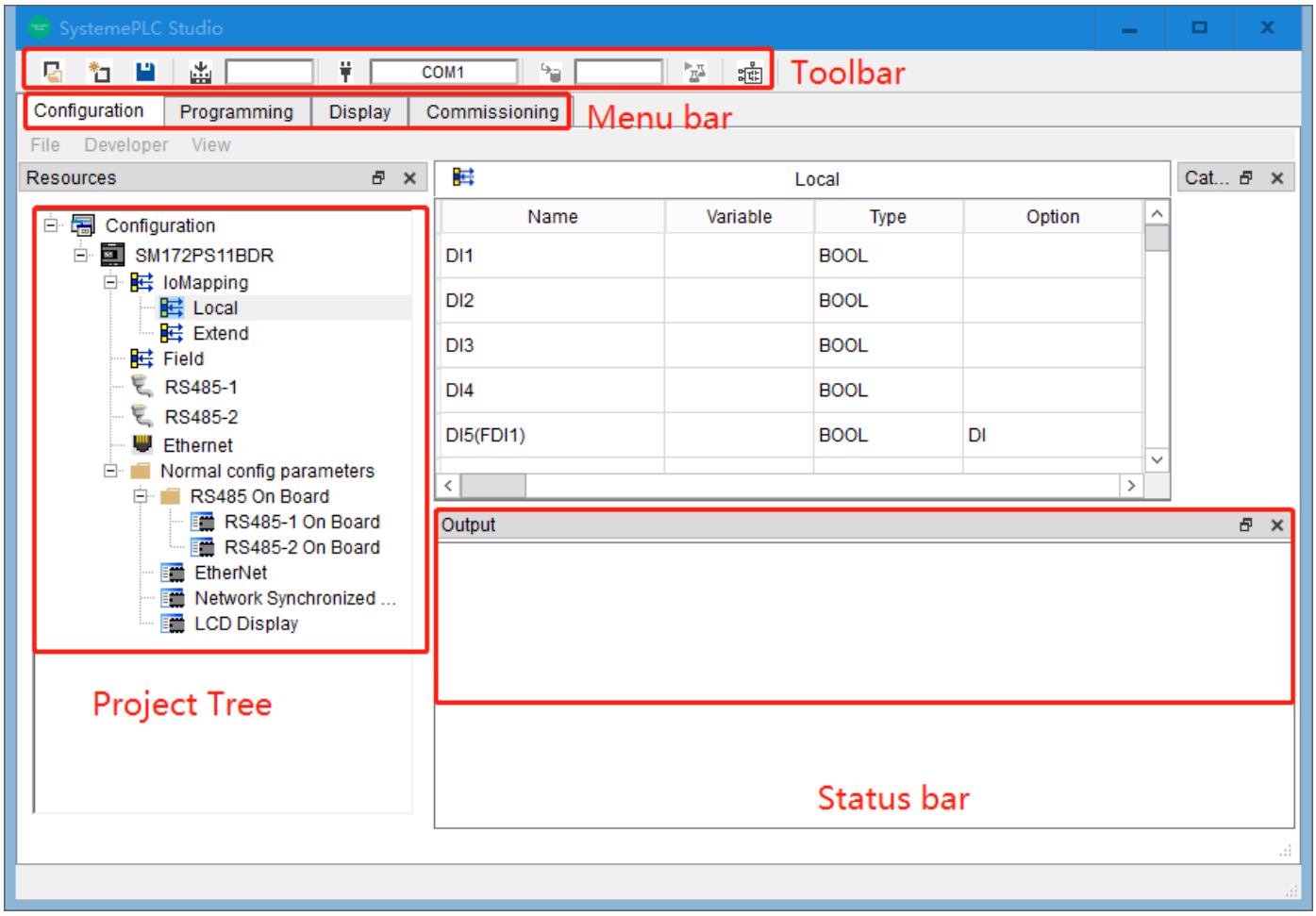
2. Set the project name and project save path. Select the target controller and expansion module according to the numbers in the figure below. The expansion module is optional and can be added later in the project. The controller is divided according to the display screen. First, select whether it has a display screen, and then select the specific CPU model.

Attention: The project save path cannot contain Chinese characters and Chinese symbols, otherwise it will cause the program to fail to run.


New Project
Project
1
Name: test
Directory: D:/E/test
Select module category
2
Select the master controller model
Reference | Power supply | Io support | Contains LCD | Communication interface
SM172PS11BDR | 24VDC | 28 IO: 8 DI, 8 ... | Yes | 2xRS485, 1xRJ45
SM172PS11BDT | 24VDC | 28 IO: 8 DI, 8 ... | Yes | 2xRS485, 1xRJ45
SM172PS11BDM | 24VDC | 28 IO: 8 DI, 8 ... | Yes | 2xRS485, 1xRJ45
3
Last step
Next step
4
OK
Cancel


New Project
Current selection
Reference SM172PS11BDR
Power supply 24VDC
Io support 28 IO: 8 DI, 8 DO (Relay), 8 AI, 4 AO
Contains LCD Yes
communication interface 2xRS485, 1xRJ45
Select extensions
Compatible extensions
Reference | Description
SM172EDM0800 | 24V Discrete IO module (4DI 4DO)
SM172EDM0810 | Digital IO module(4DI 4DO)
SM172EDM08... | 220V Discrete IO module (4DI 4DO)
Add
Delete
Selected extensions
5
Reference | Description
6
Last step
Next step
OK
Cancel

3. The basic interface of the project is as follows:



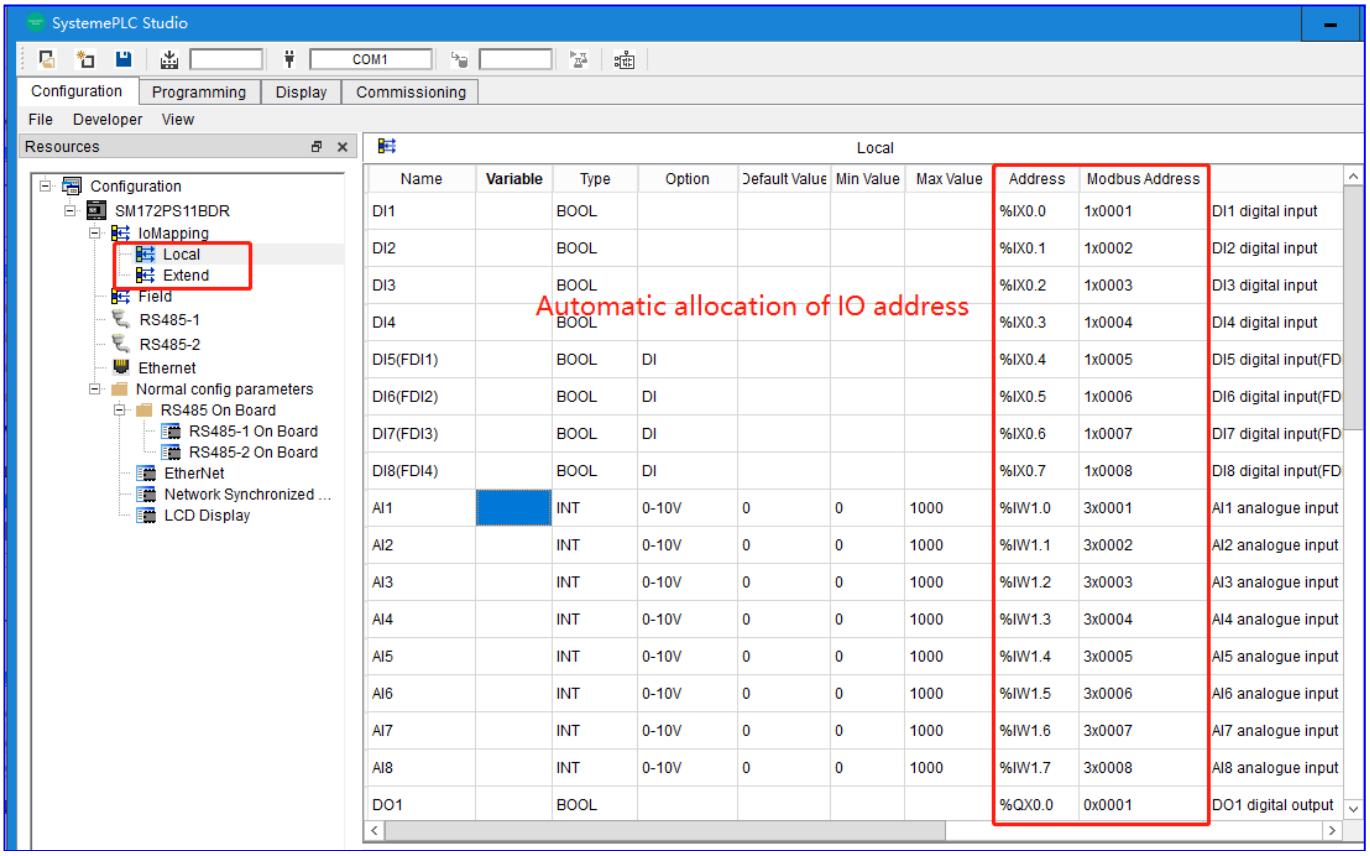
Item	Describe
Toolbar	The toolbar provides quick access to commonly used functions such as creating, opening or saving projects, compiling and downloading projects, communication, simulation, etc.
Menu bar	There are four options under the menu bar: hardware configuration, programming, screen display programming, and communication configuration.
Project Tree	Each menu has a corresponding item tree.
Status bar	Display project status, such as compile download, save project information, etc.

3.4 Hardware configuration

After create a new PLC system, hardware configuration is required. Hardware configuration refers to the process of arranging the PLC and its supporting modules according to their connection relationships and setting hardware models, parameters, etc.

IO mapping

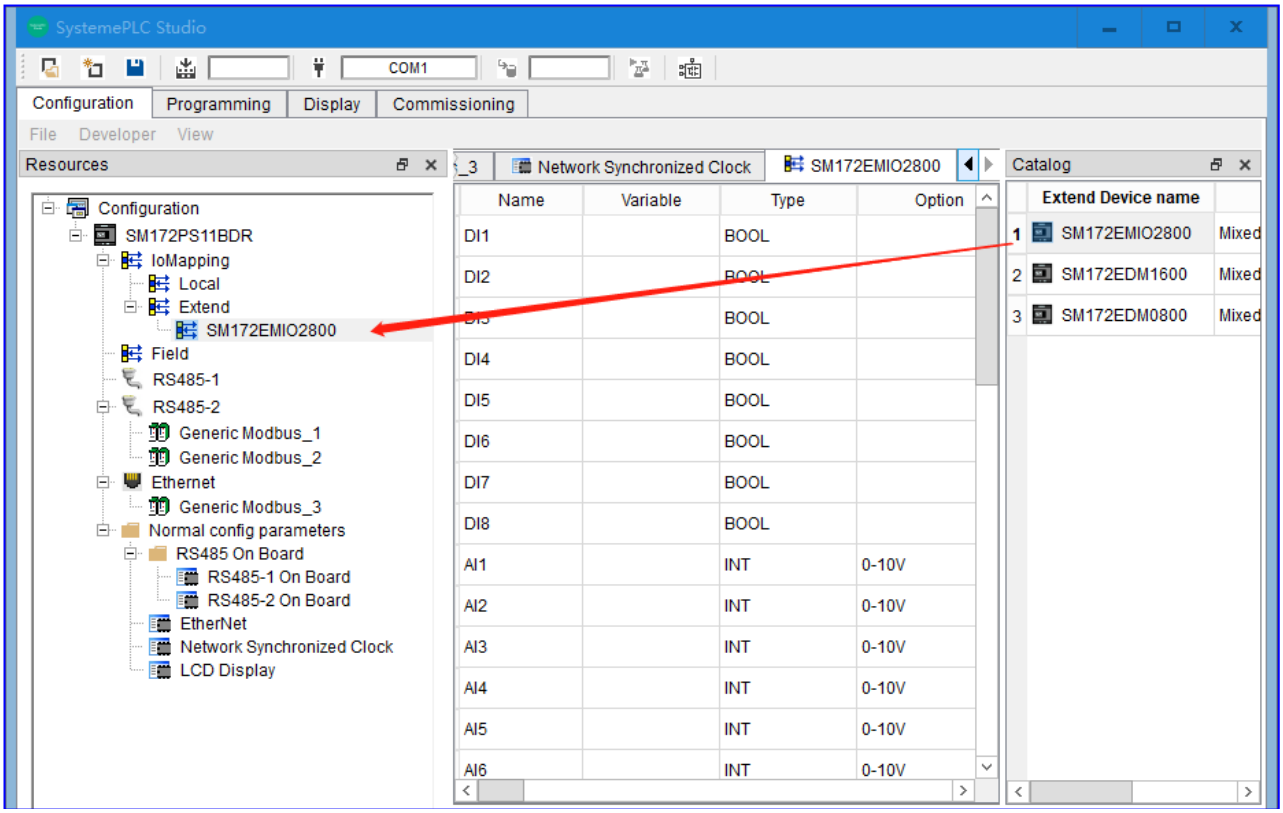
1. IO mapping is divided into local IO and extended IO mapping, Automatic allocation of IO addresses and corresponding Modbus addresses.



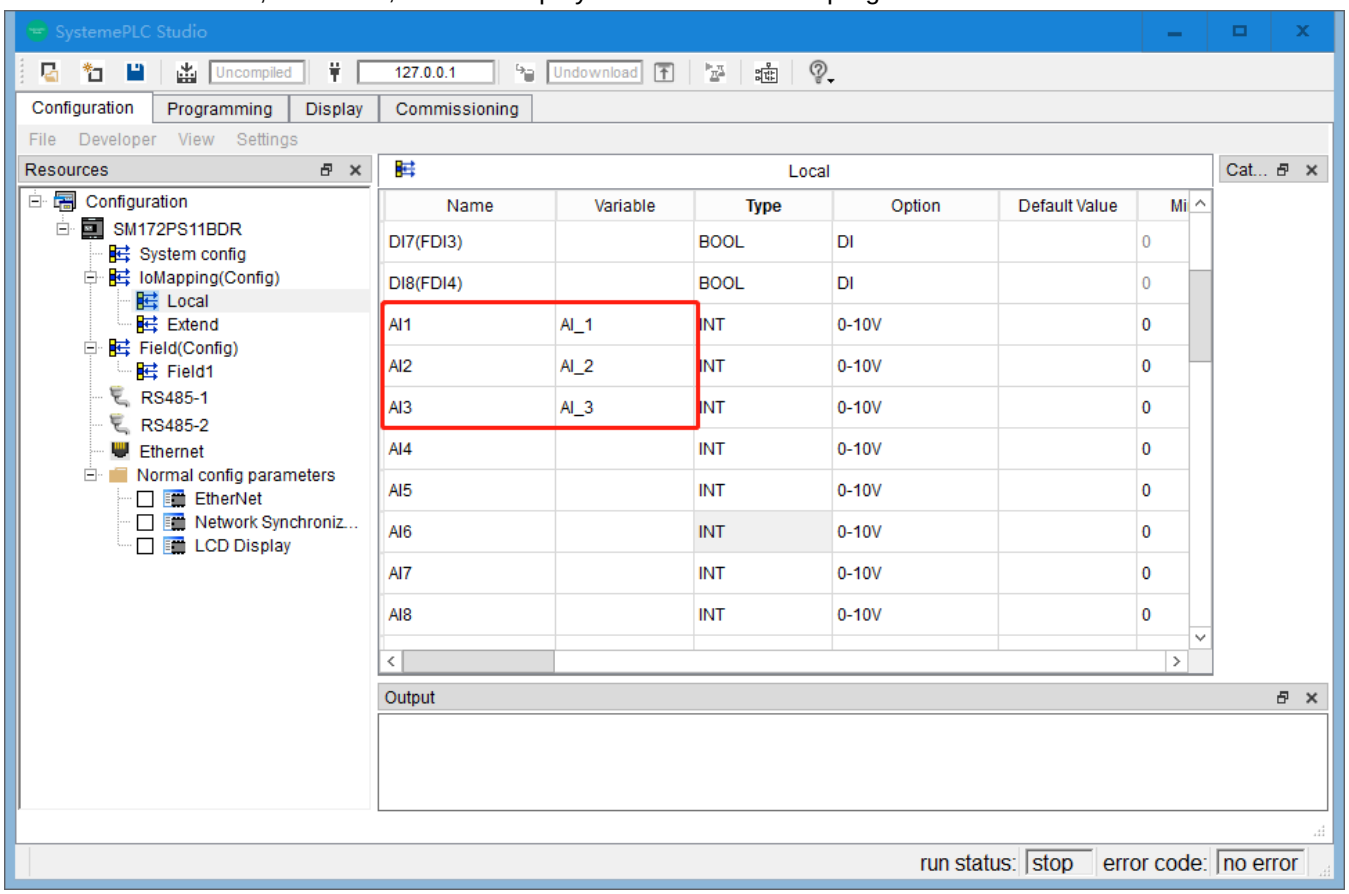
Name	Variable	Type	Option	Default Value	Min Value	Max Value	Address	Modbus Address	
DI1		BOOL					%IX0.0	1x0001	DI1 digital input
DI2		BOOL					%IX0.1	1x0002	DI2 digital input
DI3		BOOL					%IX0.2	1x0003	DI3 digital input
DI4		BOOL					%IX0.3	1x0004	DI4 digital input
DI5(FDI1)		BOOL	DI				%IX0.4	1x0005	DI5 digital input(FD
DI6(FDI2)		BOOL	DI				%IX0.5	1x0006	DI6 digital input(FD
DI7(FDI3)		BOOL	DI				%IX0.6	1x0007	DI7 digital input(FD
DI8(FDI4)		BOOL	DI				%IX0.7	1x0008	DI8 digital input(FD
AI1		INT	0-10V	0	0	1000	%IW1.0	3x0001	AI1 analogue input
AI2		INT	0-10V	0	0	1000	%IW1.1	3x0002	AI2 analogue input
AI3		INT	0-10V	0	0	1000	%IW1.2	3x0003	AI3 analogue input
AI4		INT	0-10V	0	0	1000	%IW1.3	3x0004	AI4 analogue input
AI5		INT	0-10V	0	0	1000	%IW1.4	3x0005	AI5 analogue input
AI6		INT	0-10V	0	0	1000	%IW1.5	3x0006	AI6 analogue input
AI7		INT	0-10V	0	0	1000	%IW1.6	3x0007	AI7 analogue input
AI8		INT	0-10V	0	0	1000	%IW1.7	3x0008	AI8 analogue input
DO1		BOOL					%QX0.0	0x0001	DO1 digital output

2. When connecting expansion modules, drag the expansion module on the right to the "Extend" area on the left. Each controller supports up to 7 expansion modules.

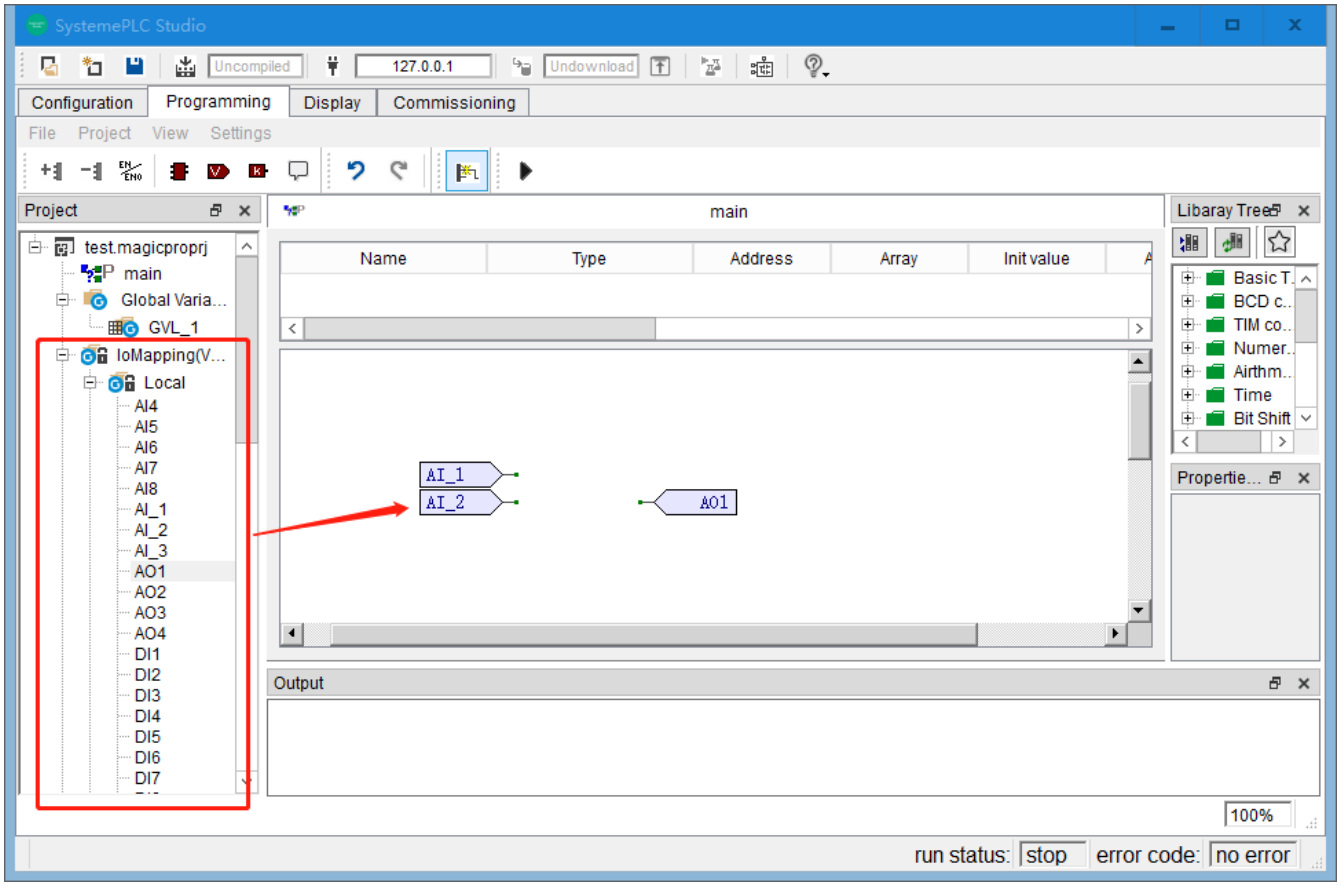
Attention: When connecting an expansion module, the controller and the expansion module must be powered on simultaneously, or the expansion module must be powered on before the controller; otherwise, the module may not be detected.



3. When associating variables in the program with hardware I/O (Input/Output), you can manually enter and modify the I/O variable name; otherwise, it will be displayed as "name" in the program.



4. After variables and hardware IO are associated, the corresponding IO variables will be mapped in Programming's IoMapping, and the variables will be directly dragged into the program to be called when programming. The same applies to the IO address mapping of the extension module.

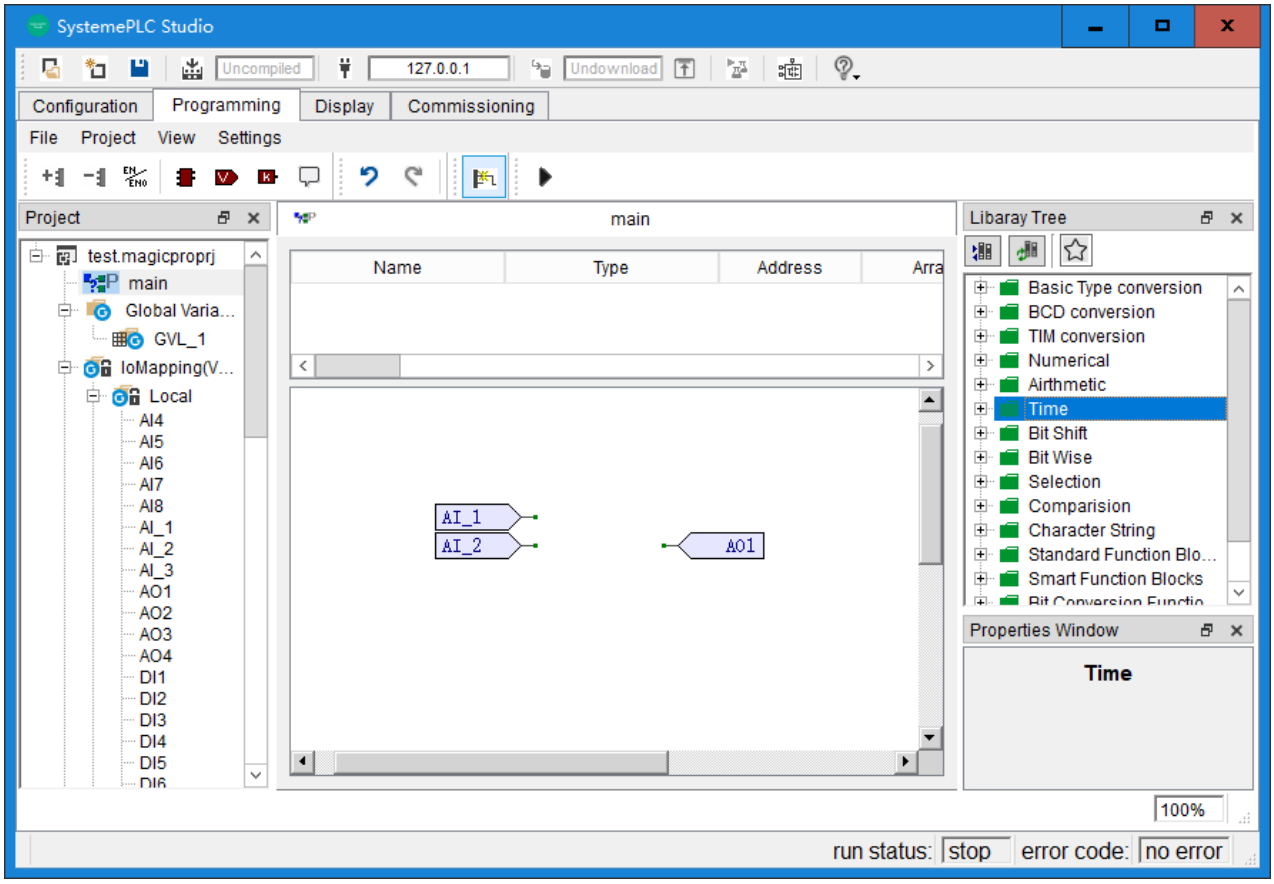


3.5 Programming

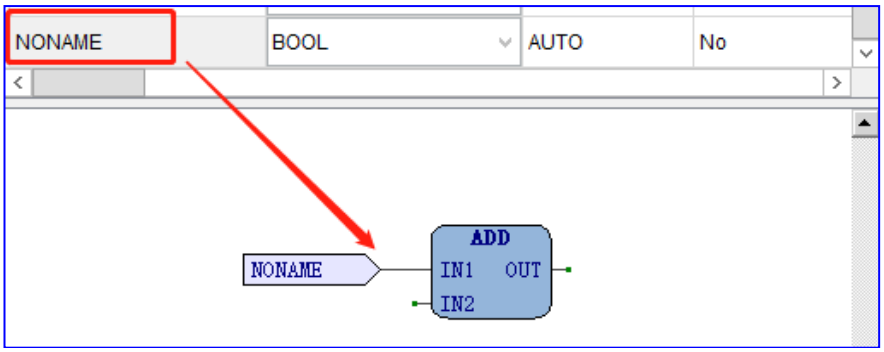
Taking a ticker program as an example, the programming steps will be explained below.

1. Each project contains a default main program, which can be edited in the Main section under Programming.

The variables and their data types in the variable declaration area can be manually modified.

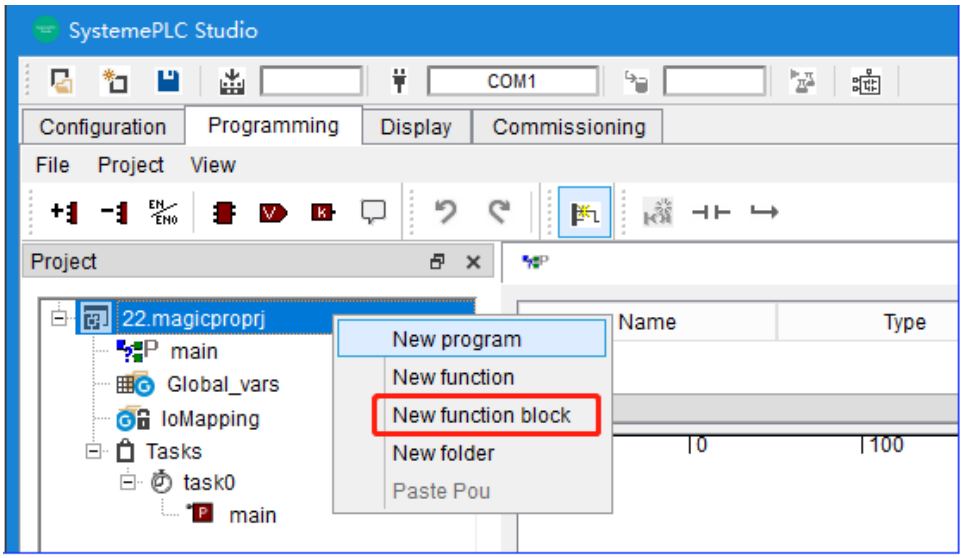


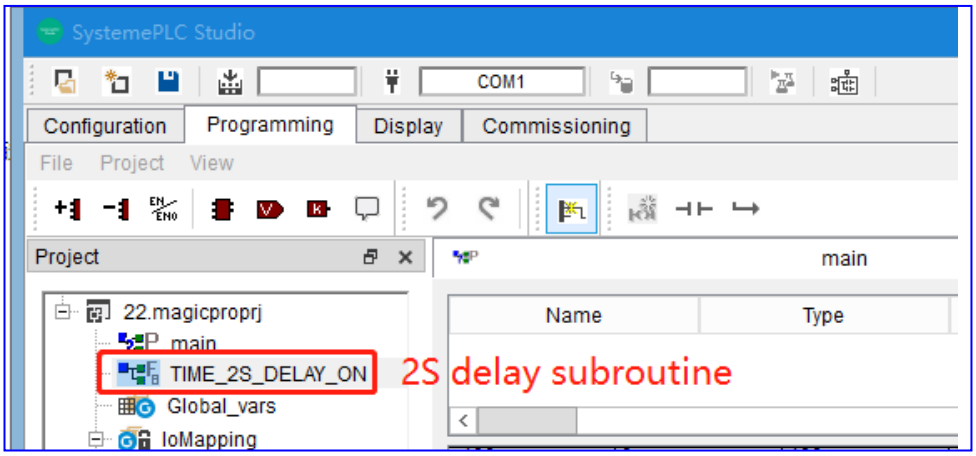
Tip: Drag and drop variables from the variable declaration area directly into the program, and the variable pins will automatically become input or output and connected.



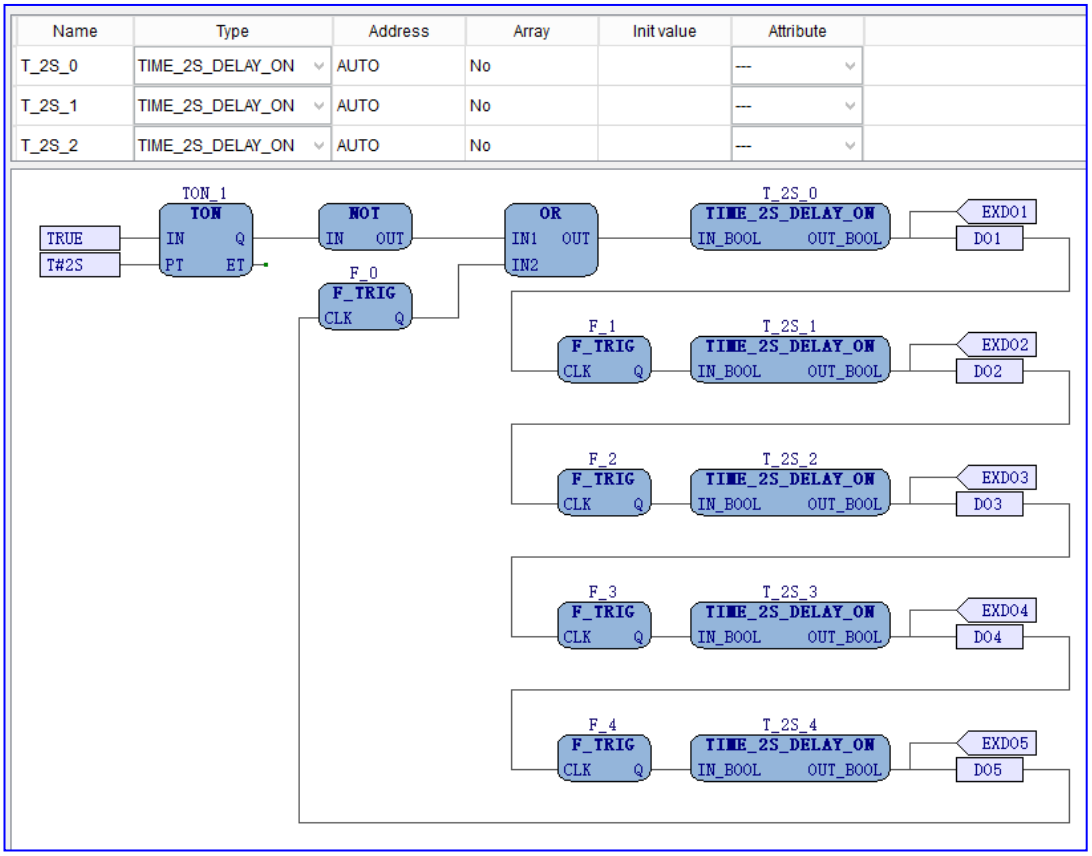
2. Right click on the project name to create a new project, function, function block, and more.

This example uses a delay subroutine, so create a new function block and encapsulate the delay subroutine within it. When the main program calls the delay subroutine, simply drag it into the program.

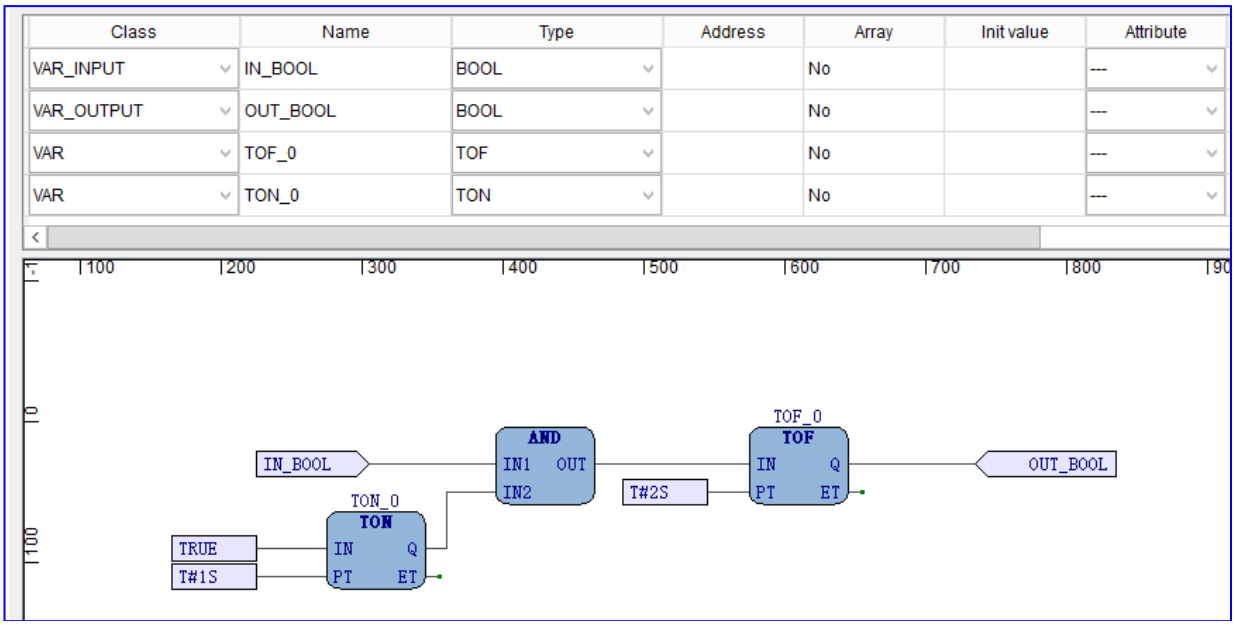




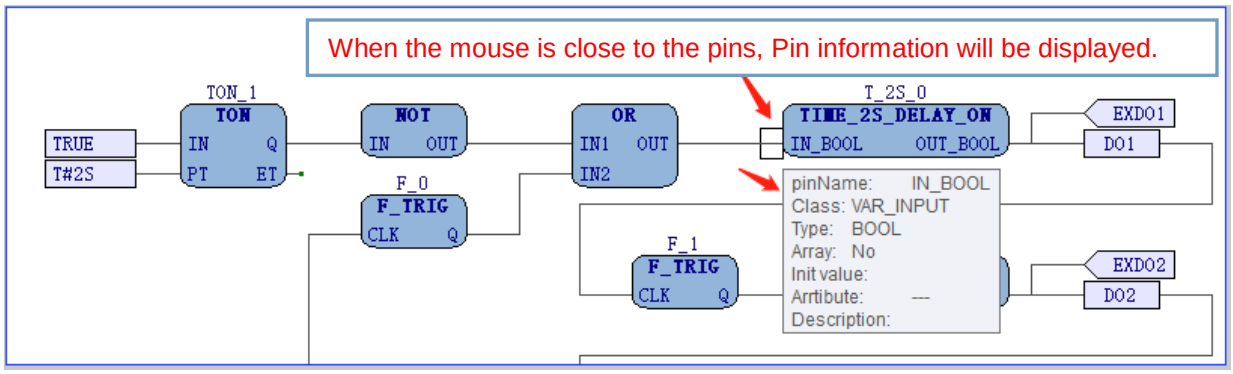
3. Program in the main program to output DO1~DO5 and EXDO0~EXDO5 every two seconds in sequence, thus achieving the function of a ticker.



The delay subroutine is as follows:

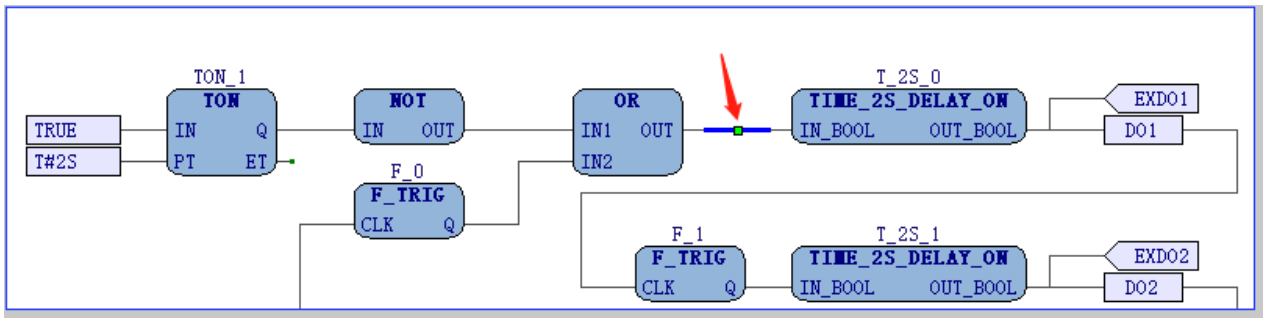


Tip 1: When the mouse is close to the pin, it will prompt for pin related information.



Tip 2: Create branch points directly online.

Place the mouse over the branch line, and a green icon will appear indicating that you can create a pivot point. Click on the green icon to drag the pivot point to the desired location.



3.6 SystemePLC Studio communication with PLC

SystemePLC Studio has two ways of communicating with the PLC, MODBUS TCP and MODBUS RTU.

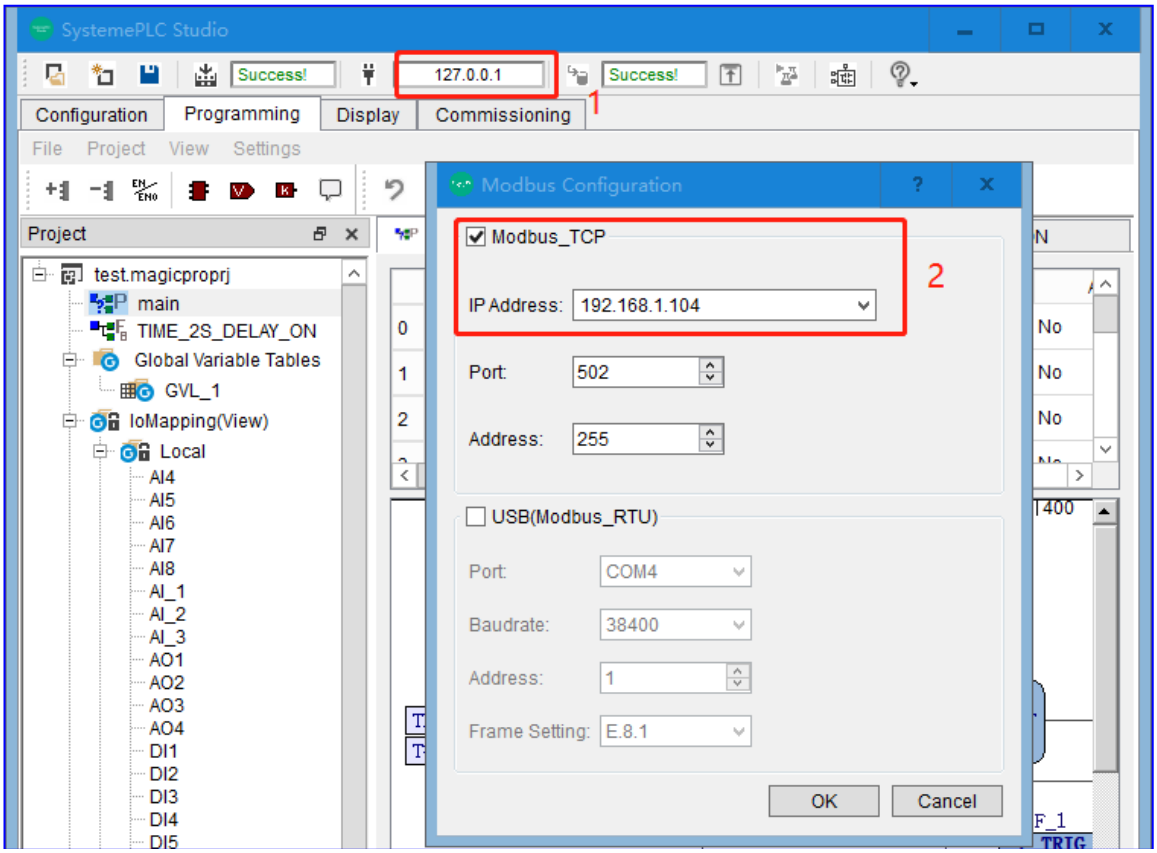
First, PLC communicates with SystemePLC Studio via MODBUS TCP.

1. Before setting the communication, you need to set the IP of the programming PC to the same network segment as the PLC.

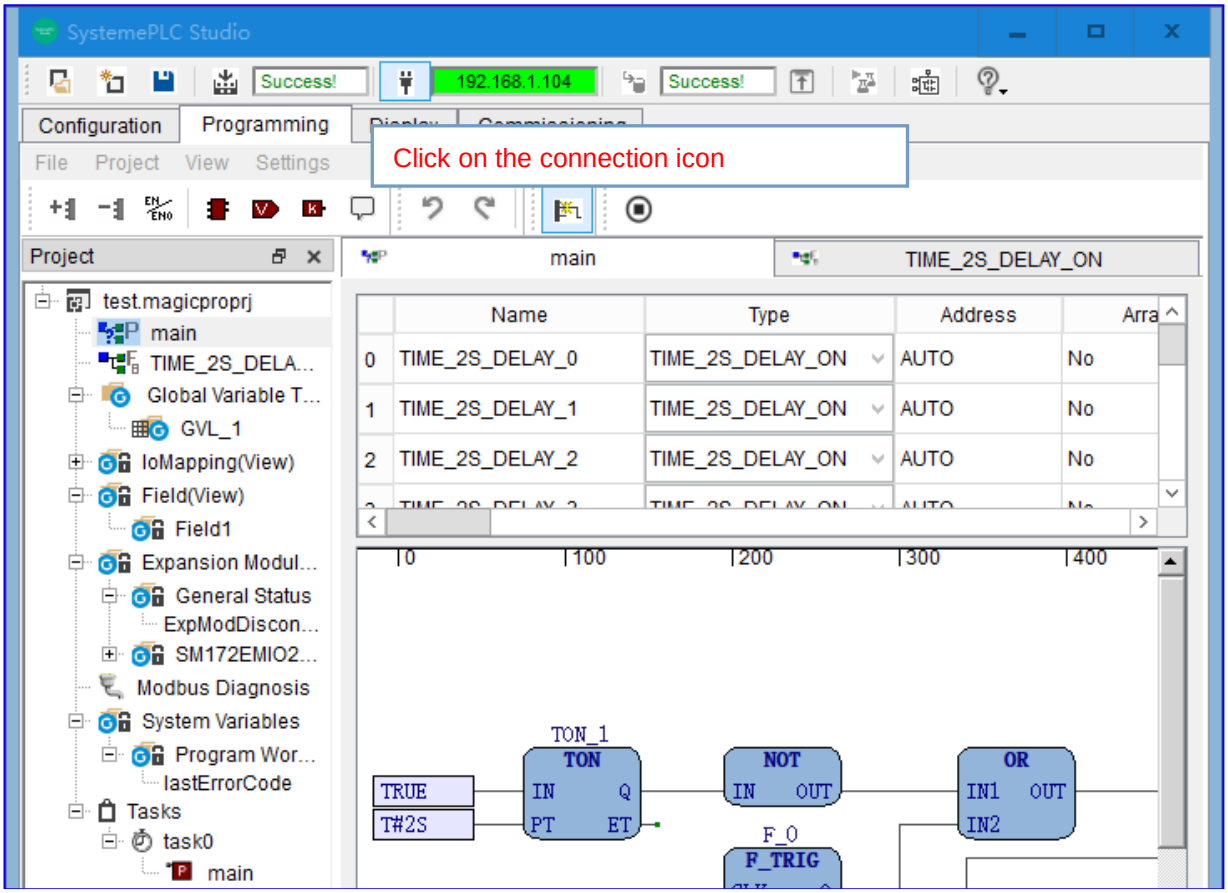
2. Click on the Connection Device dialog box in SystemePLC Studio and select MODBUS TCP, input PLC local

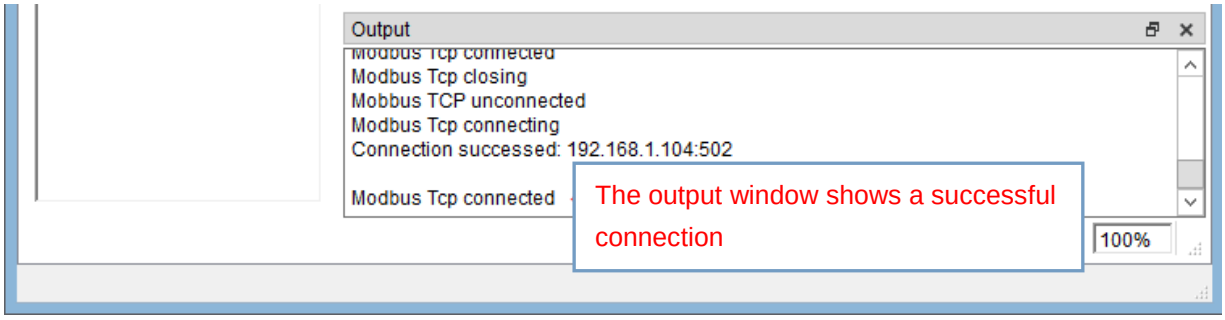
address.

<Note>Please refer to Chapter 7.1 Checking the IP address of the PLC for viewing the PLC local address.



3. Click on the connection icon , and after successful connection, the adjacent device box will display the IP address of the PLC and turn green. At the same time, the output window will also display the message of successful connection.

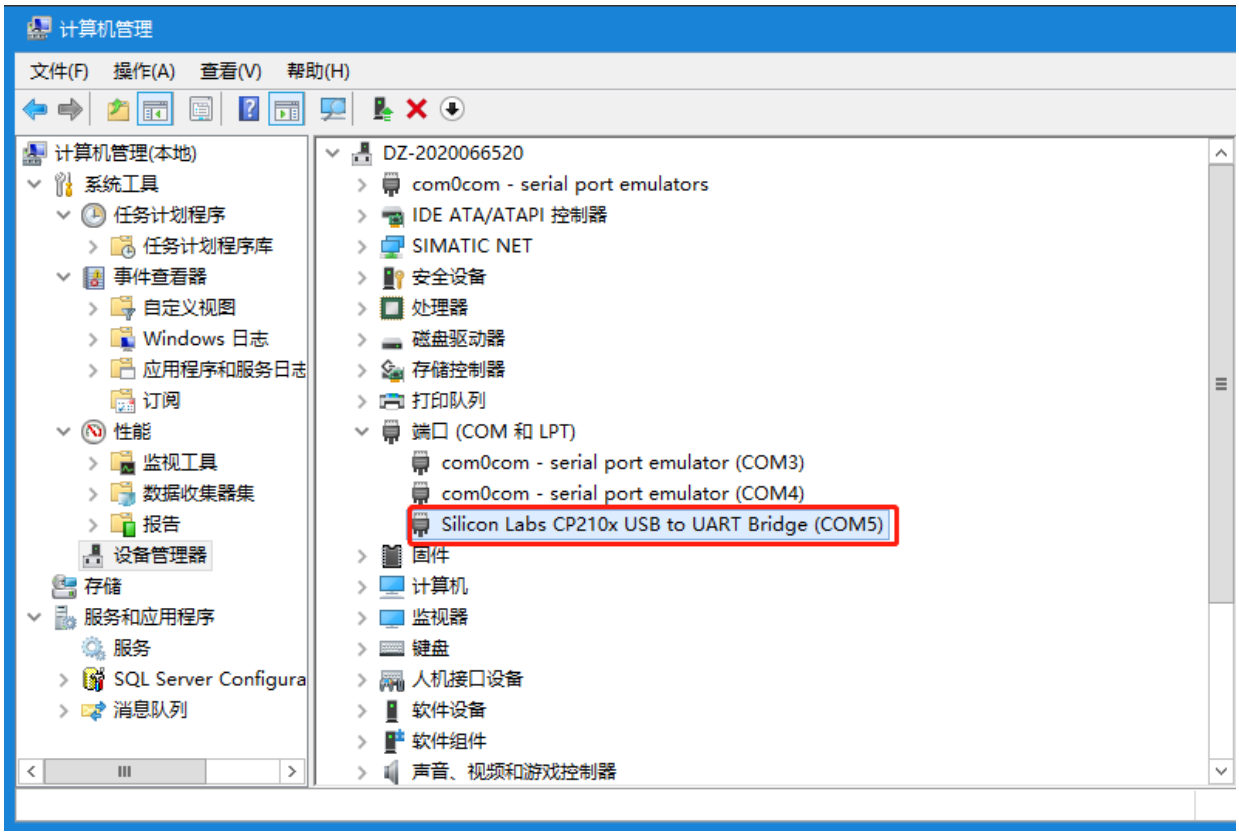




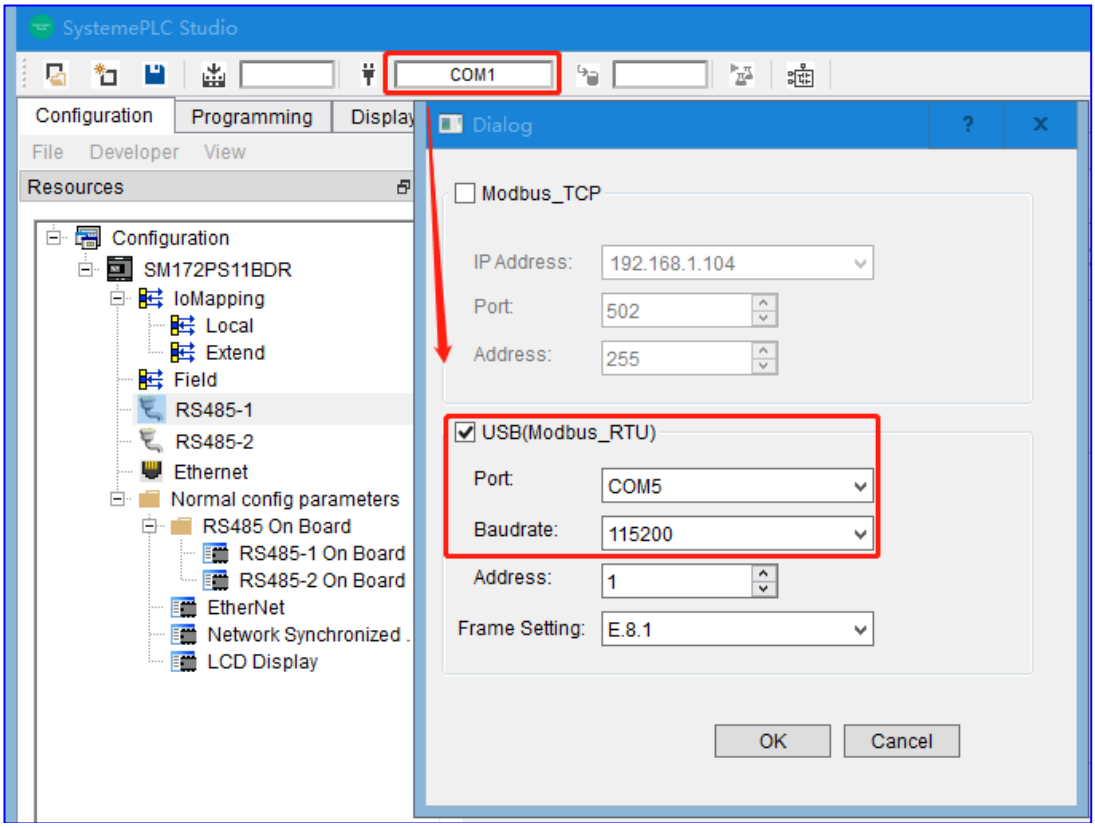
Second. PLC communicates with SystemePLC studio through Modbus RTU

1. Before configuring serial port communication, it is necessary to install the corresponding serial port driver on the computer. This is because the serial port devices produced by different manufacturers may need different drivers to carry out serial port communication. After installing the driver, the computer can detect the COM port of PLC.

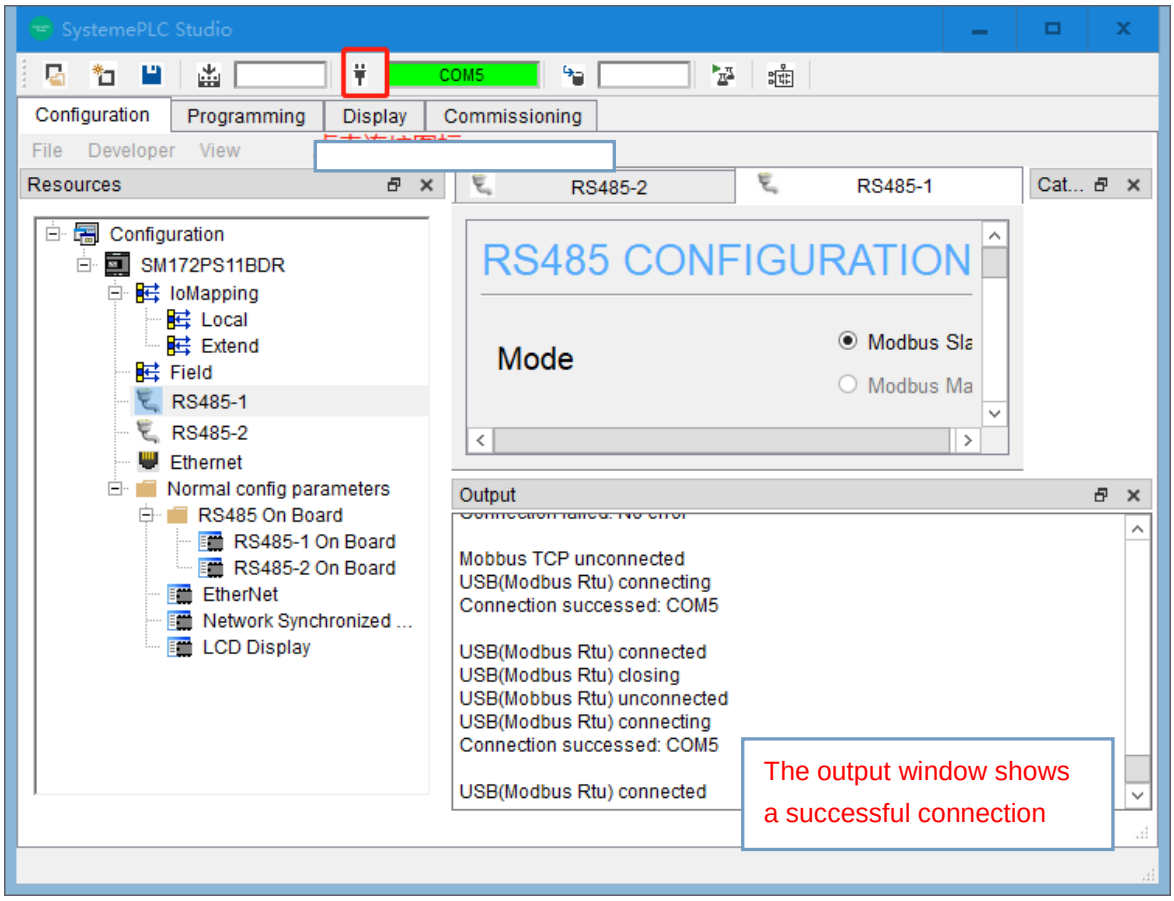
The following is the COM port of PLC detected by the computer.



2. Click the connect device dialog box of systemeplc studio, select Modbus RTU, and enter baud rate and com5 port. If you need to check the PLC baud rate, please refer to chapter 7.5 Checking the PLC baud rate.



3. Click on the connection icon , and after successful connection, the adjacent device box will display the PLC's COM port and turn green. At the same time, the output window will also display the message of successful connection.

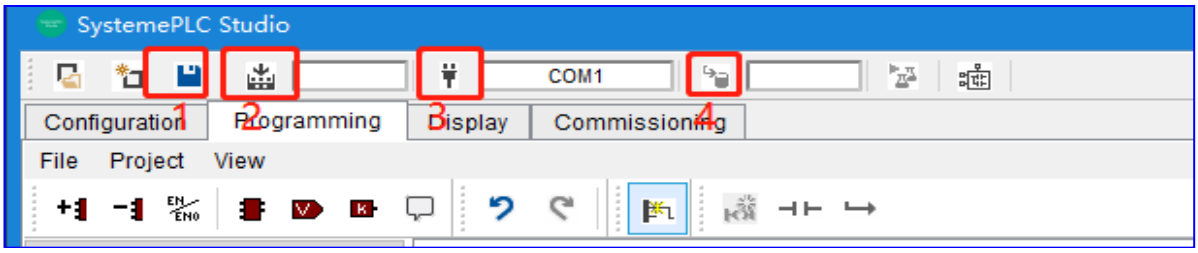


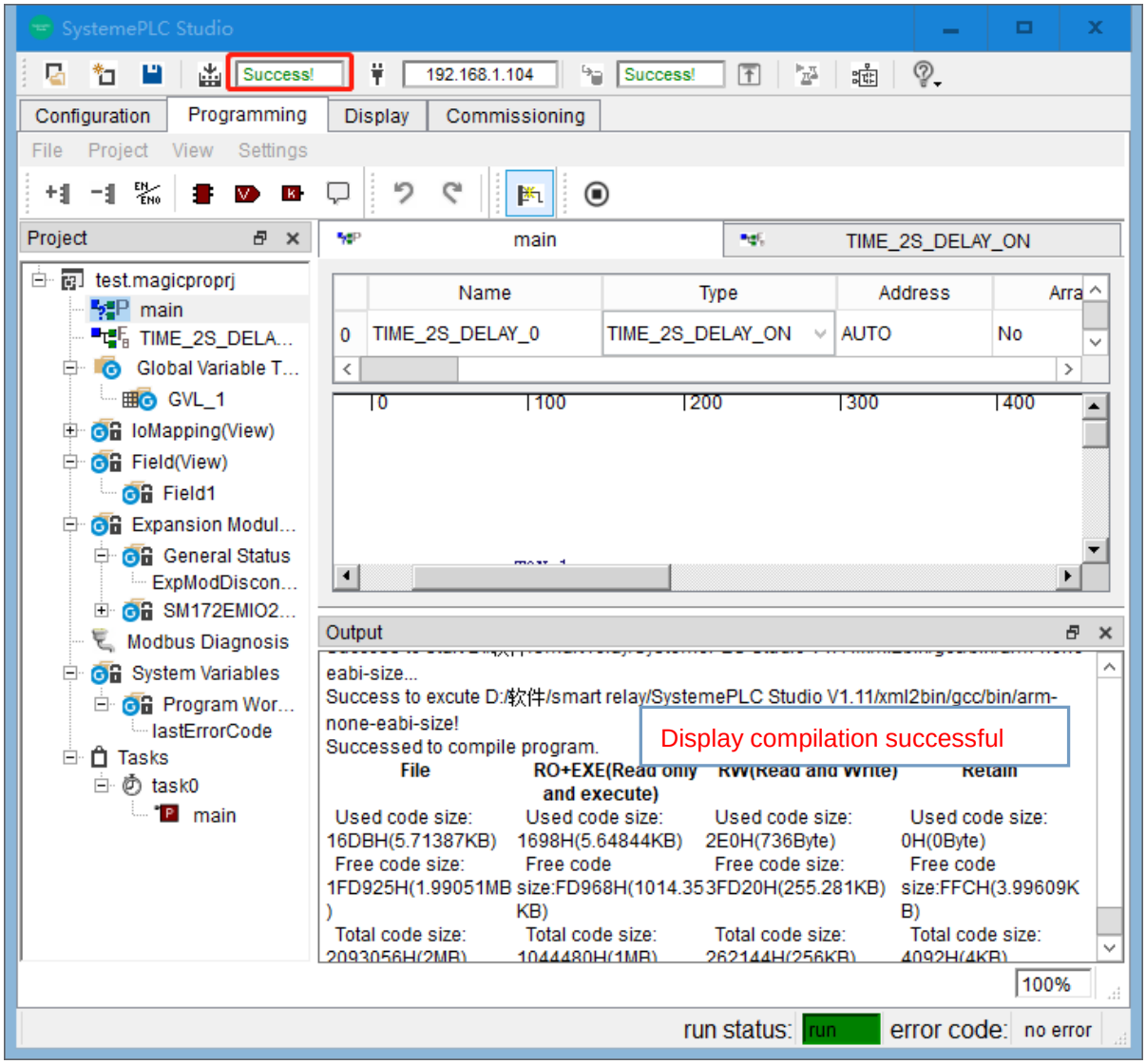
3.7 Program compilation and download

1. Save the current project and compile it

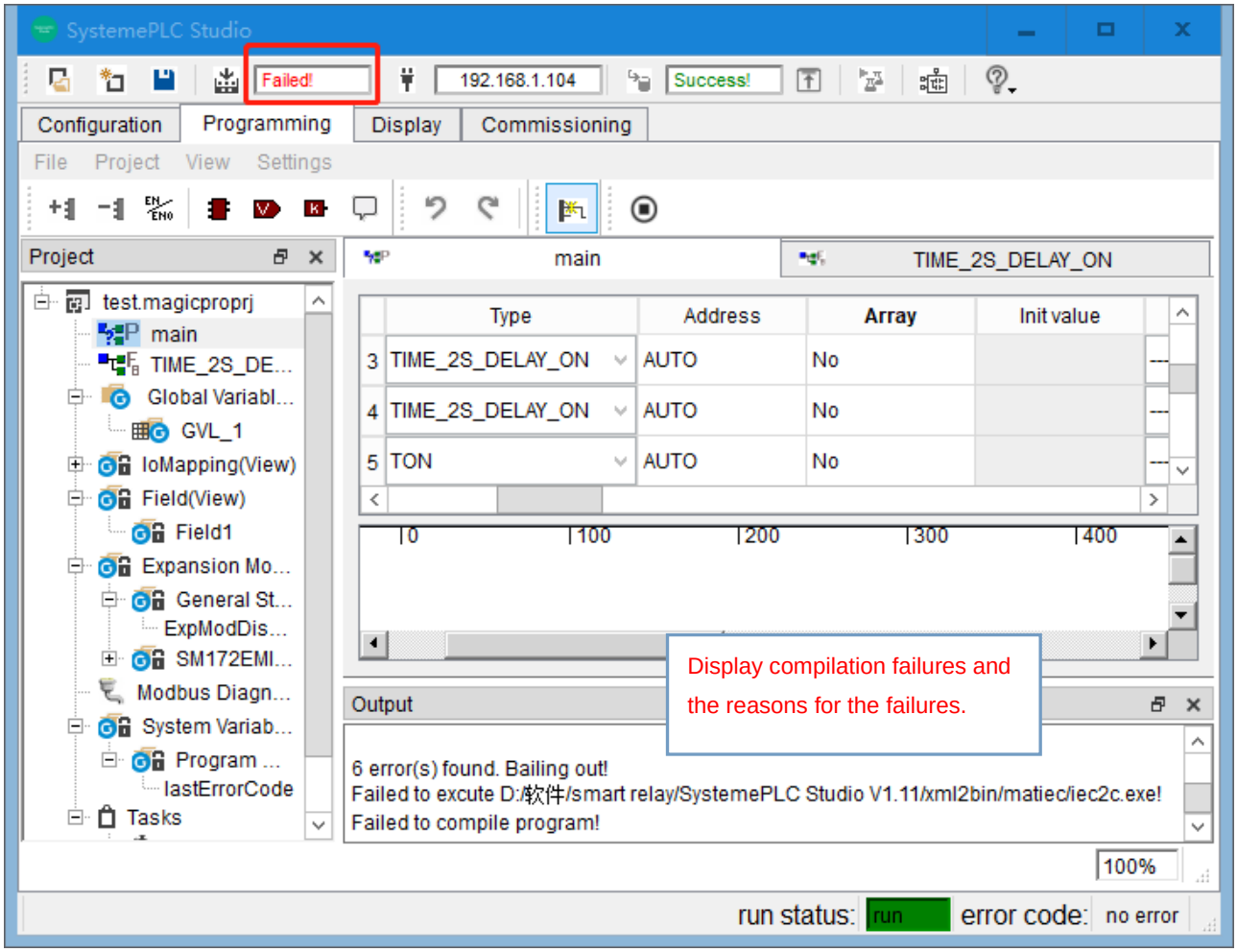
After the program is completed, save the current project, and then compile the program. The output window will display the compilation information. If there are no errors, it indicates successful compilation.

1-Save the project, 2- Compile, 3-Connect to the target device, 4-Download the program.



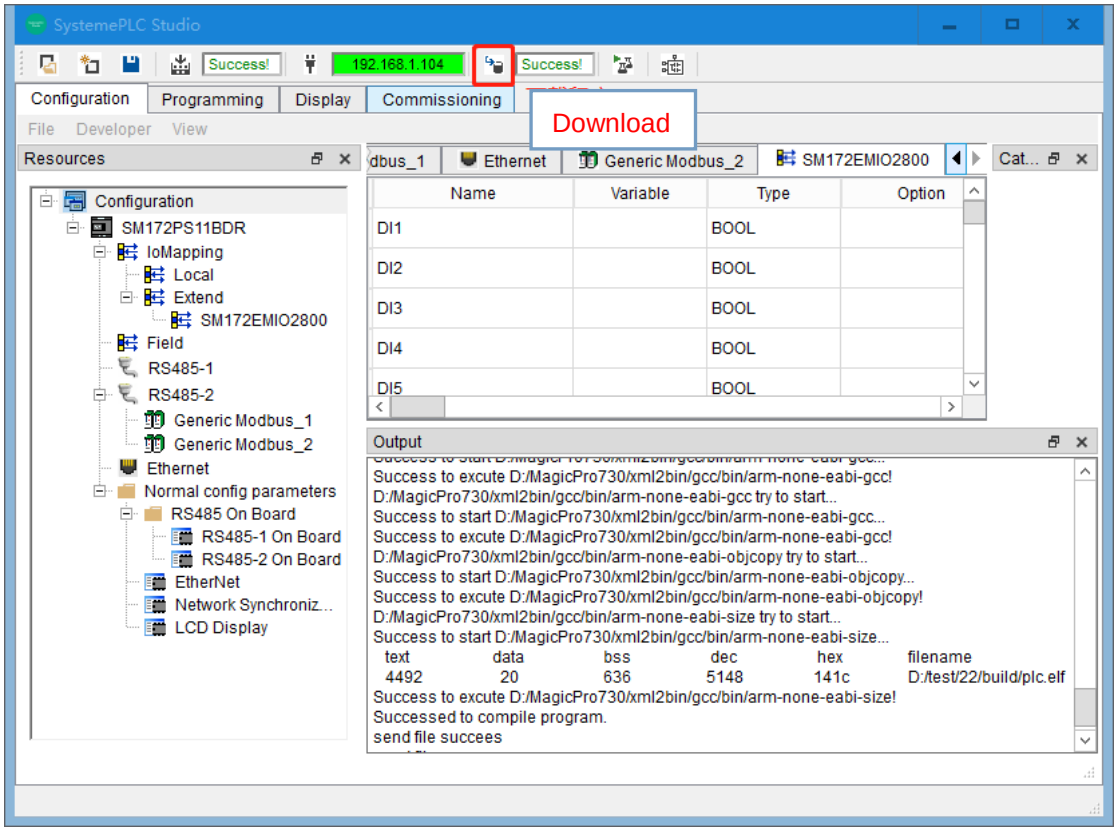


If there is an error during compilation, the output window will display the reason for the error.



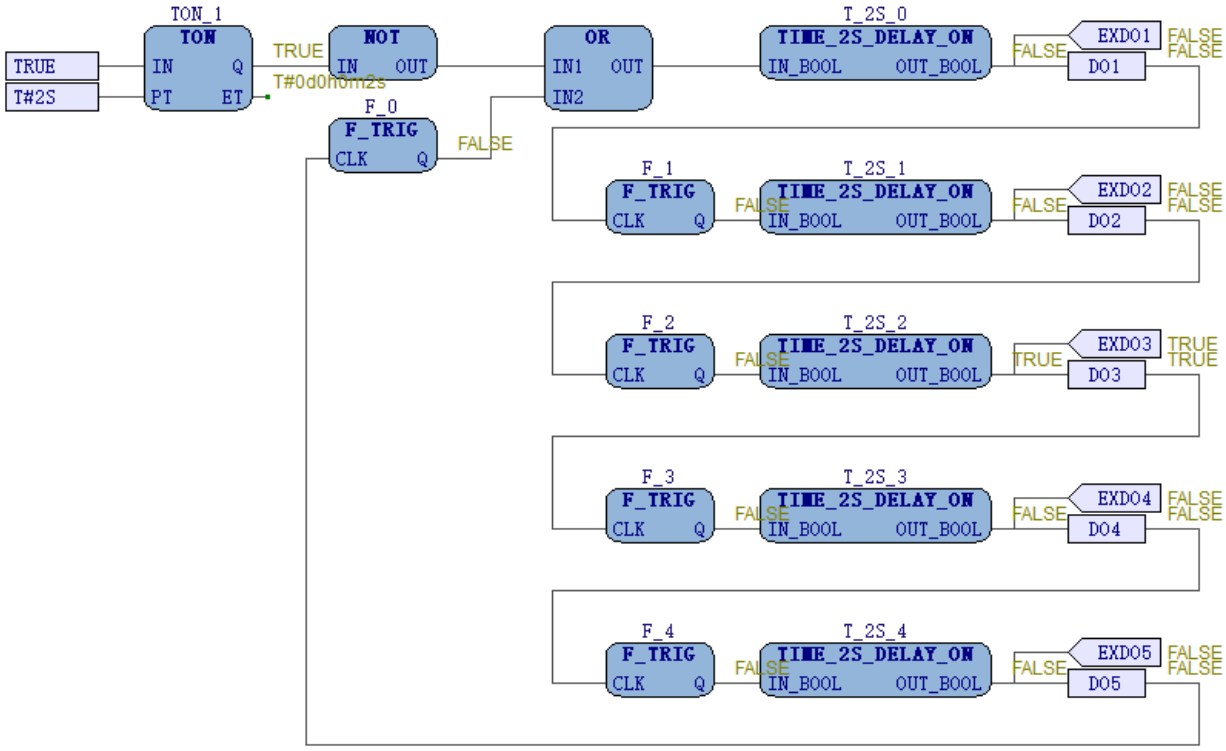
2. Download and run programs

Click the download icon to download the program to the PLC. After the download is successful, if the PLC is set to the auto run mode, the program will run directly. If the PLC is in the stop mode, enter the PLC operation menu on the display screen, and then set it to the auto run mode.

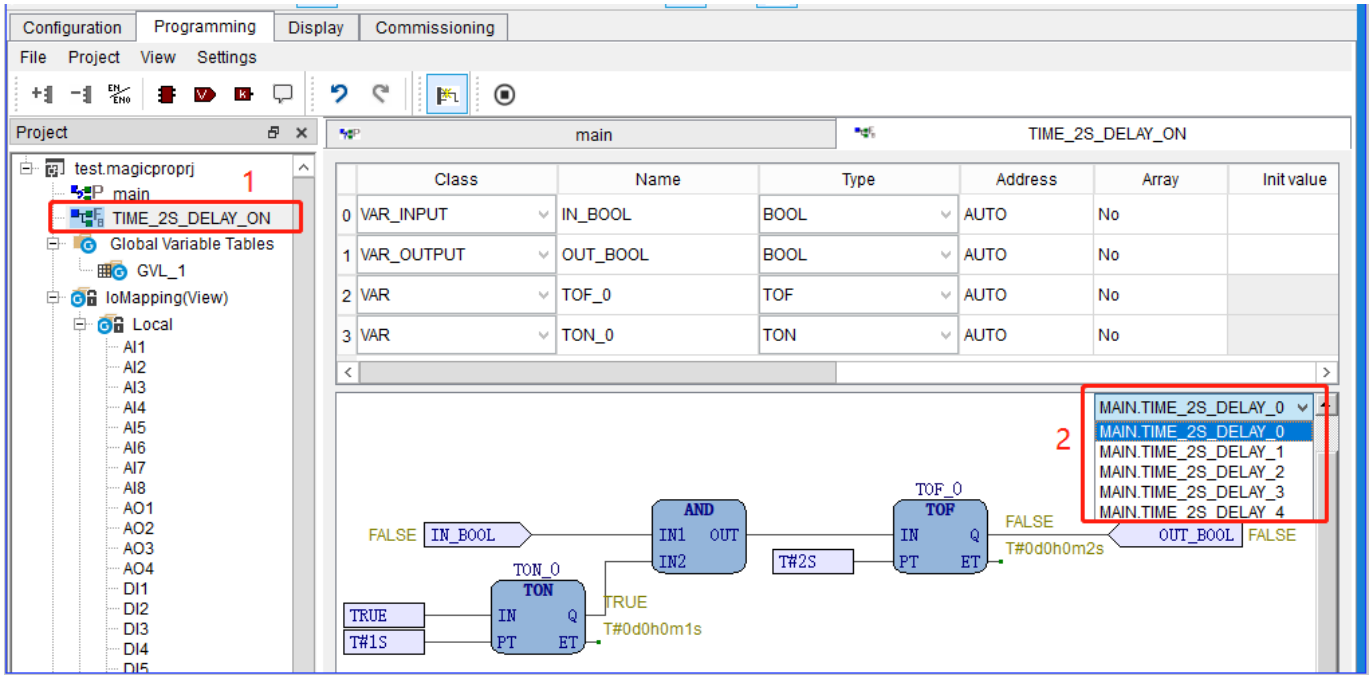


3.8 Monitoring

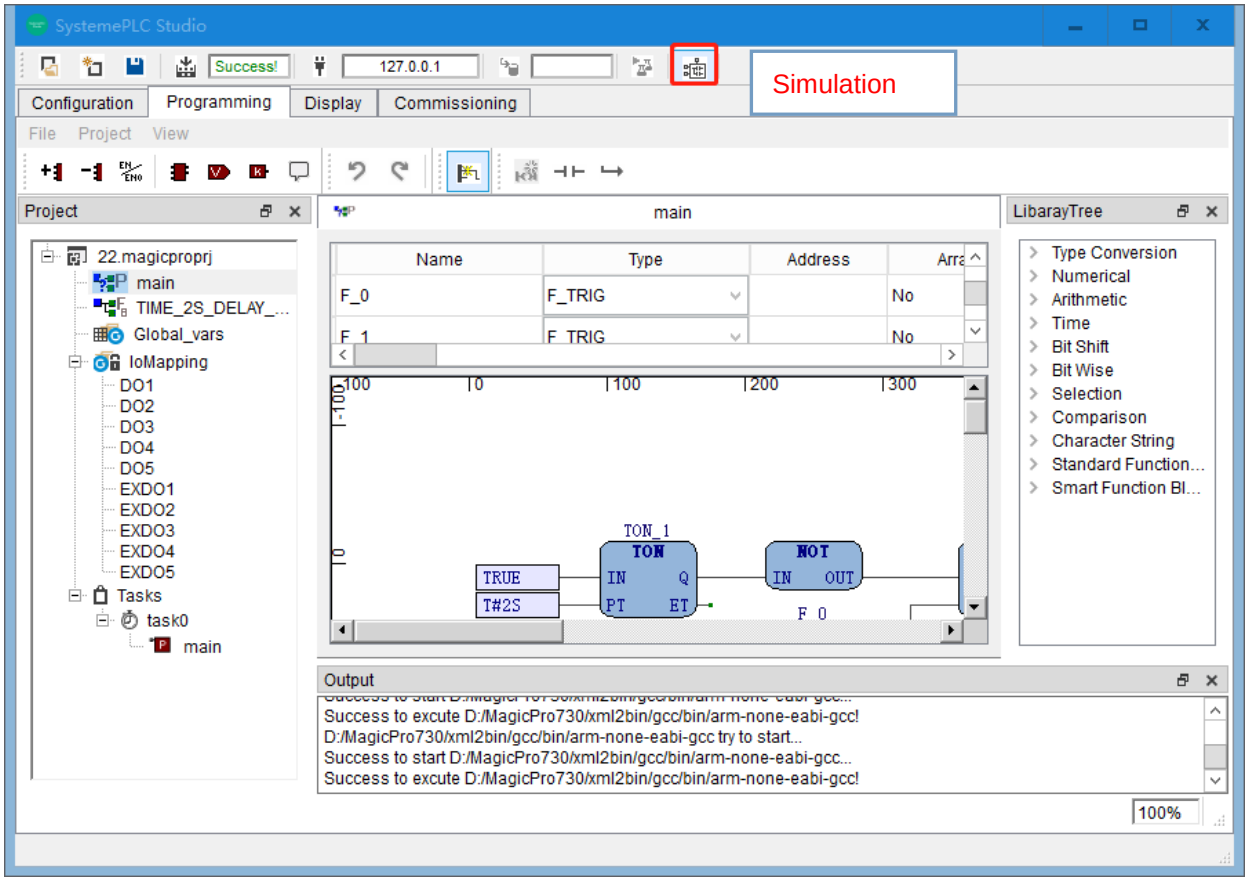
Click on the online monitoring icon, and the program will monitor as follows:



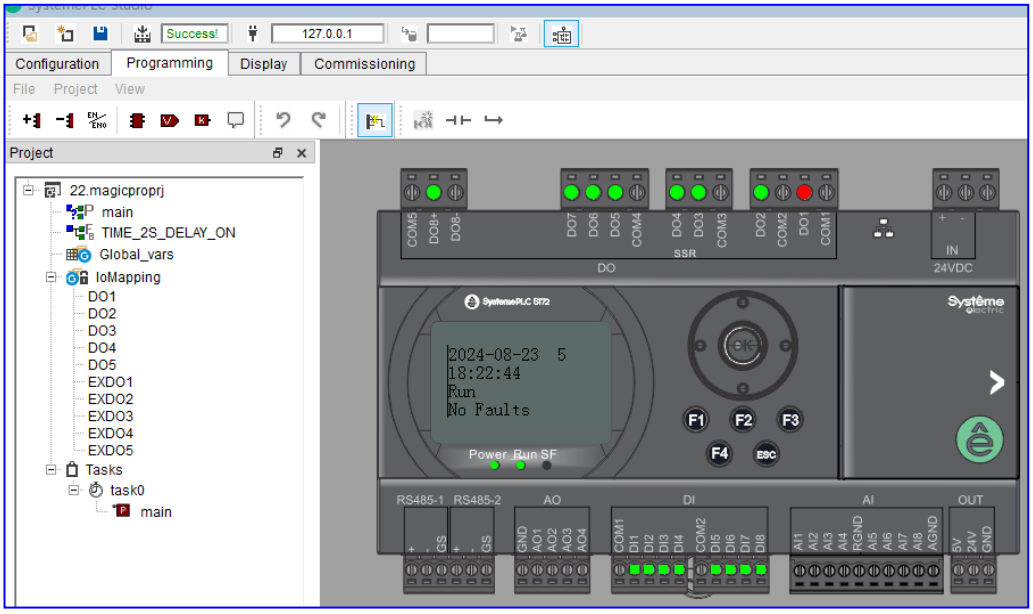
In the monitoring mode, select the corresponding FB instance to monitor each FB individually.

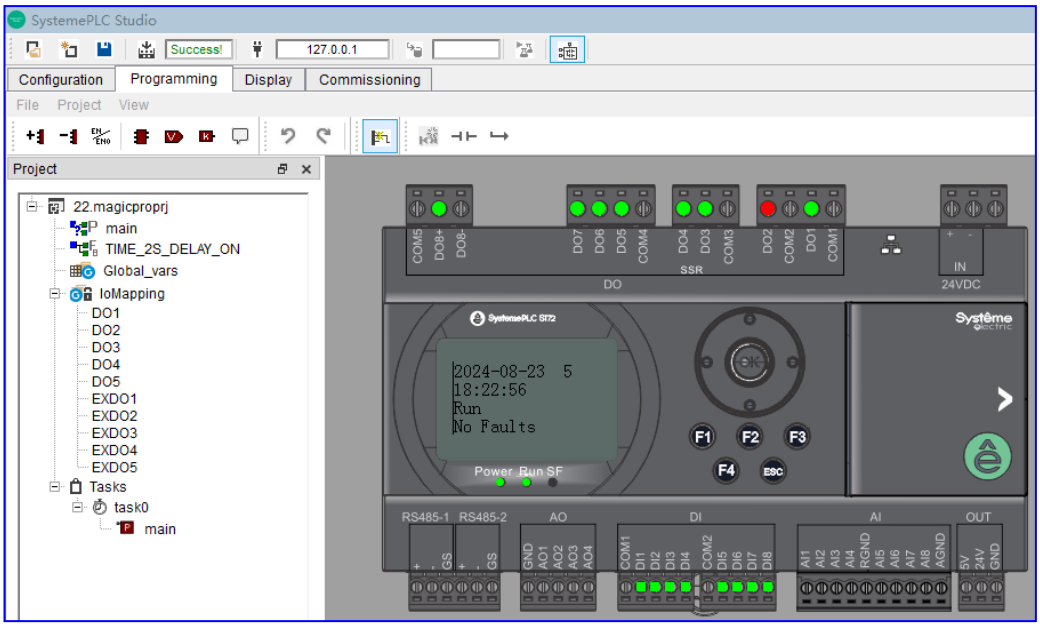


Program functionality can also be achieved through simulation

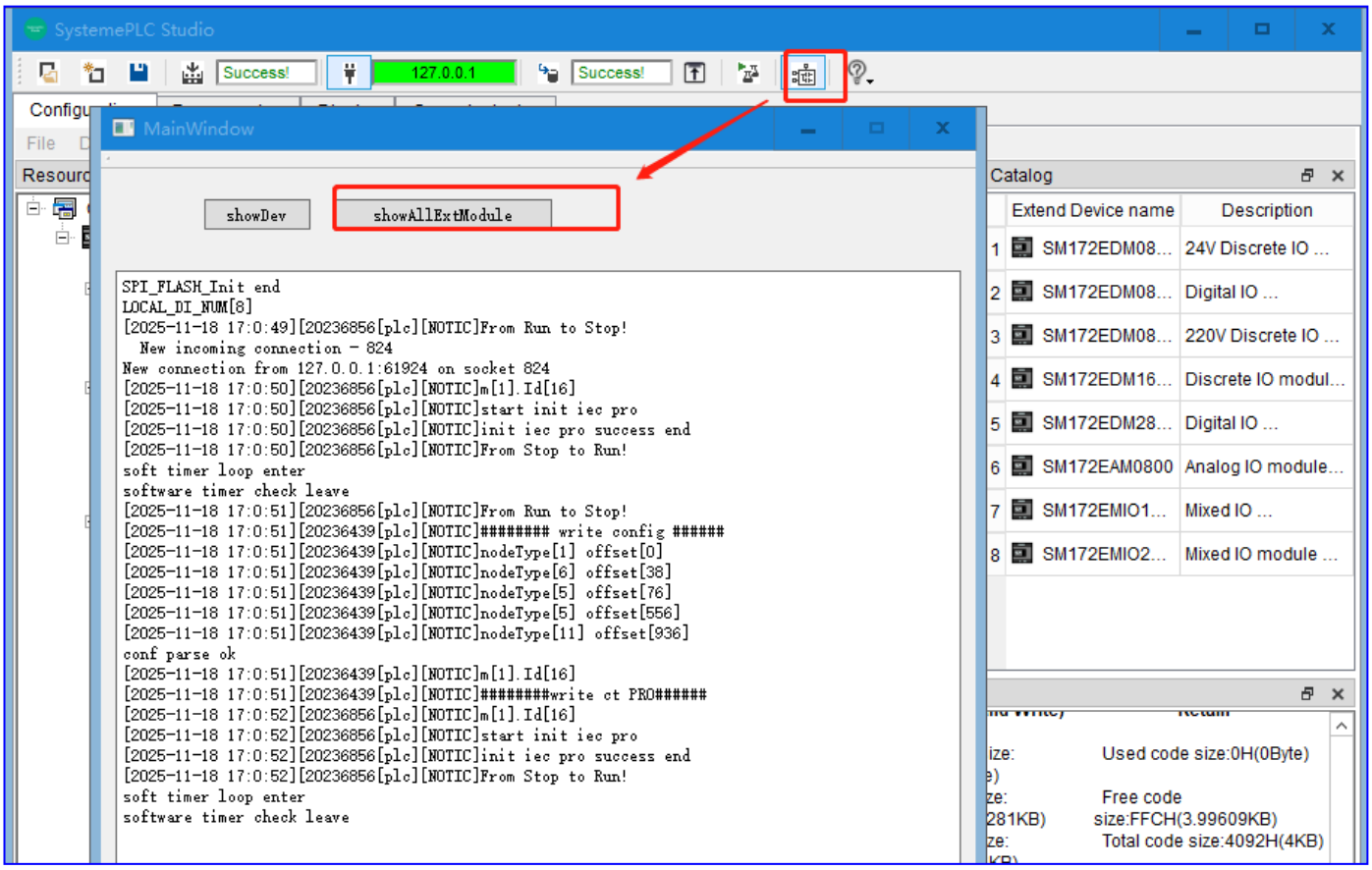


The simulation results are as follows: Every 2 seconds, output DO1~DO5 sequentially.





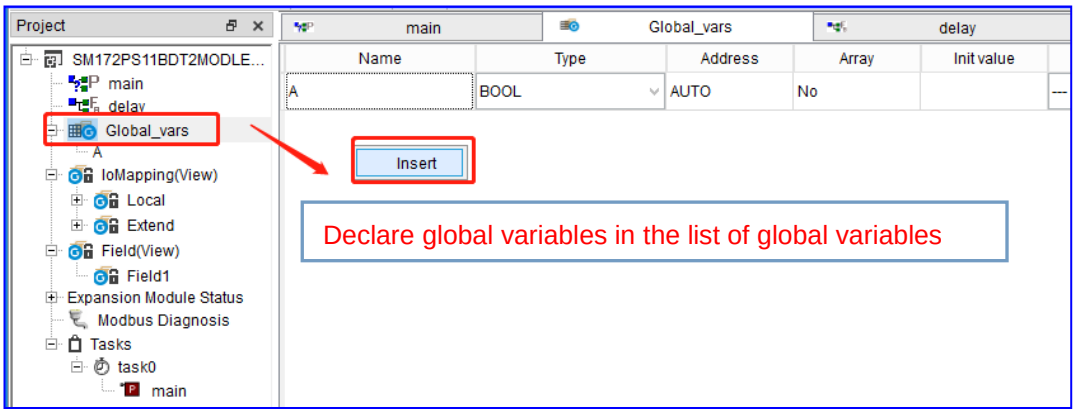
During simulation, click "showALLExtModule" to view the expansion modules on the corresponding nodes.



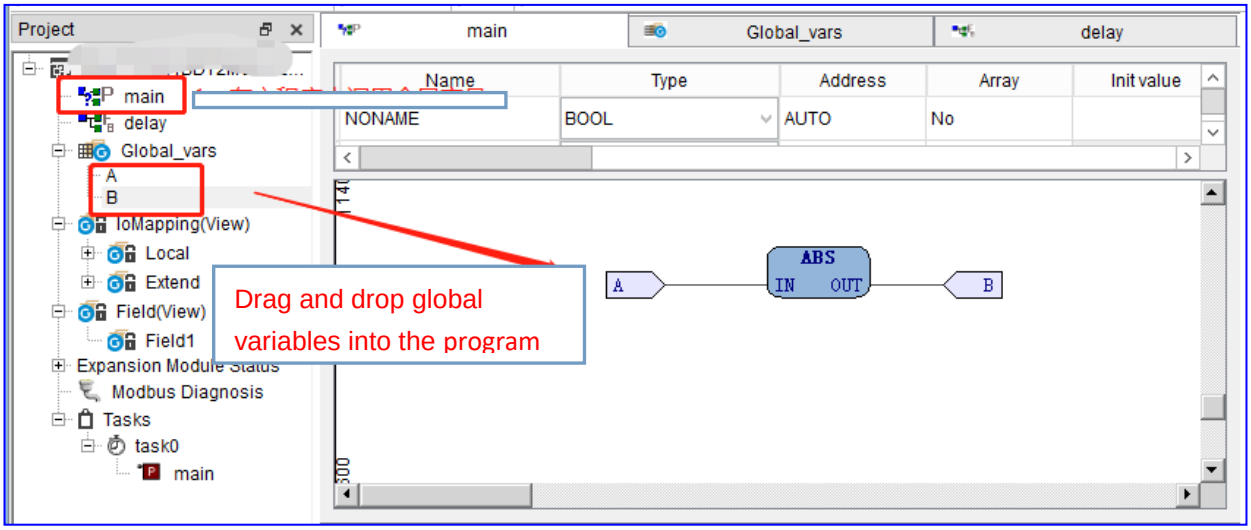
3.9 Global variables

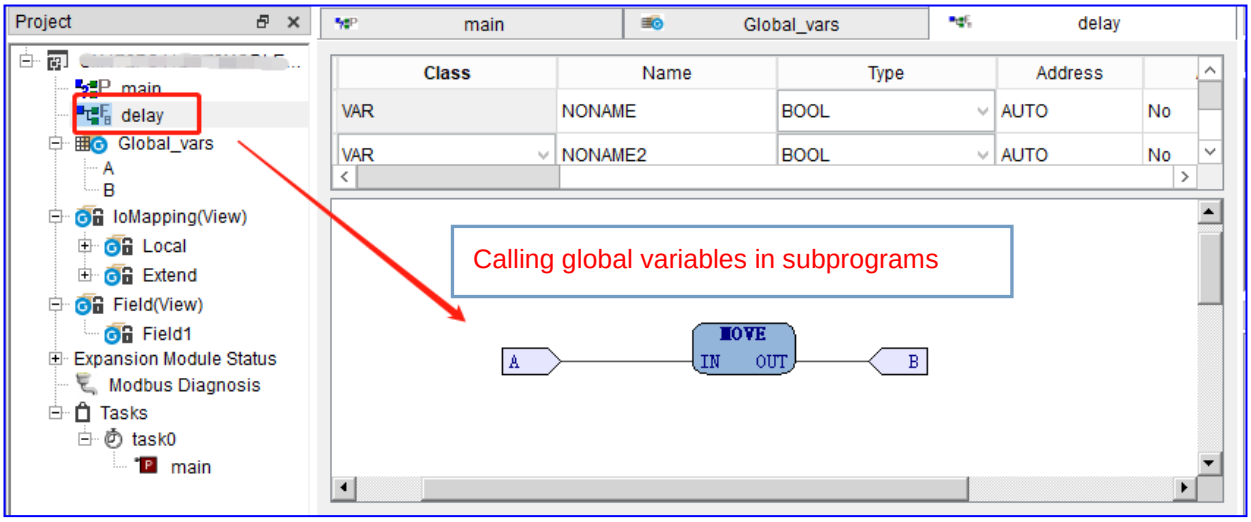
Global variables are variables that can be identified throughout the entire project. Declare global variables in the declaration section of the global variable list.

1. Declare global variables



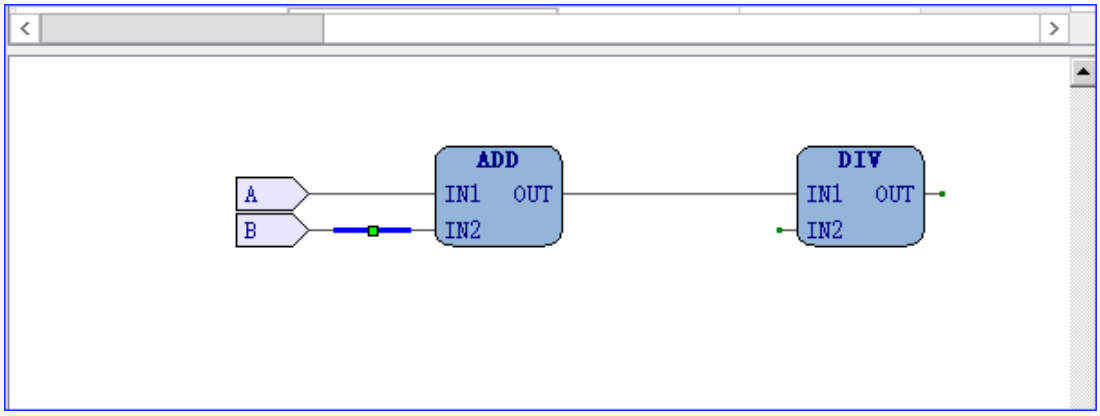
2. Global variables can be called in any program

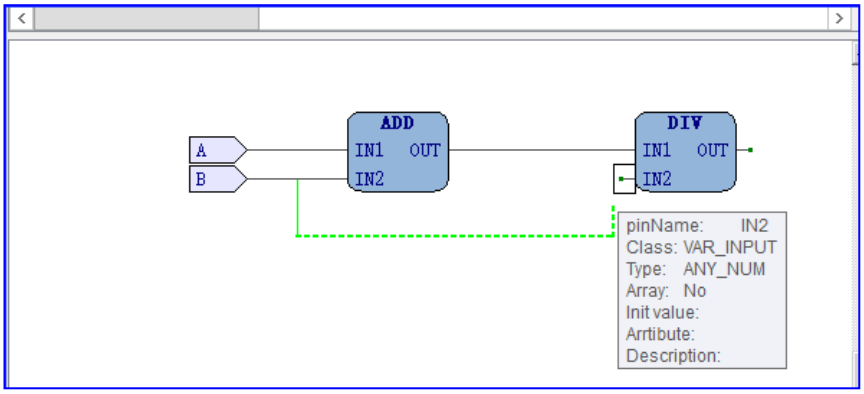


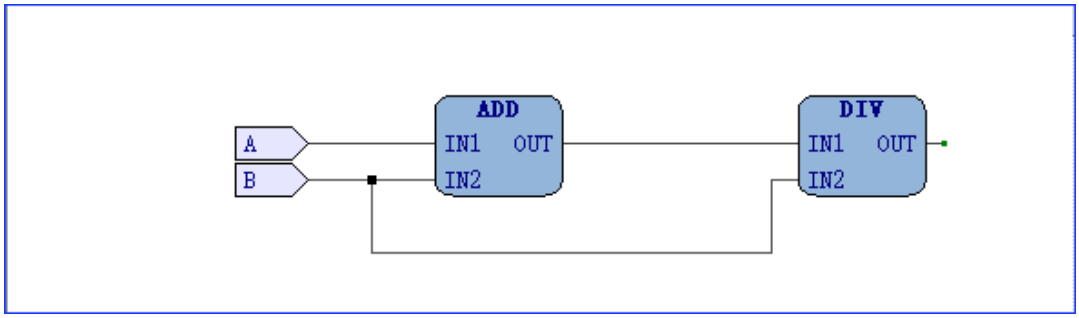


3.10 Use of branchers

Place the mouse on the branch line at the beginning of the branch until a green dot appears. Click on the green dot to connect it to other branches or inputs/outputs.





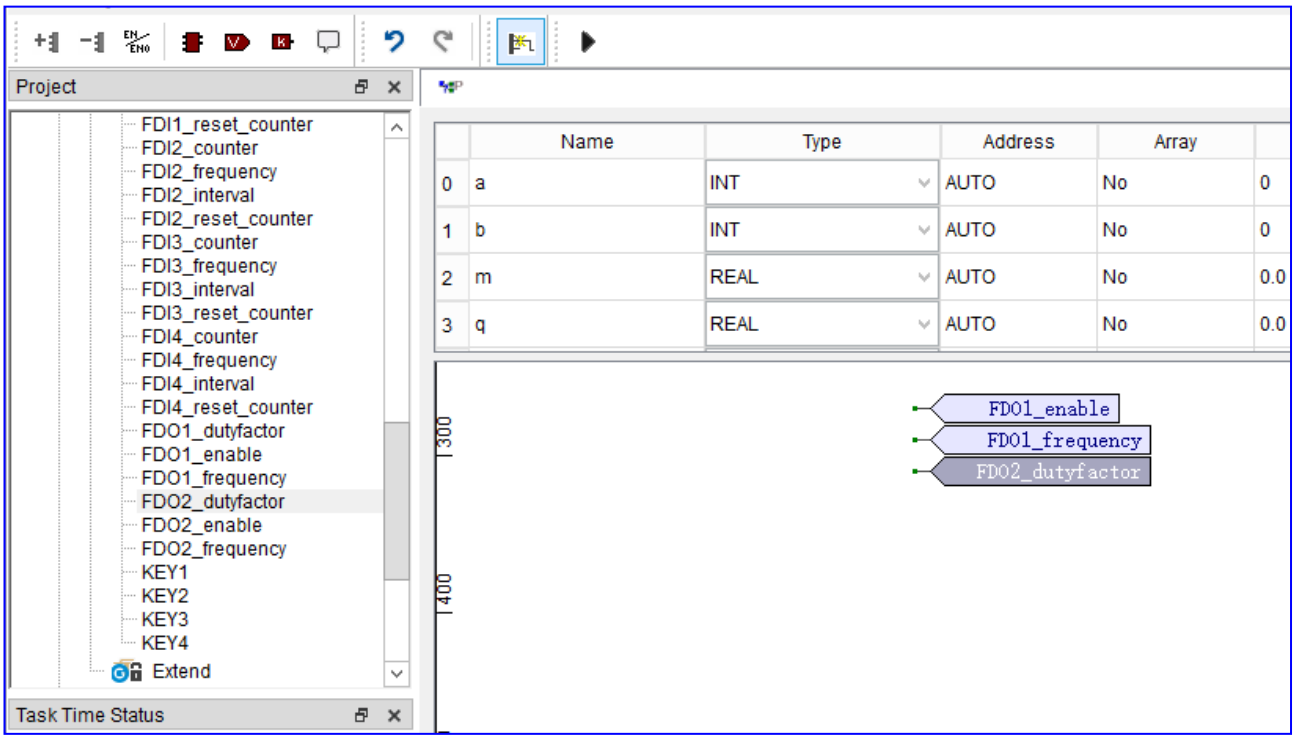


3.11 Use of FDO

The DO1~DO2 of SM172PS11BDT belong to fast output. Users can set the frequency and duty cycle for output through DO1~DO2. The following represents the attribute description of FDO1~FDO2.

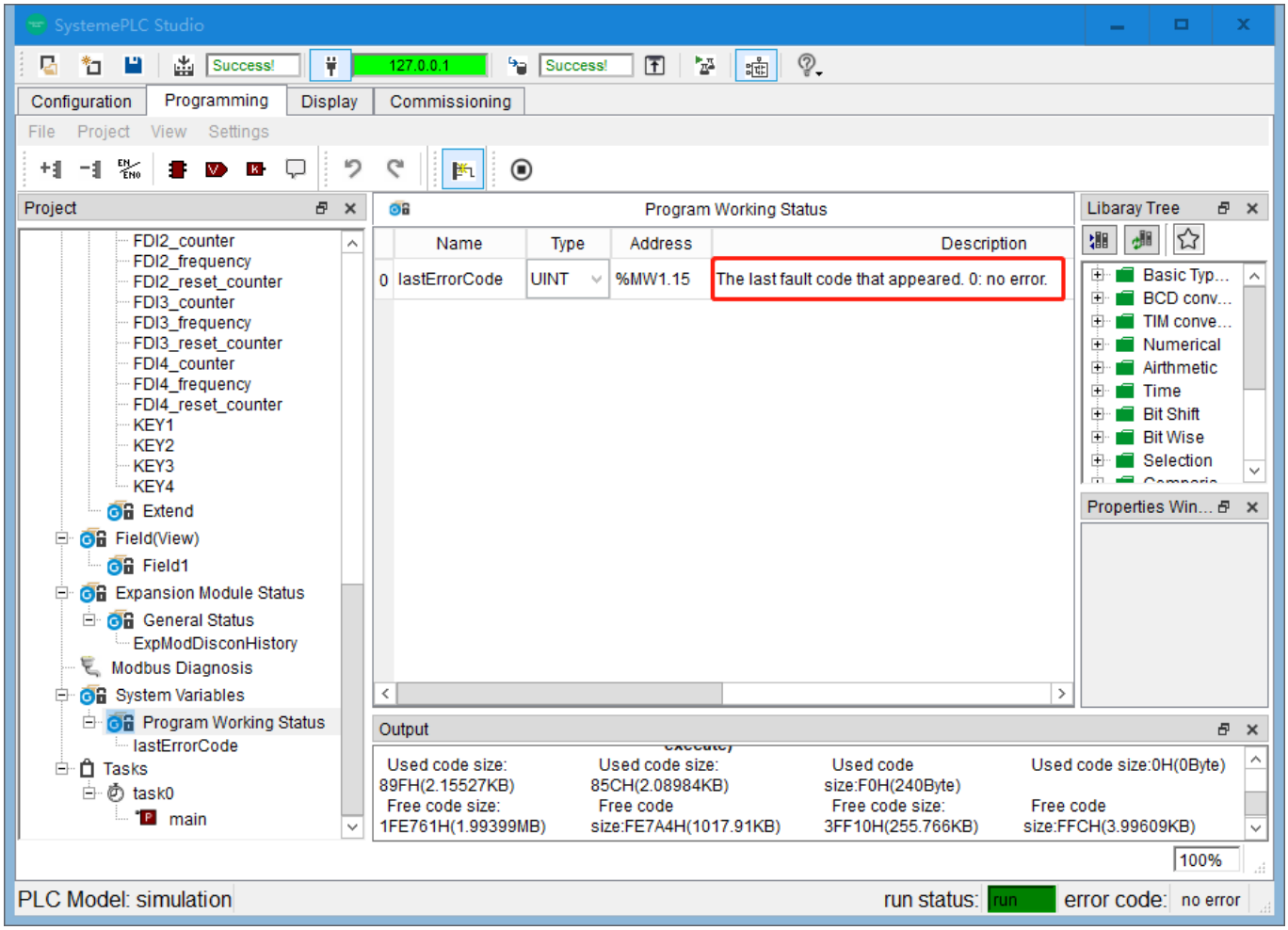
Name	Variable	Type	Option	Default Value	Min Value	Max Value	Address	Modbus Address	Description
FDI2_frequency		UDINT			0	0	%MD1...	3x0603	FDI2 frequency display
FDI2_reset_counter		BOOL		0	0	1	%MX6.1	0x0502	FDI2 reset input counter value
FDI2_interval		UINT		1000	10	1000	%MW...	4x0502	FDI2 sampling time interval of frequency(ms)
FDI3_counter		UDINT			0	0	%MD5.2	3x0505	FDI3 input counter
FDI3_frequency		UDINT			0	0	%MD1...	3x0605	FDI3 frequency display
FDI3_reset_counter		BOOL		0	0	1	%MX6.2	0x0503	FDI3 reset input counter value
FDI3_interval		UINT		1000	10	1000	%MW...	4x0503	FDI3 sampling time interval of frequency(ms)
FDI4_counter		UDINT			0	0	%MD5.3	3x0507	FDI4 input counter
FDI4_frequency		UDINT			0	0	%MD1...	3x0607	FDI4 frequency display
FDI4_reset_counter		BOOL		0	0	1	%MX6.3	0x0504	FDI4 reset input counter value
FDI4_interval		UINT		1000	10	1000	%MW...	4x0504	FDI4 sampling time interval of frequency(ms)
FDO1_enable		BOOL		0	0	1	%MX11.0	0x0551	FDO1 enables pulse output
FDO1_frequency		UINT		10000	30	30000	%MW...	4x0551	FDO1 pulse output frequency
FDO1_dutyfactor		UINT		50	0	100	%MW...	4x0561	FDO1 pulse output dutyfactor
FDO2_enable		BOOL		0	0	1	%MX11.1	0x0552	FDO2 enables pulse output
FDO2_frequency		UINT		10000	30	30000	%MW...	4x0552	FDO2 pulse output frequency
FDO2_dutyfactor		UINT		50	0	100	%MW...	4x0562	FDO2 pulse output dutyfactor
KEY1		BOOL			0	1	%MX2.4	1x0604	KEY1 F1 button(digital input)
KEY2		BOOL			0	1	%MX2.5	1x0605	KEY2 F2 button(digital input)
KEY3		BOOL			0	1	%MX2.6	1x0606	KEY3 F3 button(digital input)
KEY4		BOOL			0	1	%MX2.7	1x0607	KEY4 F4 button(digital input)

Simply drag the FDO onto the programming interface in programming to call it.



3.12 Program working status

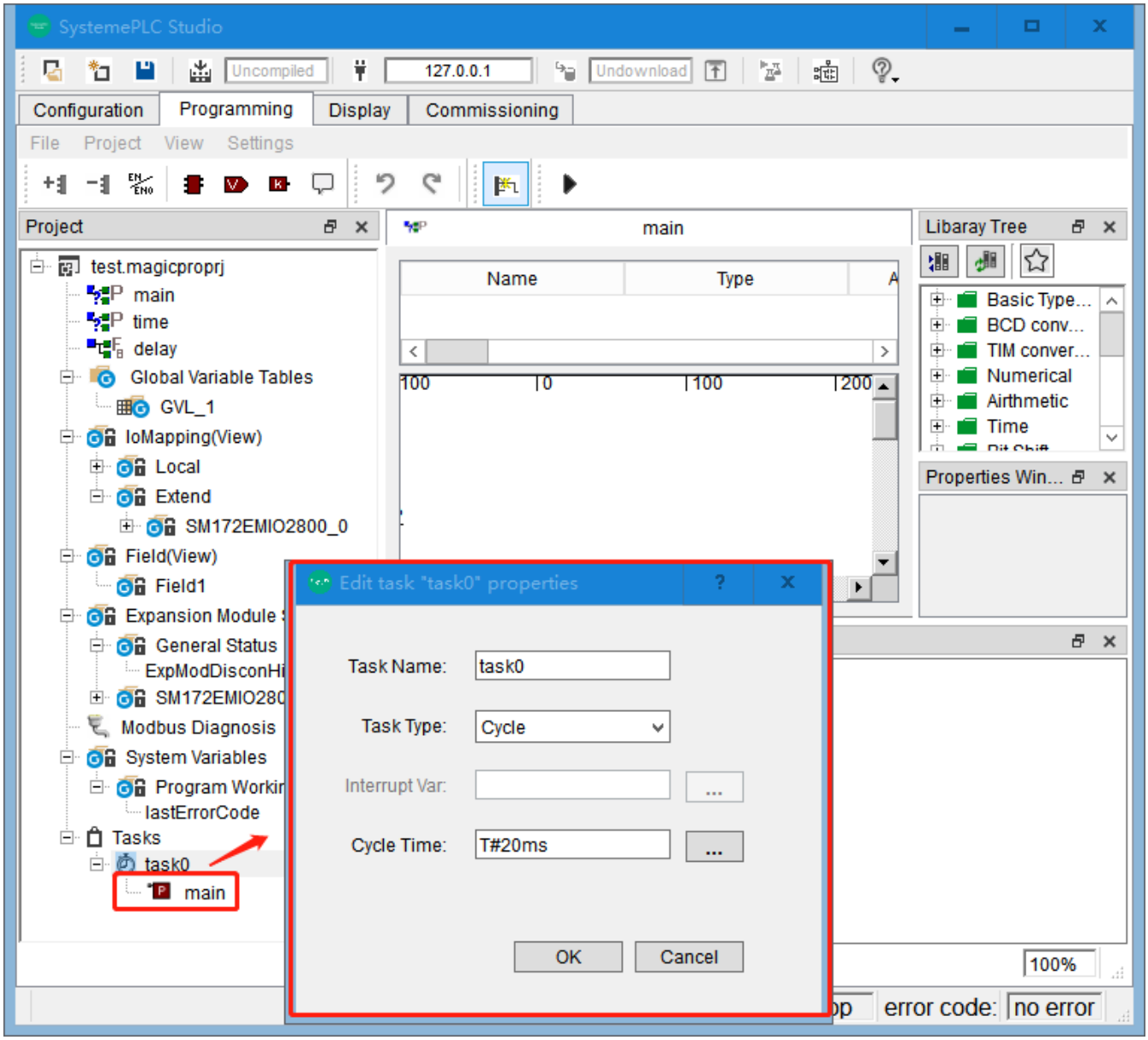
The last fault code that appeared. 0: no error. For specific fault codes, refer to Chapter 6 Fault.



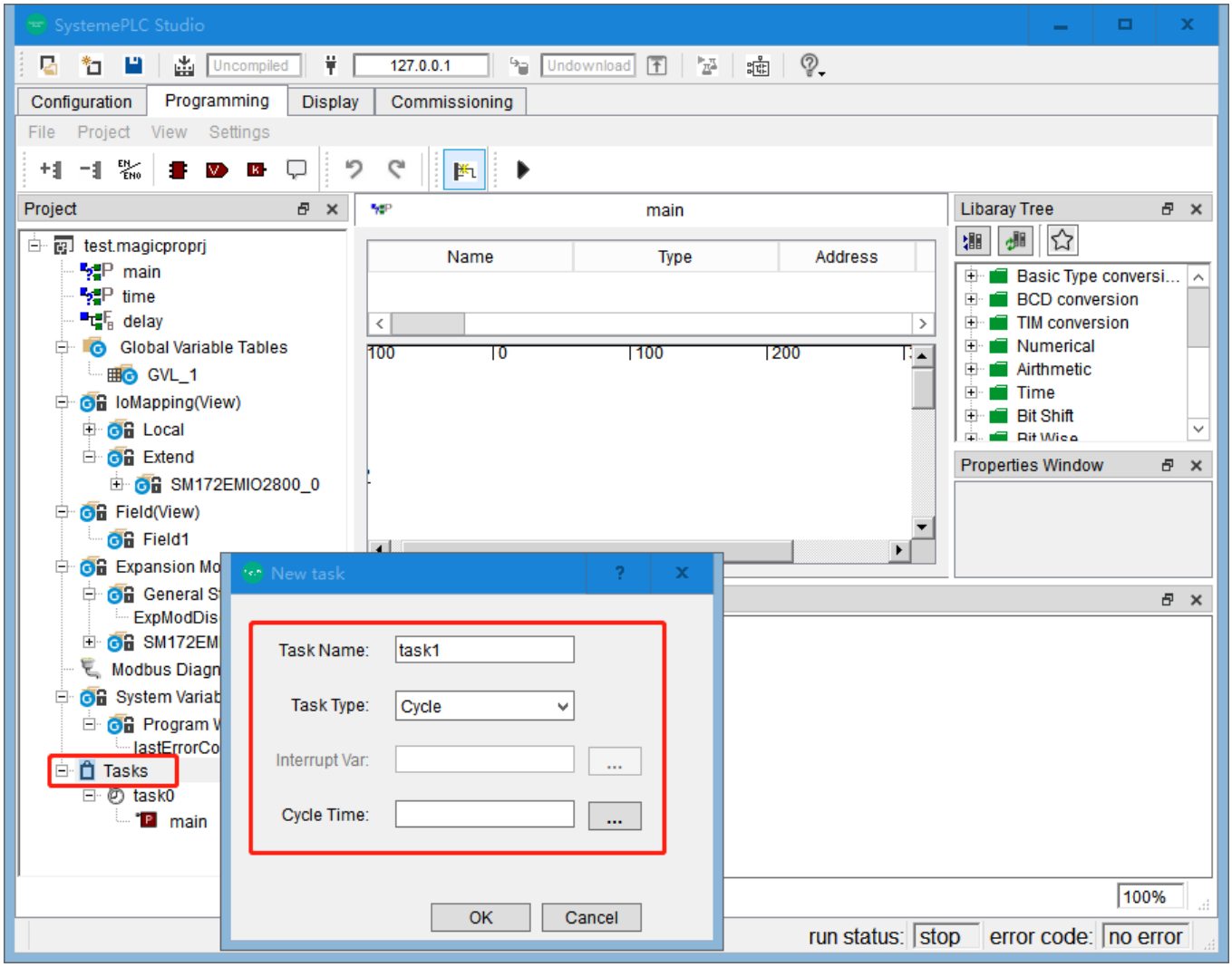
3.13 Task

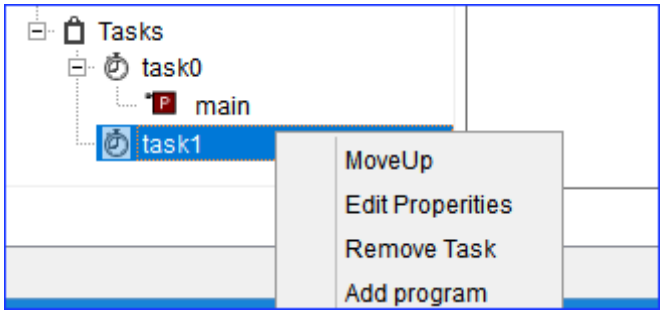
3.13.1 **Task configuration**

The Task in SystemePLC Studio is used to define and display the basic settings of tasks, ensuring the program runs in accordance with the specified requirements. When a new standard PLC project is created, a cyclically executed task is automatically generated; this task is automatically associated with "main", and the default task cycle is 20ms. A PLC program will only participate in compilation and actual execution if it is called by a task. Right-click Task0 → "Edit Properties" to configure the task type as cyclic or interrupt type. When configuring it as an interrupt task, an interrupt variable must be selected to trigger the execution of the interrupt program.



Multiple tasks can be configured to call different programs, and a maximum of 30 tasks can be created in one project. Right-click Tasks → New Task, then define the task name to complete the creation of a new task. Multiple tasks are executed from top to bottom in the order specified in the task configuration. After creating a new task, you can right-click the newly created task → Add Program, which means assigning the program to the task.





3.13.2 Task configuration principle

Step 1. Initialization and Configuration

The system performs hardware diagnosis and reads configuration parameters to ensure the device is in the correct state. Check and set the operating mode (RUN/STOP mode), and update the corresponding operation instructions according to the switched mode.

Step 2. Task List Management

The system maintains a task list (taskList), where each task includes its trigger condition and execution function. Tasks have two modes: Cycle (mode=0) and Interrupt (mode=1).

Step 3. Main Loop Execution Process

In RUN mode, enter the task processing logic. Update the actual hardware input buffer data to the I and M areas to

provide the latest status for task execution.

Step 4. Task Triggering and Filtering

Check whether the trigger condition of each task is met, and mark all tasks that meet the conditions.For time-triggered tasks, calculate the difference between the current time and the last run time to determine if the set interval is reached. For interrupt event-triggered tasks, call the trigger function and decide whether to execute the task based on the return value.

Step 5. Task Execution and Recording

After completing the inspection and marking of all tasks, the system executes the marked tasks one by one in the order of the task list, records the start time, and runs the corresponding program.After the task is completed, calculate the running time and update statistical data such as the maximum and minimum running times.

Step 6. Exception Handling Mechanism

During task execution, check for exceptions such as division-by-zero errors. If an exception occurs, decide whether to stop operation according to the system configuration, set the corresponding alarm status, and refer to Chapter 6 Fault for specific fault codes.

Step 7. Output and Status Update

After all tasks are executed, apply the data in the Q and M areas to the actual hardware or other modules.

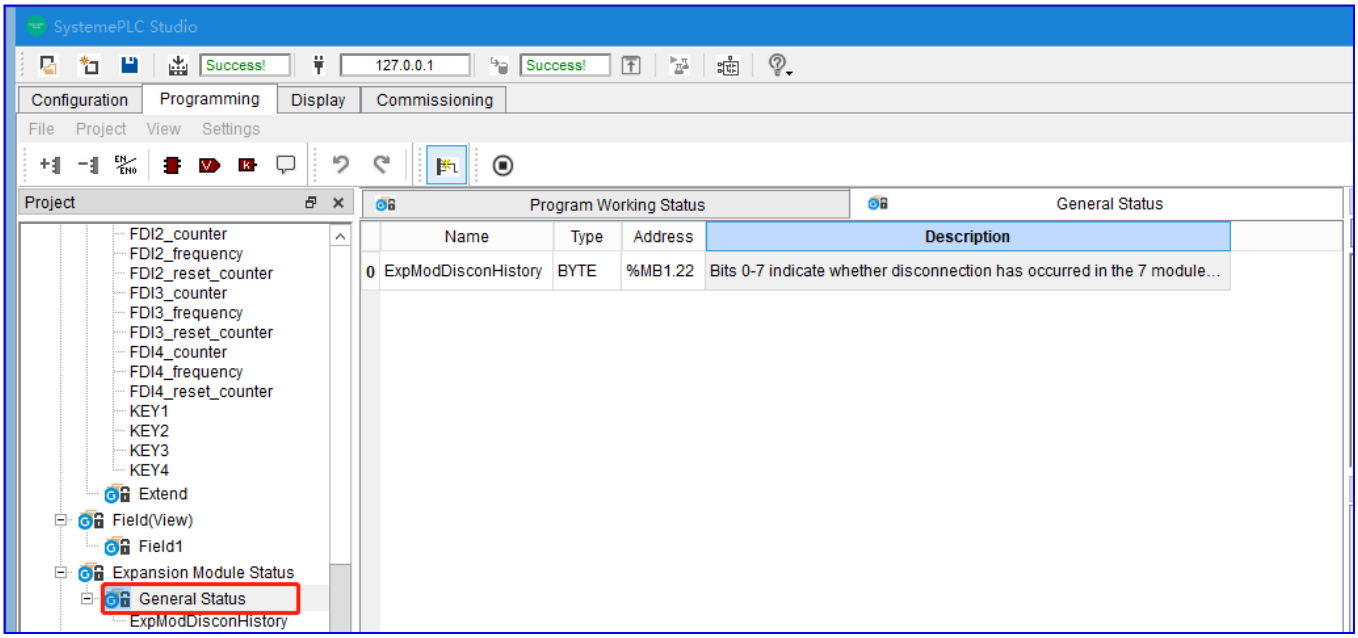
Step 8. Loop Control

After a 1ms delay, repeat the above steps.

3.14 Expansion module status

Check the connection status and error registers of the expansion module in the Expansion Module Status.Click ExpModDisconHistory to view the connection status of the expansion module; the status descriptions are as follows:

Bits0-7 indicate whether disconnection has occurred in the 7 modules (0 = Disconnected before, 1 = Never disconnected).
Bit 0: Expansion Module 0 has been disconnected (1 = occurred, 0 = not occurred).
Bit 1: Expansion Module 1 has been disconnected.
…
Bit 6: Expansion Module 6 has been disconnected.
Bit 7: Reserved.



Click Programming → Expansion Module Status, then double-click the expansion module to enter the error registers and identification registers interface.

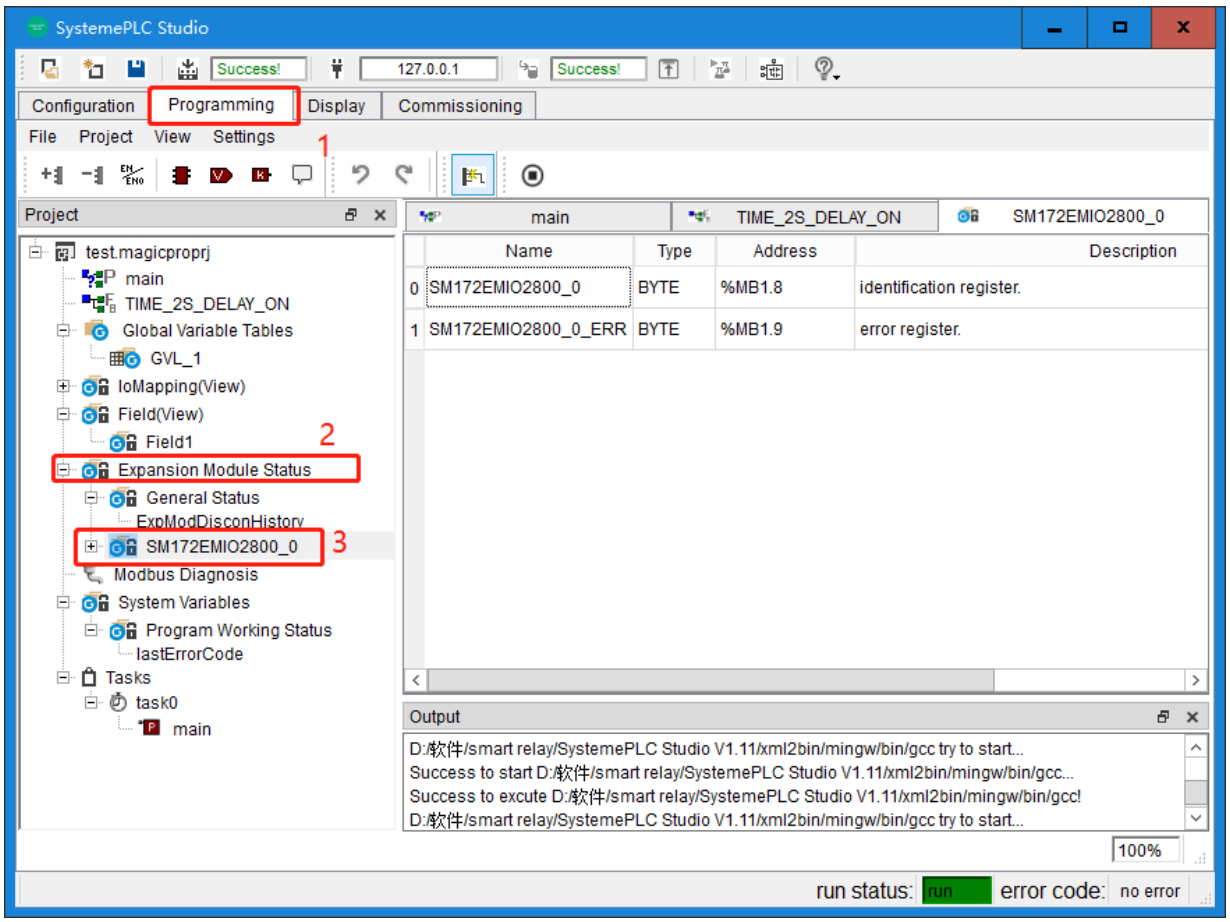
Error Registers:

0 = Normal

0xFF = Communication Abnormality

0x04 = Module Power Supply Abnormality

0x08 = Module AI Channel Out of Range

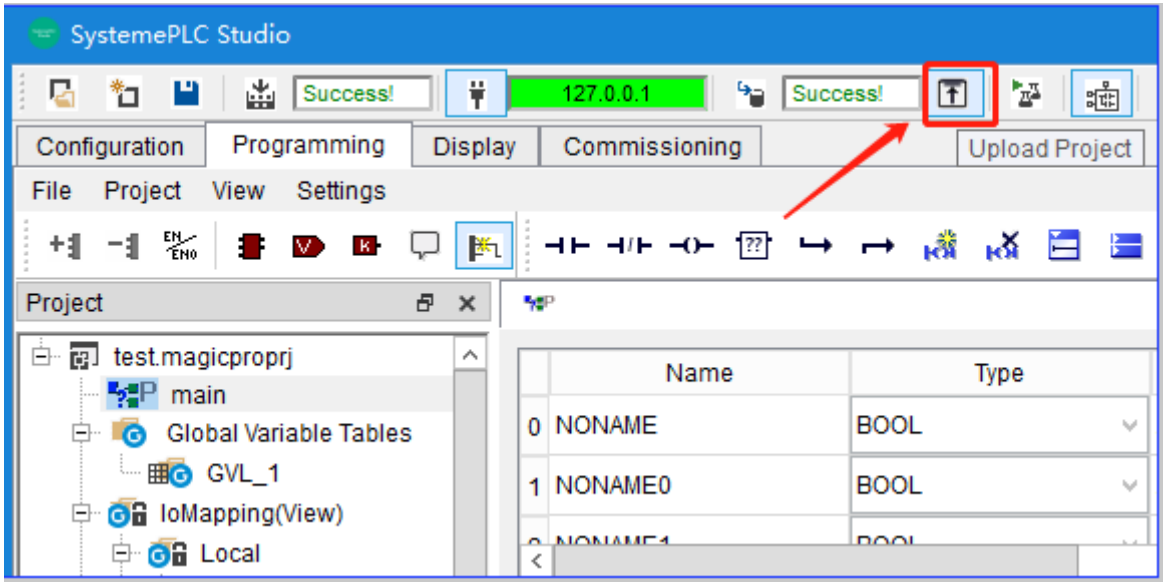


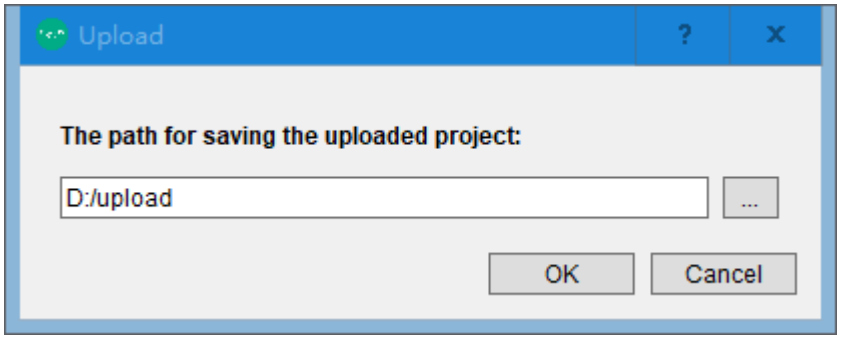
3.15 Upload

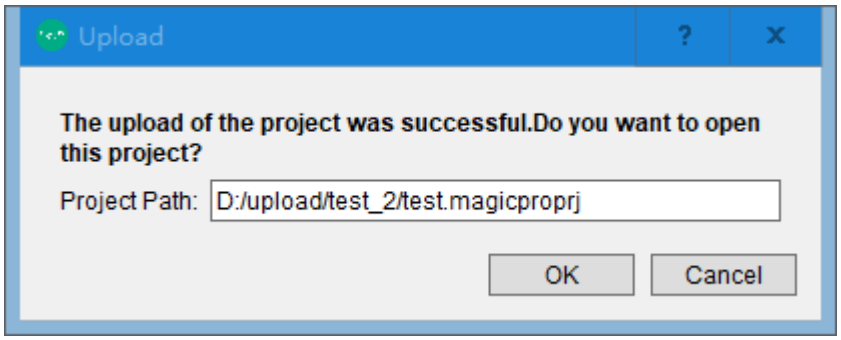
3.15.1 **Project upload**

Project Upload refers to the process of uploading a complete project file from the PLC to the programming software. The file contains all project information, including the program, hardware configuration, comments, etc. This operation enables the complete restoration of a modifiable engineering project.

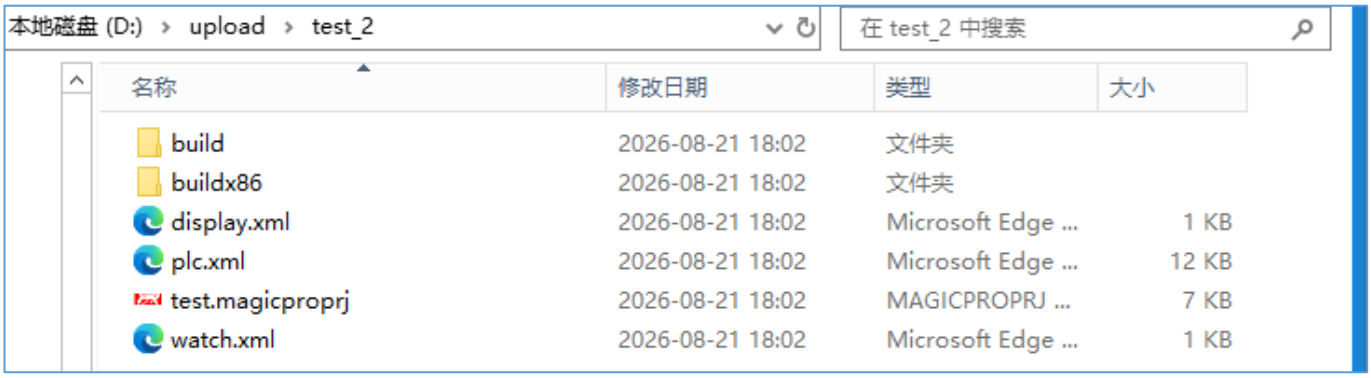
To perform the upload while the PLC is online, click the Upload button, select the upload path, and wait for the upload progress bar to indicate completion. After that, a prompt will appear asking whether you want to open the project.







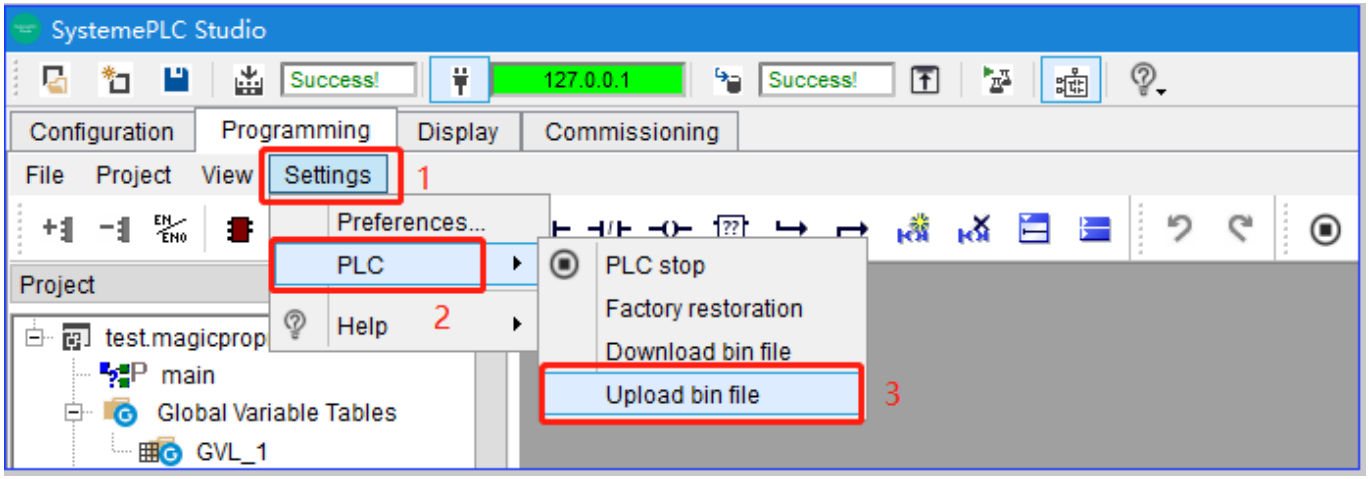
After the upload is completed, the system will automatically generate a folder in the project path, which contains the complete project information.

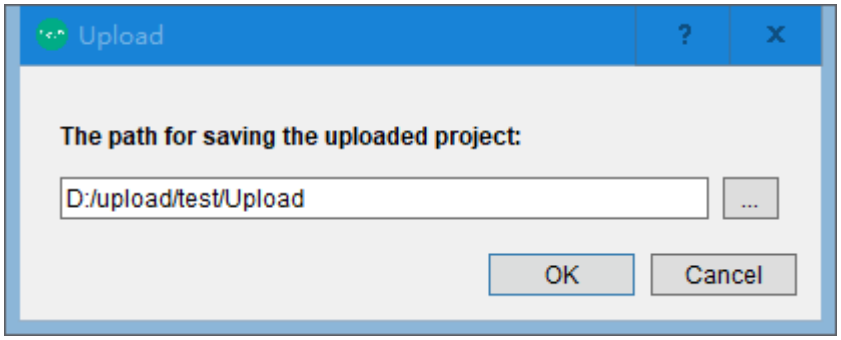


3.15.2 Bin files upload

The uploaded bin files consist of config.bin and prog.bin. The bin file is non-editable. The upload destination path is user-definable, with the default path being within the original program directory.

prog.bin: User program file: contains the primary logic and instruction set to be executed by the PLC.
config.bin: Configuration file: holds no program logic, but rather stores hardware configuration, communication parameters, I/O mapping, module settings, and other device-specific parameters.





Fourth section

Communication configuration

4.1 MODBUS RTU communication

4.2 MODBUS TCP communication

4.3 MODBUS address correspondence

4.4 Commissioning

4.5 FBD block pin inversion function

4.6 FBD connection break function

4.7 Modbus customer editor

4.8 Display

Modbus is a simple and open serial communication protocol that supports Modbus RTU and Modbus ASCII protocols. Generally, most PLC devices support Modbus RTU protocol. Modbus communication media include RS485 and Ethernet. The Smart Relay series PLC comes with two RS485 ports, RS485-1 and RS485-2 supports MODBUS master/slave.

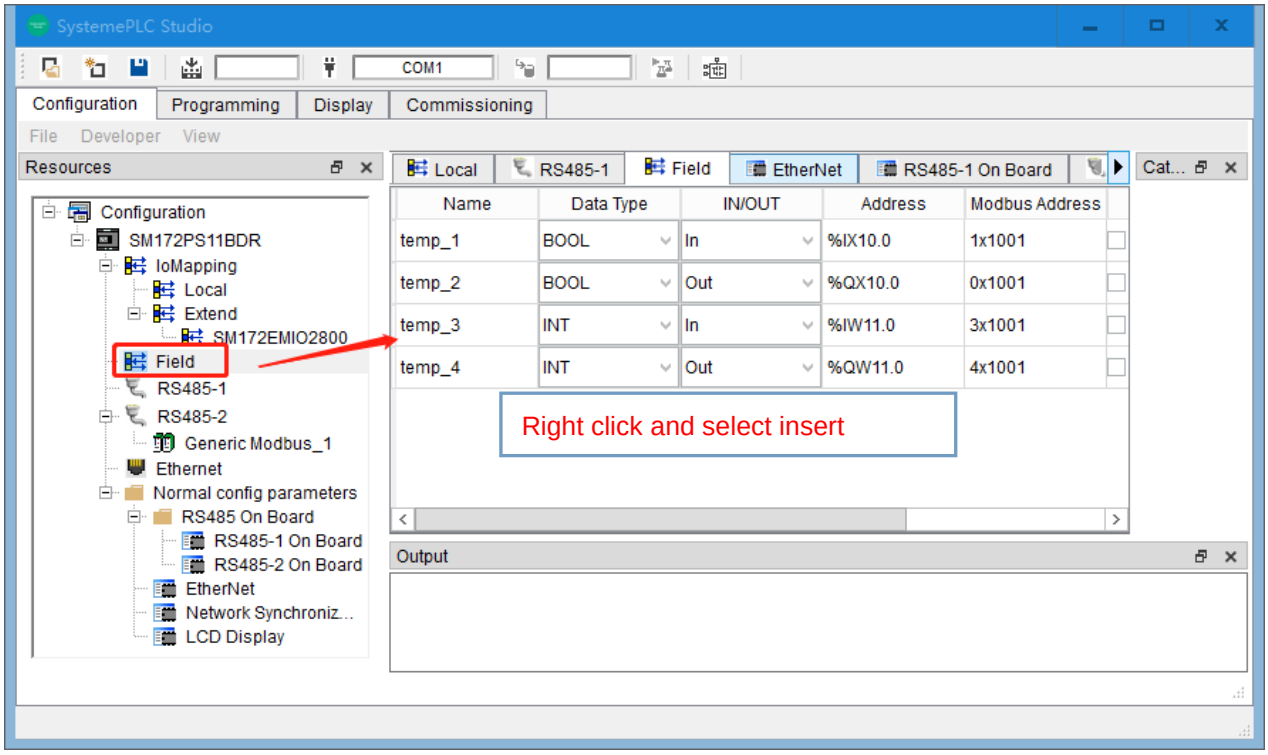
Master: When the controller serves as a master station device, it can communicate with other slave devices that use the Modbus protocol, exchange data with other devices.

Slave: When the controller acts as a slave device, it can only respond to the requirements of other master stations.

4.1 Modbus RTU communication

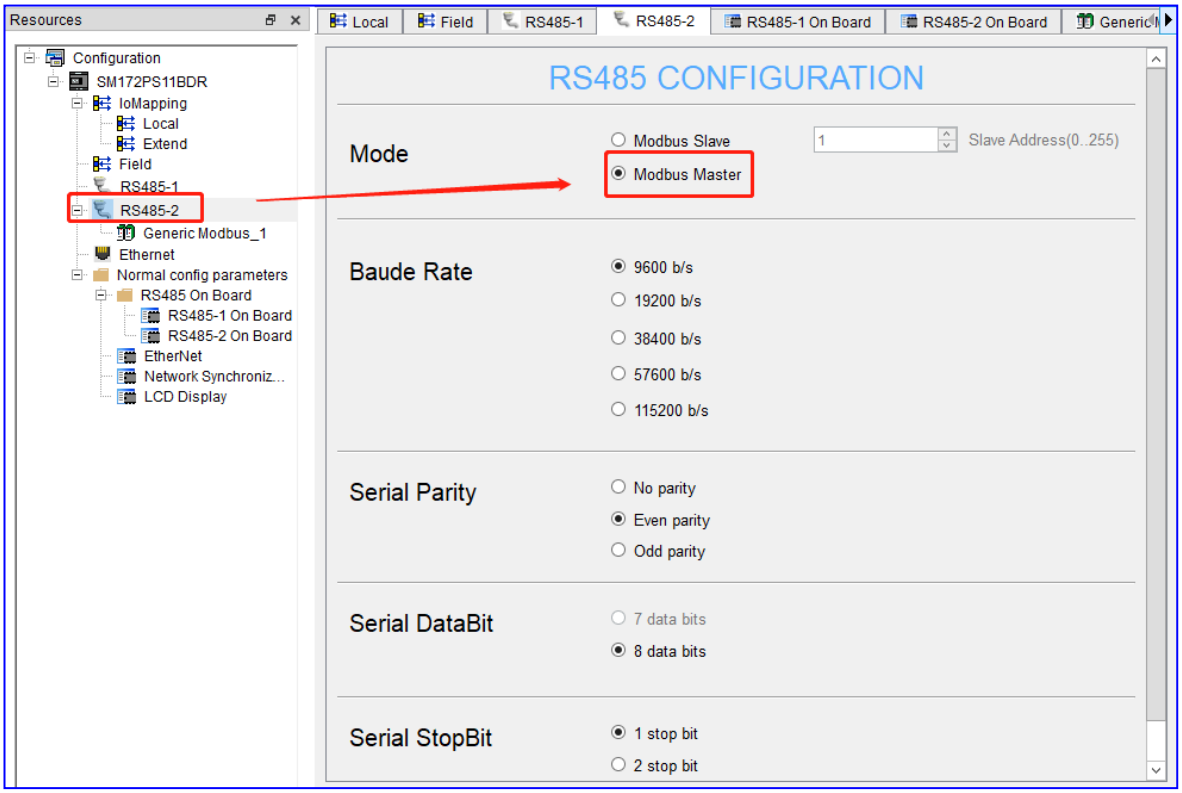
4.1.1 **Modbus RTU master communication configuration**

When the PLC is the master accessing a Modbus slave device, it first needs to create a new point in the Field directory and subsequently associate it with the point accessed by the third-party device.



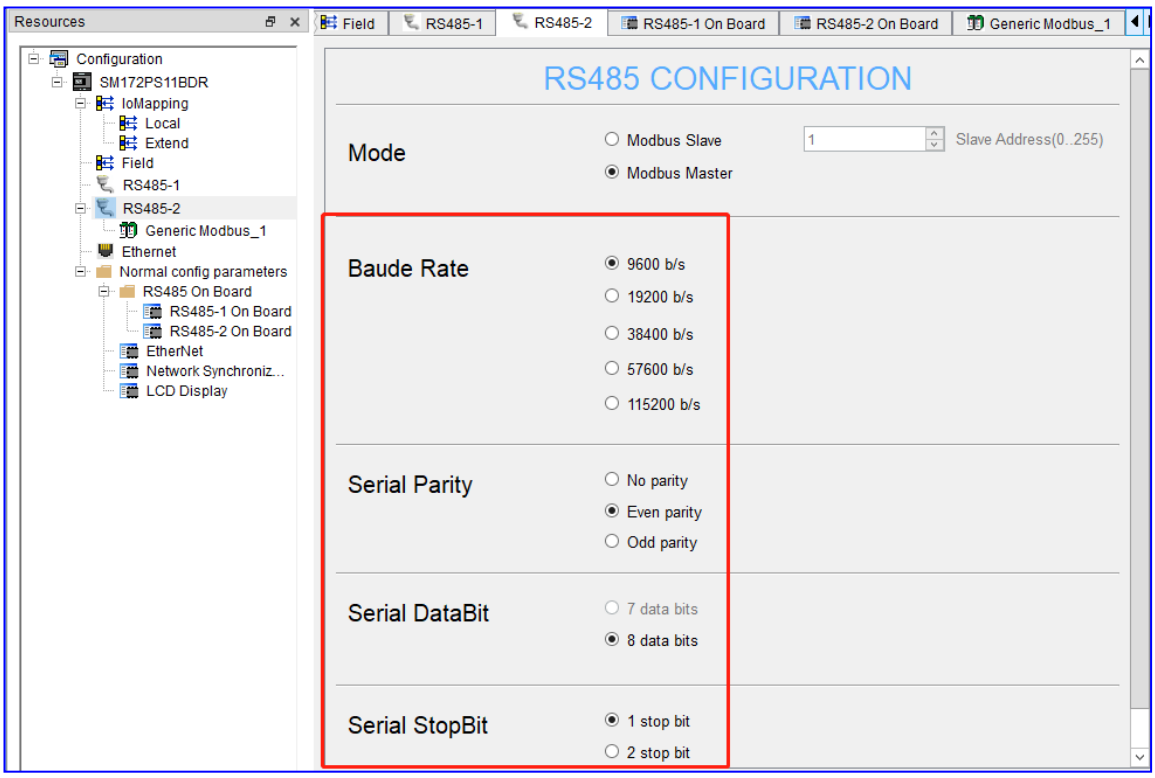
Modbus master communication is configured as follows

1. RS485-2 supports Modbus master, so set the controller as MODBUS RTU master in RS485-2 and configure the baud rate, parity and other communication parameters.

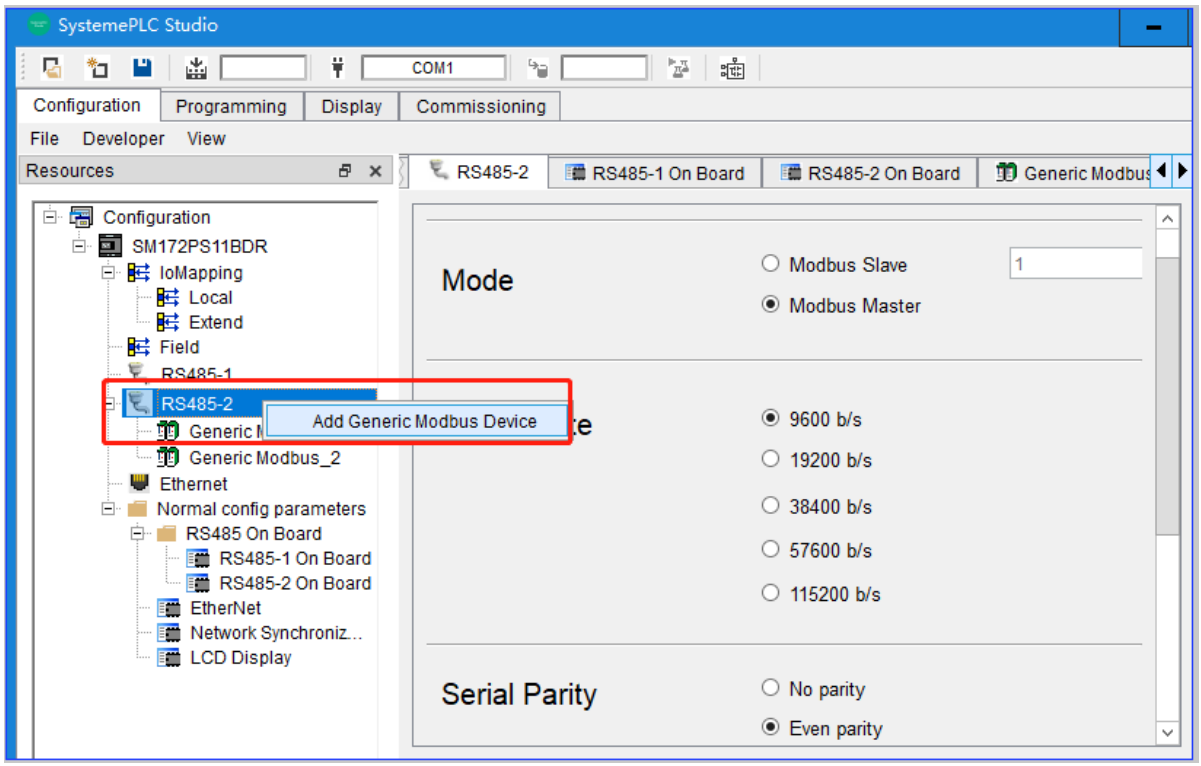


2. Configure the parameters of the master station

Item	Descriptions
Baude Rate	The baud rate of the serial port. The baud rate of the master and slave should be the same.
Serial Parity	The parity method of communication frames, Even Parity，Odd Parity，NO Parity。
Serial DataBit	The actual data bits contained in the communication frame
Serial StopBit	Identify the last bit of a single packet during communication



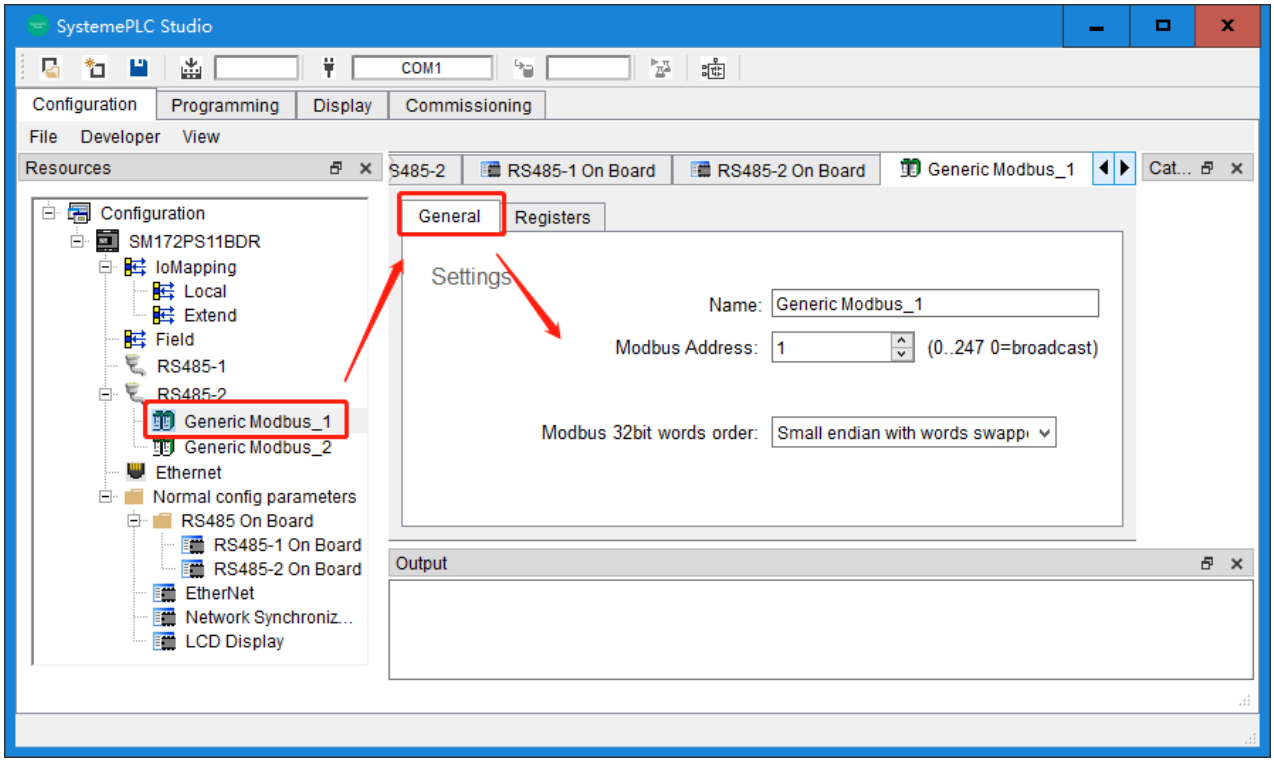
3. Right click on RS485-2 to Add Generic Modbus device.



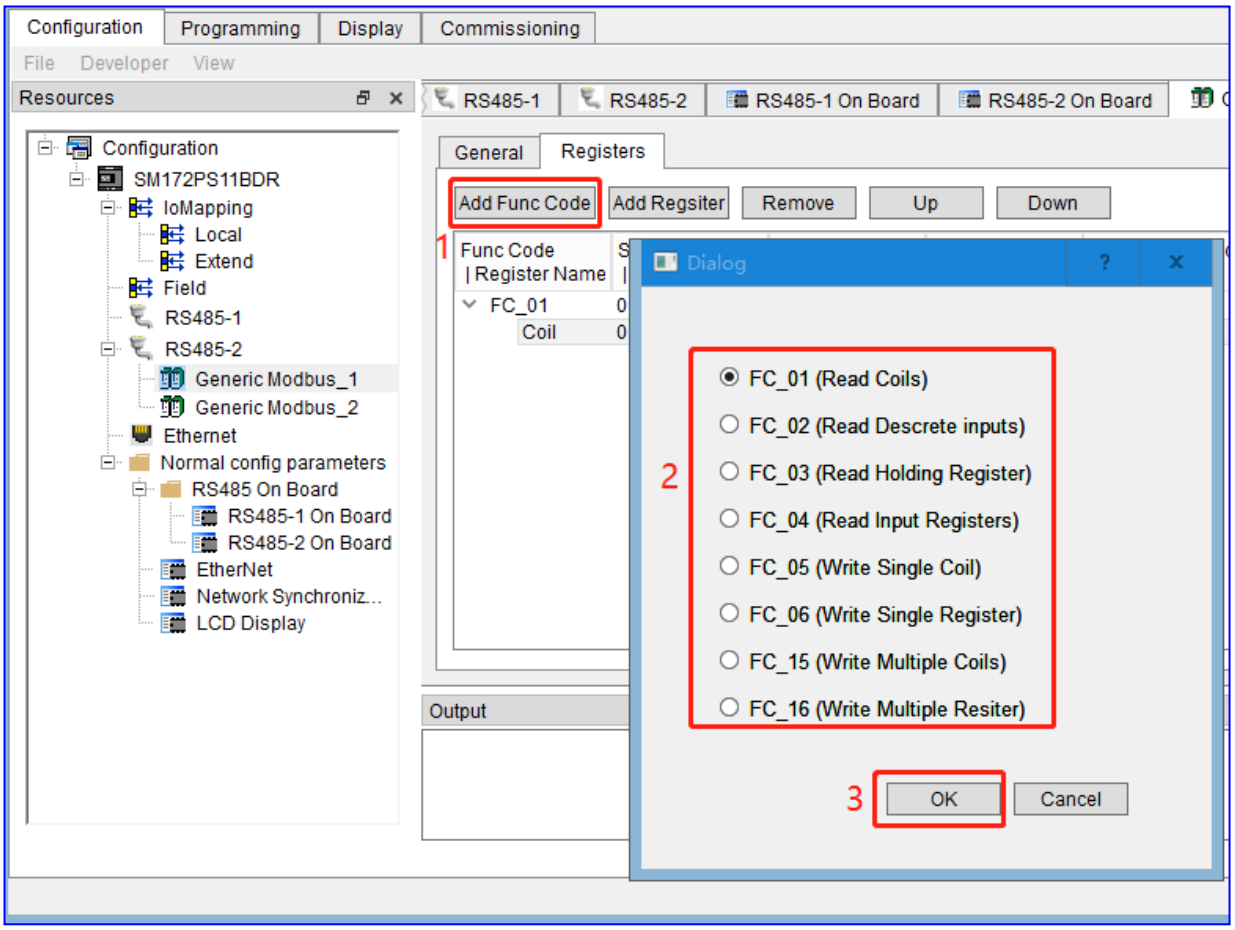
4. General MODBUS device configuration interface is as follows, configure the name of the general MODBUS device, Modbus address.

The Modbus 32-bit word sequence is described as follows:

Take the data ABCD for example, D is the low byte	
Item	**Post exchange word order**
Small endian with words swapped	CDAB
Small endian NOT words swapped	DCBA
Big endian with words swapped	BACD
Big endian NOT words swapped	ABCD

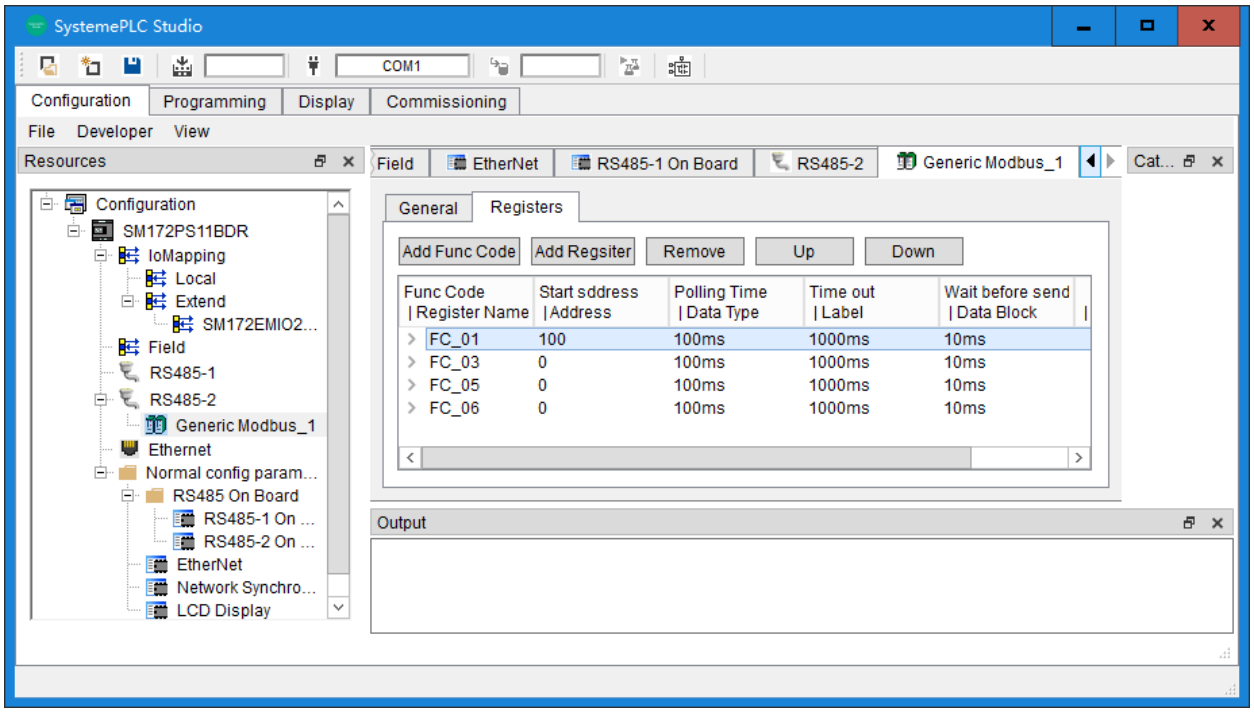


5. Add the corresponding function code according to the type of register and access method of accessing the third-party device. MODBUS RTU function code refers to the code used to specify specific operations in the MODBUS protocol. These operations include reading and writing different types of data.



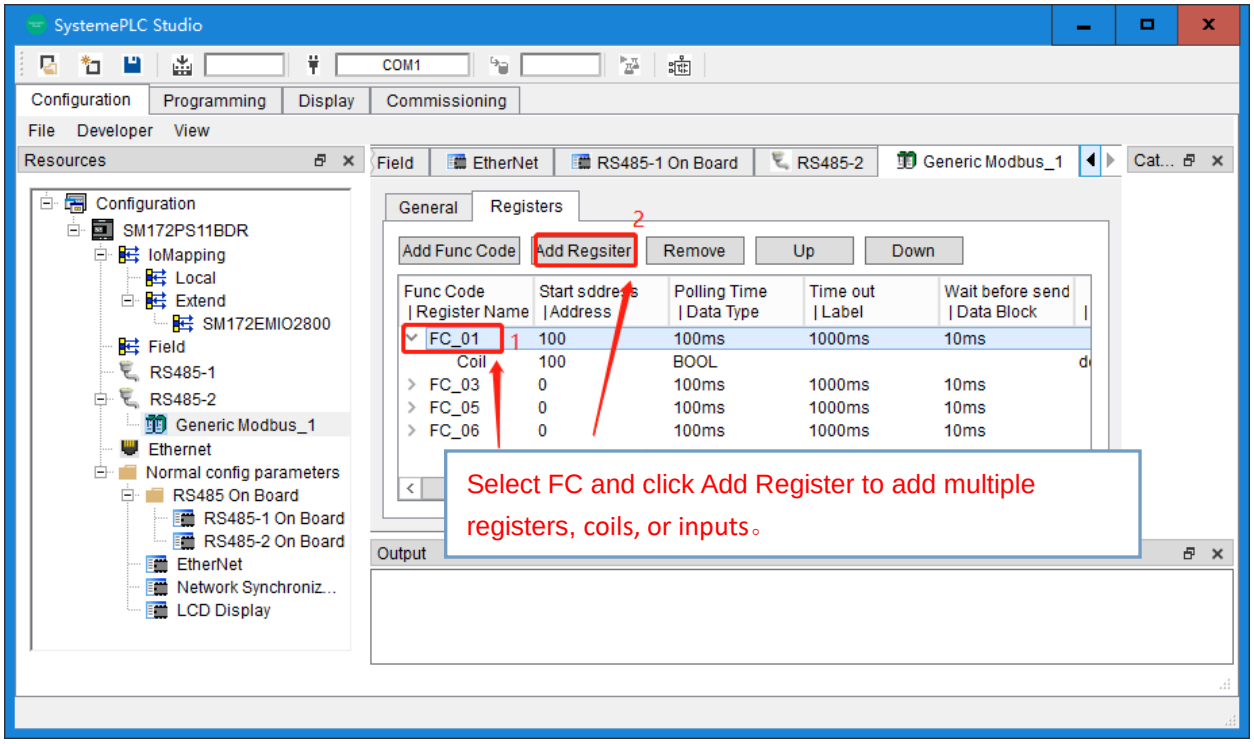
Double click on Start sddress, Polling Time, Time out, Wait before send, data type, etc. of the function code to modify the corresponding options.

<Remarks> For the write request function code, it is sent when the polling time is 0 and the data is changed.

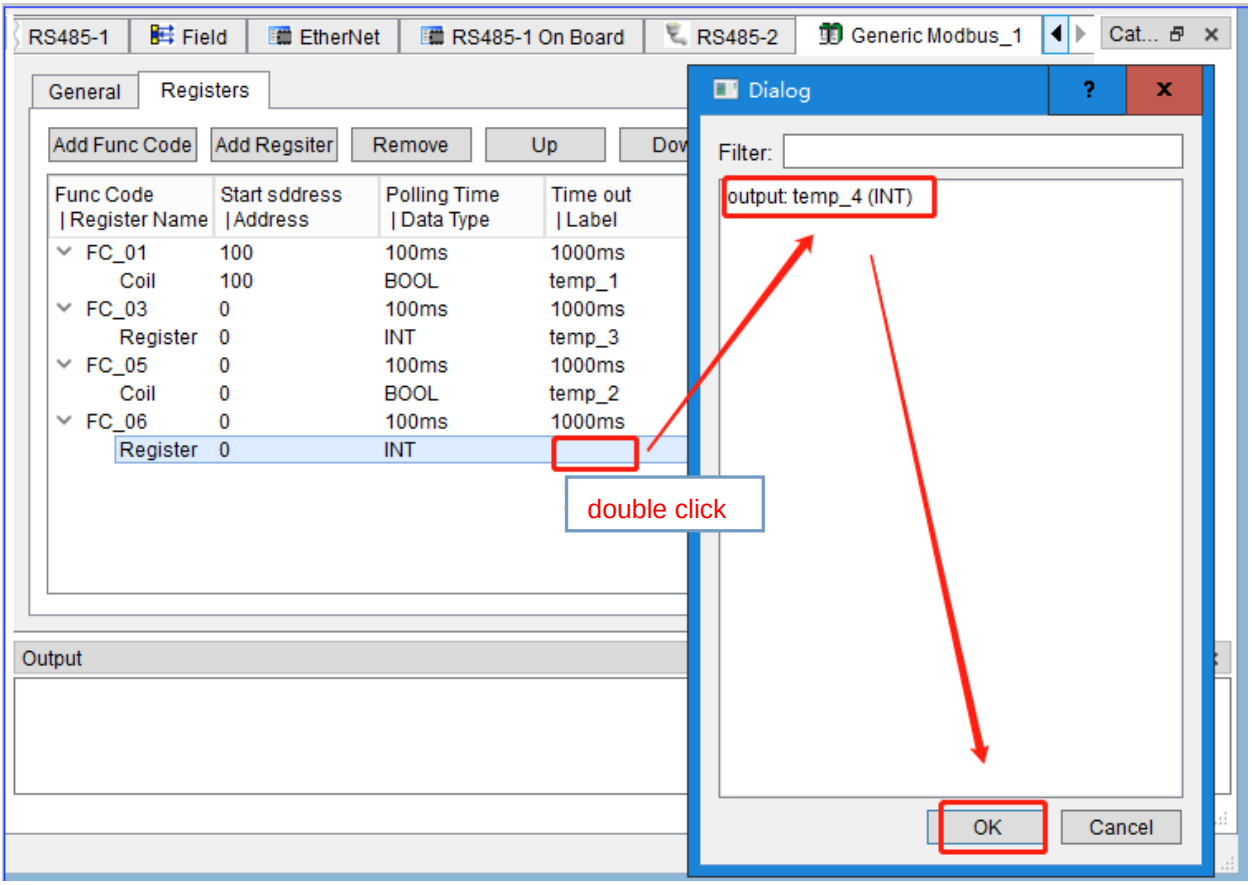


6. Read and write multiple coils or registers, etc., click Add Register to add:

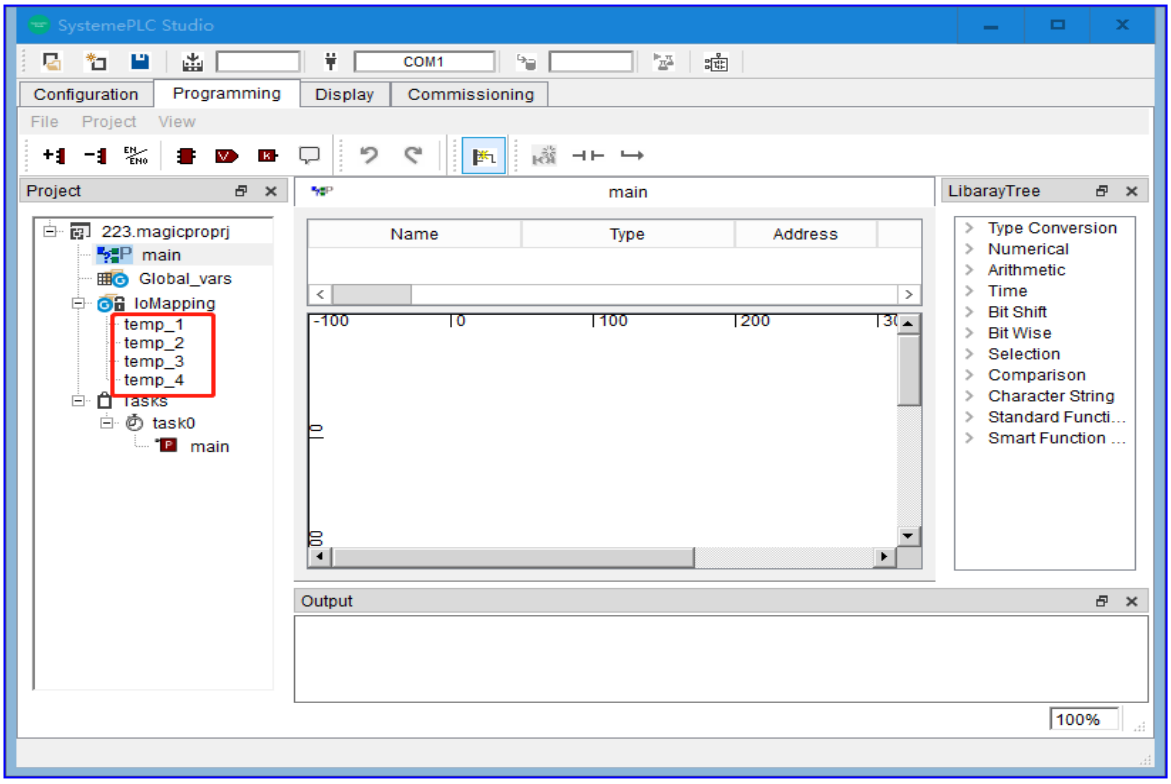
According to the actual register address accessed, you can modify the starting address of the function code, and configure the access time, interval time, etc. If you need to access multiple consecutive addresses, click Add Regsiter to add multiple registers.



7. Double-click the Label entry of the register point, you can map the register to the specified variable (previously created under Field), and when you select the variable, it will recommend the appropriate variable according to the function code and the point type. The data type of the function code and the data type of the created point must be the same in order to be associated.



8. Through the above operations, map the value of the third-party device point to the corresponding local variable, and then you can call this variable in programming to access third-party devices.

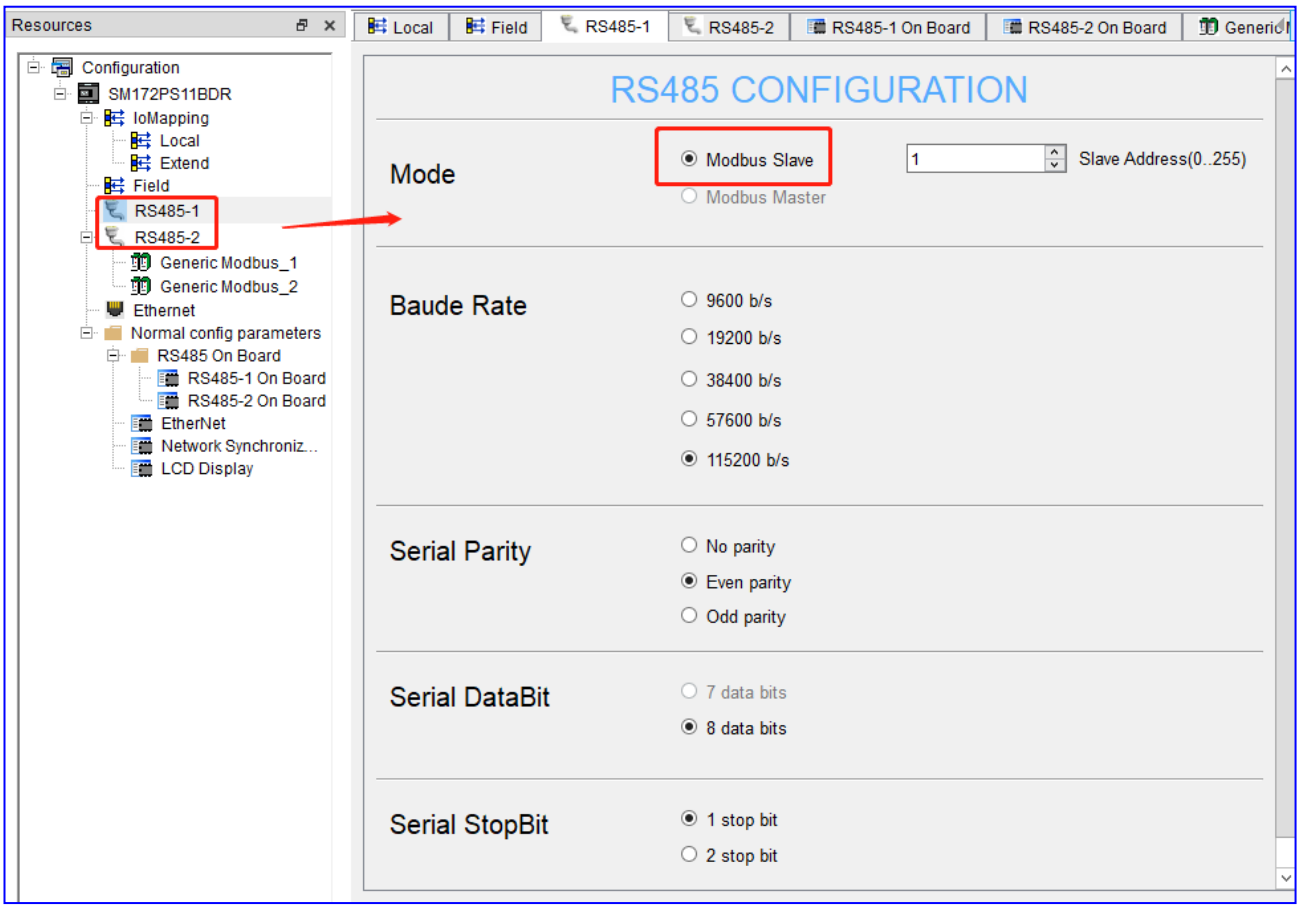


4.1.2 Modbus RTU slave communication configuration

When the PLC is a slave, the configuration is as follows:

1. RS485-1 and RS485-2 support Modbus slave, set the mode to Modbus Slave, configure the slave baud rate, parity, data bits and stop bits. Note that the configuration parameters of the slave station and the master station should be

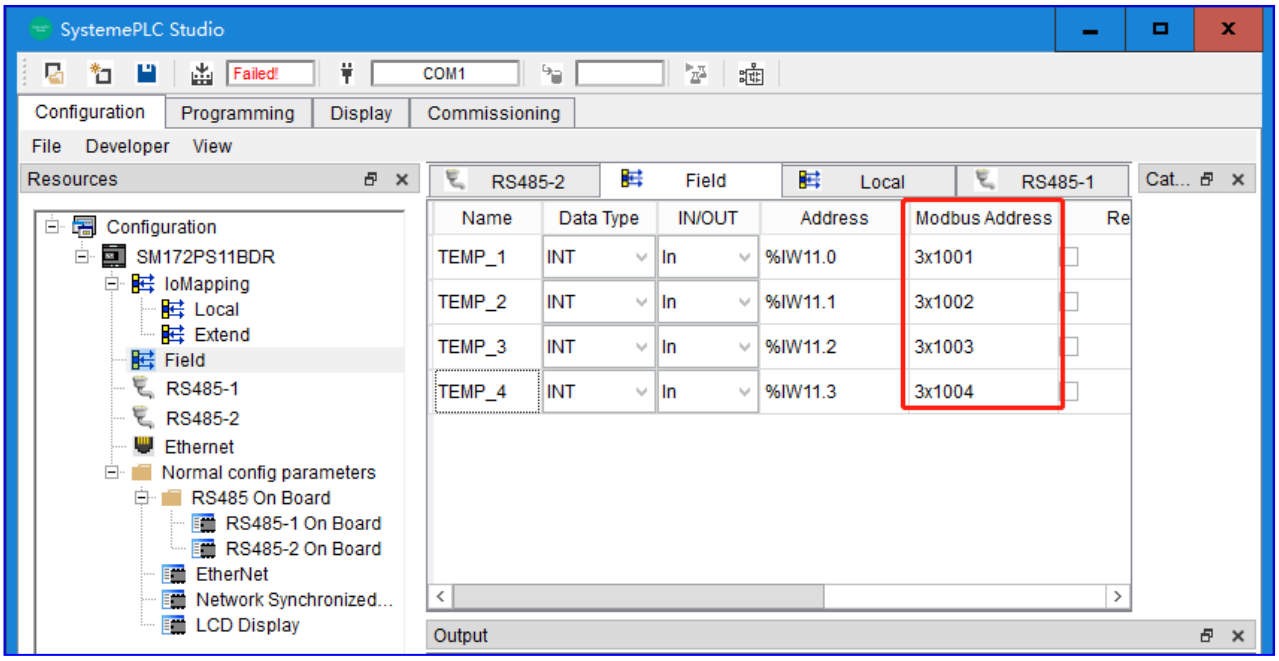
consistent, otherwise the communication will not be successful.

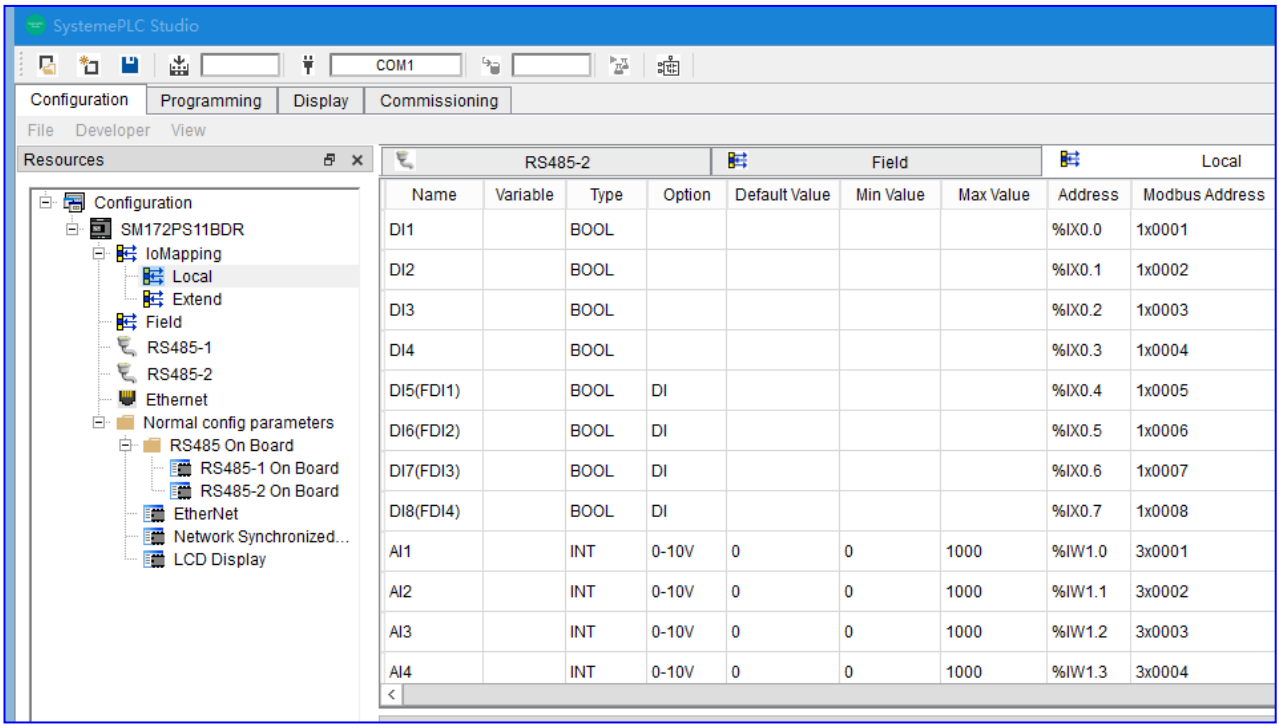


2. There are corresponding addresses for the points in the Local and Field directories.

When the PLC is a slave, if the master needs to access these points, it needs to find the corresponding Modbus mapping according to the local address of the points.

SM172 / SR2: DI/DO < 4096, AI/AO < 2048. SR1: DI/DO <512, AI/AO < 256





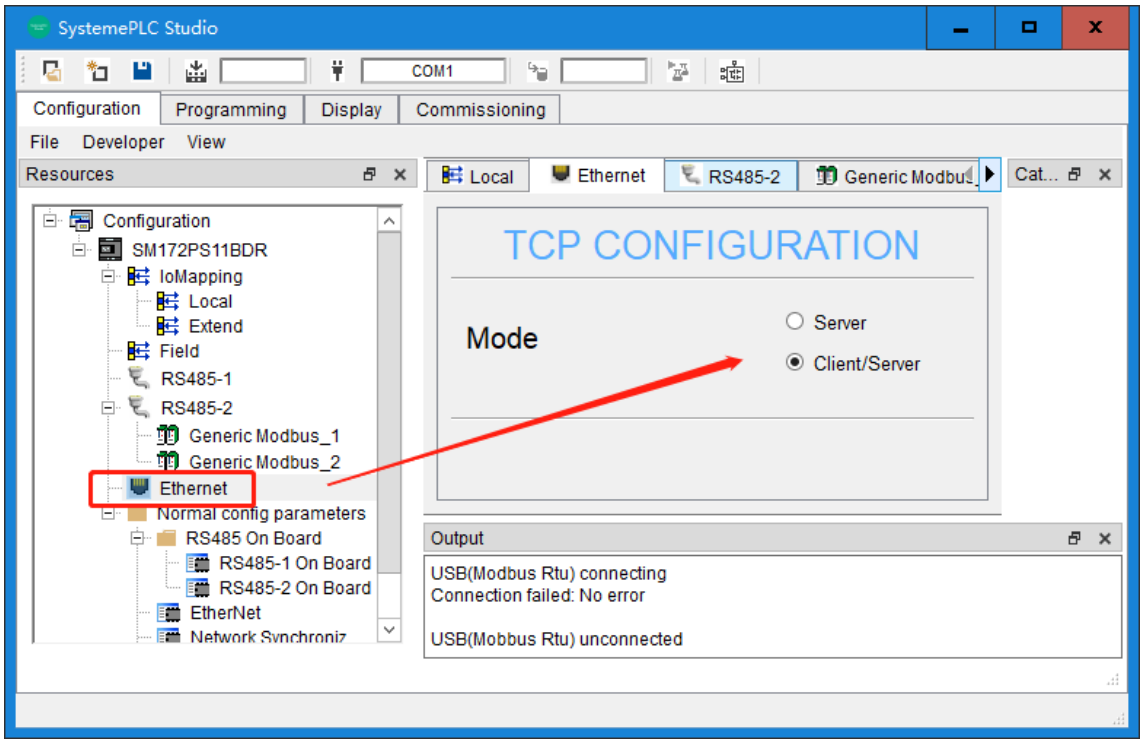
Name	Variable	Type	Option	Default Value	Min Value	Max Value	Address	Modbus Address
DI1		BOOL					%IX0.0	1x0001
DI2		BOOL					%IX0.1	1x0002
DI3		BOOL					%IX0.2	1x0003
DI4		BOOL					%IX0.3	1x0004
DI5(FDI1)		BOOL	DI				%IX0.4	1x0005
DI6(FDI2)		BOOL	DI				%IX0.5	1x0006
DI7(FDI3)		BOOL	DI				%IX0.6	1x0007
DI8(FDI4)		BOOL	DI				%IX0.7	1x0008
AI1		INT	0-10V	0	0	1000	%IW1.0	3x0001
AI2		INT	0-10V	0	0	1000	%IW1.1	3x0002
AI3		INT	0-10V	0	0	1000	%IW1.2	3x0003
AI4		INT	0-10V	0	0	1000	%IW1.3	3x0004

4.2 Modbus TCP communication

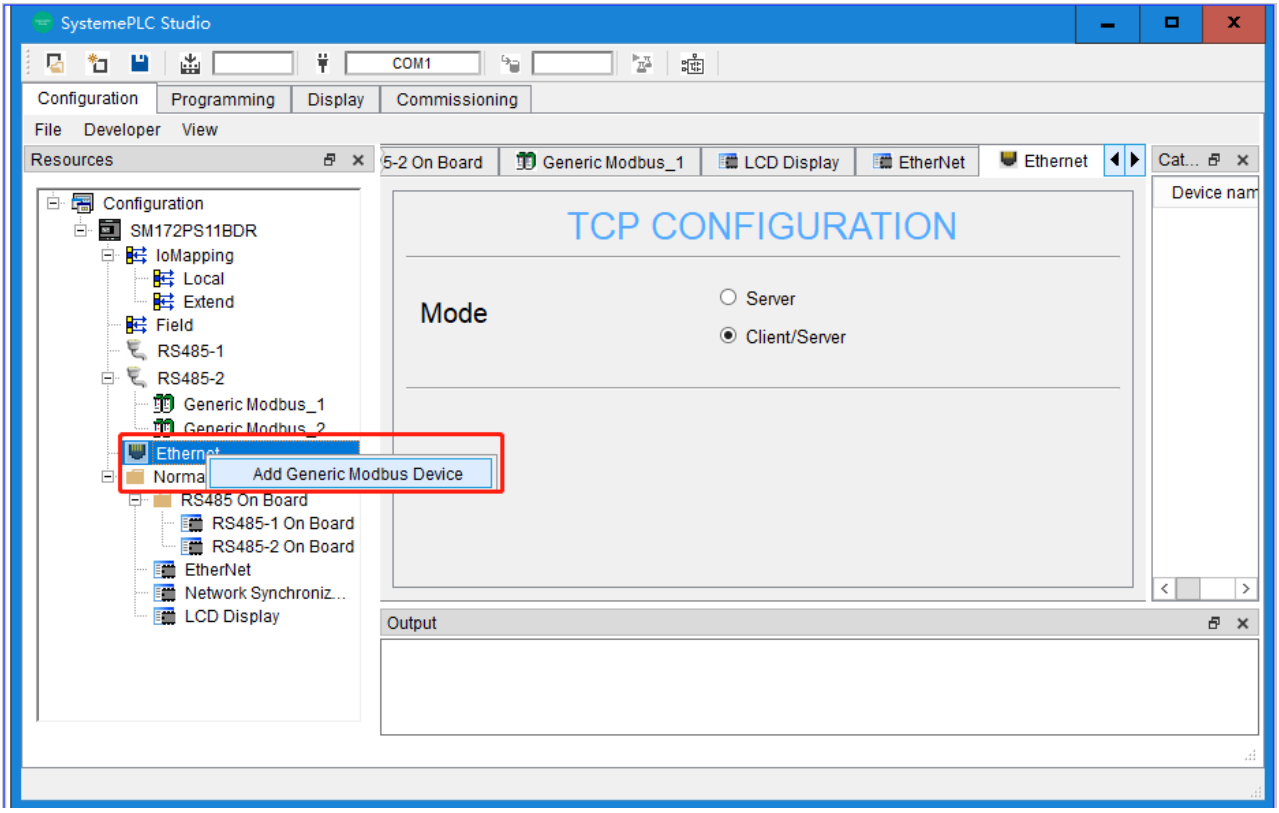
Smart Relay Series PLCs support 1-channel Modbus TCP communication and can be Modbus TCP masters and slaves. The Modbus TCP slave function allows up to 8 masters to be connected, and the master function allows up to 8 slaves to be connected.

4.2.1 **Modbus TCP master communication configuration**

1. When the PLC does Modbus TCP master, click Ethernet under the configuration project tree to configure the PLC as Modbus TCP master.

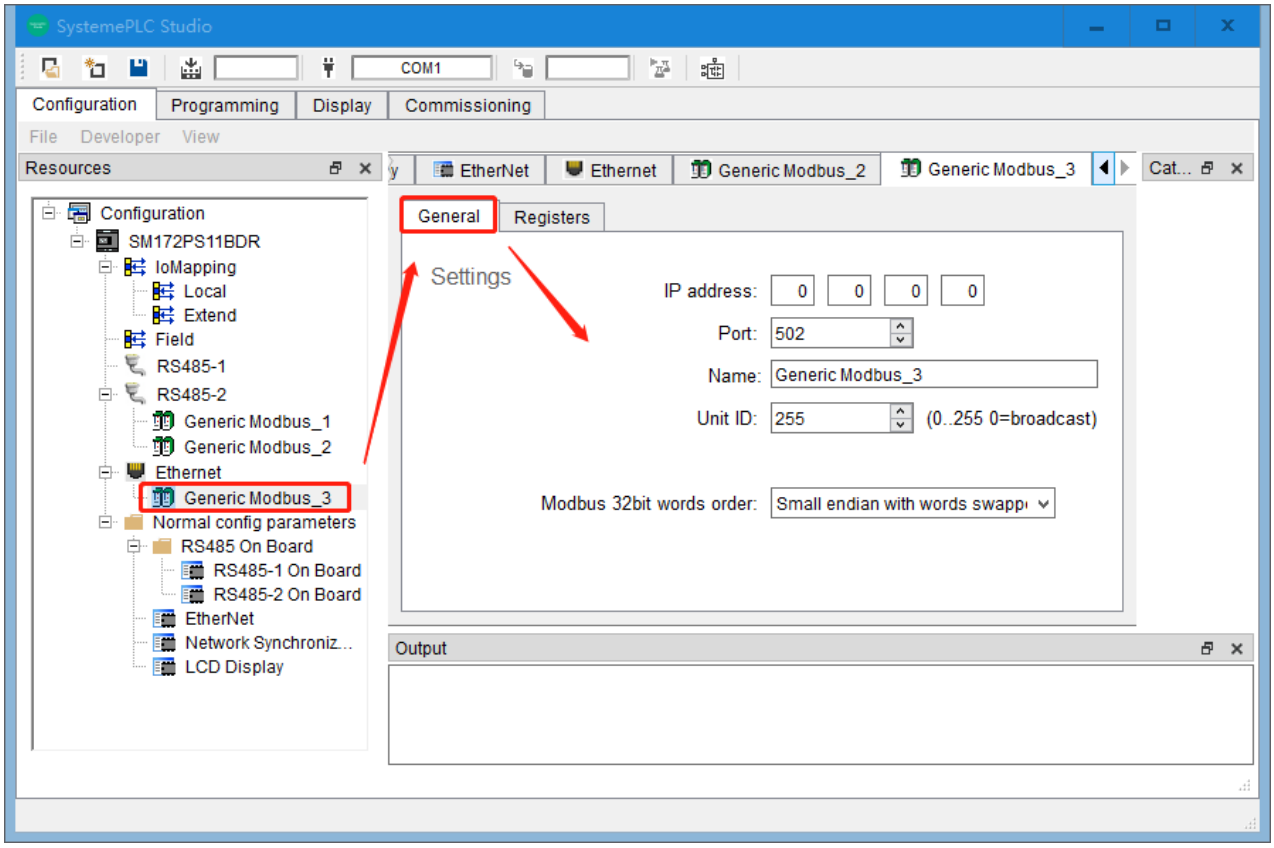


2. Right-click on Ethernet to add a generic Modbus device.

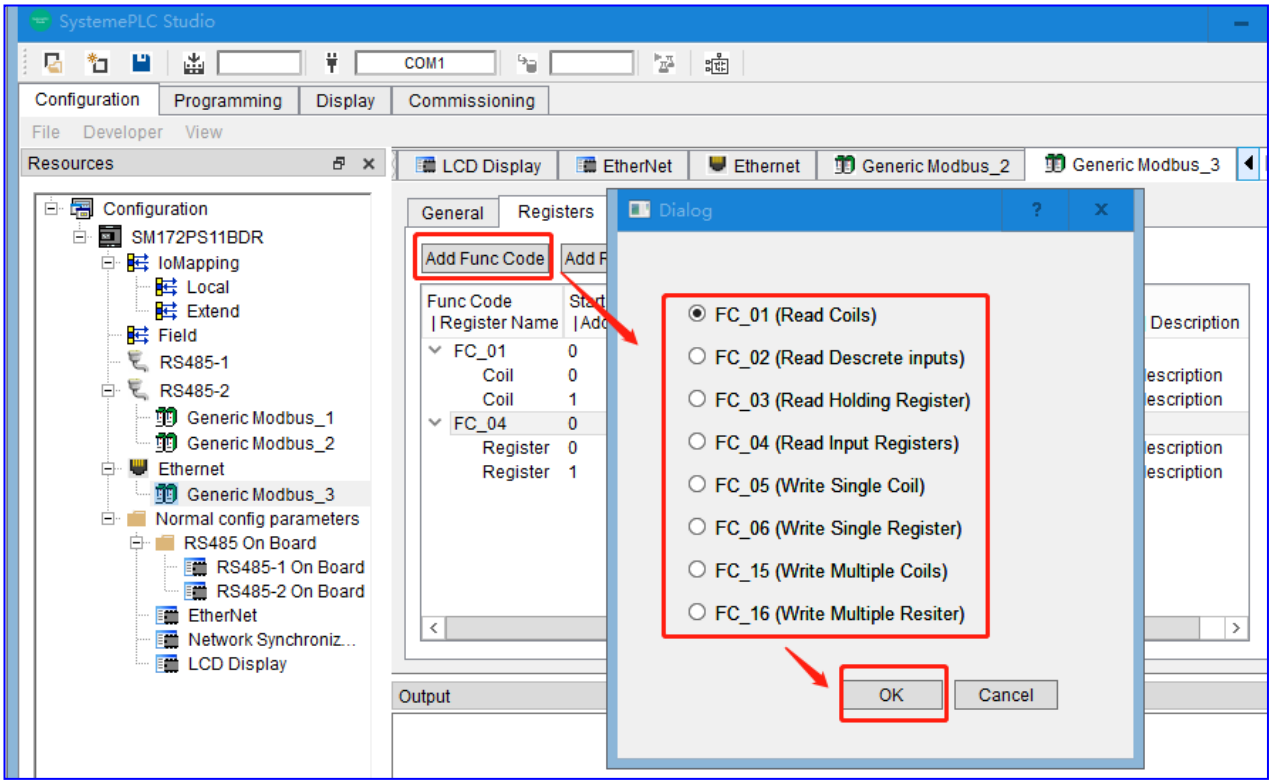


3. Modbus TCP master connects to slave configuration, the configuration parameters are described as follows:

Item	Descriptions	Example
IP address	Modbus TCP slave's IP address	192.168.1.20
Port	The TCP port number of the Modbus TCP slave connected to the master station.	502
Name	Slave name	
Unit ID	The ID is used when communicating between the master and slave stations to ensure that data is correctly sent to the designated slave device.	1
Modbus 32bit words order	--	



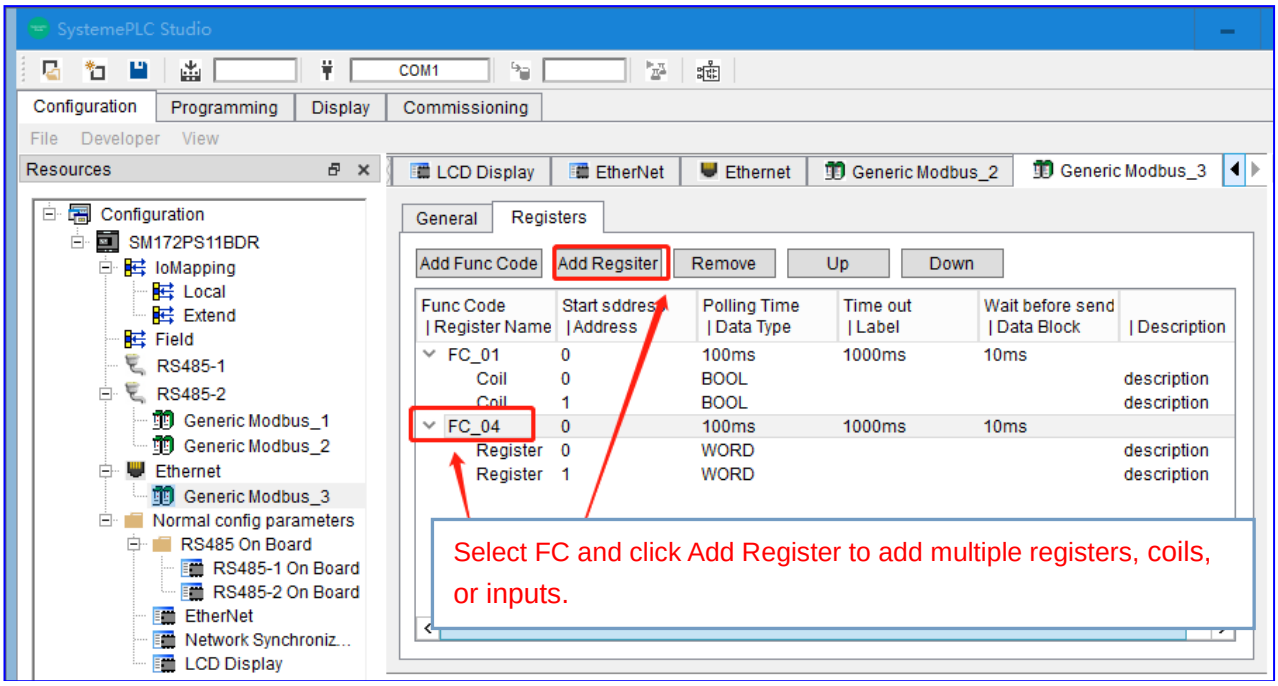
4. Add the corresponding function code according to the type of register and access method of accessing the third-party device. MODBUS function code refers to the code used to specify specific operations in the MODBUS protocol. These operations include reading and writing different types of data.



5. Read and write multiple coils or registers, etc., click Add Register to add:

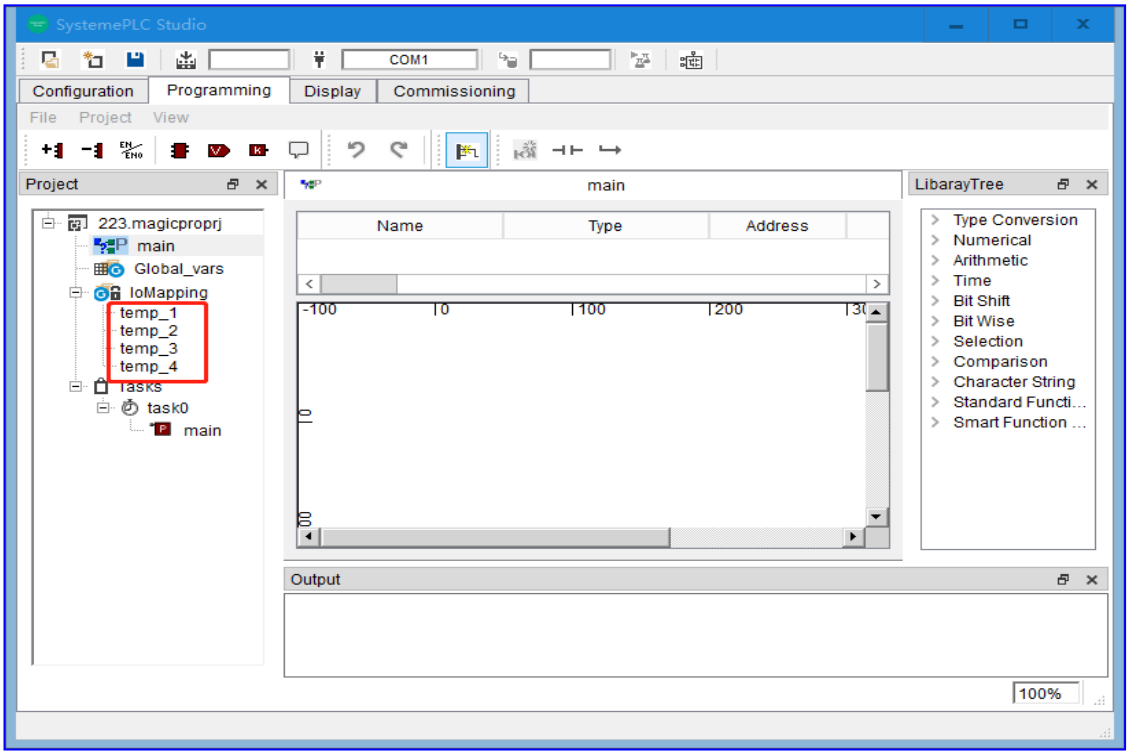
According to the actual register address accessed, you can modify the starting address of the function code, and configure the access time, interval time, etc. If you need to access multiple consecutive addresses, click Add Regsiter to add multiple registers.

<Remarks> For the write request function code, it is sent when the polling time is 0 and the data is changed.



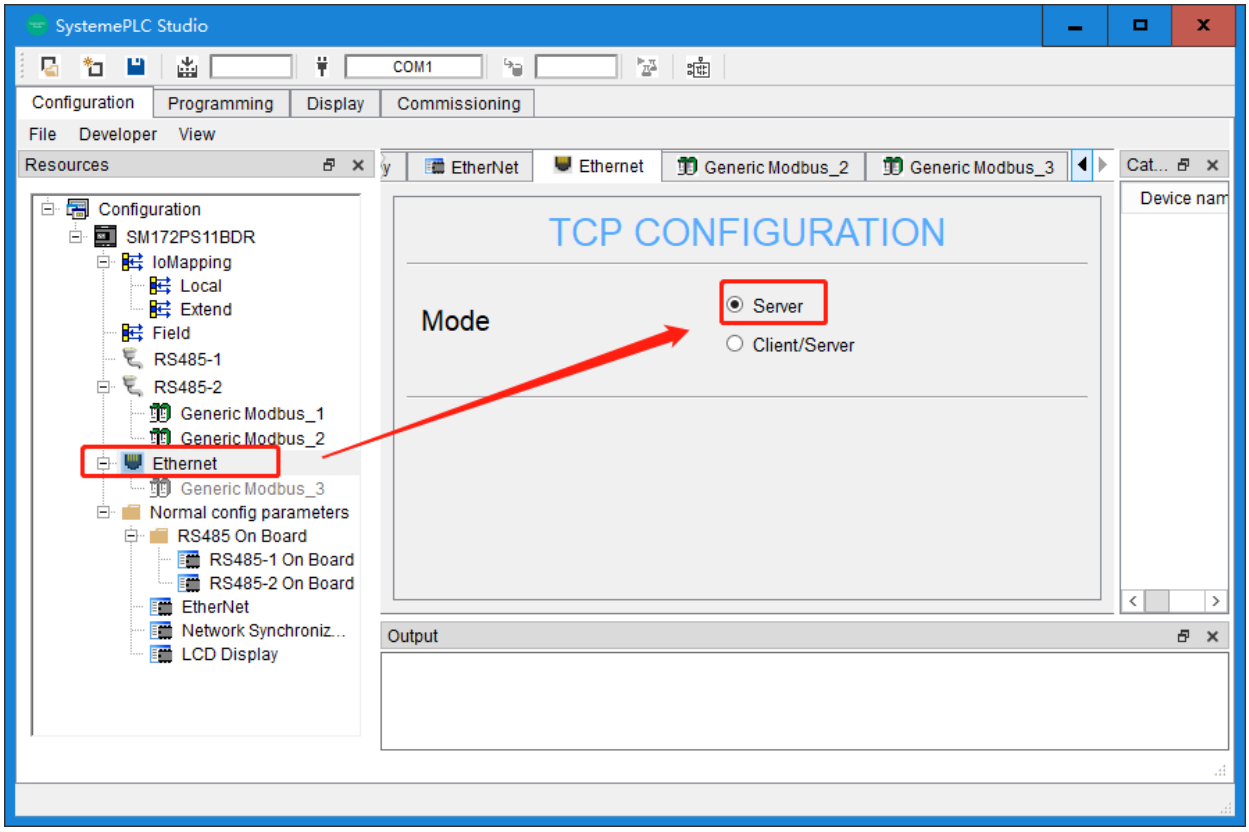
6. By double-clicking the Label of the register point, you can map the current register to a specified variable (the variable previously established under Field). When selecting a variable, an appropriate one will be recommended based on the function code and point type. The data type of the function code must be consistent with the established point data type in order to be associated.

7. After mapping the values of the third-party device points to the corresponding local variables through the above operations, you can use this variable to access the third-party device in programming.

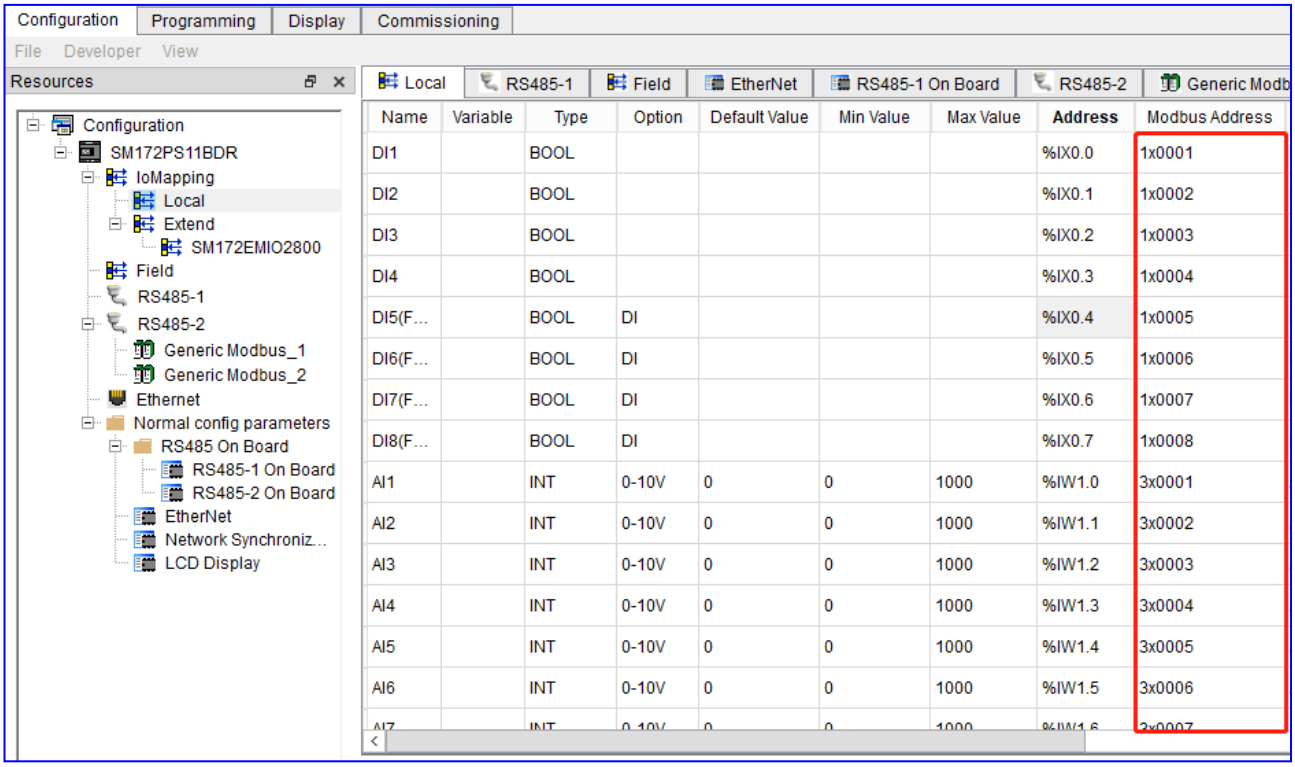


4.2.2 Modbus TCP slave communication configuration

1. When the PLC is a slave, click Ethernet under the Configuration project tree to configure the PLC as a Modbus TCP slave.



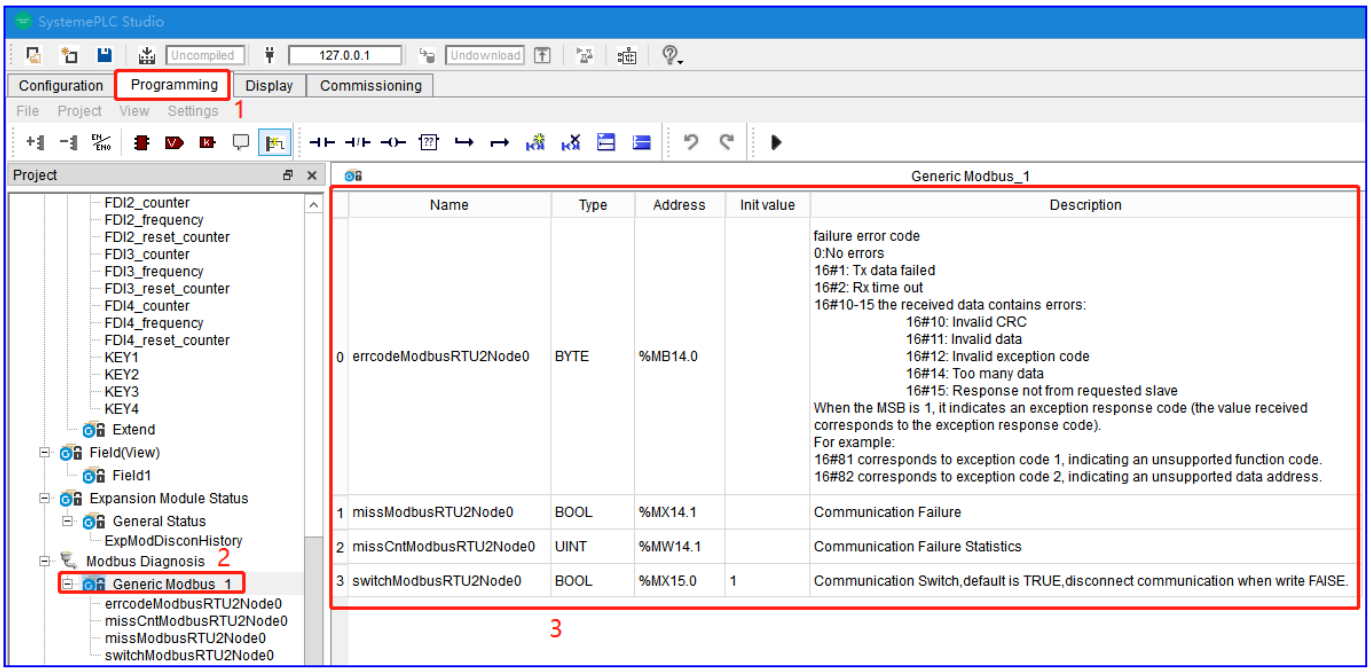
2. There will be corresponding addresses for points in the Local directory. When the device acts as a slave, the master needs to access these points by searching for the corresponding Modbus mapping based on the local address of the point.

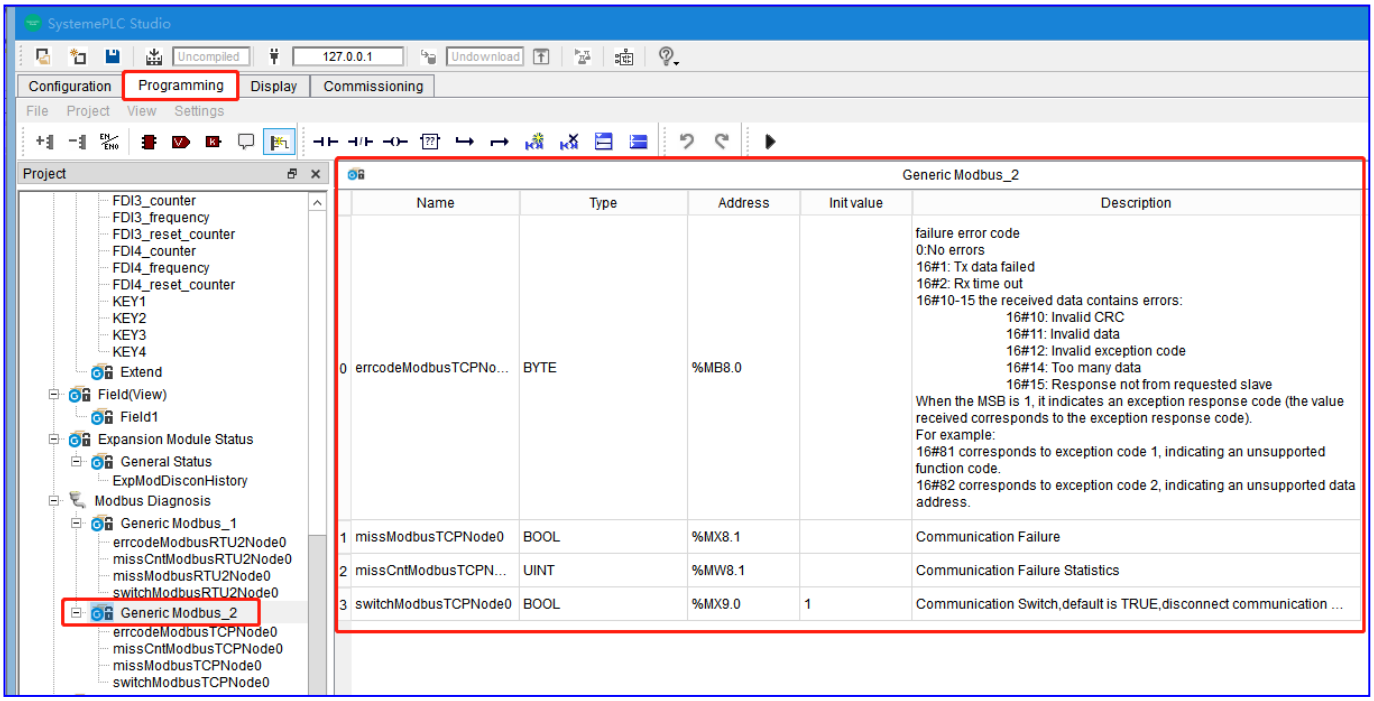


Name	Variable	Type	Option	Default Value	Min Value	Max Value	Address	Modbus Address
DI1		BOOL					%IX0.0	1x0001
DI2		BOOL					%IX0.1	1x0002
DI3		BOOL					%IX0.2	1x0003
DI4		BOOL					%IX0.3	1x0004
DI5(F...		BOOL	DI				%IX0.4	1x0005
DI6(F...		BOOL	DI				%IX0.5	1x0006
DI7(F...		BOOL	DI				%IX0.6	1x0007
DI8(F...		BOOL	DI				%IX0.7	1x0008
AI1		INT	0-10V	0	0	1000	%IW1.0	3x0001
AI2		INT	0-10V	0	0	1000	%IW1.1	3x0002
AI3		INT	0-10V	0	0	1000	%IW1.2	3x0003
AI4		INT	0-10V	0	0	1000	%IW1.3	3x0004
AI5		INT	0-10V	0	0	1000	%IW1.4	3x0005
AI6		INT	0-10V	0	0	1000	%IW1.5	3x0006
AI7		INT	0-10V	0	0	1000	%IW1.6	3x0007

4.3 Modbus Diagnosis

Modbus Master Diagnostics Description





Modbus exception response code

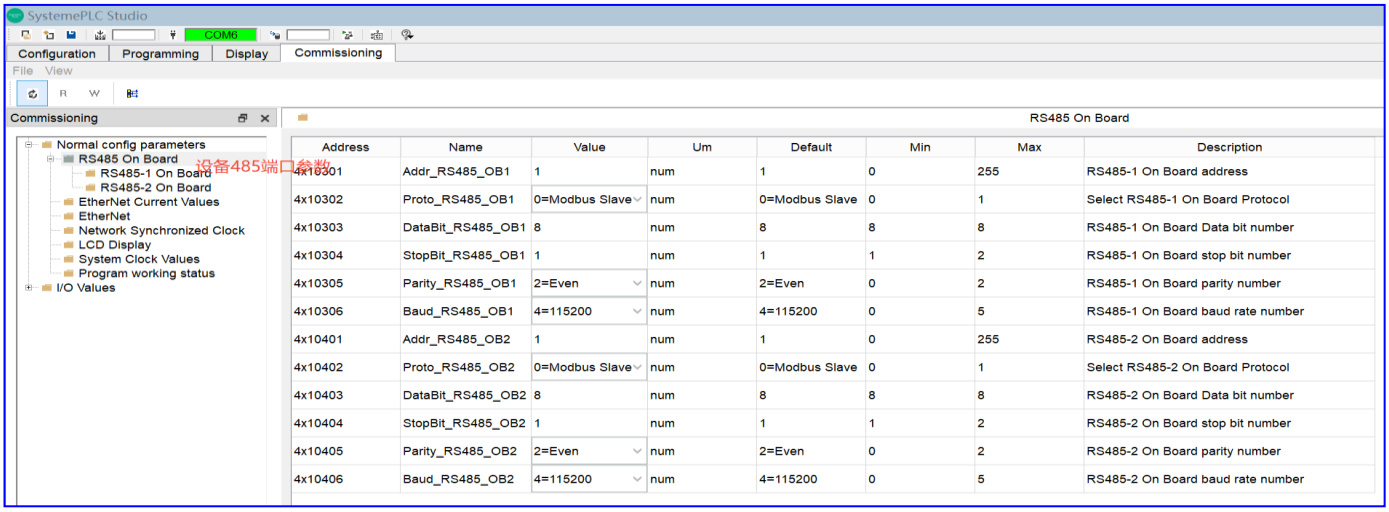
Code	Name	Meaning
01	ILLEGAL FUNCTION	The function code received in the query is not an allowable action for the server. This may be because the function code is only applicable to newer devices, and was not implemented in the unit selected. It could also indicate that the server is in the wrong state to process a request of this type, for example because it is unconfigured and is being asked to return register values.
02	ILLEGAL DATA ADDRESS	The data address received in the query is not an allowable address for the server. More specifically, the combination of reference number and transfer length is invalid. For a controller with 100 registers, the PDU addresses the first register as 0, and the last one as 99. If a request is submitted with a starting register address of 96 and a quantity of registers of 4, then this request will successfully operate (address-wise at least) on registers 96, 97, 98, 99. If a request is submitted with a

		starting register address of 96 and a quantity of registers of 5, then this request will fail with Exception Code 0x02 “Illegal Data Address” since it attempts to operate on registers 96, 97, 98, 99 and 100, and there is no register with address 100.
03	ILLEGAL DATA VALUE	A value contained in the query data field is not an allowable value for server. This indicates a fault in the structure of the remainder of a complex request, such as that the implied length is incorrect. It specifically does NOT mean that a data item submitted for storage in a register has a value outside the expectation of the application program, since the MODBUS protocol is unaware of the significance of any particular value of any particular register.
04	SERVER DEVICE FAILURE	An unrecoverable error occurred while the server was attempting to perform the requested action.
05	ACKNOWLEDGE	Specialized use in conjunction with programming commands. The server has accepted the request and is processing it, but a long duration of time will be required to do so. This response is returned to prevent a timeout error from occurring in the client. The client can next issue a Poll Program Complete message to determine if processing is completed.
06	SERVER DEVICE BUSY	Specialized use in conjunction with programming commands. The server is engaged in processing a long–duration program command. The client should retransmit the message later when the server is free.
08	MEMORY PARITY ERROR	Specialized use in conjunction with function codes 20 and 21 and reference type 6, to indicate that the extended file area failed to pass a consistency check. The server attempted to read record file, but detected a parity error in the memory. The client can retry the request, but service may be required on the server device.
0A	GATEWAY PATH UNAVAILABLE	Specialized use in conjunction with gateways, indicates that the gateway was unable to allocate an internal communication path from the input port to the output port for processing the request. Usually means that the gateway is misconfigured or overloaded.
0B	GATEWAY TARGET DEVICE FAILED TO RESPOND	Specialized use in conjunction with gateways, indicates that no response was obtained from the target device. Usually means that the device is not present on the network.

4.4 Commissioning

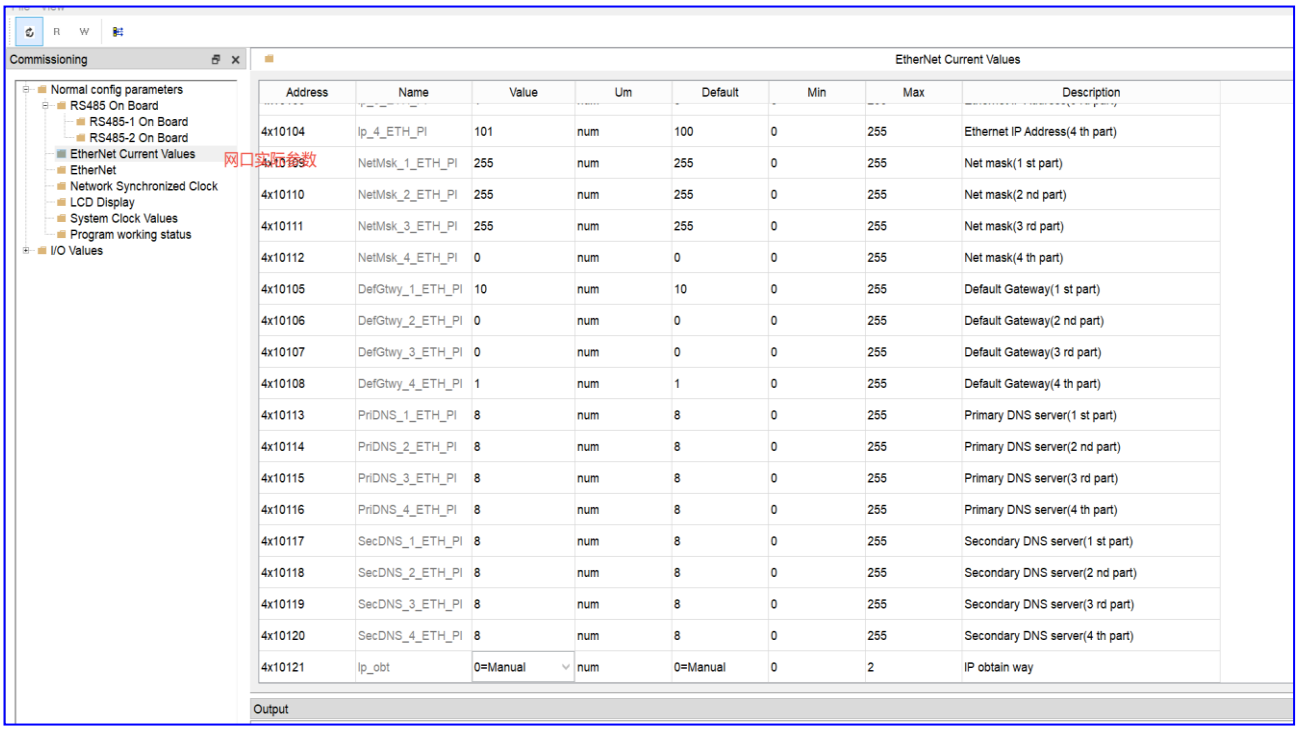
View or set internal parameters of the product in Commissioning. Gray content cannot be modified, but all other content can be modified.

1. View the 485 port parameters under RS485 On Board, or directly modify the 485 port parameters.



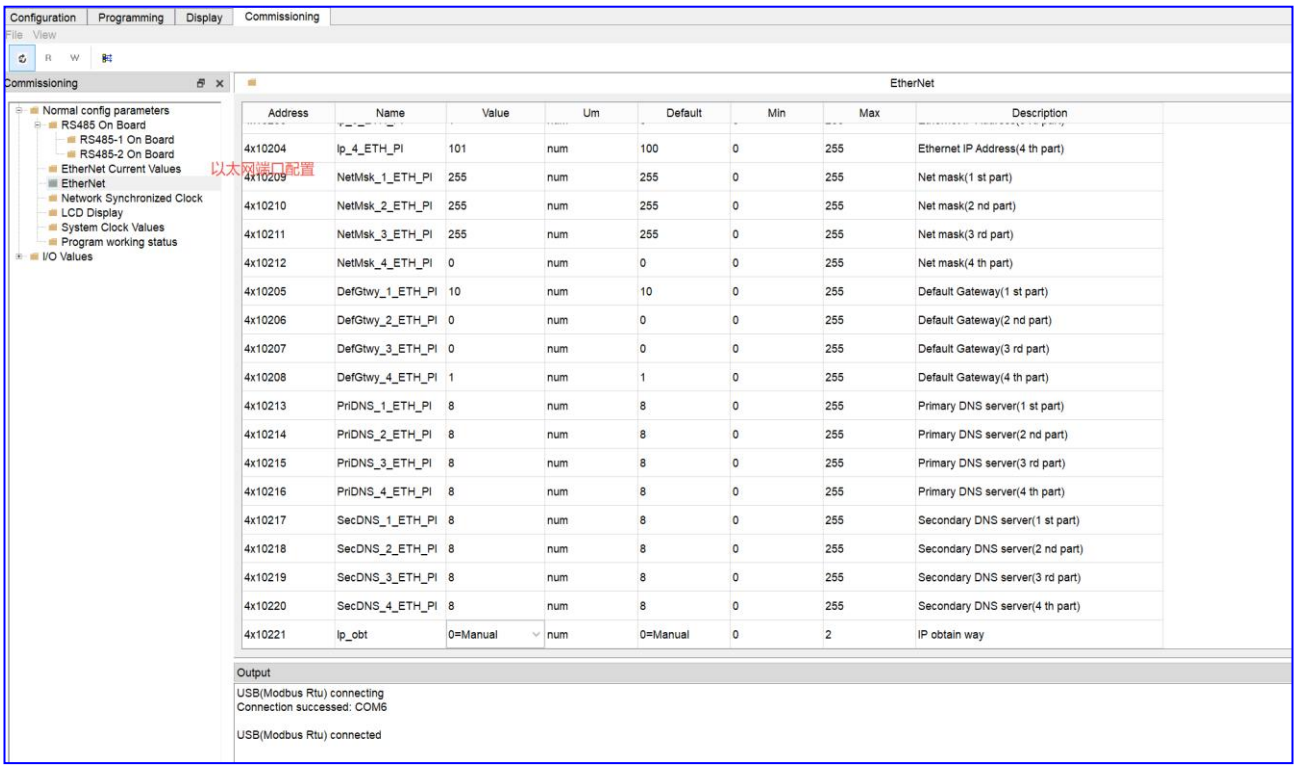
Address	Name	Value	Um	Default	Min	Max	Description
4x10301	Addr_RS485_OB1	1	num	1	0	255	RS485-1 On Board address
4x10302	Proto_RS485_OB1	0=Modbus Slave	num	0=Modbus Slave	0	1	Select RS485-1 On Board Protocol
4x10303	DataBit_RS485_OB1	8	num	8	8	8	RS485-1 On Board Data bit number
4x10304	StopBit_RS485_OB1	1	num	1	1	2	RS485-1 On Board stop bit number
4x10305	Parity_RS485_OB1	2=Even	num	2=Even	0	2	RS485-1 On Board parity number
4x10306	Baud_RS485_OB1	4=115200	num	4=115200	0	5	RS485-1 On Board baud rate number
4x10401	Addr_RS485_OB2	1	num	1	0	255	RS485-2 On Board address
4x10402	Proto_RS485_OB2	0=Modbus Slave	num	0=Modbus Slave	0	1	Select RS485-2 On Board Protocol
4x10403	DataBit_RS485_OB2	8	num	8	8	8	RS485-2 On Board Data bit number
4x10404	StopBit_RS485_OB2	1	num	1	1	2	RS485-2 On Board stop bit number
4x10405	Parity_RS485_OB2	2=Even	num	2=Even	0	2	RS485-2 On Board parity number
4x10406	Baud_RS485_OB2	4=115200	num	4=115200	0	5	RS485-2 On Board baud rate number

2. View the actual parameters of the Ethernet port under EtherNet Current Values.

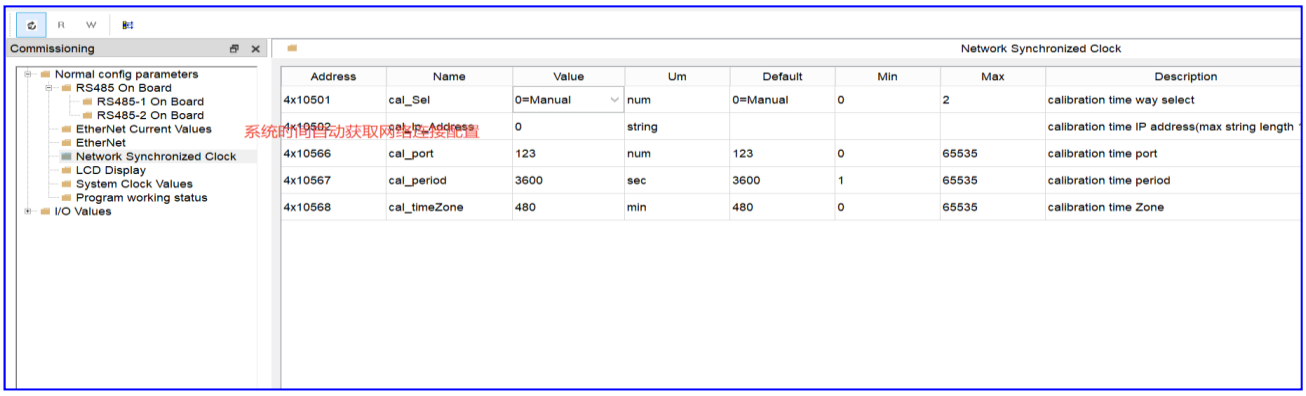


Address	Name	Value	Um	Default	Min	Max	Description
4x10104	Ip_4_ETH_PI	101	num	100	0	255	Ethernet IP Address(4 th part)
4x10109	NetMsk_1_ETH_PI	255	num	255	0	255	Net mask(1 st part)
4x10110	NetMsk_2_ETH_PI	255	num	255	0	255	Net mask(2 nd part)
4x10111	NetMsk_3_ETH_PI	255	num	255	0	255	Net mask(3 rd part)
4x10112	NetMsk_4_ETH_PI	0	num	0	0	255	Net mask(4 th part)
4x10105	DefGtwy_1_ETH_PI	10	num	10	0	255	Default Gateway(1 st part)
4x10106	DefGtwy_2_ETH_PI	0	num	0	0	255	Default Gateway(2 nd part)
4x10107	DefGtwy_3_ETH_PI	0	num	0	0	255	Default Gateway(3 rd part)
4x10108	DefGtwy_4_ETH_PI	1	num	1	0	255	Default Gateway(4 th part)
4x10113	PriDNS_1_ETH_PI	8	num	8	0	255	Primary DNS server(1 st part)
4x10114	PriDNS_2_ETH_PI	8	num	8	0	255	Primary DNS server(2 nd part)
4x10115	PriDNS_3_ETH_PI	8	num	8	0	255	Primary DNS server(3 rd part)
4x10116	PriDNS_4_ETH_PI	8	num	8	0	255	Primary DNS server(4 th part)
4x10117	SecDNS_1_ETH_PI	8	num	8	0	255	Secondary DNS server(1 st part)
4x10118	SecDNS_2_ETH_PI	8	num	8	0	255	Secondary DNS server(2 nd part)
4x10119	SecDNS_3_ETH_PI	8	num	8	0	255	Secondary DNS server(3 rd part)
4x10120	SecDNS_4_ETH_PI	8	num	8	0	255	Secondary DNS server(4 th part)
4x10121	Ip_obt	0=Manual	num	0=Manual	0	2	IP obtain way

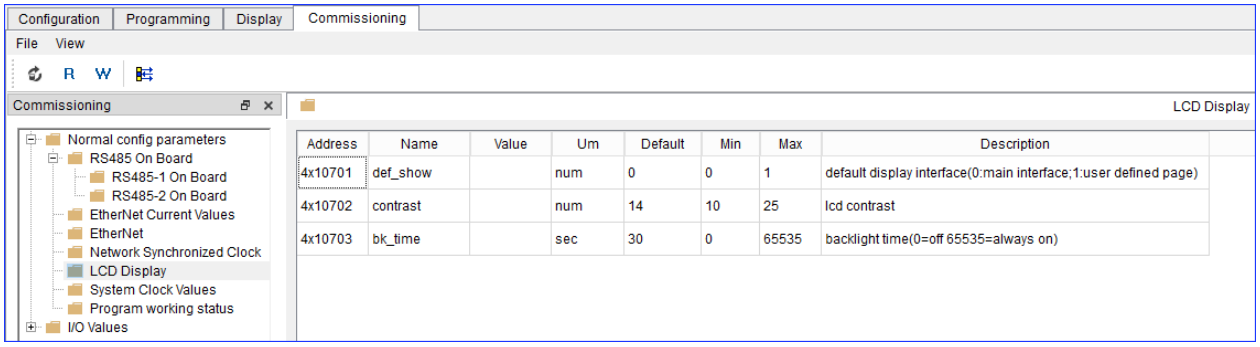
3. Configure Ethernet parameters on EtherNet



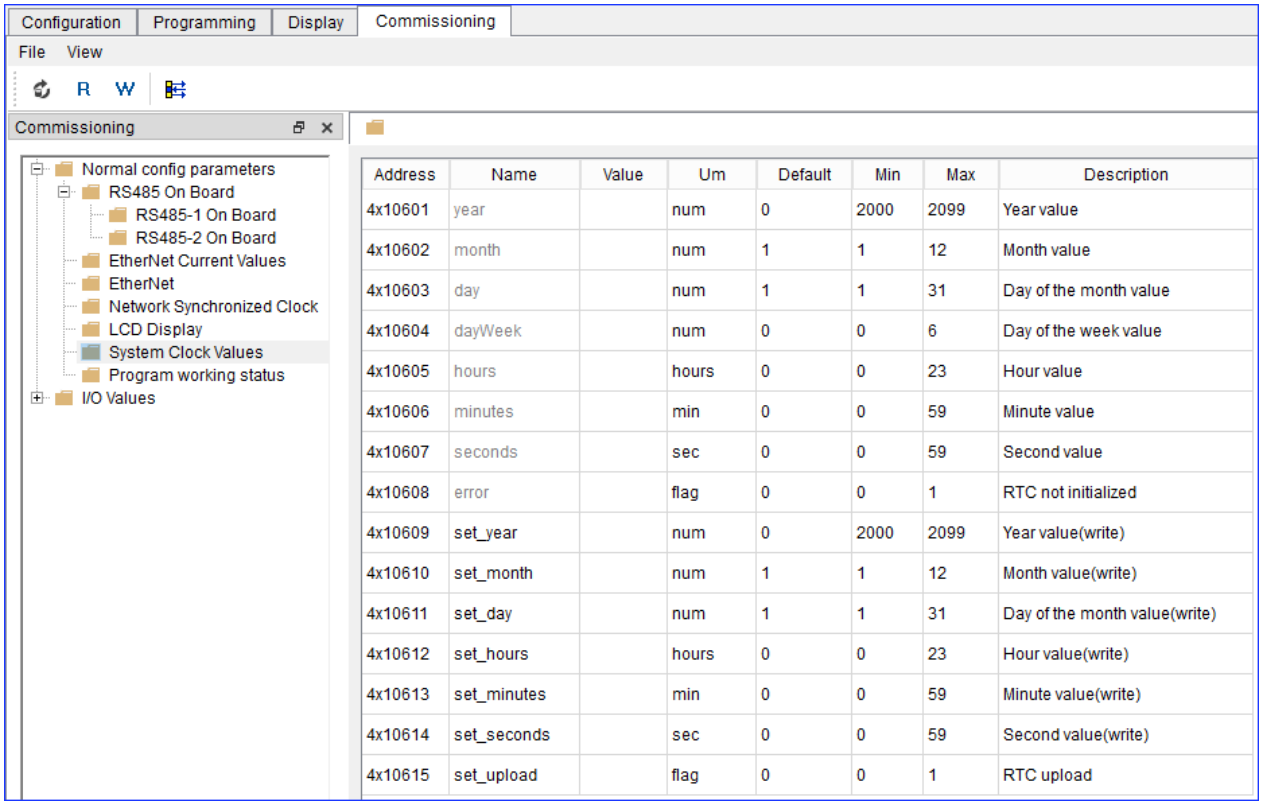
4. System time automatically retrieves network connection configuration



5. View screen parameters



6. System clock and manual settings

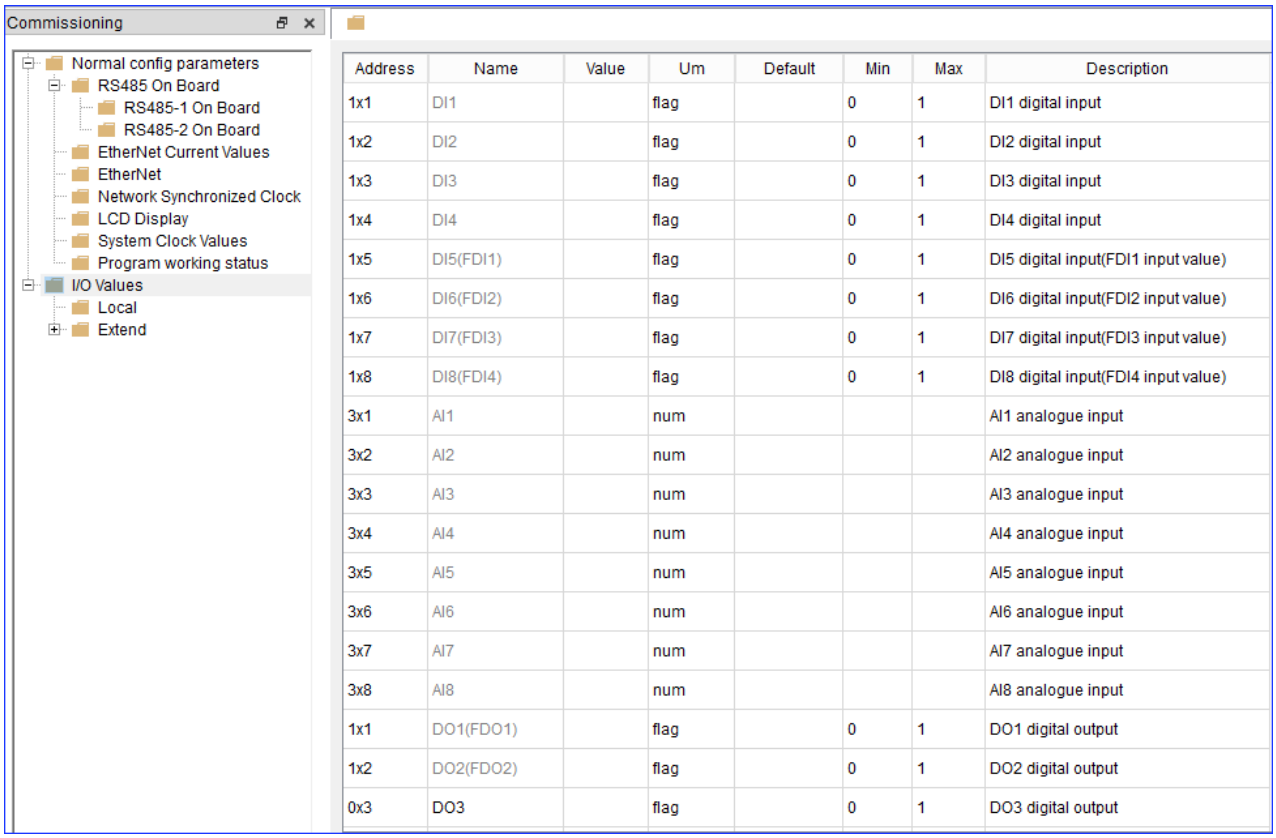


Address	Name	Value	Um	Default	Min	Max	Description
4x10601	year		num	0	2000	2099	Year value
4x10602	month		num	1	1	12	Month value
4x10603	day		num	1	1	31	Day of the month value
4x10604	dayWeek		num	0	0	6	Day of the week value
4x10605	hours		hours	0	0	23	Hour value
4x10606	minutes		min	0	0	59	Minute value
4x10607	seconds		sec	0	0	59	Second value
4x10608	error		flag	0	0	1	RTC not initialized
4x10609	set_year		num	0	2000	2099	Year value(write)
4x10610	set_month		num	1	1	12	Month value(write)
4x10611	set_day		num	1	1	31	Day of the month value(write)
4x10612	set_hours		hours	0	0	23	Hour value(write)
4x10613	set_minutes		min	0	0	59	Minute value(write)
4x10614	set_seconds		sec	0	0	59	Second value(write)
4x10615	set_upload		flag	0	0	1	RTC upload

7. View the current status of the device

Address	Name	Value	Um	Default	Min	Max	Description
4x10801	runStatus		num	0	0	65535	0:stop 1:run 0xFF:logic not loaded
4x10802	errorCode		num	0	0	65535	0:no error

8. Set the current IO value

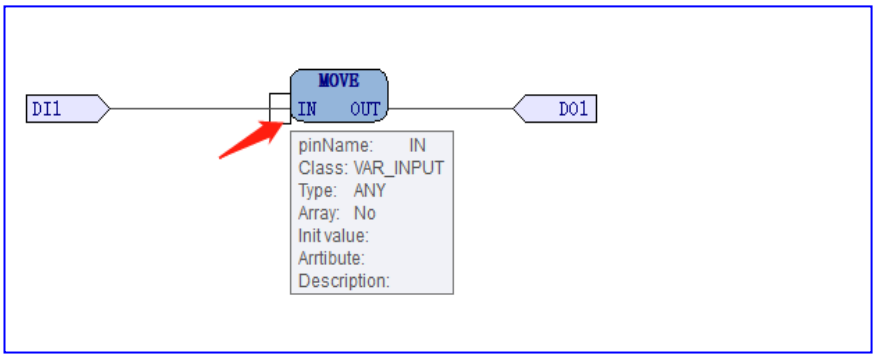


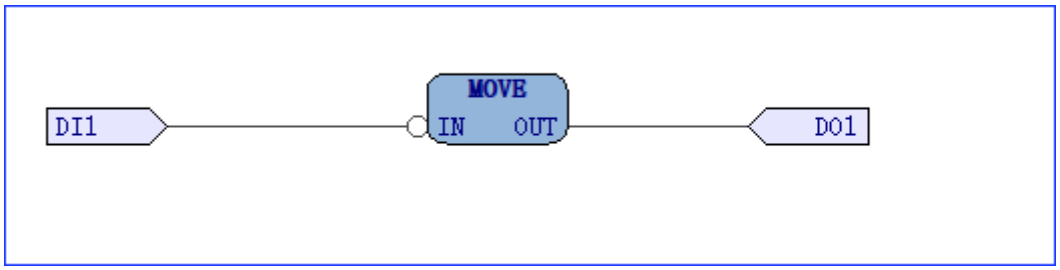
Address	Name	Value	Um	Default	Min	Max	Description
1x1	DI1		flag		0	1	DI1 digital input
1x2	DI2		flag		0	1	DI2 digital input
1x3	DI3		flag		0	1	DI3 digital input
1x4	DI4		flag		0	1	DI4 digital input
1x5	DI5(FDI1)		flag		0	1	DI5 digital input(FDI1 input value)
1x6	DI6(FDI2)		flag		0	1	DI6 digital input(FDI2 input value)
1x7	DI7(FDI3)		flag		0	1	DI7 digital input(FDI3 input value)
1x8	DI8(FDI4)		flag		0	1	DI8 digital input(FDI4 input value)
3x1	AI1		num				AI1 analogue input
3x2	AI2		num				AI2 analogue input
3x3	AI3		num				AI3 analogue input
3x4	AI4		num				AI4 analogue input
3x5	AI5		num				AI5 analogue input
3x6	AI6		num				AI6 analogue input
3x7	AI7		num				AI7 analogue input
3x8	AI8		num				AI8 analogue input
1x1	DO1(FDO1)		flag		0	1	DO1 digital output
1x2	DO2(FDO2)		flag		0	1	DO2 digital output
0x3	DO3		flag		0	1	DO3 digital output

4.5 FBD block pin inversion function

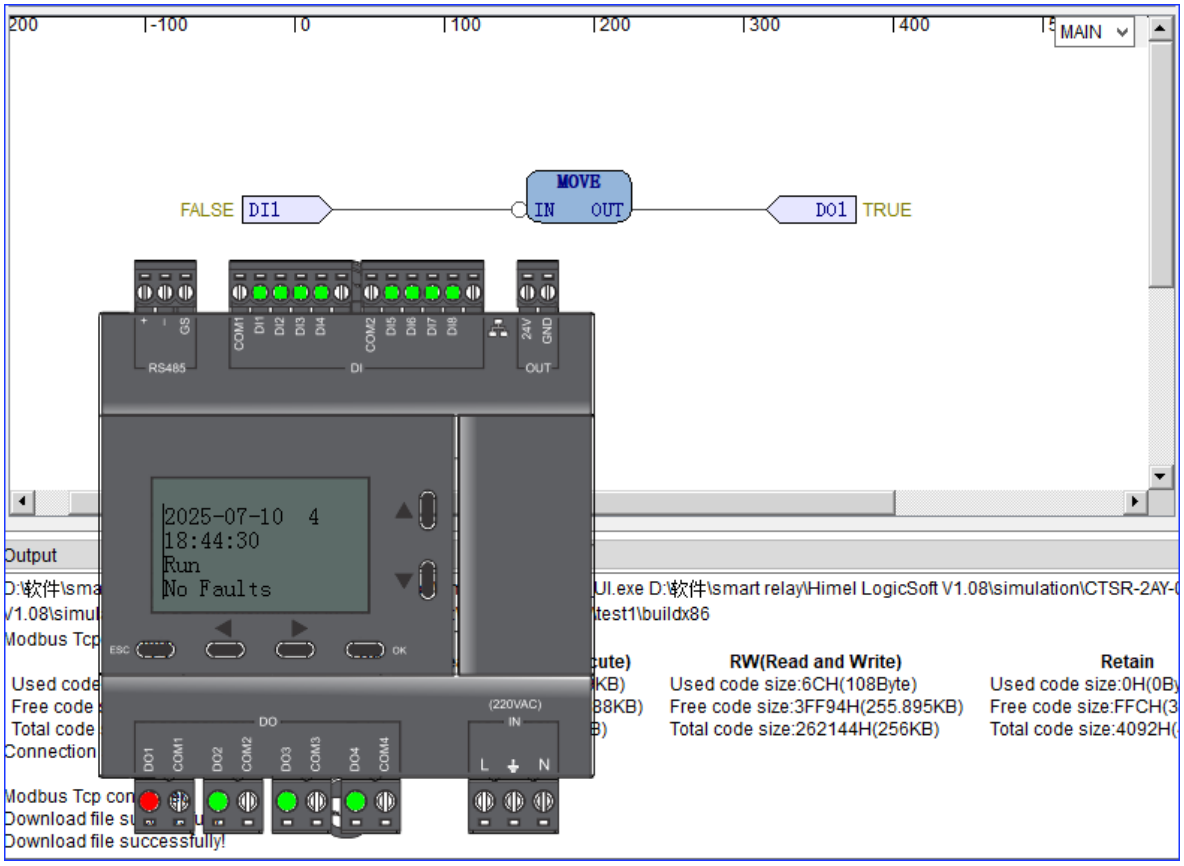
The inversion command is used to switch the inversion of the input and output pins of the FBD block. The inverted symbol is a small circle located at the corresponding input or output connection. To undo the inversion of an element, use the same cursor position and execute the same inversion command.

1. Invert input and output. Place the mouse over the IN or OUT connection, and when a small black box appears, right-click on it and select 'negate' to reverse it.





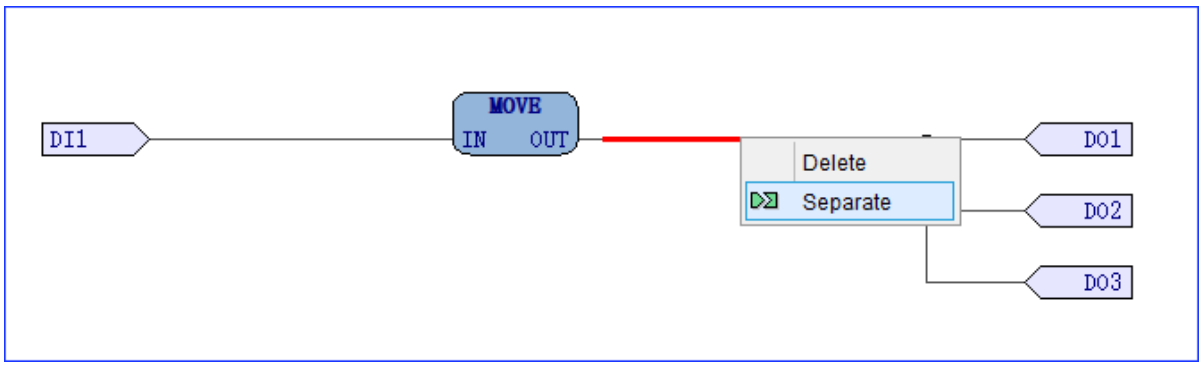
2. Compile and download, simulate and view the results as follows. When DI1 is true, DO1 is false, when DI1 is false, DO1 is true.



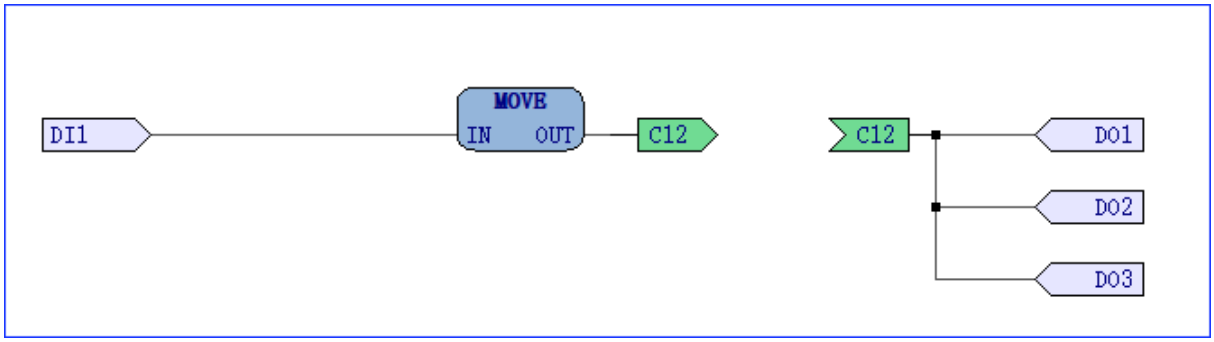
4.6 FBD connection break function

FBD connection can separate the wiring in the program at will. Right click on the wiring and select 'Separate'. The disconnected connection will display two connectors. To reconnect, right-click on the connector and select 'Connect' to reconnect the previously disconnected wires.

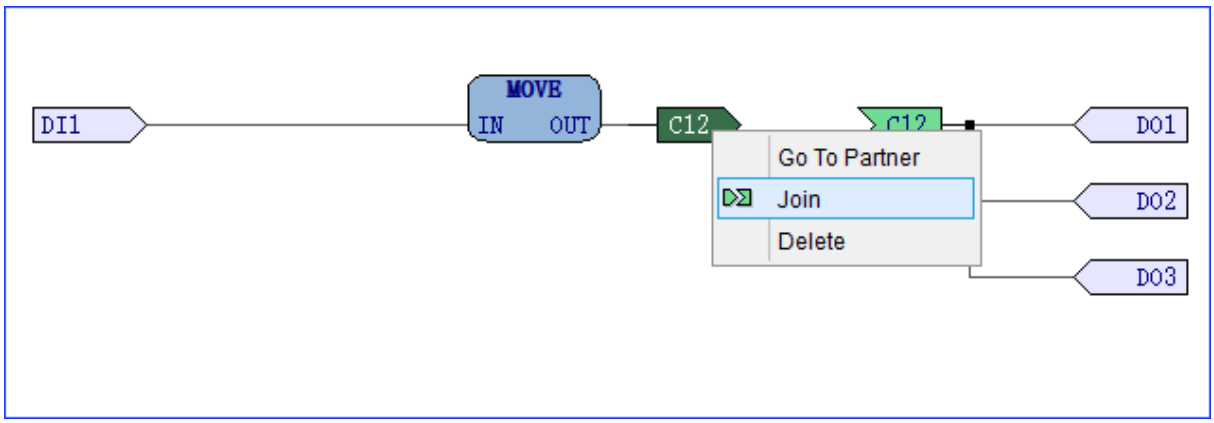
1. After selecting the wiring, right-click and choose "Separate".



2. The interrupted wiring is as follows, There will be two connectors at the separation point.

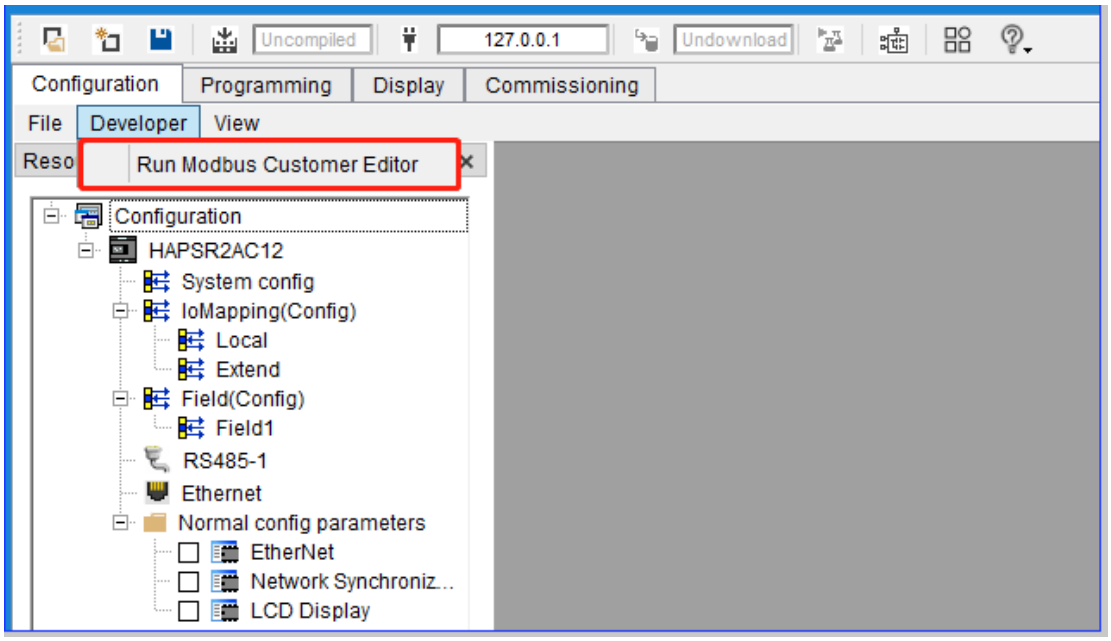


3. Right click on one of the connectors and select Jion to rewire. Select Delete to directly delete the connector, and select Go to partner to jump to the paired connector.



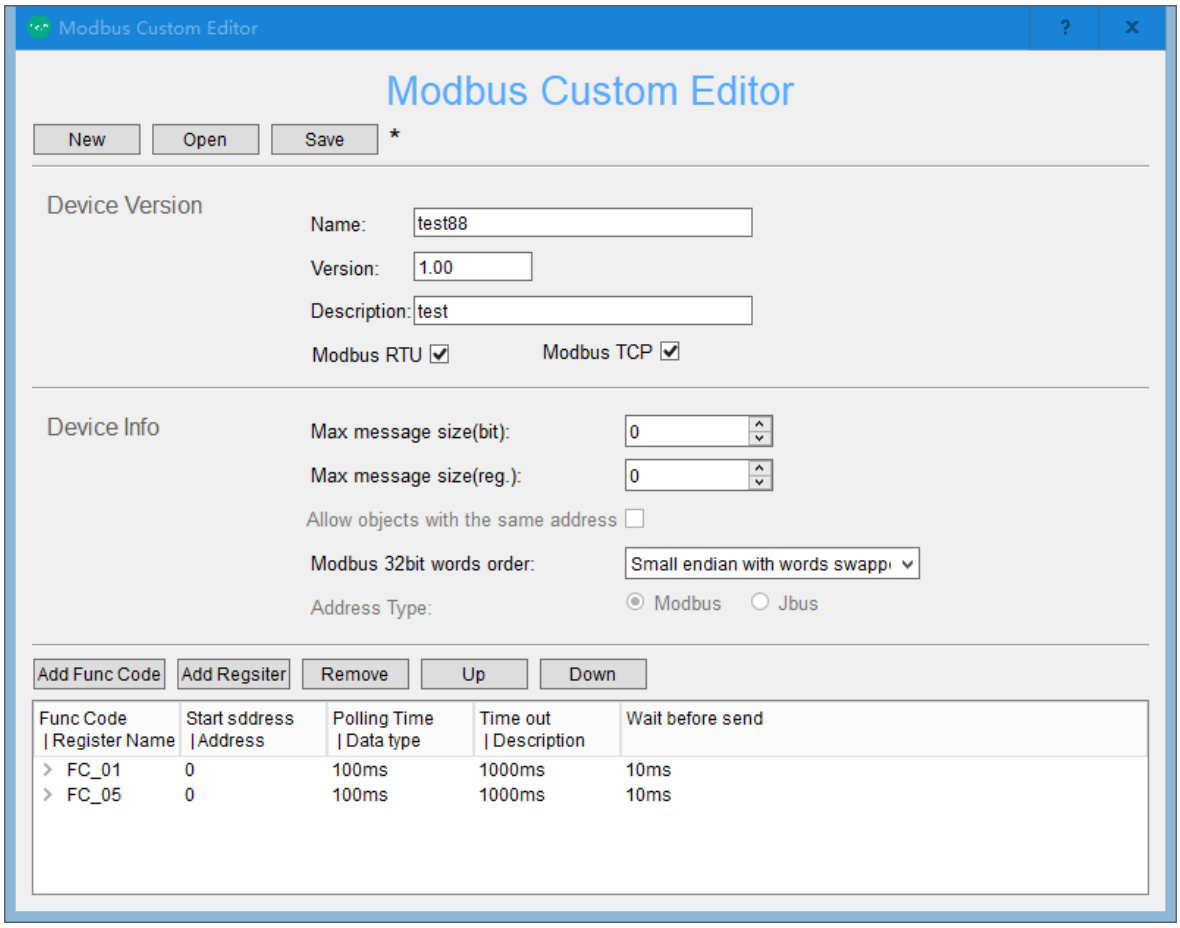
4.7 Modbus customer editor

Click on Configuration → Developer → Run Modbus Customer Editor to customize Modbus communication templates, which means you can define communication templates for a Modbus device in advance. When the project needs to communicate with this Modbus device, you can directly import this template.

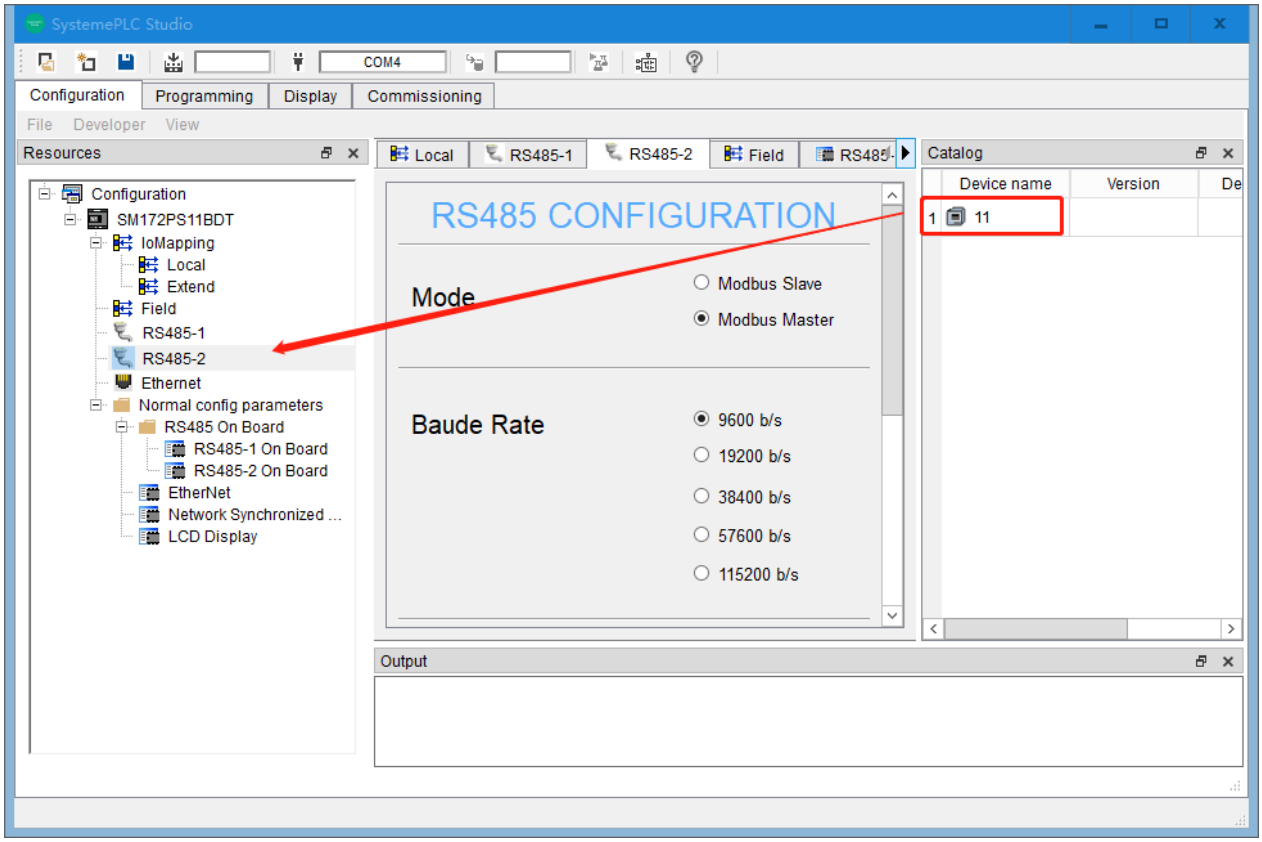


Modbus communication customization completed, click save. When modifying the communication after saving,

simply open the communication template and make the changes.



To save the template and exit, double-click on RS485-1 or RS485-2 until the previously saved template appears in the directory. Drag the template from the directory to RS485-1 or RS485-2.



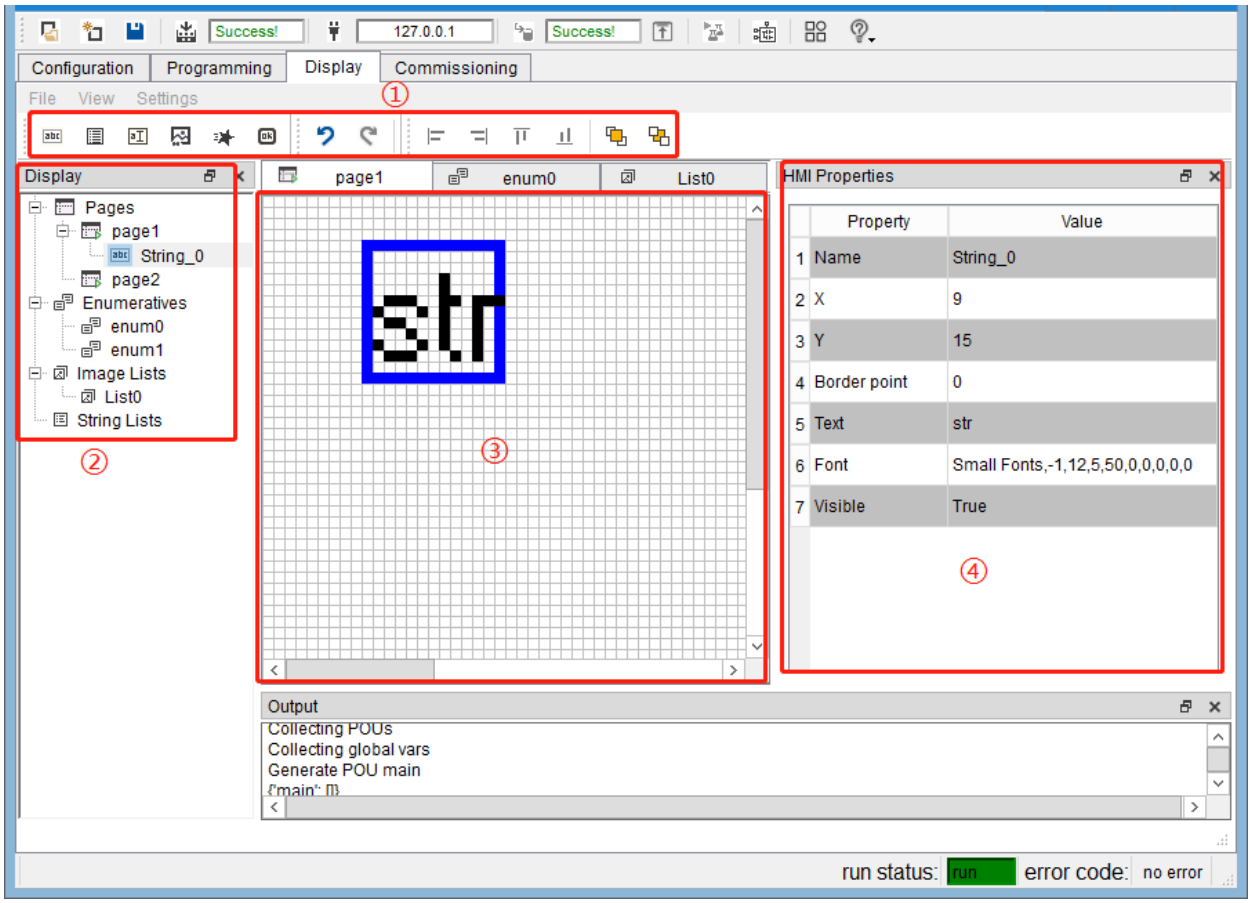
4.8 Display

4.8.1 **Overview**

Display is a user interface for embedded systems created based on HMI runtime. It implements a graphical interface in a visual manner. The pages created are displayed in Display with the same visual effect as on the final target device. With its multi-page structure, the display can support HMI (human-machine interface) applications with up to 255 pages.

Display is equipped with various controls to achieve complex display interfaces and directly interfaces with the PLC IEC1131 programming compiler to manage variables defined in the target device's PLC application.

The Display window is shown in the figure below:



① Toolbars: Includes element, alignment operations, page order, etc.

② HMI Project window: Includes pages, enumerations, image lists, and string lists.

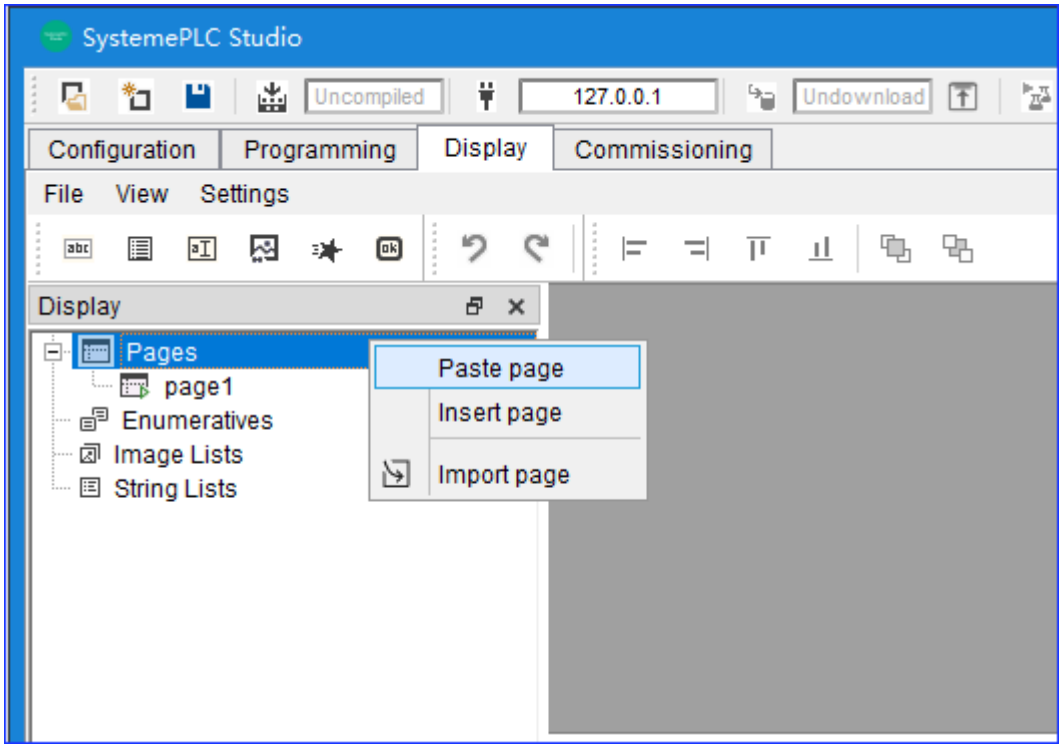
③ Editor window: The edited page appears in Display with the same display effect as on the final target device.

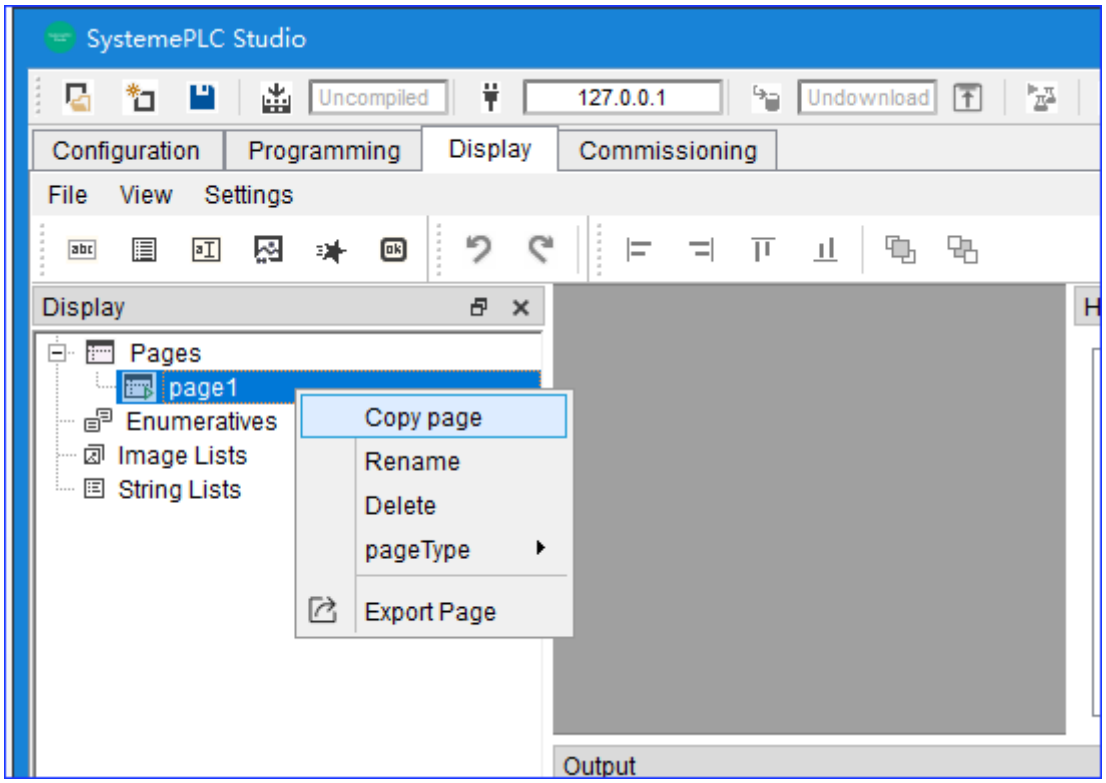
④ HMI Properties window: You can set the size of the control, its coordinate position, font format, associated variables, etc.

4.8.2 **Page introduction**

Used for configuration controls, you can copy, paste, delete, and import pages.

- Copy and paste pages: Copy and paste pages from the current project.
- Import pages: Import pages from other projects.





4.8.3 Toolbar

The toolbar is described in detail below.

Button	Description
abc	Add a new static object.
	Add a new string list
aI	Add a new edit box.

	Add a new image object.
	Add a new animation object.
	Add a new button object.
	Return to previous operation
	Restores the last action canceled by Undo.
	Align the selected objects to the left
	Align the selected objects to the right
	Align the selected objects to the bottom
	Bring to front
	Send to back

4.8.4 **HMI properties**

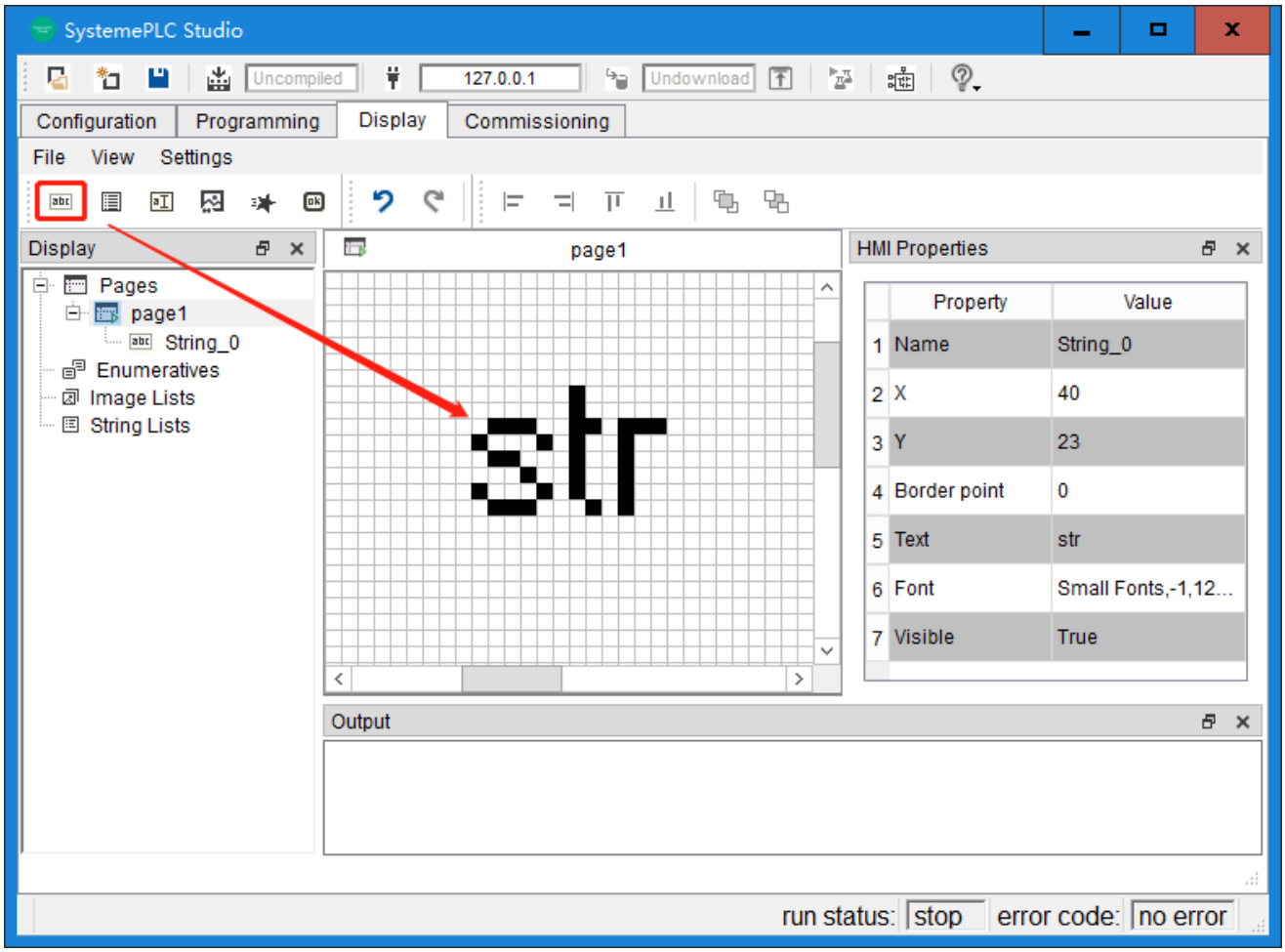
Property	Description	Applicable objects
Name	Name of object.	All of objects
X	Top-left 'x coordinate' edge relative to page.	All of objects
Y	Top-left 'y coordinate' edge relative to page.	All of objects
Width	Width of object	StringList Button Picture Animation
Height	Height of object	StringList Button Picture Animation
Text	Text or Resource ID shown in the object.	String Button
Font	Font used for drawing the text in object.	String StringList Button Edit
Border points	Border thickness.	All of objects

Action	Action executed on button pressure.	Button
Action par	Parameter associated with the action executed on button pressure. It is available only if Action is OpenPage (Action par = name of the page to open) or Call (Action par = name of the procedure to execute).	Button
String List	It contains the strings that can be displayed by the object and the range of values.	String List
Image List	It contains the images that the object can view and the value range.	Animation
Format	The format can be numeric, used to define the printf format in C language, or it can be enumerated.	Edit
Alignment	Text alignment in the object.	Button Edit
Visible	Visible status of the object. It can be constant (TRUE) or linked with a boolean variable: if boolean variable is TRUE the object is visible, otherwise it is hidden.	String Button Edit
Access	Set the read-write mode, RW (read-write) and RO (read-only).	Edit

4.8.5 Applications of static

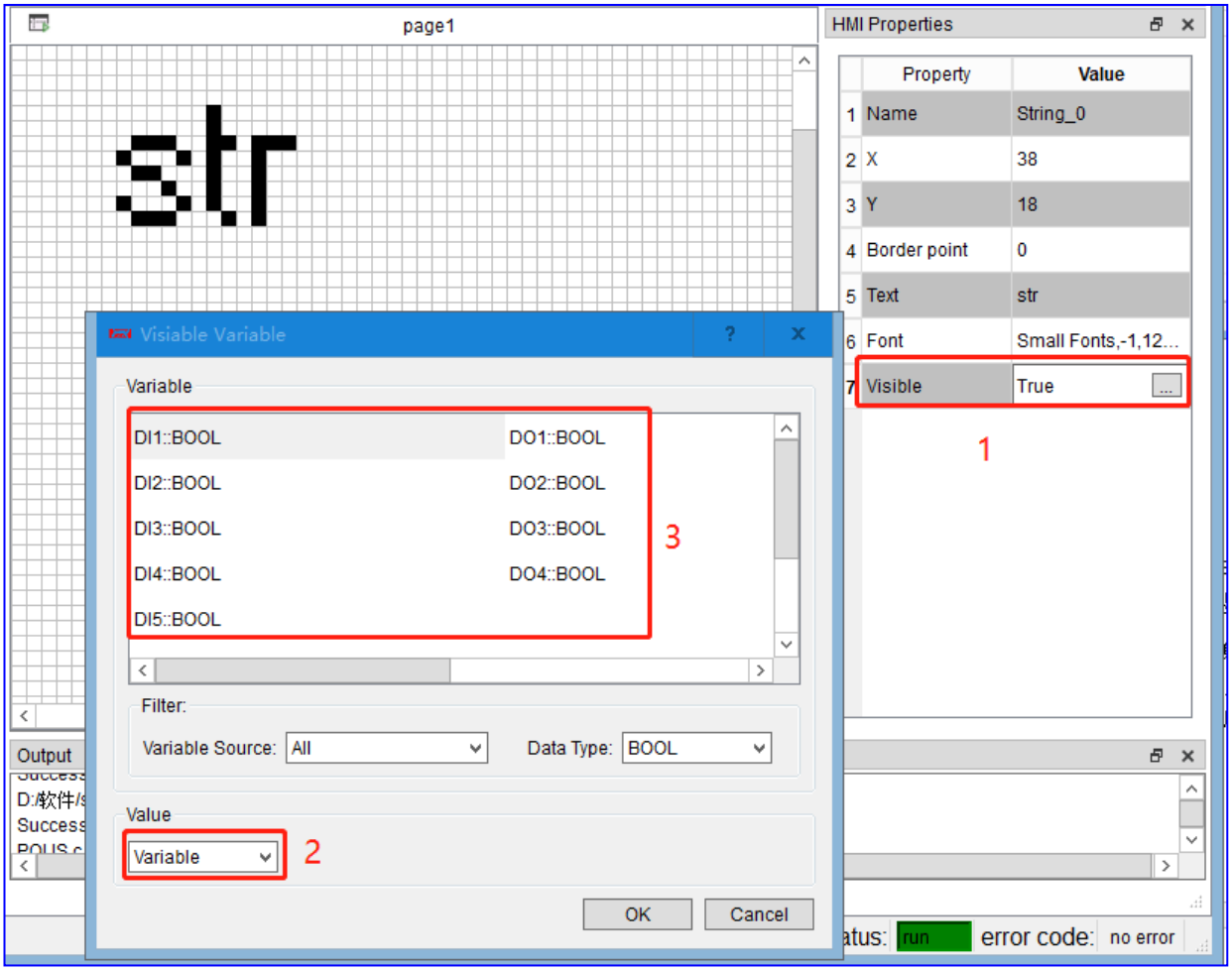
Static element display a fixed string whose content cannot be edited during execution. You must specify the text of the string directly, and then associate a Boolean variable to control the display of the string content of the static control. When the Boolean variable is true, the string content is displayed; when it is false, the string content is hidden.

1) Select the string control, and then draw it on the page.

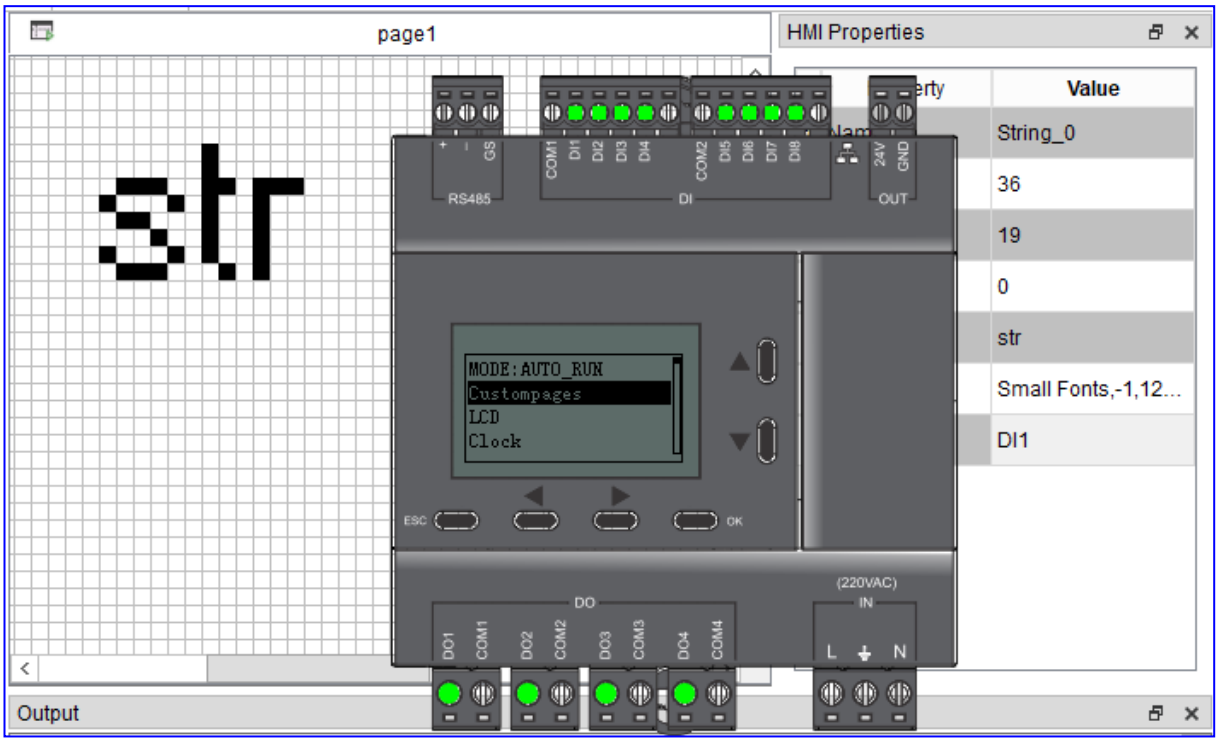


2) Configure its properties. The following mainly explains how to associate static element with variables.

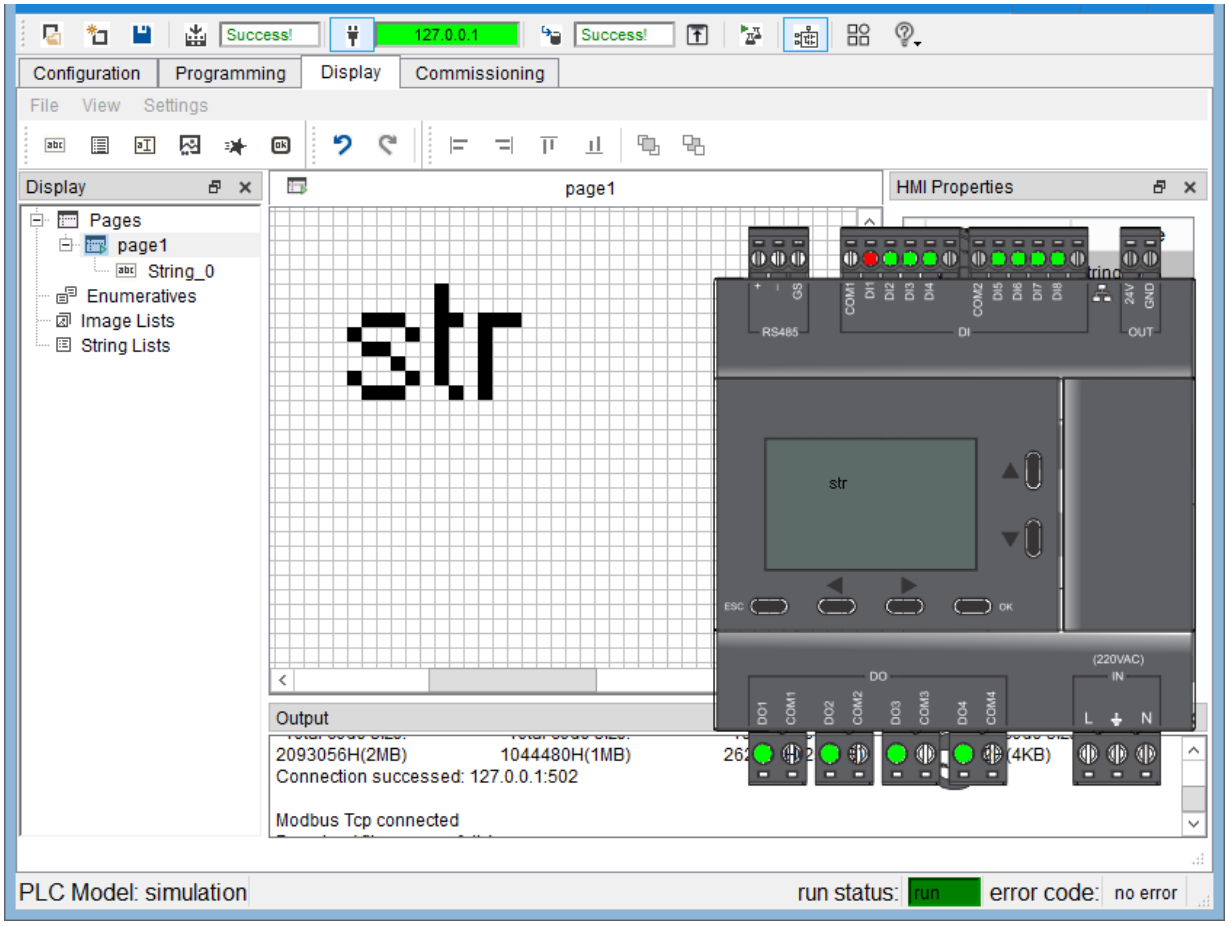
Associate variables in the order shown in the figure below: Double-click Visible, then click ..., select a variable, such as DI1.



3) After compilation, enable monitoring and simulation, and view the string display effect on the screen when DI1 is True and False.



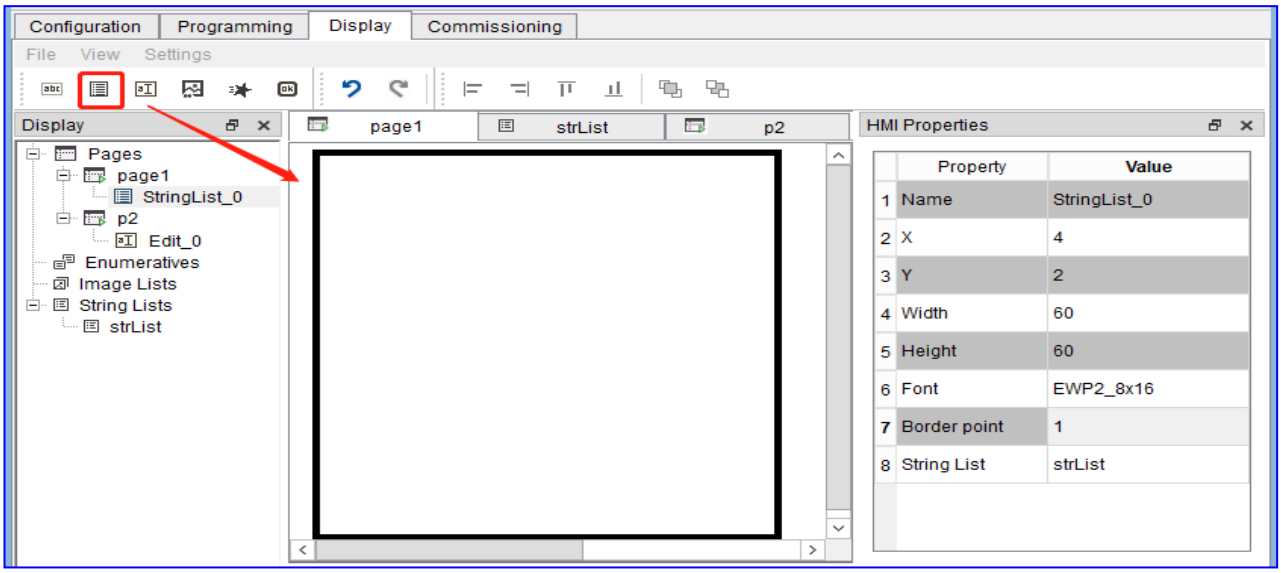
4) Enter Custompages. When DI1 is True (red), display the string content; otherwise, hide the string content.



4.8.6 Applications of string list

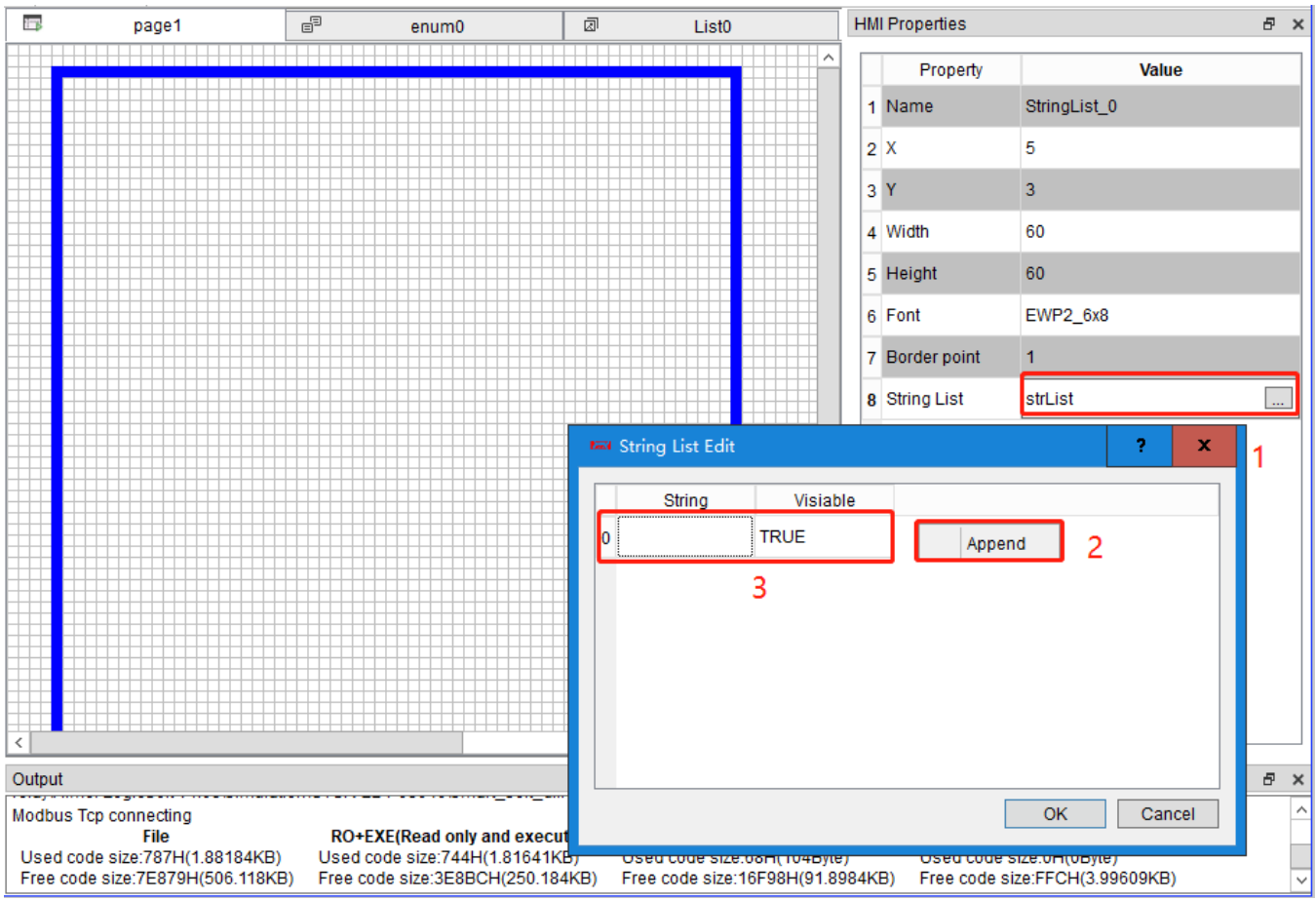
A string list displays multiple fixed strings whose content cannot be edited during execution. The text of the strings must be specified directly, and then associated with a Boolean variable to control the display of the string content in the static control. When the Boolean variable is true, the string content is displayed; when it is false, the string content is hidden.

1) Click the 'Insert String List' icon in the toolbar. Then, move the mouse to the active area of the page. A '+' symbol indicates the insertion point for the object. Click the left mouse button to insert the object into the grid. The coordinates, height, and width can be set arbitrarily.

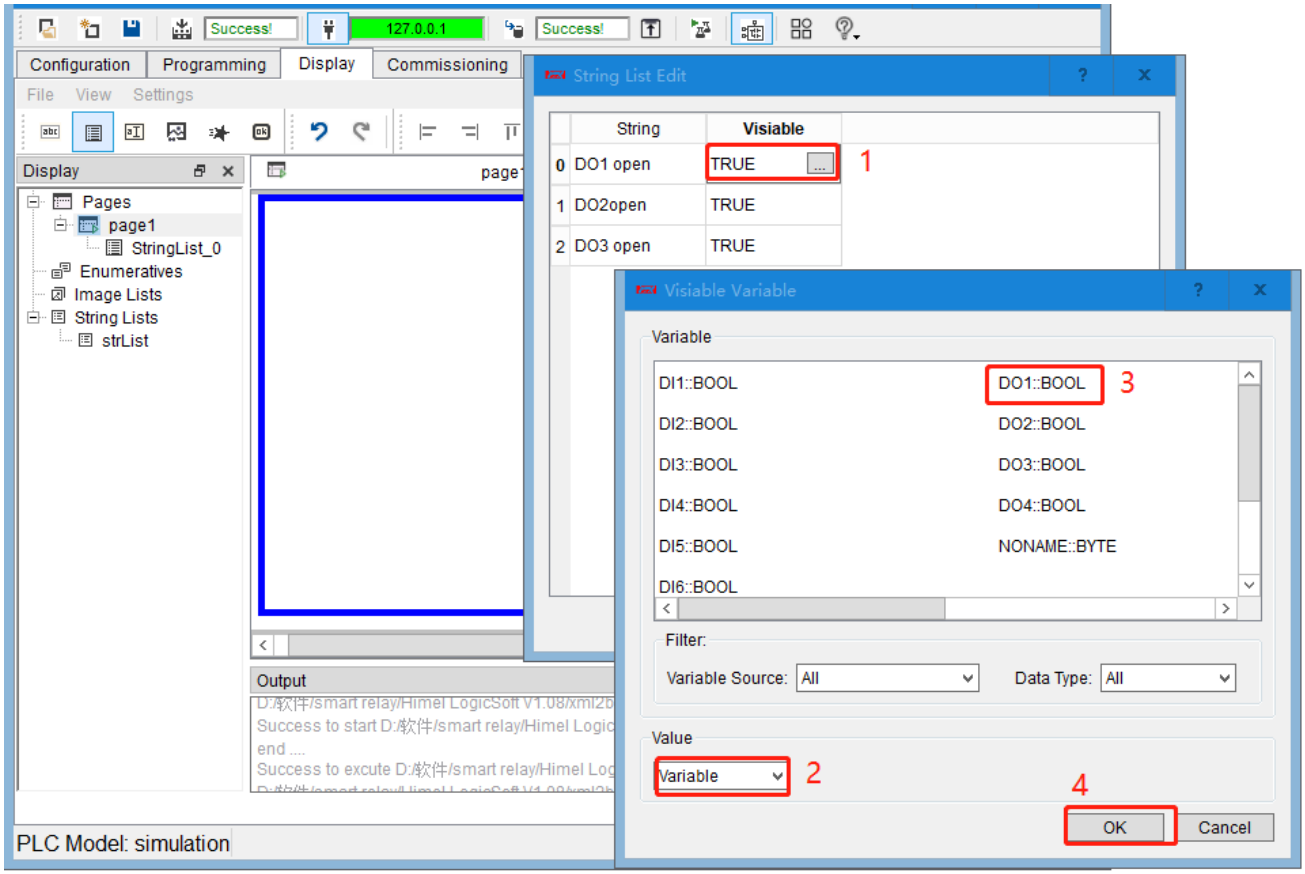


2) Create an associated string list. The string list can be associated with variables or constants.

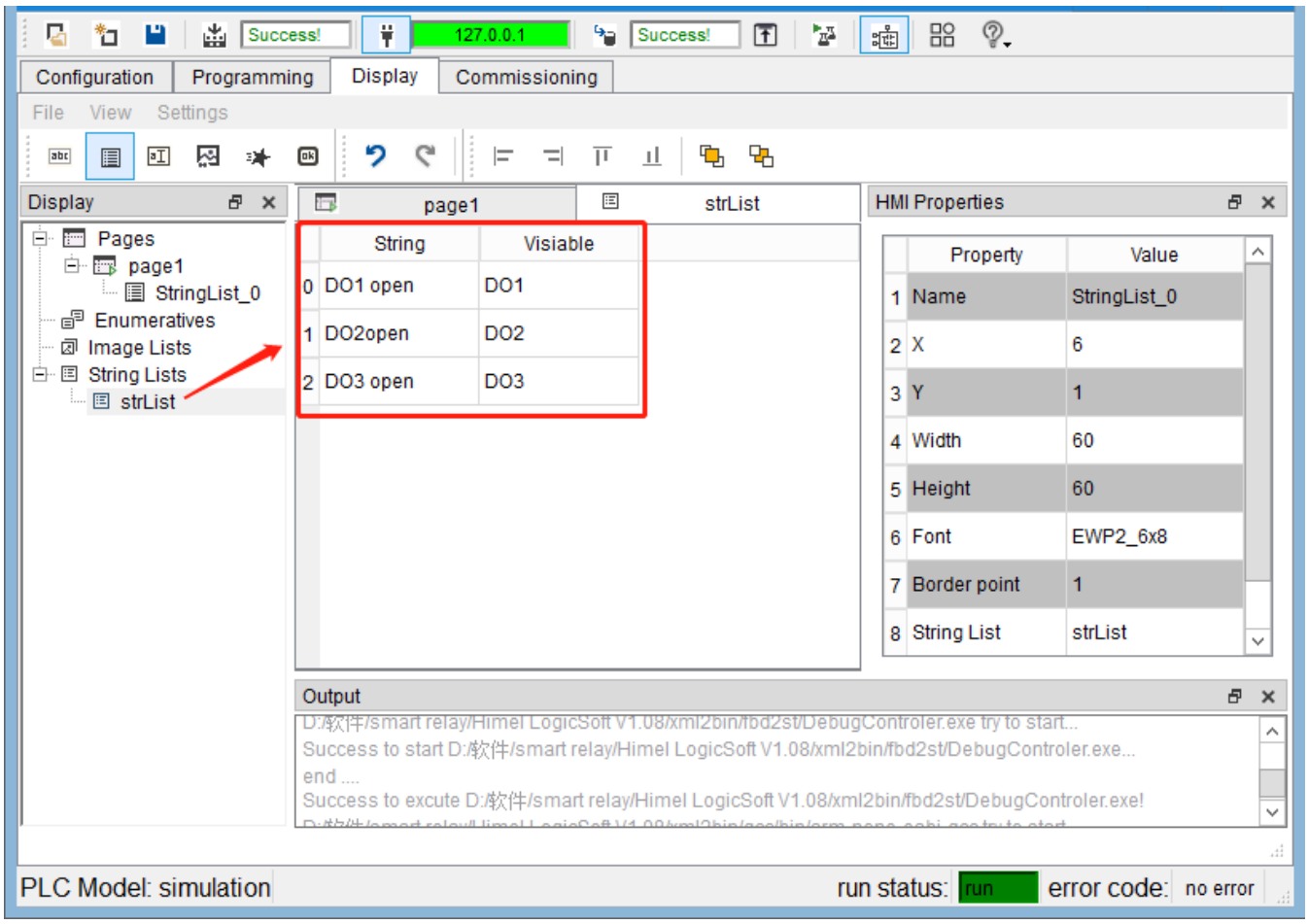
Double-click the 'String List' field, click…, and then right-click anywhere in the String List Edit to add a string.



Modify the string names, for example, change them to DO1 open, DO2 open, and DO3 open. Associate the variables as shown in the figure below.



You can view the associated variables in strList in the project tree. Do not modify the variables here.



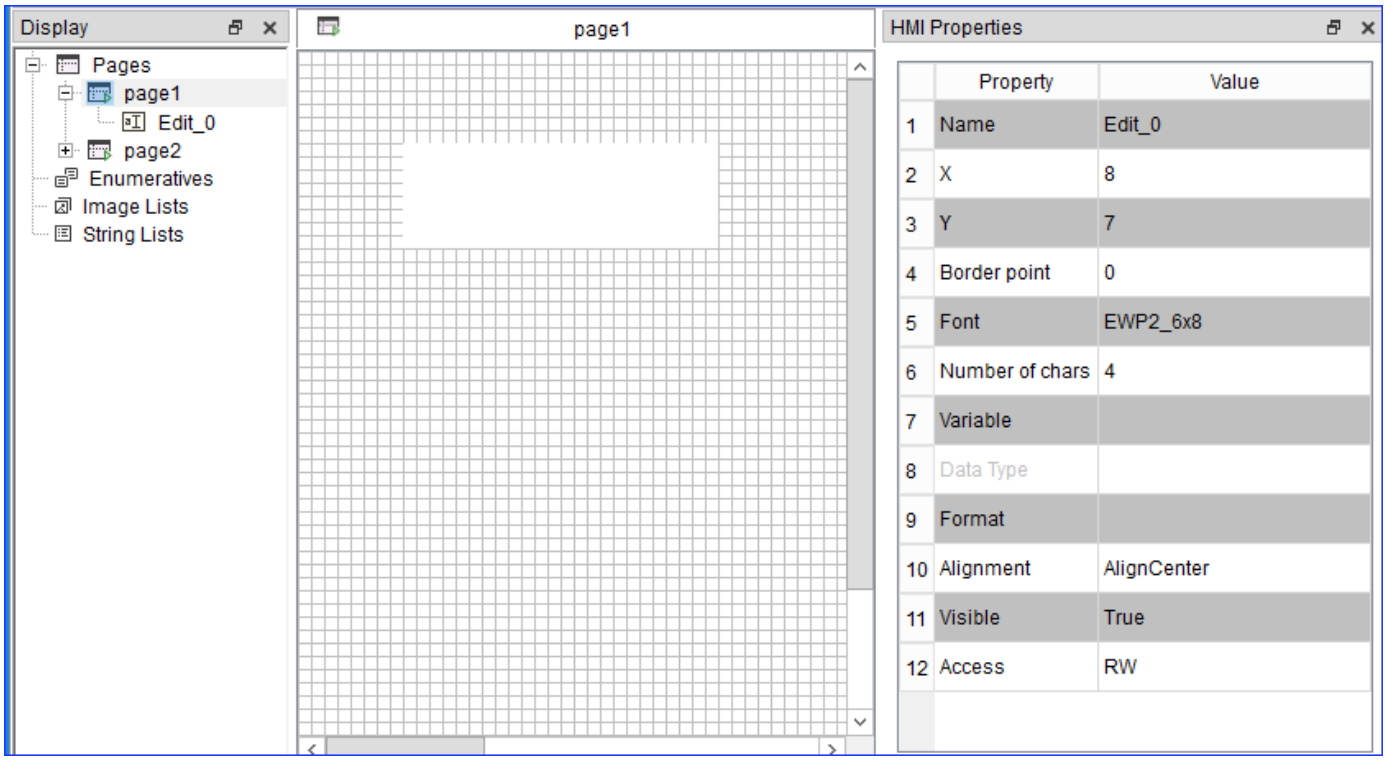
3) After compilation, enable monitoring and simulation to view the string display effect on the screen when the DO1 to DO3 variables are True and False.

4) Enter Custompages. When DO1 to DO3 are True (red), display the string content; otherwise, hide the string content.

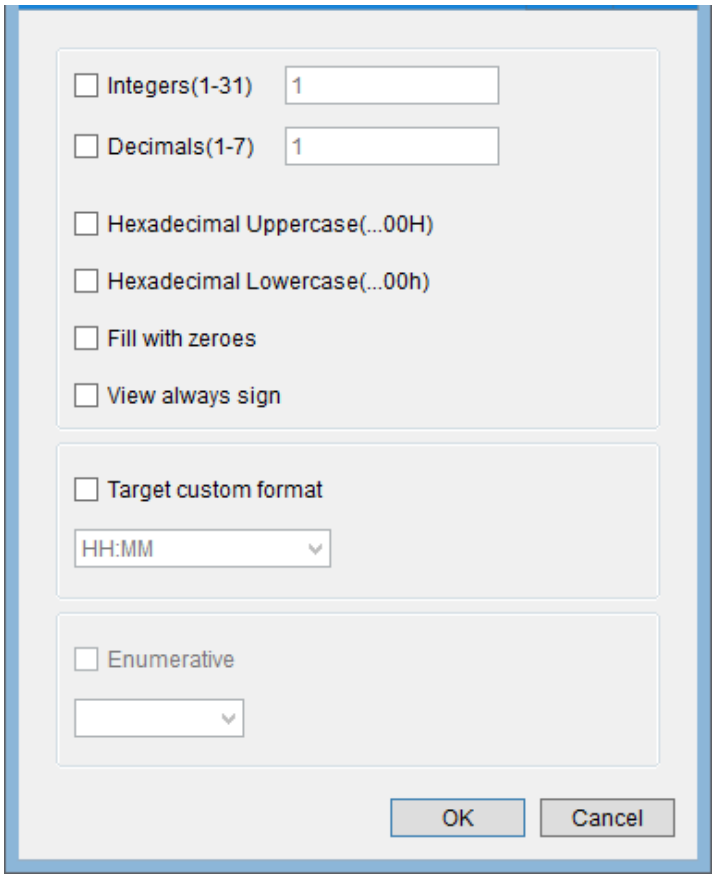
4.8.7 **Edit box**

The edit box element is a text box used to display and edit associated variables or parameters. It displays the contents of the associated variable.

1) Click the 'Insert New Edit' icon on the toolbar. Then, move the mouse to the active area of the page. A '+' symbol indicates the insertion point for the object. Left-click to insert the object into the grid.



2) Format attribute details



Integers（1-31）: Number of digits before the decimal point.

Decimals（1-7）: Number of digits after the decimal point.

Hexadecimal Uppercase（...00H）: Numbers are displayed in the format ...00H, where H is a capital letter.

Hexadecimal Lowercase（...00h）: Numbers are displayed in the form of 00h, represented by the lowercase letter h.

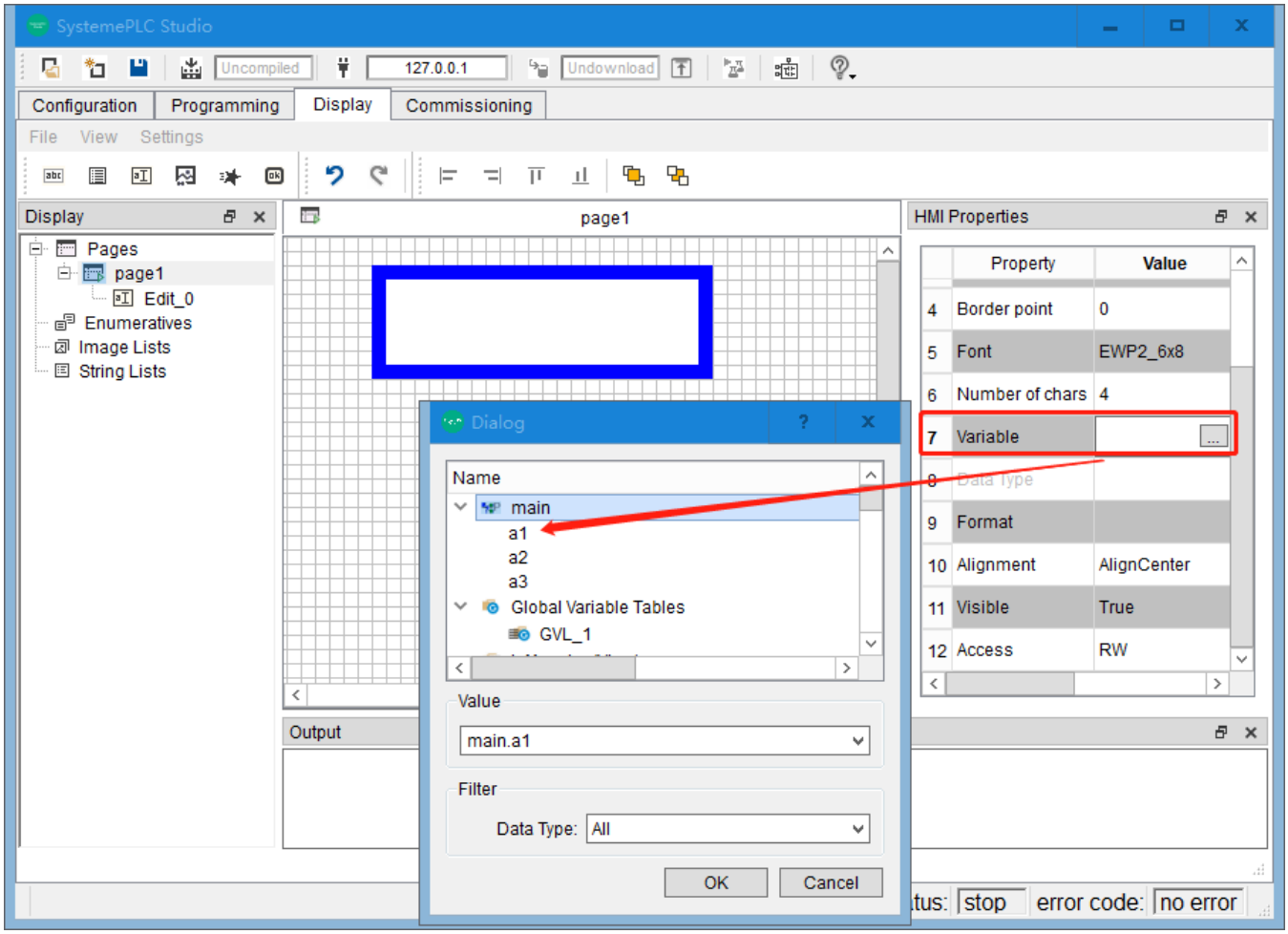
Fill with zeroes: Fill the entire edit box element with 0s in positions where there are no digits.

View always sign: Display + or - symbols in the edit box.

Enumeration type: This method allows you to select string values corresponding to the numerical values defined under the enumeration type in the project tree.

3) Edit box associated with display variable

To display values, the edit box must be associated with a variable. Click the edit box to select Variable, then select the variable in the HMI Properties window. Enter the name of the desired variable. If you do not know the name, click the '..' button to open the Property Definition window. The Property Definition window can be used to find the desired variable. Once found, select the desired variable and click OK.



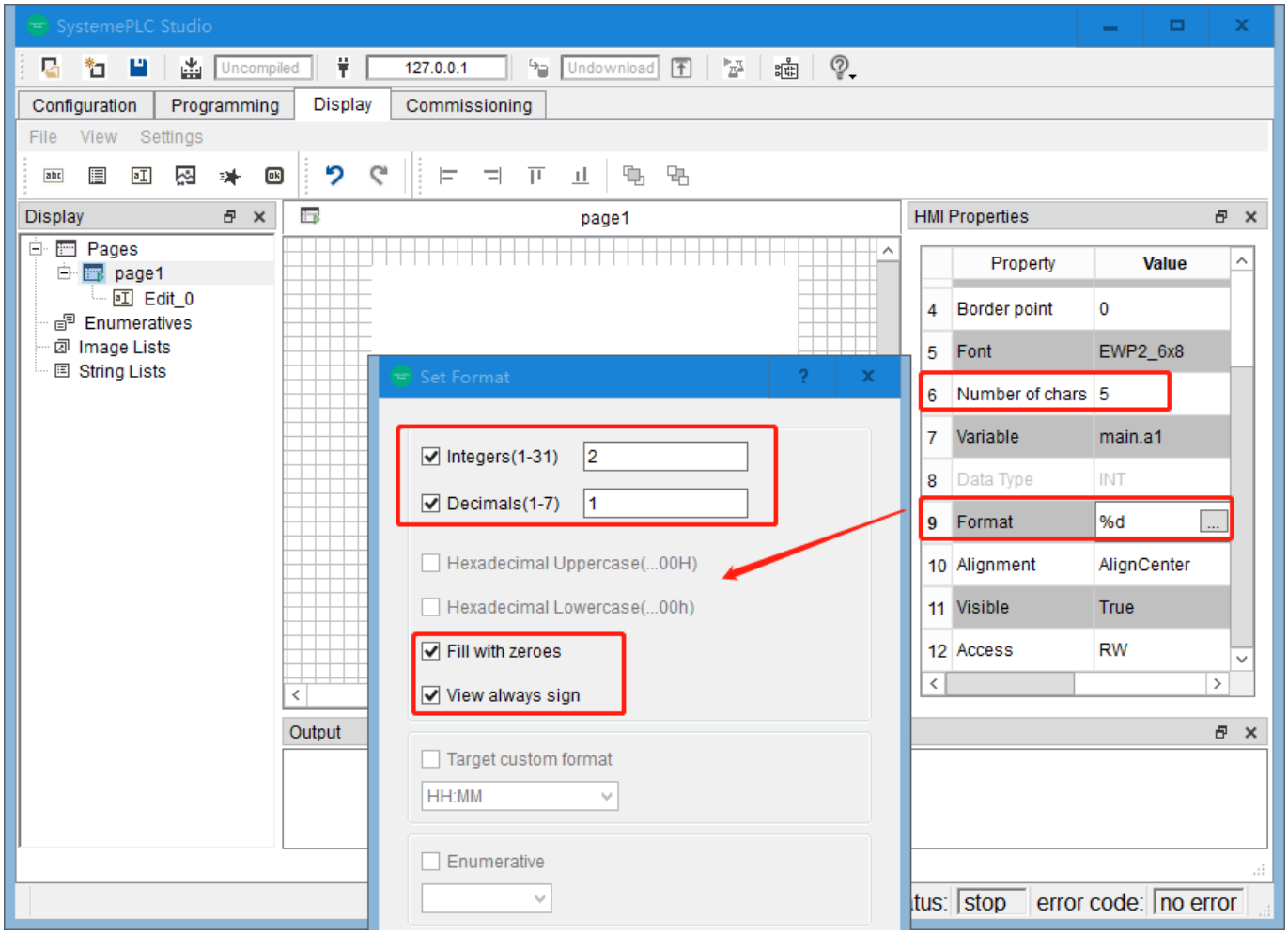
The following examples illustrate the use of Edit based on different Format attributes.

Create an integer edit box

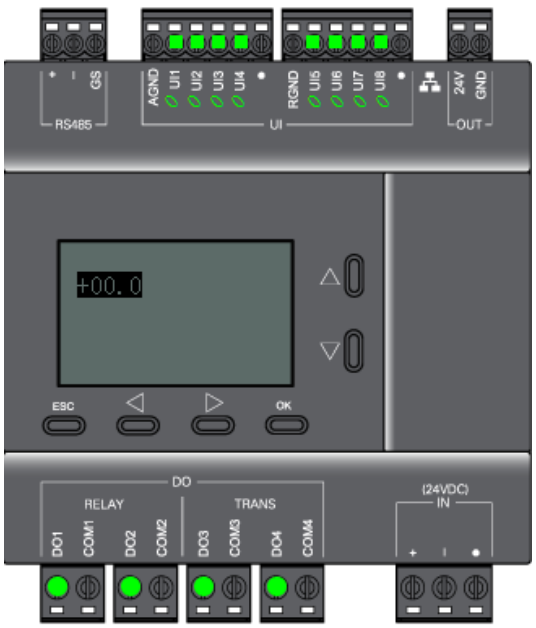
1) Click the 'Insert New Edit' icon on the toolbar. Then move the mouse to the active area of the page. A '+' symbol indicates the insertion point for the object. Click the left mouse button to insert the object into the grid.

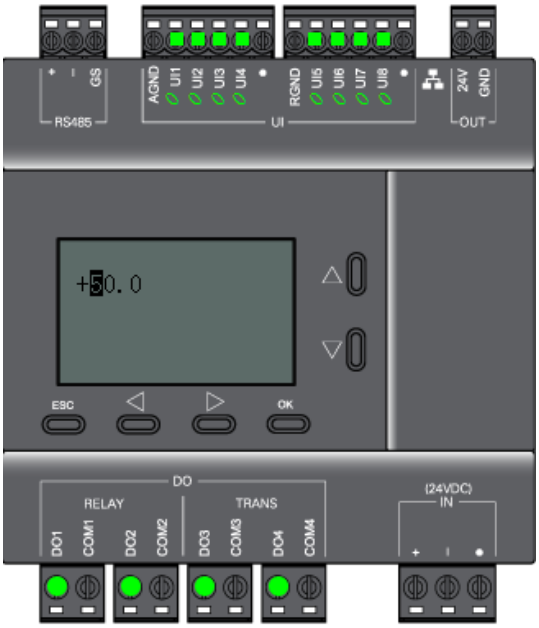
2) Double-click the Format field in the HMI properties, click Set Format, check Integers (1-31) and Decimals (1-7), and set them to 2 and 1 respectively, indicating that the Edit value consists of two integers and one decimal place. Check Fill with zeroes to fill the entire edit box control with zeros in positions without numbers. Check View always sign to allow the Edit value to be set as positive or negative.

After configuration, the Format field displays %+05.1d, which corresponds to the printf format in C language. The 5 in %+05.1d indicates the required number of characters for the Edit control, so the Number of chars field is set to 5. The Variable field is associated with an INT-type variable a1, and the Access field is set to RW, indicating that variable a1 is readable and writable.



After enabling simulation, as shown in Figure 1, press the OK key to indicate that the first digit of the data can be edited. Use the right arrow button to select the subsequent digits to be edited. The selected digits will turn black. At this point, use the up or down keys to change their values. Once editing is complete, press OK again to confirm the changes.

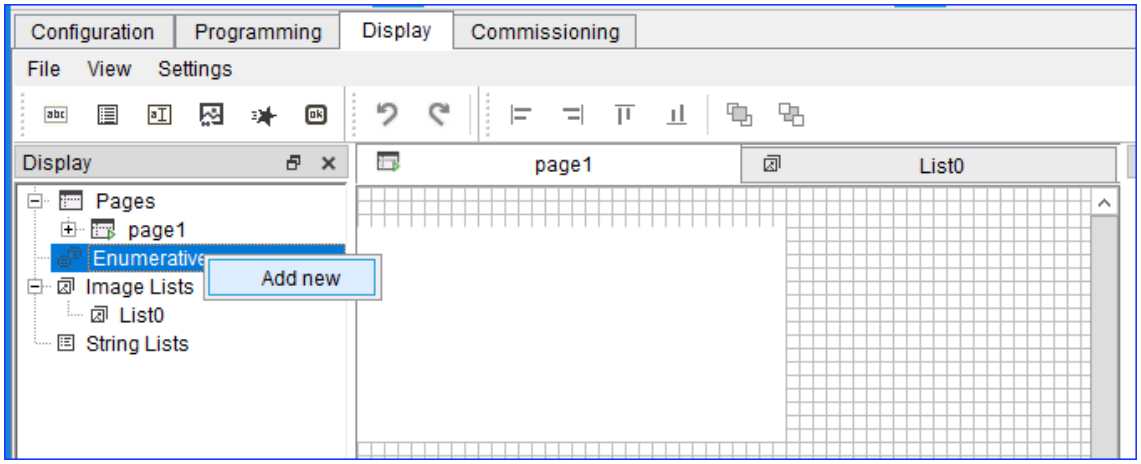


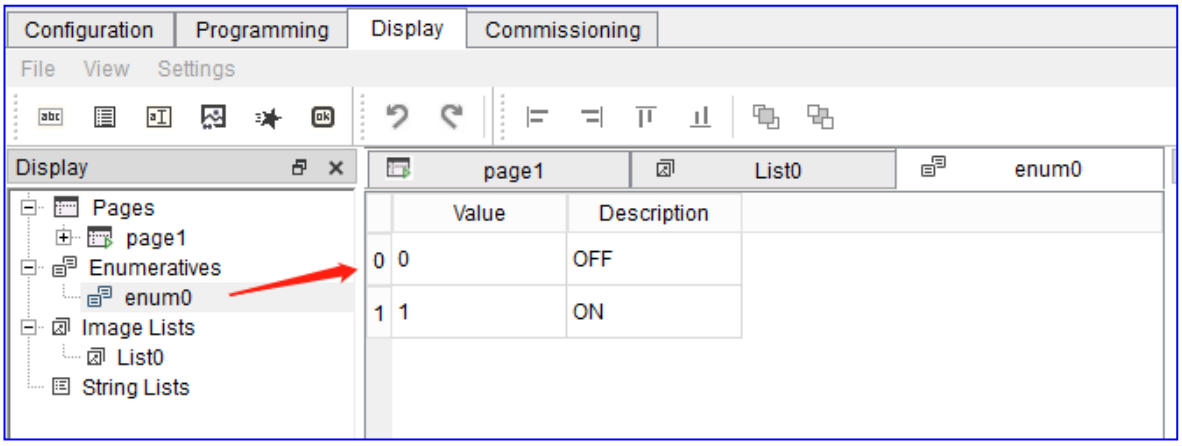


Note: The first character is a symbol character. If you edit the first character and it is a number, the edit box will change to ###, indicating that the length of the displayed content exceeds the edit box.

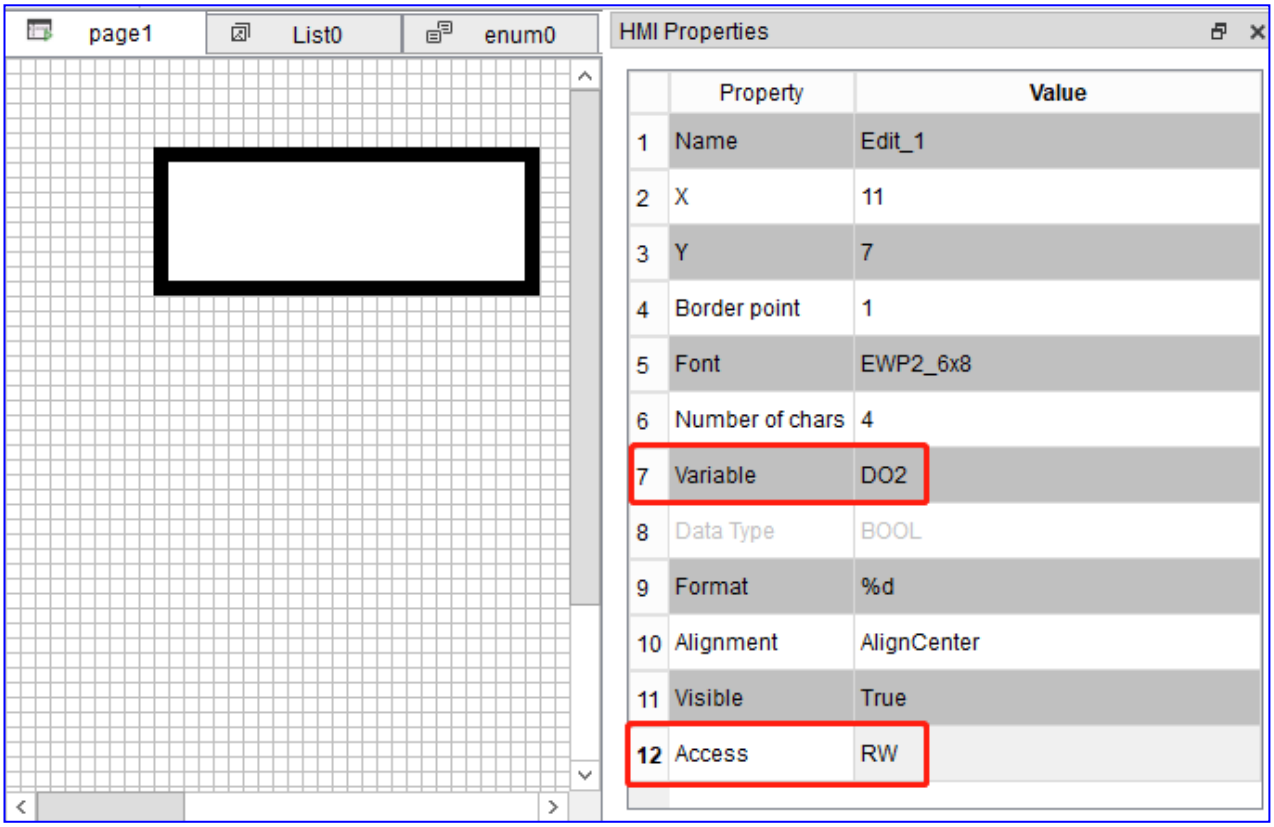
Create an enumeration format edit box

You can first create an enumeration, edit the associated variable values, and corresponding actions.

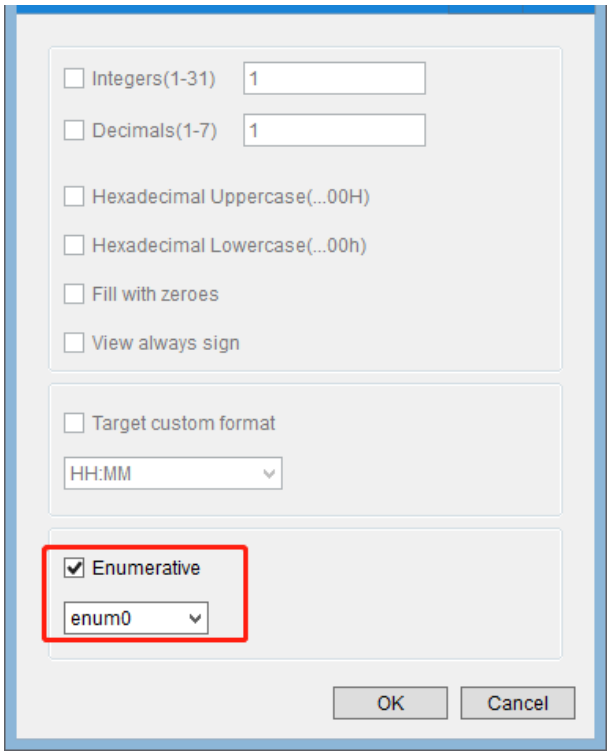




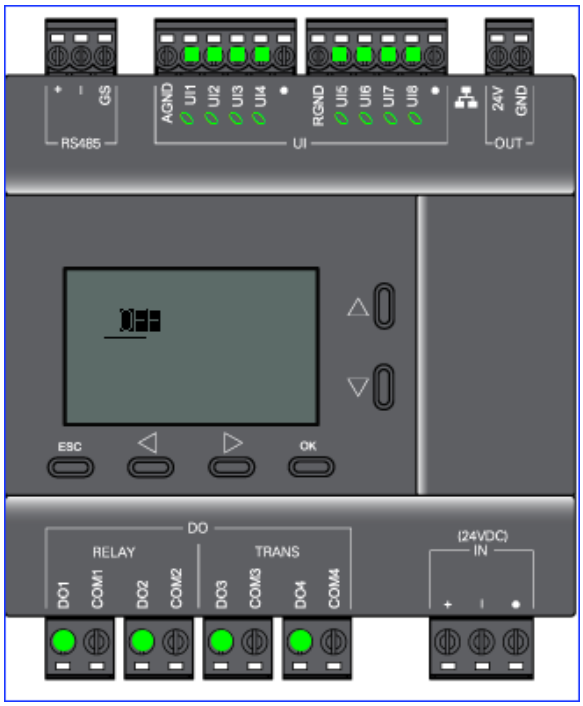
Connect DO2 to the PLC and set DO2 to be readable and writable.

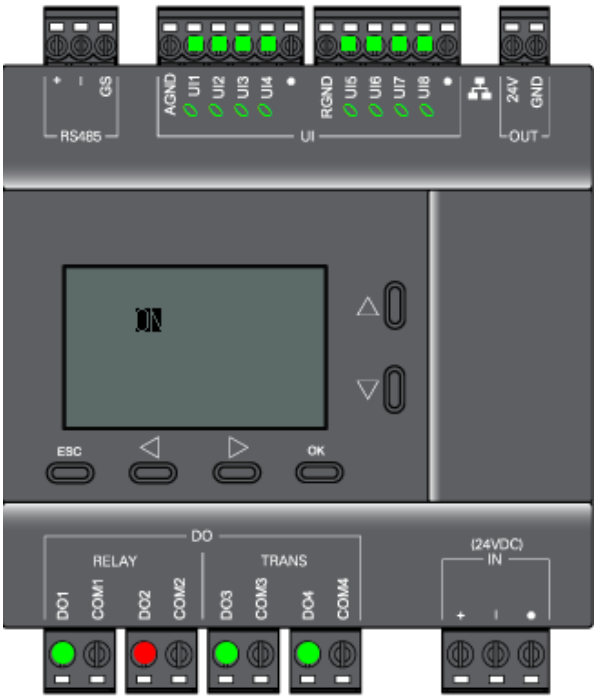


Select Enumerative in the Format field and select enum0.



After enabling simulation, enter Custompages and click OK to select Edit. The selected Edit will have a flashing underline. At this point, press the up or down key to change the DO2 status, then press OK to confirm the modification, and DO2 will change from OFF to ON status.





4.8.8 Image

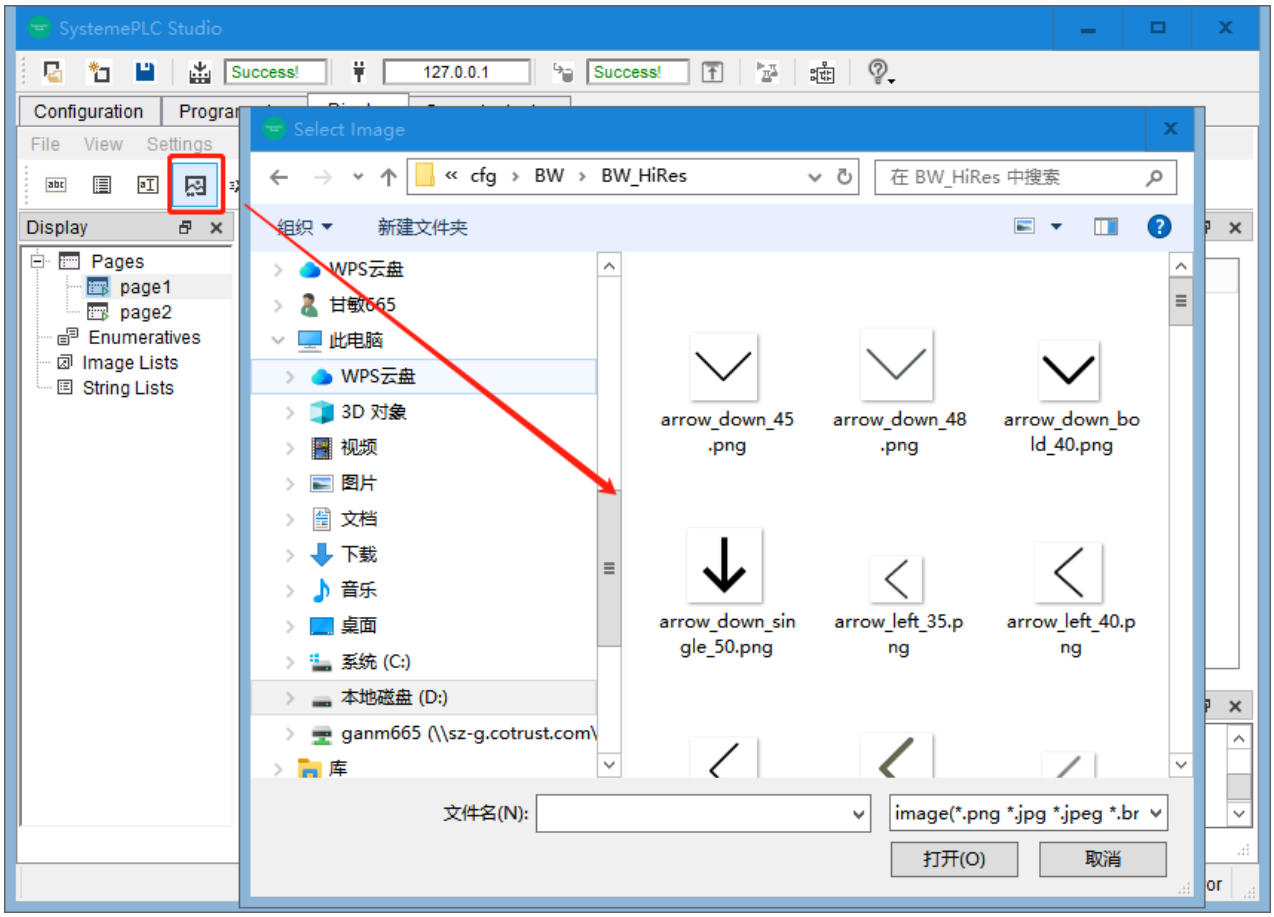
Image elements are used to display bitmap images. The following image formats are supported: *.JPG, *.JPEG, *.JPE, *.PNG.

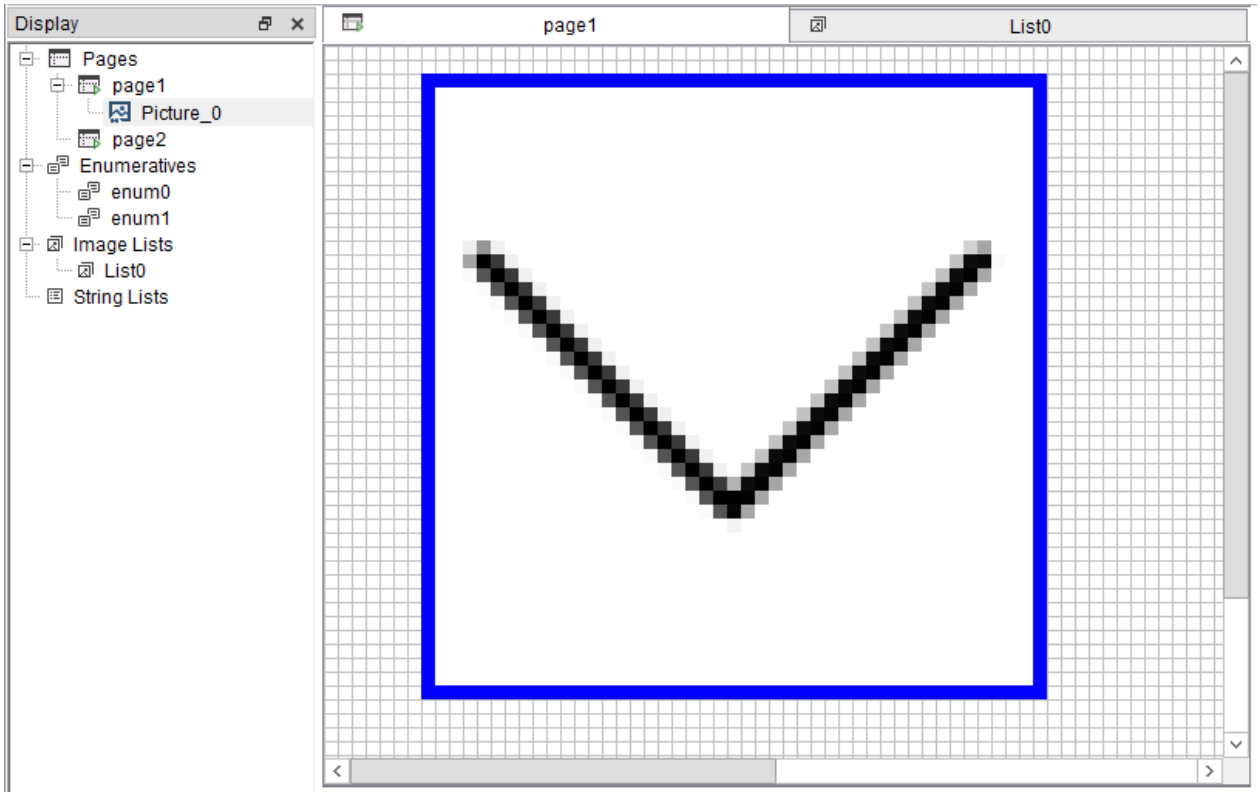
Note: When displaying images on the target device, their size will not be adjusted. Therefore, ensure that the size of the imported images does not exceed the target device's supported range.

To insert an image, click the ‘Insert Image’ icon in the HMI page toolbar. Move the mouse to the active area of the page. A ‘+’ symbol indicates the insertion point for the object. Click the left mouse button to insert the object into the

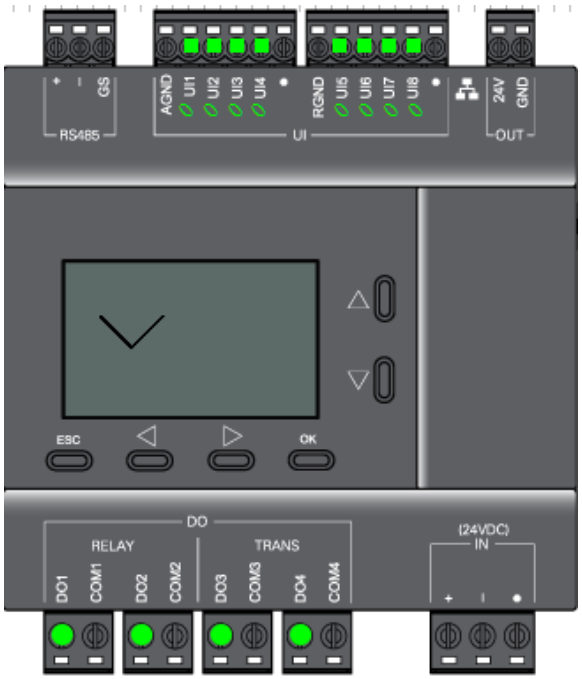
grid.

A new blank frame appears on the page. Drag the desired image from the 'Bitmap' list and drop it into the blank frame. The image control does not resize to fit the assigned bitmap dimensions.





After enabling simulation, the following is displayed:

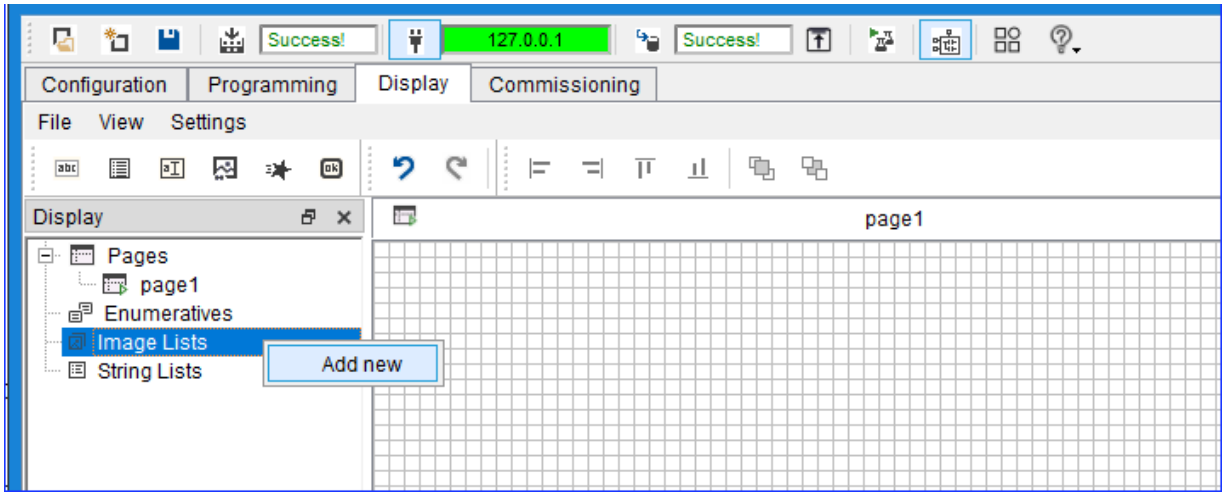


4.8.9 **Animation**

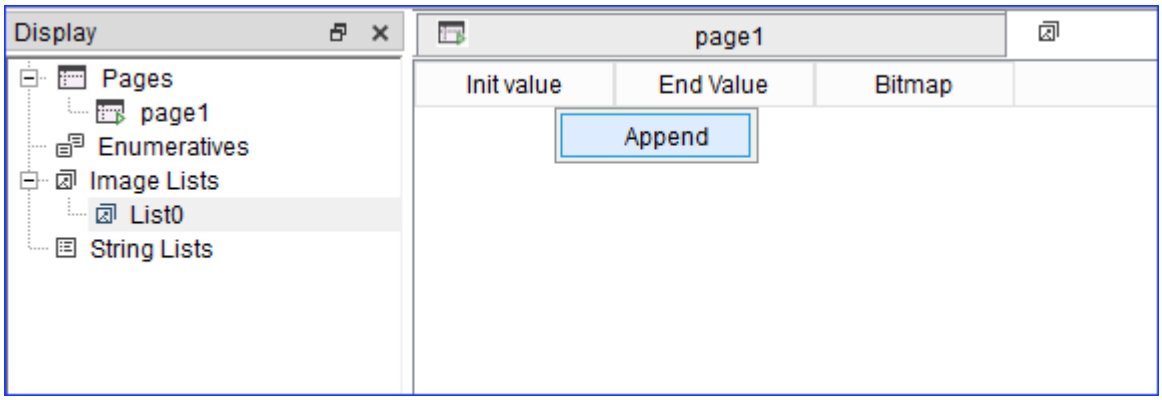
Animation elements allow you to associate each change in a variable value with the display of different images. It displays the bitmap images you select from the image list, depending on the value of the associated selection variable.

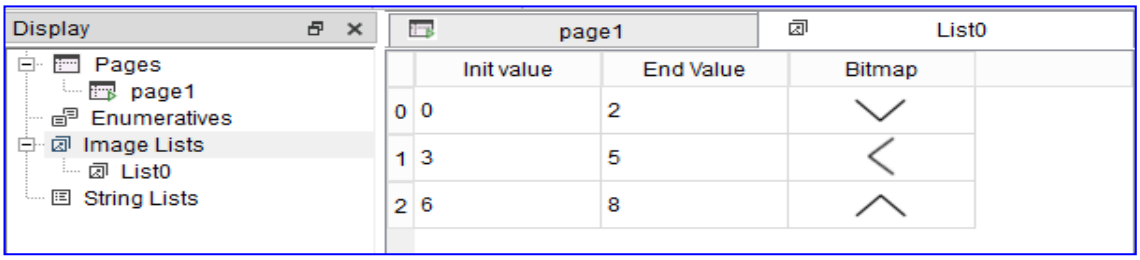
Create image list

To create an animation, you must first create an image list. The image list contains images that can be displayed by variables. Each image is associated with a value or range of values that can be taken by the variable. As shown in the figure below, right-click on 'Image Lists' in the project tree and click 'Add new'.



Add images to the list and set the value range for each image. When the variable value is within the corresponding image value range, the corresponding image will appear in the animated image. For example, when the associated variable value is 0, the image in the first column will appear in the animated image.

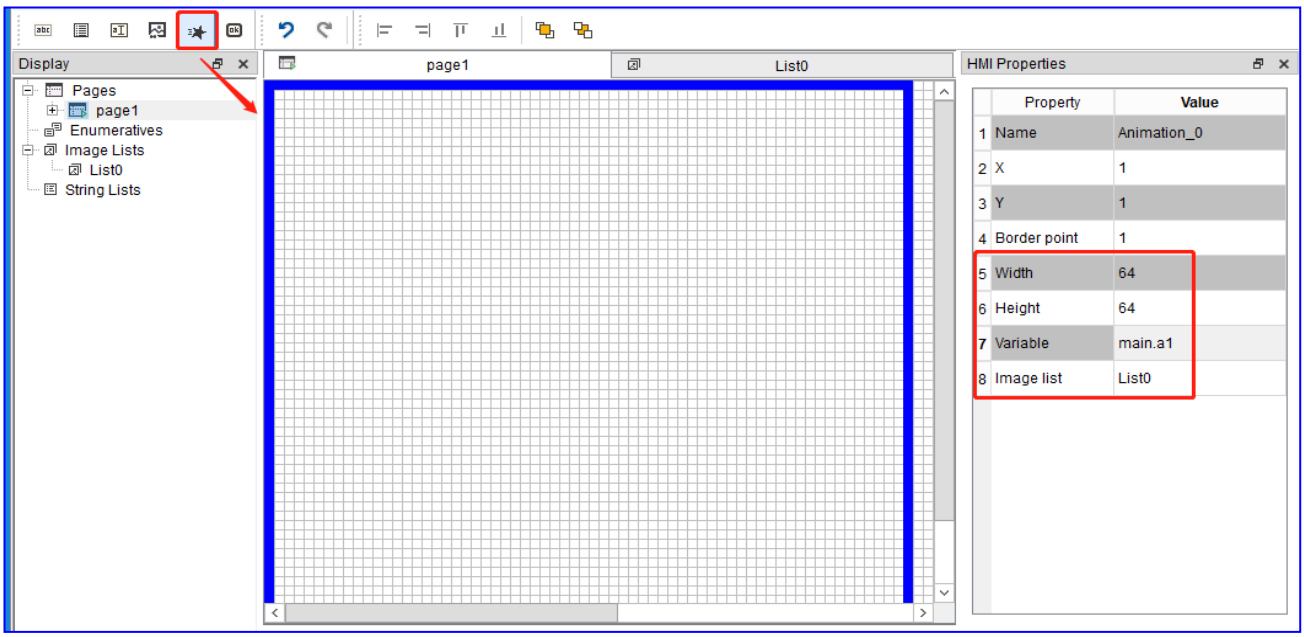




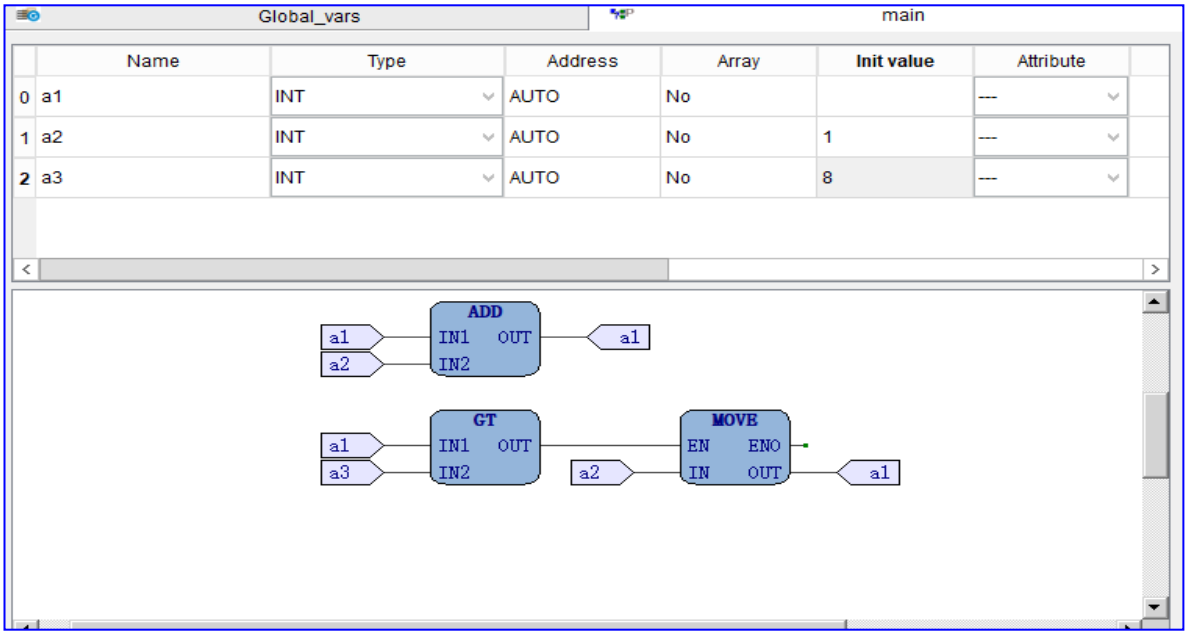
Insert animation

Click the animation icon on the toolbar. Then move the mouse to the active area of the page. A '+' symbol indicates the insertion point for the object. Click the left mouse button to insert the object into the grid. Set the height and width to a larger size to avoid the image being cut off.

In the HMI properties, associate the main program variable a1 with the Variable field. The value of a1 will display the corresponding image within the range of the image list. Select List0 in the Image list field.



The main program is as follows:



After enabling simulation, enter Custompages, then press OK to see the image conversion.

4.8.10 **Buttons**

Buttons are used to interact with the system. Button elements are divided into four types:

- InvertBooL: Used to invert the DIDO state. Includes native DIDO and extended DIDO.
- OpenPage: Used to open other pages.
- Close: Closes the current page.
- None: No action.

Inserting a button

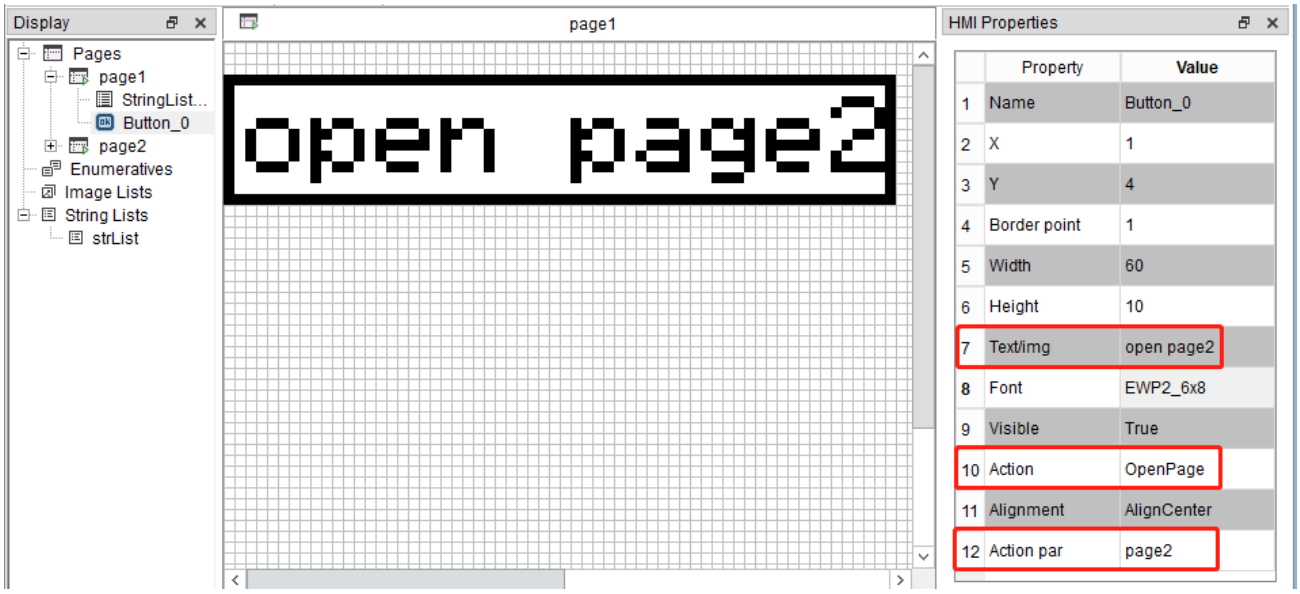
To insert a new button, click the ‘Insert New Button’ icon in the toolbar.

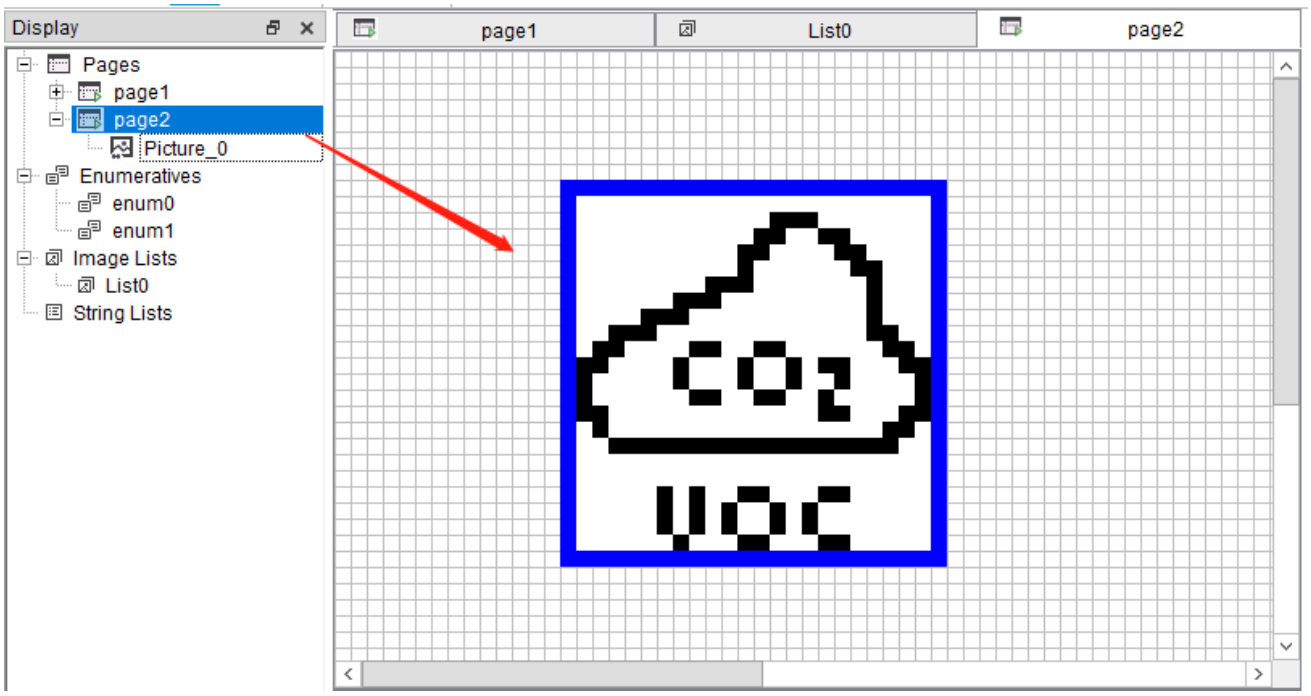
Then, move the mouse to the active area of the page. A ‘+’ symbol indicates the insertion point for the object. Click the left mouse button to insert the object into the grid.

Creating an Open Button

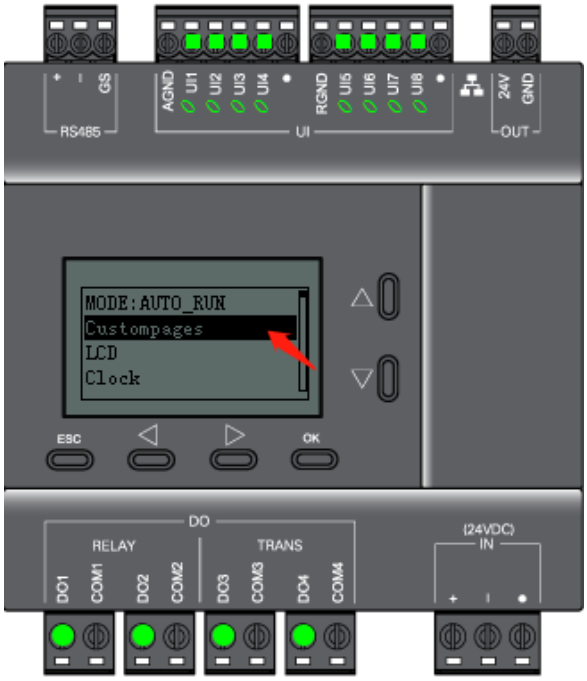
After inserting a button, you can open other pages by pressing that button.

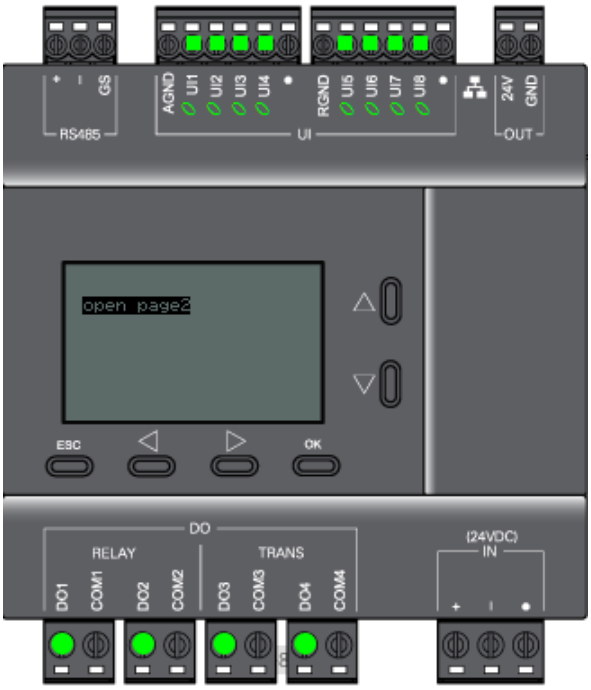
In the HMI Properties window, select ‘Test/img’ as the button name, select the “Action” field, and click ‘OpenPage.’ Then, in the ‘Action par’ field, select the page to open when the open button is pressed. In the following example, pressing the ‘OpenPage’ button on Page1 will open the Page2 page.



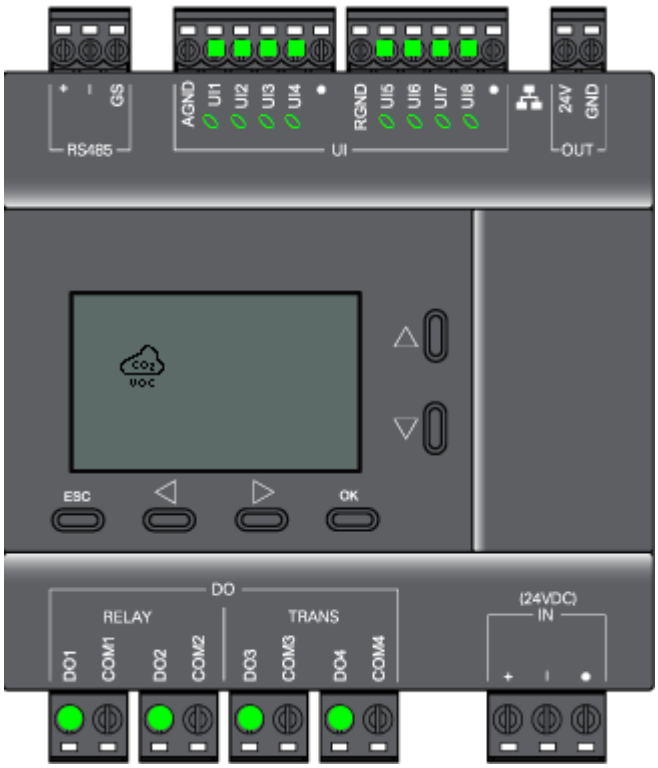


After enabling simulation, enter Custompages, then press OK to switch from page1 to page2:





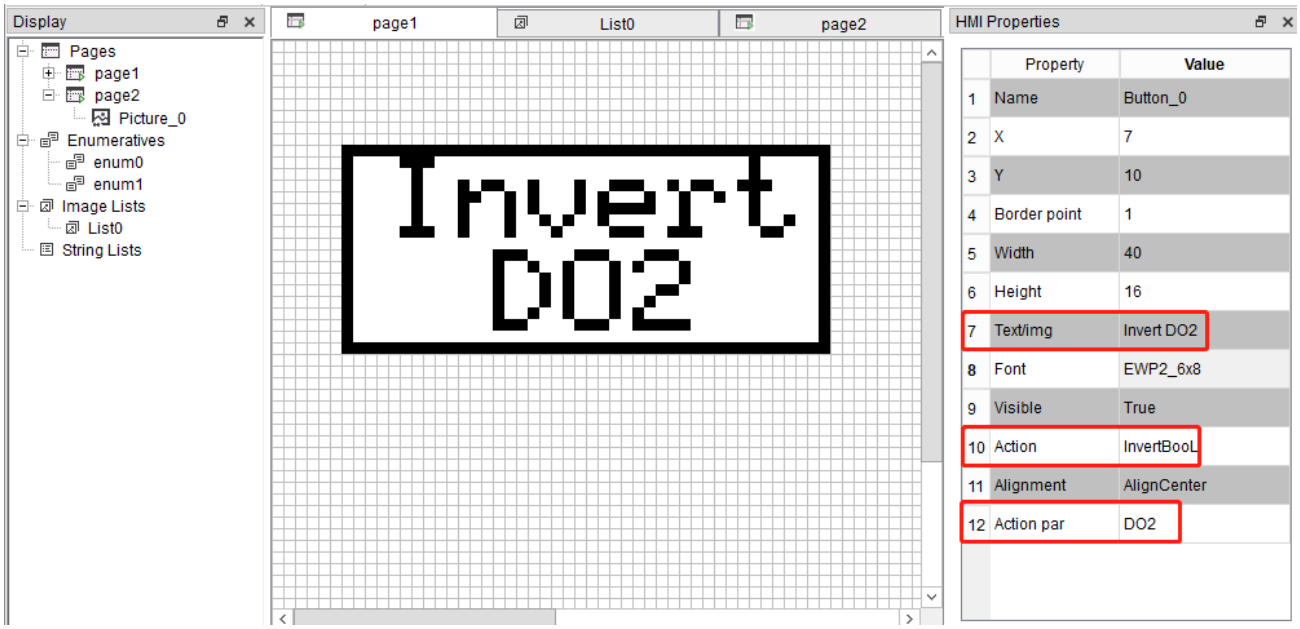
Page 2 is as follows:



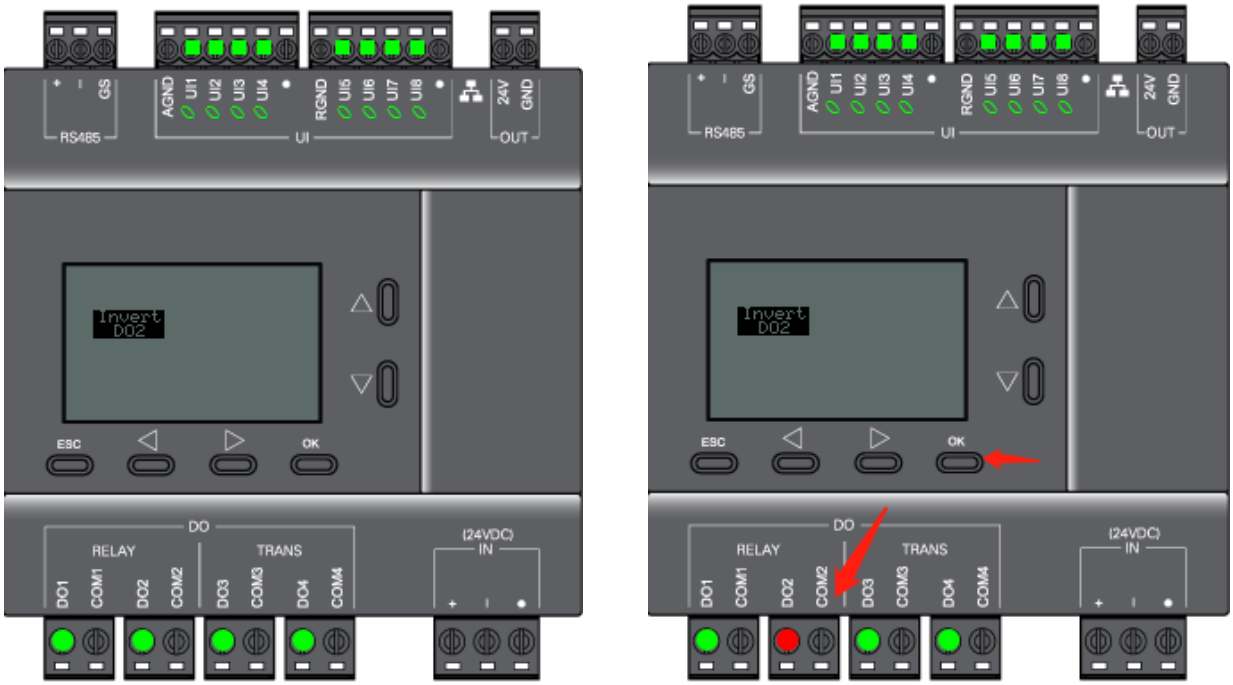
Create the InvertBooL button.

After creating the button, enter the name to be displayed on the button in the ‘test/img’ field in the HMI Properties window, such as Invert DO2.

Then select InvertBooL in the Action field and associate the inverted Boolean variable, such as DO2, in Action Par.



After enabling simulation, enter Custompages, then press OK (i.e., modify the Boolean variable value) to invert the state of DO2. As shown in the figure below, DO2 is OFF (green) by default, and after pressing OK, it is inverted to ON (red).



Create a Close button.

After inserting the button, you can close the current page by pressing it.

In the HMI Properties window, select ‘Test/img’ as the button name, select the “Action” field, and click ‘Close’. After starting the simulation, enter Custompages, then press OK (i.e., close the current page) to close the current page.

Fifth section

Instruction description

5.1 Basic type conversion

5.2 Basic type conversion BCD conversion

5.3 TIM conversion

5.4 Numerical

5.5 Airthmetic

5.6 Time

5.7 Bit Shift

5.8 Bit wise

5.9 Selection

5.10 Comparision

5.11 Character string

5.12 Standard function blocks

5.13 Smart function blocks

5.14 Bit conversion functions

5.15 Special functions

5.1 Basic type conversion

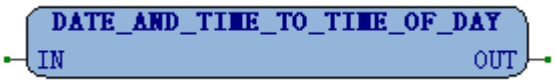
According to the IEC 61131-3 standard, type conversion functions shall have the form *_TO_**, where “*” is the type of the input variable, and “**” the type of the output variable.

5.2 Basic type conversion BCD conversion

No	Conversion Function	Conversion Details
1	BCD_TO_USINT	Convert BCD code to USINT value.
2	BCD_TO_UINT	Convert BCD code to UINT value.
3	BCD_TO_UDINT	Convert BCD code to UDINT value.
4	BCD_TO_ULINT	Convert BCD code to ULINT value.
5	USINT_TO_BCD	Convert USINT values to BCD code format
6	UINT_TO_BCD	Convert UINT values to BCD code format
7	UDINT_TO_BCD	Convert UDINT values to BCD code format
8	ULINT_TO_BCD	Convert ULINT values to BCD code format

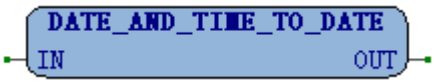
5.3 TIM conversion

5.3.1 DATE AND TIME TO TIME OF DAY



Name	Input / Output	Data Type	Describe
IN	Input	DT	
OUT	Output	TOD	

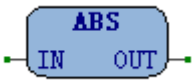
5.3.2 DATE AND TIME TO DATE



Name	Input / Output	Data Type	Describe
IN	Input	DT	
OUT	Output	DATE	

5.4 Numerical

5.4.1 ABS



Name	Input / Output	Data Type	Describe
IN	Input	ANY_NUM	absolute value For example: ABS(X)=Y
OUT	Output	ANY_NUM	

5.4.2 SQRT

Name	Input / Output	Data Type	Describe
IN	Input	ANY_REAL	Square root For example: SQRT(X)=Y
OUT	Output	ANY_REAL	

5.4.3 LN

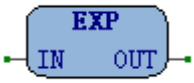
Name	Input / Output	Data Type	Describe
IN	Input	ANY_REAL	Natural logarithm For example: LN(X)=Y
OUT	Output	ANY_REAL	

5.4.4 LOG

Name	Input / Output	Data Type	Describe
IN	Input	ANY_REAL	Logarithm base 10

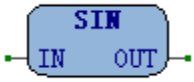
OUT	Output	ANY_REAL	For example: LOG(X)=Y

5.4.5 EXP



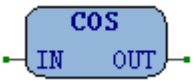
Name	Input / Output	Data Type	Describe
IN	Input	ANY_REAL	Natural exponential For example: EXP(X)=Y
OUT	Output	ANY_REAL	

5.4.6 SIN



Name	Input / Output	Data Type	Describe
IN	Input	ANY_REAL	Sine of input in radians For example: SIN(X)=Y
OUT	Output	ANY_REAL	

5.4.7 COS



Name	Input / Output	Data Type	Describe
IN	Input	ANY_REAL	Cosine in radians For example: COS(X)=Y
OUT	Output	ANY_REAL	

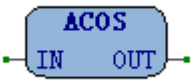
5.4.8 TAN

Name	Input / Output	Data Type	Describe
IN	Input	ANY_REAL	Tangent in radians For example: TAN(X)=Y
OUT	Output	ANY_REAL	

5.4.9 **ASIN**

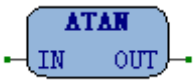
Name	Input / Output	Data Type	Describe
IN	Input	ANY_REAL	Principal arc sine For example: ASIN(X)=Y
OUT	Output	ANY_REAL	

5.4.10 **ACOS**



Name	Input / Output	Data Type	Describe
IN	Input	ANY_REAL	Principal arc cosine For example: ACOS(X)=Y
OUT	Output	ANY_REAL	

5.4.11 **ATAN**



Name	Input / Output	Data Type	Describe
IN	Input	ANY_REAL	Principal arc tangent For example: ATAN(X)=Y
OUT	Output	ANY_REAL	

5.5 Airthmetic

The ADD \ MUL \ SUB \ DIV instructions can add multiple input pins by right clicking on the function block in the program and selecting Increase pins.

5.5.1 ADD

This function block is used for variable or constant addition, the input and output variable types must be the same.

Name	Input / Output	Data Type	Describe
IN1	Input	ANY_NUM	OUT= IN1 + IN2
IN2	Input	ANY_NUM	
OUT	Output	ANY_NUM	

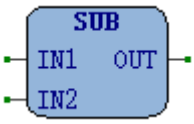
5.5.2 MUL

This function block is used for variable or constant multiplication. The input and output data types must be the same.

Name	Input / Output	Data Type	Describe
IN1	Input	ANY_NUM	OUT= IN1 * IN2
IN2	Input	ANY_NUM	
OUT	Output	ANY_NUM	

5.5.3 SUB

This function block is used for variable or constant subtraction. The input and output data types must be the same.



Name	Input / Output	Data Type	Describe
IN1	Input	ANY_NUM	OUT= IN1 - IN2
IN2	Input	ANY_NUM	
OUT	Output	ANY_NUM	

5.5.4 DIV

Function blocks are used for dividing variables or constants, the input and output variable types must be the same.

Name	Input / Output	Data Type	Describe
IN1	Input	ANY_NUM	OUT= IN1 / IN2
IN2	Input	ANY_NUM	
OUT	Output	ANY_NUM	

5.5.5 MOD

This function block is used to divide the value of IN1 by the value of IN2 to obtain the remainder, and the data types of the input and output values must be the same

Name	Input / Output	Data Type	Describe
IN1	Input	ANY_INT	OUT= IN1 Modulo IN2
IN2	Input	ANY_INT	
OUT	Output	ANY_INT	

5.5.6 EXPT

Name	Input / Output	Data Type	Describe
IN1	Input	ANY_REAL	OUT = IN1 ^{IN2}
IN2	Input	ANY_REAL	
OUT	Output	ANY_REAL	

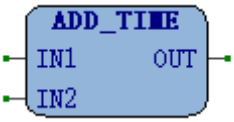
5.5.7 MOVE

This function block is used to assign the value of a constant or variable to another variable.

Name	Input / Output	Data Type	Describe
IN	Input	ANY	OUT = IN
OUT	Output	ANY	

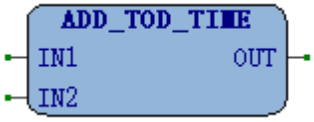
5.6 Time

5.6.1 ADD_TIME



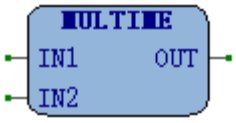
Name	Input / Output	Data Type	Describe
IN1	Input	TIME	
IN2	Input	TIME	
OUT	Output	TIME	

5.6.2 ADD_TOD_TIME



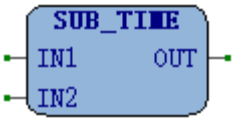
Name	Input / Output	Data Type	Describe
IN1	Input	TOD	
IN2	Input	TIME	
OUT	Output	TOD	

5.6.3 MULTIME



Name	Input / Output	Data Type	Describe
IN1	Input	TIME	
IN2	Input	ANY_NUM	
OUT	Output	TIME	

5.6.4 SUB_TIME

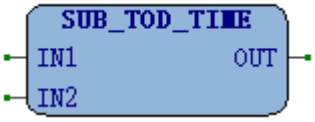


Name	Input / Output	Data Type	Describe
IN1	Input	TIME	
IN2	Input	TIME	
OUT	Output	TIME	

5.6.5 SUB_DATE_DATE

Name	Input / Output	Data Type	Describe
IN1	Input	DATE	
IN2	Input	DATE	
OUT	Output	TIME	

5.6.6 SUB_TOD_TIME



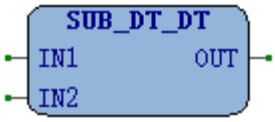
Name	Input / Output	Data Type	Describe
IN1	Input	TOD	
IN2	Input	TIME	
OUT	Output	TOD	

5.6.7 SUB_DT_TIME

Name	Input / Output	Data Type	Describe
IN1	Input	DT	
IN2	Input	TIME	

OUT	Output	DT	

5.6.8 SUB_DT_DT



Name	Input / Output	Data Type	Describe
IN1	Input	DT	
IN2	Input	DT	
OUT	Output	TIME	

5.6.9 DIVTIME

Name	Input / Output	Data Type	Describe
IN1	Input	TIME	
IN2	Input	ANY_NUM	
OUT	Output	TIME	

5.7 Bit Shift

Example:

IN = 2#0001 1001 of type BYTE, N=3

SHL(IN,3)=2#1100 1000

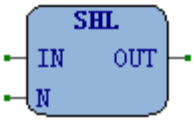
SHR(IN,3)=2#0000 0011

ROL(IN,3)=2#1100 1000

ROR(IN,3)=2#0010 0011

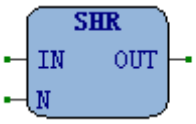
NOTE: IN of type BooL (one bit) does not make sense. It is an error if the value of the N input is less than zero.

5.7.1 SHL



Name	Input / Output	Data Type	Describe
IN	Input	ANY_BIT	
N	Input	ANY_INT	Number of bits to be shifted
OUT	Output	ANY_BIT	OUT = IN left-shifted by N bits, zero-filled on right

5.7.2 SHR



Name	Input / Output	Data Type	Describe
IN	Input	ANY_BIT	
N	Input	ANY_INT	Number of bits to be shifted
OUT	Output	ANY_BIT	OUT = IN right - shifted by N bits, zero-filled on left

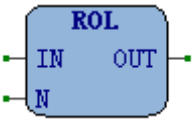
5.7.3 ROR

This function block is used to cyclically shift the operand to the right by bit, and the bit moved out on the right is directly added to the highest bit on the left.

Name	Input / Output	Data Type	Describe
IN	Input	ANY_NBIT	
N	Input	ANY_INT	Number of bits to be shifted
OUT	Output	ANY_NBIT	OUT = IN right - rotated by N bits, circular

5.7.4 ROL

This function block is used to cyclically shift IN left by bit, and the bits moved out on the left are directly added to the lowest bit on the right.



Name	Input / Output	Data Type	Describe
IN	Input	ANY_NBIT	
N	Input	ANY_INT	Number of bits to be shifted
OUT	Output	ANY_NBIT	OUT = IN left-rotated by N bits, circular

5.8 Bit wise

The AND/OR/XOR instruction can add multiple input pins. To add them, right-click on the function block in the program and select Increase pins.

5.8.1 AND

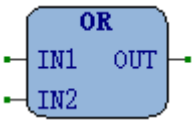
This function block is used for the operation of variables or constants, and the input and output variable types must be consistent. If all inputs are 1, then the output is 1.

Name	Input / Output	Data Type	Describe
IN1	Input	ANY_BIT	
IN2	Input	ANY_BIT	
OUT	Output	ANY_BIT	

IN1	IN2	OUT
0	0	0
0	1	0
1	0	0
1	1	1

5.8.2 OR

Function blocks are used for bitwise OR operations on variables or constants, and the input and output data types must be consistent. When at least one input is 1, the output is also 1. Otherwise, the output is 0.



Name	Input / Output	Data Type	Describe
IN1	Input	ANY_BIT	
IN2	Input	ANY_BIT	
OUT	Output	ANY_BIT	

IN1	IN2	OUT
0	0	0
0	1	1
1	0	1
1	1	1

5.8.3 XOR

This function block is used for XOR operation of variables or constants, and the input and output data types must be consistent.

When only one input is 1, the output bit also generates 1. When all inputs are 1 or 0, the output is 0.

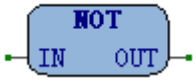
Name	Input / Output	Data Type	Describe
IN1	Input	ANY_BIT	
IN2	Input	ANY_BIT	
OUT	Output	ANY_BIT	

IN1	IN2	OUT
0	0	0
0	1	1
1	0	1
1	1	0

5.8.4 **NOT**

This function block is used for NOT binary operations on variables or constants, where NOT binary operations are performed bit by bit, and the input and output data types must be consistent.

When the input is 0, the output is 1. When the input is 1, the output is 0.



Name	Input / Output	Data Type	Describe
IN	Input	ANY_BIT	
OUT	Output	ANY_BIT	

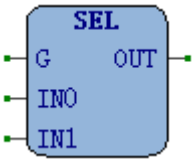
IN	OUT
0	1
1	0

5.9 Selection

The MAX, MIN, and MUX instructions can add multiple input pins by right clicking on the function block in the program and selecting Increase pins.

5.9.1 **SEL**

This function block is used to select one input variable from two input variables as the output.



Name	Input / Output	Data Type	Describe
G	Input	BOOL	Selection out of 2 inputs (gate)
IN0	Input	ANY	

IN1	Input	ANY	
OUT	Output	ANY	OUT = IN0 if G = FALSE OUT = IN1 if G = TRUE

5.9.2 MAX

This function block is used to select the maximum number from the values of two or more input variables as the output.

Name	Input / Output	Data Type	Describe
IN1	Input	ANY	
IN2	Input	ANY	
OUT	Output	ANY	OUT = MAX(IN1, IN2)

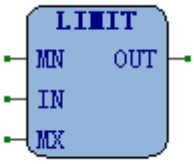
5.9.3 MIN

This function block is used to select the minimum number from the values of two or more input variables as the output.

Name	Input / Output	Data Type	Describe
IN1	Input	ANY	
IN2	Input	ANY	
OUT	Output	ANY	OUT = MIN(IN1, IN2)

5.9.4 LIMIT

This function block is used to limit the output of input variable values, that is, regardless of how the input variable IN changes, only the numerical values between MN and MX are output.

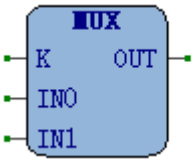


Name	Input / Output	Data Type	Describe
MN	Input	ANY	Minimum value for limitation
IN	Input	ANY	

MX	Input	ANY	Maximum value for limitation
OUT	Output	ANY	OUT = MIN if IN ≤ MN OUT = MX if IN≥ MX

5.9.5 **MUX**

This function block is used to select a variable output from multiple input pins.



Name	Input / Output	Data Type	Describe
K	Input	ANY_INT	Selection out of n inputs
IN0	Input	ANY	
IN1	Input	ANY	
OUT	Output	ANY	OUT = MUX (K, IN0, IN1..,INn) If K=0, OUT =IN0, If K=n, OUT =INn

5.10 Comparision

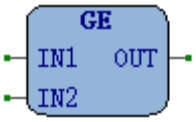
The GT \ GE \ EQ \ LT \ LE instructions can add multiple input pins. To add them, right-click on the function block in the program and select Increase pins.

5.10.1 **GT**

This is a Boolean operator. When the first operand is larger than the second, the return value is TRUE.

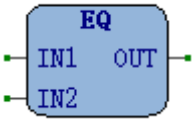
Name	Input / Output	Data Type	Describe
IN1	Input	ANY	
IN2	Input	ANY	
OUT	Output	BOOL	If the IN1 > IN2>…> INn, OUT = TRUE, Otherwise, OUT =False.

5.10.2 GE



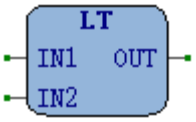
Name	Input / Output	Data Type	Describe
IN1	Input	ANY	
IN2	Input	ANY	
OUT	Output	BOOL	If the IN1≥ IN2≥…≥ INn, OUT = TRUE, Otherwise, OUT =False.

5.10.3 EQ



Name	Input / Output	Data Type	Describe
IN1	Input	ANY	
IN2	Input	ANY	
OUT	Output	BOOL	If the IN1 = IN2=…= INn, OUT = TRUE, Otherwise, OUT =False.

5.10.4 LT



Name	Input / Output	Data Type	Describe
IN1	Input	ANY	
IN2	Input	ANY	
OUT	Output	BOOL	If the IN1 < IN2 <…< INn, OUT = TRUE, Otherwise, OUT =False.

5.10.5 LE

Name	Input / Output	Data Type	Describe
IN1	Input	ANY	

IN2	Input	ANY	
OUT	Output	BOOL	If the IN1 ≤ IN2 ≤…≤ INn, OUT = TRUE, Otherwise, OUT =False.

5.10.6 NE

Name	Input / Output	Data Type	Describe
IN1	Input	ANY	
IN2	Input	ANY	
OUT	Output	BOOL	If the IN1 ≠ IN2 ≠…≠ INn, OUT = TRUE, Otherwise, OUT =False.

5.11 Character string

5.11.1 CONCAT

Merge two strings to form a new string output.

Name	Input / Output	Data Type	Describe
IN1	Input	STRING	Merge the header of a string. e.g., it
IN2	Input	STRING	Merge the tail of a string. e.g., em
OUT	Output	STRING	Output the merged string. e.g., item

5.11.2 CONCAT DATE TOD

Name	Input / Output	Data Type	Describe
IN1	Input	DATE	
IN2	Input	TOD	
OUT	Output	DT	

5.11.3 DELETE

Remove specific contiguous parts from a string, and the remaining characters form the string output.

Name	Input / Output	Data Type	Describe
IN	Input	STRING	Initial String
L	Input	ANY_INT	Delete the length of characters
P	Input	ANY_INT	The position to be deleted starting from the left count (including this position)
OUT	Output	STRING	Output the merged string

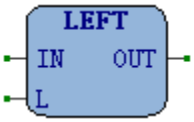
5.11.4 FIND

Find the initial position of a specific string from a string.

Name	Input / Output	Data Type	Describe
IN1	Input	STRING	Initial String
IN2	Input	STRING	The searched string
OUT	Output	INT	The starting position of IN2 in IN1,If IN2 is not found in IN1, output '0'

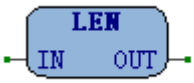
5.11.5 LEFT

Extract a specific number of characters from the leftmost part of the string to the right Form a new string output.



Name	Input / Output	Data Type	Describe
IN1	Input	STRING	Input string. e.g., character
IN2	Input	ANY_INT	Left position within character string. e.g., 5
OUT	Output	STRING	Output string. e.g., chara

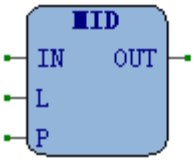
5.11.6 **LEN**



Name	Input / Output	Data Type	Describe
IN	Input	STRING	Input string. e.g., character
OUT	Output	INT	Output string length. e.g., 9

5.11.7 **MID**

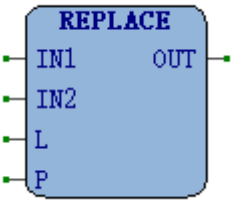
Extract a specific number of characters from the middle of a string to form a new string output.



Name	Input / Output	Data Type	Describe
IN	Input	STRING	Input string. e.g., character
L	Input	ANY_INT	Number of extracted digits. e.g., 5
P	Input	ANY_INT	Start extracting digits, count from left. e.g., 3
OUT	Output	STRING	Output string. e.g., racte

5.11.8 **REPLACE**

Replace a portion of a longer string with a string to form a new string output.



Name	Input / Output	Data Type	Describe
IN1	Input	STRING	Initial String. e.g., character
IN2	Input	STRING	String used for replacement. e.g., AA
L	Input	ANY_INT	The length of the replaced characters. e.g., 3
P	Input	ANY_INT	The position to be replaced starting from the left count (including this position). e.g., 3
OUT	Output	STRING	Output the merged string. e.g., chaAAter

5.11.9 **RIGHT**

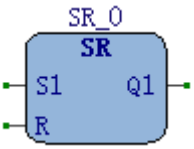
Extract a specific number of characters from the far right of the string to the left to form a new string output.

Name	Input / Output	Data Type	Describe
IN	Input	STRING	Input string. e.g., character
L	Input	ANY_INT	Number of extracted digits. e.g., 5
OUT	Output	STRING	Output string. e.g., acter

5.12 Standard function blocks

5.12.1 **SR**

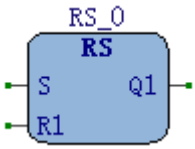
Describe: The SR bistable is a latch where the Set dominates.



Name	Input / Output	Data Type	Describe
S1	Input	BOOL	This input terminal is a set input. When it is true, the set input is set, and when it is false, the set input disappears.
R	Input	BOOL	This input terminal is a reset input. When it is true, the reset input is reset, and when it is false, the reset input disappears
Q1	Output	BOOL	When SET1 is true, regardless of whether RESET is true or not, Q1 output is true. When SET1 is set to false, if Q1 outputs true, once RESET is set to true, Q1 outputs immediately reset to false. If Q1 output is false, regardless of whether RESET is true or false, Q1 output remains false.

5.12.2 RS

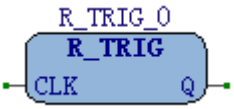
Describe: The RS bistable is a latch where the Reset dominates



Name	Input / Output	Data Type	Describe
S	Input	BOOL	This input terminal is a set input. When it is true, the set input is set, and when it is false, the set input disappears.
R1	Input	BOOL	This input terminal is a reset input. When it is true, the reset input is reset, and when it is false, the reset input disappears.
Q1	Output	BOOL	When RESET1 is TRUE, regardless of whether SET is TRUE or not, Q1 output is always False. When RESET1 is set to false, if Q1 outputs false, once SET is set to true, Q1 outputs immediately set to true. If Q1 output is TRUE, regardless of whether SET is TRUE or False, Q1 output remains TRUE.

5.12.3 R_TRIG

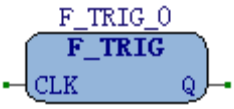
Describe: The output produces a single pulse when a rising edge is detected



Name	Input / Output	Data Type	Describe
CLK	Input	BOOL	The input terminal is a Boolean signal input with a detected rising edge
Q	Output	BOOL	Trigger state output, once a rising edge signal input is detected in CLK, Q outputs TRUE. When CLK changes from False to True, the Q output first changes to True and then the Q output changes to False. If CLK remains consistently false or true, Q output remains consistently false.

5.12.4 F_TRIG

Describe: The output produces a single pulse when a falling edge is detected

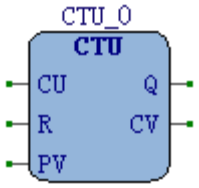


Name	Input / Output	Data Type	Describe
CLK	Input	BOOL	The input terminal is a Boolean type signal input with a detected falling edge.
Q	Output	BOOL	Trigger state output, once a falling edge signal input is detected in CLK, Q outputs TRUE

			When CLK changes from True to False, the Q output first changes to True and then to False. If CLK remains consistently false or true, Q output remains consistently false.

5.12.5 CTU

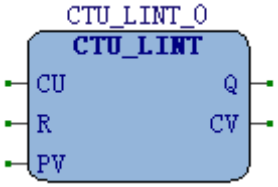
Describe: The up-counter can be used to signal when a count has reached a maximum value



Name	Input / Output	Data Type	Describe
CU	Input	BOOL	Detecting rising edge signal input triggers output CV increment
R	Input	BOOL	Reset the counter, and if it is true, reset CV to 0
PV	Input	INT	Used to set the high limit of output CV increment count
Q	Output	BOOL	Counter status output: When the output CV increases to the upper limit PV, Q outputs TRUE.
CV	Output	INT	The output is an incremental count real-time value, which displays the increasing values from 0 to PV in sequence according to the rising edge of CU. When RESET is set to TRUE, regardless of whether CU detects the rising edge, the incremental count cannot be triggered and CV remains at 0. When RESET is set to False, when there is a rising edge at the CU end that changes from False to True, CV will increase by 1. When CV increases to the upper limit PV, Q output will be set to True.

5.12.6 CTU_LINT

Describe: The up-counter can be used to signal when a count has reached a maximum value.

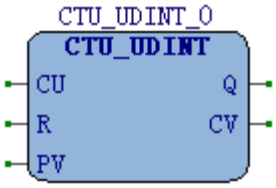


Name	Input / Output	Data Type	Describe
CU	Input	BOOL	
R	Input	BOOL	
PV	Input	LINT	

Q	Output	BOOL	
OUT	Output	LINT	

5.12.7 CTU_UDINT

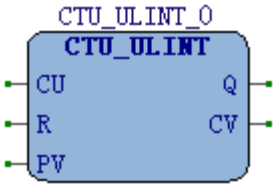
Describe: The up-counter can be used to signal when a count has reached a maximum value.



Name	Input / Output	Data Type	Describe
CU	Input	BOOL	
R	Input	BOOL	
PV	Input	UDINT	
Q	Output	BOOL	
CV	Output	UDINT	

5.12.8 CTU_ULINT

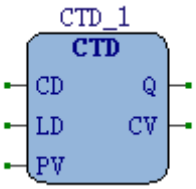
Describe: The up-counter can be used to signal when a count has reached a maximum value



Name	Input / Output	Data Type	Describe
CU	Input	BOOL	
R	Input	BOOL	
PV	Input	ULINT	
Q	Output	BOOL	
CV	Output	ULINT	

5.12.9 CTD

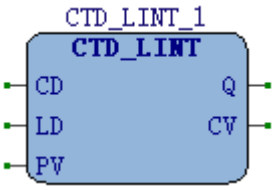
Describe: The down-counter can be used to signal when a count has reached zero, on counting down from a preset value.



Name	Input / Output	Data Type	Describe
CD	Input	BOOL	Detecting the signal input of the rising edge triggers the output CV to decrease
LD	Input	BOOL	Used to reset the counter, if it is true, CV is reset to the upper limit value PV.
PV	Input	INT	Used to set the high limit of output CV countdown.
Q	Output	BOOL	Counter status output: When the output CV decreases to 0, Q outputs TRUE
CV	Output	INT	Display the increasing values from PV to 0 according to the rising edge of CD. When LOAD is TRUE, regardless of whether CD detects the rising edge, it cannot trigger the countdown, and CV maintains the upper limit of the countdown PV value. When LOAD is set to False and there is a rising edge on the CD side that changes from False to True, if CV is greater than 0, CV will decrease by 1. When CV is equal to 0, Q output will be set to True

5.12.10 **CTD_LINT**

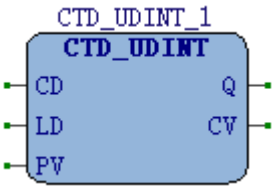
Describe: The down-counter can be used to signal when a count has reached zero, on counting down from a preset value.



Name	Input / Output	Data Type	Describe
CD	Input	BOOL	
LD	Input	BOOL	
PV	Input	LINT	
Q	Output	BOOL	
CV	Output	LINT	

5.12.11 **CTD_UDINT**

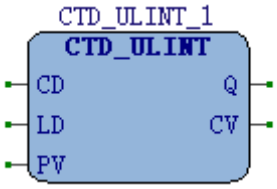
Describe: The down-counter can be used to signal when a count has reached zero, on counting down from a preset value.



Name	Input / Output	Data Type	Describe
CD	Input	BOOL	
LD	Input	BOOL	
PV	Input	UDINT	
Q	Output	BOOL	
CV	Output	UDINT	

5.12.12 CTD_ULINT

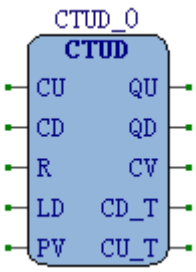
Describe: The down-counter can be used to signal when a count has reached zero, on counting down from a preset value.



Name	Input / Output	Data Type	Describe
CD	Input	BOOL	
LD	Input	BOOL	
PV	Input	ULINT	
Q	Output	BOOL	
CV	Output	ULINT	

5.12.13 CTUD

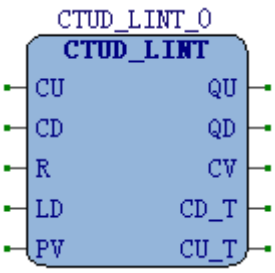
Describe: The up-down counter has two inputs CU and CD. It can be used to both count up on one input and down on the other.



Name	Input / Output	Data Type	Describe
CU	Input	BOOL	Detecting rising edge signal input triggers output CV increment
CD	Input	BOOL	Detecting the signal input of the rising edge triggers the output CV to decrease
R	Input	BOOL	Reset the increment counter, and if it is true, reset CV to 0
LD	Input	BOOL	Reset the countdown counter, and if it is true, reset CV to the upper limit value PV
PV	Input	INT	Set the upper limit of counting for decreasing or decreasing output CV
QU	Output	BOOL	Counter status output: When the output CV increases to the upper limit PV, QU outputs TRUE
QD	Output	BOOL	Counter status output: When the output CV decreases to 0, QD outputs TRUE
CV	Output	INT	The output is an incremental or decremental real-time count value. When RESET is set to TRUE, regardless of whether CU or CD detects a rising edge, the counter cannot be triggered and CV remains at 0 When LOAD is TRUE, regardless of whether CU or CD detects a rising edge, the counter cannot be triggered, and CV maintains the upper limit PV value for decreasing counting
CD_T	Output	R_TRIG	
CU_T	Output	R_TRIG	

5.12.14 CTUD_LINT

Describe: The up-down counter has two inputs CU and CD. It can be used to both count up on one input and down on the other.

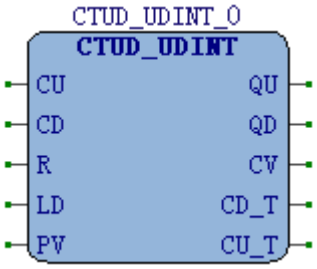


Name	Input / Output	Data Type	Describe
CU	Input	BOOL	
CD	Input	BOOL	
R	Input	BOOL	
LD	Input	BOOL	
PV	Input	LINT	
QU	Output	BOOL	

QD	Output	BOOL	
CV	Output	LINT	
CD_T	Output	R_TRIG	
CU_T	Output	R_TRIG	

5.12.15 CTUD_UDINT

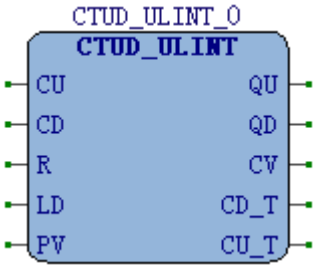
Describe: The up-down counter has two inputs CU and CD. It can be used to both count up on one input and down on the other.



Name	Input / Output	Data Type	Describe
CU	Input	BOOL	
CD	Input	BOOL	
R	Input	BOOL	
LD	Input	BOOL	
PV	Input	UDINT	
QU	Output	BOOL	
QD	Output	BOOL	
CV	Output	UDINT	
CD_T	Output	R_TRIG	
CU_T	Output	R_TRIG	

5.12.16 CTUD_ULINT

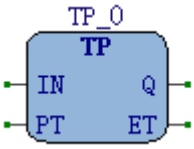
Describe: The up-down counter has two inputs CU and CD. it can be used to both count up on one input and down on the other.



Name	Input / Output	Data Type	Describe
CU	Input	BOOL	
CD	Input	BOOL	
R	Input	BOOL	
LD	Input	BOOL	
PV	Input	ULINT	
QU	Output	BOOL	
QD	Output	BOOL	
CV	Output	ULINT	
CD_T	Output	R_TRIG	
CU_T	Output	R_TRIG	

5.12.17 **TP**

Describe: The pulse timer can be used to generate output pulses of a given time duration



Name	Input / Output	Data Type	Describe
IN	Input	BOOL	Detecting rising edge signal input triggers timer
PT	Input	TIME	This input is a time constant used to set the upper limit of the ET timing output
Q	Output	BOOL	Timer status output: When the output ET reaches the upper limit PT of the count, Q outputs TRUE
ET	Output	TIME	This output is the real-time timing value, which starts counting from the rising edge of IN. When the rising edge of IN is detected, regardless of whether IN remains true or not, output ET to start timing with millisecond accuracy. Before ET timing reaches the set upper limit of PT timing, output Q as true. As long as ET timing reaches the set upper limit of PT timing, output Q as false. If IN does not detect a rising edge, Q output is false.

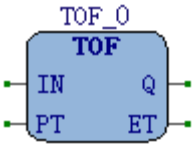
5.12.18 TON

Describe: The on-delay timer can be used to delay setting an output true, for fixed period after an input becomes true.

Name	Input / Output	Data Type	Describe
IN	Input	BOOL	Detecting the signal input of the rising edge triggers the power on delay timer
PT	Input	TIME	Set the power on delay time
Q	Output	BOOL	Timer status output, when outputting ET to the delay time PT, Q outputs TRUE
ET	Output	TIME	This output is the real-time delay value, which is the time value counted from the rising edge of IN. When the rising edge of IN is detected, the output ET starts timing. Only when the input IN continues to be true and reaches the set time PT, the timer status output Q is true. If the input IN changes from True to False before the timer reaches the set time PT, the timer status output Q will still be False.

5.12.19 TOF

Describe: The off-delay timer can be used to delay setting an output false, for fixed period after input goes false.

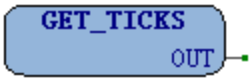


Name	Input / Output	Data Type	Describe
IN	Input	BOOL	Detecting the signal input of the falling edge triggers the power-off delay timer.
PT	Input	TIME	Set power outage delay time
Q	Output	BOOL	Timer status output: When the output ET reaches the delay time PT, Q outputs False.
ET	Output	TIME	This output is the real-time delay value, which is the time value counted from the falling edge of IN. When IN is TRUE, Q is TRUE, and ET is 0. When the falling edge of IN is detected, the output ET starts timing. Only when the input IN continues to be false and reaches the set time PT, the timer status output Q is true. If the input IN changes from False to True before the timer reaches the set time PT, the timer status output Q will still be True.

5.13 Smart function blocks

5.13.1 GET_TICKS

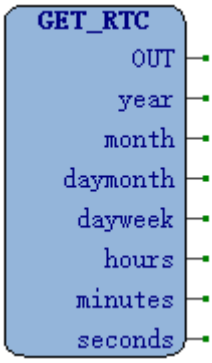
Describe:Returns the number of milliseconds passed since the plc was powered on. This number will overlow (go back to zero), after approximately 50 days.



Name	Return	Data Type	Describe
OUT	Return	UDINT	

5.13.2 GET_RTC

Describe: Get system clock. The function successfully returns 0

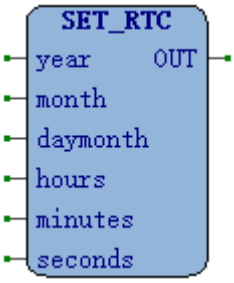


Name	Input / Output	Data Type	Describe
year	Output	UINT	year [2000...2099]
month	Output	UINT	month [1...12]
daymonth	Output	UINT	day of month [1...31]
dayweek	Output	UINT	day of week [1...7]
hours	Output	UINT	hours [0...23]
minutes	Output	UINT	minutes [0...59]
seconds	Output	UINT	seconds [0...59]

Name	Return	Data Type	Describe
OUT	Return	DINT	

5.13.3 SET_RTC

Describe: Set system clock. The function successfully returns 0



Name	Input / Output	Data Type	Describe
year	Input	UINT	year [2000...2099]
month	Input	UINT	month [1...12]
daymonth	Input	UINT	day of month [1...31]
hours	Input	UINT	hours [0...23]
minutes	Input	UINT	minutes [0...59]
seconds	Input	UINT	seconds [0...59]

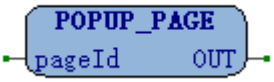
Return:

Name	Input / Output	Data Type	Describe
OUT	Return	DINT	

5.13.4 POPUP_PAGE

Describe: pop-up page, return:

0: Success. 1: Popped out already. 2. Page Id does not exist



Name	Input / Output	Data Type	Describe
pageId	Input	UINT	The ID of the page to be popped out.

Return:

Name	Return	Data Type	Describe
OUT	Return	DINT	

5.13.5 GEN_RAND

Describe: Generates a 32-bitrandom number

Name	Input / Output	Data Type	Describe
maxOut	Input	UDINT	Maximum output. If max Out is equal to 0, then there is no limit.

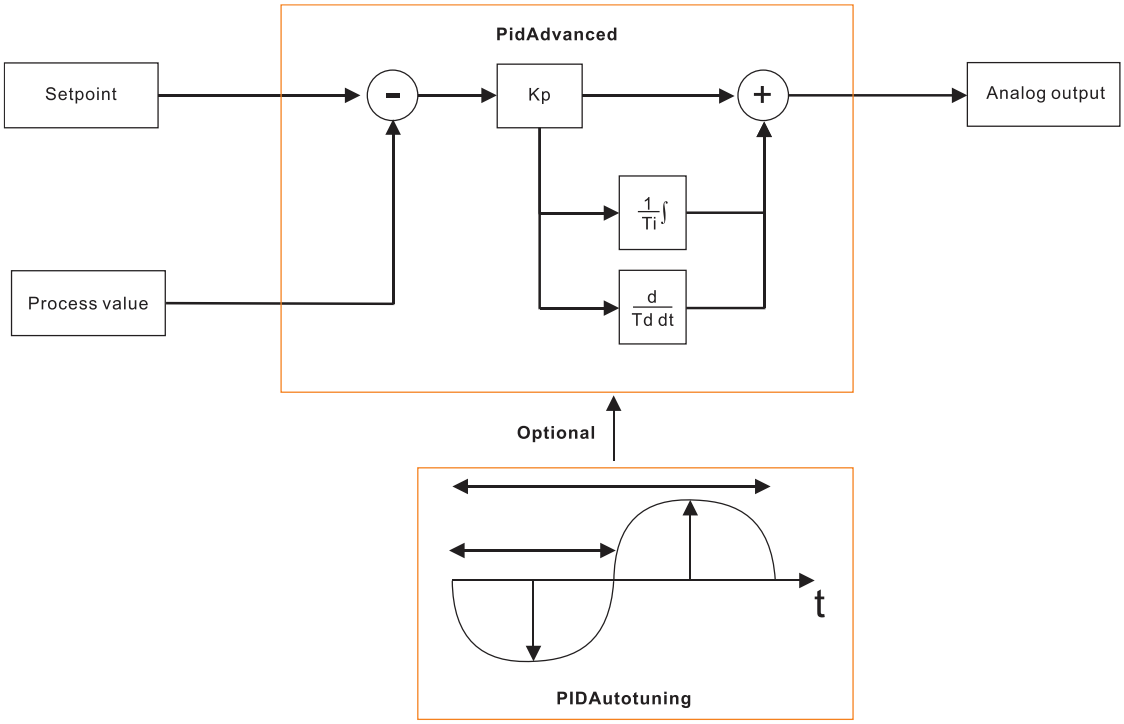
Return:

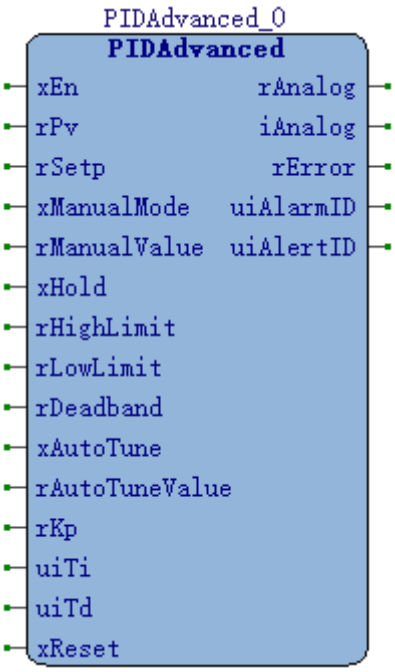
Name	Return	Data Type	Describe
OUT	Return	UDINT	

5.13.6 PIDAdvanced

Describe: The PIDAdvanced function block is designed for monitoring and controlling a wide variety of dependant processes. It includes the functions of dead band, manual mode, and hold. The PIDAdvanced function block can be used on various applications requiring regulation such as temperature control, pressure control, and flow control. This function block incorporates a PID algorithm.

NOTE: The PIDAdvanced function block can be extended with the autotuning function.





Name	Input / Output	Data Type	Describe
xEn	Input	BOOL	Enable/Disable the function block. TRUE: Enable
rPv	Input	REAL	Actual value from the process. **NOTE**: The process value rPV and the setpoint rSetp must have the same unit. Unit: User unit
rSetp	Input	REAL	Value of setpoint. Unit: User unit
xManualMode	Input	BOOL	TRUE: Enables manual mode, the outputs rAnalog and iAnalog take the value of rManualValue. FALSE: Disables manual mode, the calculation starts at the last entered manual value rManualValue. Default: FALSE
rManualValue	Input	REAL	Manual value for the analog outputs iAnalog and rAnalog. Unit: %
xHold	Input	BOOL	This pin is used to freeze the analog output (xHold=TRUE). The internal calculation of the integral term also freezes. TRUE: Holds the PID action FALSE: Resumes the PID action Default: FALSE NOTE: If set to FALSE the calculation starts at the last frozen value off the analog outputs (rAnalog and iAnalog).
rHighLimit	Input	REAL	High limit of PID output. Default: 100.0. Unit: % rHighLimit = 100.0 (%), rLowLimit = 0.0 (%): The PIDAdvanced controls the analog output between 0.0 and 100.0 %. rHighLimit = 100.0 (%), rLowLimit = 20.0 (%): The PIDAdvanced controls the analog output between 20.0% and 100.0 %.
rLowLimit	Input	REAL	Low limit of PID output. Default: 0.0. Unit: %
rDeadband	Input	REAL	The value at this pin defines the limit of the dead band for a detected error. This dead band function is used to make the PIDAdvanced output to settle down. Example: rSetp = 45.0 °C, rPv = 0…60.0 °C, rHighLimit = 100.0 (%), rLowLimit = 0.0 (%), rDeadband = 2.5 °C (+/-), The following

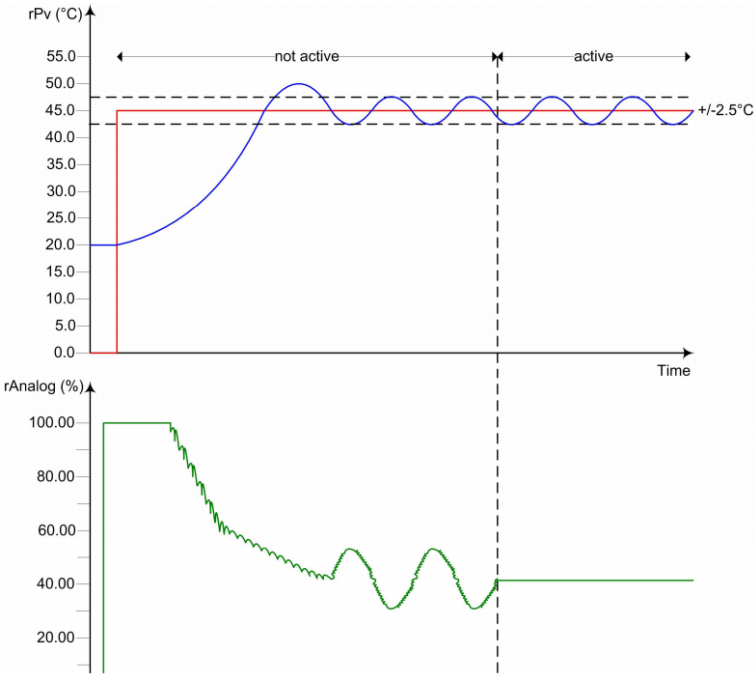
Name	Input / Output	Data Type	Describe
			graphic presents the dead band function:
xAutoTune	Input	BOOL	FALSE: Not active TRUE: External autotuning function is active. Default: FALSE
rAutoTuneValue	Input	REAL	Value for rAnalogOut determined by the external autotuning function. Default: 0.0. Unit: %
rKp	Input	REAL	Proportional gain (-300.0 to 300.0 Default: 4.0)
uiTi	Input	UINT	Value of the integral time configured by you or autotuning. Default: 0. Unit: 0.1s
uiTd	Input	UINT	Value of derivative damping time configured by you. Default: 0. Unit: 0.1s
xReset	Input	BOOL	FALSE: not active TRUE: Alarms are reset by a rising edge. NOTE: This pin is used to reset only the alarms. To reset the alarms, modify the parameter which caused the alarm and give a rising edge on pin xReset. NOTE: Alerts are reset automatically
rAnalog	Output	REAL	Analog output (rLowLimit to rHighLimit %)
iAnalog	Output	INT	Analog output (0 to1000 *0.1%)
rError	Output	REAL	Error between rSp and rPv (User Unit)
uiAlarmID	Output	UINT	Alarm identification
uiAlertID	Output	UINT	Alert identification

◆ xManualMode is set to TRUE: the manual mode is active and the outputs rAnalog and iAnalog take the value of rManualValue.

◆ xManualMode is set to FALSE: the calculation starts at the last entered manual value rManualValue.

◆ rHighLimit = 100.0 (%), rLowLimit = 0.0 (%):The PIDAdvanced controls the analog output between 0.0 and 100.0 %.

◆ rDeadband: The value at this pin defines the limit of the dead band for a detected error. This dead band function is used to make the PIDAdvanced output to settle down. Example: rSetp = 45.0 °C rPv = 0…60.0 °C rHighLimit = 100.0 (%) rLowLimit = 0.0 (%) rDeadband = 2.5 °C (+/-) The following graphic presents the dead band function:



◆ Proportional Gain rKp

rKp > 0: Direct mode, for example, control for heating

Kp = 0: The outputs rAnalog and iAnalog

Kp < 0: Reverse mode, for example, control for cooling (inverse control).

◆ Integral Time uiTi

uiTi = 1: Fast integration time, causes a fast influence on the outputs rAnalog and iAnalog.

uiTi = 10: 10 times slower than the fast integration time (a) and causes a slower influence on the outputs rAnalog and iAnalog.

uiTi = 0: uiTi is deactivated

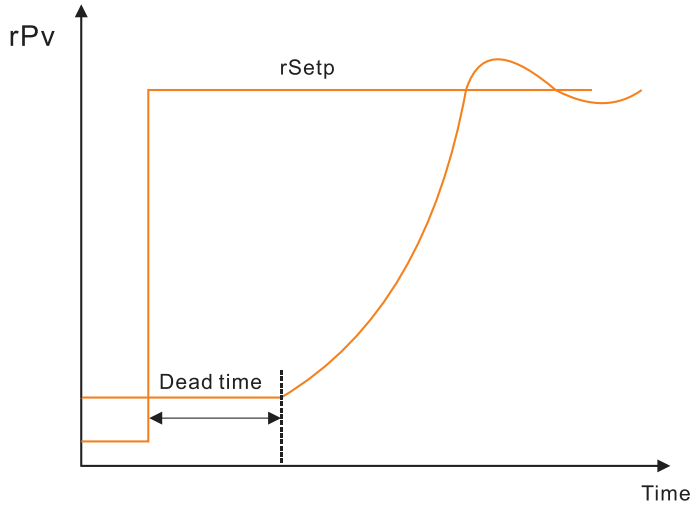
◆ Derivate Time uiTd

uiTd = 1: The smallest damping, causes a high influence to the outputs rAnalog and iAnalog.

uiTd = 10: 1/10 of the smallest damping, causes a lower influence on the outputs rAnalog.

uiTd = 0: uiTd is deactivated.

NOTE: In systems with dead time, uiTd should be set to 0. The value of uiTd must be greater than the cycle time. If it is less than the cycle time, then the uiTd value is overwritten with the value of the cycle time.



◆ uiAlarm ID Description

Alarm Bit	Alarm Cause	Effect
0	Application cycle time is greater than 2000ms	The outputs rAnalog and ianalog are set to 0.
1	Invalid value for the parameters (rHighLimit, rLowLimit)	
2	Invalid value of the parameter rDeadband	
3	Invalid values of the parameters rkp, uiTi or uiTd	
4~15	Reserved	

◆ The output uiAlertID represents a value from 0 to 15 whereby each bit represents a detected alert. The table contains the bits and their description:

Alarm Bit	Cause	Effect
0	Invalid value of the input rManualValue	rAnalog and ianalog outputs are set to rHighLimit or rLowLimit
1	rLowlimit is equal to rHighLimit	rAnalog and iAnalog outputs are set to rLowLimit
2	Autotuning is active, the input xAutoTune is set to TRUE.	rAnalog and ianalog outputs are set to rAutoTune Value
3	rkp is set to 0.0	rAnalog and iAnalog outputs are set to 0
4~15	Reserved	

◆ Troubleshooting

Alarm/Alert	Problem	Solution
uiAlarmID.0	Application cycle time is greater than 2000 ms.	Reduce the application and size.
uiAlarmID.1	rLowLimit > rHighLimit or rLowLimit > 100.0 or rLowLimit < 0.0 or rHighLimit > 100.0 or rHighLimit < 0.0	Verify that the parameter values are within their respective ranges.
uiAlarmID.2	rDeadBand < 0.0 rDeadBand > rHighLimit	Verify that the parameter values are within their respective ranges.
uiAlarmID.3	rKp < -300.0 or rKp > +300.0 or uiTi < 0 or uiTi > 6000 or uiTd < 0 or uiTd > 6000	Verify that the parameter values are within their respective ranges.

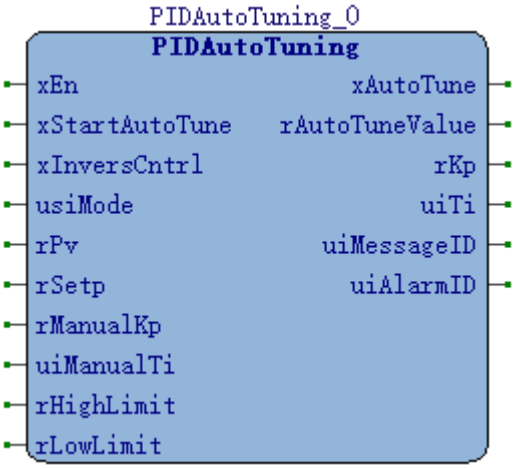
uiAlertID.0	rManualValue > rHighLimit rManualValue < rLowLimit	Verify that the parameter values are within their respective ranges.
uiAlertID.1	rLowLimit is equal to rHighLimit	Verify that the parameter values are within their respective ranges.
uiAlertID.2	Autotuning is active, the input xAutoTune is set to TRUE.	Wait until autotuning is completed.
uiAlertID.3	rKp is set to 0.0	Verify that the parameter values are within their respective ranges.

5.13.7 **PIDAutoTuning**

Describe: The PIDAutoTuning function block measures the dynamic response of a control system and calculates automatically the parameters uiTi (integral time in seconds) and rKp (proportional gain) for that control system. These parameters can be connected to the parameter inputs of a PID AFB (for example PIDAdvanced).

NOTE 1: This function block must be used together with the PIDAdvanced. When autotuning is enabled, it induces oscillations in the process around the set_point. After the completion of 3 oscillations, autotuning calculates a set of PI parameters.

NOTE2: During the autotuning process the system is set to the minimum and maximum operating limits to measure the process response time. Ensure that the minimum and maximum operating limits are set correctly.



Name	Input / Output	Data Type	Describe
xEn	Input	BOOL	Enable the function block
xStartAutoTune	Input	BOOL	Start / Stop the Auto tuning function (TRUE: rising edge activate autotuning, FALSE: autotuning disabled)
xInversCntrl	Input	BOOL	Inverse the control (TRUE: Cooling, FALSE: Heating)
usiMode	Input	USINT	Modifies the PI parameters of the outputs rKp and uiTi. 0= Manual parameter 1= ATR Para (slow) 2= ATR Para (medium) 3= ATR Para (fast) **NOTE:** If you have not performed autotuning before, set usiMode to 0 (manual parameters)

rPv	Input	REAL	Process value (User Unit)
rSetp	Input	REAL	Set point (User Unit) **NOTE:** rSetp is the value of the setpoint and is the intended working process value for the PIDAdvanced AFB.
rManualKp	Input	REAL	Manual process gain (-300.0 to 300.0) Note: When usiMode is set to 0, the rKp is active.
uiManualTi	Input	UINT	Manual integral time (0 to 6000 *0.1 s) **NOTE:** The value is active at uiTi if usiMode is set to 0.
rHighLimit	Input	REAL	High limit of PID output. (0.0 to 100.0 %)
rLowLimit	Input	REAL	Low limit of PID output. (0.0 to 100.0 %)
xAutoTune	Output	BOOL	TRUE: Autotuning process is active. FALSE: Autotuning process is not active. **NOTE:** If xAutoTune is set from TRUE to FALSE (falling edge), autotuning is finished.
rAutoTuneValue	Output	REAL	Auto tuning Value (rLowLimit to rHighLimit %)
rKp	Output	REAL	Manual or calculated proportional gain (-300.0 to 300.0)
uiTi	Output	UINT	Manual or calculated integral time (0 to 6000 * 0.1 s)
uiMessageID	Output	UINT	Alarm message
uiAlarmID	Output	UINT	Alarm identification

◆ rSetp, rPv

A fluctuating signal causes an incorrect calculation of the PI parameters.

The setpoint rSetp and the process value rPv must be of the same unit.

◆ xInversCntrl

If xInversCntrl is set to TRUE, the input xInversCntrl inverts the calculation.

the output rAutoTuneValue is 100.0% and rPv increases: set xInversCntrl to FALSE.

the output rAutoTuneValue is 100.0% and rPv decreases: set xInversCntrl to TRUE.

◆ rLowLimit, rHighLimit

These parameters define the range of your process (rAutoTuneValue).

The PID controller (for example, PIDAdvanced AFB) must have the same values. The maximum range is 0.0 to 100.0%.

NOTE: In oversized or undersized systems it is necessary to limit the range of the output rAutoTuneValue (for example, 0.0 to 50.0% for an oversized system).

Results:

The PIDAutoTuning AFB controls the output rAutoTuneValue between 0.00 and 50.00 %. ◦ The process value increases slower (for example temperature).

The calculated PI parameters work better for an oversized system.

◆ usiMode

0 = manual: Manual PI- parameters are set to the outputs rKp and uiTi.

1 = slow: Calculated PI- parameters are set to the outputs rKp and uiTi.

2 = medium: Calculated PI- parameters are set to the outputs rKp and uiTi

3 = fast: Calculated PI- parameters are set to the outputs rKp and uiTi.

◆ uiMessageID

Message Bit	Cause	Effect
0	Tuning is in progress.	Tuning is in progress.
1	Tuning is completed.	Autotuning is finished and the controlling was successful.
2	Tuning is completed. System is oversized.	Autotuning is finished and the calculated PI parameters indicates an oversized system. The calculated parameters or manual values can be used. **NOTE:** The range of the output rAutoTuneValue can be limited with rHighLimit.
3	Tuning is completed. System is undersized.	Autotuning is finished and the calculated PI parameters indicates an undersized system. The calculated parameters or manual values can be used.
4~15	Reserved	Reserved

◆ uiAlarmID

Alarm Bit	Cause	Effect
0	Invalid cycle time.	rAutoTuneValue is set to 0 and autotuning will not start or will be canceled.
1	Invalid value of the parameters rHighLimit or rLowLimit.	
2	Invalid value of the parameters rManualKp or uiManualTi.	
3	Invalid value of the parameter usiMode).	
4	xInversCntrl is modifed during autotuning is running.	
5	The PI parameters cannot be calculated for this system.	◆ Autotuning is finished and the controlling detected an incalculable system. ◆ rAutoTuneValue is set to 0 and autotuning will not start or will be canceled. ◆ The previously calculated PI parameters (rKp, uiTi) will be deleted.
6~15	Reserved	Reserved

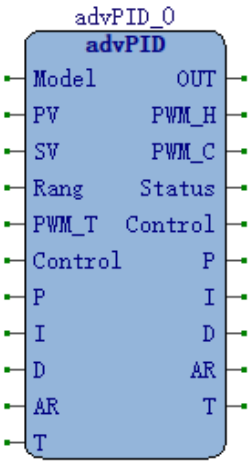
◆ Troubleshooting

Alarm	Problem	Solution
uiAlarmID.0 TRUE	Cycle time > 2000 msec	The application is too large. Reduce the application and check the cycle time.
uiAlarmID.1 TRUE	rHighLimit > rLowLimit rHighLimit > 100.0%	Verify that the parameter values are within their respective ranges.

	rLowLimit = rHighLimit	
uiAlarmID.2 TRUE	rManualKp < -300.0 rManualKp > +300.0 uiManualTi < 0 uiManualTi > 6000	Verify that the parameter values are within their respective ranges.
uiAlarmID.3 TRUE	Invalid value of the parameter usiMode	Verify that the parameter values are within their respective ranges.
uiAlarmID.4 TRUE	xInversCntrl modifications during auto- tuning	Do not modify the value as long as autotuning is in progress.
uiAlarmID.5 TRUE	◆ Auto- tuning run for a long time (>45 minutes). ◆ The system is slow.	◆ Stop autotuning. ◆ Set the PI parameters manually.
uiAlarmID.5 TRUE	◆ Internal variables detected values which are not calculable (for example Time1<=1 second or Time4>2700 seconds). ◆ There is a fluctuating signal of the actual process value (but you cannot see it in the program). ◆ The message Tuning completed system is incalculable will be presented.	If the parameters do not work, set the manual PI parameters (usiMode = 0).

5.13.8 advPID

Describe: The advPID features advanced PID algorithms capable of controlling temperature, humidity, flow rate, tension, pressure, position, speed, etc. it also has functions such as self_tuning, online PID parameter self-correction regulation algorithm (CT-TUNE), and preset PID model parameters. Users do not need complex programming, they only need to set some simple parameters to use it. Note: When using this library, you need to set the task cycle period of AdvPID to 10ms.



Name	Input / Output	Data Type	Describe
Model	Input	INT	The PID algorithm mode selection can be based on the controlled object to choose the corresponding mode, and the PID block will provide the corresponding PID parameters. • 0: Default mode, standard PID control, used for fixed model control objects.

			• 1: Standard PID algorithm, while starting the parameter online intelligent correction mode, in this mode, CT-TUNE algorithm will perform real-time correction of PID parameters based on the adjustment effect, used to control frequent changes in the object model. • 2: Standard PID algorithm, motion control mode, control word will be set to 1ms time base, differential action will be turned off, default running at 10ms cycle, used for tension control, position loop, speed loop control objects. • 3: Incremental algorithm PI control mode, the control word will be set to incremental algorithm, and the differential action will be turned off. It is generally used for actuators with feedback valves or objects that cannot generate large disturbances to the output. • 4: Incremental algorithm PID control mode. Note: When switching modes, the corresponding control word parameters will be changed and the running state will be turned off. Please reset the control word running bit during operation.
PV	Input	INT	PV (Process Variable) represents the value of the controlled object. For example, the temperature, pressure, flow rate, etc. of the controlled object. The last digit is a decimal, for example, 4000 represents 400.0. Range: -32000-32000
SV	Input	INT	Indicate the expected value and set value for the controlled object. The last digit is a decimal, for example, 4000 represents 400.0. Range: -32000-32000.
Rang	Input	INT	Reserve
PWM_T	Input	INT	Set the PWM output cycle, with an effective range of 0.1-120.0 seconds Note: In order to reduce the lifespan loss of peripheral output components, it is recommended to take a value of no less than 5 seconds for mechanical relays And the PWM_T time should be greater than or equal to T. Range: 1-1200
OUT	Output	INT	The output range of PID is determined by the combination of control words bit2 and bit3 Range: Heating mode 0-32000 Cooling Mode -32000-0 Heating and cooling modes -32000-32000
PWM_H	Output	BOOL	PWM hot end output, 1 is ON, 0 is OFF
PWM_C	Output	BOOL	PWM cold end output, 1 is ON, 0 is OFF

<table>
<tr><td>Status</td><td>Output</td><td>Word</td><td>The running status word of the current PID is represented by corresponding bits to indicate the status, and the status bits not listed are reserved bits.
<table>
<tr><th>status bit</th><th>describe</th></tr>
<tr><td>Bit 0</td><td>1: PID is running
0: PID stops running</td></tr>
<tr><td>Bit 1</td><td>1: PID self-tuning is running
0: PID self-tuning not running</td></tr>
<tr><td>Bit 2</td><td>1: PID adaptive is running
0: PID adaptive not running</td></tr>
<tr><td>Bit 3</td><td>1: PWM hot end output ON
0: PWM hot end output OFF</td></tr>
<tr><td>Bit 4</td><td>1: PWM cold end output ON
0: PWM cold end output OFF</td></tr>
<tr><td>Bit 14</td><td>1: PID calculation failure, usually caused by PV exceeding the limit
0: Normal state</td></tr>
<tr><td>Bit 15</td><td>1: PID self-tuning failed
0: Normal state</td></tr>
</table></td></tr>
<tr><td>Control</td><td>InOut</td><td>INT</td><td>The control setting of PID algorithm selects functions based on the 16 bits of the word.
Note: 1. When starting the PID, heating, cooling, or simultaneous control of hot and cold must be selected simultaneously, otherwise the PID will not start normally.
2. Bit2 and Bit3 are combined controls. When joint control of hot and cold is required, please set Bit2 and Bit3 to 1 at the same time. At this time, the out output range is -32000-32000, and the PWM hot and cold ends output.
3. Unlisted bits are reserved bits.
<table>
<tr><th>status bit</th><th>describe</th></tr>
<tr><td>Bit 0</td><td>1: Start PID calculation
0: Close PID, PID will stop running, output is 0</td></tr>
<tr><td>Bit 1</td><td>1: Start the PID self-tuning algorithm and automatically reset it after tuning is completed
0: Disable PID self-tuning algorithm</td></tr>
<tr><td>Bit 2</td><td>1: Output is single ended heating control, PWM only outputs at the heating end, used for heating control, out output range is 0-32000
0: Not selected</td></tr>
<tr><td>Bit 3</td><td>1: Output is single ended cooling control, PWM only outputs at the cold end for cooling control, with an out output range of -32000-0
0: Not selected</td></tr>
<tr><td>Bit 4</td><td>1: Turn off the function of points
0: Not selected</td></tr>
<tr><td>Bit 5</td><td>1: Close differential action
0: Not selected</td></tr>
<tr><td>Bit 6</td><td>1: Activate PID Expert Online Adaptive Algorithm (CT-TUNE)</td></tr>
</table></td></tr>
</table>

			0: Not selected Bit 13: 1: PID calculation cycle base 1ms 0: PID calculation cycle base 100ms Bit 14: 1: Online adaptive CT-TUNE algorithm initiates bandwidth expansion 0: Not selected Bit 15: 1: PID parameter reset 0: None
P	InOut	INT	Proportional band, set the proportional range of PID parameters, with the last decimal place. Range: 0-32000
I	InOut	INT	Integral time, last decimal place, maximum 3200.0 seconds Note: When the value is 0, the integral function is canceled Range: 0-32000
D	InOut	INT	Differential time, with the last decimal place and a maximum of 3200.0 seconds. Note: When the value is 0, the differential action is canceled. Range: 0-32000
AR	InOut	INT	Self-tuning parameter performance level. Range: 0-2
T	InOut	INT	The PID calculation cycle time has an effective range of 0.001-10.0 seconds. Note: The base number is based on the control word bit13. Range: 1-1000.

PID Control

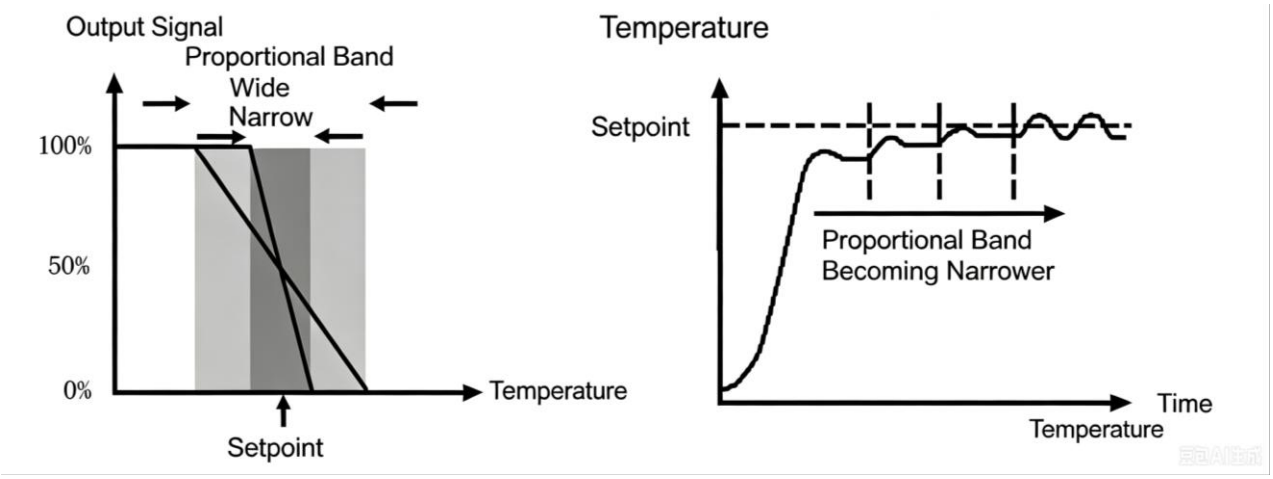
PID control, also known as three-term control, is an algorithm that continuously adjusts the output according to a set of rules to compensate for changes in the process variable. However, its parameter settings must suit the characteristics of the controlled process.

The three terms are:
Proportional band (PB)
Integral time (TI)
Derivative time (TD)

The advPID algorithm is based on a positional (non-incremental) ISA-type algorithm. The algorithm output is the combined result of all three terms. Its simplified Laplace transform expression is:
OP/ER = (100/PB) · (1 + 1/sTI + sTD)

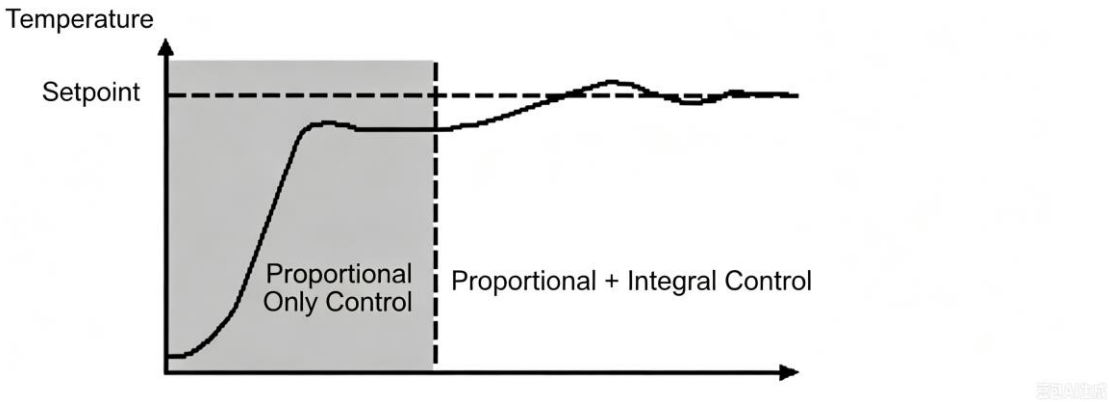
The output is a function of the magnitude and duration of the error signal, as well as the rate of change of the process variable.

The integral and derivative terms can be disabled, allowing control with only the proportional term (P), or with proportional plus integral (PI), or proportional plus derivative (PD).
An example of PI control (i.e., without the D term) is in process industries (flow, pressure, level), where inherent instability and noise tend to cause severe valve oscillation. PD control, on the other hand, can be used for servo-mechanism control.



In proportional-only control, an error must exist between the setpoint and the PV in order to drive the control algorithm. Adding the integral term reduces this error to zero, achieving steady-state control.
For any error between the setpoint and the measured value, the integral term slowly modifies the output level. If the measured value is below the setpoint, the integral term gradually increases the output to reduce the error. If the measured value is above the setpoint, the integral term gradually decreases the output or increases cooling power to correct the error.

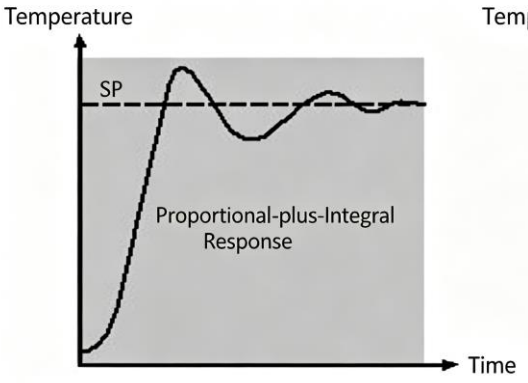
The diagram below illustrates the effect of introducing the integral term.

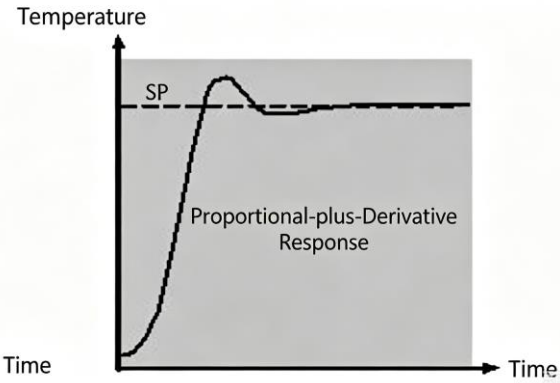


The action of the integral unit is determined by its time setting. The longer the integral time constant, the slower the output changes and the more sluggish the response. If the integral time constant is too short, it may cause overshoot and even oscillation. The integral term can be disabled by setting its value to 0 (off); in this case, manual reset must be enabled.

Derivative Term (TD)

When the error changes rapidly, the derivative term causes a significant change in the output. If the measured value drops quickly, the derivative action increases the output change accordingly, correcting the disturbance before the measured value falls too low. The derivative term is most effective in helping the system recover from small process disturbances.





By controlling the output, the derivative term suppresses the influence of the rate of change of the error. As a result, the output changes, the process variable (PV) changes accordingly, and the effects of transient disturbances are eliminated. Increasing the derivative time shortens the loop's settling time for transient changes.

Derivative is often mistakenly used to suppress overshoot rather than to improve transient response. In fact, derivative should not be relied upon to suppress overshoot during startup, as this impairs steady-state performance. The best way to handle overshoot is to adjust the control parameters and set high/low limits.

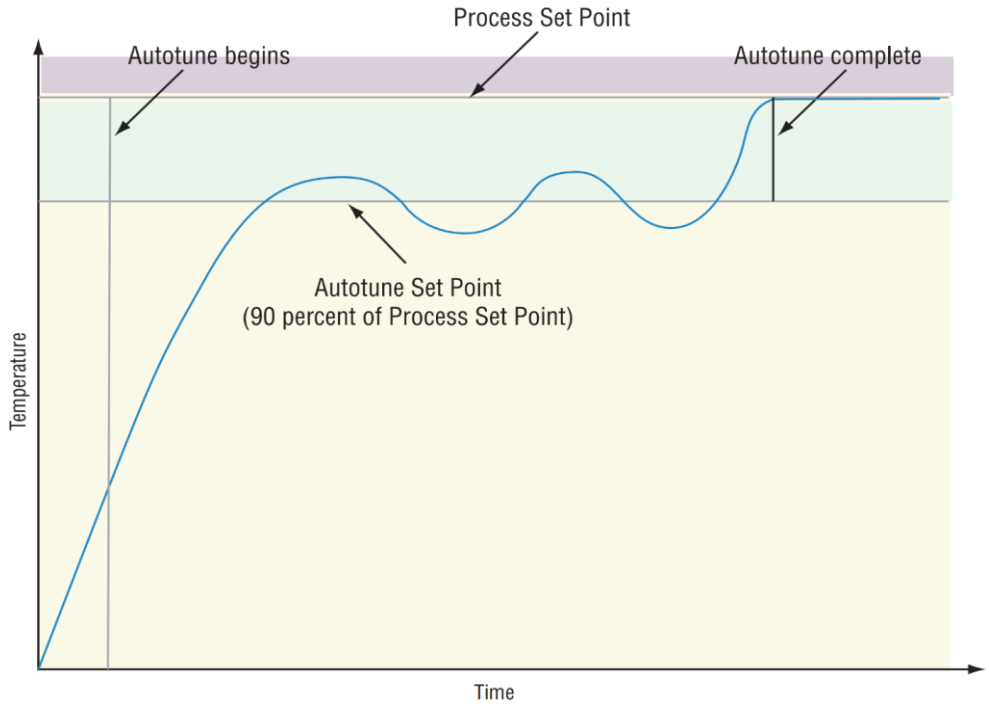
Derivative is generally used to increase loop stability; however, in some cases enabling derivative can actually cause instability. For example, when the PV contains electrical noise, derivative action amplifies the noise, causing excessive output fluctuations. In such situations, it is usually advisable to disable the derivative term and retune the loop parameters.

Setting the derivative time to 0 disables the derivative term.

Self-Tuning

During self-tuning, the controller automatically selects PID parameters for optimal control based on the system's thermal response, although these values may not always provide the best possible control. The PID values can be adjusted either automatically or manually. When self-tuning is initiated, the current setpoint is used to calculate the tuning setpoint, and the controller ignores any setpoint changes until the tuning process is completed. For example, if the setpoint is set to 180° before starting self-tuning, and then changed to 200° after startup, the self-tuning function will still use 180° for tuning. In some cases, reducing the self-tuning setpoint (SP) can help prevent excessive overshoot during the tuning process.

Self-tuning determines the optimal heating and/or cooling PID parameter settings based on the system's response. Self-tuning can be enabled regardless of whether the STC self-correction function is active. The PID parameters generated by self-tuning will remain in effect until the self-tuning function is re-run, the PID values are manually adjusted, or the expert online adaptive algorithm (STC) is activated. If self-tuning cannot be completed within 120 minutes, a timeout occurs and the original settings are restored. The temperature must cross the autotune setpoint five times to complete the self-tuning process. Once completed, the controller will operate at the normal setpoint using the new parameters.



If necessary, additional settings are provided to adjust the effect of the tuning parameters.

The self-tuning function offers adjustable levels for the tuning parameters. After self-tuning is completed, it generates new values for the proportional band (P), integral time (I), and derivative time (D). These three parameters are derived according to the three levels of the AR parameter, as follows:

Level 0: Highest performance with the largest acceptable overshoot.

Level 1: Balanced performance with moderate overshoot.

Level 2: Relatively lower performance with minimum overshoot.

When the control word is set to PI control (with derivative disabled), the function will provide parameters optimized for PI control accordingly.

CT-TUNE Adaptive Algorithm

The CT-TUNE adaptive algorithm is a built-in expert online parameter self-tuning algorithm. It optimizes the PID values of the advPID function block online, enhancing control effectiveness for dynamic processes. CT-TUNE monitors the Process Value (PV) and automatically adjusts the control parameters to maintain the PV at the Setpoint (SV) during setpoint changes and load disturbances. When operating in adaptive control mode, the advPID function block determines the appropriate output signal and progressively adjusts the control parameters over time to optimize responsiveness and stability.

The recommended approach is to first establish initial control PID parameters and then apply the adaptive mode for fine-tuning. Alternatively, you can initially set the control mode to self-tuning, and after the tuning is complete, continue using the adaptive mode for further fine-tuning. In practice, set the initial PID parameter values first, then switch to adaptive mode to refine the PID parameters.

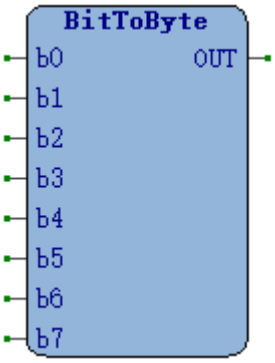
In a rare scenario (characterized by high-frequency, large-amplitude PV fluctuations), CT-TUNE might fail to operate. In such cases, try setting control word bit 14 to 1 to increase the CT-TUNE calculation bandwidth.

Note: This algorithm is effective only in the standard PID algorithm mode.

5.14 Bit conversion functions

5.14.1 **BitToByte**

Describe: Compose a byte from 8 bits



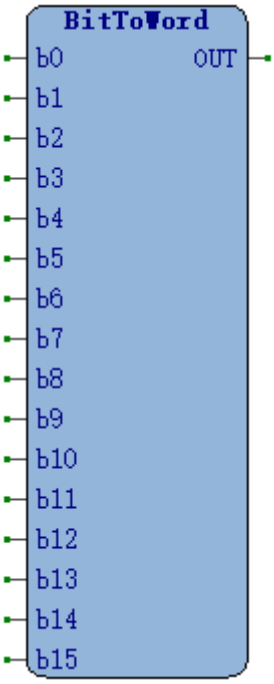
Name	Input / Output	Data Type	Describe
b0	Input	BOOL	Bit 0
b1	Input	BOOL	Bit 1
b2	Input	BOOL	Bit 2
b3	Input	BOOL	Bit 3
b4	Input	BOOL	Bit 4
b5	Input	BOOL	Bit 5
b6	Input	BOOL	Bit 6
b7	Input	BOOL	Bit 7

Return:

Name	Return	Data Type	Describe
OUT	Return	BYTE	

5.14.2 **BitToWord**

Describe: Compose a word from 16 bits



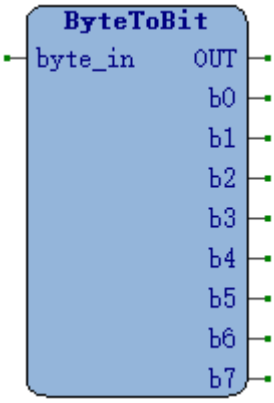
Name	Input / Output	Data Type	Describe
b0	Input	BOOL	Bit 0
b1	Input	BOOL	Bit 1
b2	Input	BOOL	Bit 2
b3	Input	BOOL	Bit 3
b4	Input	BOOL	Bit 4
b5	Input	BOOL	Bit 5
b6	Input	BOOL	Bit 6
b7	Input	BOOL	Bit7
b8	Input	BOOL	Bit8
b9	Input	BOOL	Bit9
b10	Input	BOOL	Bit10
b11	Input	BOOL	Bit11
b12	Input	BOOL	Bit12
b13	Input	BOOL	Bit13
b14	Input	BOOL	Bit14
b15	Input	BOOL	Bit15

Return:

Name	Return	Data Type	Describe
OUT	Return	WORD	

5.14.3 **ByteToBit**

Describe: Split a byte into bits, return byte in



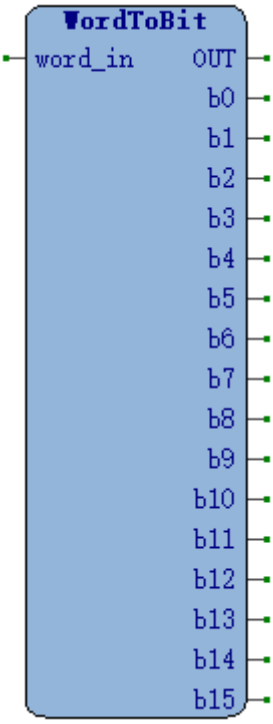
Name	Input / Output	Data Type	Describe
byte_in	Input	BYTE	Byte Input
b0	Output	BOOL	Bit 0
b1	Output	BOOL	Bit 1
b2	Output	BOOL	Bit 2
b3	Output	BOOL	Bit 3
b4	Output	BOOL	Bit 4
b5	Output	BOOL	Bit 5
b6	Output	BOOL	Bit 6
b7	Output	BOOL	Bit 7

Return:

Name	Return	Data Type	Describe
OUT	Return	BYTE	

5.14.4 **WordToBit**

Describe: Split a word into bits, return word in



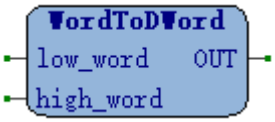
Name	Input / Output	Data Type	Describe
word_in	Input	WORD	Word Input
b0	Output	BOOL	Bit 0
b1	Output	BOOL	Bit 1
b2	Output	BOOL	Bit 2
b3	Output	BOOL	Bit 3
b4	Output	BOOL	Bit 4
b5	Output	BOOL	Bit 5
b6	Output	BOOL	Bit 6
b7	Output	BOOL	Bit7
b8	Output	BOOL	Bit8
b9	Output	BOOL	Bit9
b10	Output	BOOL	Bit10
b11	Output	BOOL	Bit11
b12	Output	BOOL	Bit12
b13	Output	BOOL	Bit13
b14	Output	BOOL	Bit14
b15	Output	BOOL	Bit15

Return:

Name	Return	Data Type	Describe
OUT	Return	WORD	

5.14.5 WordToDWord

Describe: Combines two 16-bit WORD into one 32-bit DWORD



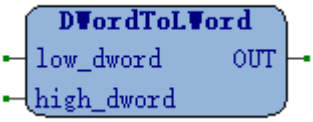
Name	Input / Output	Data Type	Describe
low_word	Input	WORD	Low word
high_word	Input	WORD	High word

Return:

Name	Return	Data Type	Describe
OUT	Return	DWORD	

5.14.6 DWordToLWord

Describe: Combines two 32-bit DWORD into one 64-bit LWORD



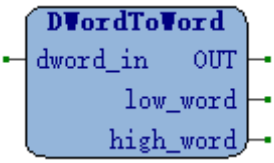
Name	Input / Output	Data Type	Describe
low_word	Input	DWORD	Low dword
high_word	Input	DWORD	High dword

Return:

Name	Return	Data Type	Describe
OUT	Return	LWORD	

5.14.7 DWordToWord

Describe: Split a 32-bit DWORD into two 16-bit WORD. Return dword in.



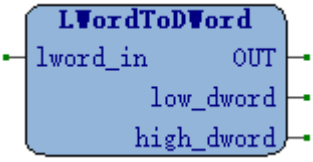
Name	Input / Output	Data Type	Describe
dword_in	Input	DWORD	DWORD Input
low_word	Output	WORD	low WORD
high_word	Output	WORD	High word

Return:

Name	Return	Data Type	Describe
OUT	Return	DWORD	

5.14.8 **LWordToDWord**

Describe: Split a 64-bit LWORD into two 32-bit DWORD. Return lword in.



Name	Input / Output	Data Type	Describe
lword_in	Input	LWORD	
low_dword	Output	DWORD	low DWORD
high_dword	Output	DWORD	low DWORD

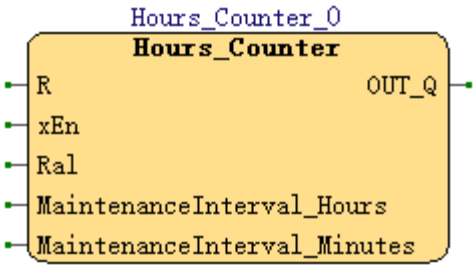
Return:

Name	Return	Data Type	Describe
OUT	Return	LWORD	

5.15 Special functions

5.15.1 Hours_Counter

Describe: A configured time is triggered with a signal at the monitoring input. The output is set when this time has expired.



Name	IN/OUT	Data Type	Describe
R	IN	BOOL	A positive edge (0 to 1 transition) at input R resets output Q and sets a configured value MI at the counter for the duration of the time-to-go (MN).
xEn	IN	BOOL	En is the monitoring input. Device scans the On Time of this input.
Ral	IN	BOOL	A positive edge at input Ral (Reset all) resets the hours counter (OT) and the output, and sets the time-to-go value (MN) to the configured maintenance interval (MI): Output Q = 0; The measured operating hours OT = 0;The time-to-go of the maintenance interval MN = MI.
MaintenanceInterval_Hours	IN	UINT	Maintenance interval to be specified in units of hours and minutes
MaintenanceInterval Minutes	IN	UINT	Maintenance interval to be specified in units of hours and minutes
OUT_Q	OUT	BOOL	

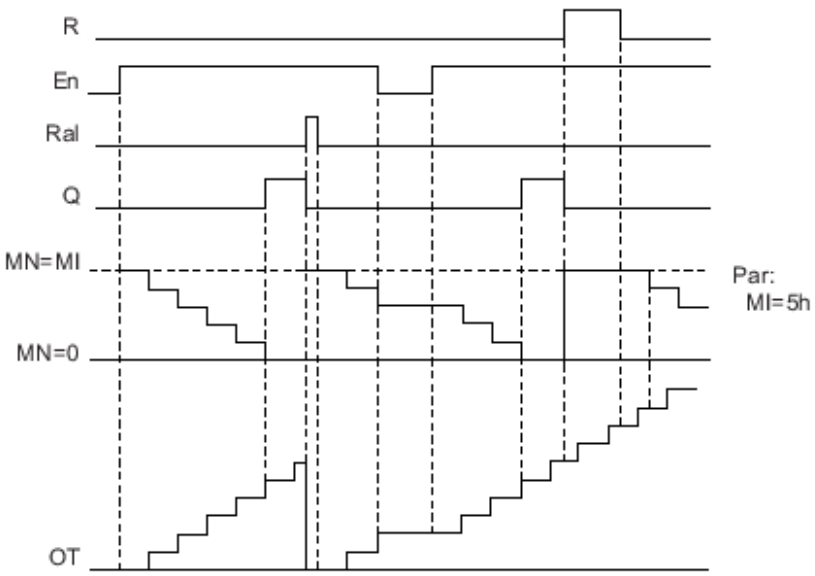
The hours counter monitors input En. As long as the status at this input is 1, SystemePLC calculates the expired time and the time-to-go MN. SystemePLC displays these times when set to configuration mode. The output is set to 1 when the time-to-go is equal to zero.

You reset output Q and the time-to-go counter to the specified value MI with a signal at input R. The operation hour counter OT remains unaffected.

You reset output Q and the time-to-go counter to the specified value MI with a signal at input Ral. The operation hour counter OT is reset to 0.

Depending on your configuration of the Q parameter, the output is either reset with a reset signal at input R or Ral ("Q → R"), or when the reset signal is 1 or the En signal is 0 ("Q → R+En").

Timing diagram



MI = Configured time interval
MN = Time-to-go
OT = Total time expired since the last hi signal at input Ral

Limit value of OT

The value of the operating hours in OT is retained when you reset the hours counter with a signal at input R. The hours counter OT will be reset to zero with a transition from 0 to 1 at Ral. The hours counter OT continues the count as long as En = 1, irrespective of the status at the reset input R. The counter limit of OT is 99999 h. The hours counter stops when it reaches this value.

In programming mode, you can set the initial value of OT. MN is calculated according to the following formula when reset input R is never enabled: MN = MI - (OT % MI). The % operator provides an integer division remainder.

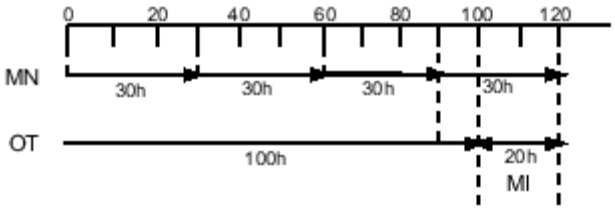
Example:

MI = 30h, OT = 100h

MN = 30 - (100 % 30)

MN = 30 -10

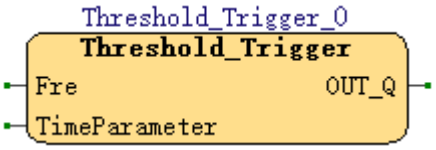
MN = 20h



In runtime mode, the value OT cannot be preset. If the value for MI is changed, there would be no calculation for the MN. MN would take on the value of MI.

5.15.2 Threshold_Trigger

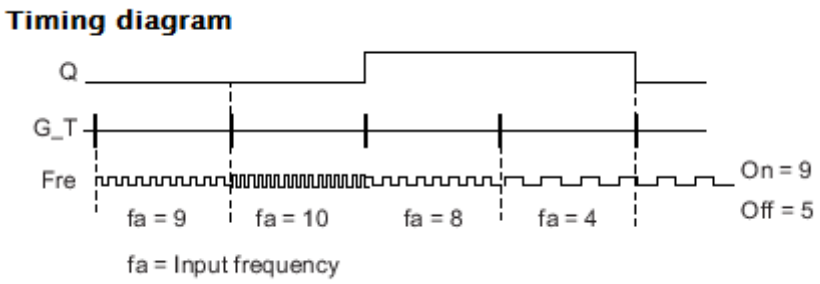
Describe: The outputs switched on and off depending on two configurable frequencies



Name	IN/OUT	Data Type	Describe
Fre	IN	BOOL	The function counts 0 to 1 transitions at input Fre. Transitions from 1 to 0 are not counted.
TimeParameter	IN	TIME	Time interval or gate time during which the input pulses are measured. On: On threshold Range of values: 0000 to 9999 Off: Off threshold Range of values: 0000 to 9999 G_T: Time interval or gate time during which the input pulses are measured. Range of values: 00:00 s to 99:99 s
OUT_Q	OUT	BOOL	Q is set and reset according to the actual value at Cnt and the set thresholds.

The trigger measures the signals at input Fre. The pulses are captured during a configurable period G_T.

Q is set or reset according to the set thresholds. See the following calculation rule.



Calculation rule

If the threshold (On) ⩾ threshold (Off), then:

Q = 1, if fa > On

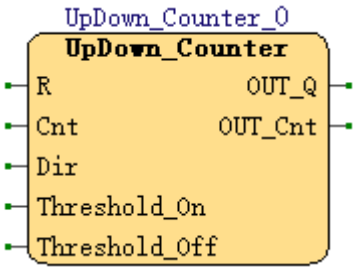
Q = 0, if fa ⩽ Off.

If the threshold (On) < threshold (Off), then Q = 1, if

On ⩽ fa < Off.

5.15.3 UpDown_Counter

Describe: An input pulse increments or decrements an internal value, depending on the parameter setting. The output is set or reset when a configured threshold is reached. The direction of count can be changed with a signal at input Dir.



Name	IN/OUT	Data Type	Describe
R	IN	BOOL	You reset the output and the internal counter value to the start value (StartVal) with a signal at input R (Reset)
Cnt	IN	BOOL	This function counts the 0 to 1 transitions at input Cnt. It does not count 1 to 0 transitions.
Dir	IN	BOOL	Input Dir (Direction) determines the direction of count: Dir = 0: Up， Dir = 1: Down
Threshold_On	IN	UDINT	On threshold
Threshold_Off	IN	UDINT	Off threshold
OUT_Q	OUT	BOOL	Q is set and reset according to the actual value at Cnt and the set thresholds.
OUT_Cnt	OUT	UDINT	

The function increments (Dir = 0) or decrements (Dir = 1) the internal counter by one count with every positive edge at input Cnt.

You can reset the internal counter value to the start value with a signal at the reset input R. As long as R=1, the output Q is 0 and the pulses at input Cnt are not counted.

Output Q is set and reset according to the actual value at Cnt and the set thresholds. See the following rules for calculation.

Calculation rule

If the on threshold ⩾ off threshold, then:

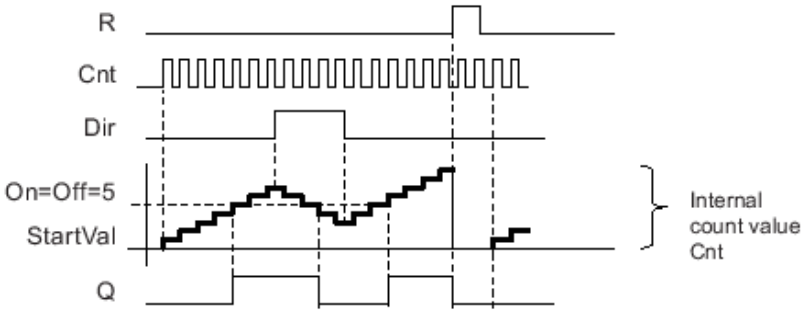
Q = 1, if Cnt ⩾ On

Q = 0, if Cnt < Off.

If the on threshold < off threshold, then:

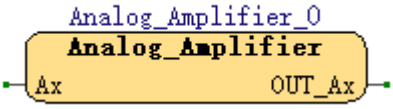
Q = 1, if On ⩽ Cnt < Off.

Timing diagram



5.15.4 Analog_ Amplifier

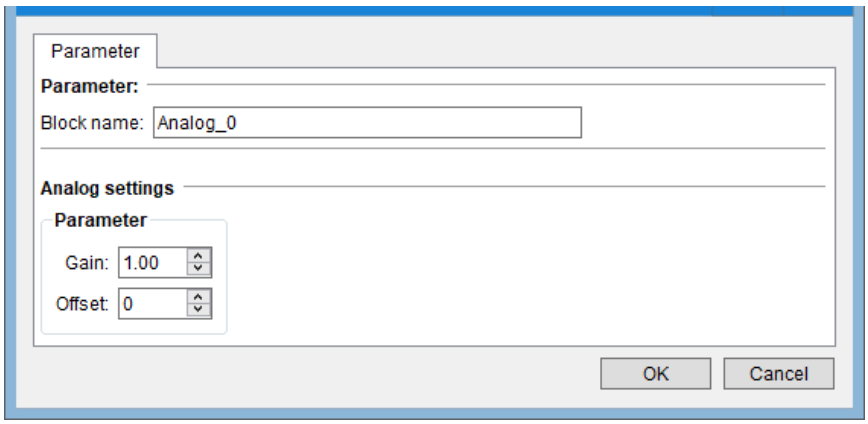
Describe: This FBD amplifies an analog input value and returns it at the analog output.



Name	IN/OUT	Data Type	Describe
Ax	IN	INT	Analog signals
OUT_Ax	IN	INT	Range: -32768 ~ +32767

Set Param

Double-click the command to set the parameters



Gain

Range of values: -10.00 to 10.00

Offset

Range of values: -10000 to 10000

Description of the function

The function reads the value of an analog signal at the analog input Ax.

This value is multiplied by the gain parameter A. Parameter B (offset) is added to the product, as follows:

(Ax * gain) + offset = Actual value Ax.

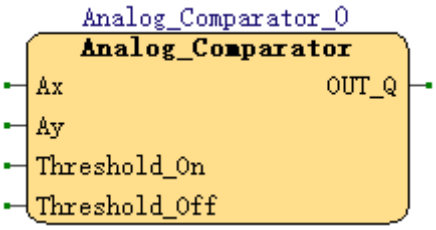
The actual value Ax is OUT_Ax.

Analog output

If you connect this special function to a real analog output, then note that the analog output can only process values from 0 to 1000. To do this, connect an additional amplifier between the analog output of the special function and the real analog output. With this amplifier you standardize the output range of the special function to a value range of 0 to 1000.

5.15.5 Analog_Comparator

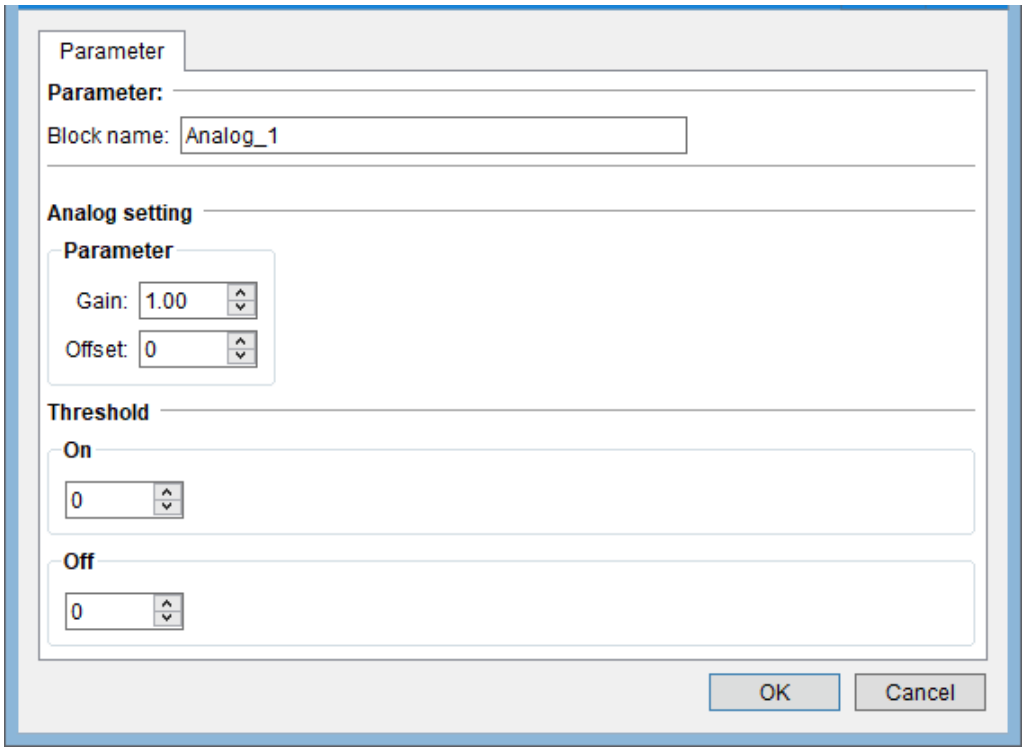
Describe: The output is set and reset depending on the difference Ax-Ay and on two configurable thresholds.



Name	IN/OUT	Data Type	Describe
Ax	IN	INT	Analog signals
Ay	IN	INT	Analog signals
Dir	IN	INT	
Threshold_On	IN	INT	On threshold
Threshold_Off	IN	INT	Off threshold
OUT_Q	OUT	BOOL	Q is set or reset depending on the set thresholds.

Set Param

Double click the command to set the parameters



Gain

Range of values: -10.00 to 10.00

Offset

Range of values: -10000 to 10000

On: On threshold

Range of values: -20000 to 20000

Off: Off threshold

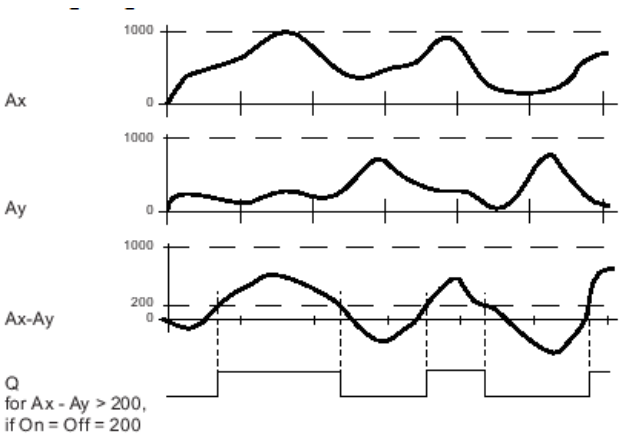
Range of values: -20000 to 20000

Parameter p (number of decimals)

Parameter p applies only to Ax, Ay, Delta, On and Off values displayed in a message text.

Parameter p does not apply to the comparison of on and off values. (The compare function ignores the decimal point.)

Timing diagram



Description of the function

The function reads the value of the signal at the analog input Ax.

This value is multiplied by the value of parameter A (gain). Parameter B (offset) is added to the product, hence

(Ax * gain) + offset = Actual value Ax.

(Ay * gain) + offset = Actual value Ay.

Output Q is set or reset depending on the difference of the actual values Ax – Ay and the set thresholds. See the following calculation rule.

Calculation rule

If threshold On ⩾ threshold Off, then:

Q = 1, if (actual value Ax - actual value Ay) > On

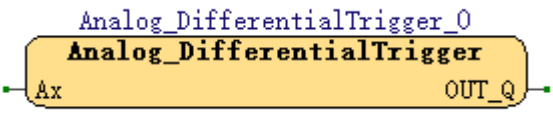
Q = 0, if (actual value Ax - actual value Ay) ⩽ Off.

If threshold On < threshold Off, then Q = 1, then:

On ⩽ (actual value Ax - actual value Ay) < Off.

5.15.6 Analog_DifferentialTrigger

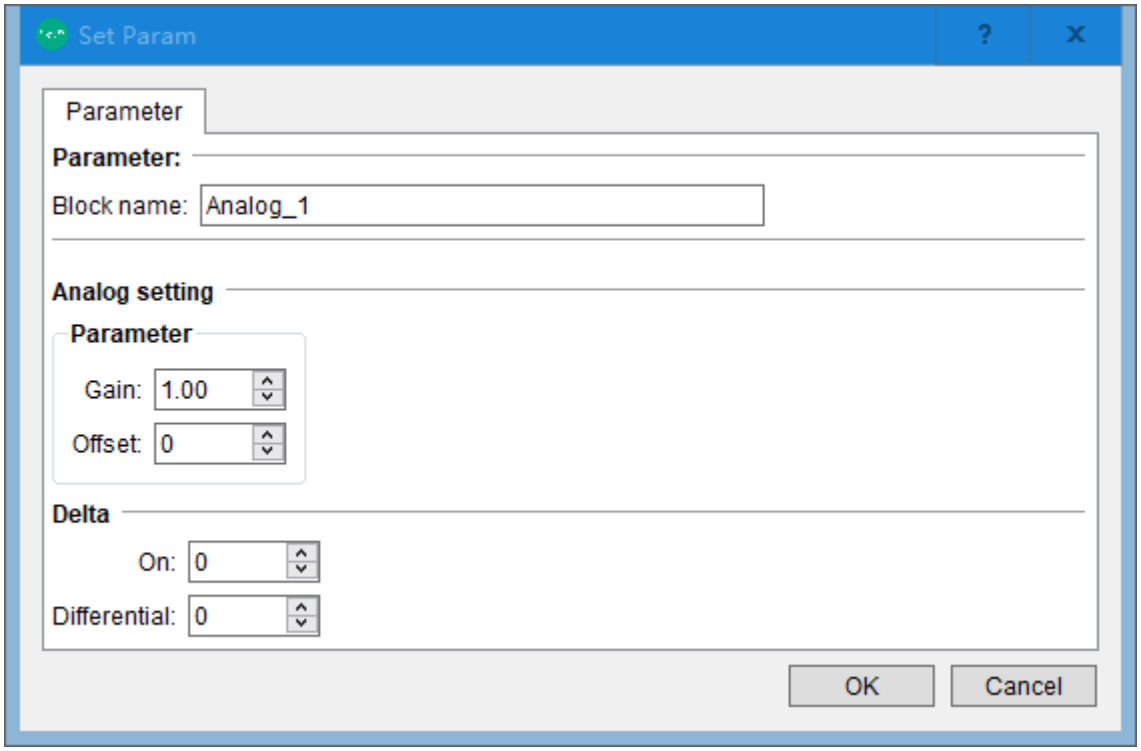
Describe:The output is set and reset depending on a configurable threshold and a differential value.



Name	IN/OUT	Data Type	Describe
Ax	IN	INT	Analog signals
OUT_Q	OUT	BOOL	Q is set or reset, depending on the threshold and difference values.

Set Param

Double click the command to set the parameters



Gain

Range of values: -10.00 to 10.00

Offset

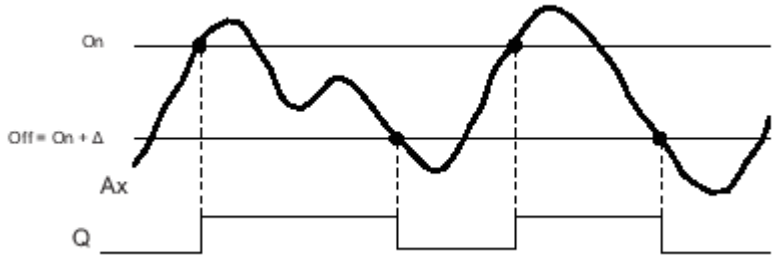
Range of values: -10000 to 10000

On: On/Off threshold

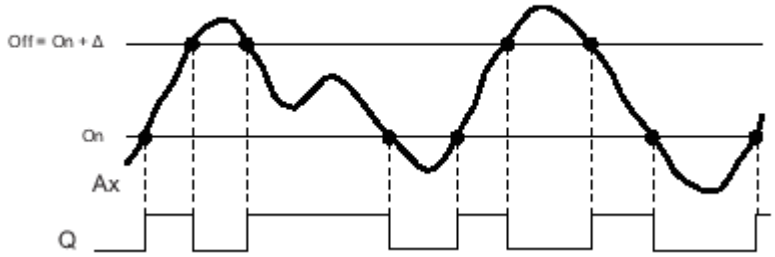
Range of values: -20000 to 20000

DifferentialΔ: Differential value for calculating the off parameter
Range of values: -20000 to 20000

Timing diagram A: Function with negative difference Delta



Timing diagram B: Function with positive difference Delta



Description of the function

The function fetches the analog signal at input Ax.

Ax is multiplied by the value of the A (gain) parameter, and the value at parameter B (offset) is added to product, i.e.

(Ax * gain) + offset = actual value of Ax.

Output Q is set or reset, depending on the set (On) threshold and difference value (Delta). The function automatically calculates the Off parameter: Off = On + Delta, whereby Delta may be positive or negative. See the calculation rule below.

Calculation rule

When you set a negative differential value Delta, the On threshold ≥ Off threshold, and:

Q = 1, if the actual value Ax > On

Q = 0, if the actual value Ax ≤ Off.

See the timing diagram A.

When you set a positive differential value Delta, the On threshold < the Off threshold, and Q = 1, if:

On ≤ the actual value Ax < Off.

See the timing diagram B.

5.15.7 Analog_Filter

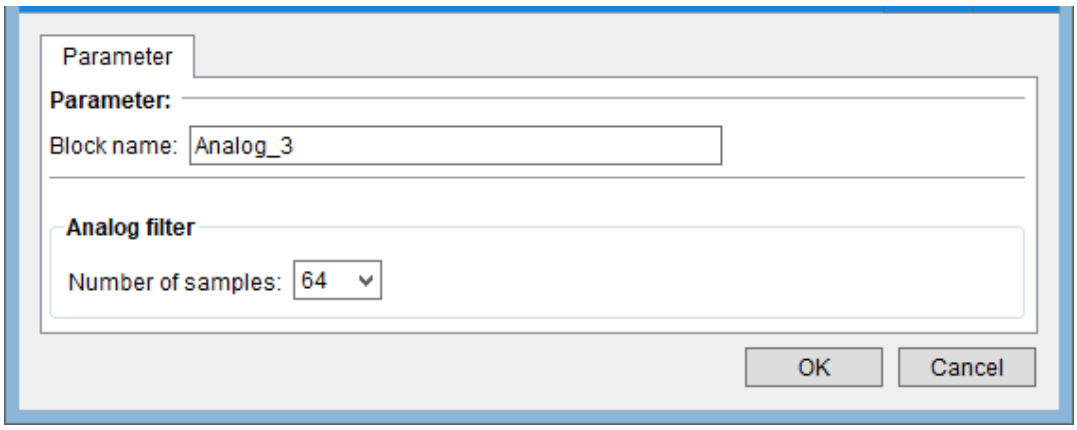
Describe: The Analog filter function block is used to smooth the analog input signal.

Name	IN/OUT	Data Type	Describe
Ax	IN	INT	Analog signals

OUT_AQ	OUT	INT	AQ outputs an average value of the analog input Ax over the current number of samples, and it is set or reset depending on the analog input and the number of samples.

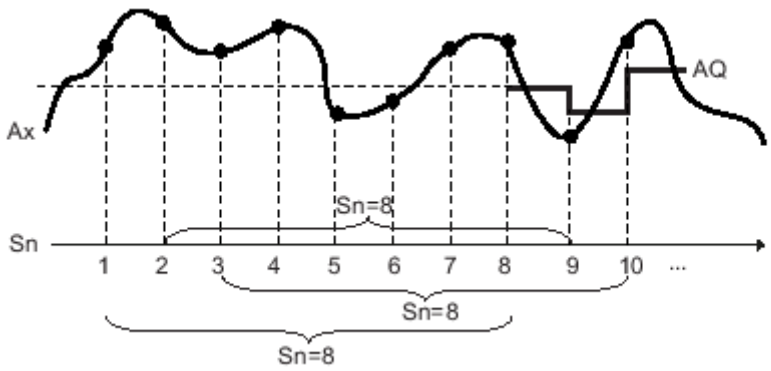
Set Param

Double click the command to set the parameters



After you set the parameter, the analog filter calculates the average value of the samples and assigns this value to AQ.

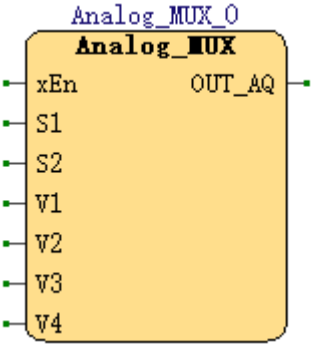
Timing diagram



The function outputs the average value after sampling the analog input signal according to the set number of samples. This SFB can reduce the error of analog input signal.

5.15.8 Analog_MUX

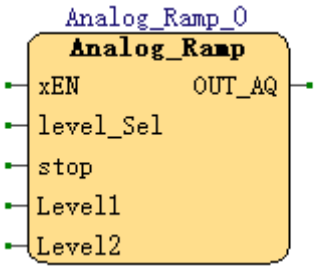
When enabled, the analog multiplexer SFB displays one of four pre-defined analog values, depending on input conditions. S1 = 0 and S2 = 0: The value V1 is issued. S1 = 0 and S2 = 1: The value V2 is issued. S1 = 1 and S2 = 0: The value V3 is issued. S1 = 1 and S2 = 1: The value V4 is issued.



Name	IN/OUT	Data Type	Describe
xEn	IN	BOOL	1 on input En (Enable) switches, dependent on S1 and S2, a parameterized analog value to the output AQ.0 on input EN switches 0 to the output AQ.
S1	IN	BOOL	S1 and S2 (selectors) for selecting the analog value to be issued.
S2	IN	BOOL	S1 and S2 (selectors) for selecting the analog value to be issued.
V1	IN	INT	S1 = 0 and S2 = 0: The value V1 is issued
V2	IN	INT	S1 = 0 and S2 = 1: The value V2 is issued
V3	IN	INT	S1 = 1 and S2 = 0: The value V3 is issued
V4	IN	INT	S1 = 1 and S2 = 1: The value V4 is issued
OUT_Q	OUT	INT	

5.15.9 Analog_Ramp

Describe: The Analog Ramp instruction allows the output to be changed from the current level to the selected level at a specified rate.

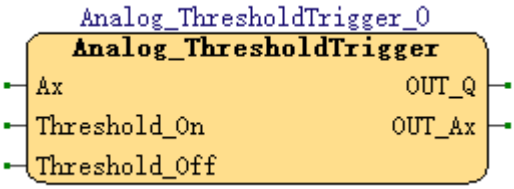


Name	IN/OUT	Data Type	Describe
xEn	IN	BOOL	A change in the status from 0 to 1 at input En (Enable) applies the start/stop level (Offset "B" + StSp) to the output for 100 ms and starts the ramp operation to the selected level.A change in the status from 1 to 0 immediately sets the current level to Offset "B", which makes output AQ equal to 0.
level_Sel	IN	BOOL	level_Sel = 0: The step 1 (level 1) is selected.level_Sel = 1: The step 2 (level 2) is selected.A change in status of Sel causes the current level to start changing to the selected level at the specified rate.
stop	IN	BOOL	A change in the status from 0 to 1 at input St (Decelerated Stop) causes the current level to decrease at a constant rate until the start/stop level (Offset "B" + StSp) is reached. The start/stop level is maintained for 100 ms and then the current level is

			set to Offset "B", which makes output AQ equal to 0.
Level1	IN	INT	
Level2	IN	INT	
OUT_AQ	OUT	INT	The output AQ is scaled using the formula: (Current Level - Offset "B") / Gain "A"

5.15.10 Analog_ThresholdTrigger

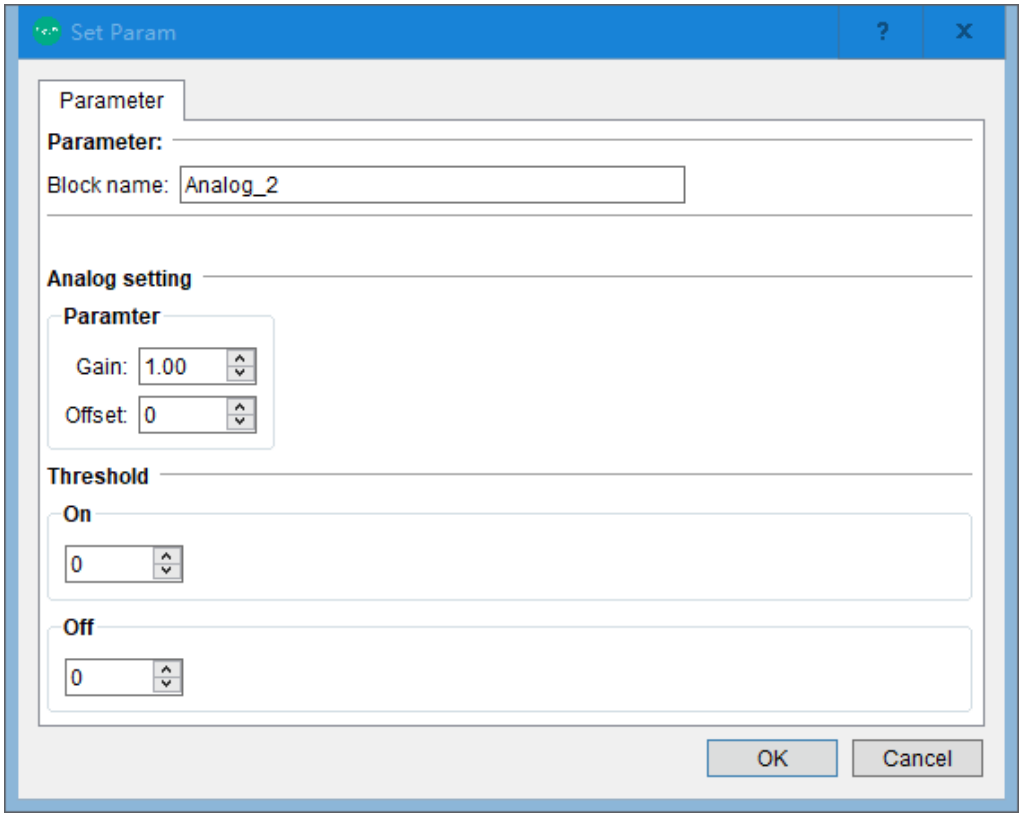
Describe: The output is set or reset depending on two configurable thresholds (hysteresis).



Name	IN/OUT	Data Type	Describe
Ax	IN	INT	Analog signal
Threshold_On	IN	INT	On threshold, Range of values: -20000 to 20000
Threshold_Off	IN	INT	Off threshold, Range of values: -20000 to 20000
OUT_Q	OUT	BOOL	Q is set or reset depending on the set thresholds.
OUT_Ax	OUT	INT	

Set Param

Double click the command to set the parameters



Gain

Range of values: -10.00 to 10.00

Offset

Range of values: -10000 to 10000

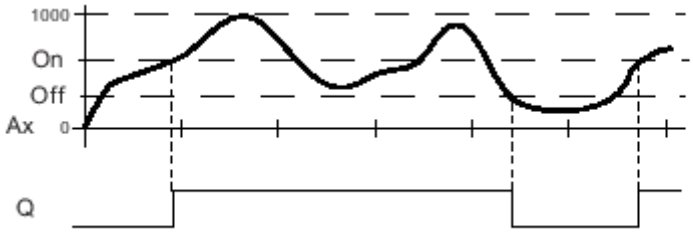
On: On threshold

Range of values: -20000 to 20000

Off: Off threshold

Range of values: -20000 to 20000

Timing diagram



Calculation rule

If threshold (On) ⩾ threshold (Off), then:

Q = 1, if the actual value Ax > On

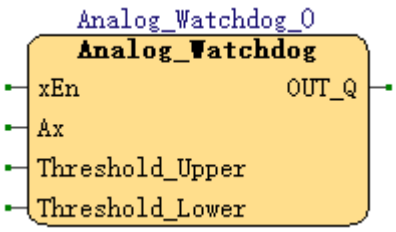
Q = 0, if the actual value Ax ⩽ Off.

If threshold (On) < threshold (Off), then Q = 1,

If On ⩽ the actual value Ax < Off.

5.15.11 Analog_Watchdog

Describe: This special function saves the process variable of an analog input to memory and sets the output when the output variable exceeds or drops below this stored value plus a configurable offset.

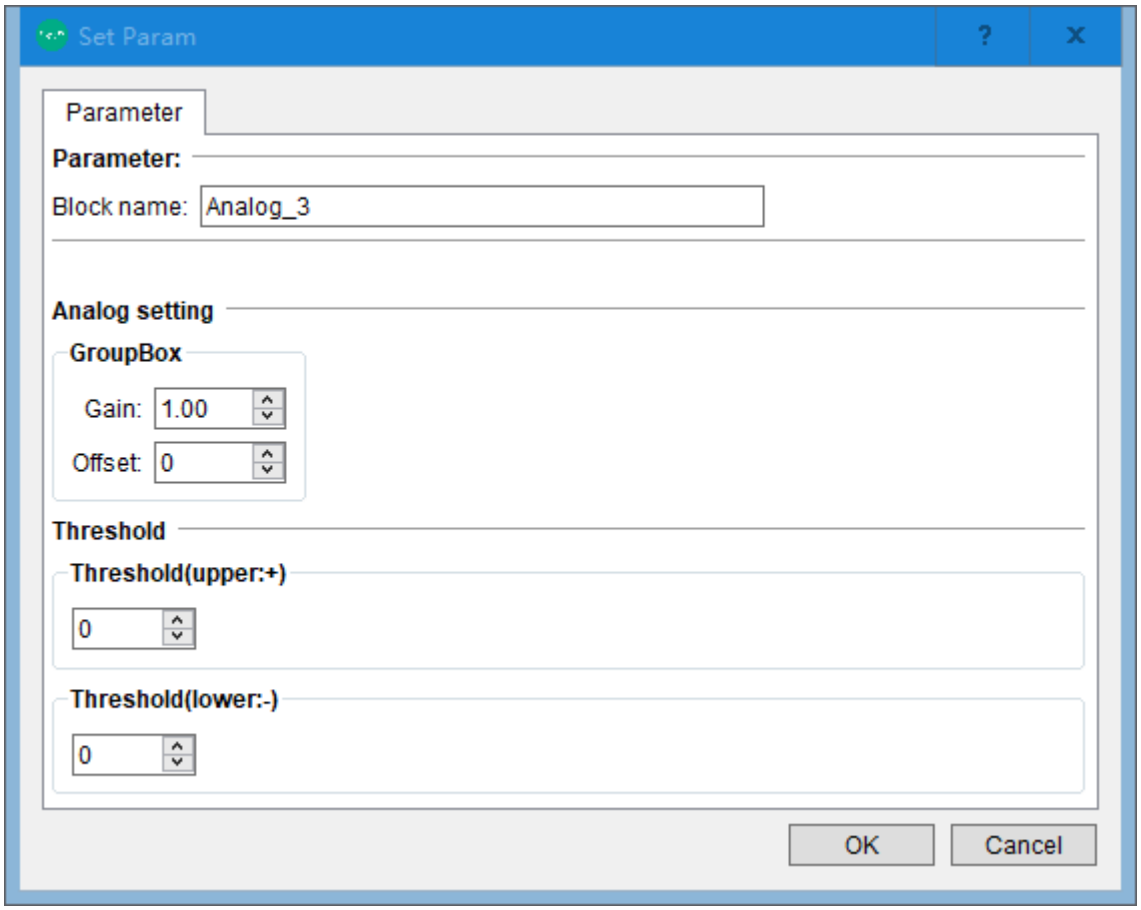


Name	IN/OUT	Data Type	Describe
xEn	IN	BOOL	A positive edge (0 to 1 transition) at input En saves the analog value at input Ax ("Aen") to memory and starts monitoring of the analog range Aen +- Delta.
Ax	IN	INT	
Threshold_Upper	IN	INT	Difference value above Aen: on/off threshold
Threshold_Lower	IN	INT	Difference value below Aen: on/off threshold

OUT_Q	OUT	BOOL	Q is set/reset, depending on the stored analog value and the offset.

Set Param

Double click the command to set the parameters



Gain

Range of values: -10.00 to 10.00

Offset

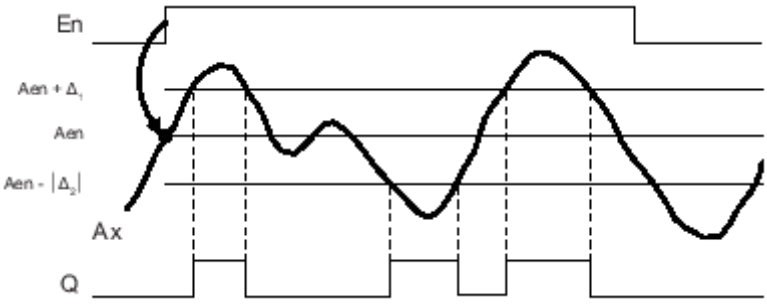
Range of values: -10000 to 10000

Threshold 1: Difference value above Aen: on/off threshold

Range of values: 0 to 20000

Threshold 2: Difference value below Aen: on/off threshold

Range of values: 0 to 20000



Description of the function

A 0 to 1 transition at input En saves the value of the signal at the analog input Ax. This saved process variable is

referred to as "Aen".

Both the analog actual values Ax and Aen are multiplied by the value at parameter A (gain), and parameter B (offset) is then added to the product, as follows:

(Ax * gain) + offset = Actual value Aen, when input En changes from 0 to 1, or

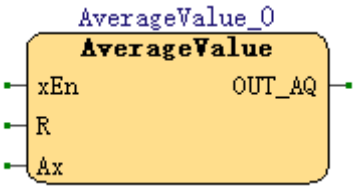
(Ax * gain) + offset = Actual value Ax.

Output Q is set when the signal at input En = 1 and if the actual value at input Ax is out of range of Aen + Threshold 1 / Aen - Threshold 2.

Output Q is reset, when the actual value at input Ax lies within the range of Aen + Threshold 1 / Aen - Threshold 2, or when the signal at input En changes to lo.

5.15.12 AverageValue

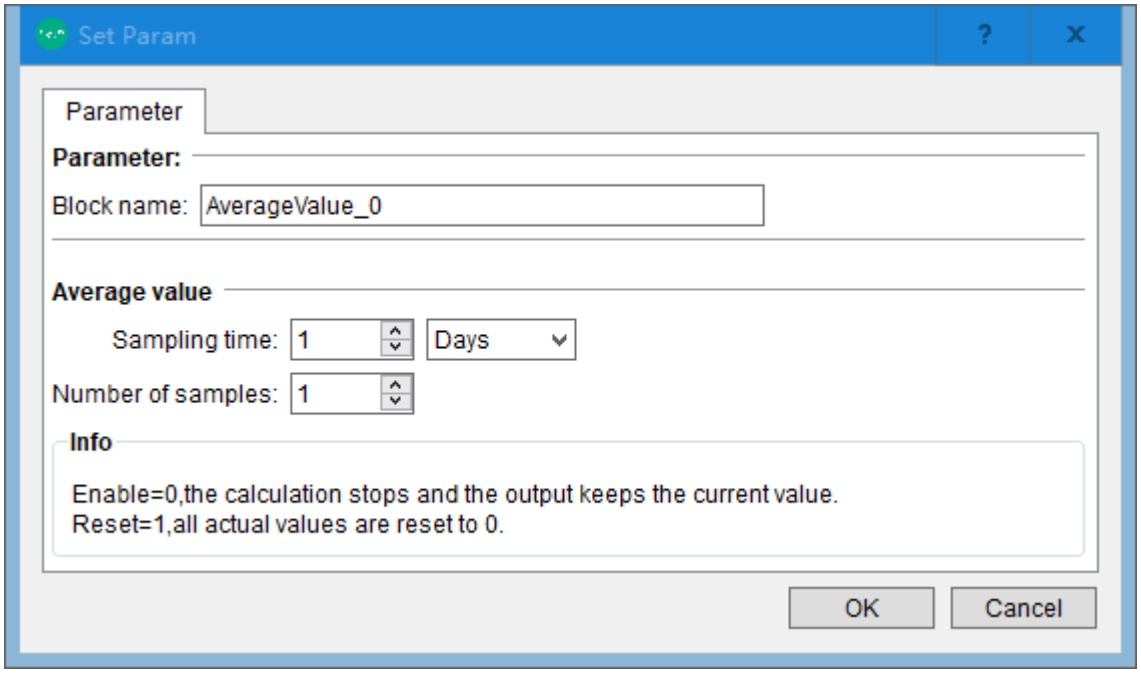
Describe: The average value function samples the analog input signal during configured time period and outputs the average value at AQ.



Name	IN/OUT	Data Type	Describe
xEn	IN	BOOL	A positive edge (0 to 1 transition) at input En (Enable) sets the output AQ to the average value of input Ax after the configured time. A negative edge (1 to 0 transition) holds the output at its last calculated value.
R	IN	BOOL	A positive edge (0 to 1 transition) at input R (Reset) resets the output AQ to 0.
Ax	IN	INT	Analog signals
OUT_AQ	OUT	INT	AQ outputs the average value over the specified time of sampling.

Set Param

Double click the command to set the parameters



St (Sampling time): You can set it to Seconds, Days, Hours or Minutes.

Range of values:

If St = Seconds: 1 to 59

If St = Days: 1 to 365

If St = Hours: 1 to 23

If St = Minutes: 1 to 59

Sn (Number of samples):

Range of values:

If St = Seconds: 1 to St*100

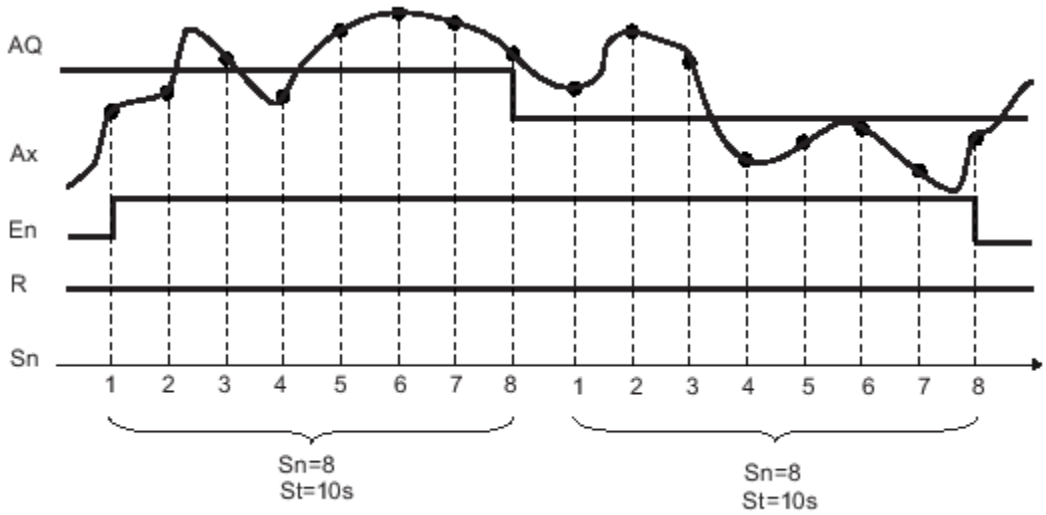
If St = Days: 1 to 32767

If St = Hours: 1 to 32767

If St = Minutes and St ⩽ 5 minutes: 1 to St*6000

If St = Minutes and St ⩾ 6 minutes: 1 to 32767

Timing diagram



Parameter St represents the sampling time and parameter Sn represents the number of samples.

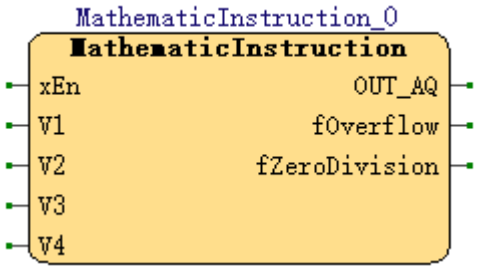
Description of the function

When En = 1, the average value function calculates the average value of the samples during the configured time interval. At the end of the sampling time, this function sets output AQ to this calculated average value.

When En = 0, the calculation stops, and AQ retains the last calculated value. When R = 0, AQ is reset to 0.

5.15.13 Mathematicinstruction

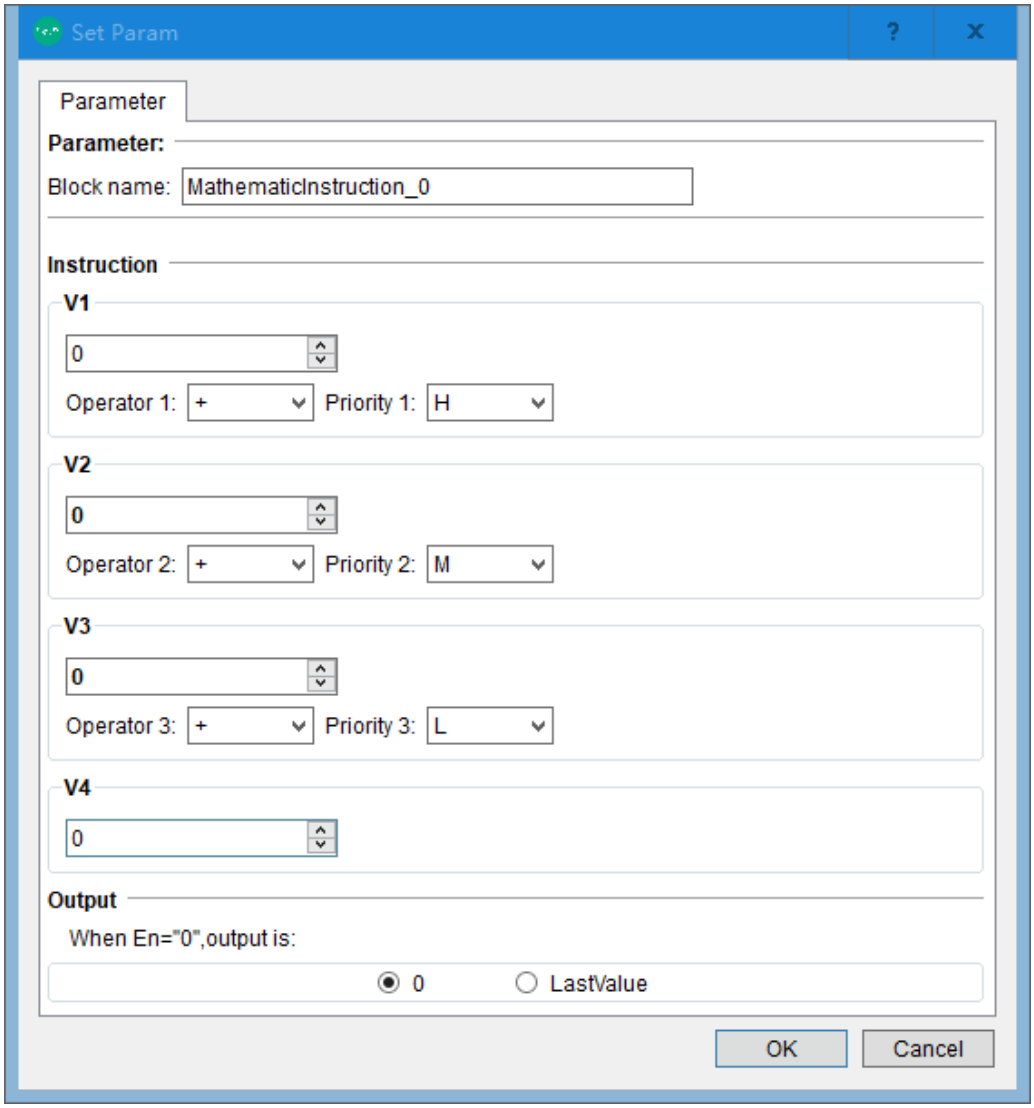
Describe: The mathematic instruction block calculates the value AQ of an equation formed from the user-defined operands and operators.



Name	IN/OUT	Data Type	Describe
xEn	IN	BOOL	A positive edge at input En enables the mathematic instruction function block
V1	IN	INT	Value 1: First operand
V2	IN	INT	Value 2: Second operand
V3	IN	INT	Value 3: Third operand
V4	IN	INT	Value 4: Fourth operand
OUT_AQ	OUT	INT	The output AQ is the result of the equation formed from the operand values and operators.
fOverflow	OUT	BOOL	Output TRUE if an overflow occurs in the current period.
fZeroDivision	OUT	BOOL	Output TRUE when the current period is divided by 0.

Set Param

Double-click the command to set the parameters



V1: Value 1: First operand

V2: Value 2: Second operand

V3: Value 3: Third operand

V4: Value 4: Fourth operand

Range of values: -32768 to 32767

Operator 1: First operator

Operator 2: Second operator

Operator 3: Third operator

Priority 1: Priority of first operation

Priority 2: Priority of second operation

Priority 3: Priority of third operation

Description of the function

The mathematic instruction function combines the four operands and three operators to form an equation. The operator can be any one of the four standard operators: +, -, *, or /. For each operator, you must set a unique priority of High ("H"), Medium ("M"), or Low ("L"). The high operation will be performed first, followed by the medium operation,

and then by the low operation. You must have exactly one operation of each priority. The operand values can reference another previously-defined function to provide the value. The mathematic instruction function rounds the result to the nearest integer value.

The number of operand values is fixed at four and the number of operators is fixed at 3. If you need to use fewer operands, use constructions such as " + 0" or " * 1" to fill the remaining parameters.

You can also configure the behavior of the function when the Enable parameter "En"=0. The function block can either retain its last value or be set to 0.

Examples

The following tables show some simple example mathematic instruction block parameters, and the resulting equations and output values:

V1	Operator1 (Priority 1)	V2	Operator2 (Priority 2)	V3	Operator3 (Priority 3)	V4
12	+ (M)	6	/ (H)	3	- (L)	1

Equation: (12 + (6 / 3)) - 1
Result: 13

V1	Operator1 (Priority 1)	V2	Operator2 (Priority 2)	V3	Operator3 (Priority 3)	V4
2	+ (L)	3	* (M)	1	+ (H)	4

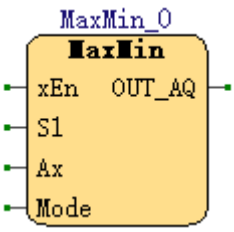
Equation: 2 + (3 * (1 + 4))
Result: 17

V1	Operator1 (Priority 1)	V2	Operator2 (Priority 2)	V3	Operator3 (Priority 3)	V4
100	- (H)	25	/ (L)	2	+ (M)	1

Equation: (100 - 25) / (2 + 1)
Result: 25

5.15.14 MaxMin

Describe: The Max / Min function block records the maximum or minimum value. Mode = 0: AQ=Min. Mode=1: AQ=Max, Mode=2 and S1=0 (low): AQ= Min, Mode =2 and S1=1 (high): AQ=Max, Mode = 3 or a block value is referenced: AQ =Ax.

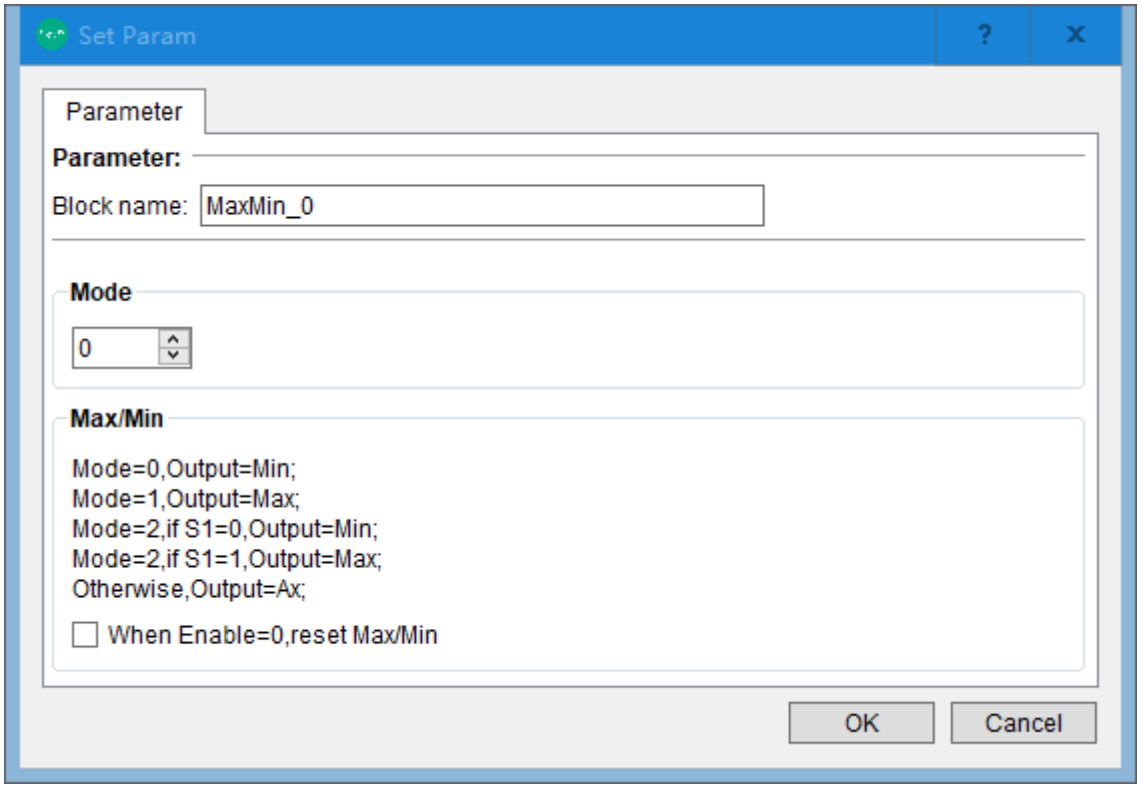


Name	IN/OUT	Data Type	Describe
xEn	IN	BOOL	The function of input En (Enable) depends on the settings of parameter Mode and the selection of check box "when En = 0, reset Max/Min".
S1	IN	BOOL	This input is enabled when you set Mode =2: A positive transition (0 to 1) at input S1 sets the output AQ to the maximum value. A negative transition (1 to 0) at input S1 sets the output AQ to the minimum value.
Ax	IN	INT	Analog signals

Mode	IN	USINT	Possible settings: 0, 1, 2, 3 Mode = 0: AQ = Min Mode = 1: AQ = Max Mode = 2 and S1= 0 (low): AQ = Min Mode = 2 and S1= 1 (high): AQ = Max Mode = 3 or a block value is referenced: AQ = Ax
OUT_AQ	OUT	INT	AQ outputs a minimum, maximum, or actual value depending on the inputs, or is reset to 0 if configured to do so when function is disabled.

Set Param

Double click the command to set the parameters



If you select the check box "when Enable = 0, reset Max/Min":

Enable = 0: The function sets the AQ value to 0.

Enable = 1: The function outputs a value at AQ, depending on the settings of Mode and S1.

If you do not select the check box "when Enable = 0, reset Max/Min":

Enable = 0: The function holds the value of AQ at the current value.

Enable = 1: The function outputs a value at AQ, depending on the settings of Mode and S1.

Mode = 0: The function sets AQ to the minimum value

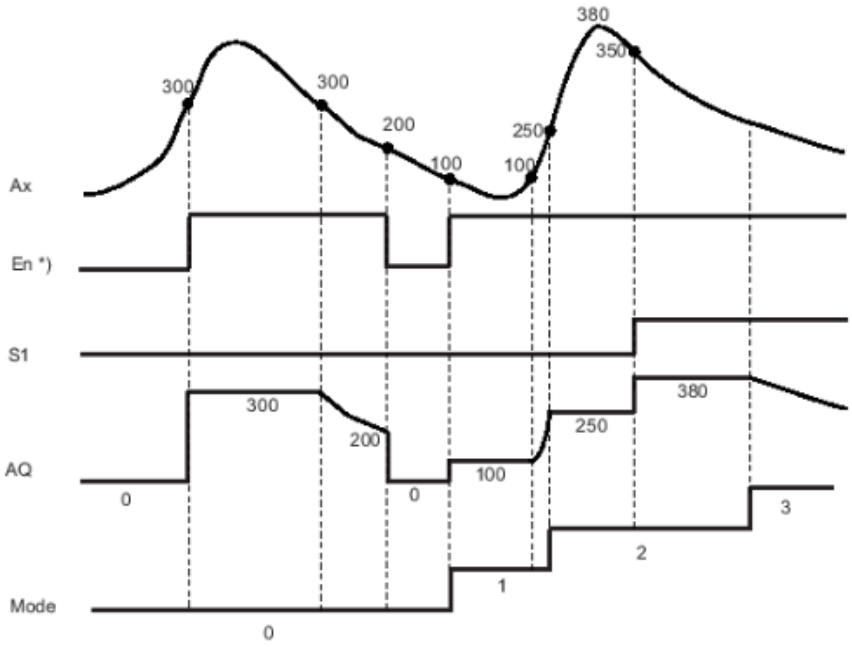
Mode = 1: The function sets AQ to the maximum value

Mode = 2 and S1 = 0: The function sets AQ to the minimum value

Mode = 2 and S1 = 1: The function sets AQ to the maximum value

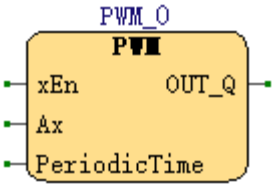
Mode = 3 or a block value is referenced: The function outputs actual analog input value.

Timing diagram



5.15.15 PWM

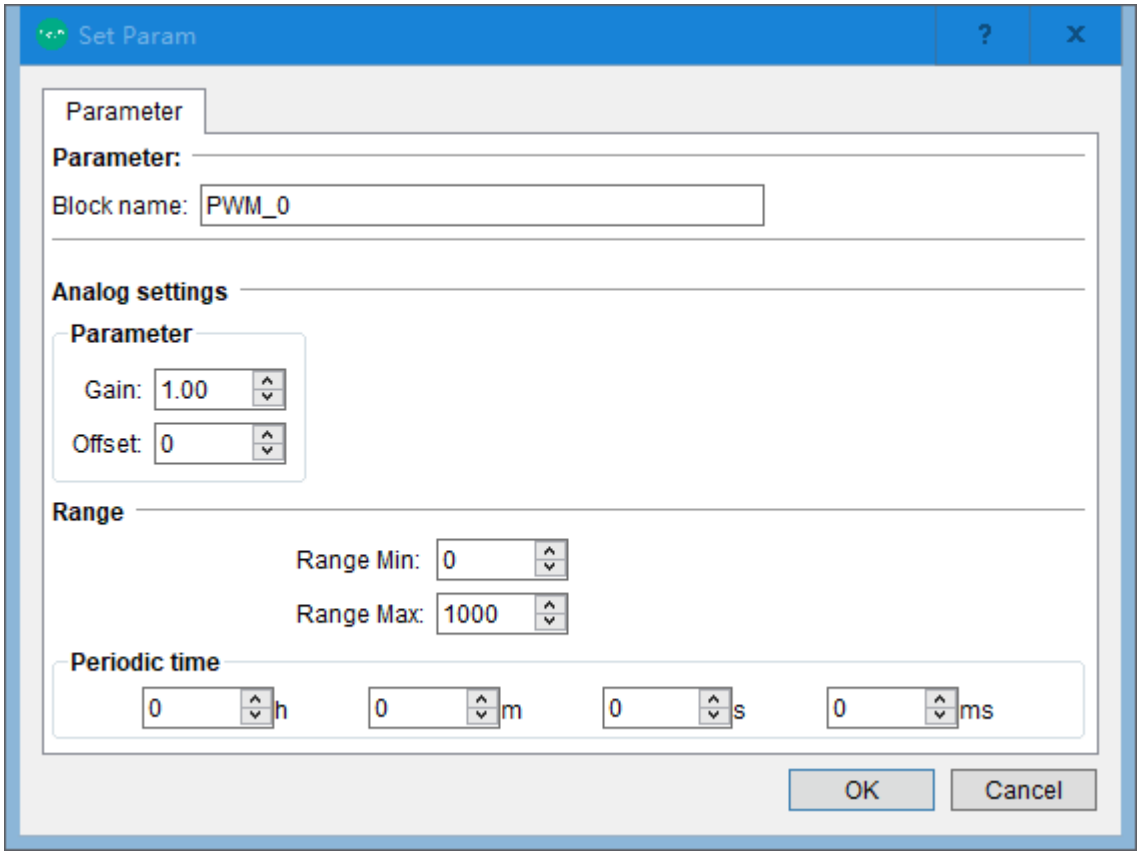
Describe: The Pulse Width Modulator (PWM) instruction modulates the analog input value Ax to a pulsed digital output signal. The pulse width is proportional to the analog value Ax.



Name	IN/OUT	Data Type	Describe
xEn	IN	BOOL	A positive edge (0 to 1 transition) at input En enables the PWM function block.
Ax	IN	INT	
PeriodicTime	IN	TIME	Periodic time over which the digital output is modulated
OUT_Q	OUT	BOOL	Q is set or reset for the proportion of each time period according to the proportion of the standardized value Ax to the analog value range.

Set Param

Double click the command to set the parameters



Gain

Range of values: -10.00 to 10.00

Offset

Range of values: -10,000 to 10,000

PT: Periodic time over which the digital output is modulated

PT: Periodic time over which the digital output is modulated

Description of the function

The function reads the value of the signal at the analog input Ax.

This value is multiplied by the value of parameter A (gain). Parameter B (offset) is added to the product, as follows:

(Ax * Gain) + Offset = Actual value Ax

The function block calculates the proportion of the value Ax to the range. The block sets the digital output Q high for the same proportion of the PT (periodic time) parameter, and sets Q low for the remainder of the time period.

Examples with Timing Diagrams

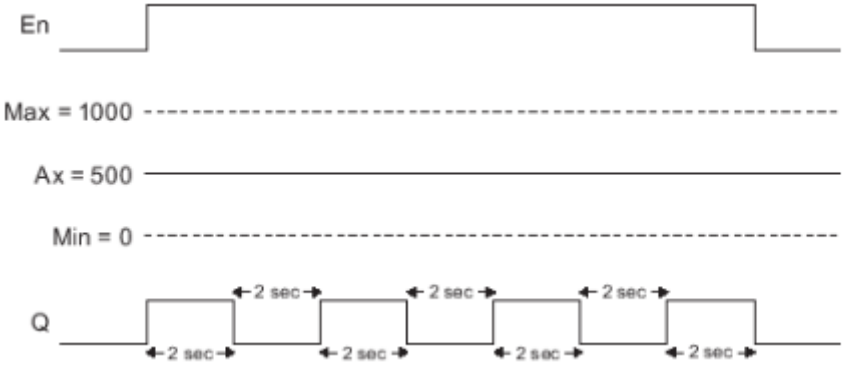
The following examples show how the PWM instruction modulates a digital output signal from the analog input value:

Example 1

Analog input value: 500 (range 0 to 1000)

Periodic time T: 4 seconds

The digital output of the PWM function is 2 seconds high, 2 seconds low, 2 seconds high, 2 seconds low and continues in that pattern as long as parameter "En" = high.

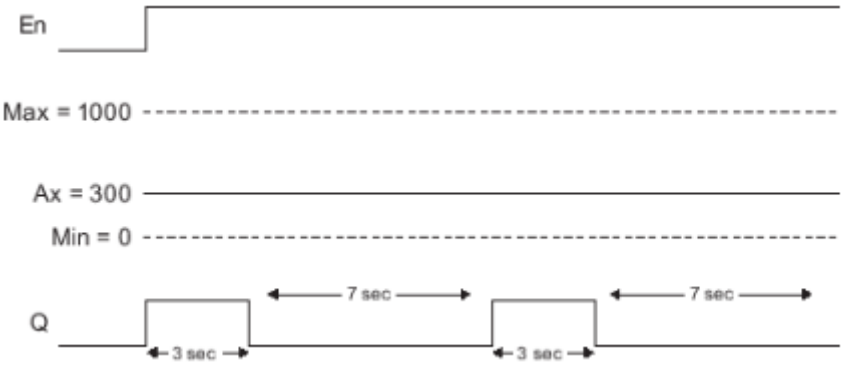


Example 2

Analog input value: 300 (range 0 to 1000)

Periodic time T: 10 seconds

The digital output of the PWM function is 3 seconds high, 7 seconds low, 3 seconds high, 7 seconds low and continues in that pattern as long as parameter "En" = high.



Calculation rule

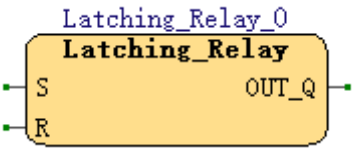
Q = 1, for (Ax – Min) / (Max – Min) of time period PT

Q = 0, for PT – [(Ax – Min) / (Max – Min)] of time period PT.

Note: Ax in this calculation refers to the actual value Ax as calculated using the Gain and Offset. Min and Max refer to the minimum and maximum values specified for the range.

5.15.16 Latching_Relay

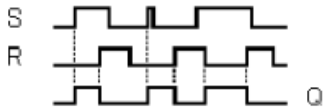
Describe: A signal at input S sets output Q. A signal at input R resets output Q.



Name	IN/OUT	Data Type	Describe
S	IN	BOOL	Set output Q with a signal at input S (Set).
R	IN	BOOL	Reset output Q with a signal at input R (Reset). Output Q is reset if S and R are both set (reset has priority over set).
OUT_Q	OUT	BOOL	Q is set with a signal at input S and remains set until it is reset with signal at input

			R.

Timing diagram



Description of the function

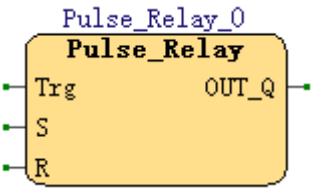
The latching relay represents a simple binary memory logic. The output value depends on the input states and the previous status at the output.

Logic table of the latching relay:

S	R	Q	Remark
0	0	X	Status unchanged
0	1	0	Reset
1	0	1	Set
1	1	0	Reset

5.15.17 Pulse_Relay

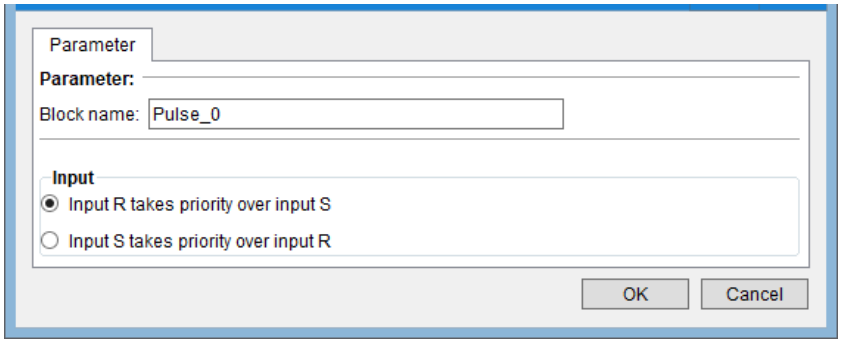
Describe: The output is set and reset with a short one-shot at the input.



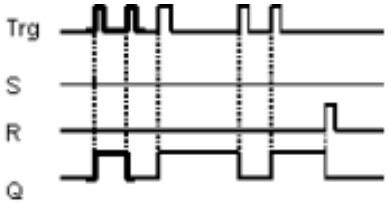
Name	IN/OUT	Data Type	Describe
Trg	IN	BOOL	You switch output Q on or off with a signal at input Trg (Trigger) input.
S	IN	BOOL	A one-shot at input S (Set) sets the output to logical 1.
R	IN	BOOL	A one-shot at input R (Reset) resets the output to logical 0.
OUT_Q	OUT	BOOL	Q is switched on with a signal at Trg and is reset again at the next Trg pulse, if both S and R = 0.

Set Param

Double-click the command to set the parameters



Timing diagram



Description of the function

The status of output Q changes with each 0 to 1 transition at input Trg and if both S and R = 0, that is, the output is switched on or off.

Input Trg does not influence the SFB when S = 1 or R = 1.

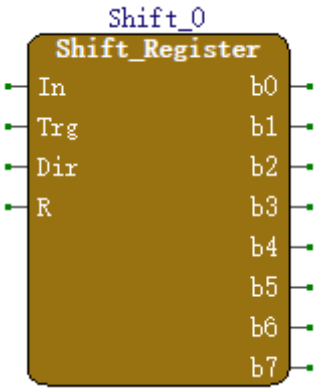
A one-shot at input S sets the pulse relay, that is, the output is set to logical 1.

A one-shot at input R resets the pulse relay to its initial state, that is, the output is set to logical 0.

Either the input R takes priority over input S (the signal at input S has no effect as long as R = 1), or the input S takes priority over input R (the signal at input R has no effect as long as S = 1), depending on your configuration.

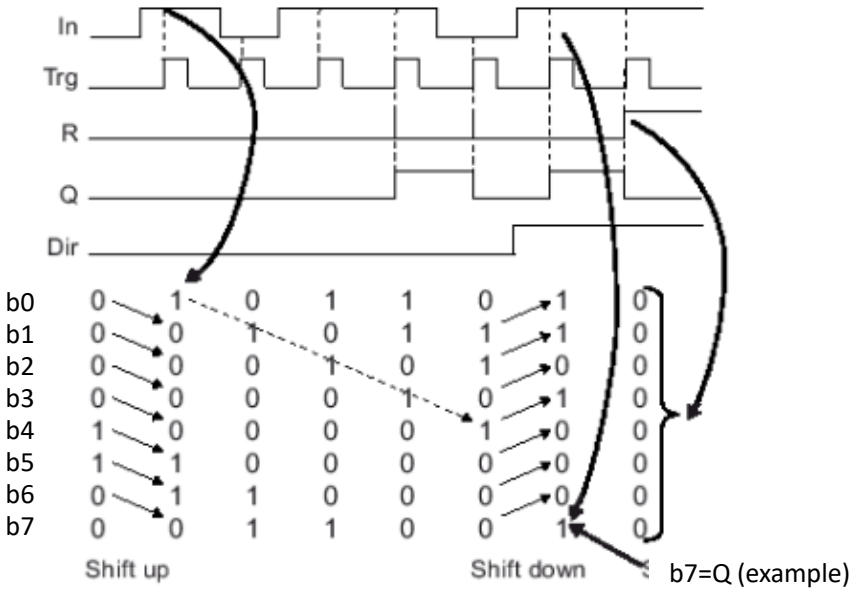
5.15.18 Shift_Register

Describe: The shift register function reads an input value and shifts the bits. The outputvalue corresponds with the configured shift register bit The shift direction can be changed at a special input.



Name	IN/OUT	Data Type	Describe
In	IN	BOOL	The function when started reads this input value.
Trg	IN	BOOL	The FBD is started with a positive edge (0 to 1 transition) at input Trg (Trigger).A 1 to 0 transitionis irrelevant.
Dir	IN	BOOL	You define the shift direction of the shift register bits b0...b7 at the Dir input: Dir=0: shift up: Dir=shift down.
R	IN	BOOL	The Function Block (FB) resets when input R (Reset is 1.When the FB is reset, all the shift register bit set to 0.
b0	OUT	BOOL	Bit 0
b1	OUT	BOOL	Bit 1
…	…	…	…
b7	OUT	BOOL	Bit 7

Timing diagram



Description of the function

This value is written to shift register bits b0 to b7, depending on the set shift direction:

Dir = 0 (Shift up): b0 accepts the value of input In, the previous value of b0 is shifted to b1, b1 to b2 ... b6 to b7

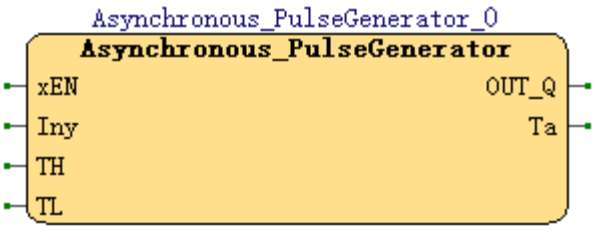
Dir = 1 (Shift down): b7 accepts the value of input In; the previous value of b7 is shifted to b6, b6 to b7 ... b1 to b0.

A positive edge (0 to 1 transition) at input R (Reset) reset the shift register. All the shift register bit (b0 to b7) and the output at Q are set to 0.

Q outputs the value of the configured shift register bits.

5.15.19 Asynchronous_PulseGenerator

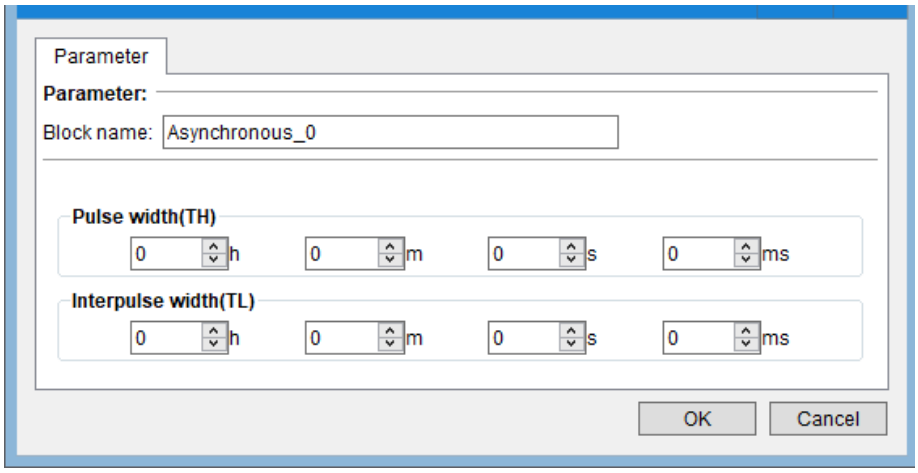
Describe: The pulse shape at the output can be modified by a configurable pulse/pause ratio.



Name	IN/OUT	Data Type	Describe
xEN	IN	BOOL	You enable/disable the asynchronous pulse generator with the signal at input En.
Iny	IN	BOOL	The Iny input can be used to invert the output signal of the active asynchronous pulse generator.
TH	IN	TIME	The pulse width.
TL	IN	TIME	The interpulse width.
OUT_Q	OUT	BOOL	Q is toggled on and off cyclically with the pulse/pause times TH and TL.
Ta	OUT	TIME	

Set Param

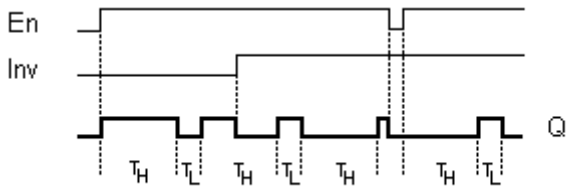
Double-click the command to set the parameters



You can set the pulse/pause ratio at the TH (Time High) and TL (Time Low) parameters.

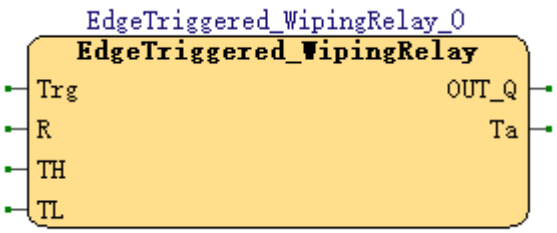
The INV input can be used to invert the output signal. The input block INV only inverts the output signal if the block is enabled with EN.

Timing diagram



5.15.20 EdgeTriggered_WipingRelay

Describe: An input pulse generates a preset number of output pulses with a defined pulse/pause ratio (retriggerable), after a configured delay time has expired.

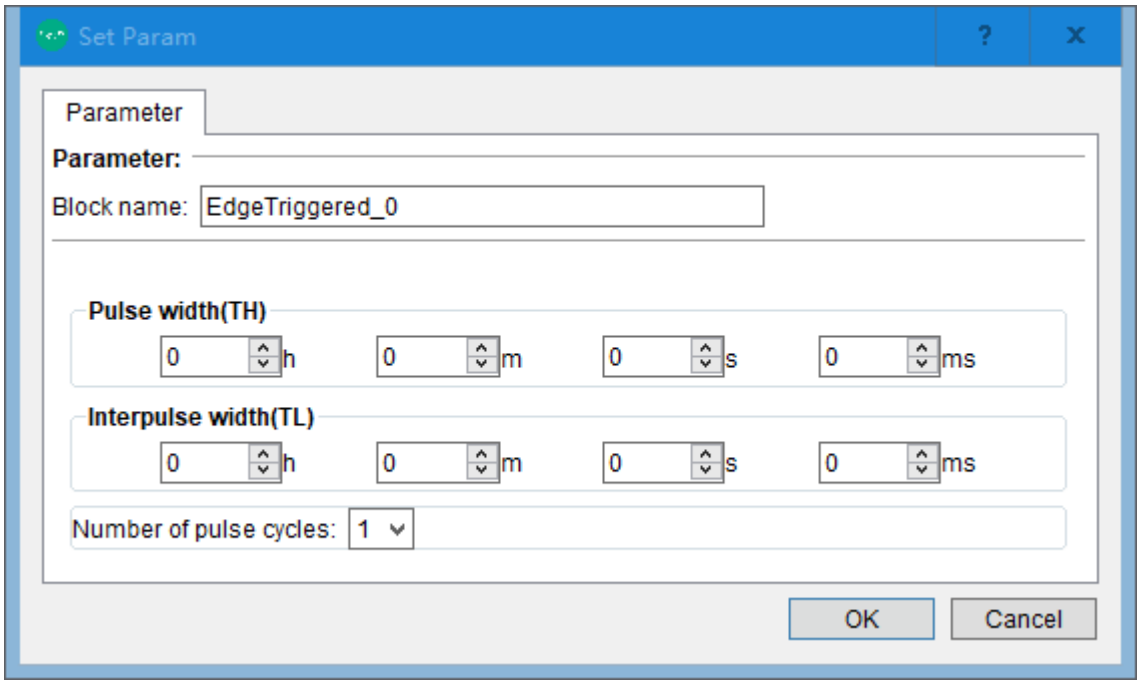


Name	IN/OUT	Data Type	Describe
Trg	IN	BOOL	You trigger the times for the Edge-triggered wiping relay with a signal at input Trg (Trigger).
R	IN	BOOL	The output and the current time Ta resets to 0 with a signal at input R.
TH	IN	TIME	The pulse width.
TL	IN	TIME	The interpulse width.
OUT_Q	OUT	BOOL	Output Q sets when the time TL has expired and resets when TH has expired.

Ta	OUT	TIME	

Set Param

Double-click the command to set the parameters

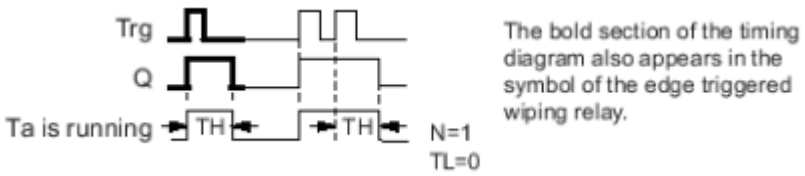


Description of the function

The change at input Trg to 1 triggers the time TL (time low). After the time TL has expired, SystemePLC sets the output Q to 1 for the duration of the time TH (time high).

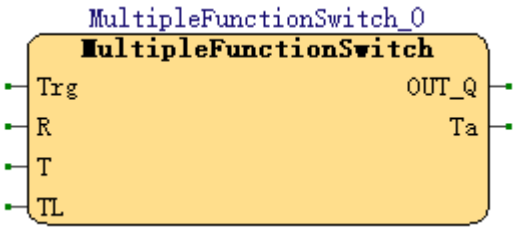
If SystemePLC retriggers the input Trg prior to the expiration of the preset time (TL + TH), the time Ta resets and the pulse/pause period is restarted.

Timing diagram



5.15.21 MultipleFunctionSwitch

Describe: Switch with two different functions: Pulse switch with off delay and Switch (continuous light)

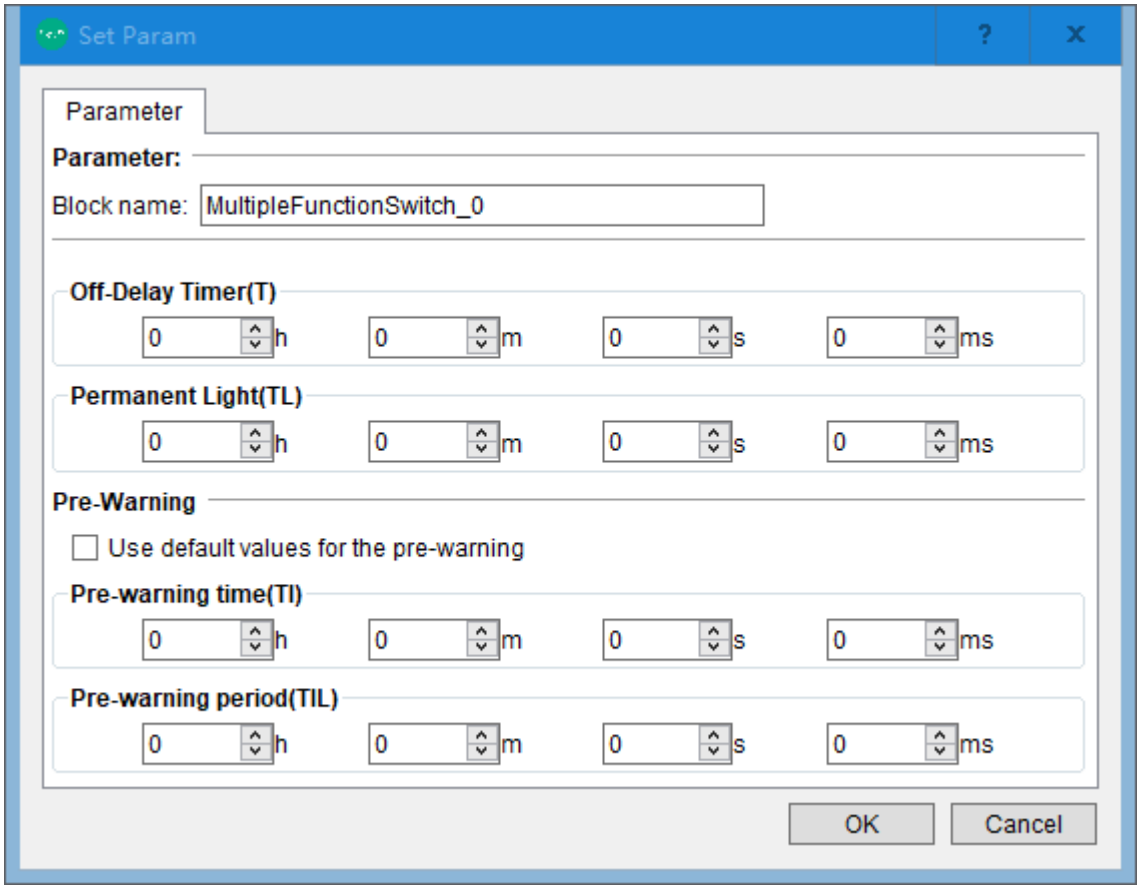


Name	IN/OUT	Data Type	Describe

Trg	IN	BOOL	A signal at input Trg (Trigger) sets output Q (permanent light) or resets Q with an off-delay. When active, output Q can be reset with a signal at input Trg.
R	IN	BOOL	A signal at input R resets the current time Ta and resets the output.
T	IN	TIME	Determines the off-delay time. The output is reset (1 to 0 transition) when the time T expires.
TL	IN	TIME	Determines the period during which the input must be set in order to enable the permanent light function.
OUT_Q	OUT	BOOL	Output Q is set with a signal at input Trg, and it is reset again after a configured time has expired and depending on the pulse width at input Trg, or it is reset with another signal at input Trg.
Ta	OUT	TIME	

Set Param

Double click the command to set the parameters



T: determines the off-delay time. The output is reset (1 to 0 transition) when the time T expires.

TL: determines the period during which the input must be set in order to enable the permanent light function.

TI: determines the on delay for the prewarning time.

TIL: determines the length of the prewarning time period.

Retentivity on = the status is retentive in memory.

Description of the function

Output Q is set to 1 with a 0 to 1 signal transition at Trg.

If output Q = 0, and input Trg is set hi for at least the duration of TL, the permanent lighting function is enabled and

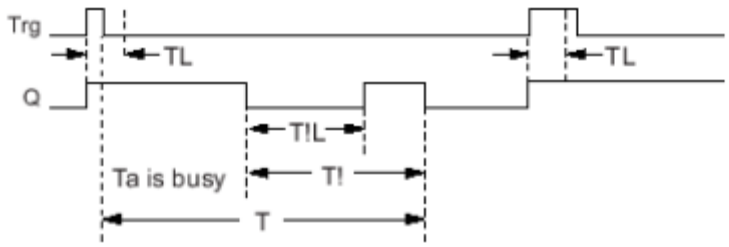
output Q is set accordingly.

The off-delay time T is triggered when the status at input Trg changes to 0 before the time TL has expired.

Output Q is reset when the Ta = T.

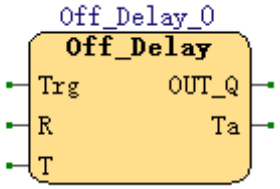
You can output an off-warning signal prior to the expiration of the off-delay time (T - TI) that resets Q for the duration of the off prewarning time TIL. A subsequent signal at input Trg always resets T and output Q.

Timing diagram



5.15.22 Off_Delay

Describe: The output with off-delay resets after a defined time has expired



Name	IN/OUT	Data Type	Describe
Trg	IN	BOOL	Start the off-delay time with a negative edge (1 to 0 transition) at input Trg (Trigger).
R	IN	BOOL	Reset the off-delay time and set the output to 0 via the R (Reset) input.Reset has priority over Trg.
T	IN	TIME	The output is switched off on expiration of the delay time T (output signal transition 1 to 0).
OUT_Q	OUT	BOOL	Q switches on for the duration of the time T after a trigger at input Trg.
Ta	OUT	TIME	

Description of the function

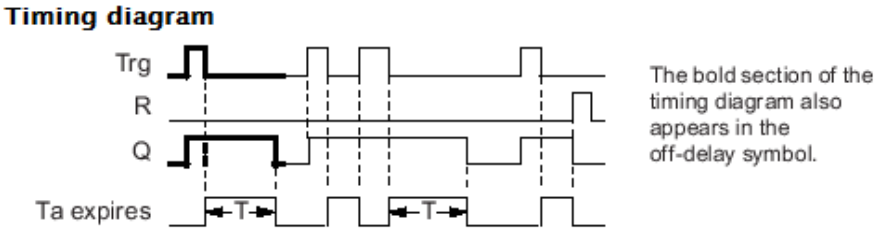
A 0 to 1 transition on input Trg sets output Q to 1 momentarily.

When a 1 to 0 transition occurs at input Trg, the current time T is retriggered and the output remains set. When Ta reaches the value specified by T (Ta=T), the output Q is reset to 0 (off delay).

A single trigger of input Trg retriggers the time Ta.

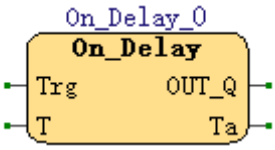
Time Ta and the outputs can be reset before the end of the Ta time with input R (Reset).

Timing diagram



5.15.23 On_Delay

Describe: The output does not switch on until a configured delay time has expired.



Name	IN/OUT	Data Type	Describe
Trg	IN	BOOL	The Trg (Trigger) input triggers the on-delay time.
T	IN	TIME	Represents the on-delay time after which the output is switched on (output signal transition 0 to 1).
OUT_Q	OUT	BOOL	Q switches on after a specified time T has expired, provided Trg is still set.
Ta	OUT	TIME	

Description of the function

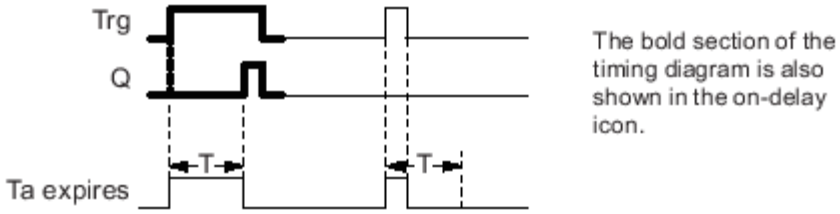
The 0 to 1 transition at input Trg triggers the time Ta.

If the status at input Trg stays 1 at least for the duration of the configured time T, the output is set to 1 when this time has expired (the on signal of the output follows the on signal of the input with delay).

The time is reset if the status at input Trg changes to 0 again before the time T has expired.

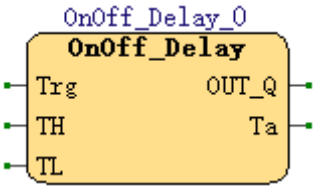
The output is reset to 0 when input Trg is 0.

Timing diagram



5.15.24 Onoff_Delay

Describe: The on-/off- delay function block can set an output after a configured on-delay time and then reset it again upon expiration of a second configured time.



Name	IN/OUT	Data Type	Describe
Trg	IN	BOOL	You trigger the on-delay with a positive edge (0 to 1 transition) at input Trg (Trigger).You trigger the off-delay with a negative edge (1 to 0 transition).
TH	IN	TIME	TH is the on-delay time for the output (output signal transition 0 to 1).
TL	IN	TIME	TL is the off-delay time for the output (output signal transition 1 to 0).
OUT_Q	OUT	BOOL	Q switches on upon expiration of a configured time TH if Trg is still set. It switches off again upon expiration of the time TL and if Trg has not been set again.
Ta	OUT	TIME	

Description of the function

A 0 to 1 transition of the input Trg triggers the time TH.

If the state of the input Trg is 1 for at least the duration of the configured time TH, the output is set to a logic 1 at the end of this time (the output is delayed on with the input signal).

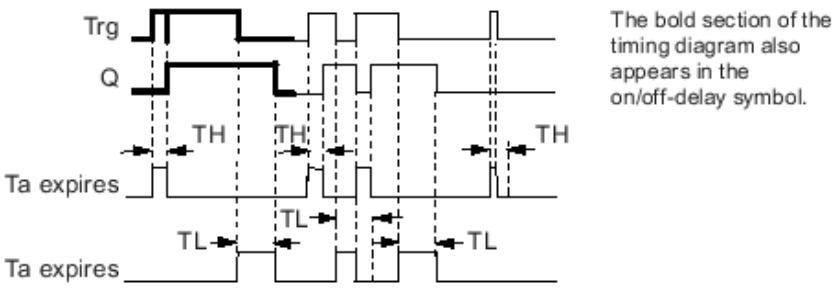
If the state of input Trg is reset to 0 before the end of this time, time TH is reset.

The transition from 1 to 0 at the output triggers time TL.

If the state of input Trg remains 0 for at least the configured duration of time TL, the output is reset to 0 at the end of this time (output off delayed to input signal).

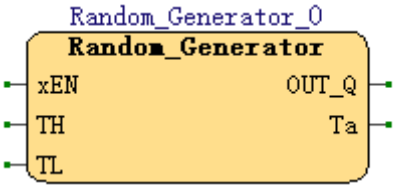
If the state of input Trg returns to 1 before the end of time TL, time TL is reset.

Timing diagram



5.15.25 Random_Generator

Describe: The output of a random generator is toggled within a configurable time



Name	IN/OUT	Data Type	Describe
xEN	IN	BOOL	The positive edge (0 to 1 transition) at the enable input En (Enable) triggers the on-delay for the random generator. The negative edge (1 to 0 transition) triggers the off-delay for the random generator.
TH	IN	TIME	The on-delay is determined at random and lies between 0 s and TH.
TL	IN	TIME	The off-delay is determined at random and lies between 0 s and TL.
OUT_Q	OUT	BOOL	Q is set on expiration of the on-delay if En is still set. It is reset when the off-delay time has expired and if En has not been set again.
Ta	OUT	TIME	

Description of the function

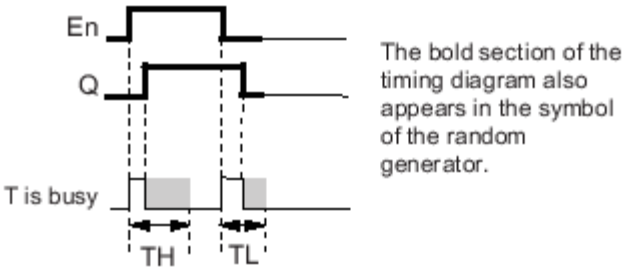
With the 0 to 1 transition at input En, a random time (on-delay time) between 0 s and TH is set and triggered. If the status at input En is 1 for at least the duration of the on-delay, the output is set to 1 when this on-delay time has expired.

The time is reset if the status at input En is reset to 0 before the on-delay time has expired.

When input En is reset 0, a random time (off-delay time) between 0 s and TL is set and triggered.

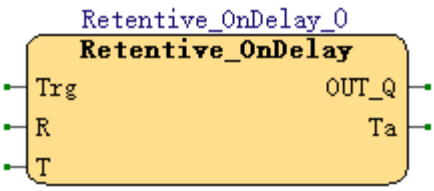
If the status at input En is 0 at least for the duration of the off-delay time, the output Q is reset to 0 when the off-delay time has expired.

Timing diagram



5.15.26 Retentive_OnDelay

Describe: A one-shot at the input triggers a configurable time. Set the output upon expiration of this time.



Name	IN/OUT	Data Type	Describe
Trg	IN	BOOL	Trigger the on-delay time via the Trg (Trigger) input.
R	IN	BOOL	Reset the on-delay time and reset the output to 0 via input R (Reset).Reset takes priority over Trg.
T	IN	TIME	T is the on-delay time for the output (output signal transition 0 to 1).

OUT_Q	OUT	BOOL	Q switches on upon expiration of the time T.
Ta	OUT	TIME	

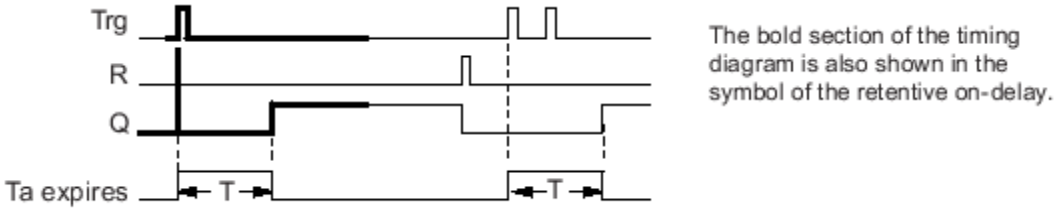
Description of the function

The 0 to 1 signal transition at input Trg triggers the current time Ta. When Ta reaches time T, output Q is set to 1.

The outputs and the time Ta are only reset to 0 if the signal on input R is 1.

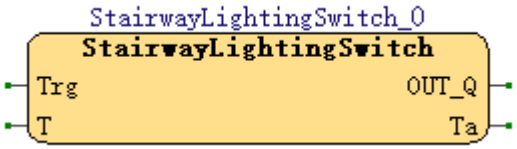
If no holdover is set, the output Q and the expiry time are reset after a power failure.

Timing diagram



5.15.27 StairwayLightingSwitch

Describe: The edge of an input pulse triggers a configurable time. The output is reset when this time has expired. An off warning can be output prior to the expiration of this time.



Name	IN/OUT	Data Type	Describe
Trg	IN	BOOL	You trigger the time (off-delay) for the stairway switch with a signal at input Trg (Trigger).
T	IN	BOOL	The output is reset (1 to 0 transition) when the off-delay time T has expired.
OUT_Q	IN	BOOL	Q is reset after the time T has expired. A warning signal can be output before this time has expired.
Ta	OUT	TIME	

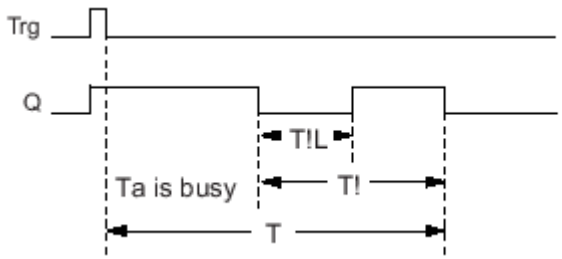
Description of the function

Output Q is set to 1 with a 0 to 1 signal transition at input Trg. The 1 to 0 transition at input Trg triggers the current time and output Q remains set.

Output Q is reset to 0 when Ta reaches the time T. Before the off delay time (T - T!) has expired, you can output a prewarning that resets Q for the duration of the off prewarning time T!L.

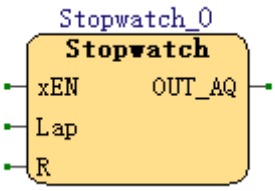
Ta is retriggered (optional) at the next high/low transition at input Trg and if Ta is expiring.

Timing diagram



5.15.28 Stopwatch

Describe: The stopwatch records the time elapsed since it was enabled



Name	IN/OUT	Data Type	Describe
xEN	IN	BOOL	En (Enable) is the monitoring input. Set the current elapsed time to 0 and begins counting elapsed time when En transitions from 0 to 1. When En transitions from 1 to 0, the elapsed time is frozen.
Lap	IN	BOOL	A positive edge (0 to 1 transition) at input Lap pauses the stopwatch, and sets output to lap time. A negative edge (1 to 0 transition) at input Lap resumes the stopwatch, and set the output to current elapsed time.
R	IN	BOOL	A signal at input R (Reset) clears the current elapsed time and lap time.
OUT_AQ	OUT	TIME	The output AQ outputs value of the current elapsed time when it is a negative edge (1 to 0 transition) at the input Lap, and outputs value of the Lap time when it is a positive edge (0 to 1 transition) at the input Lap. A positive edge (0 to 1 transition) resets the value at output AQ to 0.

When En = 1, the current time increases.

When En = 0, the current time counting pauses.

When En = 1 and Lap = 0, the output AQ outputs the value of the current elapsed time.

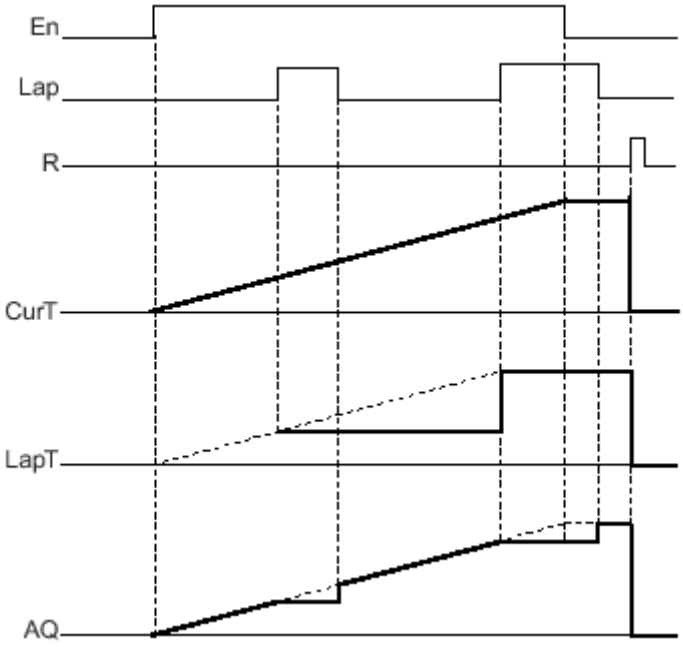
When En = 1 and Lap = 1, the current time continue increasing, but the output AQ outputs the value of the Lap time.

When En = 0 and Lap =1, the output AQ outputs the value of the Lap time.

When En = 0 and Lap = 0, the output AQ outputs the value of the latest current time.

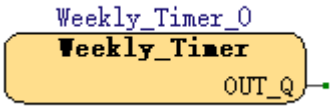
When R = 1, both the current time and the Lap time are reset.

Timing diagram



5.15.29 Weekly_Timer

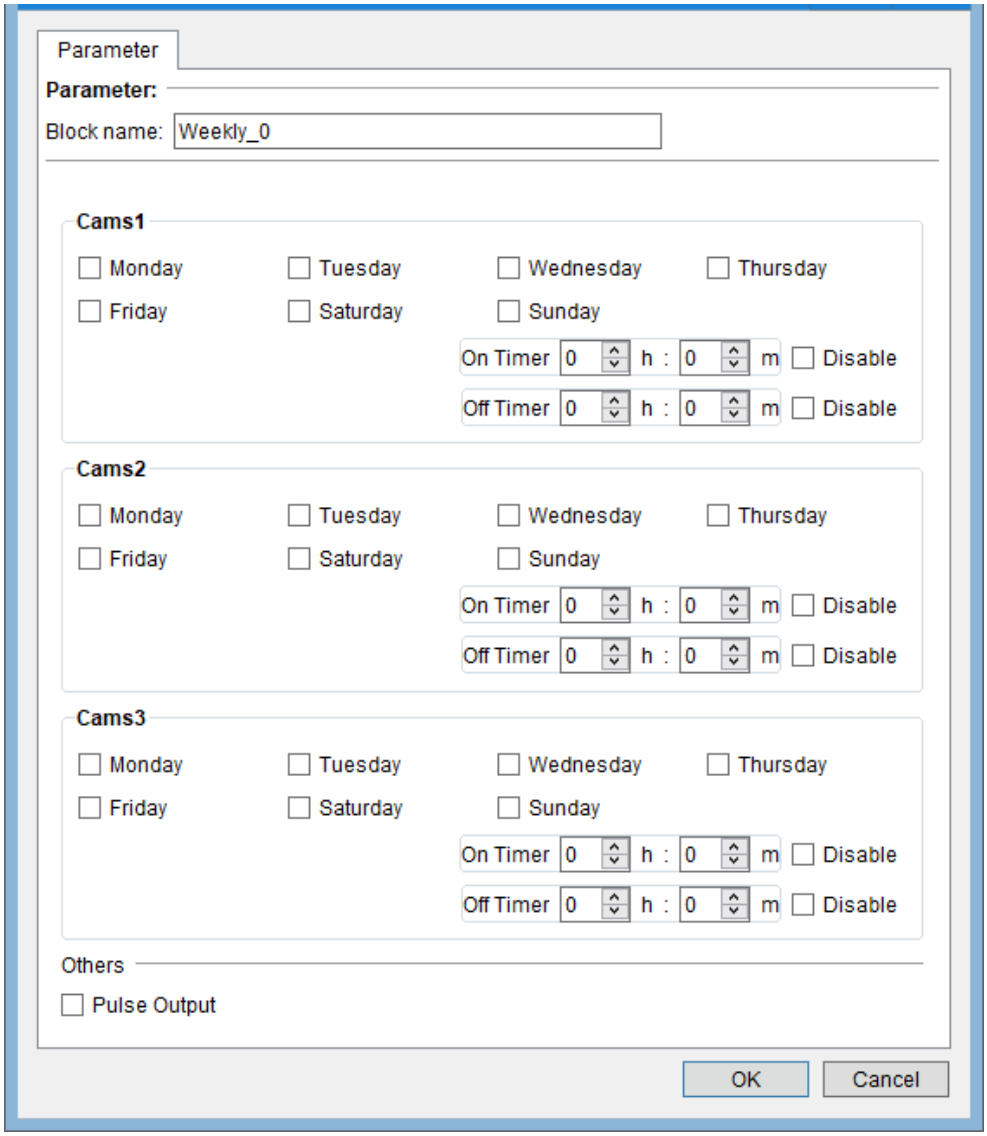
Describe: The output is controlled by means of a configurable on/off date. The function supports any combination of weekdays.



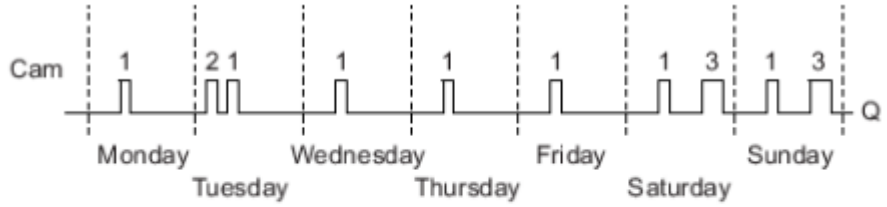
Name	IN/OUT	Data Type	Describe
OUT_Q	OUT	BOOL	Q is set when the configured cam is actuated.

Set Param

Double click the command to set the parameters



Timing diagram (three practical examples)



Cam 1:	Daily:	06:30 h to 8:00 h
Cam 2:	Tuesday:	03:10 h to 04:15 h
Cam 3:	Saturday and Sunday:	16:30 h to 23:10 h

Description of the function

Each weekly timer is equipped with three cams. You can configure a time hysteresis for each individual cam. At the cams you set the on- and off-hysteresis. The weekly timer sets the output at a certain time, provided it is not already set.

The weekly timer resets the output at the off-time if you configured an off-time, or at the end of the cycle if you specified a pulse output. A conflict is generated in the weekly timer when the on-time and the off-time at another cam are identical. In this case, cam 3 takes priority over cam 2, while cam 2 takes priority over cam 1.

The switching status of the weekly timer is determined by the status at the No1, No2 and No3 cams.

On-times

The on-time is any time between 00:00 h and 23:59 h. You can also configure the on-time to be a pulse signal. The timer block will be activated at the specified time for one cycle and then the output is reset. The off-time is disabled in this case as it is not applicable.

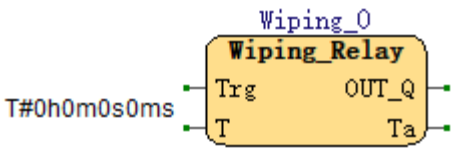
Special characteristics to note when configuring

The block properties window offers a tab for each one of the three cams. Here you can set the day of the week for each cam. Each tab offers you in addition an option of defining the on- and off-times for each cam in hour and minute units. Hence, the shortest switching cycle is one minute. Also on each tab you have the option of specifying a pulse output for the cam.

You can disable the on- and off-times individually. You can achieve switching cycles extending across more than one day, for example, by setting the on-time for cam 1 to Monday 7:00 h and the off-time of cam 2 to Wednesday 13:07 h, while disabling the on time for cam 2.

5.15.30 Wiping_Relay

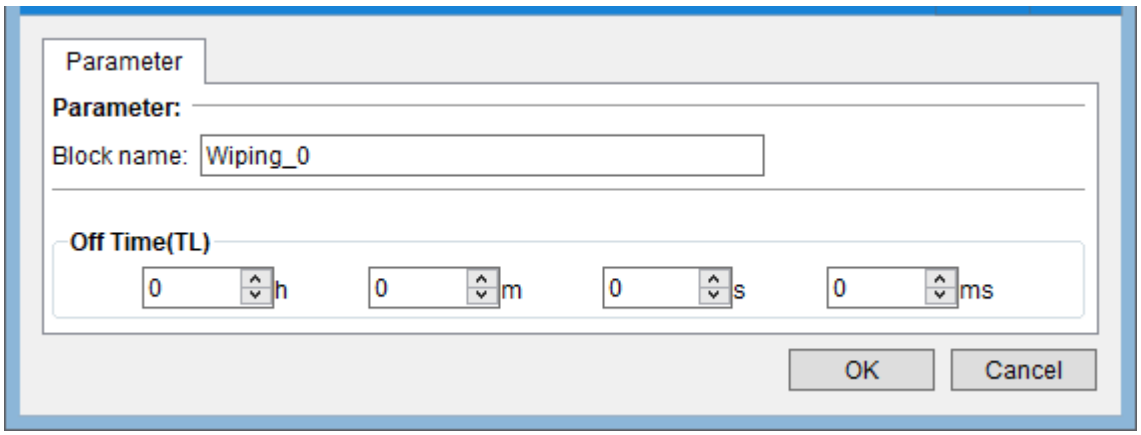
Describe: An input signal generates an output signal of a configurable length.



Name	IN/OUT	Data Type	Describe
Trg	IN	BOOL	You trigger the time for the wiping relay with a signal at input Trg (Trigger).
T	IN	TIME	T represents the time after which the output resets (output signal transition 1 to 0).
OUT_AQ	OUT	BOOL	A pulse at Trg sets Q. The output stays set until the time T has expired and if Trg = 1 for the duration of this time. A 1 to 0 transition at Trg prior to the expiration of T also resets the output to 0.
Ta	OUT	TI ME	

Set Param

Double click the command to set the parameters



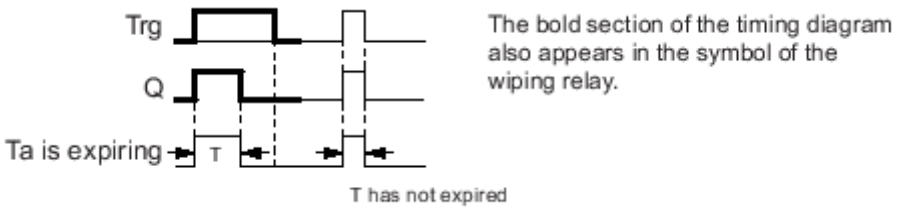
Description of the function

The input signal Trg = 1 sets the output Q to 1. The signal also triggers the time Ta, while the output remains set.

When Ta reaches the value defined at T (Ta=T), the output Q resets to 0 state (pulse output).

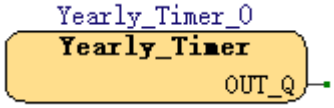
If the signal at input Trg changes from 1 to 0 before this time has expired, the output immediately resets from 1 to 0.

Timing diagram



5.15.31 Yearly_Timer

Describe: The output is controlled by means of a configurable on/off date. You can configure the timer to activate on a yearly, monthly, or user-defined time basis. With any mode, you can also configure the timer to pulse the output during the defined time period.

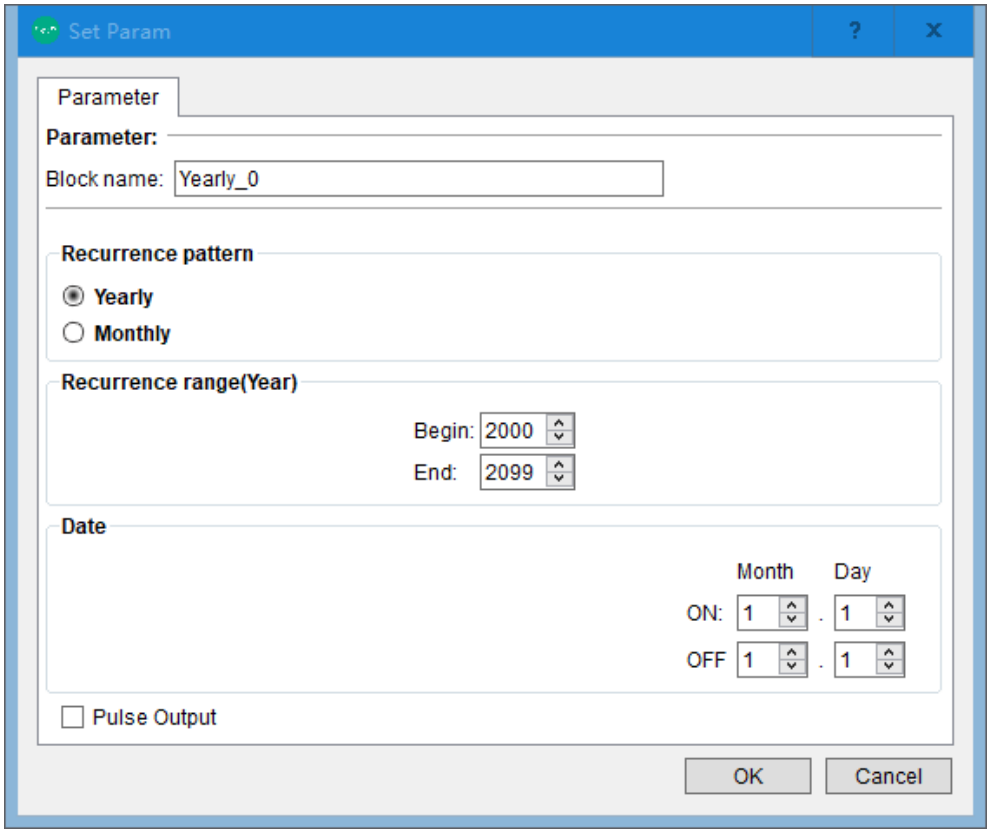


Name	IN/OUT	Data Type	Describe
OUT_Q	OUT	BOOL	Q is set on when the configured cam is switched on.

Set Param

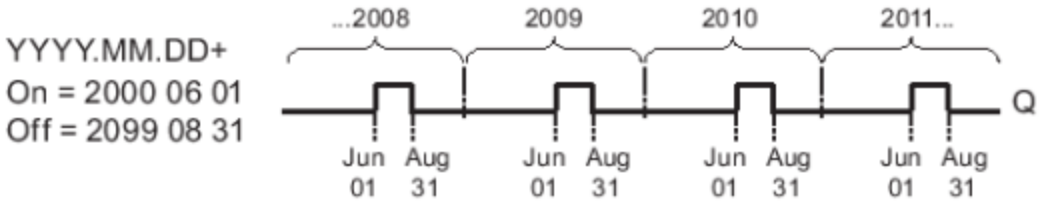
Double-click the command to set the parameters

At the **No** (cam) parameter, you configure the timer mode, the on-/off-times for the timer, and whether the output is a pulse output.

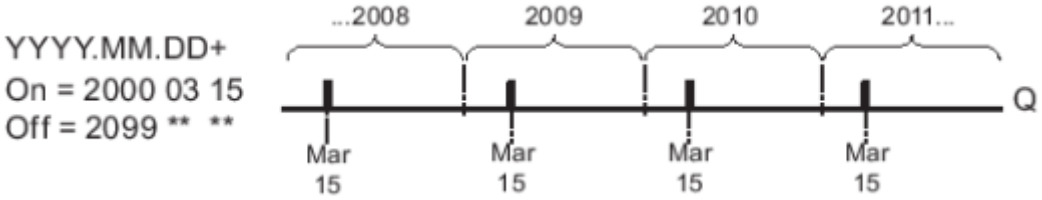


Timing diagram

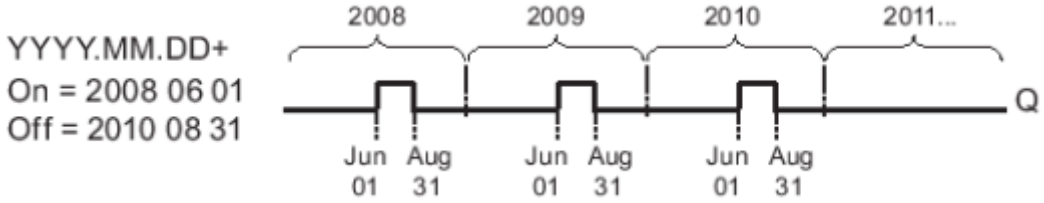
Example 1: Yearly selected, On Time = 2000.06.01, Off Time = 2099.08.31, Every year on June 1 the timer output switches on and remains on until August 31.



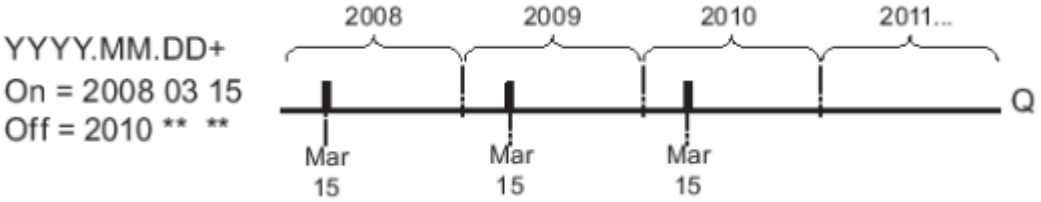
Example 2: Yearly selected, Pulse Output selected, On Time = 2000.03.15, Off Time = 2099.**.**. Every year on March 15, the timer switches on for one cycle.



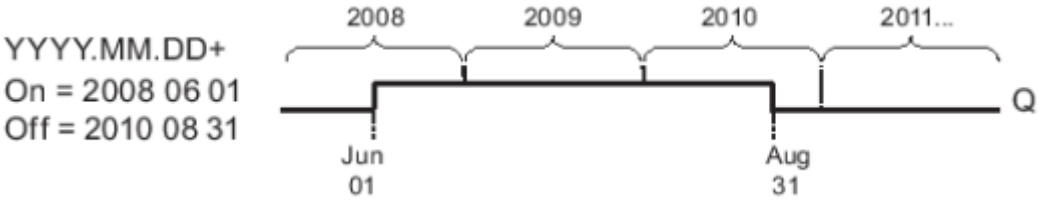
Example 3: Yearly selected, On Time = 2008.06.01, Off Time = 2010.08.31. On June 1 of 2008, 2009, and 2010, the timer output switches on and remains on until August 31.



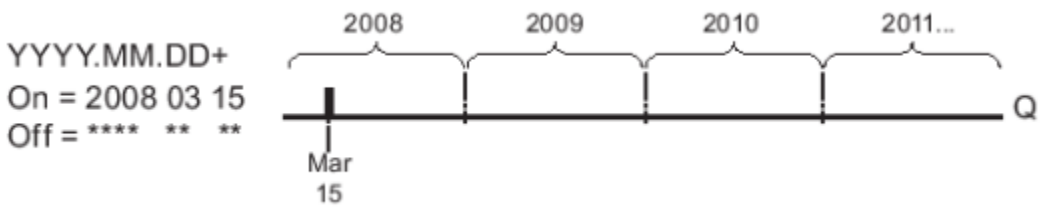
Example 4: Yearly selected, Pulse Output selected, On Time = 2008.03.15, Off Time = 2010.**.**. On March 15 of 2008, 2009, and 2010, the timer output switches on for one cycle.



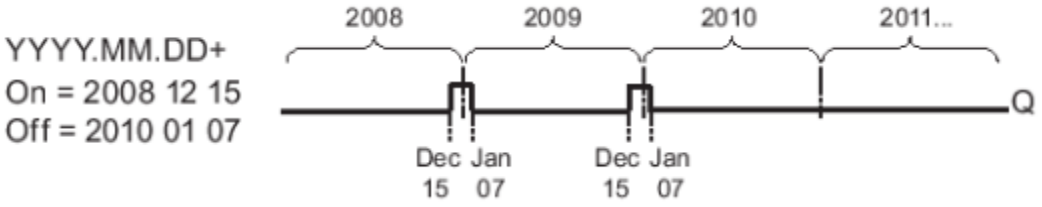
Example 5: Monthly not selected, Yearly not selected, On Time = 2008.06.01, Off Time = 2010.08.31. On June 1, 2008 the timer output switches on and remains on until August 31, 2010.



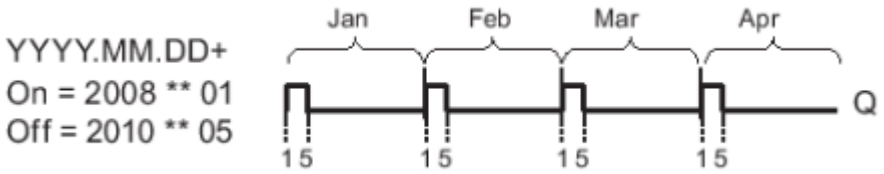
Example 6: Monthly not selected, Yearly not selected, Pulse Output selected, On Time = 2008.03.15, Off Time = ****.**.**. On March 15, 2008 the timer switches on for one cycle. Because the timer does not have a monthly action or yearly action, the timer output pulses only one time at the specified On Time.



Example 7: Yearly selected, On Time = 2008.12.15, Off Time =2010. 01.07. On December 15 of 2008 and 2009, the timer output switches on and remains on until January 7 of the following year. When the timer output turns off on January 7, 2010 it does NOT turn on again the following December 15.



Example 8: Monthly selected, On Time = 2008.**.01, Off Time = 2010.**.05. Starting in 2008, on the first day of each month the timer output switches on and switches off on the fifth day of the month. The timer continues in this pattern through the last month of 2010.



Description of the function

The yearly timer sets and resets the output at specific on and off dates. Sets and resets are executed at 00:00. If your application requires a different time, use a weekly timer together with a yearly timer in your circuit program.

The On Time specifies the month and day when the timer is set. The Off Time identifies the month and day on which the output is reset again. For the on and off times, note the order of the fields: The first field defines the year, the second the month and the third the day.

When you select the Monthly check box, the timer output switches on each month at the specified day of the start time and remains on until the specified day of the Off Time. The On Year specifies the initial year in which the timer is activated. The Off Year defines the last year in which the timer turns off. The maximum year is 2099.

If you select the Yearly check box, the timer output switches on each year at the specified month and day of the start time and remains on until the specified month and day of the Off Time. The On Year specifies the initial year in which the timer is activated. The Off Year defines the last year in which the timer turns off. The maximum year is 2099.

If you select the Pulse Output check box, the timer output switches on at the specified On Time for one cycle and then the timer output is reset. You can choose to pulse a timer on a monthly or yearly basis, or just a single time.

If you select none of the Monthly, Yearly, or Pulse check boxes, you can define a specific time period with the On Time and Off Time. It can span any time period that you choose.

For a process action that is to be switched on and off at multiple but irregular times during the year, you can define multiple yearly timers with the outputs connected by an OR function block.

Sixth section

LD program

6.1 Characteristics of Programming Elements

6.2 Creating LD Main Programs and Subprograms

6.3 Adding Contact Call Points, Adding Inverted Contacts, or Modifying Inverted Contacts

6.4 Branches

6.5 Adding Instructions

6.6 Quick Call of Points on Instructions

6.7 Deleting Contacts or Instructions

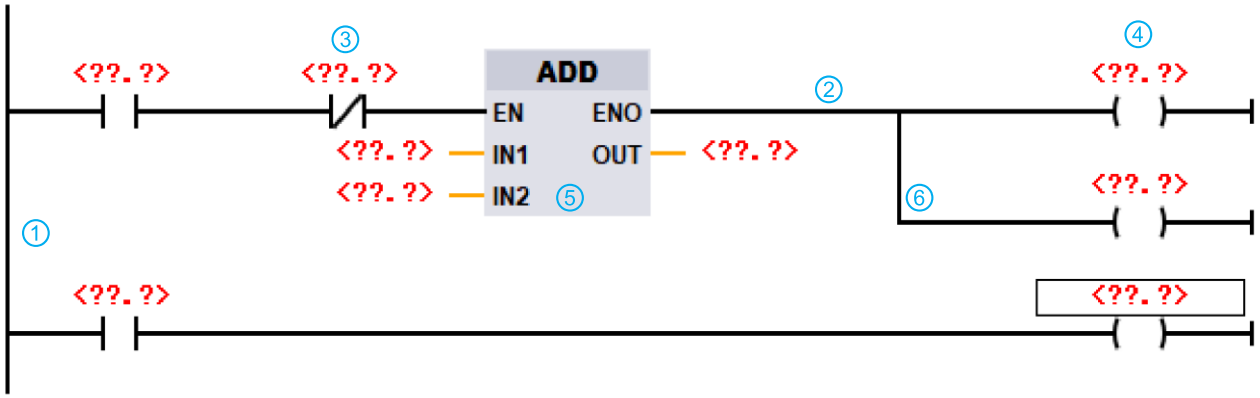
6.8 Crossover Points

6.9 Example of a Chasing Light Program

6.10 Display of LD Program Execution Status

Ladder Diagram (LD) is a graphical programming language. It adopts a circuit diagram-based notation, where a program is represented by one or more program segments. Each program segment includes a power rail on the left side of the rung's starting position, and binary signals are arranged as contacts on the rungs. The sequential arrangement of elements on a rung forms a series branch, while the arrangement on parallel branches forms a parallel branch. Complex functions are represented by function blocks.

6.1 Characteristics of Programming Elements



① Power Rail
② Rung
③ Contact
④ Coil
⑤ Function Block
⑥ Branch

Power Rail

Every LD program segment contains power rails with at least one rung. Program segments can be expanded by adding additional rungs. Branches can be used to create parallel structures within specific rungs.

Contact

Contacts are used to establish or interrupt current-carrying connections between two elements. Current flows from left to right. Contacts are employed to query the signal status or value of operands and control them based on the current result. The following types of contacts are available in LD programs:

Normally Open (NO) Contact: Transmits current if the signal status of the specified binary operand is "1".

Normally Closed (NC) Contact: Transmits current if the signal status of the specified binary operand is "0".

Coil

Coils are used to control binary operands. They can set or reset binary operands based on the signal status of logical operation results.

Function Block

Function blocks are LD elements that implement complex functions. In LD programs, function blocks with the EN/ENO mechanism can be used: A function block is executed only when the signal status of the enable input "EN" is "1". If the function block is processed correctly, the signal status of the enable output "ENO" is set to "1"; if an error occurs during processing, the enable output "ENO" is reset.

6.1.1 Contact Elements

-||-	Normally Open (NO) Contact
-|/|-	Normally Closed (NC) Contact
-|NOT|-	Invert RLO
-|N|-	Rising Edge
-|P|-	Falling Edge

6.1.2 Coil Elements

-()-	Output Coil: The coil turns on when conditions are met
-(/)-	Output Coil: The coil turns off when conditions are met
-(R)	Output Coil: Reset output
-(S)	Output Coil: Set output
-(P)-	Output Coil: Set operand on positive signal edge
-(N)-	Output Coil: Set operand on negative signal edge

6.1.3 Elements Introduction

--|P|--: Scan operand for positive signal edge

You can use the "Scan operand for positive signal edge" instruction to determine if there is a "0" to "1" change in the signal state of a specified variable (<Var1>). The instruction compares the current signal state of <Var1> with the signal state of the previous scan, which is saved in an edge memory bit (<Var2>). If the instruction detects a change in the result of logic operation (RLO) from "0" to "1", there is a positive, rising edge.

The following figure shows the change in the signal state in case of a negative and a positive signal edge:

This instruction checks for the rising edge of the signal upon every execution. If a rising edge is detected, the signal state of <Var1> will remain set to "1" for one program cycle. Under all other conditions, the state of this variable is "0".
Specify the variable to be monitored (<Variable 1>) in the variable placeholder above the instruction. Specify the edge memory bit (<Var2>) in the variable placeholder below the instruction.

Note
Modification of the address of the edge memory bit

The address of the edge memory bit must not be used more than once in the program, otherwise the bit memory is overwritten. This step influences the edge evaluation and the result is therefore no longer unique.

Parameter

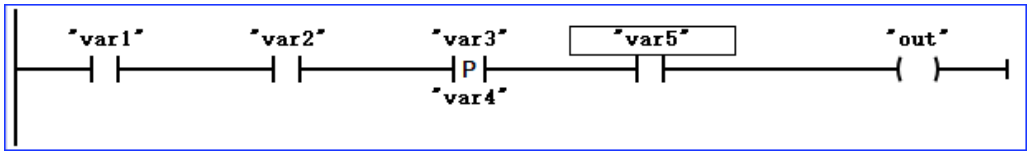
The following table shows the parameters of the "Scan operand for negative signal edge" instruction:

Parameter	Declaration	Data type	Description
< Var1>	Input	BOOL	Signal to be scanned

< Var2>	InOut	BOOL	Edge memory bit in which the signal state of the previous query is saved.

Example

The following example shows how the instruction works:



Variable "Out" is set when the following conditions are fulfilled:

•The "Var1" and "Var2"have signal state "1".

•There is a rising edge at "Var3". The signal state of the previous scan is stored in the edge memory bit "Var4".

•The signal state of the "Var5" is "1".

--|N|--：Scan operand for negative signal edge

Description

You can use the "Scan operand for negative signal edge" instruction to determine if there is a "1" to "0" change in the signal state of a specified Variable (<Var1>). The instruction compares the current signal state of <Var1> with the signal state of the previous scan that is saved in an edge memory bit <Var2>. If the instruction detects a change in the result of logic operation (RLO) from "1" to "0", there is a negative, falling edge.

The following figure shows the change in the signal state in case of a negative and a positive signal edge:

The negative signal edge is queried each time the instruction executes. When a negative signal edge is detected, <Var1> is set to signal state "1" for one program cycle. In all other cases, the operand has the signal state "0".

Specify the operand to be queried (<Var1>) in the operand placeholder above the instruction. Specify the edge memory bit (<Var2>) in the operand placeholder below the instruction.

Note

Modification of the address of the edge memory bit

The address of the edge memory bit must not be used more than once in the program, otherwise the bit memory is overwritten. This step influences the edge evaluation and the result is therefore no longer unique.

Parameter

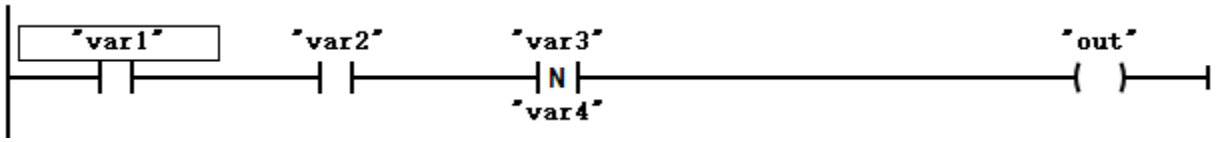
The following table shows the parameters of the "Scan operand for negative signal edge" instruction:

Parameter	Declaration	Data type	Description
< Var1>	Input	BOOL	Signal to be scanned
< Var2>	InOut	BOOL	Edge memory bit in which the signal state of the previous query is

			saved.

Example

The following example shows how the instruction works:



Variable "Out" is set when the following conditions are fulfilled:

•The "Var1" and "Var2"have signal state "1".

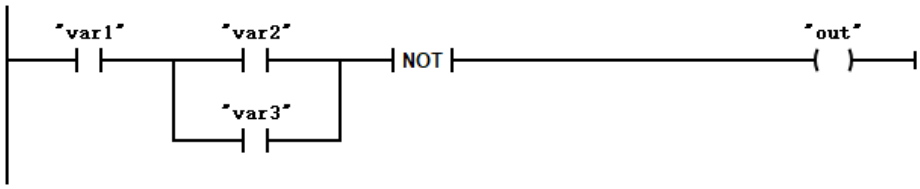
•There is a negative signal edge at "Var3". The signal state of the previous scan is stored in the edge memory bit "Var4".

--|NOT|--: Invert RLO

Description

You use the "Invert RLO" instruction to invert the signal state of the result of logic operation (RLO). If the signal state is "1" at the input of the instruction, the output of the instruction has signal state "0". If the signal state is "0" at the input of the instruction, the output has the signal state "1".

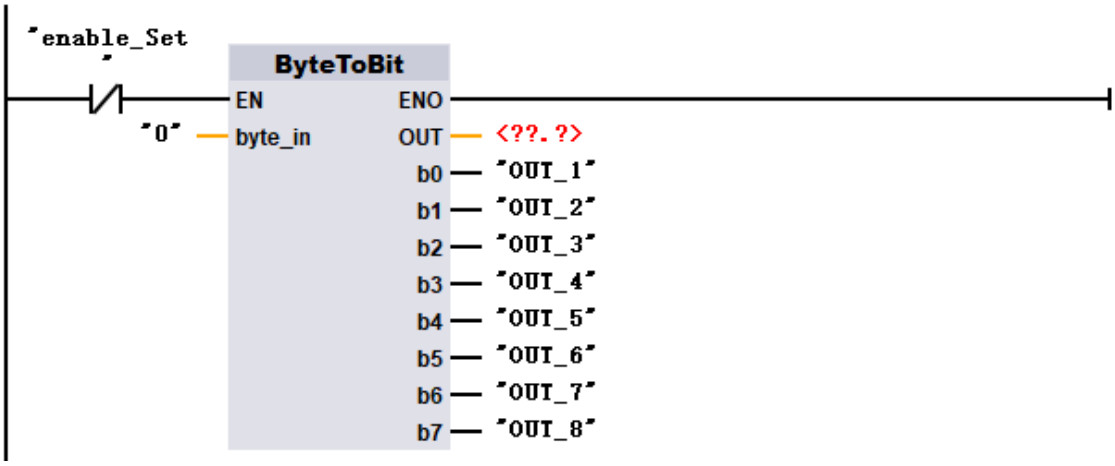
The following example shows how the instruction works:



Variable "Out" is reset when one of the following conditions is fulfilled:
The operand "var_1" and "var_2" has the signal state "1".
The signal state of the operands "var_1" and "var_3" is "1".

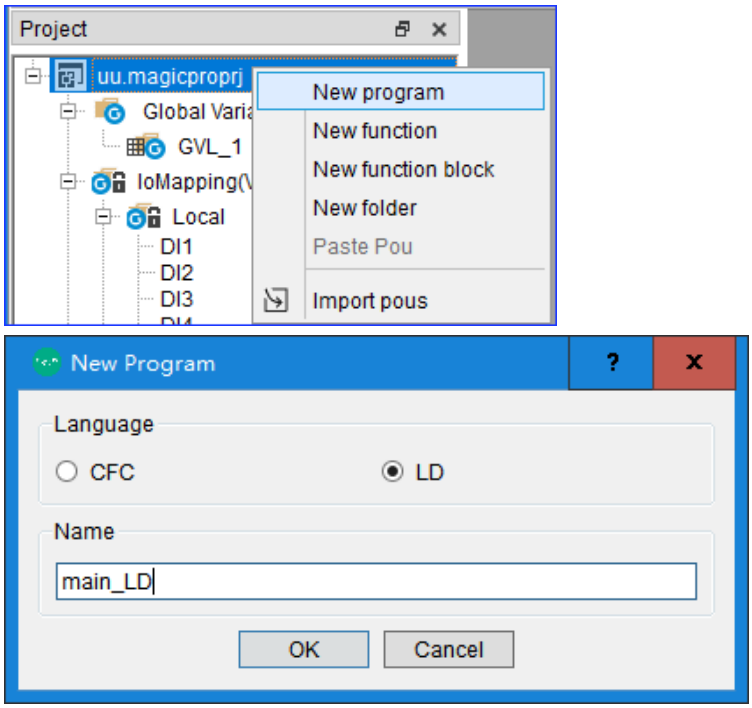
6.1.4 Function Block Elements

The black pins of a function block represent the BOOL type, and the orange pins represent non-BOOL types.
For the BOOL-type input pins of a function block element, you can either enter a variable name or make a wire connection. For non-BOOL-type pins, only variable names can be entered.
Among the output pins of a function block element, only the first output pin supports wire connection; variable names can only be entered for the other output pins.
（Note: The function block diagram description is omitted as images are not presented. The related function block is ByteToBit, with EN (enable input), ENO (enable output), byte_in (input, set to "0"), and OUT_1 to OUT_8 (outputs corresponding to bits b0 to b7)）

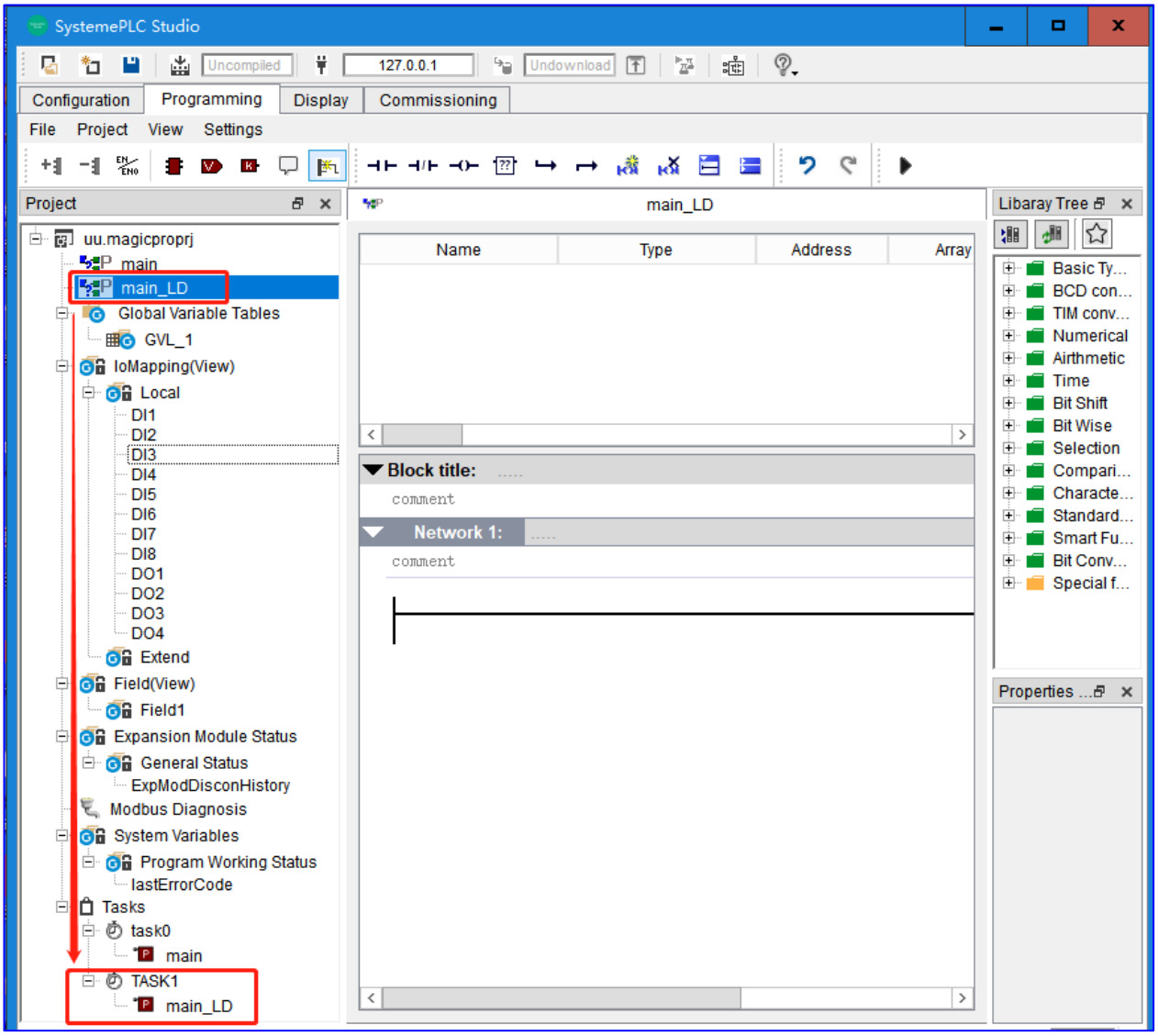


6.2 Creating LD Main Programs and Subprograms

The default programming language for new projects in 3.1 SystemePLC Studio is FBD (Function Block Diagram). To program using LD (Ladder Diagram), LD main programs and subprograms need to be created. The specific operation steps are as follows: Enter the Programming interface, right-click the project name, and create new LD main programs and subprograms.



The "main_LD" program must be dragged into a newly created task, only then can the main program be executed effectively. The LD programming interface is as follows:

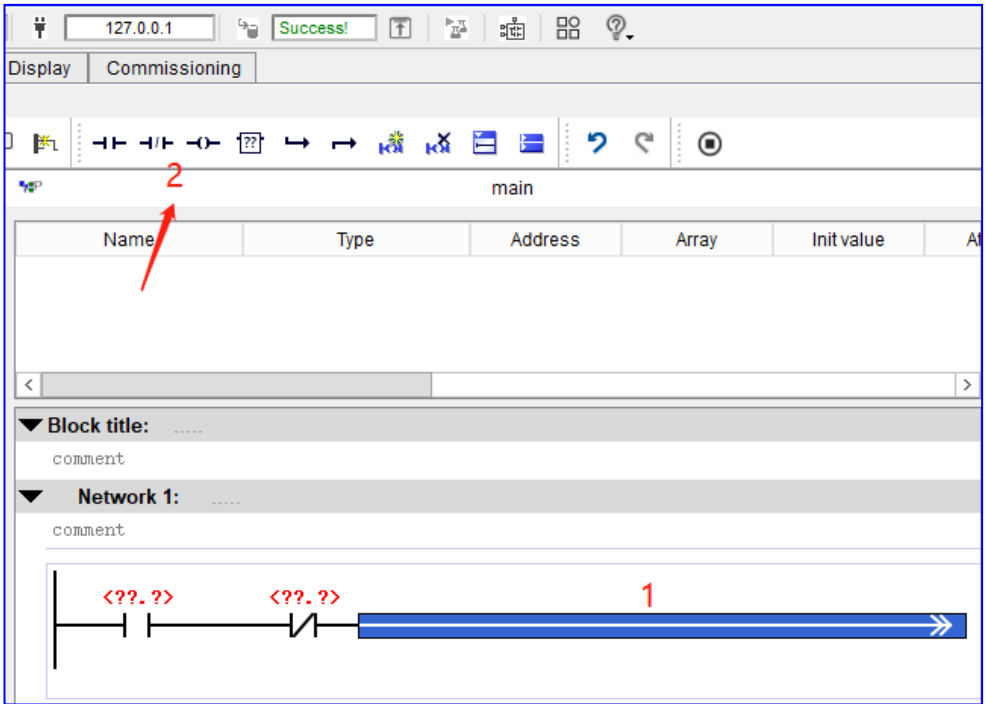


Properties of the Toolbar

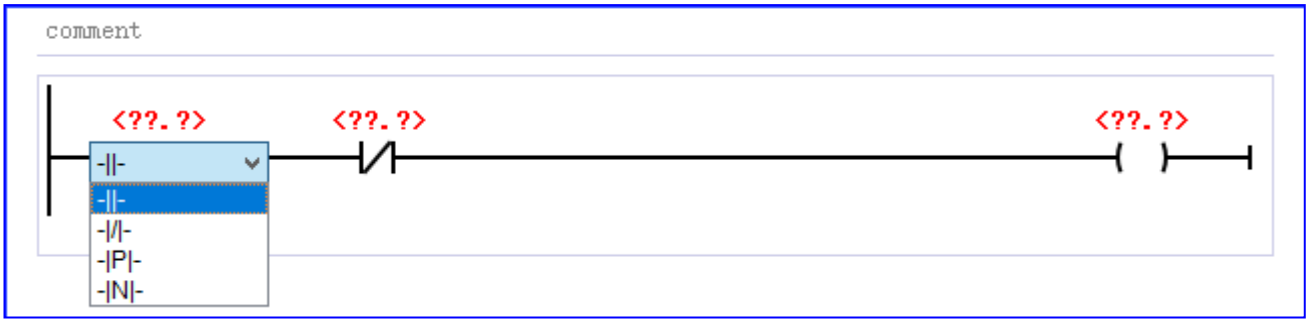
Icon	Name	Description
	Insert contact	Insert a normally open contact
	Insert close contact	Insert a normally closed contact
	Insert coil	Insert a coil
	Insert block	Insert a block
	Insert branch	Insert a downward branch
	Insert up branch	Insert an upward branch
	Insert network	Insert a network
	Delete network	Delete a network
	Open all network	Close all networks
	Close all network	Close all network

6.3 Adding Contact Call Points, Adding Inverted Contacts, or Modifying Inverted Contacts

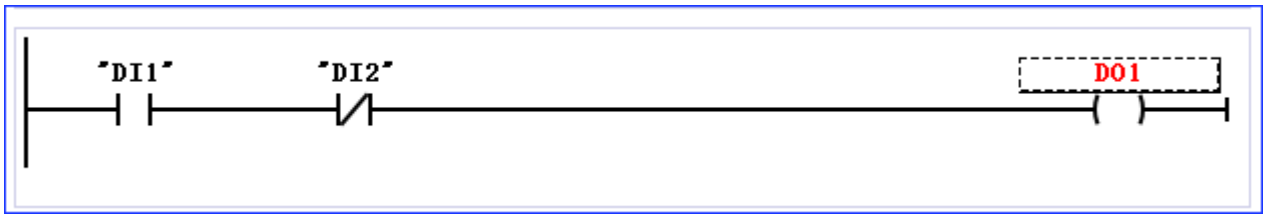
1. Select the position on the network rung where the contact is to be inserted.
2. Click the contact to add it to the network.



3. Double-click the contact on the network to modify the inverted contact.



4. Enter the variable name above the contact to bind the variable to the contact.



6.4 Branches

In Ladder Diagram (LD) programming, series branches and parallel branches are two core logical combination methods. Their essential difference lies in the current flow path and the logical execution sequence.

Series Branch: Multiple contacts/branches are connected in series in sequence. Current can only pass through when

all contacts/branches are conducting, implementing the logical AND function. Series branches have higher priority than parallel branches.

Parallel Branch: Multiple contacts/branches are connected in parallel. Current can pass through as long as any one branch is conducting, implementing the logical OR function.

Specifications for Branches

A parallel branch can only be inserted if the main branch contains at least one LD element.

Parallel branches can be opened downward or directly connected to the power rail; a branch must contain at least one element to be dragged and closed.

To delete a parallel branch, all LD elements in the branch must be deleted. When the last LD element is deleted from the branch, the remaining part of the branch must also be deleted.

Requirements

There must be an available program segment.

The program segment must contain elements.

Deleting a Branch in an LD Program Segment

Select the wire connecting the branch to the main branch.

In the context menu, select the "Delete" command.

Reorganizing Branches in an LD Program Segment

Delete the wire connecting the branch to the main branch.

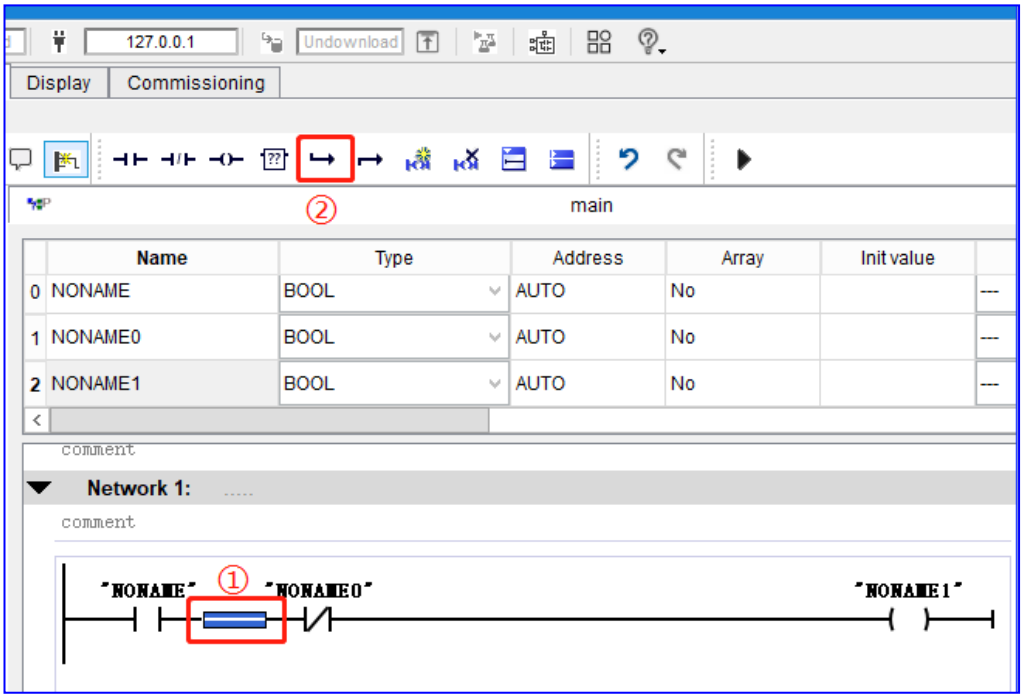
Select the right side of the branch with the mouse, and drag the mouse to reorganize the branch.

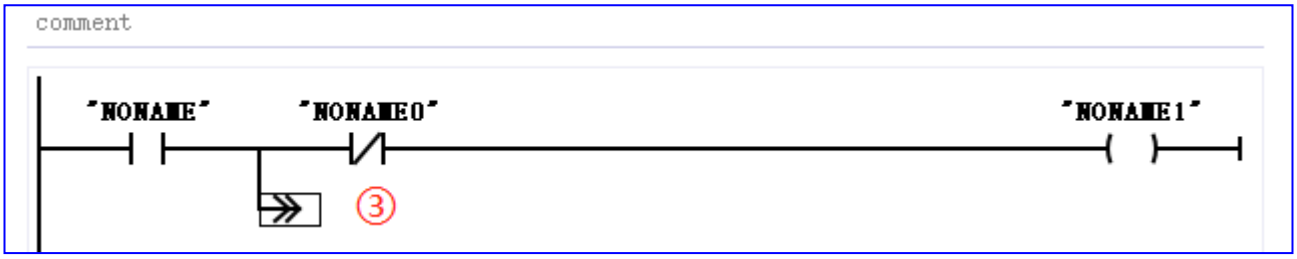
Example

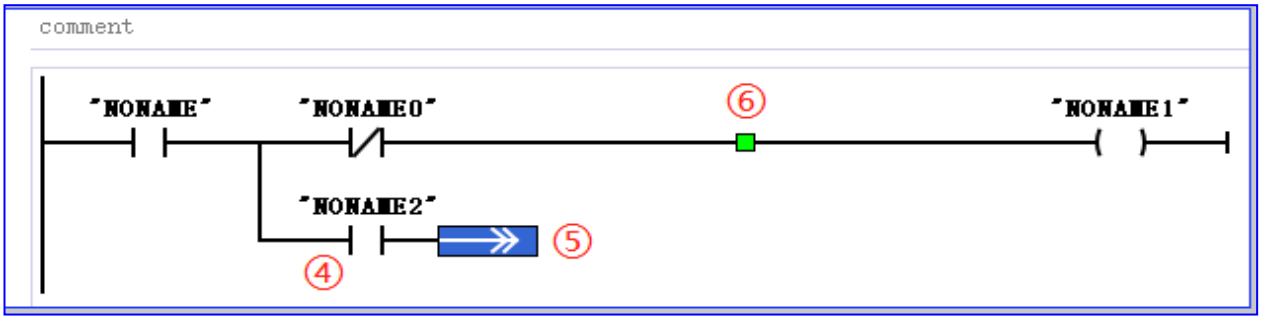
First, write the series branch.

Then add subsequent contacts via the "Insert Branch" function.

After adding the branch contacts, drag the end of the branch to the main branch. A crossover point appearing on the main branch indicates that the branch can be connected.

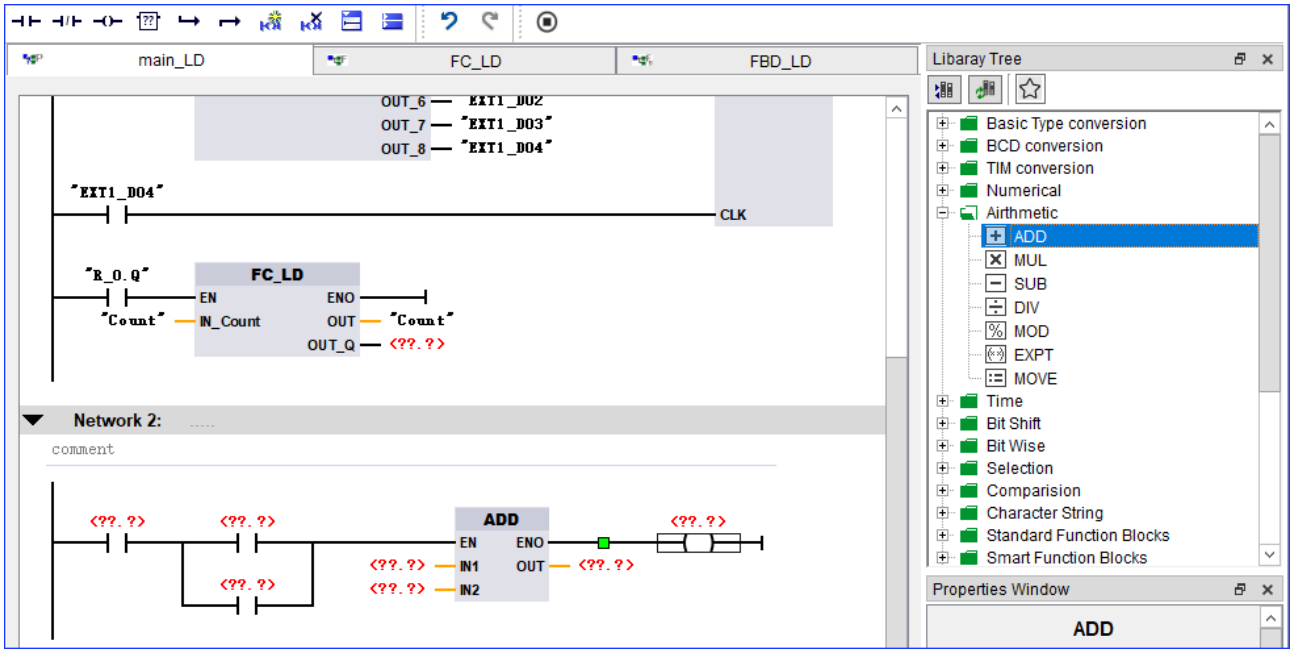






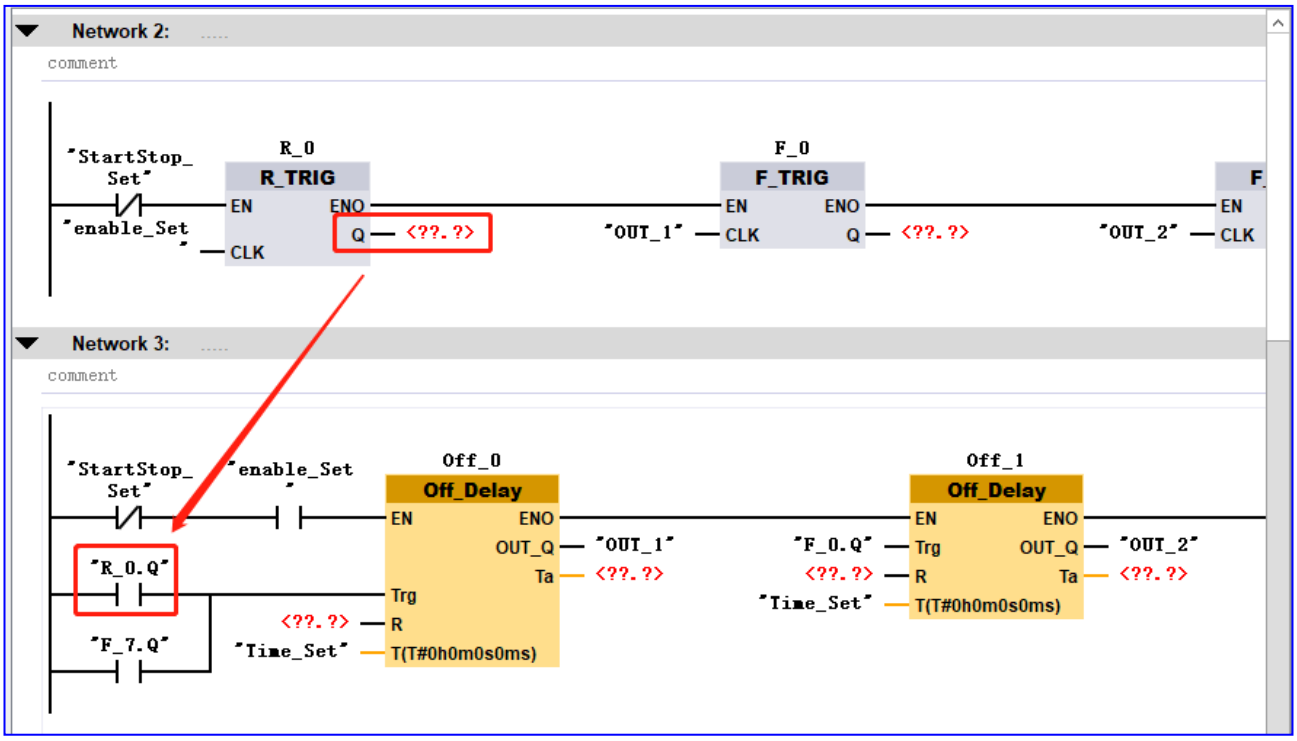
6.5 Adding Instructions

Select an instruction and drag it to the branch. A crossover point appearing on the branch indicates that the instruction can be placed there.



6.6 Quick Call of Points on Instructions

Points on an instruction can be quickly called into the program. The calling format is: Instruction Name.Point Name. For example, to call the output Q point of R_0, directly enter "R_0.Q".

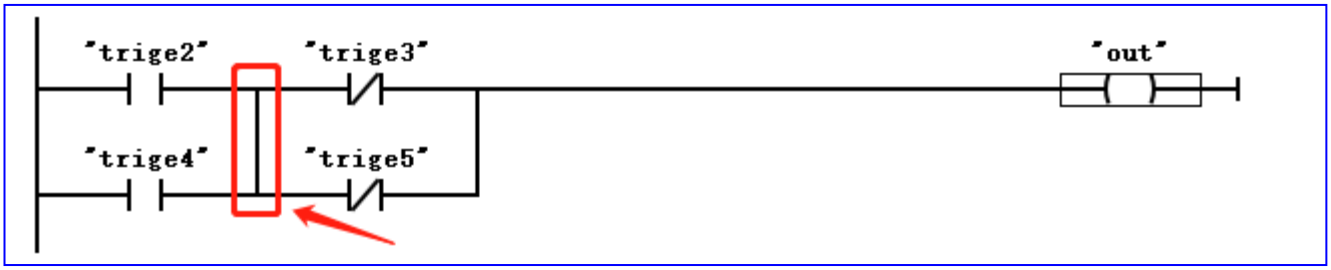


6.7 Deleting Contacts or Instructions

Right-click the contact or instruction and click "Delete" to remove it.

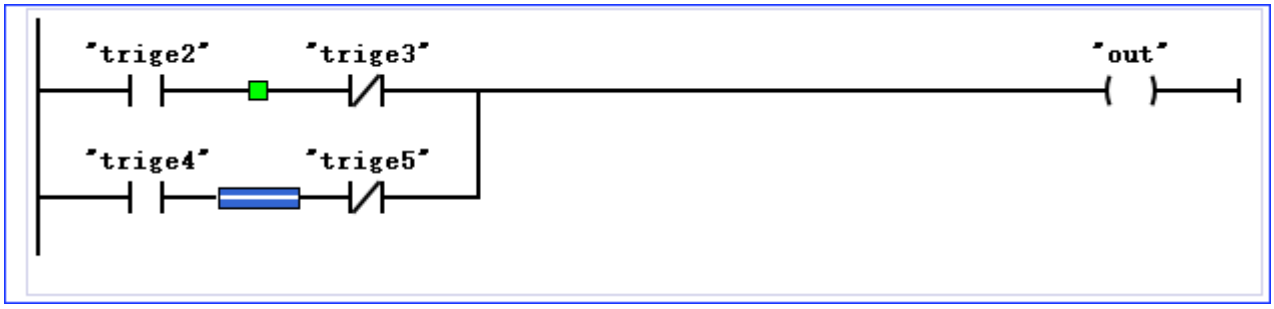
6.8 Crossover Points

Crossover points are inserted into an LD program segment by creating connections between the main branch and additional branches, or between two different branches.



Inserting a Crossover Point

Select the branch to be crossed, press and hold the mouse to drag it to another branch. When a green crossover point appears, drag the mouse to the crossover point to complete the connection.



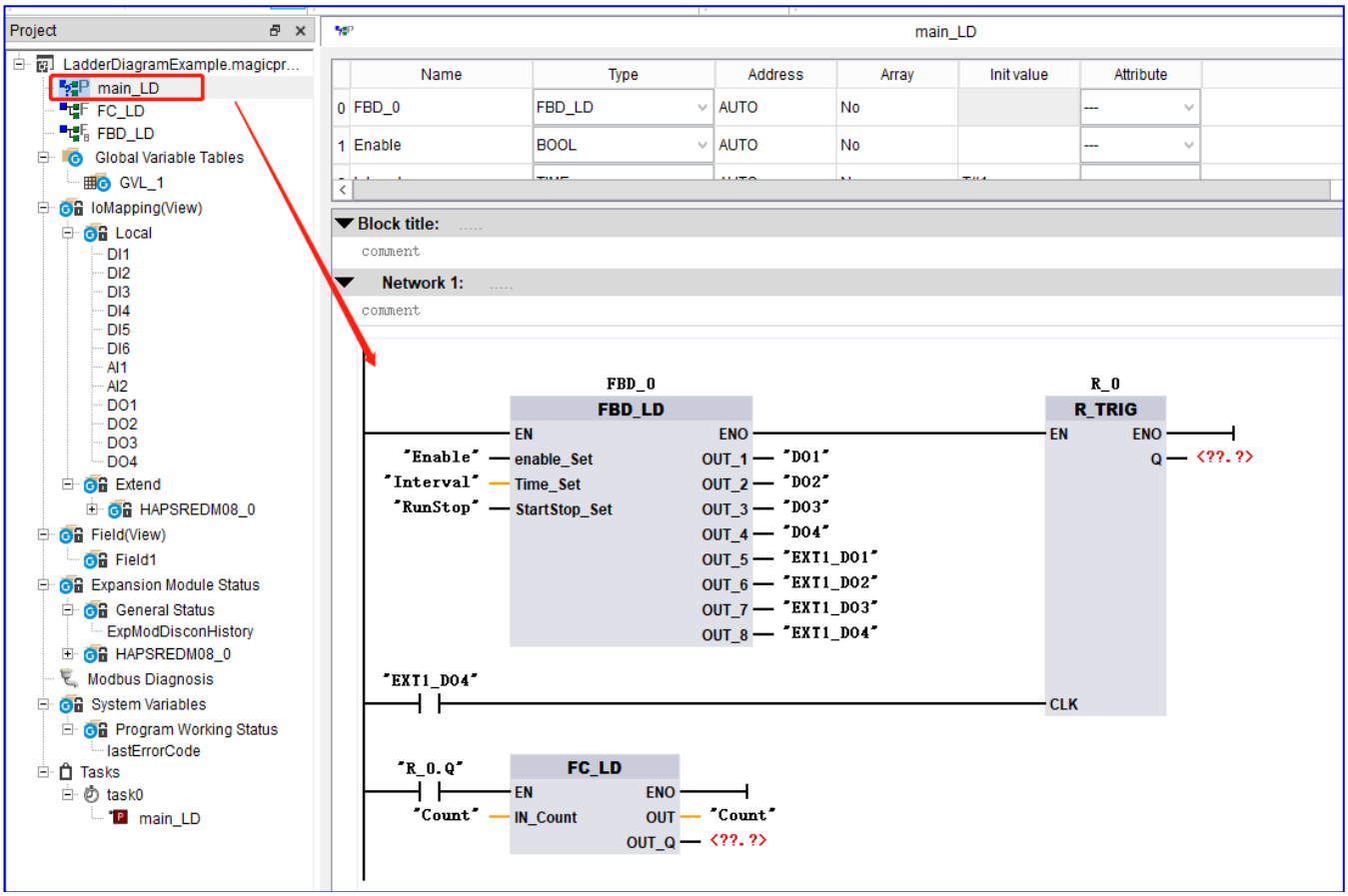
Deleting a Crossover Point

Select the wire defining the crossover point in the corresponding branch.

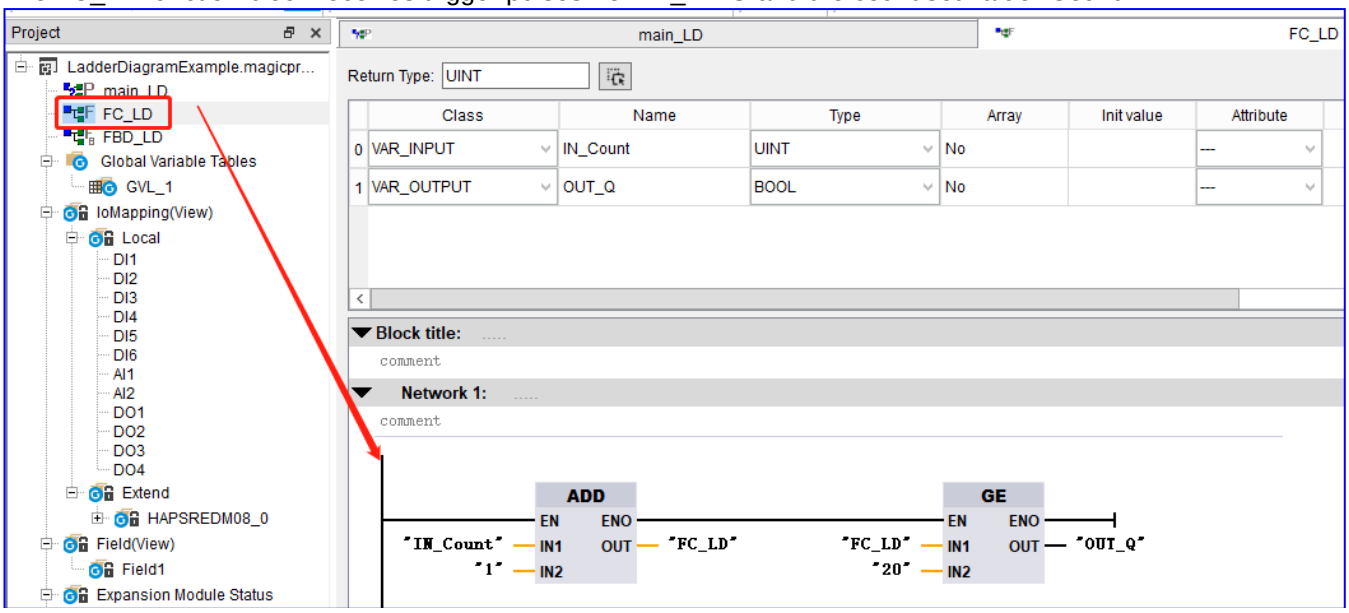
Right-click and select the "Delete" command

6.9 Example of a Chasing Light Program

Call the FBD block to implement the chasing light program.



The FC_LD function block receives trigger pulses from R_TRIG and the count set value "Count".

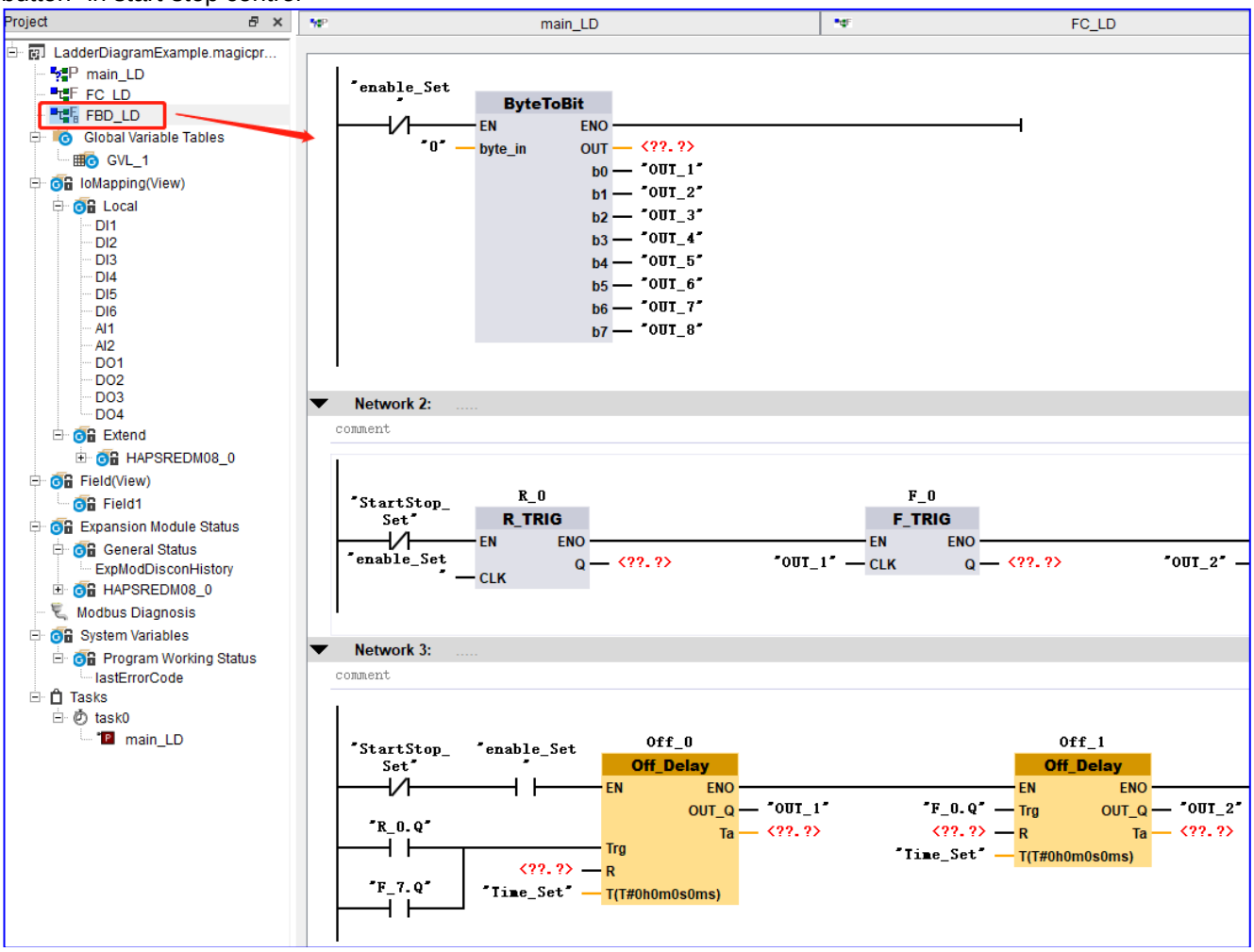


Description of FBD_LD

Network 1: Initialize 8 DOs (Digital Outputs) via enable_Set. When enable_Set is false, the 8 DOs are reset.

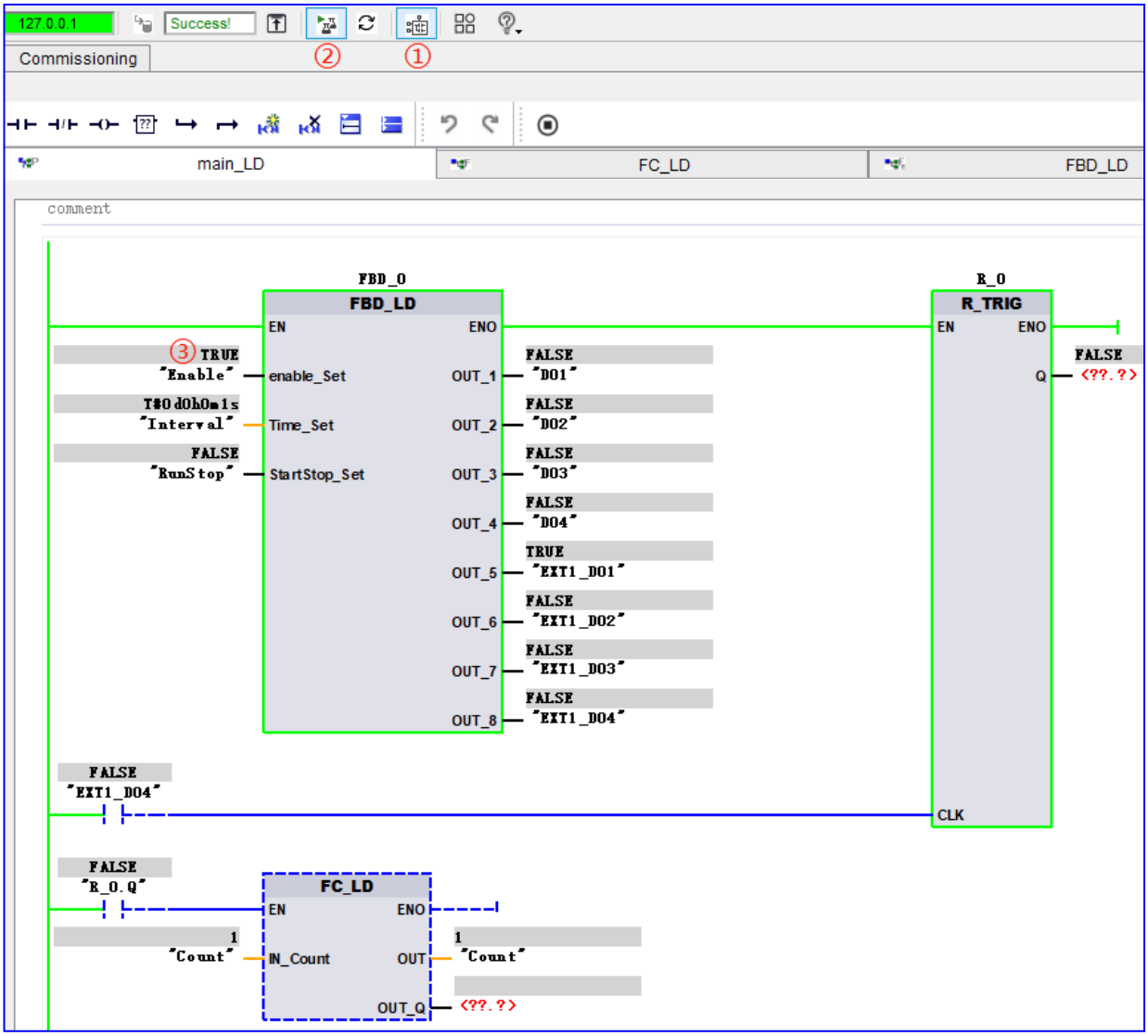
Network 2: Chasing light control. A start-stop button controls the start and stop of the entire chasing light program. R_TRIG captures the start signal, and F_TRIG captures the status reset signal of the 8 DOs and outputs a single pulse to FC_LD.

Network 3: Implement "off-delay" via Off_Delay, which is suitable for the scenario of "delayed stop after releasing the button" in start-stop control

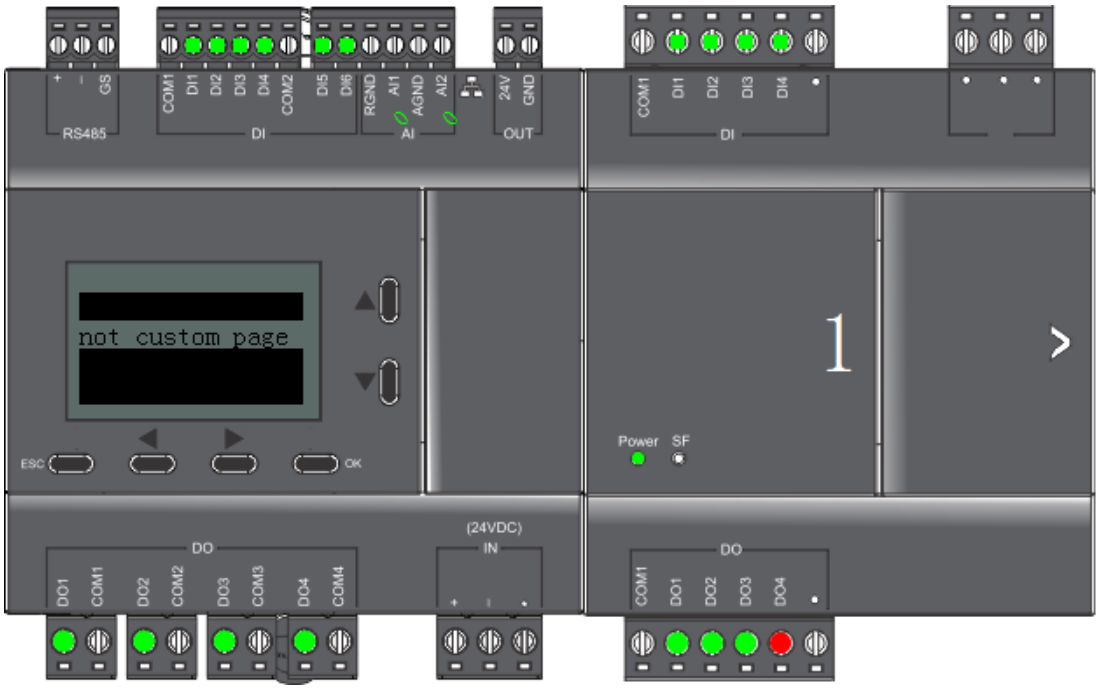


6.10 Display of LD Program Execution Status

1. Enable simulation.
2. Run the program and monitor it.
3. Double-click the actual value above the "enable" variable, modify it to TRUE, and write the change to start the operation of the chasing light program.



8 DOs (Digital Outputs) perform polling output with a value of FALSE.



Seventh section

Fault

Check the error code in the Commissioning interface.

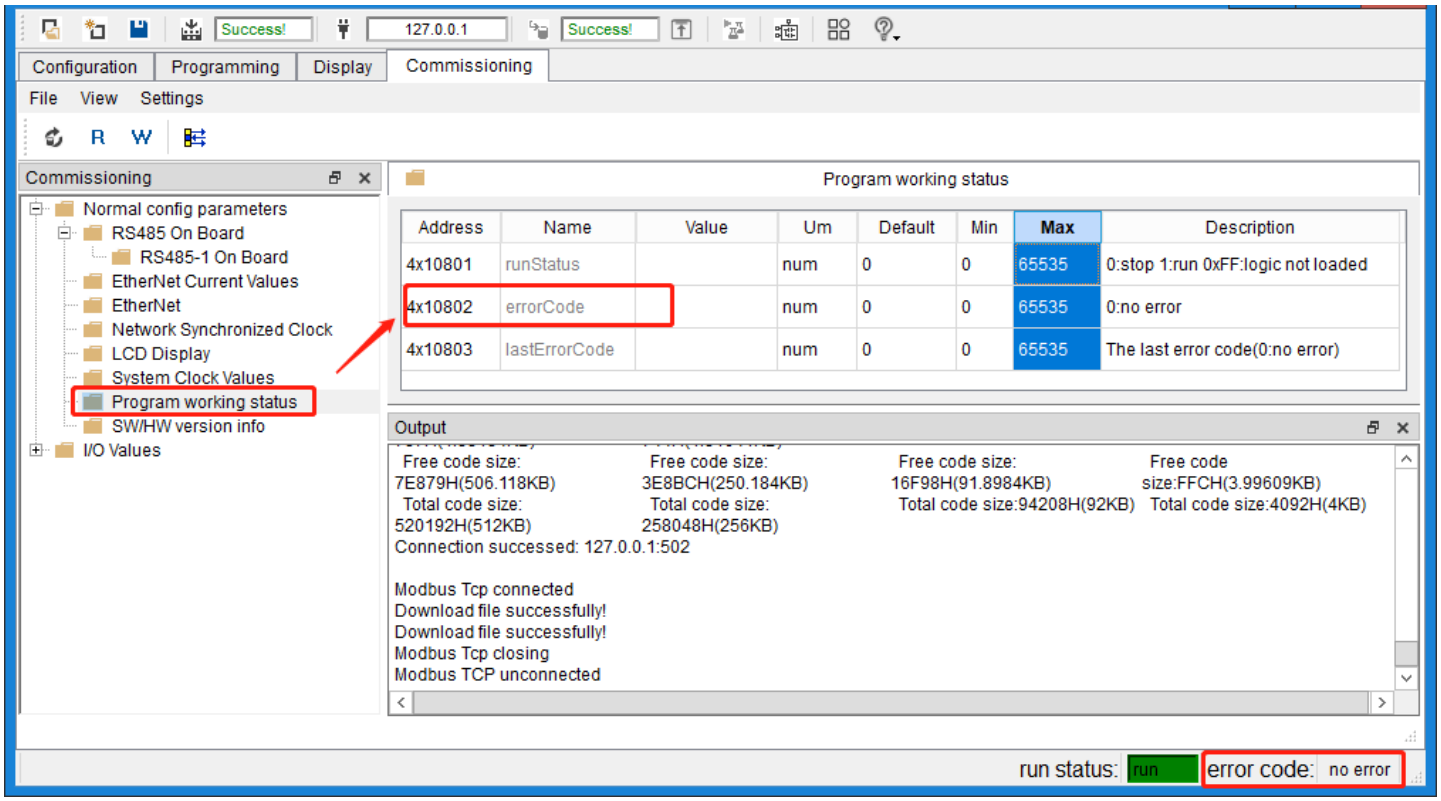


Table 6-1 Fault Table

Code	Alarm Descriptions	LCD display
0	No alarm	No Faults
5	Firmware version incompatible with hardware	
6	Main controller analog input hardware fault	ADC Fault
7	Main controller missing primary power supply	No Power
8	Restart alarm triggered by watchdog timeout	
18	Manual operation stop alarm	Manual Stop
30	User program type error	
31	User program entry initialization failure	
32	User program memory usage exceeds limit	
33	User program address invalid	
34	Retain variable usage exceeds limit	
35	Read/write variable type index out of range	
36	Read/write variable type error	
37	User program FLASH area error	
38	Division-by-zero error in program	ZeroDivision Err
110	Expansion module 1 communication fault	Ext:1 disconnect
111	Expansion module 2 communication fault	Ext:2 disconnect
112	Expansion module 3 communication fault	Ext:3 disconnect
113	Expansion module 4 communication fault	Ext:4 disconnect

114	Expansion module 5 communication fault	Ext:5 disconnect
115	Expansion module 6 communication fault	Ext:6 disconnect
116	Expansion module 7 communication fault	Ext:7 disconnect
118	Configured expansion module not connected to expansion bus	ExtM not found
126	Expansion module 1 configuration mismatch with hardware	Ext:1 config err
127	Expansion module 2 configuration mismatch with hardware	Ext:2 config err
128	Expansion module 3 configuration mismatch with hardware	Ext:3 config err
129	Expansion module 4 configuration mismatch with hardware	Ext:4 config err
130	Expansion module 5 configuration mismatch with hardware	Ext:5 config err
131	Expansion module 6 configuration mismatch with hardware	Ext:6 config err
132	Expansion module 7 configuration mismatch with hardware	Ext:7 config err
134	Expansion module 1 primary power supply abnormal	Ext:1 no power
135	Expansion module 2 primary power supply abnormal	Ext:2 no power
136	Expansion module 3 primary power supply abnormal	Ext:3 no power
137	Expansion module 4 primary power supply abnormal	Ext:4 no power
138	Expansion module 5 primary power supply abnormal	Ext:5 no power
139	Expansion module 6 primary power supply abnormal	Ext:6 no power
140	Expansion module 7 primary power supply abnormal	Ext:7 no power
210	Expansion module 1 AI channel out of range	Ext:1 AI OR
211	Expansion module 2 AI channel out of range	Ext:2 AI OR
212	Expansion module 3 AI channel out of range	Ext:3 AI OR
213	Expansion module 4 AI channel out of range	Ext:4 AI OR
214	Expansion module 5 AI channel out of range	Ext:5 AI OR
215	Expansion module 6 AI channel out of range	Ext:6 AI OR
216	Expansion module 7 AI channel out of range	Ext:7 AI OR
232	User test warning indicator	Test Le

Eighth section

Appendix

8.1 Checking the IP address of the PLC

8.2 Modifying the PLC IP and gateway

8.3 PLC run/stop

8.4 PLC clock

8.5 Checking the PLC baud rate

8.6 Firmware upgrades

8.7 PLC info

8.8 LCD

8.9 Ordering information

8.9 Update firmware from USB flash

8.10 Update firmware from USB type-C cable

8.11 Update firmware from Ethernet cable

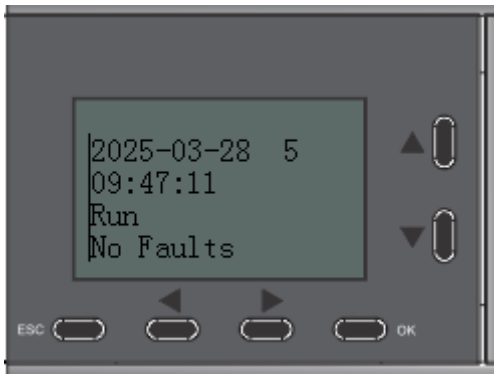
8.12 Expansion bus characteristics

8.13 Ordering information

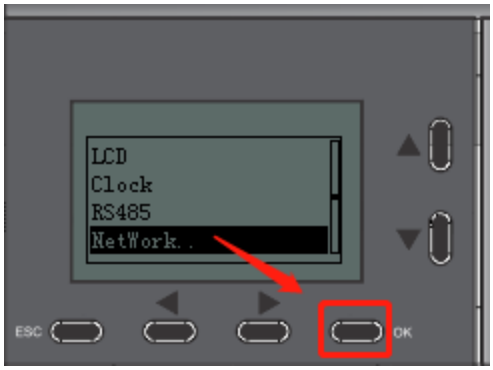
8.1 Checking the IP address of the PLC

After the PLC is powered on, the PLC display will show the IP address of the PLC, and the specific viewing steps are as follows:

1. PLC power on the PLC will display some information about the PLC.



2. Press any button to enter the main menu, through the down button to find “ Network...”. Then press OK to enter “ Network...”. In “ Network...”, you can check the IP address and gateway of PLC.

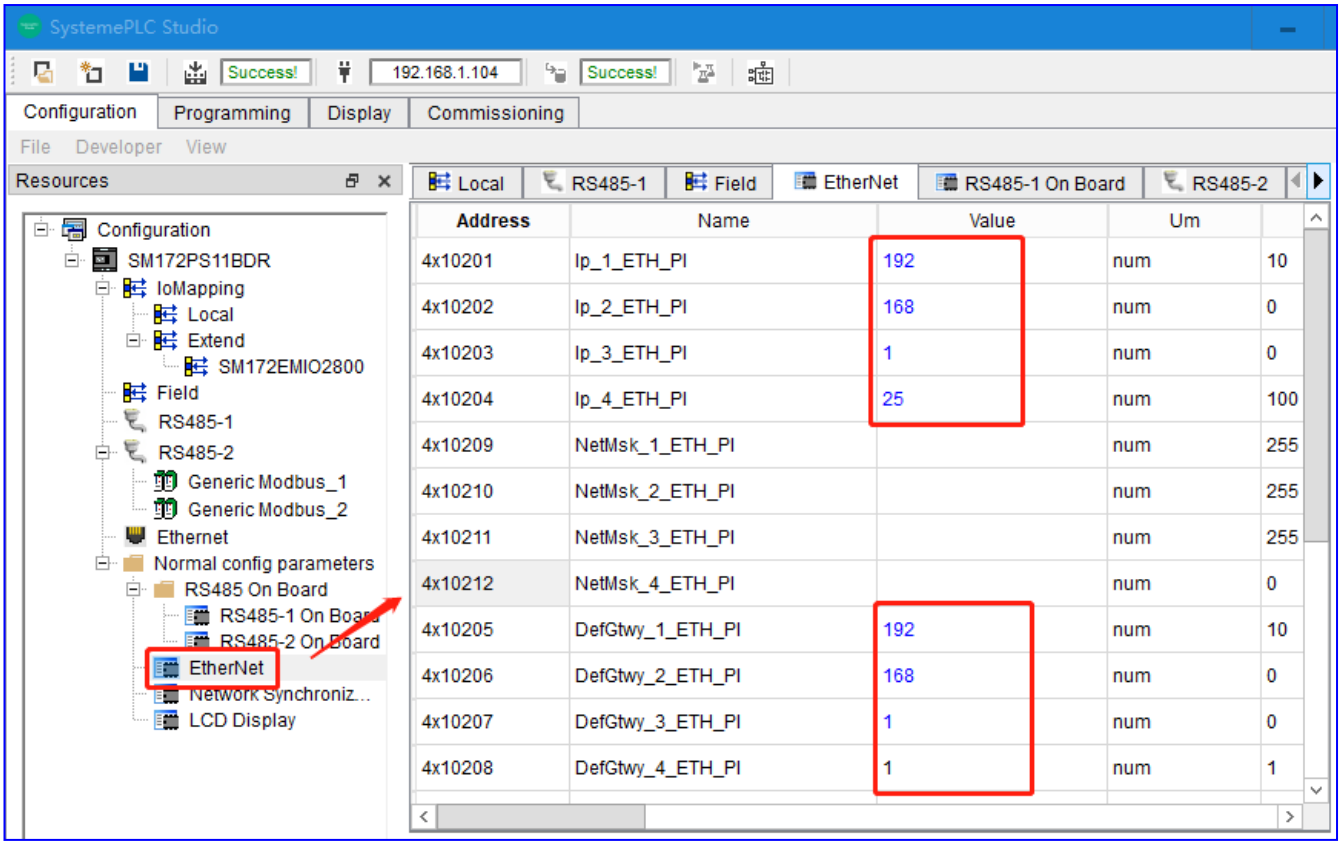


8.2 Modifying the PLC IP and gateway

There are two ways to modify the PLC IP and gateway

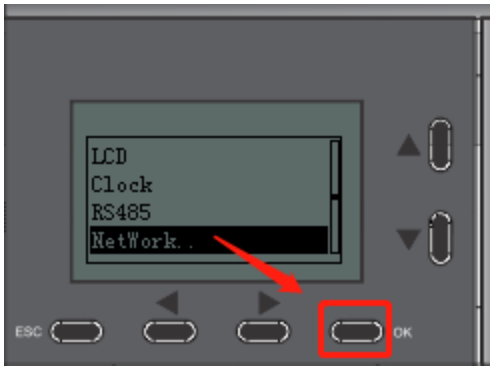
Method 1: Modify the IP and gateway of PLC through the host computer.

Enter the Ethernet page, manually input the IP address and gateway of PLC, and then restart after power off after downloading to PLC.



Method 2: manually modify the IP address and gateway by PLC button.

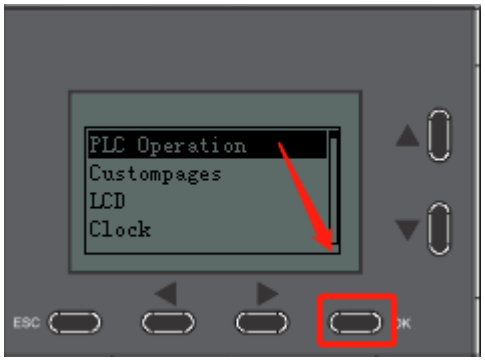
Enter the main menu of the display and find “Network...” and press OK to enter “Network...”, you can modify the IP address and gateway of PLC.

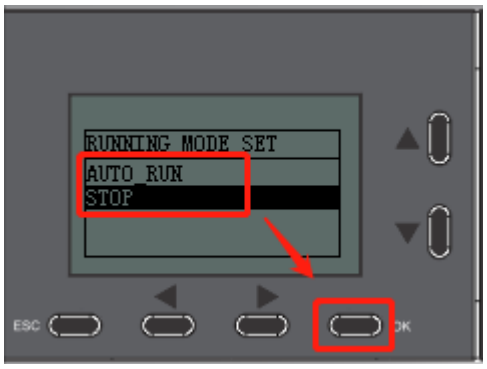


To modify the IP, select the IP line and press OK to confirm, then the first value of the IP will be selected, then select the specific IP field by using the left and right buttons, and modify the IP by using the up and down buttons, and finally confirm. The operation of modifying gateway is the same as modifying IP. Reboot the device after modifying IP and gateway to take effect.

8.3 PLC run/stop

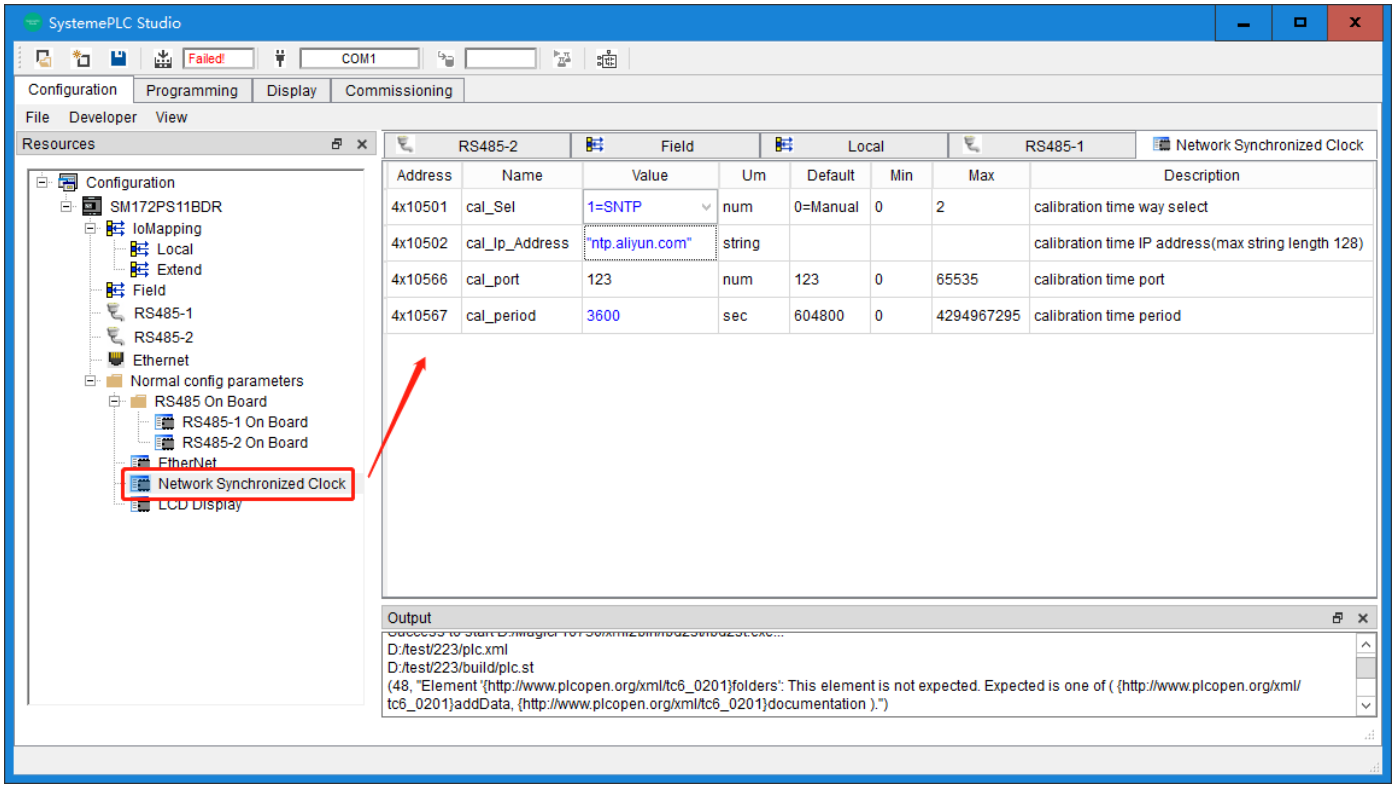
After the PLC is powered on, enter the PLC Operation menu on the display and set the PLC to AUTO RUN mode or STOP mode.

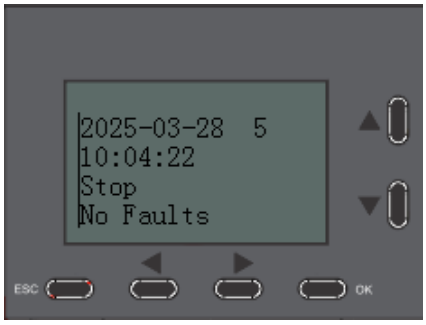




8.4 PLC clock

Select the SNTP protocol, input the real-time clock address or IP, input the port number, and after downloading to the PLC, the PLC clock will be synchronised with the network clock.

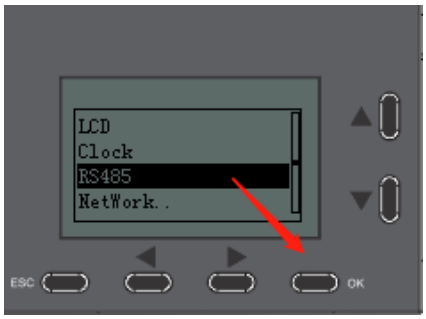


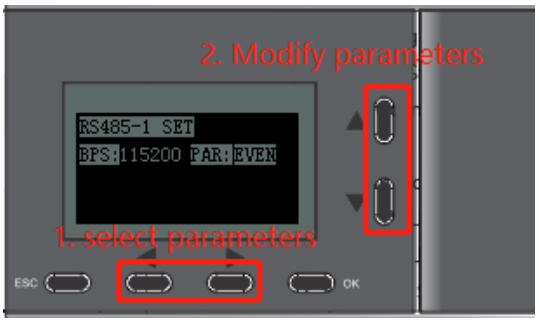


8.5 Checking the PLC baud rate

To check the baud rate of PLC from the display, operate as follows:

Press any button to enter the main menu, find RS485 by down key, then press OK to enter the RS485 menu, you can view the baud rate of PLC in the RS485 menu. You can also directly modify the 485 parameters.





8.6 Firmware upgrades

Insert the USB flash drive into the type A port of PLC, the RUN light turns red and flashes on behalf of the loading process, the screen displays the loading progress, and the device automatically reboots after loading is completed, and the RUN light turns green.

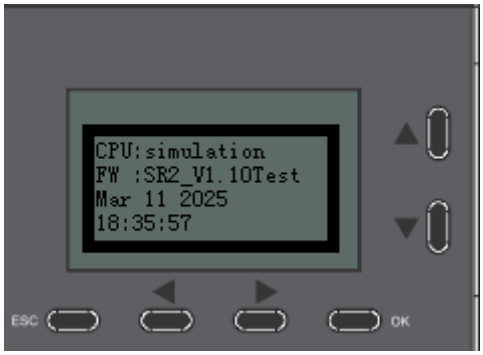
8.7 PLC info

Display basic information of PLC:

1. CPU: Display the actual model of the CPU (as shown in the figure below, the PLC is currently being simulated)

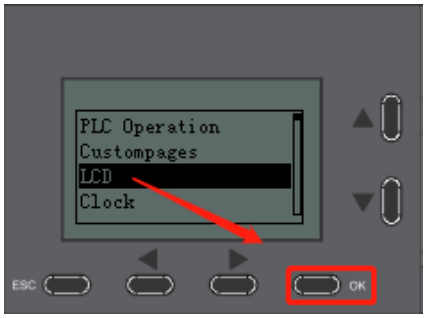
2. FW: Firmware information

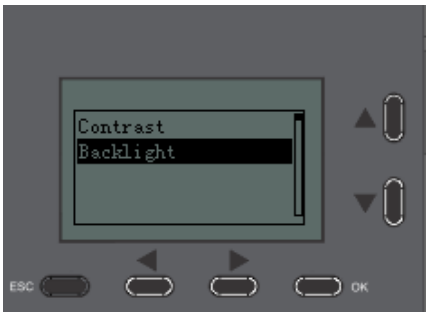
3. PLC time



8.8 LCD

Set Contrast and Backlight, the backlight can be set to "Always On" and "Auto Turn Off."





8.9 Update firmware from USB flash

Memory capacity of USB flash: 32GB or smaller.

File system format USB flash: FAT32.

Firmware upgrade via Type A port. PLC models with a built-in Type A port are as follows: ZR2PP11P7, ZR2PP11BD, SM172PS11BDR, SM172PS11BDM, SM172PS11BDT.

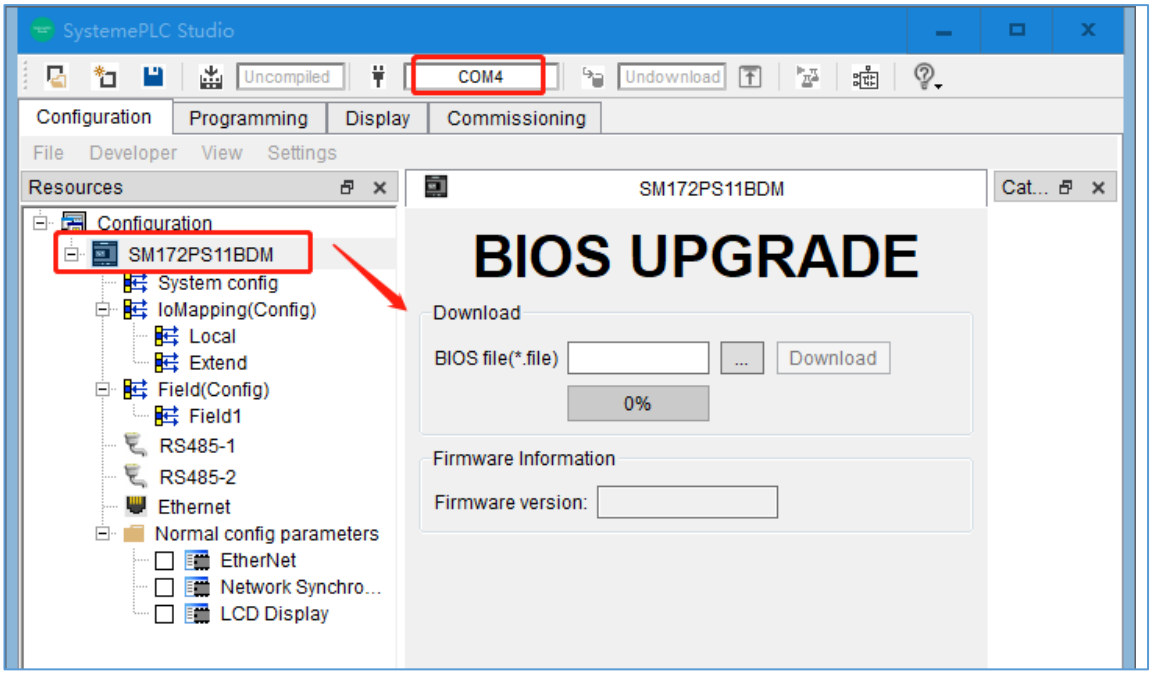
Step:

1. Rename the firmware file to firmware.bin and place it in the root directory of the USB flash drive.
2. Connect the USB flash to the PLC's Type A port,
3. access "USB Flash" on the PLC display, select "PLC Firmware", press "OK" to start the upgrade.
4. After completion, the device will automatically restart.

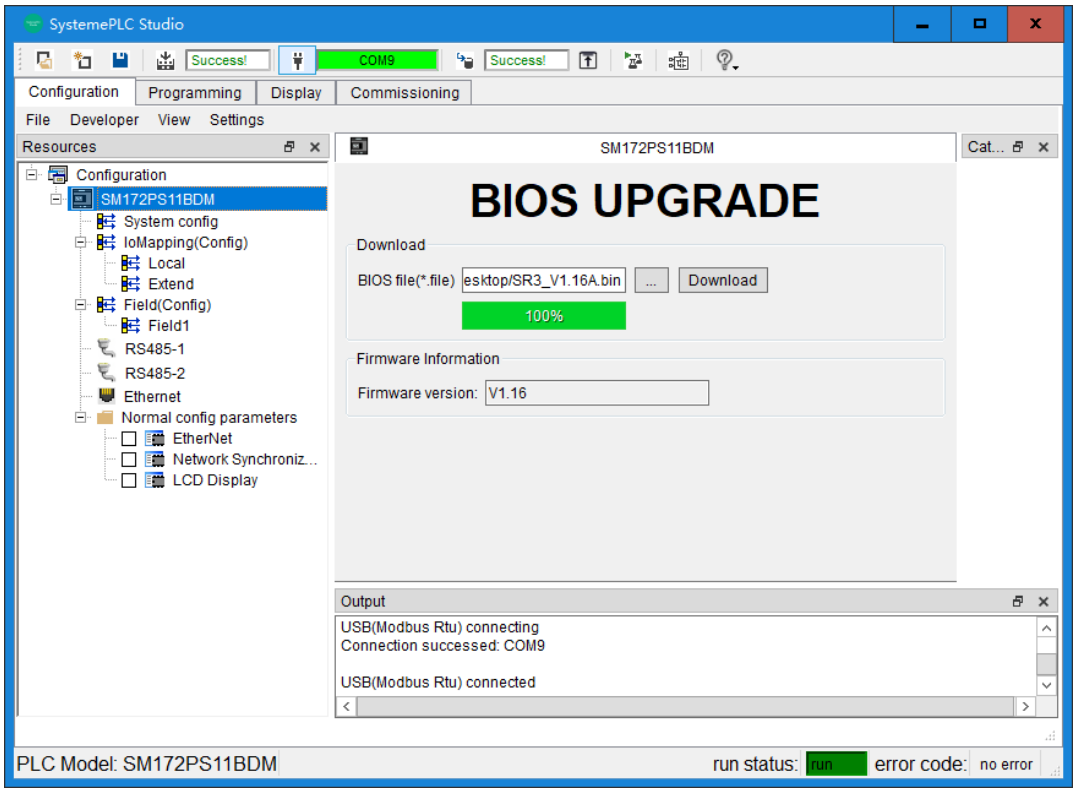
8.10 Update firmware from USB type-C cable

The steps to upgrade the firmware from USB Type-C cable are as follows:

1. Double-click the device model in the Configuration interface and select the COM port to upgrade the firmware. Once the device is connected, its model will appear in the bottom-left corner of the software. The device undergoing firmware upgrade must correspond to the model of the engineering equipment.



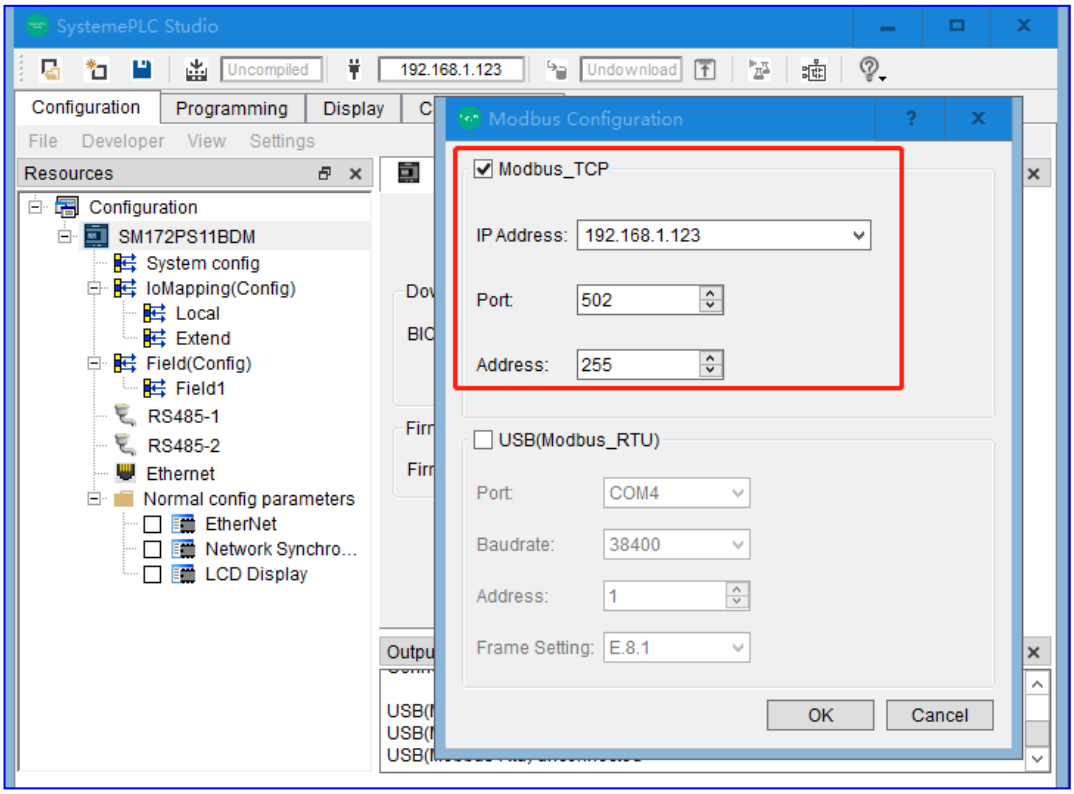
2. Select the firmware file and click Download. A progress bar will appear in the software. For devices with screens, the download progress will be displayed on the screen and the RUN indicator will flash red. For devices without screens, the RUN indicator will flash red. After the firmware transfer completes and the device upgrades, it will reboot. The software will also display a download success message, and the connection will be disconnected due to the device restart.



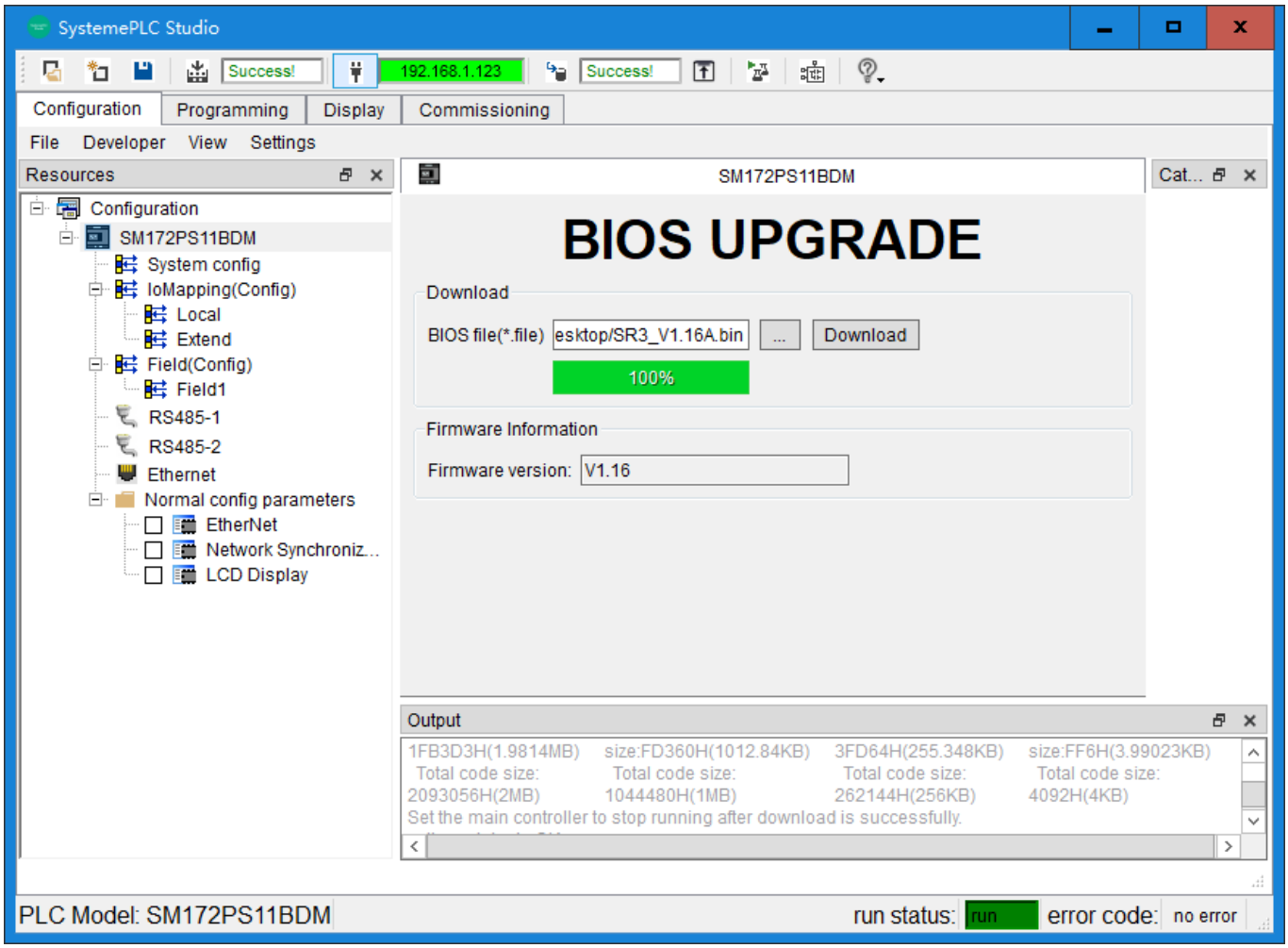
8.11 Update firmware from Ethernet cable

PLC models supporting firmware upgrades via Ethernet port: SM172PS11BDR, SM172PS11BDT, SM172PS11BDM, ZR2PB11P7, ZR2PA11BD, ZR2PP11BD2A.

Connect the PLC's Ethernet port to SystemePLC Studio using a standard Ethernet cable. In SystemePLC Studio, check the "Modbus_TCP " and ensure the IP address matches that of the device.

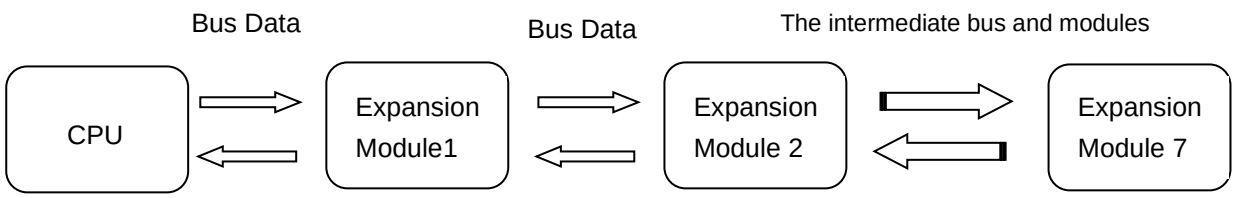


Double-click the device under Configuration. After connecting the device, the device model will be displayed at the bottom left corner of the software, and the device for firmware upgrade must match the project's device model. Select the firmware file and click "Download". During the download process, the RUN indicator will flash, and the download progress will be displayed on the PLC's built-in display screen.



8.12 Expansion bus characteristics

Block Diagram of the Expansion Bus



Expansion Bus Characteristics

The CPU supplies power to the expansion bus, which covers the bus control circuits of all expansion modules. The modules are connected in series end-to-end via their bus connectors, ensuring the accuracy of the bus power-on timing data when the CPU is powered on.

The Smart Relay expansion bus can support up to 7 expansion modules. The module address is determined by three address buses, with address codes ranging from 000 to 110 (binary). The address of the first module is assigned by the CPU as 0, and the addresses of subsequent modules are incremented by 1 in sequence, each determined by the previous module. This address assignment process is handled by an independent bus coprocessor, and once assigned, the addresses remain unchanged. Therefore, hot-swapping of expansion modules via the bus is not supported; otherwise, it will cause abnormalities in the pre-assigned module addresses during CPU power-on, leading to data access errors.

Module Startup Configuration Process

After the CPU is powered on or restarted, it first verifies the number of normally connected modules on the bus. It then checks the module types and addresses configured in the user program, comparing them with the actually connected modules to identify configuration errors or missing modules. The CPU will output corresponding error alarm information. Upon completion of the configuration check, the CPU adjusts the I/O signals of itself and all expansion modules, such as the signal types and adjustment ranges of AI (Analog Input) and AO (Analog Output).

Process of Accessing Expansion Modules

Once the CPU is started and the expansion modules are configured, when the user program begins running, the CPU reads and writes the I/O data of the expansion modules via the expansion bus in each logic cycle.

During the process of reading and writing to the expansion modules, the CPU simultaneously monitors the module status. If a module malfunctions, the fault information (such as bus anomalies, module power supply failures, module configuration errors, and I/O over-range) is transmitted to the alarm-related data fields.

8.13 IO Mapping

The SR1 controller does not support expansion modules; the maximum value of n is determined by the onboard I/O. The SR2 / SM172 controller supports expansion modules. The total-channel limits for onboard and expansion I/O are as follows: DI/DO ＜ 240, AI ＜ 72, AO ＜ 32.

Folder	Label	Memory-mapped programming address	PLC register address	Modbus slave	R/W	Rang
DI	LocalDigInput[0]	%IX0.0	1x0001	DiscreteInputsReg 0	R	0、1
	LocalDigInput[1]	%IX0.1	1x0002	DiscreteInputsReg 1		
	...	...	...	...		
	LocalDigInput[n]	%IX0.n	1x（0001+n）	DiscreteInputsReg n		
AI	LocalAInput[0]	%IW1.0	3x0001	inputReg 0	R	-
	LocalAInput[1]	%IW1.1	3x0002	inputReg 1		
	...	...	...	...		
	LocalAInput[n]	%IW1.n	3x（0001 + n）	inputReg n		
DO	LocalDigOutput[0]	%QX0.0	0x0001	Coil Reg 1	R/W	0、1
	LocalDigOutput[1]	%QX0.1	0x0002	Coil Reg 2		
	...	...	...	...		
	LocalDigOutput[n]	%QX0.n	0x（0001+n）	Coil Reg n		
AO	LocalAOutput[0]	%QW1.0	4x0001	Holding Reg 0	R/W	-
	LocalAOutput[1]	%QW1.1	4x0002	Holding Reg 1		
	...	...	...	...		

	LocalAOutput[n]	%QW1.n	4x（0001 + n）	Holding Reg n		

The I/O mapping extended via MODBUS RTU/TCP is shown in the following table, with the limit on n as follows:

SM172 / SR2: DI/DO < 4096, AI/AO < 2048。SR1: DI/DO <512, AI/AO < 256.

Folder	Label	Memory-mapped programming address	PLC register address	Modbus slave	R/W	Rang
DI	MBExtDigInput[0]	%IX10.0	1x1001	DiscreteInputsReg 1000	R	0、1
	MBExtDigInput[1]	%IX10.1	1x1002	DiscreteInputsReg 1001		
	...	...	...	...		
	MBExtDigInput[n]	%IX10.n	1x（1001+n）	DiscreteInputsReg 1000+n		
AI	MBExtAInput[0]	%IW11.0	3x1001	inputReg 1000	R	-
	MBExtAInput[1]	%IW11.1	3x1002	inputReg 1001		
	...	...	...	...		
	MBExtAInput[n]	%IW11.n	3x(1001 + n)	inputReg 1000+n		
DO	MBExtDigOutput[0]	%QX10.0	0x1001	Coil Reg 1000	R/W	0、1
	MBExtDigOutput[1]	%QX10.1	0x1002	Coil Reg 1001		
	...	...	...	...		
	MBExtDigOutput[n]	%QX10.n	0x(1001+n)	Coil Reg 1000 + n		
AO	MBExtAOutput[0]	%QW11.0	4x1001	Holding Reg 1000	R/W	-
	MBExtAOutput[1]	%QW11.1	4x1002	Holding Reg 1001		
	...	...	...	...		
	MBExtAOutput[n]	%QW11.n	4x(1001 + n)	Holding Reg 1000+n		

HSC (FDI) is stored in the M memory area.

Label	Memory-mapped programming address	PLC register address	Modbus slave	R/W	Description	Rang
HSC[0].cnt	%MD5.0	3x0501	inputReg 500	R	-	-
...	...	...	...			
HSC[n].cnt	%MD5.n	3x0501+n	inputReg 500+2n			
HSC[0].Hz	%MD16.0	3x0601	inputReg 600	R	frequency	-
...	...	...	...			
HSC[n].Hz	%MD16.n	3x0601+n	inputReg 600+2n			

HSC[0].rst	%MX6.0	0x0501	Coil Reg 500	R/W	-	0、1
...	...	...	...			
HSC[n].rst	%MX6.n	0x0501+n	Coil Reg 500+n			

FDO is stored in the M memory area.

Label	Memory-mapped programming address	PLC register address	Modbus slave	R/W	Description	Rang
PWM[0].en	%MX11.0	0x0551	Coil Reg 550	R/W	FDO Enable Pulse Output	-
...	...	...	...	R/W		
PWM[n].en	%MX11.n	0x0551+n	Coil Reg 550+n	R/W		
PWM[0].Hz	%MW12.0	4x0551	Holding Reg 550	R/W	FDO Pulse Output Frequency Range: 30~30000Hz	-
...	...	...	...	R/W		
PWM[n].Hz	%MW12.n	4x0551+n	Holding Reg 550+n	R/W		
PWM[0].df	%MW13.0	4x0561	Holding Reg 560	R/W	Duty Cycle Adjustment Range: 1-100	0、1
...	...	...	...	R/W	FDO Pulse Output Duty Cycle	0、1
PWM[n].df	%MW13.n	4x0561+n	Holding Reg 560+n	R/W		0、1

KEY is stored in the M memory area.

Folder	Label	Memory-mapped programming address	Modbus slave	R/W	Rang
KEY	UP	%MX2.0	DiscreteInputsReg 600	R	0、1
	DOWN	%MX2.1	DiscreteInputsReg 601	R	0、1
	LEFT	%MX2.2	DiscreteInputsReg 602	R	0、1
	RIGHT	%MX2.3	DiscreteInputsReg 603	R	0、1
	F1	%MX2.4	DiscreteInputsReg 604	R	0、1
	F2	%MX2.5	DiscreteInputsReg 605	R	0、1
	F3	%MX2.6	DiscreteInputsReg 606	R	0、1
	F4	%MX2.7	DiscreteInputsReg 607	R	0、1
	ESC	%MX2.8	DiscreteInputsReg 608	R	0、1
	OK	%MX2.9	DiscreteInputsReg 609	R	0、1

8.14 Ordering information

Table 7-1 Ordering Information

Specification Description	Order No
Controller	
Universal type 1M program space, 256KB data space, 24VDC power supply, with display screen, supporting up to 7 module extensions. 8DI（24VDC）, 8DO (Electromagnetic Relay 3A), 8AI(Voltage signal: 0-10VDC, Current signal: 4-20mA/0-20mA, Resistance signal: 0-30kΩ, support NTC_10K (3950) or PT1000 PT100), 4AO(0-10 V, 4-20 mA), 2 RS485 communication ports, 1 USB interface (Type C), 1 USB-A interface, 1 RJ45 communication port.	SM172PS11BDR
Universal type 1M program space, 256KB data space, 24VDC power supply, with display screen, supporting up to 7 module extensions. 8DI（24VDC）, 8DO (transistor drain output 0.5A), 8AI(Voltage signal: 0-10VDC, Current signal: 4-20mA/0-20mA, Resistance signal: 0-30kΩ, support NTC_10K (3950) or PT1000 PT100), 4AO(0~10V, 4~20mA), 2 RS485 communication ports, 1 USB interface (Type C), 1 USB-A interface, 1 RJ45 communication port.	SM172PS11BDT
Universal type 1M program space, 256KB data space, 24VDC power supply, with display screen, supporting up to 7 module extensions. 8DI（24VDC）, 8DO (6 channel 3A electromagnetic relay outputs, 2 channel 0.3A/240VAC solid-state relay outputs.), 8AI(Voltage signal: 0-10VDC, Current signal: 4-20mA/0-20mA, Resistance signal: 0-30kΩ, support NTC_10K (3950) or PT1000 PT100), 4AO(0-10 V, 4-20 mA), 2 RS485 communication ports, 1 USB interface (Type C), 1 USB-A interface, 1 RJ45 communication port.	SM172PS11BDM
Basic type 1MB program space, 256KB data space, 220VAC power supply, with display screen, supporting up to 7 extension modules.	ZR2PB11P7

8DI (220VAC), 8DO (Electromagnetic Relay 3A), 1 RS485 communication Interface, 1 USB interface (Type C), 1 RJ45 communication interface.	
Basic type 1MB program space, 256KB data space, 24VDC power supply, with display screen, supporting up to 7 extension modules. 8UI (Supports DI or AI) (AI supports Voltage signal: 0-10VDC, Current signal: 4-20mA/0-20mA, Resistance signal: 0-30kΩ, support NTC_10K (3950) or PT1000 PT100), 4DO (2-channel electromagnetic relay 3A, 2-channel transistor drain output 0.5A), 1 RS485 communication Interface, 1 USB-C Interface, 1 RJ45 communication interface.	ZR2PA11BD
Basic type 1MB program space, 256KB data space, 24VDC power supply, with display screen, supporting up to 7 extension modules. 6 DI (24VDC), 4DO (Electromagnetic Relay 3A), 2AI(Voltage signal: 0-10VDC, Current signal: 4-20mA/0-20mA, Resistance signal: 0-30kΩ, support NTC_10K (3950) or PT1000 PT100), 1 RS485 communication Interface, 1 USB interface (Type C), 1 RJ45 communication interface.	ZR2PP11BD2A
Basic type 1MB program space, 256KB data space, 220 VAC power supply; with display screen, supporting up to 7 extension modules. 16 DI(220VAC), 12 DO (Relay, 3 A); 1xRS485, 1xRJ45, 1xUSB (Type C), 1xUSB (Type A).	ZR2PP11P7
Basic type 1MB program space, 256KB data space, 24 VDC power supply; with display screen, supporting up to 7 extension modules. 16 DI(24VDC), 12 DO (Relay, 3A), 1xRS485, 1xRJ45, 1xUSB (Type C), 1xUSB (Type A).	ZR2PP11BD
Economic type 256KB program space, 92KB data space, 220VAC power supply, no display screen, not supporting extension modules. 8DI (220VAC), 4DO (Electromagnetic Relay 3A), 1 RS485 communication Interface,	ZR1PB00P7

1 USB interface (Type C).	
Economic type 256KB program space, 92KB data space, 220VAC power supply, no display screen, not supporting extension modules. 16DI (220VAC), 8DO (Electromagnetic Relay 3A), 1 RS485 communication Interface, 1 USB interface (Type C).	ZR1PA00P7
Economic type 256KB program space, 92KB data space, 24VDC power supply, no display screen, not supporting extension modules. 8DI (24VDC), 4DO (Electromagnetic Relay 3A), 1 RS485 communication Interface, 1 USB interface (Type C).	ZR1PB00BD
Economic type 256KB program space, 92KB data space, 24VDC power supply, no display screen, not supporting extension modules. 16DI (24VDC), 8DO (Electromagnetic Relay 3A), 1 RS485 communication Interface, 1 USB interface (Type C).	ZR1PA00BD
Economic type 256KB program space, 92KB data space, 24VDC power supply, no display screen, not supporting extension modules. 6DI (24VDC), 4DO (Electromagnetic Relay 3A), 2AI(Voltage signal: 0-10VDC, Current signal: 4-20mA/0-20mA, Resistance signal: 0-30kΩ, support NTC_10K (3950) or PT1000 PT100), 1 RS485 communication Interface, 1 USB interface (Type C).	ZR1PP00BD2A
Economic type 256KB program space, 92KB data space, 24VDC power supply, no display screen, not supporting extension modules. 12DI (24VDC), 8DO (Electromagnetic Relay 3A), 4 AI (Voltage signal: 0-10VDC, Current signal: 4-20mA/0-20mA, Resistance signal: 0-30kΩ, support NTC_10K (3950) or PT1000 PT100), 1 RS485 communication Interface, 1 USB interface (Type C).	ZR1PP00BD4A
Extension module	
Mixed IO module, 24 VDC power supply; 8DI（24VDC）, 8DO (Electromagnetic Relay 3A), 8 AI (Voltage signal: 0-10VDC, Current signal: 4-20mA/0-20mA, Resistance signal: 0-30kΩ, support NTC_10K (3950) or PT1000 PT100) 4AO (0~10 V, 4~20 mA).	SM172EMIO2800

Mixed IO module, 24 VDC power supply; 4 DI(24VDC), 2 DO (Electromagnetic Relay 3A), 2 AI (Voltage signal: 0-10VDC, Current signal: 4-20mA/0-20mA, Resistance signal: 0-30kΩ, support NTC_10K (3950) or PT1000 PT100)	SM172EMIO1000
Digital IO module; 24 VDC power supply; 28 IO: 16 DI(24VDC), 12 DO (Electromagnetic Relay 3A)	SM172EDM2800
Digital IO module; 24 VDC power supply; 16 IO: 8 DI(24VDC), 8 DO (Electromagnetic Relay 3A)	SM172EDM1600
Digital IO module; 24 VDC power supply; 8 IO: 4 DI(24VDC), 4 DO (Electromagnetic Relay 3A)	SM172EDM0800
Digital IO module; 24 VDC power supply; 8 IO: 4 DI(24VDC) , 4 DO (transistor drain output 0.5A)	SM172EDM0810
Digital IO module; 220 VAC power supply; 8 IO: 4 DI(24VDC), 4 DO (Electromagnetic Relay 3A)	SM172EDM0800 P7
Analog IO module; 24 VDC power supply; 8 IO: 4 AI (Voltage signal: 0-10VDC, Current signal: 4-20mA/0-20mA, Resistance signal: 0-30kΩ, support NTC_10K (3950) or PT1000 PT100), 4 AO (0-10 V, 4-20 mA)	SM172EAM0800
Extension cable for IO modules (length 1000 mm)	SM172C1000

Social networks

systemeelectric_official

youtube.com/c/SystemeElectric

vk.com/Systemeelectric

Systeme Electric

More about the company

www.systeme.ru

Our brands

Systême electric

Dēkraft

Механотроника

Systême soft

IA-MAN-SYSPLC-SW-25

08/2026