RS OEMax

# X8 Instruction Manual

## Reference Manual

Catalog Number(s): X8-M14DDT, X8-M16DDR, X8-M32DDT

RS Reliable & Smart Automation

# Important User Information

Solid state equipment has operational characteristics differing from those of electromechanical equipment. Because of this difference, and also because of the wide variety of uses for solid state equipment, all persons responsible for applying this equipment must satisfy themselves that each intended application of this equipment is acceptable.

In no event will RS Automation Co., Ltd. be responsible or liable for indirect or consequential damages resulting from the use or application of this equipment.

The examples and diagrams in this manual are included solely for illustrative purposes. Because of the many variables and requirements associated with any particular installation, RS Automation Co., Ltd. cannot assume responsibility or liability for actual use based on the examples and diagrams.

No patent liability is assumed by RS Automation Co., Ltd. with respect to use of information, circuits, equipment, or software described in this manual.



Throughout this manual, when necessary, we use notes to make you aware of safety considerations.

| | |
|---|---|
| **WARNING** | Identifies information about practices or circumstances that can cause an explosion in a hazardous environment, which may lead to personal injury or death, property damage, or economic loss. |
| **IMPORTANT** | Identifies information that is critical for successful application and understanding of the product. |
| **ATTENTION** | Identifies information about practices or circumstances that can lead to personal injury or death, property damage, or economic loss. Attentions help you identify a hazard, avoid a hazard, and recognize the consequence. |

Read this preface to familiarize yourself with the rest of the manual.

## Who Should Use this Manual

This manual is intended for engineers or technicians directly involved in the programming of the **X8 series PLC**, and programmers directly involved in the operation, field maintenance, and integration of the system with the **X8 series PLC**.

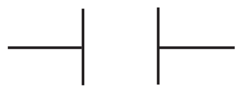
If you do not have a basic understanding of the **X8 series PLC**, contact your local RS Automation sales representative before using this product, for information on available training courses.

## How to Use This Manual

### Table for Instruction Setting

This manual uses the following table for parameter description.

**Table 0.1 Table for instruction Setting**

| | |
|---|---|
| Symbol | X 1. 3. 0.0  |
| Instruction Name | No |
| Description | Examine if set |
| Parameter | Examine Bit 3 of word 0 in the Slot 3<br>ON : True , Off : False |
| Initial Value | |
| Applicable Mode | |
| Others | |

## Related Documents

The following documents contain additional information concerning related X8 series PLC.
http://www.rsautomation.biz.

| Description | Manual Name | Publication |
|---|---|---|
| X8 Series PLC installation | X8 Series PLC Installataion Instuructions | X8-IN001A |
| X8 IO module X8 Base (CPU) module. | X8 Series PLC User Manual | X8-UM001A |

# Safety Instructions

Please read this manual and the related documentation thoroughly and familiarize yourself with the directions before installing, operating, performing inspection and preventive maintenance. Make sure to follow the directions correctly to ensure normal operation of the product and your safety.

## Environment and Enclosure

**ATTENTION**

- This equipment is intended for use in a Pollution Degree 2 industrial environment, in overvoltage Category II applications (as defined in IEC 60664-1), at altitudes up to 2000 m (6562 ft) without derating.
- This equipment is considered Group 1, Class A industrial equipment according to IEC/CISPR 11.Without appropriate precautions, there may be difficulties with electromagnetic compatibility in residential and other environments due to conducted and radiated disturbances.
- It must be mounted within an enclosure that is suitably designed for those specific environmental conditions that will be present and appropriately designed to prevent personal injury resulting from accessibility to live parts. The enclosure must have suitable flame-retardant properties to prevent or minimize the spread of flame, complying with a flame spread rating of 5VA, V2, V1, V0 (or equivalent) if non-metallic. The interior of the enclosure must be accessible only by the use of a tool. Subsequent sections of this publication may contain additional information regarding specific enclosure type ratings that are required to comply with certain product safety certifications.
- NEMA Standard 250 and IEC 60529, as applicable, for explanations of the degrees of protection provided by different types of enclosure.

## Preventing Electrostatic Discharge

This equipment is sensitive to electrostatic discharge, which can cause internal damage and affect normal operation.

Follow these guidelines when you handle this equipment.

**WARNING**

- If you connect or disconnect the serial cable with power applied to this module or the serial device on the other end of the cable, an electrical arc can occur. This could cause an explosion in hazardous location installations. Be sure that power is removed or the area is nonhazardous before proceeding.
- The local programming terminal port is intended for temporary use only and must not be connected or disconnected unless the area is assured to be nonhazardous.
- The USB port is intended for temporary local programming purpose only and not intended for permanent connection. If you connect or disconnect the USB cable with power applied to this module or any device on the USB network, an electrical arc can occur. This could cause an explosion in hazardous location installation. Be sure that power is removed or the area in nonhazardous before proceeding.
- Exposure to some chemicals may degrade the sealing properties of materials used in the Relays. It is recommended that the User periodically inspect these devices for any degradation of properties and replace the module if degradation is found.
- If you insert or remove the plug-in module while main power is on, an electrical arc can occur. This could cause an explosion in hazardous location installation. Be sure that power is removed or the area is nonhazardous before proceeding.
- When you connect or disconnect the Removable Terminal Block (RTB) with field side power applied, an electrical arc can occur. This could cause an explosion in hazardous location installations. Be sure that power is removed or the area is nonhazardous before proceeding.

**ATTENTION**

- Touch a grounded object to discharge potential static.
- Wear an approved grounding wrist strap.
- Do not touch connectors or pins on component boards.
- Do not touch circuit components inside the equipment.
- Use a static-safe workstation, if available.
- Store the equipment in appropriate static-safe packaging when not in use.
- The USB is not to exceed 3.0 m (9.84 ft).
- Do not wire more than 2 conductors on any single terminal.
- Do not remove the Removable Terminal Block (RTB) until power is removed.
- Electrostatic discharge can damage semiconductor devices inside the module. Do not touch the connector pins or other sensitive area.
- To comply with the CE Low Voltage Directive (LVD), this equipment must be powered from a source compliant with the following: Safety Extra Low Voltage (SELV) or Protected Extra Low Voltage (PELV).
- To comply with UL restrictions, this equipment must be powered from a Class 2 source.
- Do not remove the protective debris strips until after the controller and all other equipment in the panel near the module are mounted and wired. Remove strips before operating the controller. Failure to remove strips before operating can cause overheating.
- Be careful when stripping wires. Wire fragments that fall into the controller could cause damage. Once wiring is complete, make sure the controller is free of all metal fragments.

## Preface



## Chapter 1 IO Configuration

**Chapter 6 Using High-Speed Outputs**

## Chapter 7 Relay-Type (Bit) Instructions



## Chapter 8 Timer and Counter Instructions

## Chapter 9 Compare Instructions



## Chapter 10 Math Instructions



## Chapter 11 Conversion Instructions



## Chapter 12 Logical Instructions

## Chapter 13 Move Instructions



## Chapter 14 Data Table Instructions



## Chapter 15 Shift and FIFO_LIFO Instructions



## Chapter 16 Sequencer Instructions



## Chapter 17 Program Control Instructions



## Chapter 18 Input/Output Instructions



## Chapter 19 Using Interrupts

## Chapter 20 PID Control



## Chapter 21 ASCII Instructions

## Chapter 22 Communication Instructions

# I/O Configuration

## Introduction

This chapter describes the general matters of Input and Output features of the **X8 Series PLC**. Each controller comes with a certain amount of embedded I/O, which is physically located on the controller. The controller also allows for adding expansion I/O.

## Embedded I/O

The **X8 Series PLC** provide basic I/O that is built into the controller as listed in the following table. These I/O points are referred to as Embedded I/O.

| Catalog No. | Description | | | |
|---|---|---|---|---|
| | Input Power | Type | Embedded Digital I/O | Comm. Ports |
| X8-M14DDT | 24VDC | Modular | 8 In , 6 Sink out | 2x RS-232/RS-485<br>1x Ethernet<br>1x USB |
| X8-M16DDR | 24VDC | Modular | 8 In , 8 Relay out, HSC 2CH | |
| X8-M32DDT | 24VDC | Modular | 16 In , 16 Sink out | |
| X8-B48DDX[1)] | 24VDC | Brick | 28 In , 20 Sink out | |
| X8-B24ADR[1)] | 110/220V AC | Brick | 14 In , 10 Relay out | |
| X8-B24DDT[1)] | 24VDC | Brick | 14 In , 10 Sink out | |
| X8-B24AAR[1)] | 110/220V AC | Brick | 14 AC In, 10 Relay out | |
| Specifications and designs are subject to change without prior notice. | | | | |

1) It will be released in 2013

## Expantion I/O

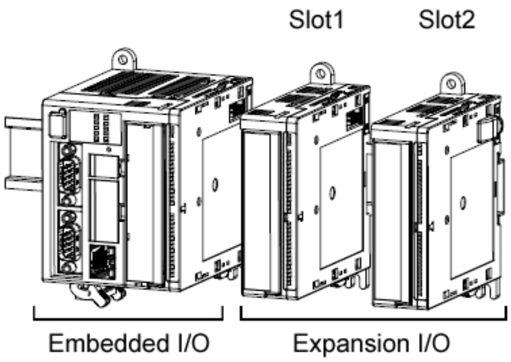
If the application requires more I/O than the controller provides, you can attach I/O modules using serial techniques. **X8 Series PLC** have I/O expandability.

### X8 Series PLC Expansion I/O

For the **X8 Series PLC** expansion I/O is used to provide special module such as digital I/O, analog module, position module and communication module. You can attach up to max. 80 expansion I/O modules with power supply for each 12 I/O modules in any combination.

### Addressing Expansion I/O Slots

The figure below shows the addressing for the base and expantion I/O module of **X8 Series PLC** . The expansion I/O is addressed as slots 1 to 80. Modules are counted from left to right as shown below.




**TIP**

In most cases, you can use the following address format:

X:s.b (X = Table type letter, s = slot number, b = bit number)

For example I3:0.0

# X8 Series Embedded I/O Memory Mapping

## Digital I/O Configuration of X8 Series PLC base module

### X8-M14DDT, X8-M16DDR Input Image

For X8-M14DDT and X8-M16DDR base module, the Bit positions 0 to 7 correspond to input terminals 0 to 7.

| Word | Bit Position | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | **15** | **14** | **13** | **12** | **11** | **10** | **9** | **8** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| **0** | x | x | x | x | x | x | x | x | r | r | r | r | r | r | r | r |

r = read only, x = not used, always at a 0 or OFF state

### X8-M32DDT Input Image

For X8-M32DDT base module, the Bit positions 0 to 15 together with word 0/1 correspond to input terminals 0 to 15.

| Word | Bit Position | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | **15** | **14** | **13** | **12** | **11** | **10** | **9** | **8** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| **0** | r | r | r | r | r | r | r | r | r | r | r | r | r | r | r | r |
| **1** | r | r | r | r | r | r | r | r | r | r | r | r | r | r | r | r |

r = read only, x = not used, always at a 0 or OFF state

## X8-B48DDX Input Image

For X8-B48DDX base module, the Bit positions 0 to 15 with word 0 correspond to input terminals 0 to 15. And Bit position 0 to 11 with word 1 correspond to input terminal 0 to 11.

| Word | Bit Position | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | **15** | **14** | **13** | **12** | **11** | **10** | **9** | **8** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| **0** | r | r | r | r | r | r | r | r | r | r | r | r | r | r | r | r |
| **1** | x | x | x | x | r | r | r | r | r | r | r | r | r | r | r | r |

r = read only, x = not used, always at a 0 or OFF state

it will be released in 2013.

## X8-B24ADR, X8-B24DDT and X8-B24AAR Input Image

For X8-B24ADR, X8-B24DDT and X8-B24AAR base module, the Bit positions 0 to 13 correspond to input terminals 0 to 13.

| Word | Bit Position | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | **15** | **14** | **13** | **12** | **11** | **10** | **9** | **8** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| **0** | x | x | r | r | r | r | r | r | r | r | r | r | r | r | r | r |

r = read only, x = not used, always at a 0 or OFF state

## X8-M14DDT Output Image

For *X8-M14DDT* base module, the Bit positions 0 to 5 together with word 0 correspond to output terminals 0 to 5.

| Word | Bit Position | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| 0 | x | x | x | x | x | x | x | x | x | x | w | w | w | w | w | w |

w= write only, x = not used, always at a 0 or OFF state

## X8-M16DDR Output Image

For X8-M16DDR base module, the Bit positions 0 to 7 correspond to output terminals 0 to 7.

| Word | Bit Position | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| 0 | x | x | x | x | x | x | x | x | w | w | w | w | w | w | w | w |

w= write only, x = not used, always at a 0 or OFF state

## X8-M32DDT Output Image

For X8-M32DDT base module, the Bit positions 0 to15 together with word 0/1 correspond to output terminals 0 to 15.

| Word | Bit Position | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| 0 | w | w | w | w | w | w | w | w | w | w | w | w | w | w | w | w |
| 1 | w | w | w | w | w | w | w | w | w | w | w | w | w | w | w | w |

w= write only, x = not used, always at a 0 or OFF state

## X8-B48DDX Output Image

For X8-B48DDX base module, the Bit positions 0 to 15 with word 0 correspond to output terminals 0 to 15. And the Bit position 0 to 3 with word 1 correspond to output terminals 0 to 3.

| Word | Bit Position | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| 0 | w | w | w | w | w | w | w | w | w | w | w | w | w | w | w | w |
| 1 | x | x | x | x | x | x | x | x | x | x | x | x | w | w | w | w |

w= write only, x = not used, always at a 0 or OFF state

### X8-B24ADR, X8-B24DDT and X8-B24AAR Output Image

For X8-B24ADR, X8-B24DDT and X8-B24AAR output module, the Bit positions 0 to 13 correspond to output terminals 0 to 13.

| Word | Bit Position | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| 0 | x | x | w | w | w | w | w | w | w | w | w | w | w | w | w | w |

w= write only, x = not used, always at a 0 or OFF state

# X8 Series Expansion I/O Memory Mapping

## Digital I/O Input Module Configuration

### X8-XU16, X8-XA16 Input Image

16-point input module has a total area of the 16-bit from bit 0 to 15 with one word.

| Word | Bit Position | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| 0 | r | r | r | r | r | r | r | r | r | r | r | r | r | r | r | r |

r = read only, x = not used, always at a 0 or OFF state

### X8-XU32 Input Image

32-points input module has a total area of the 32-bit from bit 0 to 15 with two words.

| Word | Bit Position | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| 0 | r | r | r | r | r | r | r | r | r | r | r | r | r | r | r | r |
| 1 | r | r | r | r | r | r | r | r | r | r | r | r | r | r | r | r |

r = read only, x = not used, always at a 0 or OFF state

* it will be released in 2013.

## X8-XU64 Input Image

64-point input module has a total area of the 64-bit from bit 0 to 15 with four words.

| Word | Bit Position | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| 0 | r | r | r | r | r | r | r | r | r | r | r | r | r | r | r | r |
| 1 | r | r | r | r | r | r | r | r | r | r | r | r | r | r | r | r |
| 2 | r | r | r | r | r | r | r | r | r | r | r | r | r | r | r | r |
| 3 | r | r | r | r | r | r | r | r | r | r | r | r | r | r | r | r |

r = read only, x = not used, always at a 0 or OFF state

## X8-TPOT8 Input Image

TPOT8 input module has a total area of the 8-bit from bit 0 to 7with one word.

| Word | Bit Position | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| 0 | x | x | x | x | x | x | x | x | r | r | r | r | r | r | r | r |

r = read only, x = not used, always at a 0 or OFF state

* it will be released in 2013.

## Digital I/O Output Module Configuration

### X8-YN16, X8-YR16 Output Image

16-point output module has a total area of the 16-bit from bit 0 to 15 with one word.

| Word | Bit Position | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| 0 | w | w | w | w | w | w | w | w | w | w | w | w | w | w | w | w |

w= write only, x = not used, always at a 0 or OFF state

### X8-YN32 Input Image

32-point output module has a total area of the 32-bit from bit 0 to 15 with two words.

| Word | Bit Position | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| 0 | w | w | w | w | w | w | w | w | w | w | w | w | w | w | w | w |
| 1 | w | w | w | w | w | w | w | w | w | w | w | w | w | w | w | w |

w= write only, x = not used, always at a 0 or OFF state

### X8-YN64 Input Image

64-point output module has a total area of the 64-bit from bit 0 to 15 with four words.

| Word | Bit Position | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| 0 | w | w | w | w | w | w | w | w | w | w | w | w | w | w | w | w |
| 1 | w | w | w | w | w | w | w | w | w | w | w | w | w | w | w | w |
| 2 | w | w | w | w | w | w | w | w | w | w | w | w | w | w | w | w |
| 3 | w | w | w | w | w | w | w | w | w | w | w | w | w | w | w | w |

w= write only, x = not used, always at a 0 or OFF state

It will be released on 2013.

### X8-YR6C Output Image

YR6C output module has a total area of the 6-bit from bit 0 to 5 with one word.

| Word | Bit Position | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| 0 | x | x | x | x | x | x | x | x | x | x | w | w | w | w | w | w |

w= write only, x = not used, always at a 0 or OFF state

## Analog I/O Module Configuration

Analog module has different memory configuration area with digital I/O module. It has separate parts that outputs data input from actual each channel , shows the state of each channel, and configuration area allows set the each channel's operation mode.

### X8-AI8 Input Image

X8-AI8 module is the high-end expantion analog input module with 8-channel voltage and current input. A total of 12 words are assigned to the area that shows the state and the input of data, and the control area is allocated for each set of channel x word.

**Data Address Mapping**

| Word | Bit Position | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | **15** | **14** | **13** | **12** | **11** | **10** | **9** | **8** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| **0** | Input Channel 0 , Read Only | | | | | | | | | | | | | | | |
| **1** | Input Channel 1 , Read Only | | | | | | | | | | | | | | | |
| **2** | Input Channel 2 , Read Only | | | | | | | | | | | | | | | |
| **3** | Input Channel 3 , Read Only | | | | | | | | | | | | | | | |
| **4** | Input Channel 4 , Read Only | | | | | | | | | | | | | | | |
| **5** | Input Channel 5 , Read Only | | | | | | | | | | | | | | | |
| **6** | Input Channel 6 , Read Only | | | | | | | | | | | | | | | |
| **7** | Input Channel 7 , Read Only | | | | | | | | | | | | | | | |
| **8** | | | | | | | | | S7 | S6 | S5 | S4 | S3 | S2 | S1 | S0 |
| **9** | | | | | | | | | 07 | 07 | 07 | 07 | 07 | 07 | 07 | 07 |
| **10** | | | | | | | | | | | | | | | | |
| **11** | | | | | | | | | | | | | | | | |
| **12** | | | | | | | | | | | | | | | | |
| **13** | Firmware Rev Major | | | | | | | | Firmware Rev Minor | | | | | | | |

w= write only, x = not used, always at a 0 or OFF state

**Contorl Area Address Mapping**

| Word | Bit Position | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | **15** | **14** | **13** | **12** | **11** | **10** | **9** | **8** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| **0** | Input Channel 0 Data | | | | | | | | | | | | | | | |
| **1** | Input Channel 1 Data | | | | | | | | | | | | | | | |
| **2** | Input Channel 2Data | | | | | | | | | | | | | | | |
| **3** | Input Channel 3 Data | | | | | | | | | | | | | | | |
| **4** | Input Channel 4 Data | | | | | | | | | | | | | | | |
| **5** | Input Channel 5 Data | | | | | | | | | | | | | | | |
| **6** | Input Channel 6 Data | | | | | | | | | | | | | | | |
| **7** | Input Channel 7 Data | | | | | | | | | | | | | | | |

w= write only, x = not used, always at a 0 or OFF state

- User Scale Data type can be specified using the SCALE instruction.

## Data Configuration of Control Area

### Display Format

| Bit Position | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **15** | **14** | **13** | **12** | **11** | **10** | **9** | **8** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| | | | | | | | | | | | | 0 | 0 | 0 | 0 |
| | | | | | | | | | | | | 0 | 0 | 0 | 1 |
| | | | | | | | | | | | | 0 | 0 | 1 | 0 |
| | | | | | | | | | | | | 0 | 0 | 1 | 1 |
| | | | | | | | | | | | | 0 | 1 | 0 | 0 |
| | | | | | | | | | | | | 0 | 1 | 0 | 1 |

0 (0000): Raw Data
1 (0001): Left Align
2 (0010) : Voltage Value
3 (0011) : Percent Value
4 (0100) : Scaled for PID
5 (0101) : User Scale

### Filter Frequency Format

| Bit Position | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **15** | **14** | **13** | **12** | **11** | **10** | **9** | **8** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| | | | | | | | | 0 | 0 | 0 | 0 | | | | |
| | | | | | | | | 0 | 0 | 0 | 1 | | | | |
| | | | | | | | | 0 | 0 | 1 | 0 | | | | |
| | | | | | | | | 0 | 0 | 1 | 1 | | | | |
| | | | | | | | | 0 | 1 | 0 | 0 | | | | |
| | | | | | | | | 0 | 1 | 0 | 1 | | | | |

0(0000) : No Filter
1(0001) : 1 KHz
2(0010) : 500 Hz
3(0011) : 100 Hz
4(0100) : 50 Hz
5(0101) : 10 Hz

### Reserved

| Bit Position | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **15** | **14** | **13** | **12** | **11** | **10** | **9** | **8** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| | | | | Reserved | | | | | | | | | | | |

- Reserved for future use.

**Input Data Type Selection**

| Bit Position | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| 0 | 0 | 0 | 0 | | | | | | | | | | | | |
| 0 | 0 | 0 | 1 | | | | | | | | | | | | |
| 0 | 0 | 1 | 0 | | | | | | | | | | | | |
| 0 | 0 | 1 | 1 | | | | | | | | | | | | |
| 0 | 1 | 0 | 0 | | | | | | | | | | | | |
| 0 | 1 | 0 | 1 | | | | | | | | | | | | |

0 (0000): Voltage Input 0~5V
1 (0001) : Voltage Input 0~10V
2 (0010) : Voltage Input -5~5V
3 (0011) : Voltage Input -10~10V
4 (0100) : Current Input 0~20mA
5 (0101): Current Input 4~20mA

## X8-AO4 Input Image

X8-AO4 module is the high-end expantion analog input module with 4-channel voltage and current input. A total of 6 words are assigned to the area that shows the state and the input of data, and the control area is allocated for each set of channel x word.

**Data Address Mapping**

| | Bit Position | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| **0** | Input Channel 0 | | | | | | | | | | | | | | | |
| **1** | Input Channel 1 | | | | | | | | | | | | | | | |
| **2** | Input Channel 2 | | | | | | | | | | | | | | | |
| **3** | Input Channel 3 | | | | | | | | | | | | | | | |

w= write only, x = not used, always at a 0 or OFF state

### Status Output Mapping by Channel

| Word | Bit Position | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| 0 | Reserved | | | | | | | | | | | | S3 | S2 | S1 | S0 |
| 1 | Reserved | | | | | | | | | | | | F3 | F2 | F1 | F0 |
| 2 | Reserved | | | | | | | | | | | | | | | |
| 3 | Firmware Rev major | | | | | | | | Firmware Rev minor | | | | | | | |

- S0 ~ S4: Ready Status (Normal : 0, Reset/Error:0)
- F0 ~ F3: Fault Status: (Fault: 1, Normal: 0).

### Control Area Address Mapping

| Word | Bit Position | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| 0 | Configuration Channel 0 Data | | | | | | | | | | | | | | | |
| 1 | Configuration Channel 1 Data | | | | | | | | | | | | | | | |
| 2 | Configuration Channel 2 Data | | | | | | | | | | | | | | | |
| 3 | Configuration Channel 3 Data | | | | | | | | | | | | | | | |
| 4 | Reserved | | | | | | | | | | | | | | | |
| 5 | Reserved | | | | | | | | | | | | | | | |
| 6 | Reserved | | | | | | | | | | | | | | | |
| 7 | Reserved | | | | | | | | | | | | | | | |

w= write only, x = not used, always at a 0 or OFF state

- User Scale Data type can be specified using the SCALE instruction.

### Display Format

| Bit Position | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | | | | | | | | | | | | 0 | 0 | 0 | 0 |
| | | | | | | | | | | | | 0 | 0 | 0 | 1 |
| | | | | | | | | | | | | 0 | 0 | 1 | 0 |
| | | | | | | | | | | | | 0 | 0 | 1 | 1 |
| | | | | | | | | | | | | 0 | 1 | 0 | 0 |
| | | | | | | | | | | | | 0 | 1 | 0 | 1 |

0 (0000): Raw Data
1 (0001): Left Align
2 (0010) : Voltage Value
3 (0011) : Percent Value
4 (0100) : Scaled for PID
5 (0101) : User Scale

**Reserved**

| Bit Position | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | | | | | | | | Reserved | | | | | | | |

- Reserved for future use.

**Reserved**

| Bit Position | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | | | | Reserved | | | | | | | | | | | |

- Reserved for future use.

**Input Data Type Selection**

| Bit Position | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| 0 | 0 | 0 | 0 | | | | | | | | | | | | |
| 0 | 0 | 0 | 1 | | | | | | | | | | | | |
| 0 | 0 | 1 | 0 | | | | | | | | | | | | |
| 0 | 0 | 1 | 1 | | | | | | | | | | | | |
| 0 | 1 | 0 | 0 | | | | | | | | | | | | |
| 0 | 1 | 0 | 1 | | | | | | | | | | | | |

0 (0000): Voltage Input 0~5V
1 (0001) : Voltage Input 0~10V
2 (0010) : Voltage Input -5~5V
3 (0011) : Voltage Input -10~10V
4 (0100) : Current Input 0~20mA
5 (0101): Current Input 4~20mA

### *X8-RT6 Input Image*

X8-RT6 module is the high-end expantion RTD input module with 6-channel RTD input. A total of 6 words are assigned to the area that shows the state and the input of data, and 8 words are allocated to the control area to set each channel.

**Data Address Mapping**

<table>
<tr><th rowspan="2">Word</th><th colspan="16">Bit Position</th></tr>
<tr><th>15</th><th>14</th><th>13</th><th>12</th><th>11</th><th>10</th><th>9</th><th>8</th><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td>0</td><td colspan="16">Channel 0</td></tr>
<tr><td>1</td><td colspan="16">Channel 1</td></tr>
<tr><td>2</td><td colspan="16">Channel 2</td></tr>
<tr><td>3</td><td colspan="16">Channel 3</td></tr>
<tr><td>4</td><td colspan="16">Channel 4</td></tr>
<tr><td>5</td><td colspan="16">Channel 5</td></tr>
</table>

**Control Area Address Mapping**

<table>
<tr><th rowspan="2">Word</th><th colspan="16">Bit Position</th></tr>
<tr><th>15</th><th>14</th><th>13</th><th>12</th><th>11</th><th>10</th><th>9</th><th>8</th><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td>0</td><td colspan="16">Channel 0</td></tr>
<tr><td>1</td><td colspan="16">Channel 1</td></tr>
<tr><td>2</td><td colspan="16">Channel 2</td></tr>
<tr><td>3</td><td colspan="16">Channel 3</td></tr>
<tr><td>4</td><td colspan="16">Channel 4</td></tr>
<tr><td>5</td><td colspan="16">Channel 5</td></tr>
<tr><td>6</td><td colspan="16">Reserved</td></tr>
<tr><td>7</td><td colspan="16">Reserved</td></tr>
</table>

**Display Format**

<table>
<tr><th colspan="16">Bit Position</th></tr>
<tr><th>15</th><th>14</th><th>13</th><th>12</th><th>11</th><th>10</th><th>9</th><th>8</th><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td colspan="12"></td><td>0</td><td>0</td><td>0</td><td>0</td></tr>
<tr><td colspan="12"></td><td>0</td><td>0</td><td>0</td><td>1</td></tr>
<tr><td colspan="12"></td><td>0</td><td>0</td><td>1</td><td>0</td></tr>
</table>

0 (0000): Raw Data
1 (0001): Left Align
2 (0010) : Voltage Value

### Filter Frequency Format

| Bit Position | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **15** | **14** | **13** | **12** | **11** | **10** | **9** | **8** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| | | | | | | | | 0 | 0 | 0 | 0 | | | | |
| | | | | | | | | 0 | 0 | 0 | 1 | | | | |
| | | | | | | | | 0 | 0 | 1 | 0 | | | | |
| | | | | | | | | 0 | 0 | 1 | 1 | | | | |
| | | | | | | | | 0 | 1 | 0 | 0 | | | | |
| | | | | | | | | 0 | 1 | 0 | 1 | | | | |
| | | | | | | | | 0 | 1 | 1 | 0 | | | | |

0(0000) : No Filter
1(0001) : 1 KHz
2(0010) : 500 Hz
3(0011) : 100 Hz
4(0100) : 60 Hz
5(0101) : 50 Hz
6(0110) : 10 Hz

### Enable

| Bit Position | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **15** | **14** | **13** | **12** | **11** | **10** | **9** | **8** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| | | | | | | | 0 | | | | | | | | |
| | | | | | | | 1 | | | | | | | | |

0: Disable
1: Enable

### Temperature Unit

| Bit Position | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **15** | **14** | **13** | **12** | **11** | **10** | **9** | **8** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| | | | | | | 0 | | | | | | | | | |
| | | | | | | 1 | | | | | | | | | |

0 : °F
1 : °C

**Open Circuit**

| Bit Position | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **15** | **14** | **13** | **12** | **11** | **10** | **9** | **8** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| | | | | 0 | 0 | | | | | | | | | | |
| | | | | 0 | 1 | | | | | | | | | | |
| | | | | 1 | 0 | | | | | | | | | | |
| | | | | 1 | 1 | | | | | | | | | | |

0 (00): Up Scale
1 (01): Down Scale
2 (10) : Last State
3 (11) : Zero

**Data Input Type**

| Bit Position | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **15** | **14** | **13** | **12** | **11** | **10** | **9** | **8** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| 0 | 0 | 0 | 0 | | | | | | | | | | | | |
| 0 | 0 | 0 | 1 | | | | | | | | | | | | |
| 0 | 0 | 1 | 0 | | | | | | | | | | | | |
| 0 | 0 | 1 | 1 | | | | | | | | | | | | |
| 0 | 1 | 0 | 0 | | | | | | | | | | | | |
| 0 | 1 | 0 | 1 | | | | | | | | | | | | |
| 0 | 1 | 1 | 0 | | | | | | | | | | | | |
| 0 | 1 | 1 | 1 | | | | | | | | | | | | |
| 1 | 0 | 0 | 0 | | | | | | | | | | | | |
| 1 | 0 | 0 | 1 | | | | | | | | | | | | |
| 1 | 0 | 1 | 0 | | | | | | | | | | | | |
| 1 | 1 | 0 | 0 | | | | | | | | | | | | |
| 1 | 1 | 0 | 1 | | | | | | | | | | | | |
| 1 | 1 | 1 | 0 | | | | | | | | | | | | |

0 (0000): 100 Ω Pt385
1 (0001): 200 Ω Pt385
2 (0010): 500 Ω Pt385
3 (0011): 1000 Ω Pt385
4 (0100): 100 Ω Pt3916
5 (0101): 200 Ω Pt3916
6 (0110): 500 Ω Pt3916
7 (0111): 1000 Ω Pt3916
8 (1000): 10 Ω Cu 426

9 (1001): 120 Ω Ni 618
11 (1011): 120 Ω Ni 672
12 (1100): 1~2000 Ω
13 (1101): 1~327 Ω
14 (1110): 1~1200 Ω

## *X8-TC6 Input Image*

X8-TC6 module is the high-end expantion TC input module with 6-channel TC input. A total of 6 words are assigned to the area that shows the state and the input of data, and 8 words are allocated to the control area to set each channel .

**Data Address Mapping**

| Word | Bit Position | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| 0 | Channel 0 | | | | | | | | | | | | | | | |
| 1 | Channel 1 | | | | | | | | | | | | | | | |
| 2 | Channel 2 | | | | | | | | | | | | | | | |
| 3 | Channel 3 | | | | | | | | | | | | | | | |
| 4 | Channel 4 | | | | | | | | | | | | | | | |
| 5 | Channel 5 | | | | | | | | | | | | | | | |

**Control Area Address Mapping**

| Word | Bit Position | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| 0 | Channel 0 | | | | | | | | | | | | | | | |
| 1 | Channel 1 | | | | | | | | | | | | | | | |
| 2 | Channel 2 | | | | | | | | | | | | | | | |
| 3 | Channel 3 | | | | | | | | | | | | | | | |
| 4 | Channel 4 | | | | | | | | | | | | | | | |
| 5 | Channel 5 | | | | | | | | | | | | | | | |
| 6 | Reserved | | | | | | | | | | | | | | | |
| 7 | Reserved | | | | | | | | | | | | | | | |

**Display Format**

| Bit Position | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **15** | **14** | **13** | **12** | **11** | **10** | **9** | **8** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| | | | | | | | | | | | | 0 | 0 | 0 | 0 |
| | | | | | | | | | | | | 0 | 0 | 0 | 1 |
| | | | | | | | | | | | | 0 | 0 | 1 | 0 |

0 (0000): Raw Data
1 (0001): Percent Value
2 (0010) : Scaled for PID

**Filter Frequency Format**

| Bit Position | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **15** | **14** | **13** | **12** | **11** | **10** | **9** | **8** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| | | | | | | | | 0 | 0 | 0 | 0 | | | | |
| | | | | | | | | 0 | 0 | 0 | 1 | | | | |
| | | | | | | | | 0 | 0 | 1 | 0 | | | | |
| | | | | | | | | 0 | 0 | 1 | 1 | | | | |
| | | | | | | | | 0 | 1 | 0 | 0 | | | | |
| | | | | | | | | 0 | 1 | 0 | 1 | | | | |
| | | | | | | | | 0 | 1 | 1 | 0 | | | | |

0 (0000): No Filter

1 (0001): 1KHz

2 (0010) : 500Hz

3 (0011) : 100Hz

4 (0100) : 60Hz

5 (0101) : 50Hz

6 (0110) : 10Hz

**Enable**

| Bit Position | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **15** | **14** | **13** | **12** | **11** | **10** | **9** | **8** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| | | | | | | | 0 | | | | | | | | |
| | | | | | | | 1 | | | | | | | | |

0: Disable
1: Enable

**Temperature Unit**

| Bit Position | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **15** | **14** | **13** | **12** | **11** | **10** | **9** | **8** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| | | | | | | 0 | | | | | | | | | |
| | | | | | | 1 | | | | | | | | | |

0 : ℉
1 : ℃

**Open Circuit**

| Bit Position | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **15** | **14** | **13** | **12** | **11** | **10** | **9** | **8** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| | | | | 0 | 0 | | | | | | | | | | |
| | | | | 0 | 1 | | | | | | | | | | |
| | | | | 1 | 0 | | | | | | | | | | |
| | | | | 1 | 1 | | | | | | | | | | |

0 (00): Up Scale
1 (01): Down Scale
2 (10) : Last State
3 (11) : Zero

**Data Input Type**

| Bit Position | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **15** | **14** | **13** | **12** | **11** | **10** | **9** | **8** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| 0 | 0 | 0 | 0 | | | | | | | | | | | | |
| 0 | 0 | 0 | 1 | | | | | | | | | | | | |
| 0 | 0 | 1 | 0 | | | | | | | | | | | | |
| 0 | 0 | 1 | 1 | | | | | | | | | | | | |
| 0 | 1 | 0 | 0 | | | | | | | | | | | | |
| 0 | 1 | 0 | 1 | | | | | | | | | | | | |
| 0 | 1 | 1 | 0 | | | | | | | | | | | | |
| 0 | 1 | 1 | 1 | | | | | | | | | | | | |
| 1 | 0 | 0 | 0 | | | | | | | | | | | | |
| 1 | 0 | 0 | 1 | | | | | | | | | | | | |
| 1 | 0 | 1 | 0 | | | | | | | | | | | | |
| 1 | 1 | 0 | 0 | | | | | | | | | | | | |
| 1 | 1 | 0 | 1 | | | | | | | | | | | | |
| 1 | 1 | 1 | 0 | | | | | | | | | | | | |

0 (0000): K Type

1 (0001): J Type

2 (0010): T Type

3 (0011): B Type

4 (0100): R Type

5 (0101): S Type

6 (0110): E Type

7 (0111): N Type

8 (1000): L Type

9 (1001): U Type

10 (1010): C Type

11 (1011): D Type

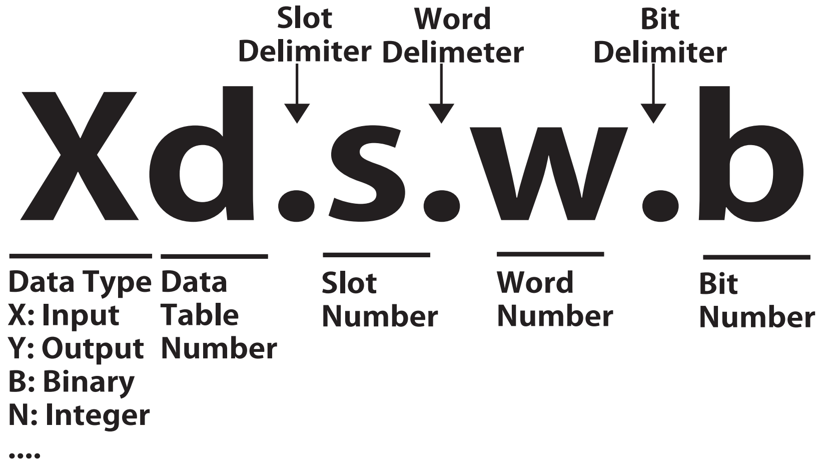
12 (1100): -78.0~78.0mV

13 (1101): -32.7~32.7mV

14 (1110): -65.5~65.5mV

# I/O Addressing

## Addressing Detail




I/O Addressing schema

| Format | Explanation | | |
|---|---|---|---|
| Od.sw.b<br>Id.sw.b | X | Data Type | Input (X), Output (Y), Status (ST), Binary (B), Integer (N), Timer (TM), Counter (CT), Control (CR), ASCII (A), Float (F), STRING (ST), Long (L), Message (MG), PID (PD), Pulse Limited Switch (PS), Routing Path (RP) |
| | d | Data Table Number | 0 = output, 1 = input to Fixed<br>Available to users from No. 3. |
| | . | Slot Delimiter | |
| | s | Slot Number | Embedded I/O Slot Number : 0 |
| | | | Expansion I/O Slot Number: 1 to 40 |
| | . | Word Delimiter | |
| | w | World Number | Required to read/write words, or if the discrete bit number is above 15. |
| | . | Bit Delimiter | |
| | b | Bit Number | 0 to 15 |

## Addressing Example

| Addressing Level | Explanation | Data Table No. | Word | Bit |
|---|---|---|---|---|
| Bit Addressing | X1.0.0.0 | 1 | Word 0 | Input bit 0 |
| | Y0.0.0.1 | 0 | Word 0 | Output bit 1 |
| | ST8.0.3 | 8 | Word 0 | Status bit 3 |
| | B3.0.15 | 3 | Word 0 | Binary bit 15 |
| | N20.0.4 | 20 | Word 0 | Integer bit 4 |

| Addressing Level | Explanation | Data Table No. | Word |
|---|---|---|---|
| Word Addressing | X1.0.1 | 1 | Word 1 |
| | Y0.3.3 | 0 | Word 3 |
| | ST.8.7 | 8 | Word 7 |
| | B13.2 | 13 | Word 2 |
| | N20.1 | 20 | Word 1 |

# I/O Forcing

I/O forcing is the ability to On/Off the specific bit at the user's discretion.

## Input Forcing

When an input is forced to specified value, the value in the input data file will be set to a user-defined state.
For digital inputs, you can force an input "on" or "off ".
When an input is forced, it no longer reflects the state of the physical input.
For embedded input port in the base module, the PLC reacts as if the force is applied to the physical input terminal.

**TIP** When an input is forced, it has no effect on the input device connected to the controller.

## Output Forcing

When an output is forced, the PLC overrides the status of the control program, and output port will be set the output to the user-defined state.
Digital I/O can be forced "on" or "off ".
The value in the output port is unaffected by the force.
It maintains the state determined by the logic in the control program.
However, the state of the physical output will be set to the forced state.

**TIP** If you force an output controlled by an executing PTO or PWM function, an instruction error is generated.

# Input Filtering

## Configurable Input

The **X8 Series PLC** allow users to configure embedded digital I/O of DC inputs for high-speed or normal operation. Users can configure each input's response time.
That is, it can be set the duration time that can recognize input signal form the PLC through a configurable input filter function.

If it is set to long time, the most absolute element to recognize the signal by the PLC is how long to keep the input signal must be "on". If it is used in electrically noisy environments, data can be entered through this function.

If it is set to short time, similarly, high-speed or short pulse signal operation can be set through the filtering function.

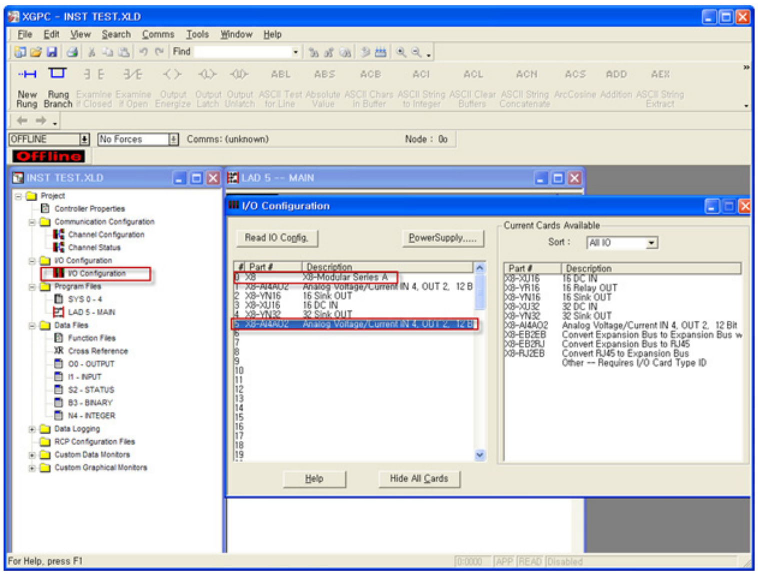
This function is used to process high-speed counters, latching inputs, and input interrupts.

Note that the maximum filtered value is 16mS , and minimum is 5µS.

Input filtering is configured using **XGPC programming software** which RS Automation is supply freely.

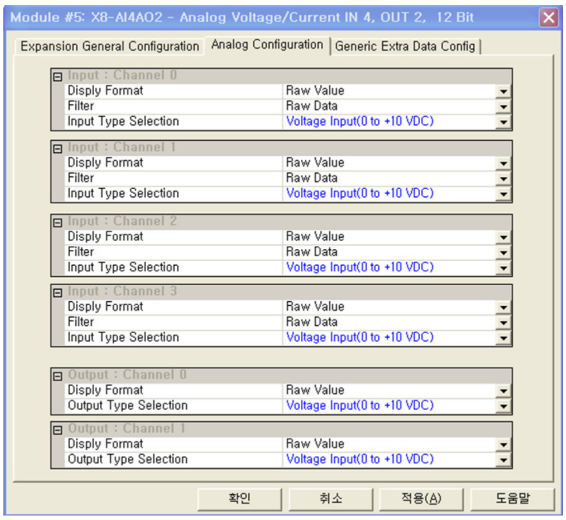
1. Select Project Tab

2. Select I/O Configuration Branch and double click.




3. Select I/O Module.

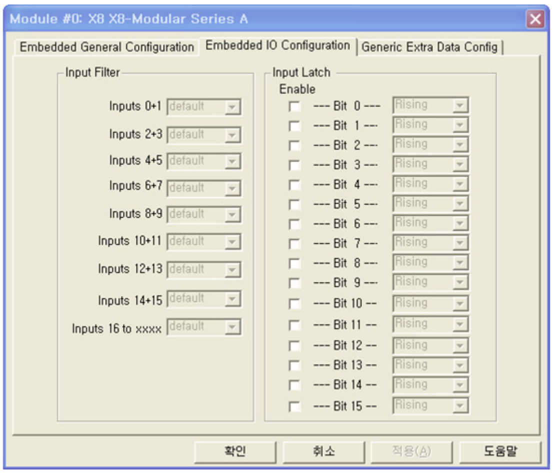
For example, if the X8-AI4AO2 is selected, user can config IO setting value by **XGPC**.



Almost the **X8 series PLC** Analog modules such as X8-AI4AO2, X8-AI8, X8-AO4, X8-RT6 and X8-TC6 are able to config internal setting value.

To use this function, the functions of analog modules, X8-AI4AO2 and X8-AI8, will be changed voltage Input or Current Input.

When select a default base module instead of above expanstion modules, user can config input settings such as input filtering and input ratching.



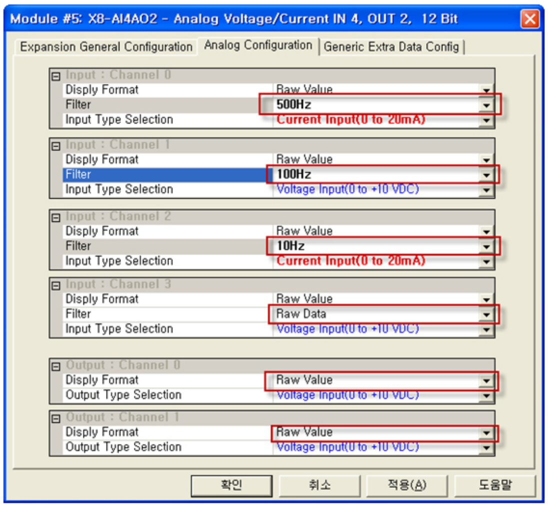

# Analog Input

## Analog Input Filter and Update times

The **X8 Series PLC**'s expansion analog input module can be configurable.

If the analog inputs have less tolerance to electrical noise, more accurate data can be input by the slower filter setting.

In order to increase the tolerance to external electrical noise, set the filtering speed to slow, and increase the speed of the filtering when the fast update time is required. However, if the filtering speed is increased, the tolerance to external electrical noise will be decreased.




X8-AI4AO2 Expastion Analog Module Filter Characteristics

| Filter Setting Value(Hz) | Filter Bandwidth (-3dB Freq Hz) | Setting Time (mSec) | Resolution (Bits) |
|---|---|---|---|
| 10 | 19 | 100.00 | 12 |
| 50 | 50 | 20.00 | 12 |
| 60 | 60 | 16.67 | 12 |
| 250 | 250 | 4 | 12 |

X8-AI8 Expastion Analog Module Filter Characteristics

| Filter Setting Value(Hz) | Filter Bandwidth (-3dB Freq Hz) | Setting Time (mSec) | Resolution (Bits) |
|---|---|---|---|
| 10 | 19 | 100.00 | 16 |
| 50 | 50 | 20.00 | 16 |
| 60 | 60 | 16.67 | 16 |
| 250 | 250 | 4 | 16 |

**TIP**

10Hz is the default setting.

The total update time is one ladder scan time plus the setting time.

If a 250 Hz filter is selected, the maximum update Time = ladder scan time + 4ms.

## Input Channel Filtering

The analog input channels use on-board signal conditioning to distinguish AC power line noise from normal variations in the input signal. Frequency components of the input signal at the filter frequency are rejected.

Frequency components below the filter bandwidth (-3 dB frequency) are passed with under 3 dB of attenuation. This pass band allows the normal variation of sensor inputs such as temperature, pressure and flow transducers to be input data to the processor.

Noise signals coupled in at frequencies above the pass band are sharply rejected. An area of particular concern is the 50/60 Hz region, where pick-up from power lines can occur.

## Converting Analog Data

The analog input circuits are able to monitor voltage signals and convert them to digital data.
The followings show sample Analog Signal and Data Word values using the nominal transfer function formula:

Analog to data word conversion (12 Bits)

$$N = Vin \times \frac{4095}{10} \times \text{ where } Vin \text{ (analog signal) is in volts (V)}$$

Analog to data word conversion (16 Bits)

$$N = Vin \times \frac{65535}{10} \times \text{ where } Vin \text{ (analog signal) is in volts (V)}$$

## Converting Analog Input Data

Analog to data word conversion (12 Bits)

| Analog Signal | Data Word |
|---|---|
| 0V | 0 |
| 5V | 2048 |
| 10V | 4095 |

Analog inputs convert voltage signals into 12 bits or 16 bits values. The followings show equations for input voltage .

12 Bits:

$$\frac{10V}{4095} x InputValue = InputVoltage(V)$$

16 Bits:

$$\frac{10V}{65535} x InputValue = InputVoltage(V)$$

For example, if an input value of 4000 is in the memory, the calculated value is as follows:

Example for 12 Bits

$$\frac{10V}{4095} x 4000 = 9.768(V)$$

Example for 16 Bits

$$\frac{10V}{65535} x 4000 = 0.610(V)$$

## Analog Output

The **X8 Series PLC** have X8-AI4AO2 that supports 2-channels, 12-bit resolution and X8-AO4 that supports 4-channels, 16-bit resolution. X8-AI4AO2 has 0~10V DC or 4~20mA output range.

The X8 Series PLC have X8-AI4AO2 that supports 2-channels, 12-bit resolution and X8-AO4 that supports 4-channels, 16-bit resolution. X8-AI4AO2 has 0~10V DC or 4~20mA output range. X8-AO4, as high functional product, supports 0~5V, 0~10V, -5~5V and -10~10V voltage output, and 0~20mA and 4~20mA current output depending on the setting.

## Latching Inputs

### Converting Analog Data to Actual Output Voltage

X8-AI4AO2 is an analog combo I/O module and supports 2-channels, 12 bit resolution and have 0~10V voltage output range.

X8-AO4 is an analog output module and supports voltage output 0~5V, 0~10V, and -5~5V, and -10~10V, and current output 0~20mA and 4~20mA depending on the setting.

Word 4 and 5 contain the value of output image (Word 4 : channel 0, Word 5 : channel 1).

The followings show equations for converting Analog Data to Actual Output Voltage.

12 Bits:

$$\frac{10V}{4095} x OutputValue = OutputValue(V)$$

16 Bits:

$$\frac{10V}{65535} x OutputValue = OutputValue(V)$$

For example, if 3000 is send to the analog voltage output port, the calculated value is as follows:

12 Bits:

$$\frac{10V}{4095} x 3000 = 7.326(V)$$

16 Bits:

$$\frac{10V}{65535} x 3000 = 0.457(V)$$

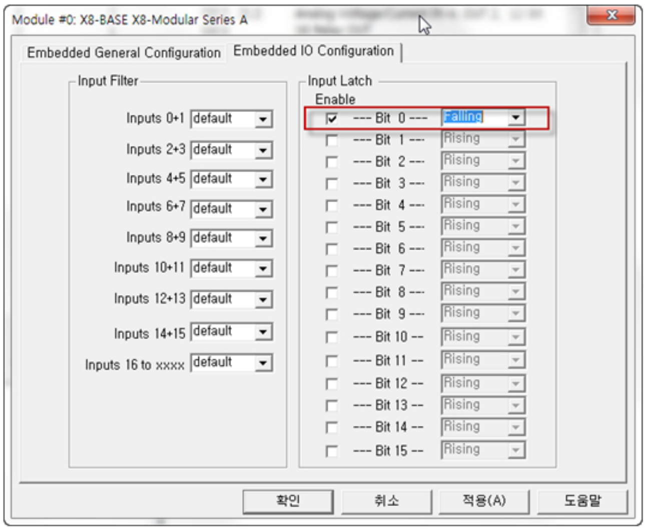
The **X8 Series PLC** provides the ability to configure latching input ports. A latching input is an input that captures a high-speed pulse signal and holds it for a single PLC scan.

The pulse width that can be captured is dependent upon the input filtering selected for that input.

| Controller | X8 Series PLC |
|---|---|
| 0V | 0~11 |

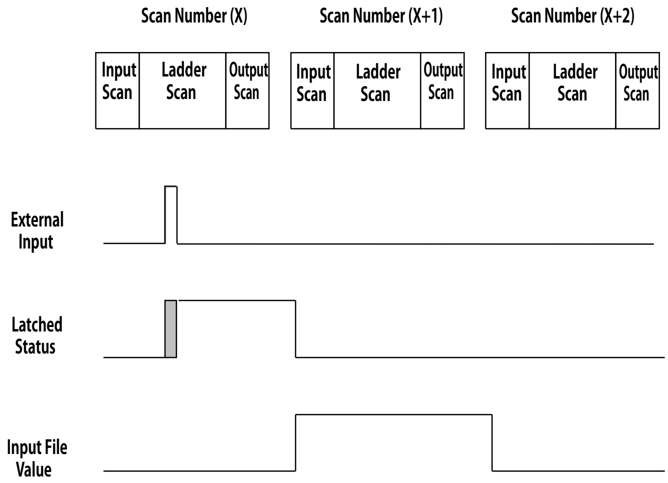
Enable this feature using **XGPC**:

1. Open the Controller folder on the left and click the Embedded I/O configuration folder.
2. Double click the base module in the Slot 0.
3. If screen appears as shown in the figure below, selects the corresponding bit to activation, and then, select rising (Rising Edge) or falling (Falling Edge).

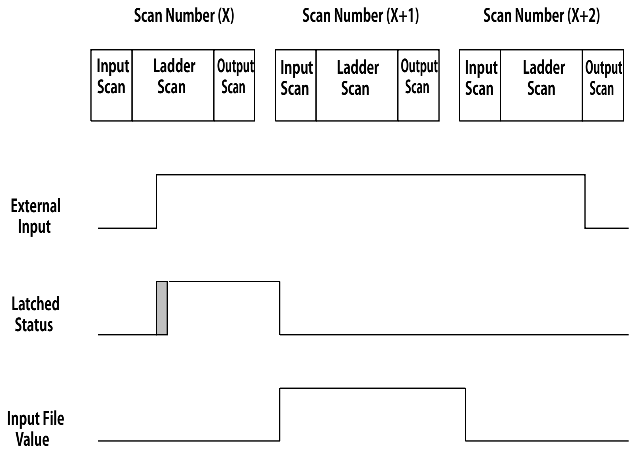



When an external signal is detected at Rising Edge, the PLC stores this event. In general, at the next input scan, this signal is turned "on" and remains "on" for the next controller scan.

## Rising Edge Time Chart-Example 1




## Rising Edge Time Chart-Example 2



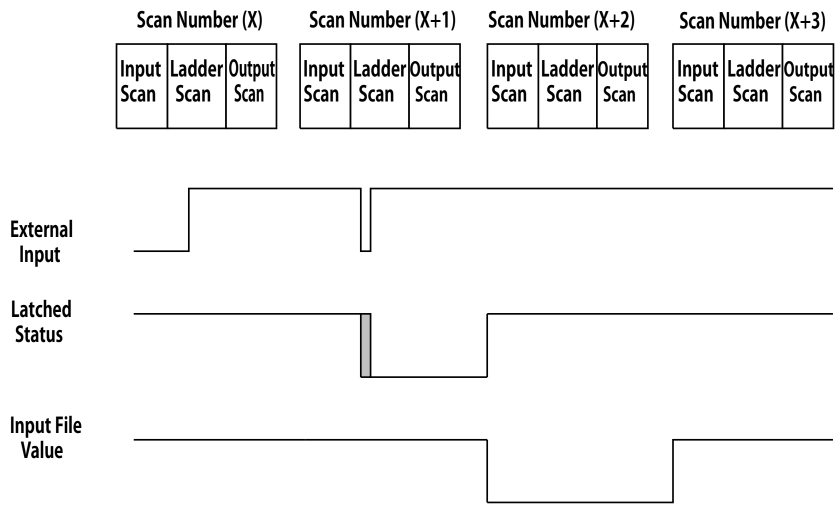

**TIP** The “gray” area of the time chart is the input filter delay.

**IMPORTANT** The external input signal does not displayed in the input data area when the input port is configured for latching behavior.

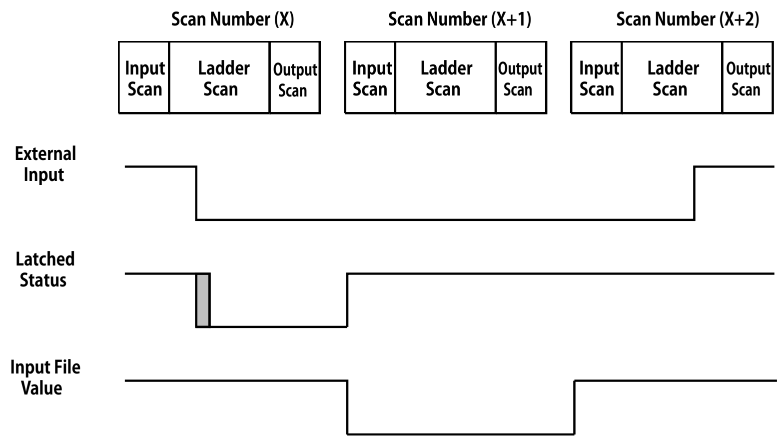
When the input port configuration is raising Edge in input data area, the data value is normally "off" and "on" when a rising edge is detected.

The previous examples demonstrate rising edge behavior. Falling edge behavior operates exactly the same way:

## Falling Edge Time Chart-Example 1



## Falling Edge Time Chart-Example 2




**TIP** The "gray" area of the time chart is the input filter delay.

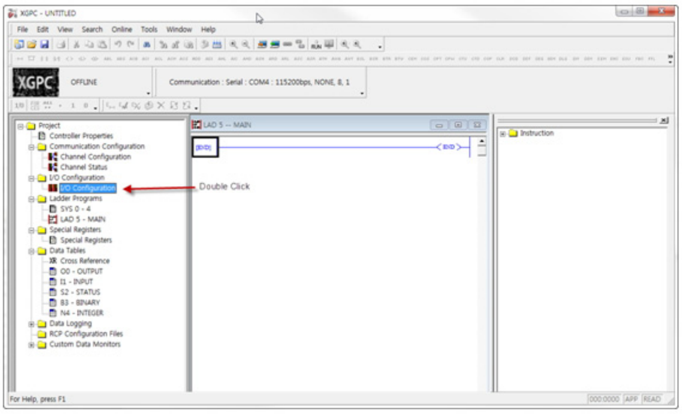
**IMPORTANT** The external input signal does not displayed in the input data area when the input port is configured for latching behavior.

When the input port configuration is falling Edge in input data area, the data value is normally "on" and "off" when a rising edge is detected.

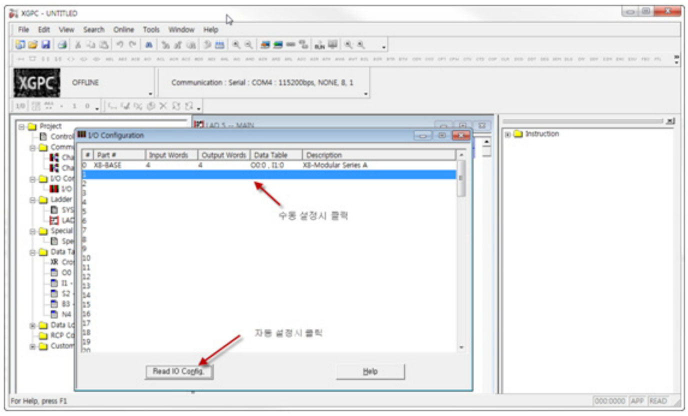
## Configure Expansion I/O Using XGPC

Configuring expansion I/O can be done by using **XGPC**.

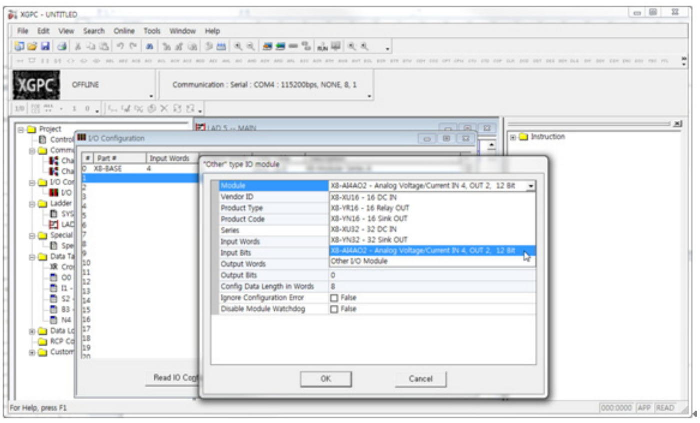
1. Double click the I/O Configuration in the project folder.



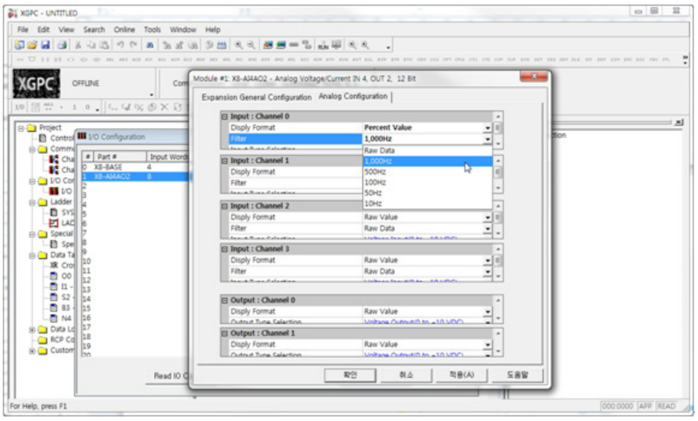

2. From the below screen, select empty slot number for manual setting and then double click. For automatic setting, click "Read I/O Config" button.




3. Expansion module selection screen will be appeared in the manual setting. In automatic setting, all expansion modules currently connected will be displayed automatically.



4. For some modules, usually with the default settings, user defined setting is required. The screen below shows setup screen of X8-AI4AO2 module.



# X8 Series PLC Memory and Data Table

## Introduction

This chapter describes Memory and Data Table used by the **X8 Series PLC**. The chapter is organized as follows:

# X8 Series PLC Memory

## Data Table Structure

The **X8 Series PLC** internal memory comprises Data Table, Special Function Register, and Ladder Program, etc.

**TIP**

The data table number 0~2 sown in the table are reserved number for internal uses and cannot be changed. (0: Internal Output Port. 1: Assigned Internal Input module, 2: Assigned to internal status register)

Various data tables from 3-1535 shown in the table below can be created in X8 Series PLC.

**PLC Internal Data Table Structure and Type**

| Data Table | | SFR (Special Function Register) | | Ladder Program | | Specialty Files | |
|---|---|---|---|---|---|---|---|
| **No.** | **Description** | **Identifier** | **Description** | **No.** | **Description** | **No.** | **Description** |
| 0 | (Y) Output Register | HSC | High Speed Counter | 0 | System File 0 | 0 | Data Log Queue 0 |
| 1 | (X) Input Register | PTO | Pulse Train Output | 1 | System File 1 | 1 | Data Log Queue 1 |
| 2 | (SR) Status Register | PWM | Pulse Width Modulation | 2 | Program File 2 | 2 | Data Log Queues 2...1535 |
| 3 | (B) Bit File | STI | Slectable Timed Interrupt | 3 | System File 3 | 0 | Recipe File 0 |
| 4 | (N) Integer Register | EII | Event Input Interrupt | 4 | System File 4 | 1 | Recipe File 1 |

**PLC Internal Data Table Structure and Type**

| Data Table | | SFR (Special Function Register) | | Ladder Program | | Specialty Files | |
|---|---|---|---|---|---|---|---|
| 5 ~ 1535 | (B) Bit Register<br>(N) Integer Register<br>(TM) Timer Register<br>(CT) Counter Register)<br>(CR) Control Register<br>(F) Floating Point Register<br>(B) Bit<br>(T) Timer<br>(CT) Counter<br>(CR) Control<br>(N) Integer<br>(F) Floating Point<br>(ST) String<br>(A) ASCII<br>(L) Long Word<br>(MG) Message<br>(PD) PID<br>(PS) Programmable Switch<br>(RP) Routing Path | RTC | Real Time Clock | 5 | Ladder Program 5 | 2~255 | Recipe File 2~255 |
| | | MMI | Memory Module Information | 6 | Ladder Program 6 | | |
| | | BHI | Base Hardware Information | 7 | Ladder Program 7 | | |
| | | CS0 | Communications Status for Channel 2 | 8 | Ladder Program 8 | | |
| | | BHI | Base Hardware Information | 9~511 | Ladder Program 9~511 | | |
| | | CS0 | Communications Status for Channel 0 | | | | |
| | | CS2 | Communications Status for Channel 2 | | | | |
| | | DLS | Data Log Status | | | | |
| | | ES1 | Ethernet Status For Channel 1 | | | | |

## User Memory

User memory is the amount of storage of X8 Series PLC available to store user defiend data such as ladder program, data tables, and I/O configuration.

User Memory consist of the system status data, I/O image files, and all other user-creatable data tables. A word is defined as a unit of memory in the PLC and its size is 16 words.

Memory consumption is allocated as follows:

- INTEGER (N)= 1 Word
- LONG INTEGER (L)= 2 Words
- TIMER (TM)= 3 Words

**TIP** Each input and output memory consumes 3 words due to the overhead associated with I/O forcing.

- For example, memory consumption in Ladder Program is allocated as follows:
  - NOCC instruction which has 1 operand, consumes 1 word
  - EQU instruction which has 2 operands, consumes 2 words
  - ADD instruction which has 3 operands, consumes 3 words
- SFR (Special Function Register) does not comsume user memory.

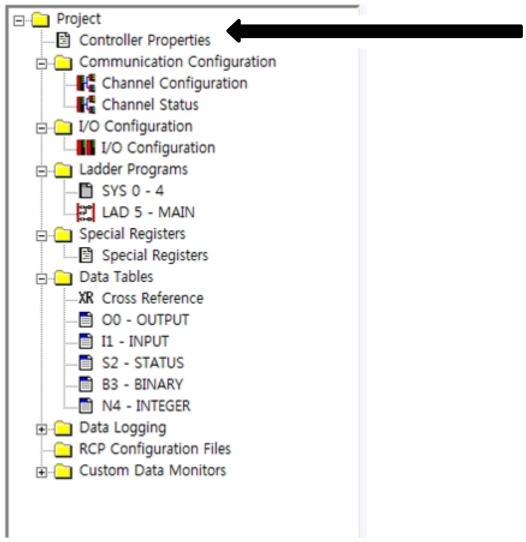
**TIP** Although the PLC allows up to 256 elements in the Data Table, it may not actually be possible to create data table (0~1535) with 256 elements due to 64KW limited user memory size in the PLC.

### X8 Series PLC User Memory

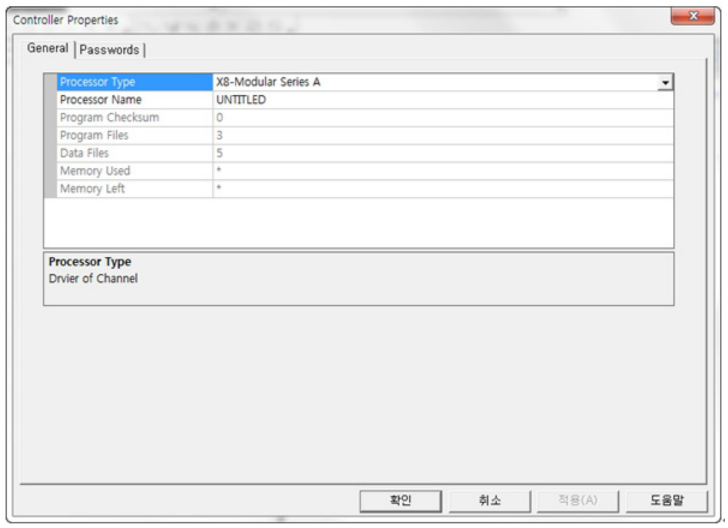
The **X8 Series PLC** supports64KW of memory for data and programs. And X8 Series PLC suuports seperate 4GB SD memory for data logging, recipe, ladder backup.

## Viewing X8 Series PLC Memory Usage

1. Select Controller Properties and double click.




2. The amount of Memory Used will appear in the Controller Properties window.




## Data Table

| Data Table | Operand | Data Table No. | Words per Element | Description |
|---|---|---|---|---|
| Output | Y | 0 (FIX) | 1 | The output image data table stores the actual output value of the embedded output port. |
| Input | X | 1 (FIX) | 1 | The input image data table stores the actual input value entered into the embedded input port. |
| SFR (Special Function Register) | S | 2 (FIX) | 1 | SFR (Special Function Register) contains internal status information of X8 Series PLC. |

| Data Table | Operand | Data Table No. | Words per Element | Description |
|---|---|---|---|---|
| Binary | B | 3~1535 | 1 | Binary typed Data Table for bit logic. |
| Timer | TM | | 3 | Data Tablefor timer operation. |
| Counter | CT | | 3 | Data Table for counter operation. |
| Contorl | CR | | 3 | The contorl data table is used for various purposes (Error, Length, Position, etc.) in the Ladder Program. |
| Integer | N | | 1 | The Integer typed Data Table for integer. |
| Floating Point | F | | 2 | Floating-point typed Data Table for floating-point logic processing. |
| String | ST | | 42 | String typed Data Table for string logic |
| ASCII | A | | 1 | ASCII typed Data Table for ASCII character logic |
| Long Word | L | | 2 | LONG INTEGER typed Data Table for 32-bit LONG INTEGER logic |
| Message | MG | | 25 | Data Table for Message transmission. |
| Programmable Limit Switch | PS | | 6 | The Programmable Limit Switch (PLS) File allows you to configure the High-Speed Counter to operate as a PLS or rotary cam switch. See Programmable Limit Switch (PLS) File for information. |
| PID | PD | | 23 | Data Table for PID control Loop and instruction. |
| Routing Path | RP | | 20 | Data Table contains communication data used to MSG insturction. |

# Protecting Data Table During Download

## Data Table Download Protection

When PLC ladder program and a user configured data must not change is stored in the PLC, if there may be a need to change/update without destroying the data, able to data protection configuration on the XGPC.

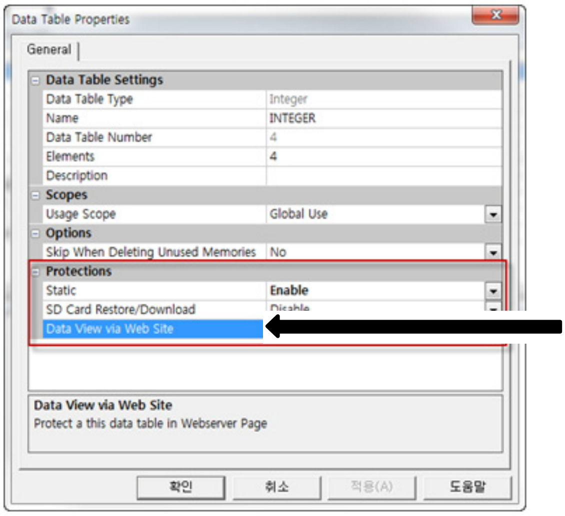
### Setting Data Table Protection

Data Table Protection can be applied to the following data types:

- Output (Y)
- Input (X)
- Binary (B)
- Timer (TM)
- Counter (CT)
- Control (CR)
- Integer (N)
- Floating Point (F)
- String (ST)
- ASCII (A)
- Long Word (L)
- Proportional Integral Derivative (PD)

- Message (MG)
- Programmable Limit Switch (PS)
- Routing Path (RP)
- Recipe

**TIP** The SFR (Special Function Register) cannot be protected.




After selecting the desired data table, select "Property" by clicking right mouse to set protection of the selected data table.

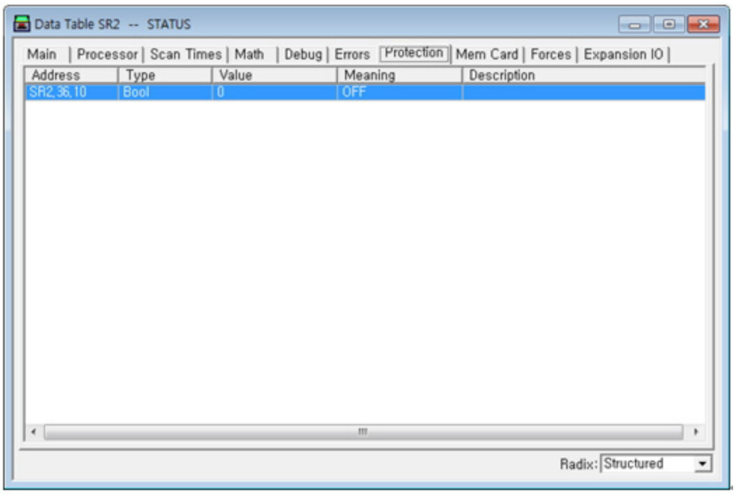
## Data Table Protection Requirements

Following are conditions for data table protection:

- The PLC contains protected data table.
- The ladder program being downloaded to the PLC has the same number of protected data tables as the ladder program to be updated stored in the internal PLC.
- All protected data table numbers, types, and sizes (number of elements) currently in the controller exactly match.

If all of these conditions are met, the X8 Series PLC will not write over any data table in the PLC that is configured as Protected.

If any of these conditions are not met, the entire data table in the internal PLC i is updated with the contents from being sent XGPC.




After transmit the ladder and data table form XGPC, if the protected data table stored in the internal PLC is changed, SR2.36.10 register status in the screen is changed to "ON" status.

**TIP** The X8 Series PLC will not reset the SR2.36.10 register automatically.

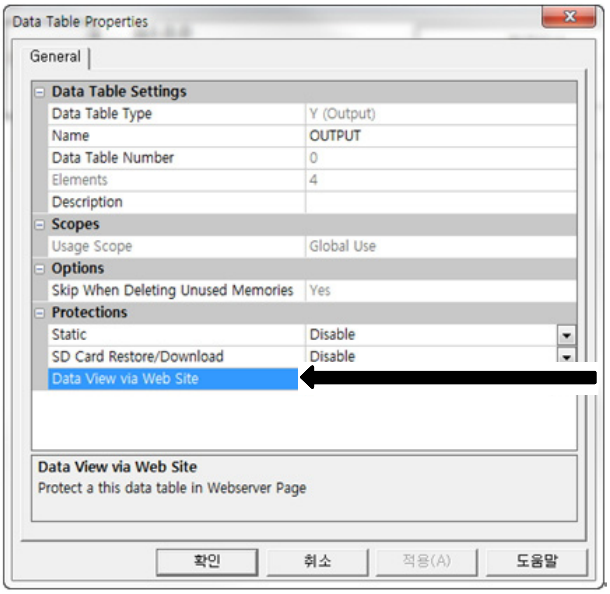
## Static

Static part of Properties in the Data Table defines whether it can change its value through communication.

Static can be applied to the following data types:

- Output (Y)
- Input (X)
- Status (S)
- Binary (B)
- Timer (TC)
- Counter (CT)
- Control (CR)
- Integer (N)
- Floating Point (F)
- String (ST)
- ASCII (A)
- Long Word (L)
- Proportional Integral Derivative (PD)
- Message (MG)

- Programmable Limit Switch (PS)
- Routing Path (RP)



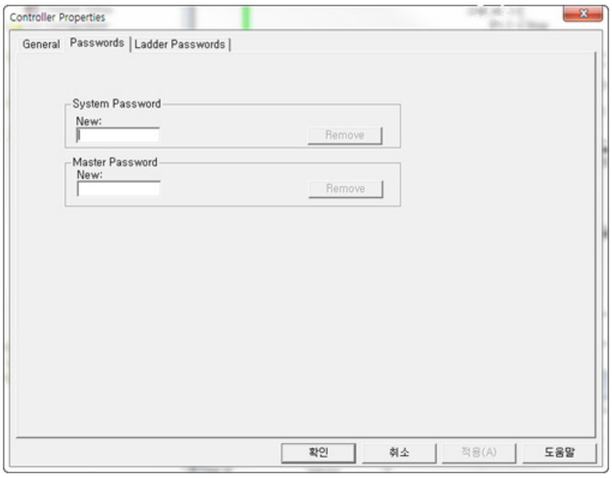

## Password

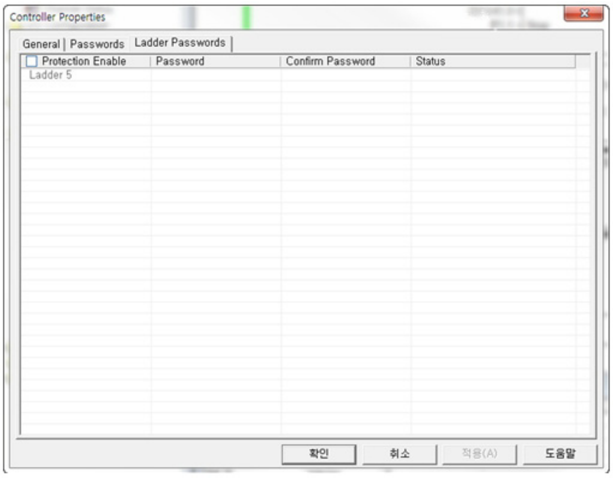
X8 Series PLC supports powerful threefold password protection system using characters include special characters never before seen in the existed small PLC.

PLC System Passwords consist of up to 12 characters include special characters.

Each PLC program may contain two passwords, the Password and the Master Password.

| Password | Description |
|---|---|
| Master Password<br>System Password | This is main password of system. By using the master password and system password, it can be set the password redundancy.<br>The Master Password takes precedence over the System Password.<br>In General, to set more powerful password protection use a common master password in the environments that uses multiple X8 series PLCs, and otherwise use different system password in each PLC. |
| Ladder Password | Separate password can be set for each ladder program. |



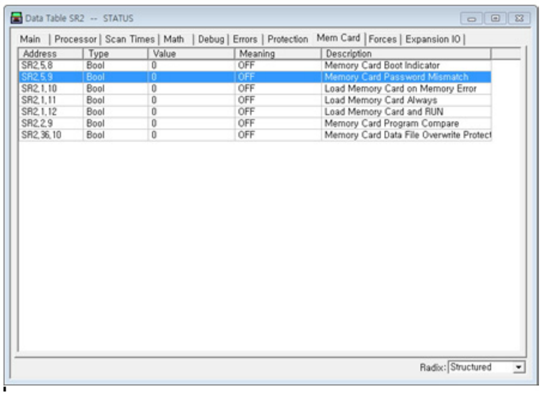

**TIP**

Even the **X8 Series PLC** password system supports more powerful threefold protection system that is consist of up to 12 characters include special characters, if a password is lost or forgotten, there is no way to bypass the password to recover the program.

The only option is to clear the PLC's memory.

**TIP**

When the password-protected ladder is transferred to the SD card, if each password does not match, SD card cannot be backed up. If password mismatch error occurred, the SR2.5.9 register status is ON.




## Clearing the Controller Memory

If the system is locked out and lost password, user should have to clear the internal PLC memory and download a new ladder program.

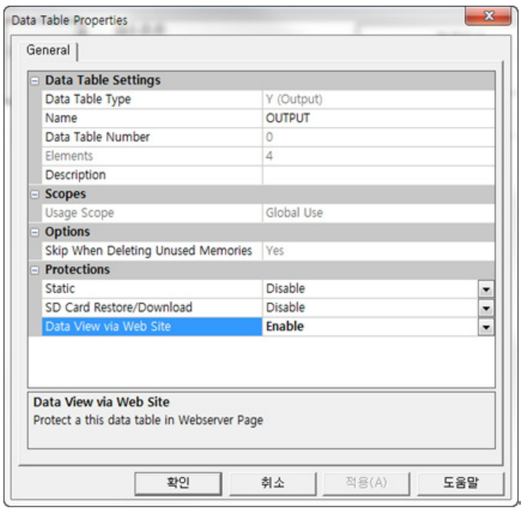
To clear the momory:

1. Enter 65257636 on **XGPC** PLC Property Dialog Box.
2. When the Programming Software detects this number has been entered, it asks if you want to clear the memory in the controller.
3. If you reply "yes" to this prompt, the programming software instructs the controller to clear PLC memory.

## Allow Access Data Table through Internet (Web)

This allows open to public X8 Series PLC's internal data table via Web.

Only offline setting is avaiable. If it is online status, it cannot be selected.



# SFR (Special Function Register)

## Introduction

This chapter describes Special Function Registers include internal configuration and status data of **X8 Series PLC**. The chapter is organized as follows.

## Overview

SFR (Special Function Resister) are embedded default resources within **the X8 series PLC**. SFR provide useful features such as RTC (Real Time Clock), HSC (High Speed Counter), and PTO (Pulse Trained Output), etc.

**SFR (Special Function Register) Type**

| SFR Name | SFR Identifier | Description |
|---|---|---|
| High-Speed Counter | HSC | This data table associated with the High-Speed Counter. |
| Pulse Train Output | PTO | This data table associated with the Pulse Train Output. |
| Pulse Width Modulation | PWM | This data table associated with the Pulse Width Modulation. |
| Selectable Timed Interrupt | STI | This data table associated with the Selectable Timed Instruction. |
| Event Input Interrupt | EII | This data table associated with the Event Input Interrupt. |
| Real-Time Clock | RTC | This data table associated with the Real-Time Clock (time of day). |
| Memory Module Information | MMI | This data table contains information about the Memory Module. |
| Base Hardware Information | BHI | This data table contains information about the PLC’s memory module. |
| Communications Status File for Channel 0 | CS0 | This contains information about the PLC's hardware. |
| Communications Status File for Channel 2 | CS2 | |
| Ethernet Status File for Channel 1 | ES1 | This contains information about the Ethernet Communications with the PLCs. |

## Real-Time Clock Register

The real-time clock provides year, month, day of month, day of week, hour, minute, and second information to the Real-Time Clock (RTC) data table in the PLC.

The Real-Time Clock parameters and their valid ranges are shown in the table below

| Feature | Address | Data Format | Range | Type | User Program Access |
|---|---|---|---|---|---|
| YR - RTC Year | RTC:0.YR | word | 1998…2097 | Status | read/write |
| MON - RTC Month | RTC:0.MON | word | 1…12 | Status | read/write |
| DAY - RTC Day of Month | RTC:0.DAY | word | 1…31 | Status | read/write |
| HR - RTC Hours | RTC:0.HR | word | 0…23 (military time) | Status | read/write |
| MIN - RTC Minutes | RTC:0.MIN | word | 0…59 | Status | read/write |
| SEC - RTC Seconds | RTC:0.SEC | word | 0…59 | Status | read/write |
| DOW - RTC Day of Week | RTC:0.DOW | word | 0…6 (Sunday to Saturday) | Status | read/only |
| DS - Disabled | RTC:0/DS | binary | 0 or 1 | Status | read/only |
| BL - RTC Battery Low | RTC:0/BL | binary | 0 or 1 | Status | read/only |

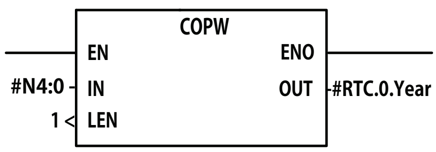
### Writing Time Data to the Real-Time Clock

There are two ways to change the RTC setting data.

The RTC settings may be changed by either the ladder program of **XGPC** programming software, and a write MSG instruction from another PLCs connected with serial or Ethernet communication.

Use the CPW (Copy Word) instruction to adjust the RTC settings within the ladder logic as follows:

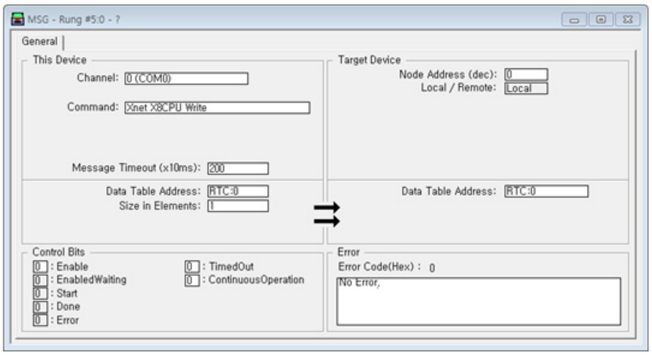
Following example shows data transmission method by using Copy Word instruction.



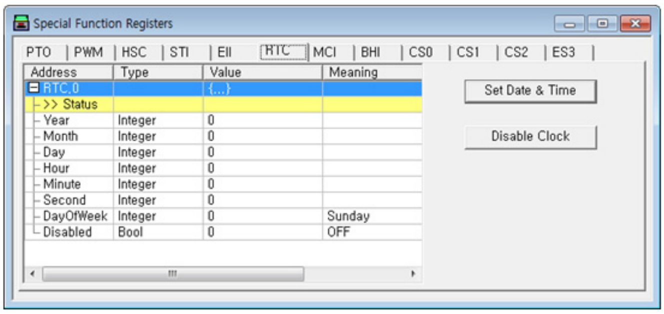

**IMPORTANT** An Error Code 44H is generated if any of the data being written to the RTC data table is invalid.

For example, setting the Seconds to 61.

An example write MSG from another **X8 Series PLC** to synchronize their RTCs is shown here:




The RTC in SFR screen is shown below:




When valid data is sent to the real-time clock from the programming device or another PLCs, the new values are stored to the RTC immediately . In **XGPC**, click “**Set Date & Time**” button in the screen above to synchronize PC time data to RTC of **X8 Series PLC.**

File screen to set the RTC time to the current time on your PC.

The real-time clock does not allow you to load or store invalid date or time data.

**TIP** Use the “**Disable Clock**” button in your programming device to disable the real-time clock before storing a module.

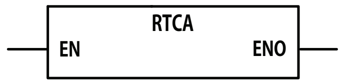
## X8 Real-Time Clock Accuracy

The following table indicates the expected accuracy of the real-time clock for various temperatures.

| Ambient Temperature | Accuracy (1) |
|---|---|
| 0°C (+32°F) | -13…-121 seconds/month |
| +25°C (+77°F) | +54…-54 seconds/month |
| +40°C (+104°F) | +29…-78 seconds/month |
| +55°C (+131°F) | -43…-150 seconds/month |

(1)These numbers are worst case values.

## RTCA-Real Time Clock Adjust Instruction




Instruction Type: output

The RTCA instruction is used to synchronize the PLC Real-Time Clock (RTC) with an external source.

The RTCA instruction will adjust the RTC to the nearest minute. The RTCA instruction adjusts the RTC based on the value of the RTC Seconds as described below.

**IMPORTANT** The RTCA instruction will only change the RTC when the RTA rung is evaluated true, after it was previously false (false-to-true transition). The RTCA instruction will have no effect if the rung is always true or false.

RTCA is set:

- If RTC Seconds are less than 30, then RTC Seconds is reset to 0.
- If RTC Seconds are greater than or equal to 30, then the RTC Minutes are increment by 1 and RTC Seconds are reset to 0.

The following conditions cause the RTCA instruction to have no effect on the RTC data:

- RTC is disabled
- An external (via communications) message to the RTC is in progress when the RTA instruction is executed. (External communications to the RTC takes precedence over the RTA instruction.)

To normal operation of RTCA instruction, the RTCA rung must become false, and then true.

**TIP** There is only one internal storage bit allocated in the system for this instruction. Do not use more than one RTCA instruction in your program.

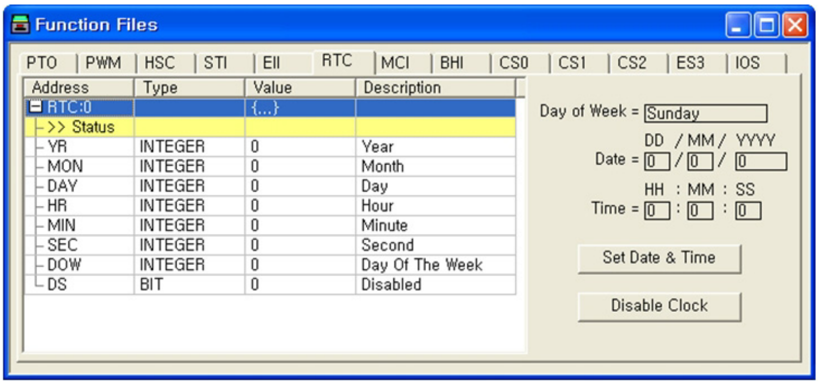
**TIP** You can also use a MSG instruction to write RTC data from **X8 Series PLC** to another to synchronize time.

## Memory Card Information Data Table

The X8 Series PLC has a Memory Card Data Table which is stores SD memory cards information. After power-up and detection of the SD memory card, various information - SD memory card model number, and revision information, etc -will be output.

If a SD memory card is not detected, zero (OFF) is written to the MCI data table.

The MCI data table of SFR screen is shown below:




The MCI parameters and their valid ranges are shown in the table below.

**MCI Data Table Parameters**

| Parameters | Address | Data Format | Type | User Program Access |
|---|---|---|---|---|
| Functionality Type | MCI.0.FUNCTIONALITYTYPE | Word | Status | Read Only |
| Error Code | MCI.0.ERRORCODE | Word | Status | Read Only |
| MemoryCardPresent | MCI.0.MEMORYCARDPRESENT | Binary | Status | Read Only |
| WriteProtected | MCI.0.FAULTOVERRIDE | Binary | Control | Read Only |
| FaultOverride | MCI.0.FAULTOVERRIDE | Binary | Control | Read Only |

**MCI Data Table Parameters**

| Parameters | Address | Data Format | Type | User Program Access |
|---|---|---|---|---|
| LoadProgramCompare | MCI.0.LOADPROGRAMCOMPARE | Binary | Control | Read Only |
| LoadOnError | MCI.0.LOADONERROR | Binary | Control | Read Only |
| LoadAlways | MCI.0.LOADALWAYS | Binary | Control | Read Only |

## Functionality Type

The flag is used to check the compatibility of the user program stored in the memory card

## MemoryCardPresent

The MemoryCardPresent flag can be used in the user program to determine when the SD memory card is present on the PLC. This bit is updated once per scan.

If a recognized memory module is removed during an executing mode, this bit updated to OFF at the end of the next ladder scan.

## WriteProtected

When the WriteProtected flag is ON, the SD memory card is write-protected and the user ladder program and system cannot be overwritten.

## FaultOverride

The FaultOverride flag shows the status of the FaultOverride of the user ladder program stored in the SD memory card.

## LoadProgramCompare

The LoadProgramCompare flag shows the status of the ladder program stored in the SD memory card.

## LoadOnError

The LoadOnError flag shows the status of the LoadOnError of the user ladder program stored in the SD memory card.

## LoadAlways

The LoadAlways flag shows the status of the LoadAlways of the user ladder program stored in the SD memory card.

ModeBehavior

The ModeBehavior flag shows the status of the ModeBehavior of the user ladder program stored in the SD memory card.

BHI (Base Hardware Information) data table is read-only that contains hardware information of the **X8 Series PLC**.

The BHI elements are as follow;

| Address | Descriptions |
|---|---|
| BHI:0.CN | CN - Catalog Number |
| BHI:0.SRS | SRS - Series |
| BHI:0.REV | REV - Revision |
| BHI:0.FT | FT - Functionality Type |

## Communications Status Data Table

The Communications Status Data Table is a read-only data table that contains status information on how the X8 Series PLC internal communication ports configuration and various registers, and located on CS0, CS1, CS2 and ES3 in the SFR.

**Communications Status Data Table Size**

| Register | Elements Size |
|---|---|
| CS0 (USB) | 71 Word |
| CS1 (COM0) | 71 Word |
| CS2 (COM1) | 71 Word |
| ES3 (EtherNet) | 171 Word |

**TIP** You can use the Communications Status Data Table as a troubleshooting tool for **X8 Series PLC** communications issues.

The Communications Status Data Table is structured as shown below:

| Word | Descriptions |
|---|---|
| 0…5 | General Communication Status Block |
| 6~22 | Diagnostic Counters Block |
| 23…42 | Data Link Layer Active Node Table Block |
| | |
| 43 | List Category Identifier Code (Always 0) |
| 43~70 | Reserved |

| Word | Descriptions |
| --- | --- |
| 43~69 | Modbus Slave Diagnostic Counters Block |
| 70 | End of List Category Identifier Code (Always 0) |

The following tables show the details of each General Communication Status block allocated from 0 to 5.

## General Communication Status Block

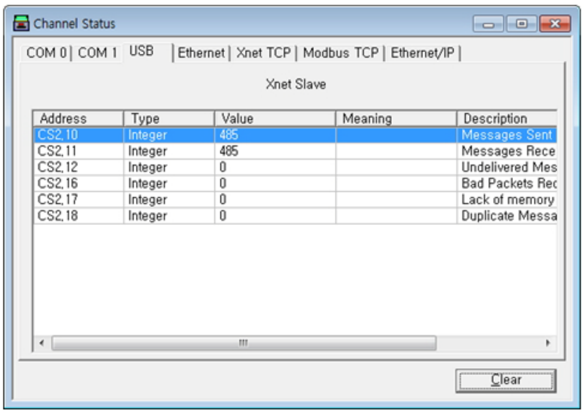
| Word | Bit | Description |
| --- | --- | --- |
| 0 | - | Communications Channel Status Information Category Identifier Code |
| 1 | - | Length |
| 2 | - | Format Code |
| 3 | - | Communications Configuration Error Code |
| 4 | 0 | ICP - Incoming Command Pending Bit<br><br>This bit is set when the **X8 Series PLC** determines that another device has requested information from this PLC. Once the request has been satisfied, the bit is cleared. |
| | 1 | MRP - Incoming Message Reply Pending Bit<br><br>This bit is set when the **X8 Series PLC** determines that another device has supplied the information requested by a MSG instruction executed by this controller. When the appropriate MSG instruction is serviced, this bit is cleared. |
| | 2 | MCP - Outgoing Message Command Pending Bit<br><br>This bit is set when the **X8 Series PLC** has one or more MSG instructions enabled and in the communication queue. This bit is cleared when the queue is empty. |
| | 3 | SSB - Selection Status Bit<br><br>This bit indicates that the **X8 Series PLC** is in the System Mode. It is always set. |
| | 4~14 | System Reserved |
| | 15 | Communications Toggle Push Button Communications Defaults Active.<br><br>This bit is set whenever Channel 0~2 are in the default communications mode.<br>The bit is cleared when Channels are in user configured communications mode. |

| Word | Bit | Description |
|---|---|---|
| 5 | 0 ~ 7 | Node Address<br><br>This byte value contains the node address of **X8 Series PLC** on the network. |
| | 8 ~15 | Baud Rate<br><br>This byte value contains the baud rate of the **X8 Series PLC** on the network. |

## Diagnostic Counter Block of Communications Status Data Table

With **XGPC**, displays of the diagnostic counters for PLCs are available.

**TIP** Select each communication port and protocol in the Channel Status screen on the **XGPC**, then click “Clear” button at the bottom of the screen to initialize counter value.



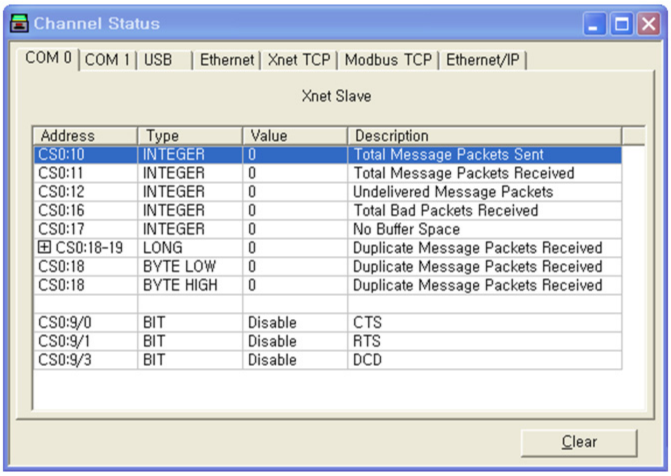

Diagnostic Counter Blocks are shown for:

- Xnet Master
- Xnet Slave
- NXPlus Master
- DF1 Full-Duplex
- DF1 Half-Duplex Slave
- DF1 Half-Duplex Master
- Modbus RTU Slave
- Modbus RTU Master
- ASCII (User Defined Protocol)

**Xnet Master Communication Port Counters Block**

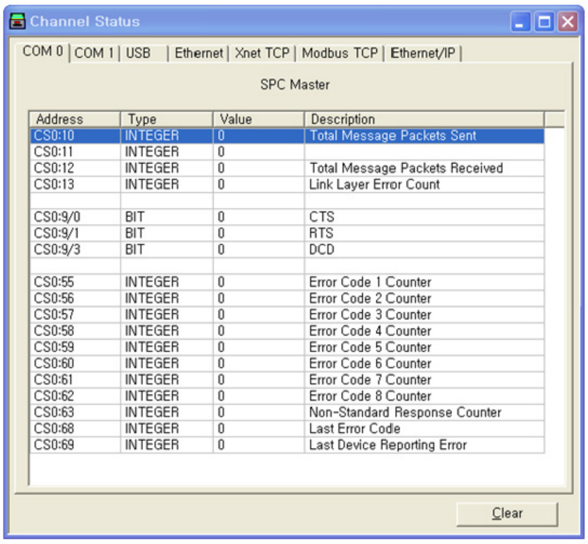
| Word | Bit | Description |
|---|---|---|
| 6 | - | Diagnostic Counters Category Identifier |
| 7 | - | Length (always 30) |
| 8 | - | Format Code (always 1) |
| 9 | 0 | CTS |
| | 1 | RTS |
| | 2 | Reserved |
| | 3 | Reserved |
| | 4~15 | Reserved |
| 10 | - | Total Message Packets Sent |
| 11 | - | Total Message Packets Received |
| 12 | - | Undelivered Message Packets |
| 13 | - | ENQuiry Packets Sent |
| 14 | - | NAK Packets Received |
| 15 | - | ENQuiry Packets Received |
| 16 | - | Bad Message Packets Received and NAKed |
| 17 | - | No Buffer Space |
| 18 | - | Duplicate Message Packets Received |
| 19~22 | - | Reserved |

Xnet Slave Communication Port Counters Block



| Word | Bit | Description |
|---|---|---|
| 6 | - | Diagnostic Counters Category Identifier |
| 7 | - | Length (always 30) |
| 8 | - | Format Code (always 1) |
| 9 | 0 | CTS |
| | 1 | RTS |
| | 2 | Reserved |
| | 3 | Reserved |
| | 4~15 | Reserved |
| 10 | - | Total Message Packets Sent |
| 11 | - | Total Message Packets Received |
| 12 | - | Undelivered Message Packets |
| 13 | - | ENQuiry Packets Sent |
| 14 | - | NAK Packets Received |
| 15 | - | ENQuiry Packets Received |
| 16 | - | Bad Message Packets Received and NAKed |
| 17 | - | No Buffer Space |
| 18 | - | Duplicate Message Packets Received |
| 19~22 | - | Reserved |

NXPlus Master Counters Block

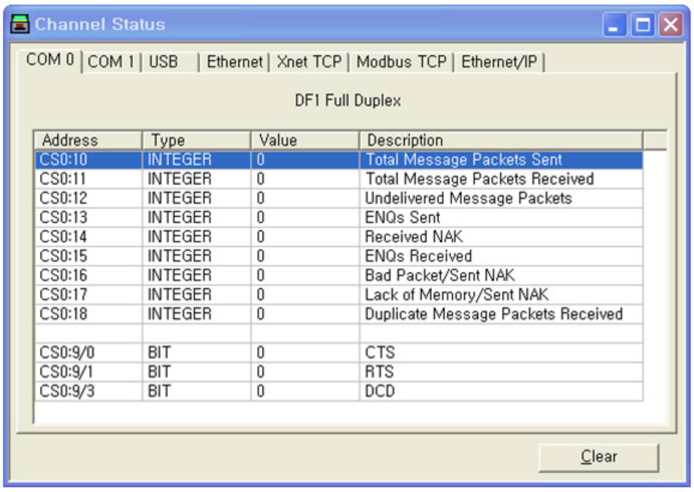


| Word | Bit | Description |
|---|---|---|
| 6 | - | Diagnostic Counters Category Identifier |
| 7 | - | Length (always 30) |
| 8 | - | Format Code (always 1) |
| 9 | 0 | CTS |
| | 1 | RTS |
| | 2 | Reserved |
| | 3 | Reserved |
| | 4~15 | Reserved |
| 10 | - | Total Message Packets Sent |
| 11 | - | Total Message Packets Received |
| 12 | - | Undelivered Message Packets |
| 13 | - | ENQuiry Packets Sent |
| 14 | - | NAK Packets Received |
| 15 | - | ENQuiry Packets Received |
| 16 | - | Bad Message Packets Received and NAKed |
| 17 | - | No Buffer Space |
| 18 | - | Duplicate Message Packets Received |
| 19~22 | - | Reserved |

DF1 Full-duplex slave Diagnostic Counters Block

| Word | Bit | Description |
|---|---|---|
| 6 | - | Diagnostic Counters Category Identifier |
| 7 | - | Length (always 30) |
| 8 | - | Format Code (always 0) |
| 9 | 0 | CTS |
| | 1 | RTC |
| | 2 | Reserved |
| | 3 | Reserved |
| | 4~15 | Reserved |
| 10 | - | Total Message Packets Sent |
| 11 | - | Total Message Packets Received |
| 12 | - | Undelivered Message Packets |
| 13 | - | ENQuiry Packets Sent |
| 14 | - | NAK Packets Received |
| 15 | - | ENQuiry Packets Received |
| 16 | - | Bad Message Packets Received and NAKed |

| Word | Bit | Description |
|---|---|---|
| 17 | - | No Buffer Space |
| 18 | - | Duplicate Message Packets Received |
| 19~22 | - | Reserved |

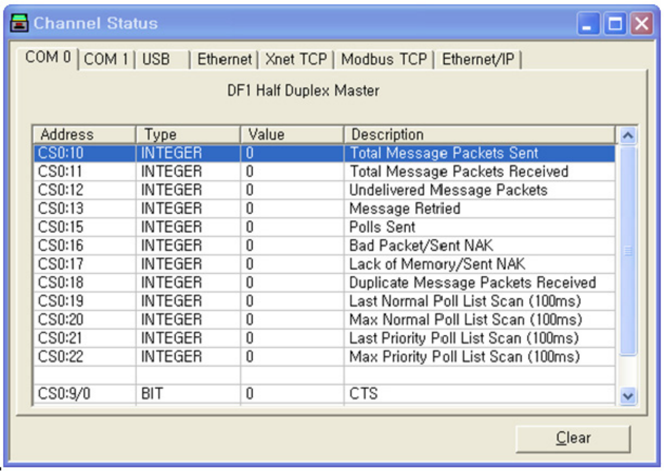



DF1 Half-Duplex Slave Diagnostic Counters Block

| Word | Bit | Description |
|---|---|---|
| 6 | - | Diagnostic Counters Category Identifier |
| 7 | - | Length (always 30) |
| 8 | - | Format Code (always 1) |
| 9 | 0 | CTS |
| | 1 | RTS |
| | 2 | Reserved |
| | 3 | Reserved |
| | 4~15 | Reserved |
| 10 | - | Total Message Packets Sent |
| 11 | - | Total Message Packets Received |
| 12 | - | Undelivered Message Packets |
| 13 | - | ENQuiry Packets Sent |
| 14 | - | NAK Packets Received |
| 15 | - | ENQuiry Packets Received |
| 16 | - | Bad Message Packets Received and NAKed |
| 17 | - | No Buffer Space |
| 18 | - | Duplicate Message Packets Received |
| 19~22 | - | Reserved |

DF1 Half-Duplex Master Diagnostic Counters Block

| Word | Bit | Description |
|---|---|---|
| 6 | - | Diagnostic Counters Category Identifier Code (always 2) |
| 7 | - | Length (always 30) |
| 8 | - | Format Code (always 3) |
| 9 | 0 | CTS |
| | 1 | RTS |
| | 2 | Reserved |
| | 3 | Reserved |
| | 4~15 | Reserved |
| 10 | - | Total Message Packets Sent |
| 11 | - | Total Message Packets Received |
| 12 | - | Undelivered Message Packets |
| 13 | - | Message Packets Retried |
| 14 | - | Reserved |
| 15 | - | Polls Sent |
| 16 | - | Bad Message Packets Received |
| 17 | - | No Buffer Space, Received Packet Dropped |
| 18 | - | Duplicate Message Packets Received |
| 19 | - | Last Normal Poll List Scan |
| 20 | - | Max. Normal Poll List Scan |
| 21 | - | Last Priority Poll List Scan |
| 22 | - | Max. Priority Poll List Scan |



Modbus RTU Slave Diagnostic Counters Block (Data Link Layer)

| Word | Bit | Description |
|---|---|---|
| 6 | - | Diagnostic Counters Category Identifier Code (always 2) |
| 7 | - | Length (always 30) |
| 8 | - | Format Code (always 4) |
| 9 | 0 | CTS |
| | 1 | RTS |
| | 2 | Reserved |
| | 3 | Reserved |
| | 4 | Reserved |
| 10 | - | Total Message Packets Sent |
| 11 | - | Total Message Packets Received for This Slave |
| 12 | - | Total Message Packets Received |
| 13 | - | Link Layer Error Count |
| 14 | - | Link Layer Error Code |
| 15~22 | - | Reserved |

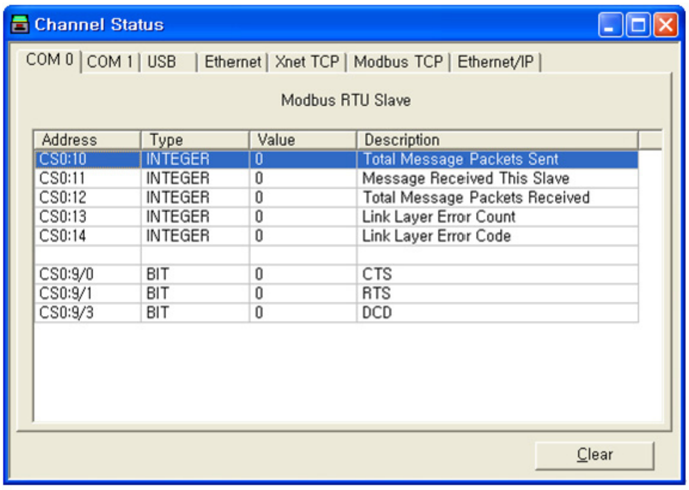
Modbus RTU Slave Diagnostic Counters Block (Presentation Layer)

| Word | Bit | Description |
|---|---|---|
| 43 | - | Diagnostic Counters Category Identifier (Always 10) |
| 44 | - | Length (Always 14) |
| 45 | - | Format Code (Always 0) |
| 46 | 0 | Pre-Send Time Delay |
| 47 | 0~7 | Node Address |
| | 8~15 | Reserved |
| 48 | 3 | Inter-Character Timeout |
| 49 | 4~15 | RTS Send Delay |
| 50 | - | RTS Off Delay |
| 51 | 0~7 | BaudRate (Communication Rate) |
| 12 | 8and9 | Parity |
| 13 | 10~15 | Reserved |
| 52 | - | Diagnostic Counters Category Identifier (Always 6) |
| 53 | - | Length (Always 32) |
| 54 | - | Format Code (Always 0) |
| 55 | - | Presentation Layer Error Code |
| 56 | - | Resentation Layer Error Count |

| Word | Bit | Description |
|---|---|---|
| 57 | - | Execution Function Error Code |
| 58 | - | Last Transmitted Exception Error Code |
| 59 | - | Data Table Number of Error Request |
| 60 | - | Element Number of Error Request |

Modbus RTU Slave Diagnostic Counters Block (Presentation Layer)

| Word | Bit | Description |
|---|---|---|
| 61 | - | Function Code 1 Message Counter |
| 62 | - | Function Code 2 Message Counter |
| 63 | - | Function Code 3 Message Counter |
| 64 | - | Function Code 4 Message Counter |
| 65 | - | Function Code 5 Message Counter |
| 66 | - | Function Code 6 Message Counter |
| 67 | - | Function Code 8 Message Counter |
| 68 | - | Function Code 15 Message Counter |
| 69 | - | Function Code 16 Message Counter |




Modbus RTU Master Diagnostic Counters Block (Data Link Layer)

| Word | Bit | Description |
|---|---|---|
| 6 | - | Diagnostic (Always 2) |
| 7 | - | Length (Always 30) |
| 8 | - | Format Code (Always 9) |

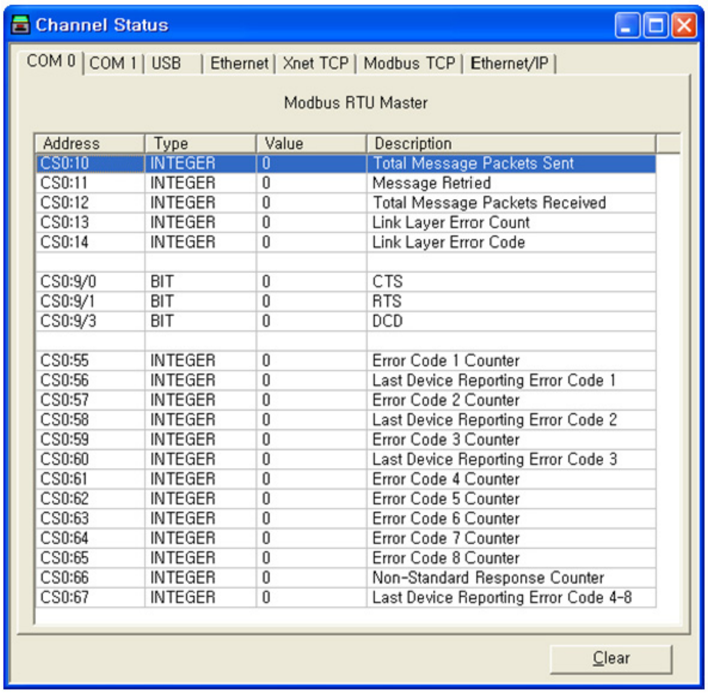
| Word | Bit | Description |
|---|---|---|
| 9 | 0 | CTS |
| | 1 | RTS |
| | 2 | Reserved |
| | 3 | Reserved |
| | 4~15 | Reserved |
| 10 | - | Total Message Packets Sent |
| 11 | - | Reserved |
| 12 | - | Total Message packets Received |

Modbus RTU Master Diagnostic Counters Block (Data Link Layer)

| Word | Bit | Description |
|---|---|---|
| 13 | - | Link Layer Error Count |
| 14 | - | Link Layer Error Code |
| 15~22 | - | Reserved |

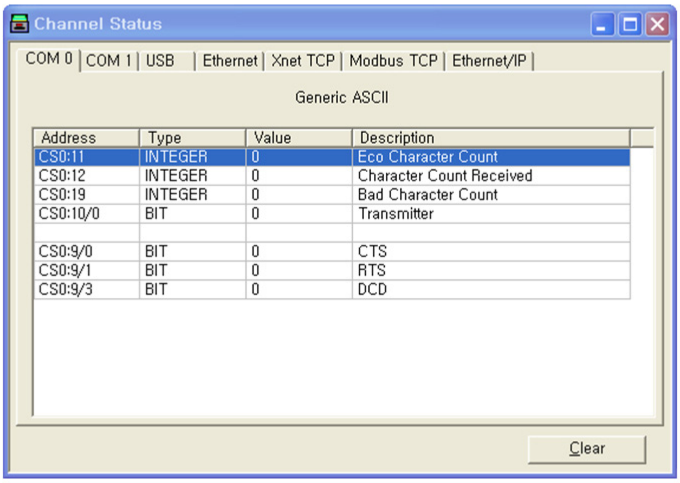
Modbus RTU Master Diagnostic Counters Block (Presentation Layer)

| Word | Bit | Description |
|---|---|---|
| 52 | - | Diagnostic Counters Category Identifier (Always 6) |
| 53 | - | Length (Always 32) |
| 54 | - | Format Code (Always 0) |
| 55 | - | ERR 1: Illegal Function |
| 56 | - | Last Device Reporting ERR 1 |
| 57 | - | ERR 2: Illegal Data Address |
| 58 | - | Last Device Reporting ERR 2 |
| 59 | - | ERR 3: Illegal Data Value |
| 60 | - | Last Device Reporting ERR 3 |
| 61 | | ERR 4: Slave Device Failure |
| 62 | | ERR 5: Acknowledge |
| 63 | | ERR 6: Slave Device Busy |
| 64 | | ERR 7: Negative Acknowledgement |
| 65 | | ERR 8: Memory Parity Error |
| 66 | | Non-Standard Response |
| 67 | | Last Device Reporting ERR 4 to ERR 8 or Non-Standard Response |
| 68and69 | | Reserved (Always 0) |




ASCII Diagnostic Counters Block

| Word | Bit | Description |
|---|---|---|
| 6 | - | Diagnostic Counters Category Identifier (Always 2) |
| 7 | - | Length (Always 30) |
| 8 | - | Format Code (Always 5) |
| 9 | 0 | CTS |
| | 1 | RTS |
| | 2 | Reserved |
| | 3 | Reserved |
| | 4~15 | Reserved |
| 10 | 0 | Software Handshaking Status |
| | 1~15 | Reserved |
| 11 | - | Echo Character Count |
| 12 | - | Received Character Count |
| 13~18 | - | Reserved |
| 19 | - | Bad Character Count |
| 20~22 | - | Reserved |




## Active Node Table Block

| Word | Description |
|---|---|
| 23 | Active Node Category Identifier Code (Always 3) |
| 24 | Length:<br><br>• Always 18 for DF1 Half-Duplex Master Communication<br>• Always 0 for DF1 Full-Duplex Master, DF1 Half-Duplex Slave, Modbus RTU Slave, Modbus RTU Master, ASCII Communication |
| 25 | Format Code (Always 0) |
| 26 | Number of Nodes:<br><br>• Always 255 for DF1 Half-Duplex Master<br>• Always 0 for DF1 Full-Duplex Master, DF1 Half-Duplex Slave, Modbus RTU Slave, Modbus RTU Master, ASCII Communication |
| 27 | Active Node Table (DH-485 and DF1 Half-Duplex Master) - Nodes 0 to 15<br><br>This is a bit-mapped register that displays the status of each node on the network. If a bit is set (1), the corresponding node is active on the network. If a bit is clear (0), the corresponding node is inactive. |
| 28 | Active Node Table (DH-485 and DF1 Half-Duplex Master) - Nodes 16 to 31 |
| 29 | Active Node Table (DF1 Half-Duplex Master) - Nodes 32 to 47 |
| ⋮ | ⋮ |
| 42 | Active Node Table (DF1 Half-Duplex Master) - Nodes 240 to 255 |

## Ethernet Communications Status Data Table

The Ethernet Communications Status Data Table is a read-only data table that contains information about Ethernet configuration data and Ethernet communication activity. The Ethernet Communications Status Data Table uses 178 word elements.

**TIP** You can use the Ethernet Communications Status Data Table as a troubleshooting tool for Ethernet communications issues.

**Ethernet Communication Status Data Table**

| Word | Description |
|---|---|
| 0 ~119 | General Channel Status Block |
| 120 ~176 | Diagnostic Counters Block |
| 177 | End of Table Identifier (Always 0) |

The following tables show the details of each block in the Ethernet Communications Status Data Table.

**General Channel Status Block**

| Word | Bit | Description |
|---|---|---|
| 0 | - | Ethernet Communication Channel Information Category Identifier (Always 1) |
| 1 | - | Length: 236 |
| 2 | - | Format Code |
| 3 | - | Communications Configuration Error Code |
| 4 | 0 | ICP - Incoming Command Pending Bit<br><br>This bit is ON when the **X8 Series PLC** determines that another device has requested information from this PLC. Once the request has been satisfied, the bit is OFF. |
| | 1 | MRP - Incoming Message Reply Pending Bit<br><br>This bit is ON when the **X8 Series PLC** determines that another device has supplied the information requested by a MSG instruction executed by this PLC. When the appropriate MSG instruction is serviced (during end-of-scan, SVC, or REF), this bit is OFF. |
| | 2 | MCP - Outgoing Message Command Pending Bit<br><br>This bit is ON when the **X8 Series PLC** has one or more MSG instructions enabled and in the communication queue. This bit is OFF when the queue is empty. |
| | 3~4 | System Reserved (Always 0) |

**General Channel Status Blcok**

| Word | Bit | Description |
|---|---|---|
| 4 | 5 | SNMP Server Status<br><br>This bit is ON when the SNMP server is enabled. The OFF means that the SNMP server is disabled. |
| | 6 | HTTP Server Status<br><br>This bit is ON when the internal X8 Series PLC web server is enabled. The OFF means that the internal web server is disabled. |
| | 7 | SMTP Client Status<br><br>This bit is set (1) when the SMTP client is enabled. The cleared bit (0) means that the SMTP client is disabled. |
| | 8~11 | System Reserved - Always 0 |
| | 12 | Modbus TCP Status<br><br>This bit is set (1) when the Modbus TCP Server/Client feature is enabled. The cleared bit (0) means that the Modbus TCP Server/Client is disabled. |
| | 13~14 | System Reserved - Always 0 |
| | 15 | Ethernet/IP Incoming Connection Status<br><br>This bit is set (1) when the Ethernet/IP Incoming connection is enabled. The cleared bit (0) means that the Incoming connection is disabled. |
| 5 | 0 | Ethernet Port Link Status<br><br>This bit is set (1) when the Ethernet link is active. |
| | 1 | Ethernet Port Connection Speed<br><br>This bit is valid when the Auto Negotiation function is enabled. This bit indicates the speed of the EtherNet Port at Ethernet port:<br>• 0: 10 Mbps<br>• 1: 100 Mbps |
| | 2 | System Reserved - Always 0 |
| | 3 | Duplex Mode<br><br>This bit is valid when the Auto Negotiation function is enabled.<br>This bit indicates the duplex mode of the Ethernet port: |
| | 4 | Auto Negotiate Status<br><br>This bit is set (1) when the Auto Negotiation function is enabled. |
| | 5 | Forced Speed Mode Status<br><br>This bit set (1) when the Auto Negotiation function is disabled and the Ethernet port speed is 100Mbps. |

**General Channel Status Blcok**

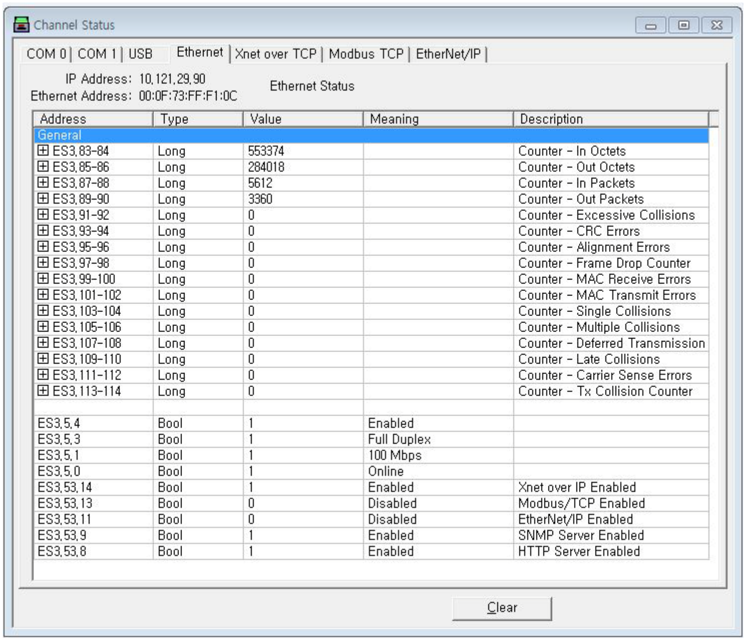
| Word | Bit | Description |
|---|---|---|
| 5 | 6 | Forced Duplex Mode Status<br><br>This bit set (1) when the Auto Negotiation function is disabled and the Ethernet port's duplex mode is Full Duplex. |
| | 6 | Forced Duplex Mode Status<br><br>This bit set (1) when the Auto Negotiation function is disabled and the Ethernet port's duplex mode is Full Duplex. |
| | 7 | System Reserved - Always 0 |
| | 8 | BOOTP Valid Flag (Default: 0, False)<br><br>This bit is set (1) when the appropriate BOOTP response has been received. |
| | 9 | DHCP Valid Flag (Default: 0, False)<br><br>This bit is set (1) when the appropriate DHCP response has been received. |
| | 10 | BOOTP Status Flag<br><br>This bit is set (1) if BOOTP is selected as IP Address configuration method. |
| | 11 | DHCP Status Flag<br><br>This bit is set (1) if DHCP is selected as IP Address configuration method. |
| 6 ~ 8 | - | Ethernet Hardware Address (6-byte string)<br><br>A unique Ethernet Hardware Address information |
| 9 ~10 | - | IP Address (in network byte order)<br><br>Specified IP Address information is stored. |
| 11 ~12 | - | Subnet Mask (in network byte order)<br><br>Specified Subnet mask information is entered. |
| 13 ~14 | - | Gateway Address (in network byte order)<br><br>Specified Gateway address is entered. |
| 15 ~ 16 | | Broadcast Address (in network byte order)<br><br>The Broadcast Address to send multicast messages is stored. |
| 17 ~ 18 | | Primary Name Server (in network byte order)<br><br>Specified Primary Name Server is stored. |
| 19 ~ 20 | - | Secondary Name Server (in network byte order)<br><br>Secondary Name Server information is entered. |
| 21 ~ 52 | - | Default Domain Name<br><br>Default domain name is entered. |

**General Channel Status Blcok**

| Word | Bit | Description |
|---|---|---|
| 53 ~ 84 | - | SNMP Contact<br><br>SNMP Contact information is stored. |
| 85 ~ 116 | | SNMP Location<br><br>SNMP Location information is stored. |
| 117 | - | Message Connection Timeout<br><br>The amount of Timeout of the MSG instruction is stored. The MSG Connection Timeout has a range of 250 ms ~ 65,500 ms. |
| 118 | - | Message Reply Timeout<br><br>The amount of Replay Timeout of hat MSG instruction is stored. The MSG Reply Timeout has a range of 250 ms ~ 65,500 ms. |
| 119 | - | Inactivity Timeout<br><br>The amount of remain time of the MSG instruction is displayed before it is terminated. The Inactivity Timeout has a range of 1~ 65,500 minutes. |

## Diagnostic Counter Block of Communications Status

The followings are the elements of the Diagnostic Counter Block for EtherNet.



| Word | Bit | Description |
|---|---|---|
| 120 | - | DLL Diagnostic Counters Category Identifier Code (always 2) |
| 121 | - | Length: 110 (55 words to follow including format code) |
| 122 | - | Counters Format Code: Ethernet (always 0) |
| 123 | Low word | RMON Rx Octets |
| 124 | High word | (RMON_R_OCTETS) |
| 125 | Low word | RMON Tx Octets |
| 126 | High word | (RMON_T_OCTETS) |
| 127 | Low word | RMON Rx Packets |
| 128 | High word | (RMON_R_PACKETS) |
| 129 | Low word | RMON Tx Packets |
| 130 | High word | (RMON_T_PACKETS) |
| 131 | Low word | Frames Transmitted with Excessive Collisions |
| 132 | High word | (IEEE_T_EXCOL) |
| 133 | Low word | Frames Received with CRC Error |
| 134 | High word | (IEEE_R_CRC) |
| 135 | Low word | Frames Received with Alignment Error |
| 136 | High word | IEEE_R_ALIGN) |
| 137 | Low word | Count of frames not counted correctly |
| 138 | High word | RMON_T_DROP) |
| 139 | Low word | Receive FIFO Overflow Count |
| 140 | High word | (IEEE_R_MACERR) |
| 141 | Low word | Frames transmitted with Tx FIFO Under-run |
| 142 | High word | (IEEE_T_MACERR) |
| 143 | Low word | Frames Transmitted with Single Collision |
| 144 | High word | IEEE_T_1COL) |
| 145 | Low word | Frames Transmitted with Multiple Collisions |
| 146 | High word | (IEEE_T_MCOL) |
| 147 | Low word | Frames Transmitted with Deferral Delay |
| 148 | High word | IEEE_T_DEF) |
| 149 | Low word | Frames Transmitted with Late Collisions |
| 150 | High word | IEEE_T_LCOL) |
| 151 | Low word | Frames Transmitted with Carrier Sense Errors |
| 152 | High word | (IEEE_T_CSERR) |

| Word | Bit | Description |
|---|---|---|
| 153 | Low word | RMON Tx Collision Count |
| 154 | High word | RMON_T_COL) |
| 155 | Low word | Total Commands Sent |
| 156 | High word | |
| 157 | Low word | Total Commands Received |
| 158 | High word | |
| 159 | Low word | Total Replies Sent |
| 160 | High word | |
| 161 | Low word | Total Replies Received |
| 162 | High word | |
| 163 | Low word | Total Replies Sent with Error |
| 164 | High word | |
| 165 | Low word | Total Replies Received with Error |
| 166 | High word | |
| 167 | Low word | Total Replies Timed Out |
| 168 | High word | |
| 169 | Low word | Total Message Connections |
| 170 | High word | |
| 171 | Low word | Incoming Message Connections |
| 172 | High word | |
| 173 | Low word | Outgoing Message Connections |
| 174 | High word | |
| 175 | Low word | Maximum Connections Allowed |
| 176 | High word | |

# Programming Instruction Overview

## Introduction

This chapter describes **X8 Series PLC** Programming Insturcution Overview. The chapter is organized as follows.

## X8 Instruction Group

The following table shows the X8 Series PLC programming instructions listed within their functional group.

| Instruction Group | Descriptions |
|---|---|
| High-Speed Counter Input | HSCS, HSCC<br>The high-speed counter instructions allow you to monitor and control the high-speed counter. Generally used with DC inputs. |
| PTP/PWM Output | PTO, PWM<br>The PTO and PWM instructions allow you to monitor and control the data. Generally used with DC inputs. |
| Relay-Type (Bit) | NO, NC, OUT, SET, RST, ONSR, EGR, ONSF<br>Relay and Contact instructions allow you to monitor the input contact and control the status bit. |
| Timer and Counter | TON, TOF, TONR, CTU, CTD, RSTA<br>Timer and Counter instructuions allow you to control operations based on time or the number of events. |
| Compare | EQ, NE, LT, LE, GT, GE, EQM, LIM<br>The compare instructions compare values by using a specific compare operation. |
| Math | ADD, SUB, MUL, DIV, NEG, CLR, ABS, SQRT, SCAL, SCAP, SWAP, CALC, COS, ATAN, ASIN, ACOS, SIN, TAN, POW, LN, LOG, DEG, RAD<br>The math instructions supports various arithmetic operations. |
| Application Specific | CLKR, CLKC<br>This is instruction for perfomance diagonsis and read and compare clock data. |
| Conversion | DECO, ENCO, BCD, BIN, GRAY<br>The Conversion instruction allows you multiplex and de-multiplex , gray code and conversion between binary and decimal (BCD) values. |
| Logical | AND, OR, XOR, NOT<br>The Logical instructions perform bit-wise logical operations on words. |
| Move | MOV, MOVM<br>The Move instructions modify and move words. |
| Data Table | COPW, COPT, FILT, BSL, BSR, FIFOL, FIFOU, LIFOL, LIFOU, SWAP<br>The Data Table Instructions perform operations on data table. |
| Sequencer | SEQIC, SEQO, SEQL<br>The Sequencer instructions are used to control repeatable equipments or prcess. |
| Program Control | JUMP, LBL, CALL, SBR, RET, SUSP, ENDT, MCR, END<br>The Program Control instructions allow you to control the ladder program execution. |
| Input and Output | REFI, REFO, EOS<br>The Input and Output insturctions calculate input and output values regardless of input and output scan. |
| Interrupt | PITS, INT, INTD, INTE, INTF<br>The user interrupt instructions allow you to interrupt your program based on defined events. |
| PID | PID<br>The process control instruction provides closed-loop control. |
| ACSCII | ARNL, ARNC, ATOI, ACB, ACN, AEX, AHS, ITOA, ARC, ARL, ASCH, ACMP, AWA, AW<br>ACSCII insturction convert and write ASCII strings. |
| Communication | MSG, SVC<br>The Communication instructions read or write data to another equipment. |
| Recipe | RCP<br>The Recipe insturction allows you to transfer data between the recipe database and a set of user data table element. |
| Data Logging | DLOG<br>The Data Logging instruction allow you to log time and data. |

## Instruction Descriptions

Throughout this manual, descriptions for all instructions have a similar table below and show applicable input condition and output condition of each insturction.

(For example) applicable addressing mode and data table.

| Parameter | Data Files | | | | | | | | | | | | | | SFR | | | | | | | | | Address Mode | | | Address Level | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Y | X | SR | B | TM, CT, CR | N | F | ST | A | L | MG, PD | RP | PLS | ASCII | RTC | HSC | PTO, PWM | STI | EII | BHI | MMI | CS-Comms | DLS-DataLog | Immediate | Direct | Indirect | Bit | Word | LongWord | Element |
| **Input A** | √ | √ | √ | √ | √ | √ | √ | √ | √ | √ | √ | | √ | | √ | √ | √ | √ | √ | √ | √ | √ | √ | √ | √ | √ | | √ | √ | |
| **Input B** | √ | √ | √ | √ | √ | √ | √ | √ | √ | √ | √ | | √ | | √ | √ | √ | √ | √ | √ | √ | √ | √ | √ | √ | √ | | √ | √ | |
| **Output** | √ | √ | √ | √ | √ | √ | √ | √ | √ | √ | √ | | √ | | √ | √ | √ | √ | √ | | | | | | √ | √ | | √ | √ | |

**IMPORTANT** You cannot use indirect addressing with: SR, MG, PD, RTC, HSC, PTO, PWM, STI, EII, BHI, MMI, CS, and DLS data file.

## Addressing Modes

The **X8 Series PLC** supports three types of data addressing:

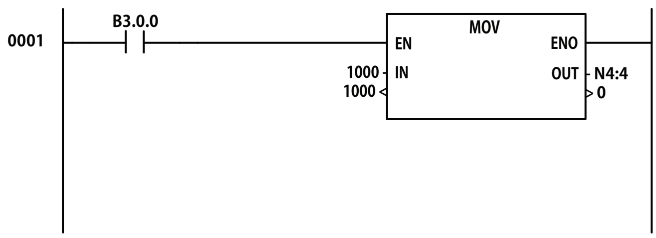
- Immediate
- Direct
- Indirect

The **X8 Series PLC** do not support indexed addressing. Indexed addressing can be duplicated with indirect addressing.

Each of the addressing modes are described below.

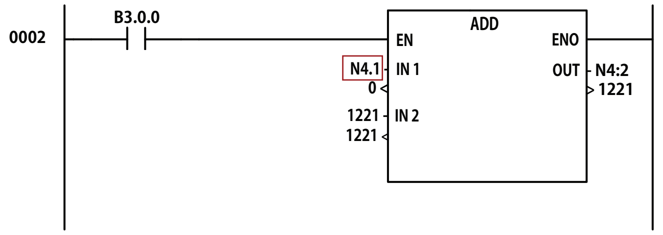
### Immediate Addressing

Immediate addressing is the most basic addressing mode and primarily used to assign numeric constants directly to **X8 Series PLC** data table.



## Direct Addressing

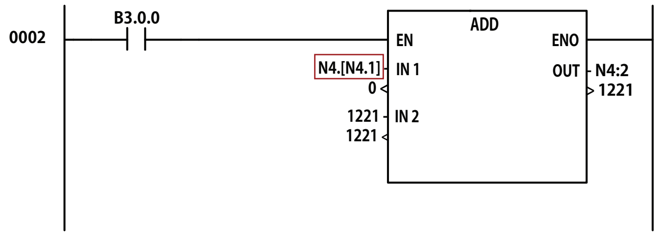
When you use direct addressing, you can input a specific data table elements of internal X8 Series PLC (below is an example of N4.1) as shown in the example below.




## Indirect Addressing

Indirect addressing allows components within the address to be used as pointers to other data locations within the controller.
This functionality can be especially useful for certain types of applications, recipemanagement, batch processing and many others.

To define which components of an address are to be indirected, a closed bracket "[ ]" is used.




The above example is the ladder program that adds 1221 to data table value that is expressed N4.[N4.1] and stores it to N4.2.

If the value of location N4.1 = 10, this instruction copies data value from data table specified by N4.10 and stores it to location N4.2 after adding 1221 to the data value.

In this example, the range of N4.1 which is supported by X8 Series PLC is from 0 to 1534, and total is 1535.

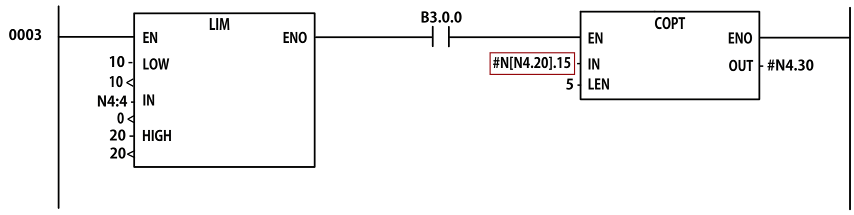
**TIP**

If you access to a number lager than the number of elements specified from the data table through Indrect Addressing method, 28 H error (runtime error and ladder progrm error) will be occurred.

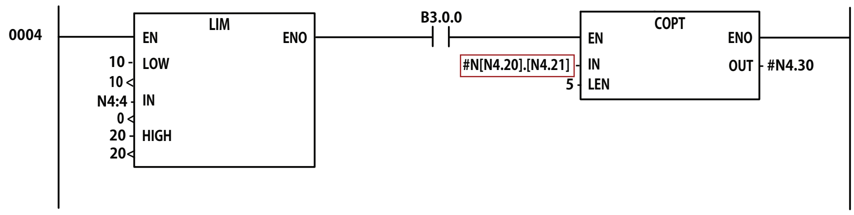
For the initial compiling, any errors could not be founded because there is no grammatical errors.

## Specifies the data table through Indirect Addressing

Specify the data table through Indirect Addressing and can refer the data.

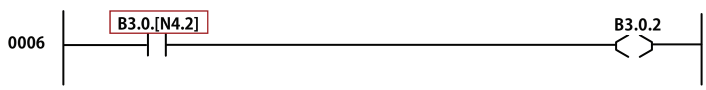



And like below, you can detailed control like double pointer in the C language since the data table and the elements supports the Indirect Addressing.




## Indirect Addressing of a File

It can be applied like above.



**TIP** Do not allocate 4,096 or more in the Indirect Addressing method.

# Using the High-Speed Counter and Programmable Limit Switch

## Introduction

This chapter describes high-speed counter and programmable limit switch. This chapter organized as follows:

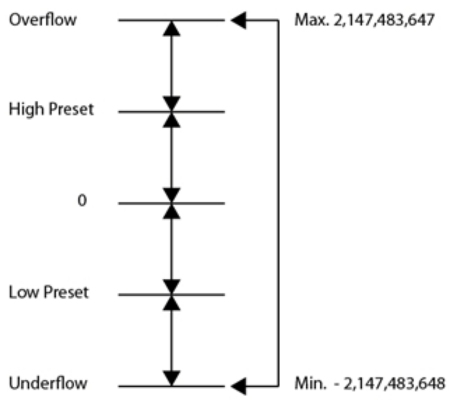
## High-Speed Counter Overview

All **X8 Series PLC** have six 100kHz high-speed counters. There are three main high-Speed counters (counter 0, 1, 2) and three sub high speed counters (counter 3, 4, 5). Each main high-speed counter has four dedicated inputs and each sub high-speed counter has two dedicated inputs.

Depending on the high-speed usage mode, the sub high-speed counter may not be able to use.

High-speed counter specifications are as below:

- 100 kHz operation
- High-speed direct control of outputs
- 32-bit data (count range of ± 2,147,483,647)
- Programmable High and Low presets, and Overflow and Underflow Set-points
- Automatic Interrupt processing based on accumulated count
- Run-time editable parameters (from the user control program)




| | |
|---|---|
| **IMPORTANT** | The HSC function can only be used with the controller's embedded I/O. It cannot be used with expansion I/O modules. |

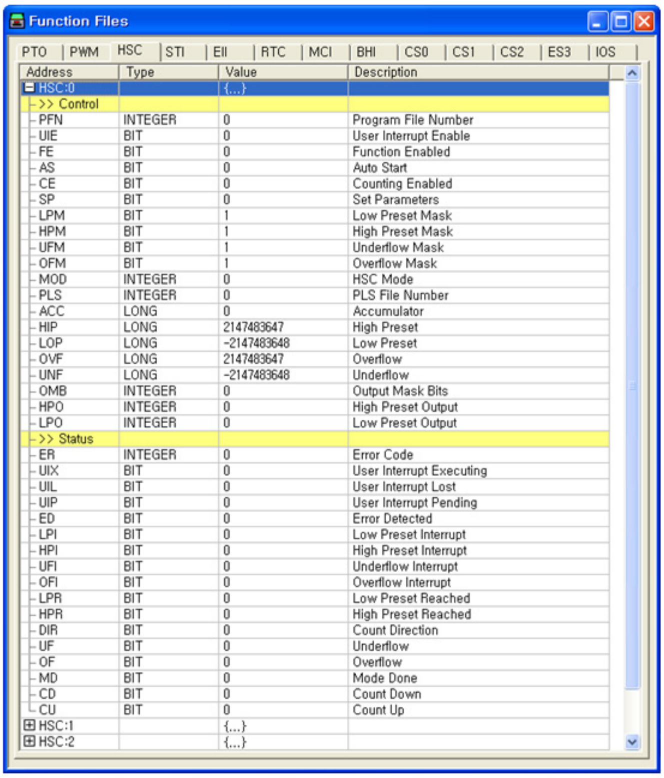
## Programmable Limit Switch

**X8 Series PLC** supports the Programmable Limit Switch function and it allows you to configure the High-Speed Counter to operate as a programmable limit switch or rotary cam switch.

## High-Speed Counter (HSC) Function Register

SFR (Special Function Register) of internal XGPC data table allows you to access to HSC configuration data and control program.

If the PLC is in the run mode, the sub-elements of internal HSC SFR may be changing.



The HSC function, along with the PTO and PWM instructions, are different than most other controller instructions. Their operation is performed by custom circuitry that runs in parallel with the main system processor. This is necessary because of the high performance requirements of these functions.

The HSC is extremely versatile; the user can select or configure the master HSC for any one of ten modes and the sub HSC for any one of five modes of operation.

## High-Speed Counter SFR Elements

The HSC SFR of **X8 Series PLC** is comprised of 37 sub-elements for control and status output .

All examples illustrate based on HSC.0.

| Address | Data Format | HSC Modes(1) | Function | User Program Access |
|---|---|---|---|---|
| HSC.0.LadderProgramNumber | Word (INT) | 0...9 | Control | Read Only |
| HSC.0.UserInterruptEnable | Bit | 0...9 | Control | Read / Write |
| HSC.0.FunctionEnabled | Bit | 0...9 | Control | Read / Write |
| HSC.0.AutoStart | Bit | 0...9 | Control | Read Only |
| HSC.0.CountingEnabled | Bit | 0...9 | Control | Read / Write |
| HSC.0.SetParameters | Bit | 0...9 | Control | Read / Write |

| Address | Data Format | HSC Modes(1) | Function | User Program Access |
|---|---|---|---|---|
| HSC.0.LowPresetMask | Bit | 2...9 | Control | Read / Write |
| HSC.0.HighPresetMask | Bit | 0...9 | Control | Read / Write |
| HSC.0.UnderflowMask | Bit | 2...9 | Control | Read / Write |
| HSC.0.OverflowMask | Bit | 0...9 | Control | Read / Write |
| HSC.0.Mode | Word (INT) | 0...9 | Control | Read Only |
| HSC.0.PlsTableNumber | Word (INT) | 0...9 | Control | Read / Write |
| HSC.0.Accumulator | long word (32-bit INT) | 0...9 | Control | Read / Write |
| HSC.0.HighPreset | long word (32-bit INT) | 0...9 | Control | Read / Write |
| HSC.0.LowPreset | long word (32-bit INT) | 2...9 | Control | Read / Write |
| HSC.0.Overflow | long word (32-bit INT) | 0...9 | Control | Read / Write |
| HSC.0.Underflow | long word (32-bit INT) | 2...9 | Control | Read / Write |
| HSC.0.OutputMaskBits | Word (16 bit Binary) | 0...9 | Control | Read Only |
| HSC.0.HighPresetOutput | Word (16 bit Binary) | 0...9 | Control | Read / Write |
| HSC.0.LowPresetOutput | Word (16 bit Binary) | 2...9 | Control | Read / Write |
| HSC.0.ErrorCode | Word (INT) | 0...9 | Status | Read Only |
| HSC.0.UserInterruptExecuting | Bit | 0...9 | Status | Read Only |
| HSC.0.UserInterruptLost | Bit | 0...9 | Status | Read / Write |
| HSC.0.UserInterruptPending | Bit | 0...9 | Status | Read Only |
| HSC.0.ErrorDetected | Bit | 0...9 | Status | Read Only |
| HSC.0.LowPresetInterrupt | Bit | 2...9 | Status | Read / Write |
| HSC.0.HighPresetInterrupt | Bit | 0...9 | Status | Read / Write |
| HSC.0.UnderflowInterrupt | Bit | 2...9 | Status | Read / Write |
| HSC.0.OverflowInterrupt | Bit | 0...9 | Status | Read / Write |
| HSC.0.LowPresetReached | Bit | 2...9 | Status | Read Only |
| HSC.0.HighPresetReached | Bit | 2...9 | Status | Read Only |
| HSC.0.Count Direction | Bit | 0...9 | Status | Read Only |
| HSC.0.Underflow | Bit | 0...9 | Status | Read / Write |
| HSC.0.Overflow | Bit | 0...9 | Status | Read / Write |
| HSC.0.Mode Done | Bit | 0 or1 | Status | Read / Write |
| HSC.0.CountDown | Bit | 2...9 | Status | Read Only |
| HSC.0.CountUp | Bit | 0...9 | Status | Read Only |

# HSC SFR Sub-Elements

37 sub-elements of the HSC SFR of **X8 Series PLC** is described. All examples illustrate based on HSC0.

## LadderProgramNumber

| Address | Data Format | HSC Mode | Function | User Program Access |
|---|---|---|---|---|
| HSC.0.LadderProgramNumber | Word(INT) | 0~9 | Control | Read / Write |

The LadderProgramNumber is subroutine number that is executed when HSC count data value is to High Preset, Low Preset, Overflow or Underflow value. A valid subroutine range is 6~511.

## UserInterruptEnable

| Address | Data Format | HSC Mode | Function | User Program Access |
|---|---|---|---|---|
| HSC.0.UserInterruptEnable | Bit | 0~9 | Control | Read / Write |

The UserInterruptEnable is used to enable or disable HSC sub-routine processing when the HSC accumulator value is reaches to the condition of High / Low Preset, Overflow, or Underflow.

Conditions are as below:

- Low preset reached
- High preset reached
- Overflow condition
- Underflow condition

## FunctionEnabled

| Address | Data Format | HSC Mode | Function | User Program Access |
|---|---|---|---|---|
| HSC.0.FunctionEnabled | Bit | 0~9 | Control | Read / Write |

The FunctionEnabled is a status/control bit that defines when the HSC interrupt is enabled, and that interrupts generated by the HSC are processed based on their priority.

This bit can be controlled by the user ladder program or is automatically set by the AutoStart bit is enabled.

## AutoStart

| Address | Data Format | HSC Mode | Function | User Program Access |
|---|---|---|---|---|
| HSC.0.AutoStart | Bit | 0~9 | Control | Read Only |

The AutoStart is stored as part of the user program and is used when the HSC function is automatically started if the X8 Series PLC is in run or test mode. The CountingEnabled bit must also be set to enable the HSC functions.

## CountingEnabled

| Address | Data Format | HSC Mode | Function | User Program Access |
|---|---|---|---|---|
| HSC.0.CountingEnabled | Bit | 0~9 | Control | Read / Write |

The Counting Enabled control bit is used to enable or disable the High-Speed Counter. When this bit is changed from set (1) to clear (0) , counting is . If this bit is disabled while the counter is running, the accumulated value is held; if the bit is then set, counting resumes.

This bit can be controlled by the user program and retains its value through a power cycle.

## SetParamters

| Description | Data Format | HSC Mode | Function | User Program Access |
|---|---|---|---|---|
| HSC.0. SetParameters | Bit | 0~9 | Control | Read / Write |

The SetParamters control bit is used to load new variables to the HSC. This bit is controlled by the user program and retains its value through a power cycle.

## LowPresetMask

| Description | Data Format | HSC Mode | Function | User Program Access |
|---|---|---|---|---|
| HSC.0. LowPresetMask | Bit | 2~9 | Control | Read/Write |

The LowPresetMask control bit is used to control the Low Preset Interrupt bit. If this bit is clear (0), the HSC user interrupt is not executed when the HSC condition is reaches to the Low Preset.

This bit is controlled by the user program and retains its value through a power cycle.

## HighPresetMask

| Description | Data Format | HSC Mode | Function | User Program Access |
|---|---|---|---|---|
| HSC.0. HighPresetMask | Bit | 0~9 | Control | Read/Write |

The HighPresetMask control bit is used to control the High Preset Interrupt bit. If this bit is clear (0), the HSC user interrupt is not executed when the HSC condition is reaches to the High Preset.

This bit is controlled by the user program and retains its value through a power cycle.

## UnderflowMask

| Description | Data Format | HSC Mode | Function | User Program Access |
|---|---|---|---|---|
| HSC.0. UnderflowMask | Bit | 2~9 | Control | Read/Write |

The UnderflowMask control bit is used to control the underflow interrupt.

If this bit is clear(0), he HSC user interrupt is not executed when the HSC condition is reaches to the High Preset.

This bit is controlled by the user program and retains its value through a power cycle.

## OverflowMask

| Description | Data Format | HSC Mode | Function | User Program Access |
|---|---|---|---|---|
| HSC.0. OverflowMask | Bit | 0~9 | Control | Read/Write |

The OverflowMask control bit is used to control the overflow interrupt.

If this bit is clear(0), he HSC user interrupt is not executed when the HSC condition is reaches to the High Preset.

This bit is controlled by the user program and retains its value through a power cycle.

## HSC Mode

| Description | Data Format | HSC Mode | Function | User Program Access |
|---|---|---|---|---|
| HSC.0. Mode | Word (INT) | 0~9 | Control | Read Only |

The Mode variable sets the High-Speed Counter to one of 10 types of operation.

This integer value is configured through the XGPC and is accessible in the X8 Series PLC as a read-only variable.

HSC0's sub counter is HSC3, HSC1's sub counter is HSC4 and HSC2's sub counter is HSC5. Each set of counters share the input port.

The following table shows the dedicated inputs for the HSCs depending on the mode.

**HSC Input Assignments**

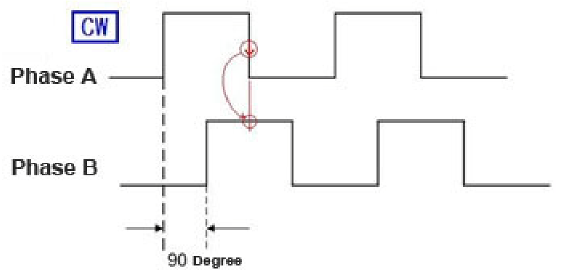
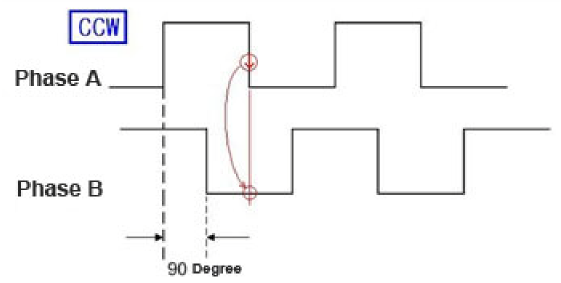
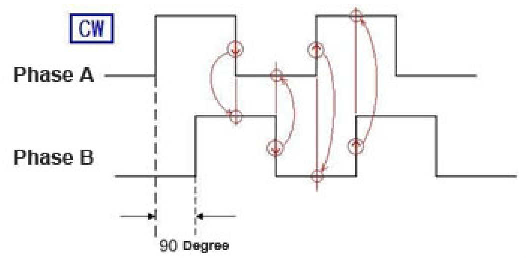
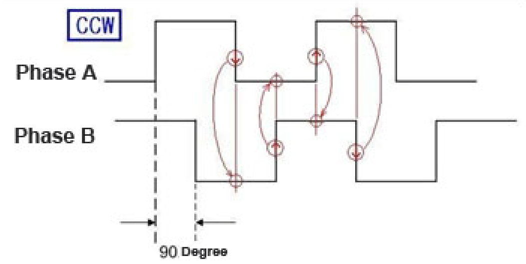
| | I.0.0.0 | I.0.0.1 | I.0.0.2 | I.0.0.3 | I.0.0.4 | I.0.0.5 | I.0.0.6 | I.0.0.7 | I.0.0.8 | I.0.0.9 | I.0.0.10 | I.0.0.11 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| HSC:0 | A/C | B/D | Reset | | | | | | | | | |
| HSC:1 | | | | | A/C | B/D | Reset | Hold | | | | |
| HSC:2 | | | | | | | | | A/C | B/D | Reset | Hold |
| HSC:3 | | | A/C | B/D | | | | | | | | |
| HSC:4 | | | | | | | A/C | B/D | | | | |
| HSC:5 | | | | | | | | | | | A/C | B/D |

**HSC Operating Modes**

| Mode Number | Type |
|---|---|
| 0 | Up Counter<br>The accumulator is immediately cleared (0) when it reaches the high preset. A low preset cannot be defined in the mode 0. |
| 1 | Up Counter with external reset and hold<br>The accumulator is immediately cleared (0) when it reaches the high preset. A low preset cannot be defined in the mode 0. |
| 2 | Counter with external direction<br>Mode for counter direction setting. |
| 3 | Counter with external direction, reset, and hold<br>Mode for counter direction, reset and hold setting. |
| 4 | Two input counter (up and down) |
| 5 | Two input counter (up and down) with external reset and hold |
| 6 | Quadrature counter (phased inputs A and B) |
| 7 | Quadrature counter (phased inputs A and B) with external reset and hold |
| 8 | Quadrature X4 counter (phased inputs A and B) |
| 9 | Quadrature X4 counter (phased inputs A and B) with external reset and hold |

* The main high-speed counters support 10 types of operation mode and the sub high-speed counters support 5 types (mode 0, 2, 4, 6, 8). If the main high-speed counter is set to mode 1, 3, 5, 7 or 9, then belong the sub high-speed counter will be disabled.

## Counting Method according to Multiplication and Phase Difference

| | Description |
|---|---|
| Phased input | Counting every falling edge in phase A, and check signal level in phase B to check direction of rotation. |
| X4 High-Speed Counter Performance | Count at falling edge and rising edge in Phase A and Phase B. Check Phase B level at rising / falling edge in Phase A and check Phase A in the rising edge and falling edge in Phase B to know the direction of rotation. |

| | I.0.0.0 | I.0.0.1 | I.0.0.2 | I.0.0.3 | I.0.0.4 | I.0.0.5 | I.0.0.6 | I.0.0.7 | I.0.0.8 | I.0.0.9 | I.0.0.10 | I.0.0.11 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| HSC:0 | A/C | B/D | Reset | | | | | | | | | |
| HSC:1 | | | | | A/C | B/D | Reset | Hold | | | | |
| HSC:2 | | | | | | | | | A/C | B/D | Reset | Hold |
| HSC:3 | | | A/C | B/D | | | | | | | | |
| HSC:4 | | | | | | | A/C | B/D | | | | |
| HSC:5 | | | | | | | | | | | A/C | B/D |

**HSC Function Operating Modes & Input Assignments**

| Modes of Operation | Input0 (HSC:0)<br>Input4 (HSC:1)<br>Input 8 (HSC:2)<br>Input 2 (HSC:3)<br>Input 6 (HSC:4)<br>Input 10 (HSC:5) | Input 1 (HSC:0)<br>Input 5 (HSC:1)<br>Input 9 (HSC:2)<br>Input 3 (HSC:3)<br>Input 7 (HSC:4)<br>Input 11 (HSC:5) | Input 2 (HSC:0)<br>Input 6 (HSC:1)<br>Input 10 (HSC:2) | Input 3 (HSC:0)<br>Input 7 (HSC:1)<br>Input 11 (HSC:2) | Mode Value in User Program |
|---|---|---|---|---|---|
| Counter with Internal Direction<br><br>(mode 1a)(1) | Count | Not Used | Not Used | Not Used | 0 |
| Counter with Internal Direction, External Reset and Hold<br><br>(mode 1b)(2) | Count | Not Used | Reset | Hold | 1 |
| Counter with External Direction<br><br>(mode 2a) (1) | Count | Direction | Not Used | Not Used | 2 |
| Counter with External Direction, Reset and Hold<br><br>(mode 2b) (2) | Count | Direction | Reset | Hold | 3 |
| Two Input Counter<br>(mode 3a) (1) | Count Up | Count Down | Not Used | Not Used | 4 |
| Two Input Counter with External Reset and Hold<br>(mode 3b) (2) | Count Up | Count Down | Reset | Hold | 5 |
| Quadrature Counter<br>(mode 4a) (1) | A Type Input | B Type Input | Not Used | Not Used | 6 |
| Quadrature Counter with External Reset and Hold<br>(mode 4b) (2) | A Type Input | B Type Input | Z Type Input | Hold | 7 |

| Modes of Operation | Input0 (HSC:0)<br>Input4 (HSC:1)<br>Input 8 (HSC:2)<br>Input 2 (HSC:3)<br>Input 6 (HSC:4)<br>Input 10 (HSC:5) | Input 1 (HSC:0)<br>Input 5 (HSC:1)<br>Input 9 (HSC:2)<br>Input 3 (HSC:3)<br>Input 7 (HSC:4)<br>Input 11 (HSC:5) | Input 2 (HSC:0)<br>Input 6 (HSC:1)<br>Input 10 (HSC:2) | Input 3 (HSC:0)<br>Input 7 (HSC:1)<br>Input 11 (HSC:2) | Mode Value in User Program |
|---|---|---|---|---|---|
| Quadrature X4 Counter (mode 5a) (1) | A Type Input | B Type Input | Not Used | Not Used | 8 |
| Quadrature X4 Counter with External Reset and Hold (2) | A Type Input | B Type Input | Z Type Input | Hold | 9 |

(1) HSC:3, HSC:4, and HSC:5 support mode 1a, 2a, 3a, 4a, and 5a only.
(2) Only valid for HSC:0, HSC:1, and HSC:2

HSC Mode 0 - Up Counter

**HSC Mode 0: Up Counter Examples**

| Input Terminals | X1.0.0.0 (HSC0) | | | | X1.0.0.0 (HSC0) | | | | X1.0.0.0 (HSC0) | | | | X1.0.0.0 (HSC0) | | | | Counting Enabled Bit | Comments |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **Function** | **Count** | | | | **Not Used** | | | | **Not Used** | | | | **Not Used** | | | | | |
| Example 1 | ↑ | | | | | | | | | | | | | | | | on (1) | HSC Accumulator + 1 count |
| Example 2 | ↑ | on (1) | ↓ | off (0) | | | | | | | | | | | | | off (0) | Hold accumulator value |

Blank cells = don't care, ↑ = rising edge, ↓ = falling edge

HSC Mode 1 - Up Counter with External Reset and Hold

**HSC Mode 1: Up Counter with External Reset and Hold Examples**

| Input Terminals | X1.0.0.0 (HSC0) | | | | X1.0.0.0 (HSC0) | | | | X1.0.0.0 (HSC0) | | | | X1.0.0.0 (HSC0) | | | | CE Bit | Comments |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **Function** | **Count** | | | | **Not Used** | | | | **Reset** | | | | **Hold** | | | | | |
| Example 1 | ↑ | | | | | | | | | on (1) | ↓ | off (0) | | | | off (0) | on (1) | HSC Accumulator +1 count |
| Example 2 | | | | | | | | | | on (1) | ↓ | off (0) | | on (1) | | | | Hold accumulator value |
| Example 3 | | | | | | | | | | on (1) | ↓ | off (0) | | | | | | Hold accumulator value |
| Example 4 | | on (1) | ↓ | off (0) | | | | | | | | | | | | | | Hold accumulator value |
| Example 5 | | | | | | | | | ↑ | | | | | | | | | Clear accumulator (=0) |

Blank cells = don’t care, ↑ = rising edge, ↓ = falling edge

## HSC Mode 2 - Counter with External Direction

**HSC Mode 2 - Counter with External Direction Example**

| Input Terminals | X1.0.0.0 (HSC0) | | | | X1.0.0.0 (HSC0) | | | | X1.0.0.0 (HSC0) | | | | X1.0.0.0 (HSC0) | | | | CE Bit | Comments |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **Function** | **Count** | | | | **Direction** | | | | **Not Used** | | | | **Not Used** | | | | | |
| Example 1 | ↑ | | | | | | | off<br>(0) | | | | | | | | | on<br>(1) | HSC Accumulator +1 count |
| Example 2 | ↑ | | | | | on<br>(1) | | | | | | | | | | | on<br>(1) | Hold accumulator value |
| Example 3 | | | | | | | | | | | | | | | | | | Hold accumulator value |

Blank cells = don’t care, ↑ = rising edge, ↓ = falling edge

## HSC Mode 3 - Counter with External Direction, Reset, and Hold

**HSC Mode 3 - Counter with External Direction, Reset, and Hold Example**

| Input Terminals | X1.0.0.0 (HSC0) | | | | X1.0.0.0 (HSC0) | | | | X1.0.0.0 (HSC0) | | | | X1.0.0.0 (HSC0) | | | | CE Bit | Comments |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **Function** | **Count** | | | | **Direction** | | | | **Reset** | | | | **Hold** | | | | | |
| Example 1 | ↑ | | | | | | | | | on<br>(1) | ↓ | off<br>(0) | | | | off<br>(0) | on<br>(1) | HSC Accumulator +1 count |
| Example 2 | ↑ | | | | | on<br>(1) | | | | on<br>(1) | ↓ | off<br>(0) | | | | off<br>(0) | on<br>(1) | HSC Accumulator -1 count |
| Example 3 | | | | | | | | | | on<br>(1) | ↓ | off<br>(0) | | on<br>(1) | | | | Hold accumulator value |
| Example 4 | | | | | | | | | | on<br>(1) | ↓ | off<br>(0) | | | | | off<br>(0) | Hold accumulator value |
| Example 5 | | on<br>(1) | ↓ | off<br>(0) | | | | | | on<br>(1) | ↓ | off<br>(0) | | | | | | Hold accumulator value |
| Example 6 | | | | | | | | | ↑ | | | | | | | | | Clear accumulator (=0) |

Blank cells = don’t care, ↑ = rising edge, ↓ = falling edge

## HSC Mode 4 - Two Input Counter (up and down)

**HSC Mode 4 Examples**

| **Input Terminals** | **X1.0.0.0 (HSC0)** | | | | **X1.0.0.0 (HSC0)** | | | | **X1.0.0.0 (HSC0)** | | | | **X1.0.0.0 (HSC0)** | | | | **CE Bit** | **Comments** |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **Function** | **Count Up** | | | | **Count Down** | | | | **Not Used** | | | | **Not Used** | | | | | |
| Example 1 | ↑ | | | | | on (1) | ↓ | off (0) | | | | | | | | | on (1) | HSC Accumulator +1 count |
| Example 2 | | on (1) | ↓ | off (0) | | | | | | | | | | | | | on (1) | HSC Accumulator -1 count |
| Example 3 | | | | | | | | | | | | | | | | | off (0) | Hold accumulator value |

Blank cells = don't care, ↑ = rising edge, ↓ = falling edge

## Mode 5 - Two Input Counter (up and down) with External Reset and Hold

| **Input Terminals** | **I1:0.0/0 (HSC0)** | | | | **I1:0.0/1 (HSC0)** | | | | **I1:0.0/2(HSC0)** | | | | **I1:0.0/3(HSC0)** | | | | **CE Bit** | **Comments** |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **Function** | **Count** | | | | **Direction** | | | | **Reset** | | | | **Hold** | | | | | |
| Example 1 | ↑ | | | | | on (1) | ↓ | off (0) | | on (1) | ↓ | off (0) | | | | off (0) | on (1) | HSC Accumulator +1 count |
| Example 2 | | on (1) | ↓ | off (0) | ⇑ | | | | | on (1) | ↓ | off (0) | | | | off (0) | on (1) | HSC Accumulator -1 count |
| Example 3 | | | | | | | | | | on (1) | ↓ | off (0) | | on (1) | | | | Hold accumulator value |
| Example 4 | | | | | | | | | | on (1) | ↓ | off (0) | | | | | off (0) | Hold accumulator value |
| Example 5 | | on (1) | ↓ | off (0) | | | | | | on (1) | ↓ | off (0) | | | | | | Hold accumulator value |
| Example 6 | | | | | | | | | ↑ | | | | | | | | | Clear accumulator (=0) |

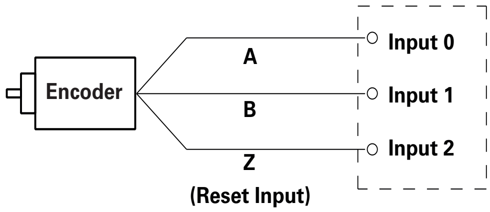
Blank cells = don't care, ↑ = rising edge, ↓ = falling edge

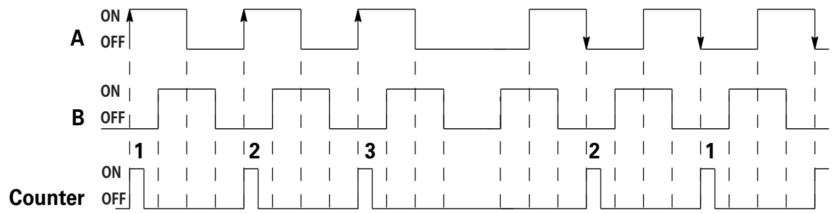
## Using the Encoder

The Encoder is used for determining direction of rotation and position for rotating.

The figure below shows a encoder connected to inputs 0, 1, and 2. The count direction is determined by the phase angle between A and B. If counter input A leads counter input B, the counter increments. If the counter input B leads the counter input A, the counter decrements.

The High-speed counter can be reset using the Z input. The Z outputs from the encoders typically provide one pulse per revolution.




HSC Mode 6 - Quadrature Counter (phased inputs A and B)

**HSC Mode 6 - Quadrature Counter (phased inputs A and B) Example**

| Input Terminals | X1.0.0.0 (HSC0) | | | | X1.0.0.0 (HSC0) | | | | X1.0.0.0 (HSC0) | | | | X1.0.0.0 (HSC0) | | | | CE Bit | Comments |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **Function** | **Count A** | | | | **Count B** | | | | **Not Used** | | | | **Not Used** | | | | | |
| Example 1 | ↑ | | | | | | | off (0) | | | | | | | | | on (1) | HSC Accumulator +1 count |
| Example 2 | | | ⇓ | | ⇑ | | | off (0) | | | | | | | | | on (1) | HSC Accumulator -1 count |
| Example 3 | | | | off (0) | | | | | | | | | | | | | | Hold accumulator value |
| Example 4 | | on (1) | | | | | | | | | | | | | | | | Hold accumulator value |
| Example 5 | | | ↓ | | | on (1) | | | | | | | | | | | | Hold accumulator value |
| Example 6 | | | | | | | | | | | | | | | | | off (0) | Hold accumulator value |

(1) Count input A leads count input B.
(2) Count input B leads count input A.
Blank cells = don't care, ↑ = rising edge, ↓ = falling edge

## HSC Mode 7 - Quadrature Counter (phased inputs A and B) With External Reset and Hold

**HSC Mode 7 - Quadrature Counter (phased inputs A and B) With External Reset and Hold Example**

| Input Terminals | X1.0.0.0 (HSC0) | | | | X1.0.0.0 (HSC0) | | | | X1.0.0.0 (HSC0) | | | | X1.0.0.0 (HSC0) | | | | CE Bit | Comments |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **Function** | **Count A** | | | | **Count B** | | | | **Z reset** | | | | **Hold** | | | | | |
| Example 1(1) | ↑ | | | | | | | off (0) | | | | | | | | off (0) | on (1) | HSC Accumulator +1 count |
| Example 2(2) | | | ↓ | | ↑ | | | off (0) | | | | off (0) | | | | off (0) | on (1) | HSC Accumulator -1 count |
| Example 3 | | | ↓ | off (0) | | | | off (0) | | on (1) | | | | | | | | Hold accumulator value |
| Example 4 | | on (1) | | | | | | | | | | | | | | | | Hold accumulator value |
| Example 5 | | | | | | on (1) | | | | | | | | | | | | Hold accumulator value |
| Example 6 | | | | | | | | | | | | off (0) | | on (1) | | | | Hold accumulator value |
| Example 7 | | | | | | | | | | | | off (0) | | | | | off (0) | |

1) Count input A leads count input B.
(2) Count input B leads count input A.
Blank cells = don’t care, ↑ = rising edge, ↓ = falling edge

## HSC Mode 8 - Quadrature X4 Counter

**HSC Mode 8 - Quadrature X4 Counter**

| X1.0.0.1(HSC0) (A) | X1.0.0/1(HSC0) (B) | Value of CE Bit | Accumulator and Counter Action |
|---|---|---|---|
| ▲ | OFF | TRUE | Count Up Acc. Value |
| ▲ | ON | TRUE | Count Down Acc. Value |
| ▼ | OFF | TRUE | Count Down Acc. Value |
| ▼ | ON | TRUE | Count Up Acc. Value |
| OFF | ▲ | TRUE | Count Down Acc. Value |
| ON | ▲ | TRUE | Count Up Acc. Value |
| OFF | ▼ | TRUE | Count Up Acc. Value |
| ON | ▼ | TRUE | Count Down Acc. Value |
| OFF or ON | OFF or ON | X | Hold Acc.Value |
| X | X | FALSE | Hold Acc.Value |

## HSC Mode 9 - Quadrature X4 Counter with External Reset and Hold

**HSC Mode 9 - Quadrature X4 Counter with External Reset and Hold Example**

| X1.0.0.1(HSC0) (A) | X1.0.0/1(HSC0) (B) | X1.0.0.2(HSC0) (reset) | X1.0.0.3(HSC0) (Hold) | Value of CE Bit | Accumulator and Counter Action |
|---|---|---|---|---|---|
| ▲ | OFF | X | - | TRUE | Count Up Acc. Value |
| ▲ | ON | X | - | TRUE | Count Down Acc. Value |
| ▼ | OFF | X | - | TRUE | Count Down Acc. Value |
| ▼ | ON | X | - | TRUE | Count Up Acc. Value |
| OFF | ▲ | X | - | TRUE | Count Down Acc. Value |
| ON | ▲ | X | - | TRUE | Count Up Acc. Value |
| OFF | ▼ | X | - | TRUE | Count Up Acc. Value |
| ON | ▼ | X | - | TRUE | Count Down Acc. Value |
| OFF or ON | OFF or ON | OFF | X | X | Hold Acc.Value |
| OFF | OFF | ON | X | X | Reset Acc.to Zero |
| X | X | OFF | ON | X | Hold Acc.Value |
| X | X | OFF | X | FALSE | Hold Acc.Value |

## Accumulator

| Description | Data Format | HSC Mode | Function | User Program Access |
|---|---|---|---|---|
| HSC.0.Accumulator | Long Word (32 bit INT) | | Control | Read/Write |

The Accumulator contains the measuring results of the HSC. If the HSC mode 0.1 is configured, or a high preset is reached or an overflow condition is detected, the value of the accumulator is cleared (0).

## HighPreset

| Description | Data Format | HSC Mode | Function | User Program Access |
|---|---|---|---|---|
| HSC.0.HighPreset | Long Word (32 bit INT) | | Control | Read/Write |

The HighPreset is the high preset configuration value of the counter that is used in the HSC.

When the SetParameter bit is changed from 0 to 1, the HighPreset value is loaded into the HSC so that can be operated from the HSC.

The HighPreset value must be less than or equal to the data resident in the Overflow parameter or an HSC error is generated.

Set various HSC parameters using the HSCS instruction except SetParamter.

## LowPreset

| Description | Data Format | HSC Mode | Function | User Program Access |
|---|---|---|---|---|
| HSC.0.LowPreset | Long Word (32 bit INT) | | Control | Read/Write |

The LowPreset is the low preset configuration value of the counter that is used in the HSC.

When the SetParameter bit is changed from 0 to 1, the HighPreset value is loaded into the HSC so that can be operated from the HSC.

The HighPreset value must be less than or equal to the data resident in the Underflow parameter or an HSC error is generated. If negative number is input as the LowPreset or Underflow value, it will be replaced by the absolute value without sign.

Set various HSC parameters using the HSCS instruction except SetParamter.

## Overflow

| Description | Data Format | HSC Mode | Function | User Program Access |
|---|---|---|---|---|
| HSC.0.Overflow | Bit | 0~9 | Status | Read/Write |

The Overflow flag is set when the accumulated value exceeds the high preset or Overflow value. If this flag is set, any PLC error does not generated.

## Underflow

| Description | Data Format | HSC Mode | Function | User Program Access |
|---|---|---|---|---|
| HSC.0.Underflow | Bit | 0~9 | Status | Read/Write |

The Underflow flag is set when the accumulated value decreases the lower setpoint or Underflow value. If this flag is set, any PLC error does not generated.

## OutputMaskBits

| Description | Data Format | HSC Mode | Function | User Program Access |
|---|---|---|---|---|
| HSC.0.OutputMaskBits | Word (16 bit binary) | | Status | Read Only |

The OutputMaskBits define which outputs on the PLC can be directly controlled by the HSC. The HSC sub-system has the ability to directly (without control program interaction) turn outputs ON or OFF based on the HSC accumulator reaching the High or Low presets. The bit pattern stored in the OMB variable defines which outputs are controlled by the HSC and which outputs are not controlled by the HSC.

The bit pattern of the OMB variable directly corresponds to the output bits on the controller. Bits that are set (1) are enabled and can be turned on or off by the HSC sub-system. Bits that are clear (0) cannot be turned on or off by the HSC sub-system. The mask bit pattern can be configured only during initial setup.

This table illustrates this relationship:

Affect of HSC Output Mask on Base Unit Outputs

**OutputMaskBis Output Example**

| Output Example | 16-Bit Signed Integer Data Word | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | **15** | **14** | **13** | **12** | **11** | **10** | **9** | **8** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| HSC:0.HPO (high preset output) | | | | | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 1 |
| | | | | | | | | | | | | | | | | |
| HSC:0.OMB (output mask) | | | | | 0 | 1 | 1 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 1 |
| | | | | | | | | | | | | | | | | |
| Y0.0.0 | | | | | 0 | **0** | **1** | 0 | 0 | 0 | **0** | **1** | 0 | 0 | **0** | **0** |

## HighPresetOutput

| Description | Data Format | HSC Mode | Function | User Program Access |
|---|---|---|---|---|
| HSC.0.HighPresetOutput | Word (16 bit binary) | | Control | Read/Write |

The HighPresetOutput shows the state (1 = ON or 0 = OFF) of the outputs on the controller when the accumulator value is reaches to the high preset. However, actual output value is defined by OutputMaskBits.

## LowPresetOutput

| Description | Data Format | HSC Mode | Function | User Program Access |
|---|---|---|---|---|
| HSC.0.LowPresetOutput | Word (16 bit binary) | | Control | Read/Write |

The LowPresetOutput shows the state (1 = ON or 0 = OFF) of the outputs on the controller when the accumulator value is reaches to the low preset. However, actual output value is defined by OutputMaskBits.

## ErrorCode

| Description | Data Format | HSC Mode | Function | User Program Access |
|---|---|---|---|---|
| HSC:0.ErrorCode | Word (INT) | 0~9 | Status | Read Only |

The ErrorCode stores the error codes when a HSC error is generated.

The table below explains the error codes.

| Error Code | Error Name | Mode | Description |
|---|---|---|---|
| 1 | Invalid Data Table Number | None | Interrupt (program) data table identified in HSC:0.PFN is less than 3, greater than 255, or does not exist |
| 2 | Invalid HSC Mode | None | This error is generated when invalid HSC mode is configured. |

| Error Code | Error Name | Mode | Description |
|---|---|---|---|
| 3 | Invail ed High Preset | 0,1 | If the high preset value is less than 0 |
| 4 | Invalid High Preset | 2~9 | This error is generated when the high preset value is less than or equal to the low preset. |
| 5 | In va il ed Underflow Value Configuration | 2~9 | This error is generated when the low preset value is less than Underflow. |

## ErrorDetected

| Description | Data Format | HSC Mode | Function | User Program Access |
|---|---|---|---|---|
| HSC:0.ErrorDetected | Bit | 0~9 | Status | Read Only |

The ErrorDetected flag is ON when a HSC error is generated. When this bit is set, the user should look at the error information in the ErrorCode flag. This bit is automatically set and cleared by the controller.

## UserInterruptExecuting

| Description | Data Format | HSC Mode | Function | User Program Access |
|---|---|---|---|---|
| HSC:0.UserInterruptExecuting | Bit | 0~9 | Status | Read Only |

The UserInterruptExecuting flag is set when the internal processing is begin by the conditions below.

- low preset reaches
- high preset reaches
- HSC value covers the Overflow condition
- HSC value covers the Underflow condition

This flag is used to detect in the user control program if an HSC interrupt is executing and will be automatically cleared when the PLC completes its processing.

## UserInterruptPending

| Description | Data Format | HSC Mode | Function | User Program Access |
|---|---|---|---|---|
| HSC:0.UserInterruptPending | Bit | 0~9 | Status | Read Only |

The UserInterruptPending status flag that represents an interrupt is pending by the specific conditions.

This flag is used to detect in the user control program if an HSC interrupt is executing and will be automatically cleared when the PLC completes its processing.

## UserInterruptLost

| Description | Data Format | HSC Mode | Function | User Program Access |
|---|---|---|---|---|
| HSC:0.UserInterruptLost | Bit | 0~9 | Status | Read/Write |

The UserInterruptLost status flag that represents an interrupt has been lost.

This bit is set by the PLC and applied by the user program.

## LowPresetInterrupt

| Description | Data Format | HSC Mode | Function | User Program Access |
|---|---|---|---|---|
| HSC:0.LowPresetInterrupt | Bit | 2~9 | Status | Read/Write |

The LowPresetInterrupt status bit is set when the HSC accumulator reaches the low preset value and the HSC interrupt has been triggered.

This bit can be used in the control program to identify that the low preset condition caused the HSC interrupt and this flag would be applied by the control program.

This bit can be cleared by the control program and is also be cleared by the HSC whenever these conditions are detected:

- . High Preset Interrupt executes
- . Underflow Interrupt executes
- . Overflow Interrupt executes

- . PLC enters an executing mode

## HighPresetInterrupt

| Description | Data Format | HSC Mode | Function | User Program Access |
|---|---|---|---|---|
| HSC:0.HighPresetInterrupt | Bit | 2~9 | Status | Read/Write |

The HighPresetInterrupt status bit is set when the HSC accumulator reaches the high preset value and the HSC interrupt has been triggered.

This bit can be used in the control program to identify that the low preset condition caused the HSC interrupt and this flag would be applied by the control program.

This bit can be cleared by the control program and is also be cleared by the HSC whenever these conditions are detected:

- High Preset Interrupt executes
- Underflow Interrupt executes
- Overflow Interrupt executes
- PLC enters an executing mode

## LowPresetReached

| Description | Data Format | HSC Mode | Function | User Program Access |
|---|---|---|---|---|
| HSC:0.LowPresetReached | Bit | 2~9 | Status | Read/Write |

The LowPresetReached flag is set when the HSC accumulator value is less than or equal to the low preset.

This flag is updated automatically when the PLC enters an executing mode.

## HighPresetReached

| Description | Data Format | HSC Mode | Function | User Program Access |
|---|---|---|---|---|
| HSC:0.HighPresetReached | Bit | 2~9 | Status | Read Only |

The HowPresetReached flag is set when the HSC accumulator value is less than or equal to the high preset.

This bit is updated continuously by the HSC s whenever the PLC is in a run mode.

## UnderFlowInterrupt

| Description | Data Format | HSC Mode | Function | User Program Access |
|---|---|---|---|---|
| HSC:0.UnderFlowInterrupt | Bit | 2~9 | Status | Read Only |

The UnderFlowInterrupt flag is set when the HSC accumulator reaches to the Underflow value and the HSC interrupt is triggered.

This bit can be used in the control program to identify that the underflow condition caused the HSC interrupt and applied to the logic of the control program.

This flag can be cleared by the control program and is also cleared by the HSC whenever these conditions are detected:

- Low Preset Interrupt executes
- High Preset Interrupt executes
- Overflow Interrupt executes
- PLC enters an executing mode

## OverFlowInterrupt

| Description | Data Format | HSC Mode | Function | User Program Access |
|---|---|---|---|---|
| HSC:0.OverFlowInterrupt | Bit | 2~9 | Status | Read Only |

The OverflowInterrupt flag is set when the HSC accumulator reaches to the Overflow value and the HSC interrupt is triggered.

This bit can be used in the control program to identify that the Overflow condition caused the HSC interrupt and applied to the logic of the control program.

This flag can be cleared by the control program and is also cleared by the HSC whenever these conditions are detected:

- Low Preset Interrupt executes
- High Preset Interrupt executes

- Overflow Interrupt executes
- PLC enters an executing mode

## CountDirection

| Description | Data Format | HSC Mode | Function | User Program Access |
|---|---|---|---|---|
| HSC:0.CountDirection | Bit | 2~9 | Status | Read Only |

The CountDirection flag represents the HSC directions. When the HSC accumulator counts up, the direction flag is set. Whenever the HSC accumulator counts down, the direction flag is cleared.

If the accumulated value stops, the direction bit retains its value. The only time the direction flag changes is when the accumulated count reverses.

This bit is updated continuously by the HSC s whenever the PLC is in a run mode.

## ModeDone

| Description | Data Format | HSC Mode | Function | User Program Access |
|---|---|---|---|---|
| HSC:0.ModeDone | Bit | 0 or 1 | Status | Read/Write |

The ModeDone flag is set by the HSC when the HSC is configured for Mode 0 or Mode 1 behavior, and the accumulator counts up.

## CountDown

| Description | Data Format | HSC Mode | Function | User Program Access |
|---|---|---|---|---|
| HSC:0.CountDown | Bit | 2~9 | Status | Read Only |

The CountDown flag is used to increment or decrement a counter of the HSC mode from 2 to 9. When the CountingEnabled Bit is set, the CountDown flag is set.

## CountUp

| Description | Data Format | HSC Mode | Function | User Program Access |
|---|---|---|---|---|
| HSC:0.CountUp | Bit | 0~9 | Status | Read Only |

The CountUp flag is used to the HSC mode from 0 to 9. When the CountingEnabled Bit is set, the CountDown flag is set.

## Overflow

| Description | Data Format | HSC Mode | Function | User Program Access |
|---|---|---|---|---|
| HSC:0.Overflow | Long Word (32 bit INT) | | Status | Read/Write |

The Overflow flag specifies the High Preset of the HSC. When the accumulator value of HSC is greater than the value defined in the Overflow, the Overflow interrupt is triggered. HSC is set the Underflow value as accumulator to keep the count. The data range for the accumulator value is -2,147,483,648 ~ 2,147,483,647 .

To set the Overflow value, toggles the Setparameter bit (from OFF to ON) and the Overflow value stored in the SFR is transferred to the HSC.

The value stored in the Overflow must be greater than or equal to the High Preset or an HSC error is generated.

## Underflow

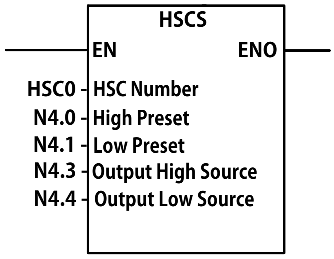
| Description | Data Format | HSC Mode | Function | User Program Access |
|---|---|---|---|---|
| HSC:0.Underflow | Long Word (32 bit INT) | | Status | Read/Write |

The Underflow flag specifies the Low Preset of the HSC. When the accumulator value of HSC is less than the value defined in the Underflow, the Underflow interrupt is triggered. HSC is set the Overflow value as accumulator to keep the count. The data range for the accumulator value is -2,147,483,648 ~ 2,147,483,647 .

To set the Underflow value, toggles the SetParameter bit (from OFF to ON) and the Underflow value stored in the SFR is transferred to the HSC.

The value stored in the Underflow must be less than or equal to the Low Preset or an HSC error is generated.

# HSCS - Set High-Speed Counter Value




Instruction Type: output

Data Size: Word, Long Word

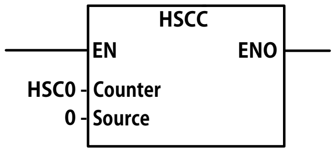
The HSCS instruction allows the high and low presets, and high and low output source to be applied to a high-speed counter. These parameters are described below:

- HSC Number
  Specifies which HSC is being used (HSC0, HSC1, HSC2, HSC3, HSC4 and HSC5)
- High Preset
  Specifies the value in the High Preset.
  - Word: -32768 ~ 32767
  - Long Word: -2,147,483,648 ~ 2,147,483,647.
- Low Preset
  Specifies the value in the Low Preset.
  - Word: -32768 ~ 32767
  - Long Word: -2,147,483,648 ~ 2,147,483,647
- Output High Source
  Data table to store the HighPresetOutputdml output register.
  The data range for the output high source is from 0 to 65,535.
- Output Low Source
  Data Table to store the LowPresetOoutputdml output.
  The data range for the output low source is from 0 to 65,535.

Valid Addressing Modes and File Types are shown below:

| Parameter | Data Table | | | | | | | | | | | | | Function Files | | | | | | | | | | Address Mode | | | Address Level | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Y | X | SR | B | TM, CT, CR | N | F | ST | A | L | MG, PD | RP | PS | ASCII | RTC | HSC | PTO, PWM | STI | EII | BHI | MMI | CS-Comms | DLS-DataLog | Immediate | Direct | Indirect | Bit | Word | LongWord | Element |
| HSC Number | | | | | | | | | | | | √ | | | | | | | | | | | | √ | | | | | | |
| High Preset | √ | √ | | √ | √ | √ | | | | √ | | √ | | | | | | | | | | | | √ | √ | √ | | √ | √ | |
| Low Preset | √ | √ | | √ | √ | √ | | | | √ | | √ | | | | | | | | | | | | √ | √ | √ | | √ | √ | |
| Output High Source | √ | √ | | √ | √ | √ | | | | √ | | √ | | | | | | | | | | | | √ | √ | √ | | √ | √ | |
| Output Low Source | √ | √ | | √ | √ | √ | | | | √ | | √ | | | | | | | | | | | | √ | √ | √ | | √ | √ | |

# HSCC - Clear High-Speed Counter Value




Instruction Type: output

The HSCC instruction resets the HSC and allows a specific value to be written to the HSC accumulator.

- Counter
  Specifies which the HSC is being used (HSC0, HSC1, HSC2, HSC3, HSC4, HSC5)
- Source
  Setting value. Input the value directly or input data table.
  The data range is from -2,147,483,648 to 2,147,483,647.

| Parameter | Data Table | | | | | | | | | | | | | Function Files | | | | | | | | | | Address Mode | | | Address Level | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Y | X | SR | B | TM, CT, CR | N | F | ST | A | L | MG, PD | RP | PS | ASCII | RTC | HSC | PTO, PWM | STI | EII | BHI | MMI | CS-Comms | DLS-DataLog | Immediate | Direct | Indirect | Bit | Word | LongWord | Element |
| HSC Number | | | | | | | | | | | | | | | | | | | | | | | | √ | | | | | | |
| Source | | | | | | | | | | √ | | | | | | | | | | | | | | √ | √ | √ | | | | |

# High-Speed Counter (HSC) Data Table

The Programmable Limit Switch function allows you to configure the HSC to operate as a rotary cam switch.

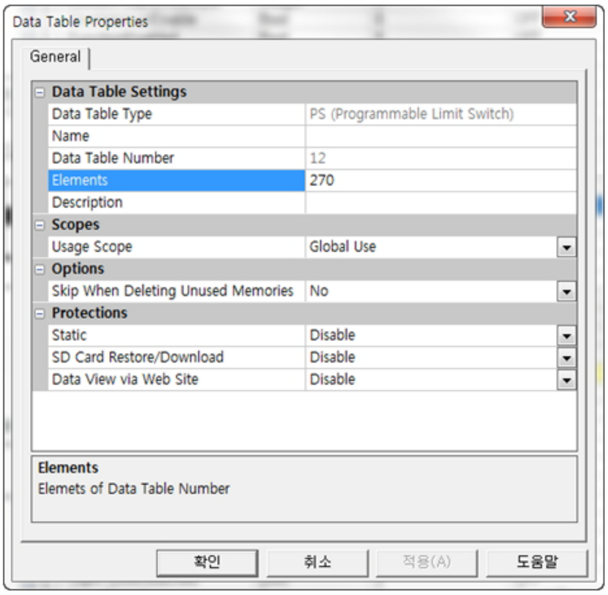
When the Programmable Limit Switch (PS) operation is enabled through the PlsTableNumber of HSC in the SFR, the PS data table can be used for Limit Position and CAM position.

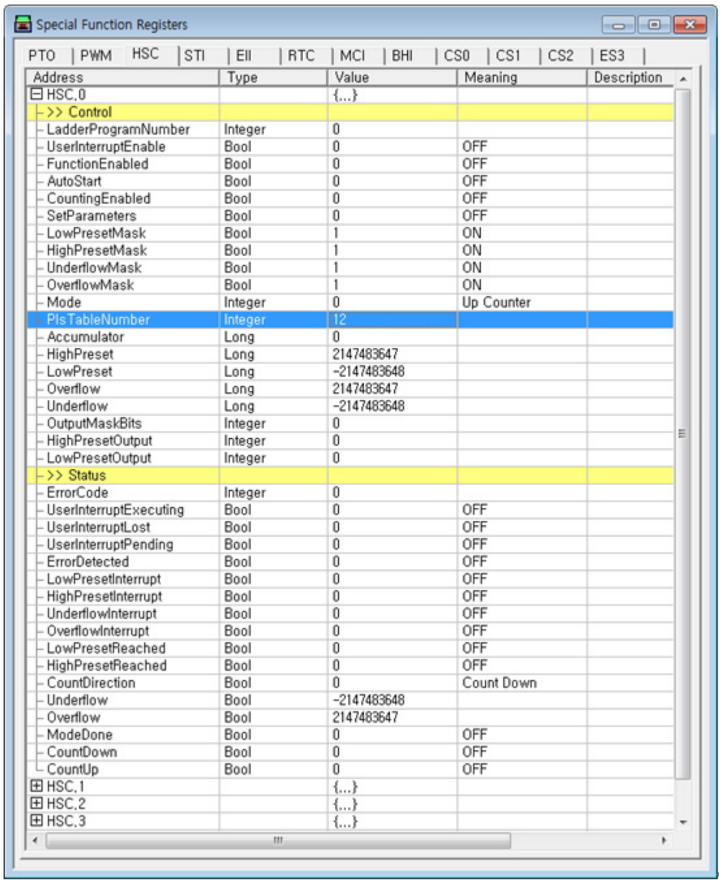
The HSC must be configured before the Programmable Limit Switch function operating.

## PS Data Table

Data files 9~1535 can be used for PS operations. Each PS data table can be up to 1535 elements long. Each element within a PS data table consumes 6 words of memory.

Following is the PS data table internal configuration information screen and an example with 270 elements.

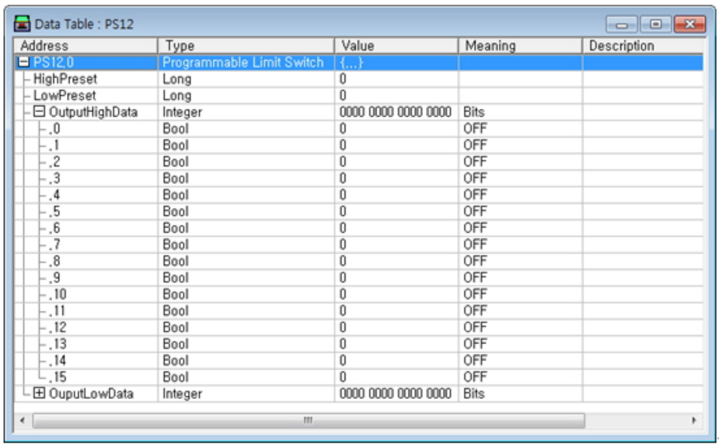




Special Function Registers

PTO | PWM | HSC | STI | EII | RTC | MCI | BHI | CS0 | CS1 | CS2 | ES3

| Address | Type | Value | Meaning | Description |
|---|---|---|---|---|
| HSC.0 | | {...} | | |
| >> Control | | | | |
| LadderProgramNumber | Integer | 0 | | |
| UserInterruptEnable | Bool | 0 | OFF | |
| FunctionEnabled | Bool | 0 | OFF | |
| AutoStart | Bool | 0 | OFF | |
| CountingEnabled | Bool | 0 | OFF | |
| SetParameters | Bool | 0 | OFF | |
| LowPresetMask | Bool | 1 | ON | |
| HighPresetMask | Bool | 1 | ON | |
| UnderflowMask | Bool | 1 | ON | |
| OverflowMask | Bool | 1 | ON | |
| Mode | Integer | 0 | Up Counter | |
| PlsTableNumber | Integer | 12 | | |
| Accumulator | Long | 0 | | |
| HighPreset | Long | 2147483647 | | |
| LowPreset | Long | -2147483648 | | |
| Overflow | Long | 2147483647 | | |
| Underflow | Long | -2147483648 | | |
| OutputMaskBits | Integer | 0 | | |
| HighPresetOutput | Integer | 0 | | |
| LowPresetOutput | Integer | 0 | | |
| >> Status | | | | |
| ErrorCode | Integer | 0 | | |
| UserInterruptExecuting | Bool | 0 | OFF | |
| UserInterruptLost | Bool | 0 | OFF | |
| UserInterruptPending | Bool | 0 | OFF | |
| ErrorDetected | Bool | 0 | OFF | |
| LowPresetInterrupt | Bool | 0 | OFF | |
| HighPresetInterrupt | Bool | 0 | OFF | |
| UnderflowInterrupt | Bool | 0 | OFF | |
| OverflowInterrupt | Bool | 0 | OFF | |
| LowPresetReached | Bool | 0 | OFF | |
| HighPresetReached | Bool | 0 | OFF | |
| CountDirection | Bool | 0 | Count Down | |
| Underflow | Bool | -2147483648 | | |
| Overflow | Bool | 2147483647 | | |
| ModeDone | Bool | 0 | OFF | |
| CountDown | Bool | 0 | OFF | |
| CountUp | Bool | 0 | OFF | |
| HSC.1 | | {...} | | |
| HSC.2 | | {...} | | |
| HSC.3 | | {...} | | |



Data Table : PS12

| Address | Type | Value | Meaning | Description |
|---|---|---|---|---|
| PS12.0 | Programmable Limit Switch | {...} | | |
| HighPreset | Long | 0 | | |
| LowPreset | Long | 0 | | |
| OutputHighData | Integer | 0000 0000 0000 0000 | Bits | |
| .0 | Bool | 0 | OFF | |
| .1 | Bool | 0 | OFF | |
| .2 | Bool | 0 | OFF | |
| .3 | Bool | 0 | OFF | |
| .4 | Bool | 0 | OFF | |
| .5 | Bool | 0 | OFF | |
| .6 | Bool | 0 | OFF | |
| .7 | Bool | 0 | OFF | |
| .8 | Bool | 0 | OFF | |
| .9 | Bool | 0 | OFF | |
| .10 | Bool | 0 | OFF | |
| .11 | Bool | 0 | OFF | |
| .12 | Bool | 0 | OFF | |
| .13 | Bool | 0 | OFF | |
| .14 | Bool | 0 | OFF | |
| .15 | Bool | 0 | OFF | |
| OuputLowData | Integer | 0000 0000 0000 0000 | Bits | |

## Using Programmable Limit Switch

The X8 Series PLC is in the run mode, and when the PLS function is enabled, the HSC will count incoming pulses. When the HSC accumulator data reaches the High Preset or Low preset defined in the PS data table, the filtered data will be written to defined data table through the HSC mask.

At that point, the next preset defined in the PS file becomes active.When the HSC counts to that new preset, the new output data is written through the HSC mask. This process continues until the last element within the PS file is loaded. At that point the active element within the PLS file is reset to zero.

This behavior is referred to as circular operation.

If invalid data is loaded during operation, an HSC error is generated (within the HSC function file). If an invalid parameter is detected, it will be skipped and the next parameter will be loaded for execution (provided it is valid).

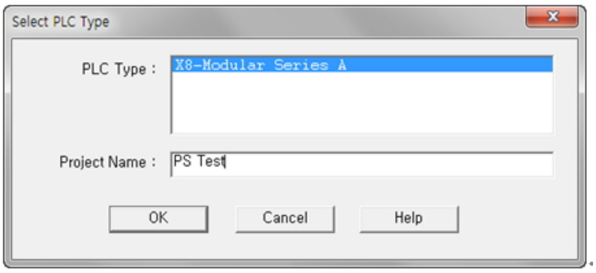
You can use the PS in both directions. If your application only counts in one direction, simply ignore the other parameters.

## Addressing PS Data Table

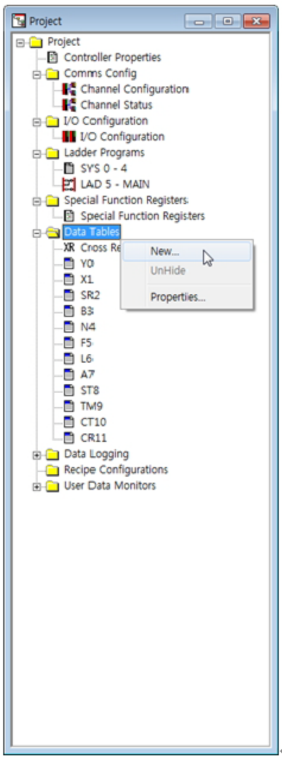
| Format | | Explanation |
|---|---|---|
| PSn.e<br>PSn.e.se<br>PSn.e.se.b<br>PSn.e.sym<br>PSn.e.sym.b | PS | Programmable Limit Switch Data Table |
| | n | Data Table Number |
| | . | Element delimiter |
| | e | Element number. Element is 6-word size. |
| | . | Sub-Element delimiter |
| | se | Sub-Element |
| | . | Symbol delimiter |
| | sym | Symbol. Not a number. Denotes the number |
| | . | Bit delimiter |
| | b | Bit Number |

## PS Data Table Example

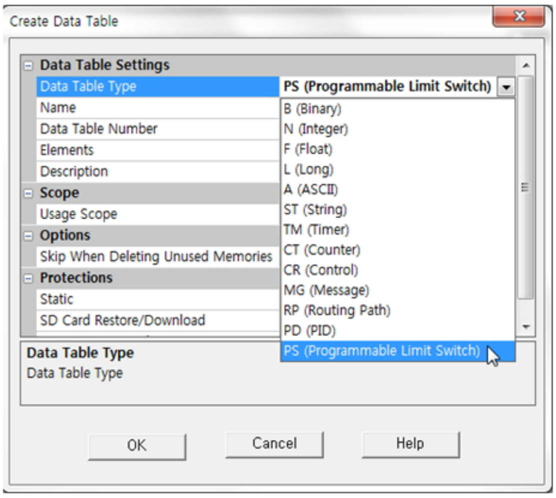
1. Using **XGPC**, create a new project.



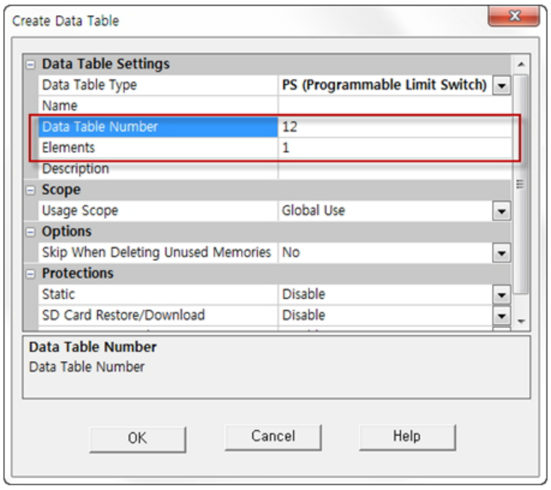

2. Right click on Data Table of XGPC and select “New” menu.



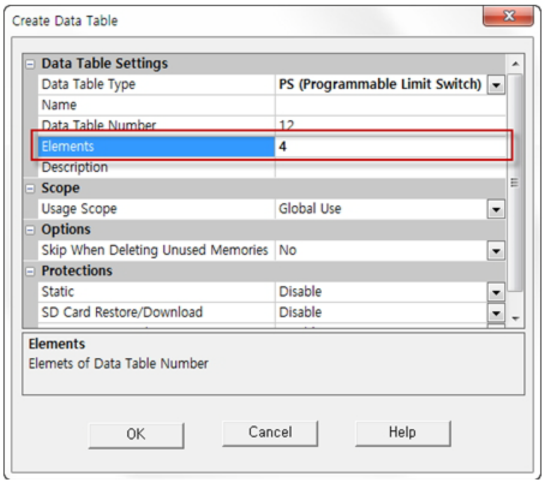

3. Select PS (Programmable Limit Switch) in the Data Table Type menu.



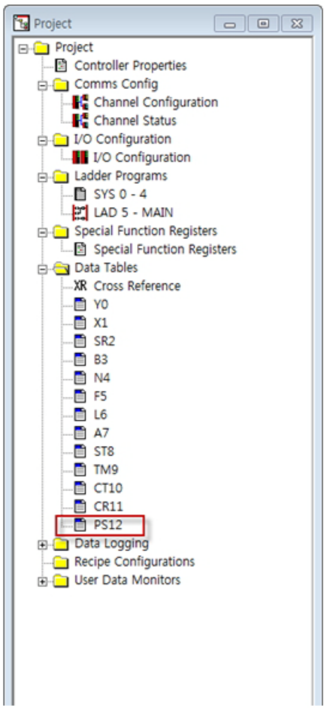

4. Enter the number of the Data Table Number (Max. 1535) and Elements (Max. 1535). Description is optional.



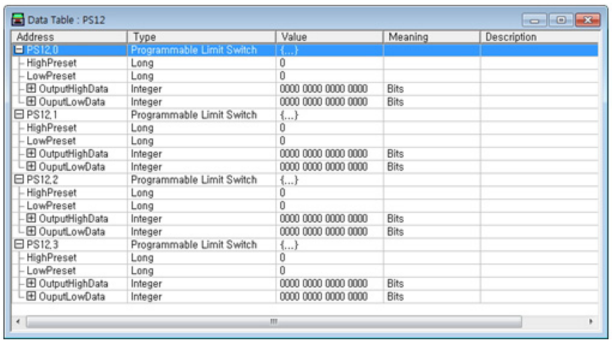

5. Then, enter the number of Elements. The number of Elements means the number of steps of the Programmable Limit Switch. In this example, No. 4 is entered. Click OK button after complete.



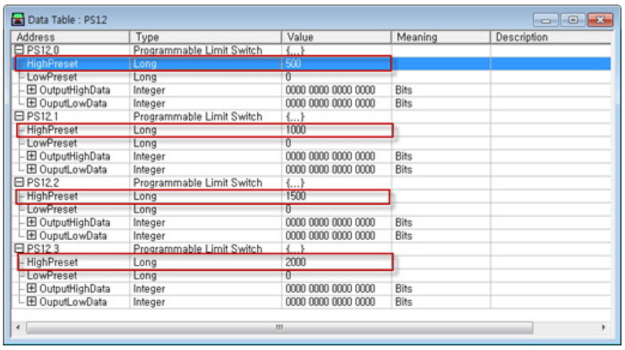

6. You can see the PS data table created before is assigned to 12 and added on the project window as below.



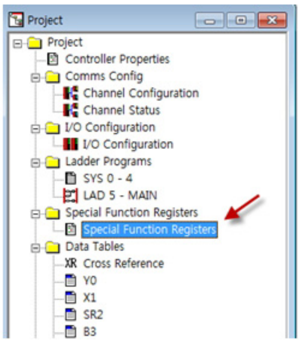

7. Double click the new PS data table and see the contents. The Elements is set to 4 on the above step, 4 elements from PS12.0 to PS12.3 are created in the screen below.



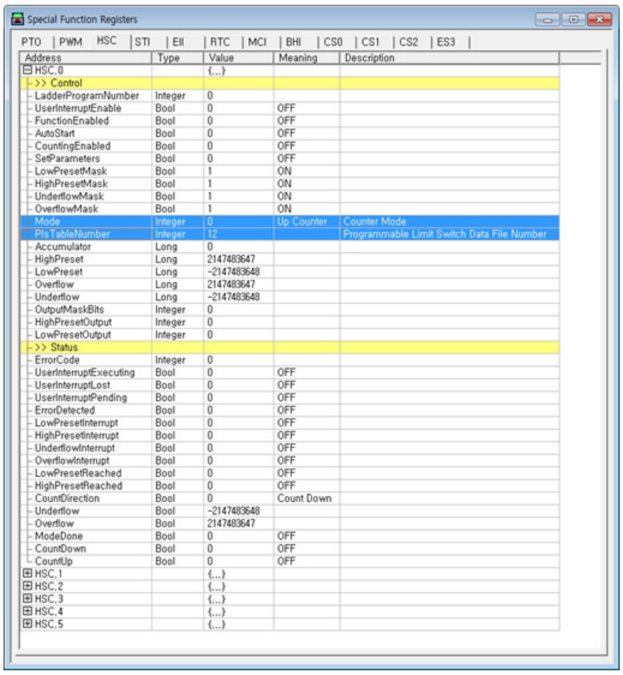
8. Enter 500,100,1500 and 2000 as HighPreset value.



9. Specifies the SFR (Special Function Register) for inter working with HSC and double click the Special Function Register in the Project window .




10. Specifies the HSC in the “HSC” of the Special Function Table. Set the HSC Mode =0 and PlsTableNumber=12 ( PS table number that is set before.




11. When the HSC accumulator value reaches to 500 specified to the HighPrest of the first PS, If you use OutputHighData value and HighPresetMask, it outputs the result value that masked value and logical ANDed value.

This behavior is repeated every 500, 1000, 1500, and 20004 intervals.

# Using High-Speed Outputs

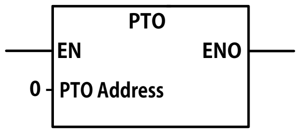
## Introduction

This chapter explains about using high-speed outputs of the **X8 Series PLC**. This chapter organized as follows:

## PTO - Pulse Train Output

High-speed output (PTO and PWM) functions of X8 Series PLC support up to 100KHz, and it would be applied to simple motion control and High-speed pulse output.

The PTO function is only available for Embedded I/O in the base module (CPU) and cannot be used as an External expansion I/O.




Instruction Type: output

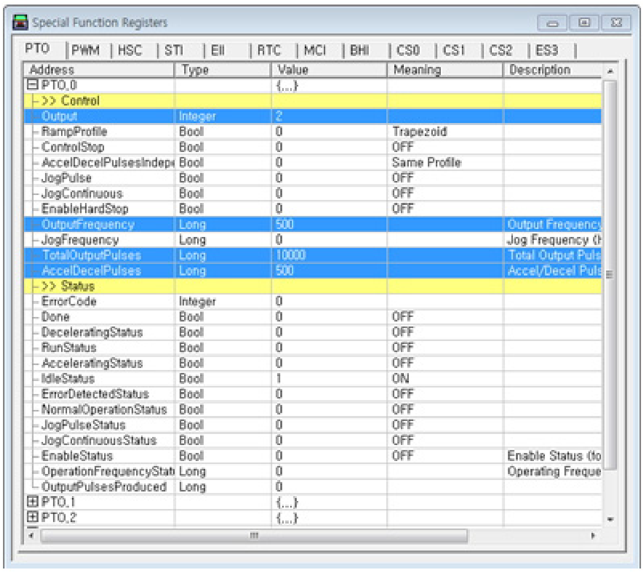
## Pulse Train Output Function

The **X8 Series PLC** supports four high-speed output ports. These output ports can be used as standard output ports or individually configured for PTO or PWM operation.

The PTO functionality allows a simple motion control or pulse output.
The pulse profile has four primary components:

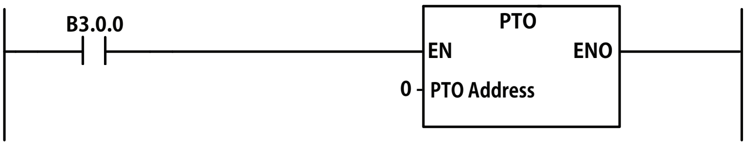
- Output port number
- Total number of pulses to be generated
- Accelerate/decelerate intervals
- Run interval

Users can input only three items above to set PTO set in SFR and create example for simple pulse output by using the PTO instruction in the ladder program.




| Address | Type | Value | Meaning | Description |
|---|---|---|---|---|
| PTO.0 | | {...} | | |
| >> Control | | | | |
| Output | Integer | 2 | | |
| RampProfile | Bool | 0 | Trapezoid | |
| ControlStop | Bool | 0 | OFF | |
| AccelDecelPulsesIndep | Bool | 0 | Same Profile | |
| JogPulse | Bool | 0 | OFF | |
| JogContinuous | Bool | 0 | OFF | |
| EnableHardStop | Bool | 0 | OFF | |
| OutputFrequency | Long | 500 | | Output Frequency |
| JogFrequency | Long | 0 | | Jog Frequency (H |
| TotalOutputPulses | Long | 10000 | | Total Output Puls |
| AccelDecelPulses | Long | 500 | | Accel/Decel Puls |
| >> Status | | | | |
| ErrorCode | Integer | 0 | | |
| Done | Bool | 0 | OFF | |
| DeceleratingStatus | Bool | 0 | OFF | |
| RunStatus | Bool | 0 | OFF | |
| AcceleratingStatus | Bool | 0 | OFF | |
| IdleStatus | Bool | 1 | ON | |
| ErrorDetectedStatus | Bool | 0 | OFF | |
| NormalOperationStatus | Bool | 0 | OFF | |
| JogPulseStatus | Bool | 0 | OFF | |
| JogContinuousStatus | Bool | 0 | OFF | |
| EnableStatus | Bool | 0 | OFF | Enable Status (to |
| OperationFrequencyStat | Long | 0 | | Operating Freque |
| OutputPulsesProduced | Long | 0 | | |
| PTO.1 | | {...} | | |
| PTO.2 | | {...} | | |

The figure above is four parameters of the SFR .

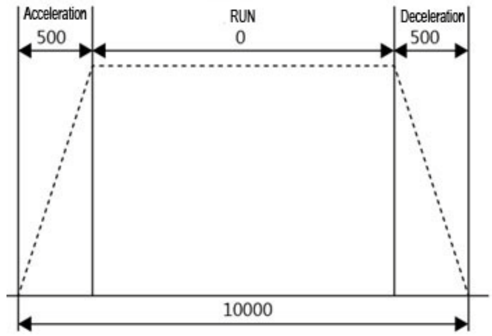



The figure above shows simple PTO operating program based on the SFR configuration.

When B3.0.0 is set, the PTO operates based on the SFR value. This is an example that when the acceleration and deceleration pulse is set to 500, total 10,000 of pulse is produced at a rate of 500Hz.

- When B3.0.0 bit is set, the AcceleratingStatsu flag shows the PTO status is set, and current frequency is displayed in the OperationFrequencyStatus. Currently generated pulse numbers are displayed in the OutpulPulsesProduced.
- During the pulse is output based on the pulse number defined in the AccelDecelPulses, the OperationFrequencyStatus frequency reaches to the OutputFrequency. When the RunStatus is set after acceleration phase completes, pulse is produced as 100Hz frequency.
- When the total number of pulse becomes 9500, the RunStatus is cleared. And go to deceleration phase when the DecelerratingStatus is set.
- When the 500 pulses are out, the OperationFrequencyStatusrk gradually decreases from 100 to zero.
- After pulse output, Done is set and PTO operation is completed.

- When B3.0.0 is cleared, the IdleStatus is set and PTO is in a Idle state.




The graph above shows the graph according to the set value of the above behavior.

Each Status Bit is updated for every scan during the execution of the PTO.

| | |
|---|---|
| **IMPORTANT** | When the invalid AccelDecelPulses and OutputFrequency data is input, PTO error will be generated.<br><br>The maximum value of the AccelDecelPulses is determined by the following formulas.<br><br>maxAccelValue = (uint32)((outputFrequency * (uint32)(outputFrequency >> 2)) + ((outputFrequency * (uint32)(outputFrequency % 4)) >> 2)); |

Followings are conditions for the PTO execution.

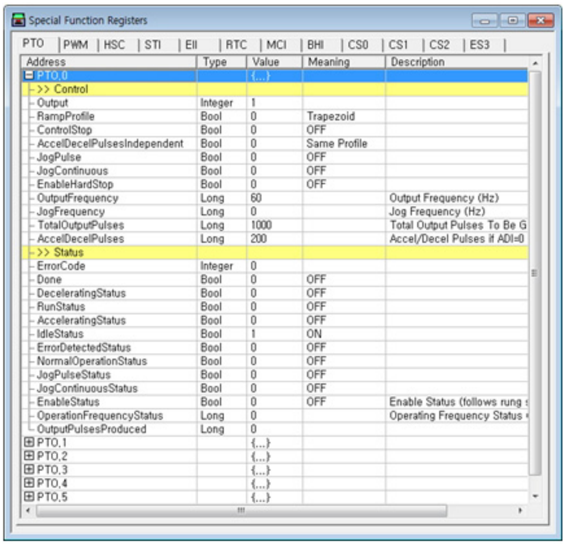
The above example can be executed as satisfy the following basic conditions.

- The PTO instruction must be in an idle state.
- The EnableStatus must be toggled from OFF to ON.
- For idle state behavior, all of the following conditions must be met:
  - Output cannot be change in Online.
  - The EnableHardStop bit must be off.
  - The JogPulse bit must be off.
  - The NormalOperationStatus bit must be off.\

The charts in the following examples illustrate the typical timing sequence/ behavior of a PTO instruction. The stages listed in each chart have nothing to do with controller scan time. They simply illustrate a sequence of events. In actuality, the controller may have hundreds or thousands of scans within each of the stages illustrated in the examples.

## PTO Function

Following is the SFR screen of the PTO instruction as mentioned above.


Special Function Registers

PTO | PWM | HSC | STI | EII | RTC | MCI | BHI | CS0 | CS1 | CS2 | ES3

| Address | Type | Value | Meaning | Description |
|---|---|---|---|---|
| PTO.0 | | {...} | | |
| >> Control | | | | |
| Output | Integer | 1 | | |
| RampProfile | Bool | 0 | Trapezoid | |
| ControlStop | Bool | 0 | OFF | |
| AccelDecelPulsesIndependent | Bool | 0 | Same Profile | |
| JogPulse | Bool | 0 | OFF | |
| JogContinuous | Bool | 0 | OFF | |
| EnableHardStop | Bool | 0 | OFF | |
| OutputFrequency | Long | 60 | | Output Frequency (Hz) |
| JogFrequency | Long | 0 | | Jog Frequency (Hz) |
| TotalOutputPulses | Long | 1000 | | Total Output Pulses To Be G |
| AccelDecelPulses | Long | 200 | | Accel/Decel Pulses if ADI=0 |
| >> Status | | | | |
| ErrorCode | Integer | 0 | | |
| Done | Bool | 0 | OFF | |
| DeceleratingStatus | Bool | 0 | OFF | |
| RunStatus | Bool | 0 | OFF | |
| AcceleratingStatus | Bool | 0 | OFF | |
| IdleStatus | Bool | 1 | ON | |
| ErrorDetectedStatus | Bool | 0 | OFF | |
| NormalOperationStatus | Bool | 0 | OFF | |
| JogPulseStatus | Bool | 0 | OFF | |
| JogContinuousStatus | Bool | 0 | OFF | |
| EnableStatus | Bool | 0 | OFF | Enable Status (follows rung |
| OperationFrequencyStatus | Long | 0 | | Operating Frequency Status |
| OutputPulsesProduced | Long | 0 | | |
| PTO.1 | | {...} | | |
| PTO.2 | | {...} | | |
| PTO.3 | | {...} | | |
| PTO.4 | | {...} | | |
| PTO.5 | | {...} | | |

Elements for PTO control is described in this chapter.

## SFR (Special Function Register) PTO Sub-Elements

The variables within each PTO sub-element are listed individually below. All examples illustrate PTO:0.

### SFR PTO Sub-Element (PTO.0)

| | Address and sub-element | Data Format | Range | User Program Access |
|---|---|---|---|---|
| **Contorl** | PTO.0.Output | Word (INT) | 2 ~ 4 | Read Only |
| | PTO.0.RampProfile | Bit | 0 or 1 | Read/Write |
| | PTO.0.ControlStop | Bit | 0 or 1 | Read/Write |
| | PTO.0.AccelDecelPulsesIndependent | Bit | 0 or 1 | Read/Write |
| | PTO.0.JogPulse | Bit | 0 or 1 | Read/Write |
| | PTO.0.JogContinuous | Bit | 0 or 1 | Read/Write |
| | PTO.0.EnableHardStop | Bit | 0 or 1 | Read/Write |
| | PTO.0.OutputFrequency | long word | 0 ~ 100,000 | Read/Write |
| | PTO.0.JogFrequency | long word | 0 ~ 100,000 | Read/Write |
| | PTO.0.TotalOutputPulses | long word | 0 ~ 2,147,483,647 | Read/Write |
| | PTO.0.AccelDecelPulses | long word | | Read/Write |

**SFR PTO Sub-Element (PTO.0)**

| | Address and sub-element | Data Format | Range | User Program Access |
|---|---|---|---|---|
| **Status** | PTO.0.ErrorCode | Word (INT) | -2 ~ 7 | Read Only |
| | PTO.0.Done | Bit | 0 or 1 | Read Only |
| | PTO.0.DeceleratingStatus | Bit | 0 or 1 | Read Only |
| | PTO.0.RunStatus | Bit | 0 or 1 | Read Only |
| | PTO.0.AcceleratingStatus | Bit | 0 or 1 | Read Only |
| | PTO.0.IdleStatus | Bit | 0 or 1 | Read Only |
| | PTO.0.ErrorDetectedStatus | Bit | 0 or 1 | Read Only |
| | PTO.0.NormalOperationStatus | Bit | 0 or 1 | Read Only |
| | PTO.0.JogPulseStatus | Bit | 0 or 1 | Read Only |
| | PTO.0.JogContinuousStatus | Bit | 0 or 1 | Read Only |
| | PTO.0.EnableStatus | Bit | 0 or 1 | Read Only |
| | PTO.0.OperationFrequencyStatus | long word | 0 ~ 100,000 | Read Only |
| | PTO.0.OutputPulsesProduced | long word | 0 ~ 2,147,483,647 | Read Only |

*Long Word = 32 bit integer

## PTO.0.Output

| Address | Data Format | HSC Mode | Function | User Program Access |
|---|---|---|---|---|
| PTO.0.Output | Word (INT) | 2 ~ 4 | Control | Read Only |

The PTO.0.Output variable defines the output port that the PTO instruction controls.

The PTO output port is assigned as below.

| | Output Value | Internal Output Port |
|---|---|---|
| 1 | PTO.0.Output=2 | Y0.0.2 |
| 2 | PTO.0.Output=3 | Y0.0.3 |
| 3 | PTO.0.Output=4 | Y0.0.4 |

This setting can be set only in offline, and cannot be changed in online status.

Forcing an output controlled by the PTO while it is running stops all output pulses and causes a PTO error.

## PTO.0.Done

| Address | Data Format | HSC Mode | Function | User Program Access |
|---|---|---|---|---|
| PTO.0.Done | Bit | 0 or 1 | Status | Read Only |

The PTO.0.Done flag bit is set when the PTO pulse output has completed its operation. When the EN signal of PTO instruction is false, or if the PTO instruction is run, the PTO.0.Done bit remains set until the PTO.0.Done bit is cleared.

## PTO.0.DeceleratingStatus

| Address | Data Format | HSC Mode | Function | User Program Access |
|---|---|---|---|---|
| PTO.0.DeceleratingStatus | Bit | 0 or 1 | Status | Read Only |

This flag bit is set when the PTO run mode is within the deceleation phase. This flag is cleard when it is within the acceleration phase or the run phase.

## PTO.0.RunStatus

| Address | Data Format | HSC Mode | Function | User Program Access |
|---|---|---|---|---|
| PTO.0.RunStatus | Bit | 0 or 1 | Status | Read Only |

The PTO.0.RunStatus bit is set when the PTO instruction is within the run phase. It is cleard when the PTO instruction is within the acceleration or deceleration phase.

## PTO.0.AcceleratingStatus

| Address | Data Format | HSC Mode | Function | User Program Access |
|---|---|---|---|---|
| PTO.0.AcceleratingStatus | Bit | 0 or 1 | Status | Read Only |

The PTO.0.AcceleratingStatus bit is set when the PTO instruction is within the acceleration phase. It is cleared when the PTO instruction is within the run phase.

## PTO.0.RampProfile

| Address | Data Format | HSC Mode | Function | User Program Access |
|---|---|---|---|---|
| PTO.0.RampProfile | Bit | 0 or 1 | Control | Read / Write |

The PTO.0.RampProfile flag bit is determined how generate the frequency stored in the PTO.0.OutputFrequency when the PTO.0.RampProfile flag is within the accelerate or decelerate phase.

This flag is set when the PTO instruction produces an S-Curve profile and is cleared when the PTO instruction to produce a Trapezoid profile.

## PTO.0.IdleStatus

| Address | Data Format | HSC Mode | Function | User Program Access |
|---|---|---|---|---|
| PTO.0.IdleStatus | Bit | 0 or 1 | Status | Read / Write |

The PTO.0.IdleStatus is set when the PTO is in an Idle state. The Idle state is defined as the PTO is not running and no errors are present.

The PTO instruction must be in an Idle state before it is running, the PTO.0.IdleStatus is used to check the PTO status before running.

## PTO.0.ErrorDetectedStatus

| Address | Data Format | HSC Mode | Function | User Program Access |
|---|---|---|---|---|
| PTO.0.ErrorDetectedStatus | Bit | 0 or 1 | Status | Read Only |

The PTO.0.ErrorDetectedStatus is set when the PTO is in an error state. If an error state is detected, the error is identified in the PTO.0.ErrorCode register.

This flag is used to check that the PTO is in an error state by the Control Program.

## PTO.0.NormalOperationStatus

| Address | Data Format | HSC Mode | Function | User Program Access |
|---|---|---|---|---|
| PTO.0.NormalOperationStatus | Bit | 0 or 1 | Status | Read / Write |

The PTO.0.NormalOperationStatus flag is set when the PTO is in the nomal state. A normal state is ACCEL, RUN, DECEL or DONE, with no PTO errors.

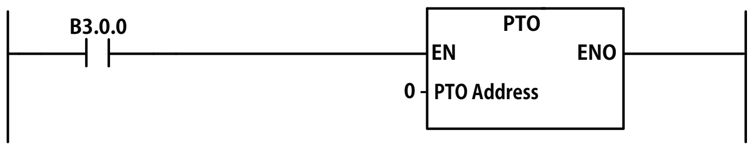
## PTO.0.EnableHardStop

| Address | Data Format | HSC Mode | Function | User Program Access |
|---|---|---|---|---|
| PTO.0.EnableHardStop | Bit | 0 or 1 | Control | Read / Write |

The PTO.0.EnableHardStop flag is used to stop the PTO immediately. Once the PTO starts operation, the only way to stop the PTO operation (Idel, RUN, Normal, Jog, Continuous, and Jog Pulse, etc) and generates a PTO sub-system error when the PTO is stop by using this flag.

## PTO.0.EnableStatus

| Address | Data Format | HSC Mode | Function | User Program Access |
|---|---|---|---|---|
| PTO.0.EnableStatus | Bit | 0 or 1 | Status | Read Only |




The PTO.0.EnableStatus flag is basic condition that can be operate the PTO instruction in the rung program. As the example above, the PTO instruction is enabed when the rung preceding the EN, the PTO input terminial is false.

## PTO.0.OutputFrequency

| Address | Data Format | HSC Mode | Function | User Program Access |
|---|---|---|---|---|
| PTO.0.OutputFrequency | Long Word | 0 ~ 100,000 | Control | Read /Write |

The PTO.0.OutputFrequency variable defines the frequency of the PTO output during the run phase. The maximum range is from 0 to 100,000Hz (100KHz). If the frequency is less than zero or greater than 100,000, generates a PTO error.

## PTO.0.OperatingFrequencyStatus

| Address | Data Format | HSC Mode | Function | User Program Access |
|---|---|---|---|---|
| PTO.0.OperatingFrequency Status | Long Word | 0 ~ 100,000 | Status | Read Only |

The PTO.0.OperatingFrequencyStatus has the pulse frequency data generated by the PTO. The maximum range is from 0 to 100,000Hz (100KHz). This flag is used to check the current PTO operation frequency.

## PTO.0.TotalOutputPulses

| Address | Data Format | HSC Mode | Function | User Program Access |
|---|---|---|---|---|
| PTO.0.TotalOutputPulses | Long Word | 0 ~ 2,147,483,647 | Control | Read / Write |

The PTO.0.TotalOutputPulses flag defines the total number of pulses to be generated for the pulse profile (accel/run/decel inclusive).

## PTO.0.OutputPulsesProduced

| Address | Data Format | HSC Mode | Function | User Program Access |
|---|---|---|---|---|
| PTO.0.OutputPulsesProduced | Long Word | 0 ~ 2,147,483,647 | Status | Read Only |

The PTO.0.OutputPulsesProduced monitors how many pulses have been generated by the PTO.

## PTO.0.AccelDecelPulsesIndependent

| Address | Data Format | HSC Mode | Function | User Program Access |
|---|---|---|---|---|
| PTO.0.AccelDecelPulsesIndependent | Bit | 0 or 1 | Control | Read / Write |

The PTO.0.AccelDecelPulsesIndependent bit is used to define whether the acceleration and deceleration intervals will be the same, or if each will have a unique value.

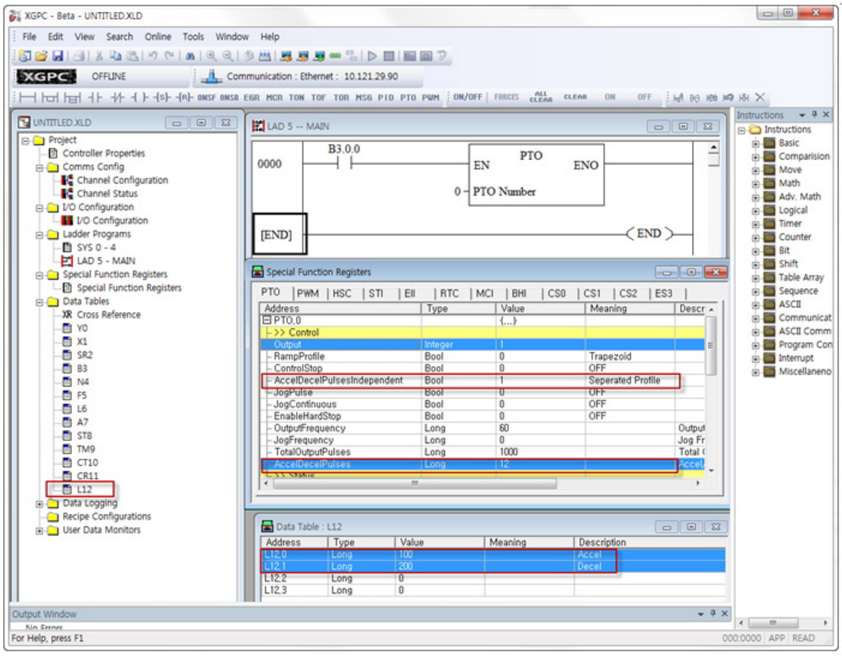
When this bit is set, separate profiles are used and the LongInteger typed data table number should be entered instead that specify the number of pulse to the PTO.0.AccelDecelPulses. However, if the table number and element number are entered in the XGPC, it is converted and entered automatically. (For example: L12.0, L12.5, etc.)

The element of the Long Integer data table to be used acceleration and deceleration profiles must be set to 4.

In the figure below, L12 data table is created and element is set to 4.

There must be four long elements available in this data table:

- Element 1 (L12.0): Acceleration Pulse Number
- Element 2 (L12.1) : Deceleration Pulse Number
- Elements 3 (L12.2) and 4 (L12.3) : System Reserved




This cannot be changed like other SFR once the program is downloaded into the controller.

If this flag is set, PTO error -3 is generated when the data combinations of data file number and element number is not entered or invalid data is entered.

## PTO.0.AccelDecelPulses

| Address | Data Format | HSC Mode | Function | User Program Access |
|---|---|---|---|---|
| PTO.0.AccelDecelPulses | Long Word | When the AccelDecelPulsesIndependent=0, 0 ~1,073,741,824<br><br>When the AccelDecelPulsesIndependent=1 0 to 2,147,483,647 | Control | Read / Write |

The PTO.0.AccelDecelPulses defines how many of the total pulses will be generate to each of the acceleration and deceleration phase.

When the PTO.0.AccelDecelPulsesIndependent is cleared, the same number of the entered acceleration and deceleration pulse is used.

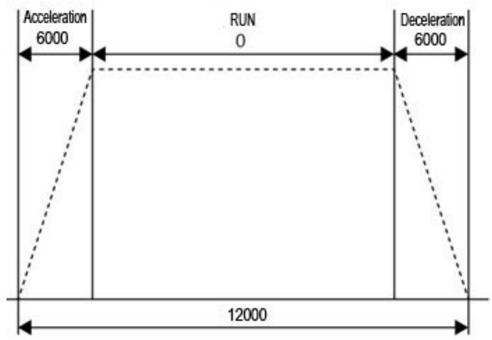
When the PTO.0.AccelDecelPulsesIndependent is set, data table number instead of pulse number is entered.

For more descriptions, refer to PTO.0.AccelDecelPulsesIndependent.

Followings are error conditions according to the PTO.0.AccelDecelPulses data

- PTO.0.AccelDecelPulses is less than 0
- The total pulses for the PTO.0.AccelDecelPulses acceleration and deceleration pulse is greater than the total output pulses (PTO.0.TotalOutputPulses) (PTO.0.AccelDecelPulsesIndependent is 0 or 1)

In the example below shows the PTO.0.AccelDecelPulsesIndependent is 0. In this example, the maximum value that could be used for accelerate/decelerate is 6000, because if both accelerate and decelerate are 6000, the total number of pulses = 12,000. The run component would be zero. This profile would consist of an acceleration phase from 0~6000. At 6000, the output frequency is generated and immediately enters the deceleration phase, 6000~12,000. At 12,000, the PTO operation would stop (output frequency = 0).




If you need to determine the ramp period (accelerate/decelerate ramp duration):

$$\textbf{ADP} = \frac{2 \text{ X PTO.0.AccelDecelPulses}}{\text{PTO.0.OutputFrequency}}$$

* ADP (Accelerate/Decelerate Phase)

The following formulas can be used to calculate the maximum value that could be used for accelerate/decelerate for both profiles.

For Trapezoid

$$= [PTO.0.OutputFrequency\ X\ (\frac{PTO.0.OutputFrequency}{4})\ ] +0.5$$

For S-Curve Profiles

$$= 0.999\ X\ PTO.0.OutputFrequency\ X\ \sqrt{\frac{PTO.0.OutputFrequency}{6}}$$

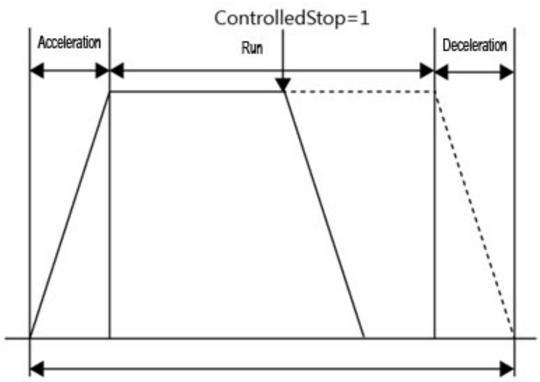
## PTO.0.ControlledStop

| Address | Data Format | HSC Mode | Function | User Program Access |
|---|---|---|---|---|
| PTO.0.ControlledStop | Bit | 0 or 1 | Control | Read / Write |

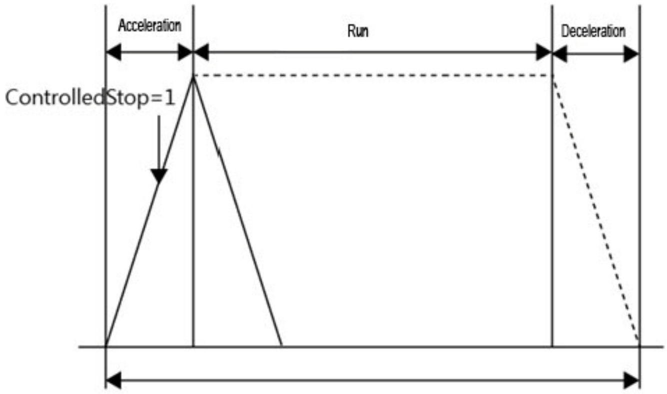
The PTO.0.ControlledStop bit is used to stop an executing PTO instruction.

Following two graphs show that the PTO is stop forcedly by PTO.0.ControlledStop=1 within the run phase, and within the acceleration phase.

- If the PTO.0.ControlledStop bit is set during the run phase, the run phase completes and the PTO immediately enters the deceleration phase.




- If the PTO.0.ControlledStop bit is set during the acceleration phase, the acceleration phase completes and the PTO immediately enters the deceleration phase.




## PTO.0.JogFrequency

| Address | Data Format | HSC Mode | Function | User Program Access |
|---|---|---|---|---|
| PTO.0.JogFrequency | Long Word | 0 ~ 100,000 | Control | Read / Write |

The PTO.0.JogFrequency variable defines the frequency of the PTO output during all Jog phases. The range of frequency is 0 ~ 100,000 (Hz). This value is typically determined by the type of device that is being driven, the mechanics of the application, or the device/components being moved. The data less than zero or greater than 100,000 generates a PTO error.

## PTO.0.JogPulse

| Address | Data Format | HSC Mode | Function | User Program Access |
|---|---|---|---|---|
| PTO.0.JogPulse | Bit | 0 or 1 | Control | Read / Write |

The PTO.0.JogPulse bit is used to instruct the PTO to generate a single pulse. The width is defined by the PTO.0.JogFrequency in the PTO.

The PTO.0.JogFrequency flag is only possible under the following conditions:

- PTO is in idle
- PTO.0.JogContinuous is not active
- PTO.0.EnableStatu is not active

## PTO.0.JogPulseStatus

| Address | Data Format | HSC Mode | Function | User Program Access |
|---|---|---|---|---|
| PTO.0.JogPulseStatus | Bit | 0 or 1 | Status | Read Only |

The PTO.0.JogPulseStatus bit is set when the PTO instruction detects the PTO has generated a Jog Pulse.

## PTO.0.JogContinuous

| Address | Data Format | HSC Mode | Function | User Program Access |
|---|---|---|---|---|
| PTO.0.JogContinuous | Bit | 0 or 1 | Control | Read / Write |

The PTO.0.JogContinuous bit is set when the PTO generates continuous pulses. The frequency generated is defined by the PTO.0.JogFrequency parameter.

When the PTO.0.JogContinuous bit is cleared, the current output pulse is truncated.

Jog Continuous operation is only possible under the following conditions:

- PTO is in idle
- PTO.0.JogContinuous is 0
- PTO.0.EnableStatus is 0

## PTO.0.JogContinuousStatus

| Address | Data Format | HSC Mode | Function | User Program Access |
|---|---|---|---|---|
| PTO.0.JogContinuousStatus | Bit | 0 or 1 | Status | Read Only |

The PTO.0.JogContinuousStatus flag shows current state when the PTO.0.JogContinuous bit generates continuous Jog Pulses.

### PTO.0.ErrorCode

| Address | Data Format | HSC Mode | Function | User Program Access |
|---|---|---|---|---|
| PTO.0.Error Code | Word (INT) | -3 ~ 7 | Status | Read Only |

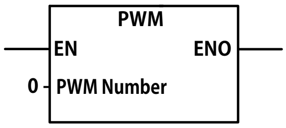
PTO PTO.0.ErrorCode shows the error code.

## PWM - Pulse Width Modulation

The X8 Series PLC High-Speed Output (PTO and PWM) function supports up to 100KHz, and is applied to simple motion control and high-speed pulse output.

The PWM function can only be used with the controller's embedded I/O. It cannot be used with expansion I/O modules.

The PWM instruction should only be used with X8 Series PLC. Relay outputs are not capable of performing very high-speed operations.




Instruction Type: output

## PWM Function

The PWM function allows a field device to be controlled by a PWM wave form. The PWM profile has two primary components:

- Frequency to be generated
- Duty Cycle interval

## PWM Data Table

The PWM instruction, along with the HSC and PTO functions, are different than all other controller instructions. Their operation is performed by custom circuitry that runs in parallel with the main system processor. This is necessary because of the high performance requirements of these instructions.

A typical operating sequence of a PWM instruction is as follows:

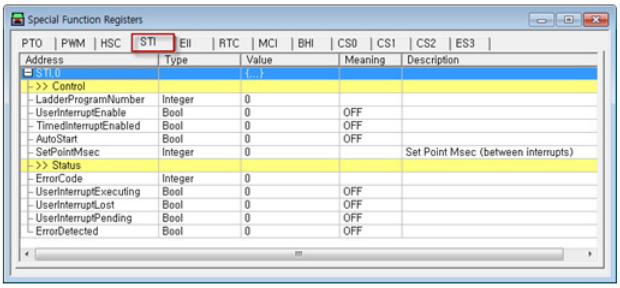
1. The rung that a PWM instruction is on is solved true (the PWM is started).

2. PWM function execution (Frequency is generated)

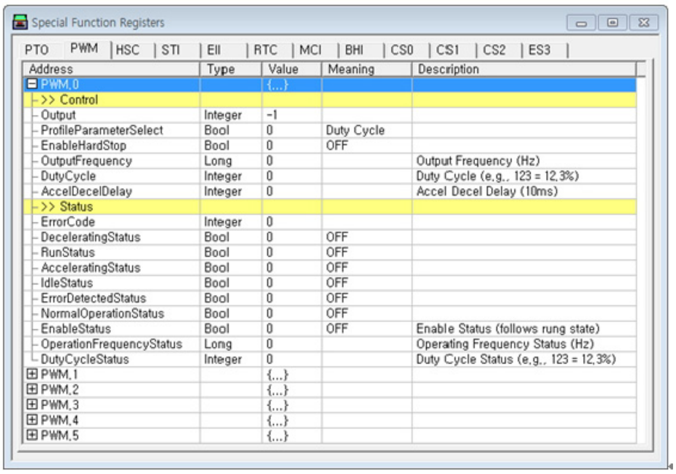
3. PWM function complete execution

4. The PWM instruction is IDLE.

Because the PTO, PWM and HSC instructions are actually being executed by a parallel system, the status bits and other information are updated each time the PWM instruction is scanned while it is running. If the scan time is longer, it can be problem of the PTO and PWM data update, it can be controlled by the STI (Selectable Timed Interrupt) that can be set in SFR.




The PWM output ports supported by the X8 Series PLC is allocated to have Y0.0.2, Y0.0.3, and Y0.0.4. The element is shown below.



## PWM Data Table Elements Summary

The variables within each PWM element, along with what type of behavior and access the control program has to those variables, are listed individually below.

| | Address and Element | Data Format | Range | User Program Access |
|---|---|---|---|---|
| **Contorl** | PTO.0.Output | Word (INT) | 2 ~ 4 | Read Only |
| | PWM.0.ProfileParameterSelect | Bit | 0 or 1 | Read Only |
| | PWM.0.EnableHardStop | Bit | 0 or 1 | Read / Write |
| | PWM.0.OutputFrequency | long word | 0 ~ 40,000 | Read / Write |
| | PWM.0.DutyCycle | Word | 0 ~ 1,000 | Read / Write |
| | PWM.0.AccelDecelDelay | Word | 0 ~ 32767 | Read / Write |
| **Status** | PTO.0.ErrorCode | Word (INT) | -2 ~ 7 | Read Only |
| | PTO.0.Done | Bit | 0 or 1 | Read Only |
| | PWM.0.ErrorCode | Word (INT) | -2 ~ 5 | Read Only |
| | PWM.0.DeceleratingStatus | Bit | 0 or 1 | Read Only |
| | PWM.0.RunStatus | Bit | 0 or 1 | Read Only |
| | PWM.0.AcceleratingStatus | Bit | 0 or 1 | Read Only |
| | PWM.0.IdleStatus | Bit | 0 or 1 | Read Only |
| | PWM.0.ErrorDetectedStatus | Bit | 0 or 1 | Read Only |
| | PWM.0.NormalOperationStatus | Bit | 0 or 1 | Read Only |
| | PWM.0.EnableStatus | Bit | 0 or 1 | Read Only |
| | PWM.0.OperationFrequencyStatus | long word | 0 ~ 40,000 | Read Only |
| | PWM.0.DutyCycleStatus | Word (INT) | 1 ~ 1000 | Read Only |

* Long Word = 32 bit integer

### PWM.0.Output

| Address | Data Format | HSC Mode | Function | User Program Access |
|---|---|---|---|---|
| PTO.0.Output | Word (INT) | 2 ~ 4 | Control | Read Only |

The PWM.0.Output variable specify the physical output port that the PWM instruction controls.

| | Output | Embedded Output Port |
|---|---|---|
| 1 | PTO.0.Output=2 | Y0.0.2 |
| 2 | PTO.0.Output=3 | Y0.0.3 |
| 3 | PTO.0.Output=4 | Y0.0.4 |

## PWM.0.ProfileParameterSelect

| Address | Data Format | HSC Mode | Function | User Program Access |
|---|---|---|---|---|
| PWM.0.ProfileParameterSelect | Bit | 0 or 1 | Status | Read Only |

The PWM.0.ProfileParameterSelect flag is used to set which waveform is output:

- Set (1) - Whenever the frequency is output.
- Cleared (0) - Whenever the Duty Cycle is set.

This bit cannot be modified while the PWM output is running.

## PWM.0.EnableHardStop

| Address | Data Format | HSC Mode | Function | User Program Access |
|---|---|---|---|---|
| PWM.0.EnableHardStop | Bit | 0 or 1 | Control | Read Only |

The PWM.0.EnableHardStop bit is used to stop the PWM instruction within the run phase immediately.

- Stop the run phase - Whenever the PWM.0.EnableHardStop is set.
- Normal run state - Whenever the PWM.0.EnableHardStop is cleared.

## PWM.0.OutputFrequency

| Address | Data Format | HSC Mode | Function | User Program Access |
|---|---|---|---|---|
| PWM.0.OutputFrequency | Long Word | 0 ~100,000 | Control | Read / Write |

The PWM.0.OutputFrequency bit is specify the frequency of the PWM. This frequency can be changed at any time unlike the PTO.

## PWM.0.DutyCycle

| Address | Data Format | HSC Mode | Function | User Program Access |
|---|---|---|---|---|
| PWM.0.DutyCycle | Word | 0 ~1000 | Control | Read / Write |

The PWM.0.DutyCycle is used to define the Duty Cycle.

- PWM.0.DutyCycle = 1000: 100% Output ON (constant, no waveform)
- PWM.0.DutyCycle = 750: 75% Output ON, 25% output OFF
- PWM.0.DutyCycle = 500: 50% Output ON, 50% output OFF
- PWM.0.DutyCycle = 250: 25% Output ON, 75% output OFF
- PWM.0.DutyCycle = 0: 0% Output OFF (constant, no waveform)

## PWM.0.AccelDecelDelay

| Address | Data Format | HSC Mode | Function | User Program Access |
|---|---|---|---|---|
| PWM.0.AccelDecelDelay | Word | 0 ~32767 | Control | Read / Write |

The PWM.0.AccelDecelDelay is set the delay time within 10mS range when the frequency is generated from 0Hz to 20KHz and the value can be modified at any time. (Actual application point of the modified value is when the Rung is executed during the ladder program scanning.)

## PWM.0.ErrorCode

| Address | Data Format | HSC Mode | Function | User Program Access |
|---|---|---|---|---|
| PWM.0.ErrorCode | Word (INT) | -2 ~ 5 | Status | Read Only |

The PWM.0.ErrorCode bit shows the error code when an error state is detected.

## PWM.0.DeceleratingStatus

| Address | Data Format | HSC Mode | Function | User Program Access |
|---|---|---|---|---|
| PWM.0.DeceleratingStatus | Bit | 0 or 1 | Status | Read Only |

The PWM.0.DeceleratingStatus bit is set when the PWM output is within deceleration phase.

- Set (1) - Whenever a PWM instruction is within the deceleration phase.
- Cleared (0) - Whenever a PWM instruction is not in the deceleration phase.

## PWM.0.RunStatus

| Address | Data Format | HSC Mode | Function | User Program Access |
|---|---|---|---|---|
| PWM.0.RunStatus | Bit | 0 or 1 | Status | Read Only |

The PWM.0.RunStatus bit is set when the PWM is within the run phase.

- Set (1) - Whenever the PWM is within the run phase.
- Cleared (0) - whenever the PWN is not in the run phase.

## PWM.0.IdleStatus

| Address | Data Format | HSC Mode | Function | User Program Access |
|---|---|---|---|---|
| PWM.0.IdleStatus | Bit | 0 or 1 | Status | Read Only |

The PWM.0.IdleStatus bit is set when the PWM is in the Idle state.

- Set (1) - Whenever the PWM is in the Idle state.
- Cleared (0) -Whenever the PWM is not in the Idle state.

## PWM.0.ErrorDetectedStatus

| Address | Data Format | HSC Mode | Function | User Program Access |
|---|---|---|---|---|
| PWM.0.ErrorDetectedStatus | Bit | 0 or 1 | Status | Read Only |

The PWM.0.ErrorDetectedStatus bit is set when the PWM error is detected.

- Set (1) - Whenever the PWM error is detected.
- Cleared (0) -Whenever the PWM error is not detected.

The detected error code is stored in the PWM.0.ErrorCode element.

## PWM.0.NormalOperationStatus

| Address | Data Format | HSC Mode | Function | User Program Access |
|---|---|---|---|---|
| PWM.0.NormalOperationStatus | Bit | 0 or 1 | Status | Read Only |

The PWM.0.NormalOperationStatus bit is set when the PWM is in the Acceleration, deceleration and run statue with no PWM errors.

- Set (1) - Whenever the PWM is in the Acceleration, deceleration and run statue with no PWM errors.
- Cleared (0) -Whenever the PWM is in its normal state.

## PWM.0.EnableStatus

| Address | Data Format | HSC Mode | Function | User Program Access |
|---|---|---|---|---|
| PWM.0.EnableStatus | Bit | 0 or 1 | Status | Read Only |

The PWM.0.EnableStatus bit is input signal that enables the PWM execution. If this bit set, rung with PWM instruction is executed.

## PWM.0.OperatingFrequencyStatus

| Address | Data Format | HSC Mode | Function | User Program Access |
|---|---|---|---|---|
| PWM.0.OperatingFrequency Status | Long Word | 0 ~100,000 | Status | Read Only |

The PWM.0.OperatingFrequencyStatus bit shows the frequency that the PWM is executed.

## PWM.0.DutyCycleStatus

| Address | Data Format | HSC Mode | Function | User Program Access |
|---|---|---|---|---|
| PWM.0.DutyCycleStatus | Word | 0 ~1000 | Status | Read Only |

The PWM.0.DutyCycleStatus bit shows the duty cycle feedback information that the PWM is executed.

# Relay-Type (Bit) Instructions

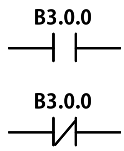
## Introduction

This chapter describes the **X8 Series PLC** relay-type (bit) instructions. This chapter organized as follows.

Use relay-type (bit) instructions to control bits such as ON/OFF input and output bits. The following instructions are described in this chapter:

| Instruction | Description |
|---|---|
| NOC | Normally Open Contact |
| NCC | Normally Closed Contact |
| OUT | Output Coil |
| SET | Set Coil |
| RST | Reset Coil |
| EGR | Edge Rising |
| ONSR | One Short Rising |
| ONSF | One Shot Falling |

## NOC- Normal Open Contact NCC - Normal Closed Contact



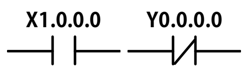

Instruction Type: input

This instruction is used to check TRUE or FALSE in various relay-type (bit) instrucions.

| Addressed Bit | NOC Instruction | NCC Instruction |
|---|---|---|
| OFF | FALSE | TRUE |
| ON | TRUE | FALSE |

Example:

- Check Input/Ouput Port Status

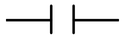



- Check Various flag status

TM9.0.Done

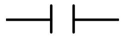
- Check Binary typed data table

B3.0.0



- Check Binary type data table value

B3.0.0

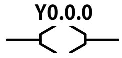


| Parameter | Data Table | | | | | | | | | | | | | SFR | | | | | | | | | | Address Mode | | | Address Level | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Y | X | SR | B | TM, CT, CR | N | F | ST | A | L | MG, PD | RP | PS | ASCII | RTC | HSC | PTO, PWM | STI | EII | BHI | MMI | CS-Comms | DLS-DataLog | Immediate | Direct | Indirect | Bit | Word | LongWord | Element |
| Bit | √ | √ | √ | √ | √ | √ | √ | √ | √ | √ | √ | | | | √ | √ | √ | √ | √ | √ | √ | √ | √ | | √ | √ | √ | | | |

You cannot use indirect addressing and SFR with:
SR, MG, PD, RTC, HSC, PTO, PWM, STI, EII, BHI, MMI, CS, DLS

# OUT - Output Coil

Y0.0.0



Instruction Type: output

This instruction outputs ON(TRUE) or OFF(FALSE) for relay-type (bit) input result.

- ON when the input rung condition become true.
- OFF when the input rung condition become false.

OUT instructions are reset (turned OFF) when:

- You enter to the program or remote program mode via XGPC

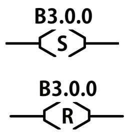
- Power Cycle (Off -> ON)
- The OUT is programmed within an inactive or false MCR (Master Control Reset) zone.

**TIP** A bit that is set within a subroutine using an OUT instruction remains set until the OUT is scanned again.

| Parameter | Data Table | | | | | | | | | | | | | Function Files | | | | | | | | | | Address Mode | | | Address Level | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Y | X | SR | B | TM, CT, CR | N | F | ST | A | L | MG, PD | RP | PS | ASCII | RTC | HSC | PTO, PWM | STI | EII | BHI | MMI | CS-Comms | DLS-DataLog | Immediate | Direct | Indirect | Bit | Word | LongWord | Element |
| Bit | √ | √ | √ | √ | √ | √ | | | √ | √ | √ | | | | √ | √ | √ | √ | √ | | | | √ | | √ | √ | √ | | | |

You cannot use indirect addressing and SFR with:
SR, MG, PD, RTC, HSC, PTO, PWM, STI, EII, BHI, MMI, CS, DLS

# SET - Set Coil
# RST - Reset Coil




Instruction Type: output

The SET and RST instructions are complementary instructions.

These instructions are usually used in pairs, with both instructions addressing the same bit. SET turns on a bit, while RST turns off a bit.

Since these are outputs, once set (or reset), they remain set (or reset) regardless of the rung condition.

**WARNING** In the event of the Power Cycle (off -> on) during operation, data stored in the data table is output again.

If these bits were cleared under error conditions, after restore, data table value is output again.

| Parameter | Data Table | | | | | | | | | | | | | Function Files | | | | | | | | | Address Mode | | | Address Level | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Y | X | SR | B | TM, CT, CR | N | F | ST | A | L | MG, PD | RP | PS | ASCII | RTC | HSC | PTO, PWM | STI | EII | BHI | MMI | CS-Comms | DLS-DataLog | Immediate | Direct | Indirect | Bit | Word | LongWord | Element |
| Bit | √ | √ | √ | √ | √ | √ | | | √ | √ | √ | | | | √ | √ | √ | √ | √ | | | | √ | | √ | √ | √ | | | |

You cannot use indirect addressing and SFR with:
SR, MG, PD, RTC, HSC, PTO, PWM, STI, EII, BHI, MMI, CS, DLS

# EGR - Edge Risng

```
  B3.0.0
-[ EGR ]-
```

Instruction Type: input

The EGR instruction is a retentive input instruction that triggers an event to occur one time. After the false-to-true rung transition (when the processor scans the rung), the EGR instruction remains true for one program scan and stores the TRUE value to the parameter.

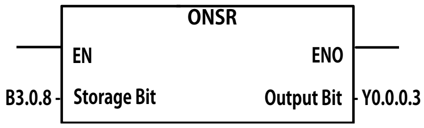
Following table shows change of the data table of storage bit according to change of the rung state.

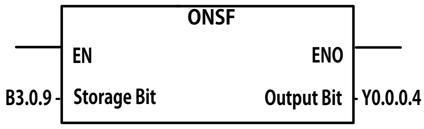
| Rung Transition | Storage Bit | Rung State after Execution |
|---|---|---|
| FALSE to TRUE (1 scan) | Storage bit is SET (1) | TRUE |
| TRUE to TRUE | Storage bit remains SET (1) | FALSE |
| TRUE to TRUE<br>FALSE to FALSE | Storage bit is CLEARD (0) | FALSE |

**TIP** The EGR instruction of the X8 Series PLC is functionally the same as the ONSR (ONe Shot Rising) instruction.

| Parameter | Data Table | | | | | | | | | | | | | SFR | | | | | | | | | | Address Mode | | | Address Level | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Y | X | SR | B | TM, CT, CR | N | F | ST | A | L | MG, PD | RP | PS | ASCII | RTC | HSC | PTO, PWM | STI | EII | BHI | MMI | CS-Comms | DLS-DataLog | Immediate | Direct | Indirect | Bit | Word | LongWord | Element |
| Bit | | | | √ | | √ | | | | | | | | | | | | | | | | | √ | | √ | | √ | | | |

# ONSR - One Shot Rising
# ONSF - One Shot Falling




Instruction Type: input

Use the ONSR and ONSF instructions to trigger an event to occur one time. These instructions trigger an event based on a change of rung state, as follows:

- Use the OSR instruction when an event must start based on the false-to-true (rising edge) change of state of the rung.
- Use the OSF instruction when an event must start based on the true-to-false (falling edge) change of state of the rung.

These instructions use two parameters.

- Storage Bit: this is the bit address that store the rung state from the previous scan.
- Output Bit: this is the bit address which is set based on a false-to-true (ONSR) or true-to-false (ONSF) rung transition.

To re-activate the ONSR or ONSF instructions, the rung must become TRUE or FALSE.

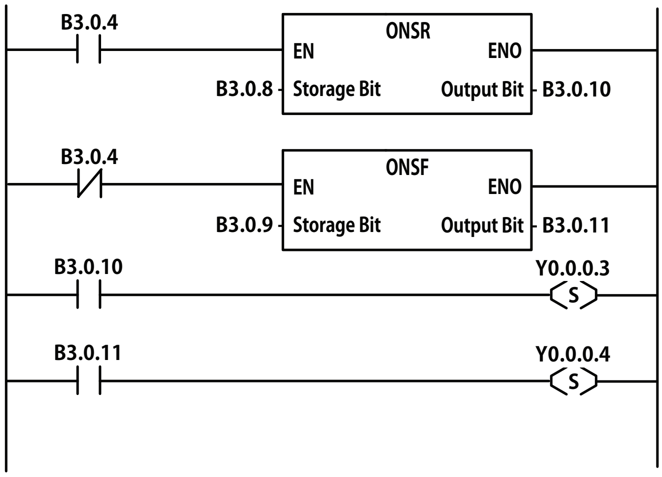
**ONSR Instruction Table**

| Rung Transition | Storage Bit | Rung State after Execution |
|---|---|---|
| FALSE to TRUE (1 Scan) | Bit | Bit |
| TRUE to TRUE | Bit | Bit |
| TRUE to TRUE<br>FALSE to FALSE | Bit | Bit |

**ONSF Instruction Table**

| Rung Transition | Storage Bit | Rung State after Execution |
|---|---|---|
| FALSE to TRUE (1 Scan) | Bit | Bit |
| TRUE to TRUE | Bit | Bit |
| TRUE to TRUE<br>FALSE to FALSE | Bit | Bit |

**TIP** Since the output bit data is remains for 1 scan, you can apply the SET instruction like below.




| Parameter | Data Table | | | | | | | | | | | | | SFR | | | | | | | | | Address Mode | | | Address Level | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Y | X | SR | B | TM, CT, CR | N | F | ST | A | L | MG, PD | RP | PS | RTC | HSC | PTO, PWM | STI | EII | BHI | MMI | CS-Comms | DLS-DataLog | Immediate | Direct | Indirect | Bit | Word | LongWord | Element |
| Storage Bit | | | | √ | | √ | | | | | | | | | | | | | | | | | | √ | | √ | | | |
| Output Bit | √ | √ | | √ | √ | √ | | | | √ | | | | | | | | | | | | | | √ | | √ | | | |

# Timer and Counter Instructions

## Introduction

In this chapter, the X8 Series PLC Timer and Counter Instructions are described. This chapter organized as follows

Timers and counters are output instructions that let you control operations based on time. The following Timer and Counter Instructions are described in this chapter:

| Instruction | Description |
|---|---|
| TON | Timer ON |
| TOF | Timer OFF |
| TONR | Retentive Timer |
| CTU | Count up |
| CTD | Count down |
| RSTA | Accumulator Initialization (Reset) |

## Timer Instructions Overview

To use timer, the TM instruction for timer data table must be used and is used to configure and monitoring.

Timer data table has three sub-elements for timer controlling. These sub-elements are:

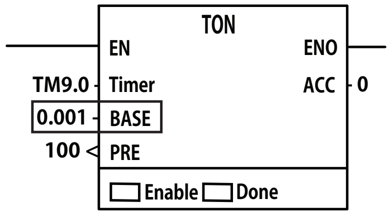
- Timer Control and Status
- Preset
  This is the value that the timer must reach before the timer times out. When the accumulator reaches this value, the DONE status bit is set.
- Accumulator
  This is the timer value that currently accumulated. It represents elapsed time. The accumulator data range is from 0 to 32767.

Timers can be set to any one of three time bases.

**Timer Base Settings**

| Time Base | Timing Range | Data Table Settings | |
|---|---|---|---|
| | | TimeBase1 | TimeBase0 |
| 0.001 seconds | 0 ~ 32.767 seconds | 0 | 1 |
| 0.01 seconds | 0 ~ 327.67 seconds | 0 | 0 |
| 1.00 seconds | 0 ~ 32,767 seconds | 1 | 0 |

The TimeBase 0 and TimeBase1 value in the table above is automatically set in the ladder program. If the BASE value in the figure below is set to 0.001, the TimeBase 0 and TimeBase1value in the TM data table related with the BASE value is set automatically.




Following table shows the timer data table elements.

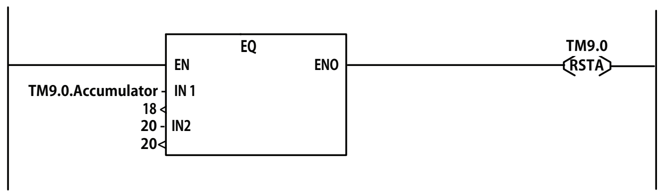
| Element | Data Type | Access | Description |
|---|---|---|---|
| TimeBase0 | Bit | Read / Write | Time setting based on timer |
| TimeBase1 | Bit | Read / Write | |
| Done | Bit | Read Only | Timer operation is completed. |
| TimerTiming | Bit | Read Only | Timer is timing. |
| Enable | Bit | Read Only | Timer is enable. |
| Preset | LongWord | Read / Write | Preset |
| Accumulator | LongWord | Read / Write | Accumulator |

**ATTENTION**

Do not copy timer elements while the timer enable bit (EN) is set. Timing could be inaccurate if the timer refers timer elements from outside since the timer uses an internal interrupt handling structurally.

**TIP**

Use an RSTA (Reset Accumulator) instruction to reset a timer's accumulator and status bits.



**TIP** Timer accuracy refers to the length of time between the moment a timer instruction is enabled and the moment the timed interval is complete.

**Timer Accuracy**

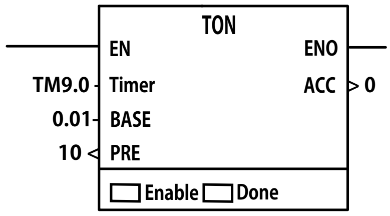
| Time Base | Accuracy |
|---|---|
| 0.001 seconds | -0.001 ~ 0.00 |
| 0.01 seconds | -0.01 ~ 0.00 |
| 1.00 seconds | -1.00 ~ 0.00 |

**TIP** Timing could be inaccurate if JUMP, LBL, CALL, or SBR instructions skip over the rung containing a timer instruction while the timer is timing.

If the skip duration is within 2.5 seconds, no time is lost; if the skip duration exceeds 2.5 seconds, an undetectable timing error occurs. When using an instruction can effect to a timer such as subroutines, a timer must be scanned at least every 2.5 seconds to prevent a timing error.

| Parameter | Data Table | | | | | | | | | | | | | SFR | | | | | | | | | Address Mode | | | Address Level | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Y | X | SR | B | TM, CT, CR | N | F | ST | A | L | MG, PD | RP | PS | RTC | HSC | PTO, PWM | STI | EII | BHI | MMI | CS-Comms | DLS-DataLog | Immediate | Direct | Indirect | Bit | Word | LongWord | Element |
| Timer | | | | | √ | | | | | | | | | | | | | | | | | | | √ | | | | | √ |
| TimeBase | | | | | | | | | | | | | | | | | | | | | | | √ | | | | | | √ |
| Preset | | | | | | | | | | | | | | | | | | | | | | | √ | | | | √ | | |
| Accumulator | | | | | | | | | | | | | | | | | | | | | | | √ | | | | √ | | |

# TON - Timer, On-Delay




Instruction Type: output

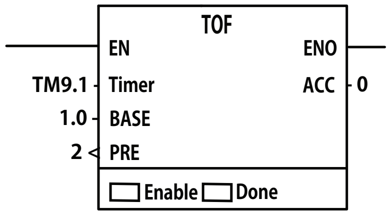
Use the TON instruction to delay turning on an output.

The TON instruction begins to count time base intervals when rung conditions become true. As long as rung conditions remain true, the timer increments its accumulator until the preset value is reached. When the accumulator equals the preset, timing stops.

The accumulator is reset (0) when rung conditions go false, regardless of whether the timer has timed out.

TON timers are reset on power cycles and mode changes.

# TOF - Timer, Off-Delay




Instruction Type: output

Use the TOF instruction to delay turning off an output.

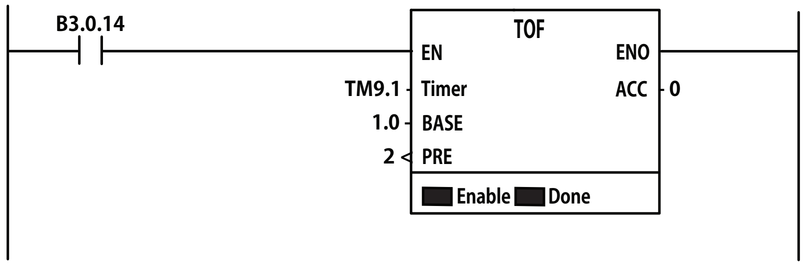
The TOF instruction begins to count time base intervals when rung conditions become false. As long as rung conditions remain false, the timer increments its accumulator until the preset value is reached.

The accumulator is reset (0) when rung conditions go true, regardless of whether the timer is timed out. TOF timers are reset on power cycles and mode changes.

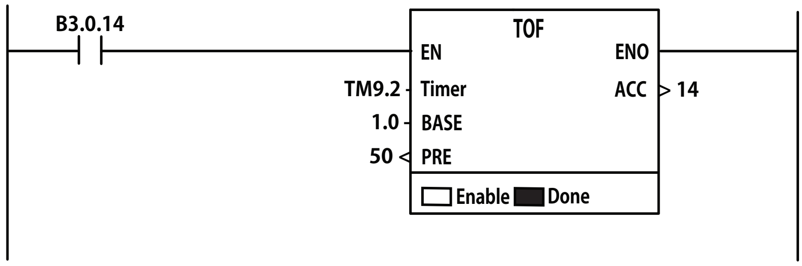
Note that, all status such as status flags are opposed to the TON instruction.

Followoing figures shows the timer status from begin to done.

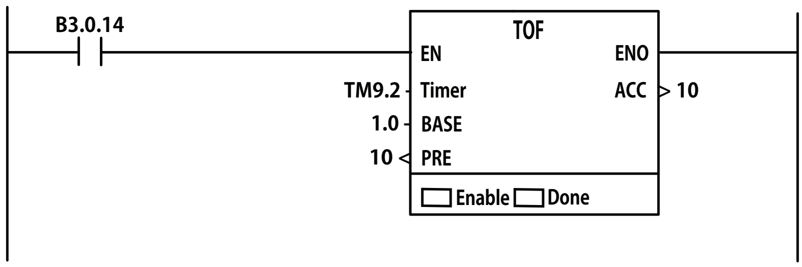
- If the rung condition is true, the TOF timer is not operate. Enable and Done bits are configured different from the TON timer.




- If the rung condition is false, the TOF timer is timing. As long as rung conditions remain false, the Enable bit is cleared. The Done bit is configured since the TOF timer is timing.



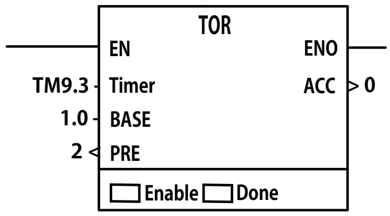

- If the rung condition is false , the TOF timer is timing. After the timer is stop, the Enable and Done bits are all cleared.



**ATTENTION**

Because the RSTA instruction resets the accumulated value and internal flags, do not use the RES instruction to reset a timer address used in a TOF instruction. If the TOF accumulated value and status bits are reset, unpredictable machine operation may occur.

## TOR - Retentive Timer, On-Delay




Instruction Type: output

Use the TOR instruction to delay turning on an output.

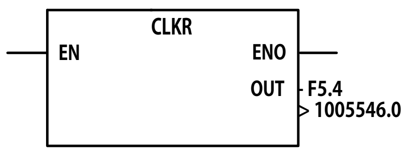
The TOR begins to count time base intervals when the rung conditions become true. As long as the rung conditions remain true, the timer increments its accumulator until the preset value is reached.

The TOR retains the accumulated value when the following occur:

- rung conditions become false
- you change the PLC mode from run or test to program
- a fault occurs

When you return the PLC to the RUN or TEST mode, or the rung conditions go true, timing continues from the retained accumulated value. TOR timers are retained through power cycles and mode changes.

## CLKR - Read High Speed Clock Time




Instruction Type: output

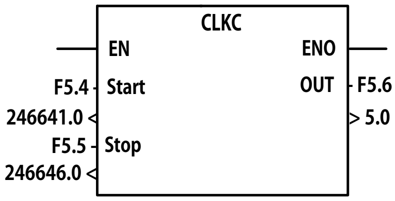
The CLKR instruction is provides a high performance timestamp.

This 32 bit value increments every 10 µs after power-up.

After the free running clock reaches 0xFFFFFFFF (42949.67295 seconds) value, it wraps around to 0 and continues incrementing.

| Parameter | Data Table | | | | | | | | | | | | | SFR | | | | | | | | | Address Mode | | | | | | Address Level | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Y | X | SR | B | TM, CT, CR | N | F | ST | A | L | MG, PD | RP | PS | RTC | HSC | PTO, PWM | STI | EII | BHI | MMI | CS-Comms | DLS-DataLog | Immediate | Direct | Indirect | Immdiate Float | Immediate Int32 | Immediate Imt16 | Bit | Word | LongWord | Element | Float |
| Output | | | | | | √ | √ | | | √ | | | | | | | | | | | | | | √ | | | | | √ | | | | |

## CLKC - Compute Time Difference




Instruction Type: output

It calculates time difference between two times gets from the CLKR instruction.

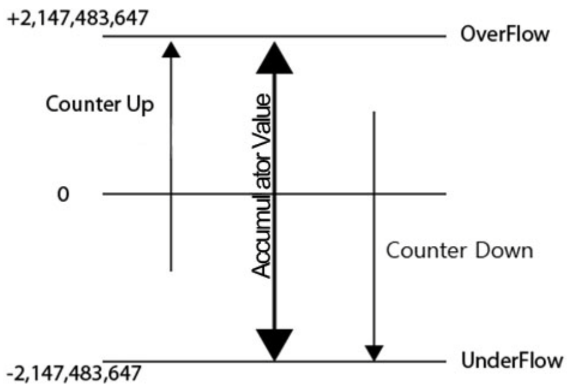
That is, this function is used to calculate time interval for each time difference when two events are generated.

Two input data should be same type.

| Parameter | Data Table | | | | | | | | | | | | | SFR | | | | | | | | | Address Mode | | | | | | Address Level | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Y | X | SR | B | TM, CT, CR | N | F | ST | A | L | MG, PD | RP | PS | RTC | HSC | PTO, PWM | STI | EII | BHI | MMI | CS-Comms | DLS-DataLog | Immediate | Direct | Indirect | Immdiate Float | Immediate Int32 | Immediate Imt16 | Bit | Word | LongWord | Element | Float |
| Input 1 | | | | | | √ | √ | | | √ | | | | | | | | | | | | | | √ | | | | | | | | | |
| Input 2 | | | | | | √ | √ | | | √ | | | | | | | | | | | | | | √ | | | | | | | | | |
| Output | | | | | | √ | √ | | | √ | | | | | | | | | | | | | | √ | | | | | | | | | |

## How Counters Work

The figure below demonstrates how a counter works. The count value must remain in the range of Low Preset -2,147,483,647 ~ High Preset +2,147,483,647. If the count value goes above +2,147,483,647, the counter status overflow bit is set (1). If the count goes below -2,147,483,647, the counter status underflow bit is set (1). The RSTA instruction is used to reset he counter.




### Using the CTU and CTD Instructions

To use counter instructions, CTU and CTD, the CT instruction in the data table is used. Following shows the elements of the CT instruction.

Data Table : CT10

| Address | Type | Value | Meaning | Description |
|---|---|---|---|---|
| ⊟ CT10.0 | Counter | {...} | | |
| Underflow | Bool | 0 | OFF | |
| Overflow | Bool | 0 | OFF | |
| Done | Bool | 0 | OFF | |
| CountDown | Bool | 0 | OFF | |
| CountUp | Bool | 0 | OFF | |
| Preset | Long | 0 | | |
| Accumulator | Long | 0 | | |
| ⊞ CT10.1 | Counter | {...} | | |
| ⊞ CT10.2 | Counter | {...} | | |
| ⊞ CT10.3 | Counter | {...} | | |
| ⊞ CT10.4 | Counter | {...} | | |

**TIP** The counter continues to count when the accumulator is greater than the CTU preset and when the accumulator is less than the CTD preset.

## CT.10.0.Underflow

| Address | Data Format | HSC Mode | Function | User Program Access |
|---|---|---|---|---|
| CT.10.0.Underflow | Bit | 0 or 1 | Status | Read / Write |

This bit is set when the accumulator value is less than -2,147,483,647.

## CT.10.0.Overflow

| Address | Data Format | HSC Mode | Function | User Program Access |
|---|---|---|---|---|
| CT.10.0.Overflow | Bit | 0 or 1 | Status | Read Only |

This bit is set when the accumulator value is greater than 2,147,483,647.

## CT.10.0.Done

| Address | Data Format | HSC Mode | Function | User Program Access |
|---|---|---|---|---|
| CT.10.0.Done | Bit | 0 or 1 | Status | Read Only |

This bit is set when the accumulator value is reach to Preset value.

## CT.10.0.CountDown

| Address | Data Format | HSC Mode | Function | User Program Access |
|---|---|---|---|---|
| CT.10.0.CountDown | Bit | 0 or 1 | Status | Read Only |

This bit is set when counter is down by the CTD instruction.

## CT.10.0.CountUp

| Address | Data Format | HSC Mode | Function | User Program Access |
|---|---|---|---|---|
| CT.10.0.CountUp | Bit | 0 or 1 | Status | Read Only |

This bit is set when counter is down by the CTD instruction.

## CT.10.0.Preset

| Address | Data Format | HSC Mode | Function | User Program Access |
|---|---|---|---|---|
| CT.10.0.Preset | LongWord | -32768 ~ 32767 | Control | Read / Write |

This is a counter preset. Done flag is set when the accumulator value is reach to Preset value.

This bit is set when counter is down by the CTD instruction.

## CT.10.0.Accumulator

| Address | Data Format | HSC Mode | Function | User Program Access |
|---|---|---|---|---|
| CT.10.0.Accumulator | LongWord | -32768 ~ 32767 | Control | Read / Write |

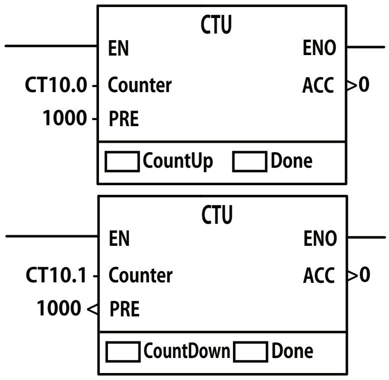
This flag contains currrent counter value. when the rung conditions go false-to-true, the accumulator value is increasd in the CTU instruction and decreased in the CTD instruction.

The accumulator is retentive when the rung conditions is false and on power cycles.

Accumulator value can be reset only by the RSTA instruction.

| Parameter | Data Table | | | | | | | | | | | | | SFR | | | | | | | | | Address Mode | | | Address Level | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Y | X | SR | B | TM, CT, CR | N | F | ST | A | L | MG, PD | RP | PS | RTC | HSC | PTO, PWM | STI | EII | BHI | MMI | CS-Comms | DLS-DataLog | Immediate | Direct | Indirect | Bit | Word | LongWord | Element |
| Counter | | | | | √ | | | | | | | | | | | | | | | | | | | √ | | | | | √ |
| Preset | | | | | | | | | | | | | | | | | | | | | | | √ | | | | √ | | |
| ACC | | | | | | | | | | | | | | | | | | | | | | | √ | | | | √ | | |

# CTU - Count Up
# CTD - Count Down

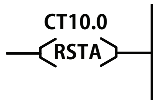



Instruction Type: output

The CTU and CTD instructions are used to increment or decrement a counter at each false-to-true rung transition.

**TIP** If the signal that makes rung conditions go false-to-true is coming from a field device, this signal should be maintained to check rung status transition in case of that the width of ON/OFF is less than the scan time.

# RSTA - Reset Accumulator




Instruction Type: output

The RSTA instruction resets the Counters, Timers and various contorl elements.

The follwong table shows the reset result by the RSTA instruction of each element.

| Timer Element | Couneter Element | Contorl Element |
|---|---|---|
| Accumulator -> 0 | Accumulator -> 0 | Position -> 0 |
| Done -> 0 | Overflow -> 0 | Enable -> 0 |
| TimerTiming -> 0 | Underflow -> 0 | Enable Unload -> 0 |
| Enable -> 0 | Done -> 0 | Done -> 0 |
| | CountUp -> 0 | Empty -> 0 |
| | CountDown -> 0 | Error -> 0 |
| | | Unload bit -> 0 |

| Parameter | Data Table | | | | | | | | | | | | | SFR | | | | | | | CS-Comms | DLS-DataLog | Address Mode | | | Address Level | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Y | X | SR | B | TM, CT, CR | N | F | ST | A | L | MG, PD | RP | PS | RTC | HSC | PTO, PWM | STI | EII | BHI | MMI | | | Immediate | Direct | Indirect | Bit | Word | LongWord | Element |
| Structure | | | | | √ | | | | | | | | | | | | | | | | | | | √ | | | | | √ |

# Compare Instructions

## Introduction

In this chapter, the X8 Series PLC compare Instructions are described.This chapter organized as follows

The following table shows the compare instructions.

| Instructions | Description |
|---|---|
| EQ : Compare for equal | ==<br>Compare whether two values are equal. |
| NE : Compare for not equal | !=. < >, ?<br>Compare whether one value is not equal to a second value. |
| LT : Compare for Less Than | <<br>Compare whether one value is less than a second value. |
| LE : Compare for Less Than or Equal | <=<br>Compare whether one value is less than or equal to a second value. |
| GT : Compare for Greater Than | ><br>Compare whether one value is greater than a second value. |
| GE - Compare for Greater Than or Equal | >=<br>Compare whether one value is greater than or equal to a second value. |
| EQM : Masked Compare for Equal | Compare two input values after Mask and AND |
| LIM - Limit Test | Compare whether one value is within the range of two other values |

| Parameter | Data Table | | | | | | | | | | | | | SFR | | | | | | | | | Address Mode | | | Address Level | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Y | X | SR | B | TM, CT, CR | N | F | ST | A | L | MG, PD | RP | PS | RTC | HSC | PTO, PWM | STI | EII | BHI | MMI | CS-Comms | DLS-DataLog | Immediate | Direct | Indirect | Bit | Word | LongWord | Element |
| Source A | √ | √ | √ | √ | √ | √ | √ | | | √ | √ | √ | √ | | √ | √ | √ | √ | √ | √ | √ | √ | | √ | √ | | √ | √ | √ |
| Source B | √ | √ | √ | √ | √ | √ | √ | | | √ | √ | √ | √ | | √ | √ | √ | √ | √ | √ | √ | √ | √ | √ | √ | | √ | √ | |

* Only use the High Speed Counter Accumulator for Source A in GT, LT, GE, and LE instructions.

## Using the Compare Instructions

Most of the compare instructions use two parameters, Source A and Source B (EQM and LIM have an additional parameter). Both sources cannot be constant values. But only one source of two can be constant value.

The valid data ranges for these instructions are:

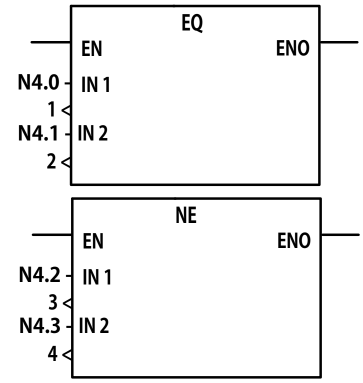
- Word (16Bit): 0 ~ 32,767
- Long Word (32Bit): 0 ~ 2,147,483,647

When at least one of the operands is a Floating Data Point value:

- For EQ, GE, GT, LE, and LT - If Source A is not a number, then rung state changes to false.

- For NE - Even the Source A and B is number, rung state remains true.

## EQ - Compare for Equal
## NE - Compare for Not Equal




Instruction Type: input

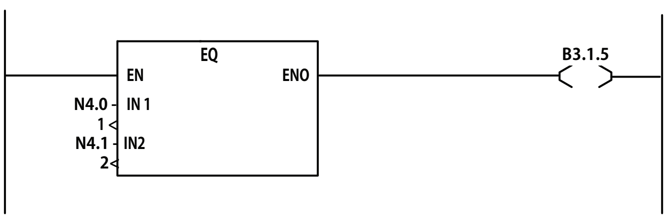
- The EQ instruction is used to compare whether one value is equal to a second value.
- The NE instruction is used to compare whether one value is not equal to a second value.

**EQ and NE Instruction Operations**

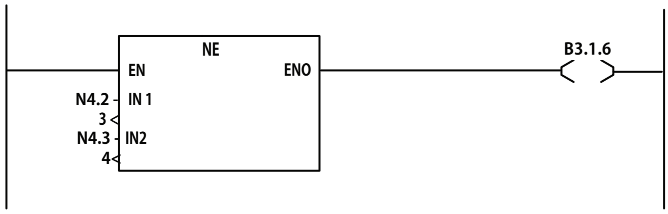
| Instruction | Relationship of Source Values | Resulting Rung State |
|---|---|---|
| EQ | A = B | TRUE |
| | A ≠ B | FALSE |
| NE | A = B | FALSE |
| | A ≠ B | TRUE |

For example:

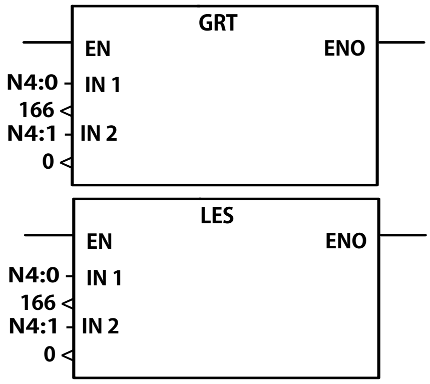
- If the EQ instruction is IN1=1, and IN2=2, because the two sources are not equal, output B3.1.5 bit state remains off.




- If the EQ instruction is IN1=3, and IN2=4, because the two sources are not equal, output B3.1.6 bit state becomes ON.




## GT - Compare for Greater Than
## LT - Compare for Less Than




Instruction Type: input

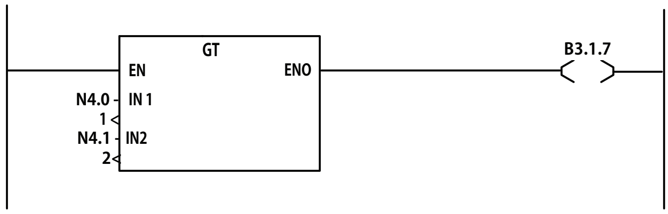
- The GT instruction is used to compare whether Source A value is greater than Source B value.
- The LT instruction is used to compare whether Source B value is greater than Source A value.

**GRT and LES Instruction Operation**

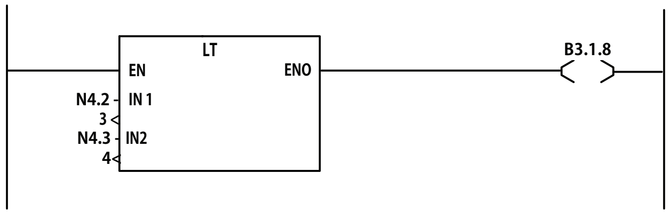
| Instruction | Relationship of Source Values | Resulting Rung State |
|---|---|---|
| GT | A > B | TRUE |
| | A ≤ B | FALSE |
| LT | A ≥ B | FALSE |
| | A < B | TRUE |

For example:

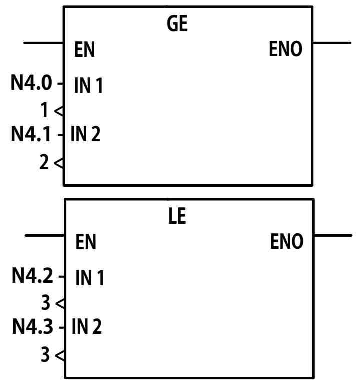
- If the GT instruction is IN1 =1, and IN2=1, it compare whether IN 1 is greater than IN 2. Since the two sources are equal, output B3.1.7 bit state remains OFF.




- If the LT instruction is IN1 =3, and IN2=4, it compare whether IN 1 is greater than IN 2. Since the IN 2 is greater than IN1, output B3.1.8 bit state remains ON.




## GE - Compare for Greater Than or Equal to LE - COmpare for Less than or Equal to




Instruction Type: input

The GE instruction is used to compare whether Source A value is greater than or equal to Source B value.

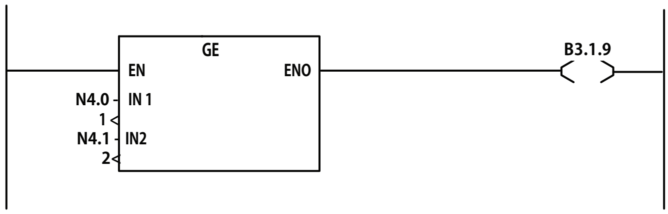
The LE instruction is used to compare whether Source B value is greater than or equal to Source A value.

**GE and LE Instruction Operation**

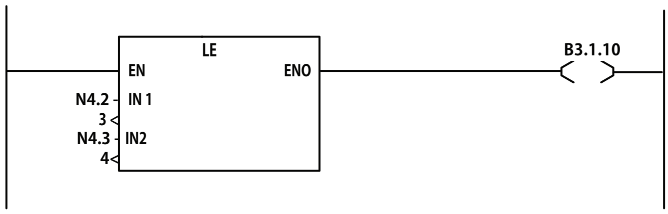
| Instruction | Relationship of Source Values | Resulting Rung State |
|---|---|---|
| GEQ | A ≥ B | true |
| | A < B | false |
| LEQ | A > B | false |
| | A ≤ B | true |

For example:

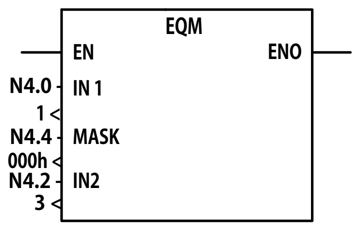
- If the GE instruction is IN1 =1, and IN2=1, it compares whether IN 1is greater than or equal to IN 2. Since the two sources are equal, output B3.1.9 bit state remains ON.




- If the LE instruction is IN1 =3, and IN2=4, it test whether IN 2 is greater than or equal to IN 1. Since the IN 2 is greater than the IN1, output B3.1.10 bit is remains ON.




# EQM - Mask Compare for Equal




Instruction Type: input

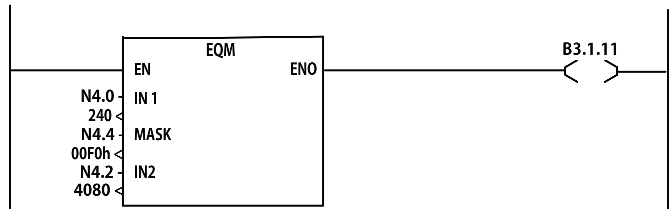
The MEQ instruction is used to compare whether Source A and result of Mask data value through ANDed is equal to Source B and result of Mask data through ANDed.

For example:

If the EQM instruction is IN1=240 (0F0H), and IN2=4080 (FF0H), compare whether the two values are equal to F0H through ANDes. Then, these results are compared to each other.

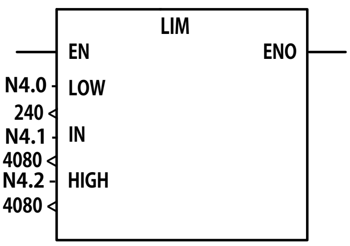
- IN1=240 (0F0H) AND MASK 0F0H = F0H
- IN1=240 (FF0H) AND MASK 0F0H = F0H

Because the Mask value of the two sources are equal, output B3.1.11 bit state remains ON.




| Parameter | Data Table | | | | | | | | | | | | | SFR | | | | | | | | | Address Mode | | | Address Level | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Y | X | SR | B | TM, CT, CR | N | F | ST | A | L | MG, PD | RP | PS | RTC | HSC | PTO, PWM | STI | EII | BHI | MMI | CS-Comms | DLS-DataLog | Immediate | Direct | Indirect | Bit | Word | LongWord | Element |
| Source | √ | √ | √ | √ | √ | √ | | | | √ | √ | √ | | | √ | √ | √ | √ | √ | √ | √ | √ | | √ | √ | | √ | √ | |
| Mask | √ | √ | √ | √ | √ | √ | √ | | | √ | √ | √ | √ | | √ | √ | √ | √ | √ | √ | √ | √ | √ | √ | √ | | √ | √ | |
| Compare | √ | √ | √ | √ | √ | √ | √ | | | √ | √ | √ | √ | | √ | √ | √ | √ | √ | √ | √ | √ | √ | √ | √ | | √ | √ | |

## LIM - Limit Test




Instruction Type: input

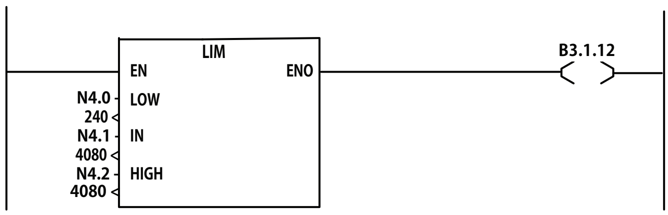
The LIM instruction is used to compare to values between Low Limit and High Limit.

**LIM Instruction Operation Based on Low Limit, Test, and High Limit Values**

| And: | Rung State |
|---|---|
| Low Limit ≤ Input≤ High Limit | true |
| Test < Low Limit or Input > High Limit | false |
| High Limit < Input < Low Limit | false |
| Test ≥ High Limit or Input ≤ Low Limit | true |

For Example:

- If the LIM instruction is IN =4080, Low Limit=240, High Limit=4080, since the source is between Low Limit and High Limit, output B3.1.12 bit state remains ON.




| Parameter | Data Table | | | | | | | | | | | | | SFR | | | | | | | | | Address Mode | | | Address Level | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Y | X | SR | B | TM, CT, CR | N | F | ST | A | L | MG, PD | RP | PS | RTC | HSC | PTO, PWM | STI | EII | BHI | MMI | CS-Comms | DLS-DataLog | Immediate | Direct | Indirect | Bit | Word | LongWord | Element |
| Low Limit | √ | √ | √ | √ | √ | √ | | | | √ | √ | √ | | | √ | √ | √ | √ | √ | √ | √ | √ | √ | √ | √ | | √ | √ | |
| Test | √ | √ | √ | √ | √ | √ | | | | √ | √ | √ | | | √ | √ | √ | √ | √ | √ | √ | √ | √ | √ | √ | | √ | √ | |
| High Limit | √ | √ | √ | √ | √ | √ | | | | √ | √ | √ | | | √ | √ | √ | √ | √ | √ | √ | √ | √ | √ | √ | | √ | √ | |

# Math Instructions

## Introduction

In this chapter, the X8 Series PLC math Instructions are described. This chapter organized as follows

The following table shows various math instructions and a specific arithmetic instructions supported from the X8 Series PLC use these output instructions to perform computations using an expression or a specific arithmetic instruction.

| Instructions | Descriptions |
|---|---|
| ADD | Add two values |
| SUB | Subtract two values |
| MUL | Multiply two values |
| DIV | Divide one value by another |
| NEG | Change the sign |
| CLR | Set all bits of a word to zero (clearer) |
| ABS | Find the absolute value of the source value |
| SQRT | Find the square root of a value, Root |
| SCAL | Scale a value |
| SCAP | Scale a value to a range determined by creating a linear relationship |
| SIN | SIN() |
| COS | CON() |
| TAN | TAN() |
| ASIN | ARC SIN() |
| ACOS | ARC COS() |
| ATAN | ARC TAN() |
| DEG | Convert Radian to Degrees |
| RAD | Convert Degrees to Radian |
| LN | Natural log |
| LOG | Log |
| POW | Raise a value to a power |
| CALC | Evaluate an expression and store the result in the destination. |

## Using the Math Instructions

The range of constants:

- Word: -32,768 ~ 32,767
- Long Word: -2,147,483648 ~ 2,147,483,647

Sources can be constants, but both sources cannot be constants.

Sources can be different data sizes, but output is stored to fit in the output data format.

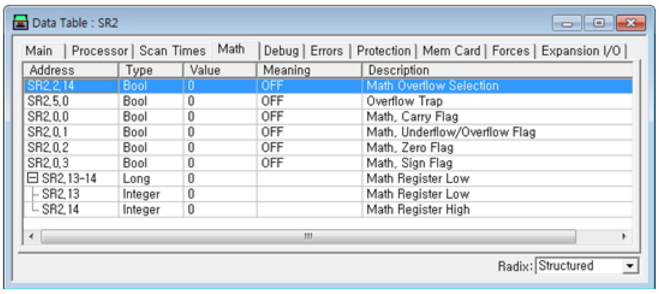
If the overflow bit is set, the overflow shall be handled as follows by using Math SR2.14 bit of the SR2 elements.

- Set(1): If the source is positive number, 32,767 (word) or 2,147,483,647 (Long Word) is stored as output value. If the source is negative number, -32,767 (WORD) or -2,147,483,647 (Long Word) is stored.
- Clear(0): According to each data type, the unsigned truncated value of the Source is stored.

## Math Arithmetic Flag

The SR flag shows Math and Arithmetic instruction state places Math category in the SR data table.

Following shows the descriptions of the each items.




| Address | Type | Value | Meaning | Description |
|---|---|---|---|---|
| SR2.2.14 | Bool | 0 | OFF | Math Overflow Selection |
| SR2.5.0 | Bool | 0 | OFF | Overflow Trap |
| SR2.0.0 | Bool | 0 | OFF | Math, Carry Flag |
| SR2.0.1 | Bool | 0 | OFF | Math, Underflow/Overflow Flag |
| SR2.0.2 | Bool | 0 | OFF | Math, Zero Flag |
| SR2.0.3 | Bool | 0 | OFF | Math, Sign Flag |
| SR2.13-14 | Long | 0 | | Math Register Low |
| SR2.13 | Integer | 0 | | Math Register Low |
| SR2.14 | Integer | 0 | | Math Register High |

- SR2.2.14: Math Overflow Selection
  If Overflow bit is set, output value can be different according to this bit.
- SR2.5.0: Overflow Trap
  This bit is set when the Overflow flag is set.
  Because the Overflow flag can effect to the system, this bit can be cleared forcedly on the ladder by the RST instruction.
- SR2.0.0: Carry Flag
  This bit is set when the Carry flag is set.
- SR2.0.1: Underflow , Overflow Flag
  This bit is set when the result of a math instruction does not fit into the output target range.
- SR2.0.2: Zero Flag
  This bit is set when the result is Zero.
- SR2.0.3 : Sign Flag
  This bit is set when the output result is negative. If negative value, the MSB (Most Significant Bit) is set.
- SR2.13: Register Low:
  The remainder is stored in SR2.13 when the division operation.
- SR2.14: Register High:
  The quotient is stored in SR2.14 when the division operation.

The data types of X8 Series PLC operations, and arithmetic instructions are determined according to table below.

| Input 1 | Input 2 | Output |
|---|---|---|
| Inteder | Integer | Integer |
| Integer | Long | Long |
| Integer | Float | Float |
| Long | Integer | Long |
| Long | Long | Long |
| Long | Float | Float |
| Float | Integer | Float |
| Float | Long | Float |
| Float | Float | Float |

## Floating Point

The **X8 Series PLC** supports Floating Point typed (F) 32 bit size data table, and Floating Point observes IEEE-754 specifications.

Like other data tables, it supports 1, 536 of elements.

Floating Point Type Data Structure

| MSB | | | | | | | | | | | | LSB |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 31 | 30 | 29 | ... | 23 | 22 | 21 | 20 | 19 | ... | 2 | 1 | 0 |
| Sign | Exponent Value | | | | Mantissa | | | | | | | |

- Bit 31 is MSB and means the sign bit.
- Bits 30 ~ 23 are the exponent.
- Bits 22 ~ 00 are the mantissa.

| Parameter | Data Table | | | | | | | | | | | | SFR | | | | | | | | | Address Mode | | | Address Level | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Y | X | SR | B | TM, CT, CR | N | F | ST | A | L | MG, PD | RP | PS | RTC | HSC | PTO, PWM | STI | EII | BHI | MMI | CS-Comms | DLS-DataLog | Immediate | Direct | Indirect | Bit | Word | LongWord | Element |
| Input 1 | √ | √ | √ | √ | √ | √ | √ | | | √ | √ | √ | | √ | √ | √ | √ | √ | √ | √ | √ | √ | √ | √ | √ | | √ | √ | |
| Input 2 | √ | √ | √ | √ | √ | √ | √ | | | √ | √ | √ | | √ | √ | √ | √ | √ | √ | √ | √ | √ | √ | √ | √ | | √ | √ | |
| Output | √ | √ | √ | √ | √ | √ | √ | | | √ | √ | √ | | √ | √ | √ | √ | √ | | | | | | √ | √ | | √ | √ | |

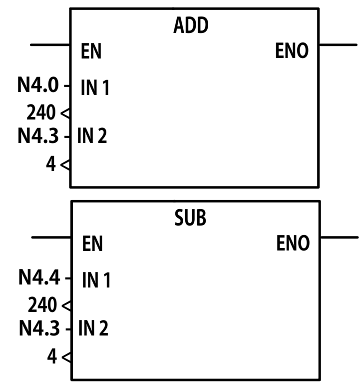
LSB Round to Even Rule

- Floating point operations are rounded. If the bits to the right of the LSB represent a value of exactly one-half LSB, the result is rounded up or down so that the LSB is an even number.

Floating Point Exception Values

- Zero: represented by an exponent and a mantissa of zero.
- Denormalized : Since denormalized numbers have very small, insignificant values, they are treated as zero in order to increase the performance.
- ∞ : represented by an exponent of 255 and a mantissa part of zero. Both positive and negative infinity are generated when operations overflow.
- NaN(Not a Number): represents mathematically undefined input and especially it is used in floating-point arithmetic. For example, most floating point devices given a invalid message and returns NaN value for operations about square root of negative.

## ADD - Add
## SUB - Subtract

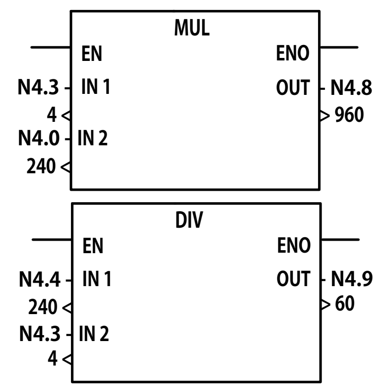



Instruction Type: Output

ADD: Use the ADD instruction to add one value to another value (Source A +Source B) and place the sum in the Destination.

SUB: Use the SUB instruction to subtract one value from another value (Source A - Source B) and place the result in the Destination.

## MUL - Multiply
## DIV - Divide

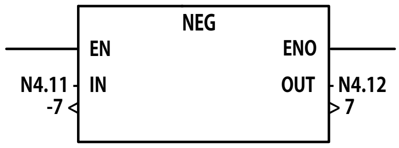



Instruction Type: input

MUL: Use the MUL instruction to multiply one value by another value (Source A x Source B) and place the result in the Destination.

DIV: Use the DIV instruction to divide one value by another value (Source A/ Source B) and place the result in the Destination.

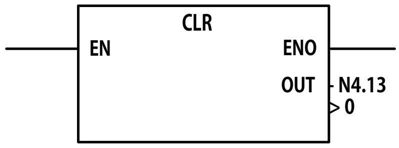
## NEG - Negate




Instruction Type: Output

Use the NEG instruction to change the sign of the Source and place the result in the Destination.

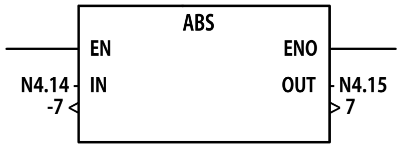
## CLR- Clear




Instruction Type: Output

Use the CLR instruction to set the specified data table to zero.

## ABS - Absolute Value




Instruction Type: Output

The ABS instruction takes the absolute value of the Source and places it in the Destination.

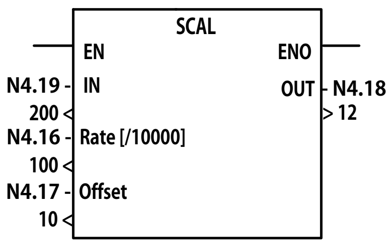
If the signed result does not fit in Destination, the following will occur.

| Input | Output | Considerations When Using Floating Point Data |
|---|---|---|
| F | N | •rounded data is output<br>•If an overflow occurs after rounding, 32767 or 2147836647 is output, SR2.2.14 Math Overflow Selection bit is ignored.<br>• Carry flag is reset.<br>•If all of Destination is zero, the Zero flag is set.<br>•If Overflow is set, the Overflow Trap flag is set.<br>•It is set if the result is ∞ , NAN or overrange<br>•If the MSB is set, the Sign flag is set. |
| ∞ | N | 32,767 or 2,147,836,647 is stored |
| N | N | •If the SR2.2.14 Math Overflow Selection bit is clearbd, 32,767 or 2,147,836,647 is output.<br>•The Carry flag is set when the input is negative<br>• If all of Destination is zero, the Zero flag is set.<br>•The Overflow Trap flag is set when the Overflow flag is set |

The following table shows data tables of the ABS instruction:

| Parameter | Data Table | | | | | | | | | | | | | SFR | | | | | | | | | Address Mode | | | Address Level | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Y | X | SR | B | TM, CT, CR | N | F | ST | A | L | MG, PD | RP | PS | RTC | HSC | PTO, PWM | STI | EII | BHI | MMI | CS-Comms | DLS-DataLog | Immediate | Direct | Indirect | Bit | Word | LongWord | Floating | Element |
| Input 1 | √ | √ | √ | √ | √ | √ | √ | | | √ | √ | √ | | √ | √ | √ | √ | √ | √ | √ | √ | √ | √ | √ | √ | | √ | √ | √ | |
| Output | √ | √ | √ | √ | √ | √ | √ | | | √ | √ | √ | | √ | √ | √ | √ | √ | | | | | | √ | √ | | √ | √ | √ | |

# SCAL - Scale with Slope




Instruction Type: Output

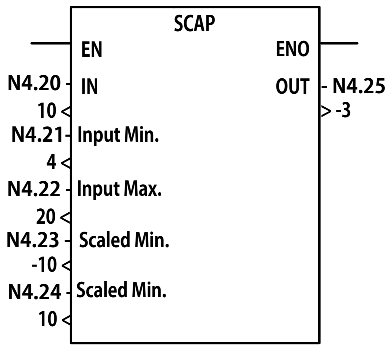
The SCAL instruction causes the value at the Source address to be multiplied by the Rate (slope) value. The resulting value is added to the Offset and the rounded result is placed in the Destination.

$$\text{Scaled Value} = \frac{\text{Input x Slope}}{10000} + \text{offset}$$

The following table shows data tables of SCAL instruction:

| Parameter | Data Table | | | | | | | | | | | | | SFR | | | | | | | | | Address Mode | | | Address Level | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Y | X | SR | B | TM, CT, CR | N | F | ST | A | L | MG, PD | RP | PS | RTC | HSC | PTO, PWM | STI | EII | BHI | MMI | CS-Comms | DLS-DataLog | Immediate | Direct | Indirect | Bit | Word | LongWord | Element |
| Input 1 | √ | √ | | √ | √ | √ | | | | | | √ | | | | | | | | | | | | √ | √ | | √ | | |
| Rate | √ | √ | | √ | √ | √ | | | | | | √ | | | | | | | | | | | √ | √ | √ | | √ | | |
| Offset | √ | √ | | √ | √ | √ | | | | | | √ | | | | | | | | | | | √ | √ | √ | | √ | | |
| Output | √ | √ | | √ | √ | √ | | | | | | √ | | | | | | | | | | | | √ | √ | | √ | | |

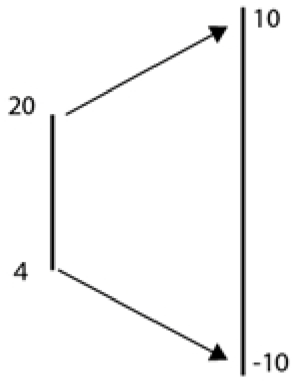
## SCAP - Scale with Parameters




Instruction Type: Output

The SCAP instruction produces a scaled output value that has a linear relationship between the input and scaled values.

The following example shows that if the left ladder number range is 4 ~ 20, and scaled output values are -10 ~ 10 , when the value is 10, scaled value becomes -3. (Convert 4~20mA input to -10V ~ 10V output )

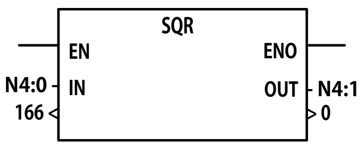



This instruction solves the following equation listed below to determine scaled output:

$$\text{Scaled Value} = \left[\frac{(y1 - y0)}{(x1 - x0)}\right] X\ (x - x0) + y0$$

| Parameter | Data Table | | | | | | | | | | | | | SFR | | | | | | | | | Address Mode | | | Address Level | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Y | X | SR | B | TM, CT, CR | N | F | ST | A | L | MG, PD | RP | PS | RTC | HSC | PTO, PWM | STI | EII | BHI | MMI | CS-Comms | DLS-DataLog | Immediate | Direct | Indirect | Bit | Word | LongWord | Element |
| Input | √ | √ | √ | √ | √ | √ | | | | √ | √ | √ | | √ | √ | √ | √ | √ | √ | √ | √ | √ | | √ | √ | | √ | √ | |
| Input Min. | √ | √ | | √ | | | | | | √ | | | | | | | | | | | | | √ | √ | √ | | √ | √ | |
| Input Max. | √ | √ | | √ | | | | | | √ | | | | | | | | | | | | | √ | √ | √ | | √ | √ | |
| Scaled Min. | √ | √ | | √ | √ | √ | | | | √ | | √ | | | | | | | | | | | √ | √ | √ | | √ | √ | |
| Scaled Max. | √ | √ | | √ | √ | √ | | | | √ | | √ | | | | | | | | | | | √ | √ | √ | | √ | √ | |
| Output | √ | √ | √ | √ | √ | √ | | | | √ | √ | √ | | √ | | √ | √ | √ | | | | | | √ | √ | | √ | √ | |

## SIN -Sine

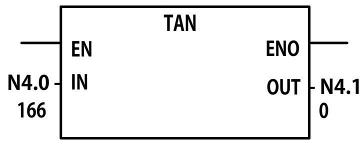



Instruction Type: Output

The SIN instruction places the sine of the Source (in radians) in the Destination. Address Levels for the operands involved in the SIN can be ALL word, ALL Long word, ALL float, or a combination. These operands shall undergo a conversion to float. The calculation of the source (in float) is then performed, and the result is then cast to the data type of Destination.

| Parameter | Data Table | | | | | | | | | | | | | SFR | | | | | | | | | Address Mode | | | | | | Address Level | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Y | X | SR | B | TM, CT, CR | N | F | ST | A | L | MG, PD | RP | PS | RTC | HSC | PTO, PWM | STI | EII | BHI | MMI | CS-Comms | DLS-DataLog | Immediate | Direct | Indirect | Immediate Float | Immediate Imt32 | Immediate Imt16 | Bit | Word | LongWord | Element | Float |
| Source | √ | √ | | √ | √ | √ | √ | | | √ | | | | | | | | | | | | | √ | √ | √ | √ | √ | √ | | √ | √ | | √ |
| Destination | √ | √ | | √ | √ | √ | √ | | | √ | | | | | | | | | | | | | | √ | √ | | | | | √ | √ | | √ |

# TAN - Tangent

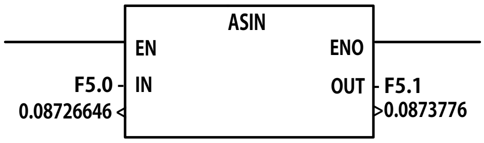



Instruction Type: Output

The TAN instruction places the sine of the Source (in radians) in the Destination. Address Levels for the operands involved in the TAN can be ALL word, ALL Long word, ALL float, or a combination. These operands shall undergo a conversion to float. The calculation of the source (in float) is then performed, and the result is then cast to the data type of Destination.

| Parameter | Data Table | | | | | | | | | | | | | SFR | | | | | | | | | Address Mode | | | | | | Address Level | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Y | X | SR | B | TM, CT, CR | N | F | ST | A | L | MG, PD | RP | PS | RTC | HSC | PTO, PWM | STI | EII | BHI | MMI | CS-Comms | DLS-DataLog | Immediate | Direct | Indirect | Immediate Float | Immediate Imt32 | Immediate Imt16 | Bit | Word | LongWord | Element | Float |
| Source | √ | √ | | √ | √ | √ | √ | | | √ | | | √ | | | | | | | | | | √ | √ | √ | √ | √ | √ | | √ | √ | | √ |
| Destination | √ | √ | | √ | √ | √ | √ | | | √ | | | √ | | | | | | | | | | | √ | √ | | | | | √ | √ | | √ |

# ASIN-Arc Sine



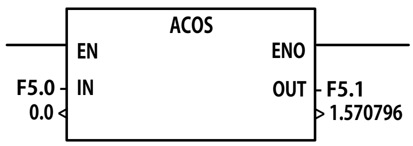

Instruction Type: Output

Use the ASN instruction to take the arc sine of a number and store the result (in radians) in the destination. Address Levels for the operands involved in the ASIN can be ALL word, ALL Long word, ALL float, or a combination. These operands shall undergo a conversion to float. The calculation of the source (in float) is then performed, and the result is then cast to the data type of Destination.

Source is always greater than or equal to -1 and less than or equal to 1. Destination is always greater than or equal to -π/2 and less than or equal to π/2.

| Parameter | Data Table | | | | | | | | | | | | | SFR | | | | | | | | | Address Mode | | | | | | Address Level | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Y | X | SR | B | TM, CT, CR | N | F | ST | A | L | MG, PD | RP | PS | RTC | HSC | PTO, PWM | STI | EII | BHI | MMI | CS-Comms | DLS-DataLog | Immediate | Direct | Indirect | Immediate Float | Immediate Imt32 | Immediate Imt16 | Bit | Word | LongWord | Element | Float |
| Source | √ | √ | | √ | √ | √ | √ | | | √ | | | √ | | | | | | | | | | √ | √ | √ | √ | √ | √ | | √ | √ | | √ |
| Destination | √ | √ | | √ | √ | √ | √ | | | √ | | | √ | | | | | | | | | | | √ | √ | | | | | √ | √ | | √ |

# ACOS - Arc Cosine



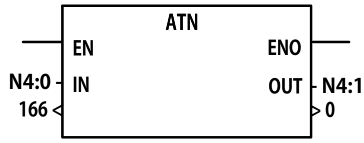

Instruction Type: Output

Use the ACOS instruction to take the Arc Cosine of a number (source in radians) and store the result (in radians) in the destination. Address Levels for the operands involved in the ACOS can be ALL word, ALL Long word, ALL float, or a combination. These operands shall undergo a conversion to float. The calculation of the source (in float) is then performed, and the result is then cast to the data type of Destination.

Source is always greater than or equal to -1 and less than or equal to 1. Destination is always greater than or equal to 0 and less than or equal to π.

| Parameter | Data Table | | | | | | | | | | | | | SFR | | | | | | | | Address Mode | | | | | | Address Level | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Y | X | SR | B | TM, CT, CR | N | F | ST | A | L | MG, PD | RP | PS | RTC | HSC | PTO, PWM | STI | EII | BHI | MMI | CS-Comms | DLS-DataLog | Immediate | Direct | Indirect | Immediate Float | Immediate Imt32 | Immediate Imt16 | Bit | Word | LongWord | Element | Float |
| Input | √ | √ | | √ | √ | √ | √ | | | √ | | | √ | | | | | | | | | | √ | √ | √ | √ | √ | √ | | √ | √ | | √ |
| Output | √ | √ | | √ | √ | √ | √ | | | √ | | | √ | | | | | | | | | | | √ | √ | | | | | √ | √ | | √ |

# ATAN - Arc Tangent



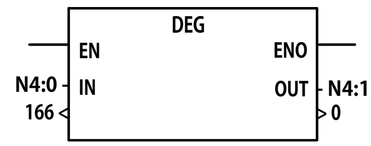

Instruction Type: Output

Use the ATAN instruction to take the arc tangent of a number (source) and store the result (in radians) in the destination. Address Levels for the operands involved in the ATAN can be ALL word, ALL Long word, ALL float, or a combination. These operands shall undergo a conversion to float. The calculation of the source (in float) is then performed, and the result is then cast to the data type of Destination.

Destination data always greater than or equal to $-\pi/2$, and less than or equal to $\pi/2$.

| Parameter | Data Table | | | | | | | | | | | | | SFR | | | | | | | | | Address Mode | | | | | | Address Level | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Y | X | SR | B | TM, CT, CR | N | F | ST | A | L | MG, PD | RP | PS | RTC | HSC | PTO, PWM | STI | EII | BHI | MMI | CS-Comms | IOS-I/O | DLS-DataLog | Immediate | Direct | Indirect | Immediate Float | Immediate Imt32 | Immediate Imt16 | Bit | Word | LongWord | Element | Float |
| Source | √ | √ | | √ | √ | √ | √ | | | √ | | | √ | | | | | | | | | | | √ | √ | √ | √ | √ | √ | | √ | √ | | √ |
| Destination | √ | √ | | √ | √ | √ | √ | | | √ | | | √ | | | | | | | | | | | | √ | √ | | | | | √ | √ | | √ |

# DEG - Radians to Degrees




Instruction Type: Output

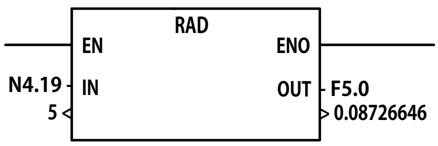
The DEG instruction converts the Source(in radians) to degrees and store the result in the Destination.

The formula applies:

Degree = Source * 180/π

| Parameter | Data Table | | | | | | | | | | | | | SFR | | | | | | | | | Address Mode | | | | | | Address Level | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Y | X | SR | B | TM, CT, CR | N | F | ST | A | L | MG, PD | RP | PS | RTC | HSC | PTO, PWM | STI | EII | BHI | MMI | CS-Comms | DLS-DataLog | Immediate | Direct | Indirect | Immediate Float | Immediate Imt32 | Immediate Imt16 | Bit | Word | LongWord | Element | Float |
| Source | √ | √ | | √ | √ | √ | √ | | | √ | | | √ | | | | | | | | | | √ | √ | √ | √ | √ | √ | | √ | √ | | √ |
| Destination | √ | √ | | √ | √ | √ | √ | | | √ | | | √ | | | | | | | | | | | √ | √ | | | | | √ | √ | | √ |

# RAD - Degrees to Radians




Instruction Type: Output

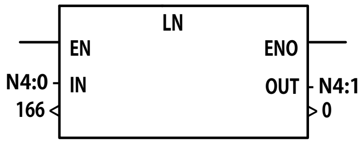
The RAD instruction converts the Source (in degrees) to radians and stores the result in the Destination.

The formula applies:

Radian = Source X π/180

| Parameter | Data Table | | | | | | | | | | | | | SFR | | | | | | | | | Address Mode | | | | | | Address Level | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Y | X | SR | B | TM, CT, CR | N | F | ST | A | L | MG, PD | RP | PS | RTC | HSC | PTO, PWM | STI | EII | BHI | MMI | CS-Comms | DLS-DataLog | Immediate | Direct | Indirect | Immediate Float | Immediate Imt32 | Immediate Imt16 | Bit | Word | LongWord | Element | Float |
| Source | √ | √ | | √ | √ | √ | √ | | | √ | | | √ | | | | | | | | | | √ | √ | √ | √ | √ | √ | | √ | √ | | √ |
| Destination | √ | √ | | √ | √ | √ | √ | | | √ | | | √ | | | | | | | | | | | √ | √ | | | | | √ | √ | | √ |

# LN - Natural Log

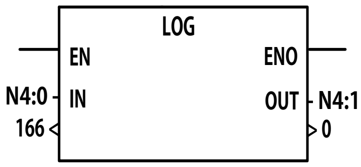



Instruction Type: output

Use the LN instruction to take the natural log of the value in the source and store the result in the destination.

| Parameter | Data Table | | | | | | | | | | | | | SFR | | | | | | | | | Address Mode | | | | | | Address Level | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Y | X | SR | B | TM, CT, CR | N | F | ST | A | L | MG, PD | RP | PS | RTC | HSC | PTO, PWM | STI | EII | BHI | MMI | CS-Comms | DLS-DataLog | Immediate | Direct | Indirect | Immediate Float | Immediate Imt32 | Immediate Imt16 | Bit | Word | LongWord | Element | Float |
| Source | √ | √ | | √ | √ | √ | √ | | | √ | | | √ | | | | | | | | | | √ | √ | √ | √ | √ | √ | | √ | √ | | √ |
| Destination | √ | √ | | √ | √ | √ | √ | | | √ | | | √ | | | | | | | | | | | √ | √ | | | | | √ | √ | | √ |

# LOG- Base 10 Logarithm

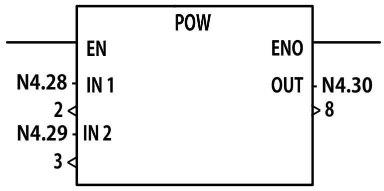



Instruction Type: output

Use the LOG instruction to take the log base 10 of the value in the source and store the result in the destination. The source must be greater than zero.

| Parameter | Data Table | | | | | | | | | | | | | SFR | | | | | | | | | Address Mode | | | | | | Address Level | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Y | X | SR | B | TM, CT, CR | N | F | ST | A | L | MG, PD | RP | PS | RTC | HSC | PTO, PWM | STI | EII | BHI | MMI | CS-Comms | DLS-DataLog | Immediate | Direct | Indirect | Immediate Float | Immediate Imt32 | Immediate Imt16 | Bit | Word | LongWord | Element | Float |
| Source | √ | √ | | √ | √ | √ | √ | | | √ | | | √ | | | | | | | | | | √ | √ | √ | √ | √ | √ | | √ | √ | | √ |
| Output | √ | √ | | √ | √ | √ | √ | | | √ | | | √ | | | | | | | | | | | √ | √ | | | | | √ | √ | | √ |

# POW - X Power Y



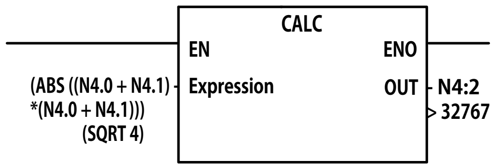

Instruction Type: output

Use the POW instruction to raise source 1 (x) to a power source 2(y) and store the result in the destination.

Destination = $x^{y}$

| Parameter | Data Table | | | | | | | | | | | | | SFR | | | | | | | | | Address Mode | | | | | | Address Level | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Y | X | SR | B | TM, CT, CR | N | F | ST | A | L | MG, PD | RP | PS | RTC | HSC | PTO, PWM | STI | EII | BHI | MMI | CS-Comms | DLS-DataLog | Immediate | Direct | Indirect | Immediate Float | Immediate Imt32 | Immediate Imt16 | Bit | Word | LongWord | Element | Float |
| Source | √ | √ | | √ | √ | √ | √ | | | √ | | | √ | | | | | | | | | | √ | √ | √ | √ | √ | √ | | √ | √ | | √ |
| Destination | √ | √ | | √ | √ | √ | √ | | | √ | | | √ | | | | | | | | | | | √ | √ | | | | | √ | √ | | √ |

# CALC - Calculate




Instruction Type: output

The CALC instruction is function for user's convenience and performs arithmetic operations by a instruction through this when you use **X8 Series PLC**.

The following table shows the expressions.

| Instructions | Sv j bol | Description |
|---|---|---|
| ADD | + | Addition |
| SUB | - | Subtraction |
| MUL | * | Multiplication |

| Instructions | Sumbol | Description |
|---|---|---|
| DIV | / | Division |
| SQRT | SQRT | Squa0b Root |
| SIN | SIN | Sine |
| COS | COS | Cosine |
| TAN | TAN | Tangent |
| ASIN | ASIN | Arc Sine |
| ACOS | ACOS | Arc Cosine |
| ATAN | ATAN | Arc Tangent |
| AND | AND | Logical And |
| OR | OR | Logical Inclusive Or |
| XOR | XOR | Logical Exclusive Or |
| NOT | NOT | Logical Not |
| NEG | - (NEG) | Negative |
| LN | LN | Natu0al Log |
| LOG | LOG | Base 10 Log |
| ABS | ABS | Absolute Value |
| DEG | DEG | Radians to Degrees |
| RAD | RAD | Degrees to Radians |
| POW | POW | X to the Power Y |
| BCD | BCD | BIN to BCD |
| BIN | BIN | BCD to BIN |

If the expressions are input to the system, the parenthesis '( )'are input automatically.

# Converion Instructions

## Instruction

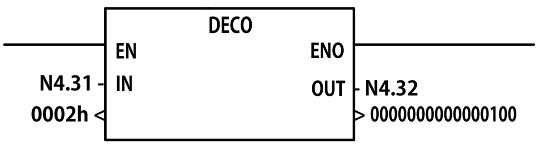
In this chapter, the X8 Series PLC conversion Instructions are described. This chapter organized as follows.

The conversion instructions multiplex and de-multiplex data and perform conversions between binary and decimal values.

| Instruction | Description |
|---|---|
| DECO | Decodes 4 to 1-of-16<br>Decodes a 4-bit value , turning on the corresponding bit in the 16-bit destination. |
| ENCO | Encode 1-of-16 to 4.<br>Encodes a 16-bit source to a 4-bit value. |
| BIN | Convert From Binary Coded Decimal<br>Convert BCD source to decimal values. |
| TOD - Convert to Binary Coded Decimal | Converts the integer source value to BCD format and stores it in the destination. |
| GCD - Gray Code | Converts Gray code data (Source) to an integer value and stores it in the destination. |

| Parameter | Data Table | | | | | | | | | | | | | SFR | | | | | | | | | Address Mode | | | Address Level | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Y | X | SR | B | TM, CT, CR | N | F | ST | A | L | MG, PD | RP | PS | RTC | HSC | PTO, PWM | STI | EII | BHI | MMI | CS-Comms | DLS-DataLog | Immediate | Direct | Indirect | Bit | Word | LongWord | Element |
| Source | √ | √ | | √ | √ | √ | | | | | | √ | | | | | | | | | | | | √ | √ | | √ | | |
| Destination | √ | √ | | √ | √ | √ | | | | | | √ | | | | | | | | | | | | √ | √ | | √ | | |

## DECD - Decode 4 to 1-of-16




Instruction Type: output

The DECD instruction uses the lower four bits of the source word to set multiplex 16 bit of the destination word.

The 4 bits of the source to destination shown in the table below:

**Decode 4 to 1-of-16**

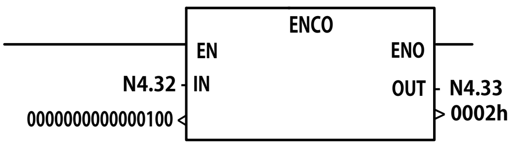
| Source Bits | | | | | Destination Bits | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 15 to 04 | 03 | 02 | 01 | 00 | 15 | 14 | 13 | 12 | 11 | 10 | 09 | 08 | 07 | 06 | 05 | 04 | 03 | 02 | 01 | 00 |
| X | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| X | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |

**Decode 4 to 1-of-16**

| Source Bits | | | | | Destination Bits | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| X | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 |
| X | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 |
| X | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| X | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |
| X | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| X | 0 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| X | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| X | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| X | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| X | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| X | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| X | 1 | 1 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| X | 1 | 1 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| X | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

X = not used

# ENCO - Encode 1-of-16 to 4




Instruction Type: Output

The ENCO instruction uses the lower four bits of the source word to set multiplex 16 bit of the destination word.

The following table shows math status bits.

| SR Address | Flag | Description |
|---|---|---|
| SR2.0.0 | Carry Flag | always resets |
| SR2.0.1 | Overflow Flag | sets if more than one bit in the source is set; otherwise resets. |
| SR2.0.2 | Zero Flag | sets if result is zero, otherwise resets |
| SR2.0.3 | Sign Flag | always resets |

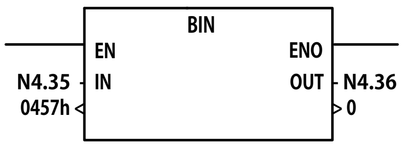
The ENCO instruction converts the values as shown in the table below:

**Encode 1-of-16 to 4**

| Source Bits | | | | | | | | | | | | | | | | Destination Bits | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **15** | **14** | **13** | **12** | **11** | **10** | **09** | **08** | **07** | **06** | **05** | **04** | **03** | **02** | **01** | **00** | **15 to 04** | **03** | **02** | **01** | **00** |
| X | X | X | X | X | X | X | X | X | X | X | X | X | X | X | 1 | 0 | 0 | 0 | 1 | 0 |
| X | X | X | X | X | X | X | X | X | X | X | X | X | X | 1 | 0 | 0 | 0 | 0 | 1 | 1 |
| X | X | X | X | X | X | X | X | X | X | X | X | X | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| X | X | X | X | X | X | X | X | X | X | X | X | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| X | X | X | X | X | X | X | X | X | X | X | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 |
| X | X | X | X | X | X | X | X | X | X | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 |
| X | X | X | X | X | X | X | X | X | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 |
| X | X | X | X | X | X | X | X | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 |
| X | X | X | X | X | X | X | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 0 |
| X | X | X | X | X | X | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 1 |
| X | X | X | X | X | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 1 |
| X | X | X | X | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 1 |
| X | X | X | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 0 |
| X | X | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 |
| X | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 |
| 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 2 |

x = determines the state of the flag

# BIN - Convert form Binary Coded Decimal (BCD)



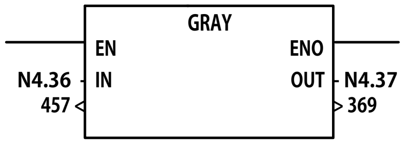

Instruction Type: Output

The BIN instruction is used to convert the Binary Coded Decimal (BCD) source value to an integer and place the result in the destination.

| SR Address | Flag | Description |
|---|---|---|
| SR2.0.0 | Carry Flag | always resets |
| SR2.0.1 | Overflow Flag | sets if more than one bit in the source is set; otherwise resets. |
| SR2.0.2 | Zero Flag | sets if result it zero, otherwise resets |
| SR2.0.3 | Sign Flag | always resets |

| Parameter | Data Table | | | | | | | | | | | | | SFR | | | | | | | | | Address Mode | | | Address Level | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Y | X | SR | B | TM, CT, CR | N | F | ST | A | L | MG, PD | RP | PS | RTC | HSC | PTO, PWM | STI | EII | BHI | MMI | CS-Comms | DLS-DataLog | Immediate | Direct | Indirect | Bit | Word | LongWord | Element |
| Source | √ | √ | √ | √ | √ | √ | | | | | | √ | | | | | | | | | | | | √ | √ | | √ | | |
| Destination | √ | √ | | √ | √ | √ | | | | | | √ | | | | | | | | | | | | √ | √ | | √ | | |

# GRAY - Gray Code




Instruction Type: output

The GRAY instruction converts GRAY code data (Source) to an Decimal .

| SR Address | Flag | Description |
|---|---|---|
| SR2.0.0 | Carry Flag | always resets |
| SR2.0.1 | Overflow Flag | sets if more than one bit in the source is set; otherwise resets. |
| SR2.0.2 | Zero Flag | sets if result it zero, otherwise resets |
| SR2.0.3 | Sign Flag | always resets |
| SR2.5.0 | Overflow Trap | sets if Overflow flag is set. otherwise resets. |

| Parameter | Data Table | | | | | | | | | | | | | SFR | | | | | | | | | Address Mode | | | Address Level | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Y | X | SR | B | TM, CT, CR | N | F | ST | A | L | MG, PD | RP | PS | RTC | HSC | PTO, PWM | STI | EII | BHI | MMI | CS-Comms | DLS-DataLog | Immediate | Direct | Indirect | Bit | Word | LongWord | Element |
| Source | √ | √ | √ | √ | √ | √ | | | | | | √ | | | | | | | | | | | | √ | √ | | √ | | |
| Output | √ | √ | | √ | √ | √ | | | | | | √ | | | | | | | | | | | | √ | √ | | √ | | |

# Logical Instructions

## Introduction

This chapter describes Logical Instructions of X8 Series PLC. The chapter is organized as follows.

The follwing table shows 4 logical instructions of the X8 Series PLC.

**Application Specific Insturcitons**

| Instruction | Used To: |
|---|---|
| AND | Bit-Wise AND<br>Perform an AND operation |
| OR | Logical OR<br>Perform an inclusive OR operation |
| XOR | Exclusive OR<br>Perform an Exclusive Or operation |
| NOT | Logical NOT<br>Perform a NOT operation |

When using logical instructions, data range is as follow:

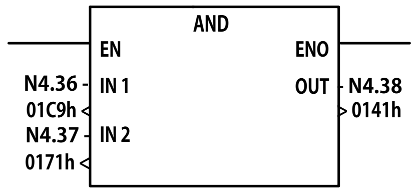
- Word: -32,768 ~ 32,767.
- Long Word: -2,147,483,648 ~ 2,147,483,647.

**Math Status Bits**

| With this Bit | | The Controller: |
|---|---|---|
| SR2.0.0 | Carry Flag | always resets |
| SR2.0.1 | Overflow Flag | always resets |
| SR2.0.2 | Zero Flag | sets if result it zero, otherwise resets |
| SR2.0.3 | Sign Flag | sets if result is negative (MSB is set), otherwise resets |

| Parameter | Data Table | | | | | | | | | | | | | SFR | | | | | | | | | Address Mode | | | Address Level | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Y | X | SR | B | TM, CT, CR | N | F | ST | A | L | MG, PD | RP | PS | RTC | HSC | PTO, PWM | STI | EII | BHI | MMI | CS-Comms | DLS-DataLog | Immediate | Direct | Indirect | Bit | Word | LongWord | Element |
| Source 1 | √ | √ | √ | √ | √ | √ | | | | √ | √ | √ | | √ | √ | √ | √ | √ | √ | √ | √ | √ | √ | √ | √ | | √ | √ | |
| Source 2 | √ | √ | √ | √ | √ | √ | | | | √ | √ | √ | | √ | √ | √ | √ | √ | √ | √ | √ | √ | √ | √ | √ | | √ | √ | |
| Destination | √ | √ | √ | √ | √ | √ | | | | √ | √ | √ | | √ | | √ | √ | √ | | | | | | √ | √ | | √ | √ | |

# AND - Bit-Wise AND




Instruction Type: output

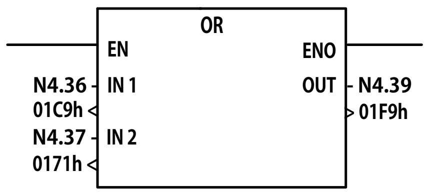
The AND instruction performs a bit-wise logical AND of two sources and places the result in the destination.

| | | Source | |
|---|---|---|---|
| | | 0 | 1 |
| Source | 0 | 0 | 0 |
| | 1 | 0 | 1 |

The above example means:

| | | |
|---|---|---|
| Source 1: | 01C9H | = 0000 0001 1100 1001 |
| Source 2: | 0171H | = 0000 0001 0111 0001 |
| Destination | | = 0000 0001 0100 0001 = 0141H- |

# OR - Logical OR




Instruction Type: output

The OR instruction performs a logical OR of two sources and places the result in the destination.

| | | Source | |
|---|---|---|---|
| | | 0 | 1 |
| Source | 0 | 0 | 1 |
| | 1 | 1 | 1 |

The above example means:

| | | |
|---|---|---|
| Source 1: | 01C9H | = 0000 0001 1100 1001 |
| Source 2: | 0171H | = 0000 0001 0111 0001 |
| Destination | | = 0000 0001 0100 0001 = 01F9H |

# XOR - Exclusive OR


Instruction Type: output

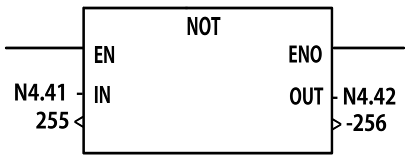
The XOR instruction performs a logical exclusive OR of two sources and places the result in the destination.

| | | Source | |
|---|---|---|---|
| | | 0 | 1 |
| Source | 0 | 0 | 1 |
| | 1 | 1 | 0 |

The above example means:

| | | |
|---|---|---|
| Source 1: | 01C9H | = 0000 0001 1100 1001 |
| Source 2: | 0171H | = 0000 0001 0111 0001 |
| Destination | | = 0000 0001 0100 0001 = 00B8H |

# NOT - Logical NOT




Instruction Type: output

The NOT instruction is used to invert the source bit-by-bit (one's complement) and then place the result in the destination.

| Source | 0 | 1 |
|---|---|---|
| | 1 | 0 |

The above example means:

| Source 1:2555 | =00FFH | = 0000 0000 1111 1111 |
|---|---|---|
| Destination | | = 1111 1111 0000 0000 = -256 |

# Move Instructions

## Introduction

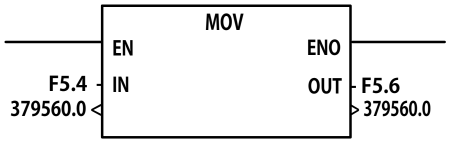
This chapter describes move Instructions of X8 Series PLC. The chapter is organized as follows.

The following table shows two move instructions.

**Application Specific Instrrcitons**

| Instruction | Abp`ofmqflk |
|---|---|
| MOV | Move<br>Move the source value to the destination |
| MOVM | Masked Move<br>Move data from a source location to a selected portion of the destination. |

# MOV - Move




Instruction Type: output

The MOV instruction is used to move data from the source to the destination. If source data and destination data size is not equal, the source is converted to the destination size when the instruction executes.

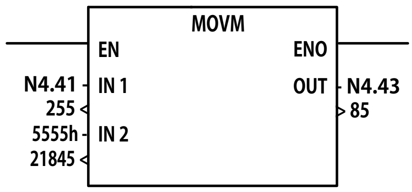
| With this Bit | | The Controller: |
|---|---|---|
| SR2.0.0 | Carry Flag | always resets |
| SR2.0.1 | Overflow Flag | sets when an overflow, infinity, or NAN (not a number) condition is detected, otherwise resets |
| SR2.0.2 | Zero Flag | sets if result is zero, otherwise resets |
| SR2.0.3 | Sign Flag | sets if result is negative (MSB is set), otherwise resets |
| SR2.5.0 | Overflow Trap | sets Math Overflow Trap minor error if the Overflow bit is set, otherwise it remains in last state |

If Sources are constants values, the data range is

- Word: -32,768 ~ 32,767
- Long Word: -2,147,483,648 ~ 2,147,483,647

| Parameter | Data Table | | | | | | | | | | | | | SFR | | | | | | | | | Address Mode | | | Address Level | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Y | X | SR | B | TM, CT, CR | N | F | ST | A | L | MG, PD | RP | PS | RTC | HSC | PTO, PWM | STI | EII | BHI | MMI | CS-Comms | DLS-DataLog | Immediate | Direct | Indirect | Bit | Word | LongWord | Element |
| Source 1 | √ | √ | √ | √ | √ | √ | √ | √ | √ | √ | √ | √ | | √ | √ | √ | √ | √ | √ | √ | | | √ | √ | √ | | √ | √ | |
| Destination | √ | √ | √ | √ | √ | √ | √ | √ | √ | √ | √ | √ | | | | | | | | | | | | √ | √ | | √ | √ | |

## MOVM - Masked Move




Instruction Type: output

The MOVM instruction is used to move data from the source to the destination, allowing portions of the destination to be masked.

The Source 1 and 2, and destination data format should be same.

| | | |
|---|---|---|
| Source 1: | OOFFH | = 0000 0000 1111 1111 |
| MASK: | 5555H | = 0101 0101 0101 0101 |
| Destination | | = 0000 0000 0101 0101 = 0055H = 85 |

| With this Bit | | The Controller: |
|---|---|---|
| SR2.0.0 | Carry Flag | always resets |
| SR2.0.1 | Overflow Flag | always resets |
| SR2.0.2 | Zero Flag | sets if result it zero, otherwise resets |
| SR2.0.3 | Sign Flag | sets if result is negative, otherwise resets |

The data range is

- Word: -32,768 ~ 32,767
- Long Word: -2,147,483,648 ~ 2,147,483,647

| Parameter | Data Table | | | | | | | | | | | | | SFR | | | | | | | | | Address Mode | | | Address Level | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Y | X | SR | B | TM, CT, CR | N | F | ST | A | L | MG, PD | RP | PS | RTC | HSC | PTO, PWM | STI | EII | BHI | MMI | CS-Comms | DLS-DataLog | Immediate | Direct | Indirect | Bit | Word | LongWord | Element |
| Source 1 | √ | √ | | √ | √ | √ | | √ | √ | √ | | √ | | | | | | | | | | | | √ | √ | | √ | √ | |
| Mask | √ | √ | | √ | √ | √ | | √ | √ | √ | | √ | | | | | | | | | | | √ | √ | √ | | √ | √ | |
| Destination | √ | √ | | √ | √ | √ | √ | √ | √ | √ | | √ | | | | | | | | | | | | √ | √ | | √ | √ | |

# Data Table Instructions

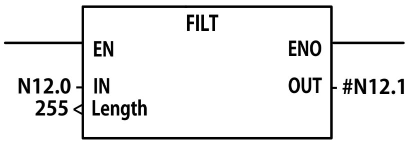
## Introduction

This chapter describes data table Instructions of X8 Series PLC. The chapter is organized as follows

The following table shows data table instructions.

| Instructions | Description |
|---|---|
| FILT | Fill Table<br>Move Source data to Destination data table. |
| COPT | Copy Table<br>After mask Source data , copy to Destination data table. |
| SWAP | Swap Word |
| COPW | Copy Word |

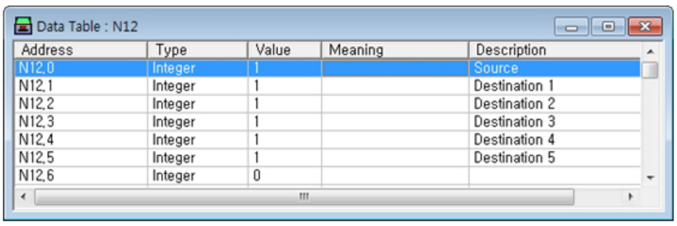
# FILT - Fill Table




Instruction Type: output

The FILT instruction fills Source 1 to Destination data table for a given length.

The above example shows that the instruction fills the N12.0 date with five source values from N12.1.


Data Table : N12

| Address | Type | Value | Meaning | Description |
|---|---|---|---|---|
| N12.0 | Integer | 1 | | Source |
| N12.1 | Integer | 1 | | Destination 1 |
| N12.2 | Integer | 1 | | Destination 2 |
| N12.3 | Integer | 1 | | Destination 3 |
| N12.4 | Integer | 1 | | Destination 4 |
| N12.5 | Integer | 1 | | Destination 5 |
| N12.6 | Integer | 0 | | |

The data range for the source is:

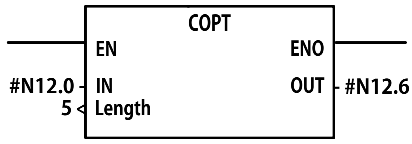
- Word: -32,768 ~ 32,767
- LONG WORD : -2,147,483,648 ~ 2,147,483,647
- IEEE-754 32-bit

The data length can range from:

- Word: 1 ~ 128
- LONG WORD : 1 ~ 64
- 3 Words: 1 ~42

| Parameter | Data Table | | | | | | | | | | | | | SFR | | | | | | | | | | Address Mode | | | Address Level | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Y | X | SR | B | TM, CT, CR | N | F | ST | A | L | MG, PD | RP | PS | RTC | HSC | PTO, PWM | STI | EII | BHI | MMI | CS-Comms | IOS-I/O | DLS-DataLog | Immediate | Direct | Indirect | Bit | Word | LongWord | Element |
| Source 1 | √ | √ | | √ | √ | √ | √ | √ | √ | √ | | √ | | | | | | | | | | | | √ | √ | √ | | √ | √ | √ |
| Length | | | | | | | | | | | | | | | | | | | | | | | | √ | | | | | | |
| Destination | √ | √ | | √ | √ | √ | √ | √ | √ | √ | | √ | | | | | | | | | | | | | √ | √ | | | | √ |

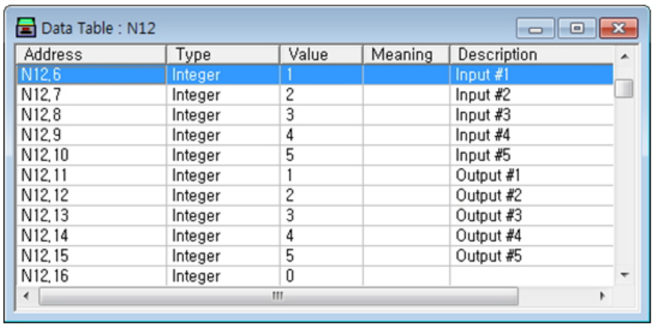
## COPT - Copy Table




Instruction Type: output

The COPT instruction copies Source data to Destination data table for a given length.

The above example shows that the COPT instruction copies N12.0 ~ N12.4 to N12.11~N12.15.



Data Table : N12

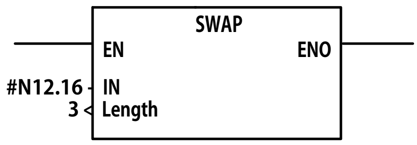
| Address | Type | Value | Meaning | Description |
|---|---|---|---|---|
| N12.6 | Integer | 1 | | Input #1 |
| N12.7 | Integer | 2 | | Input #2 |
| N12.8 | Integer | 3 | | Input #3 |
| N12.9 | Integer | 4 | | Input #4 |
| N12.10 | Integer | 5 | | Input #5 |
| N12.11 | Integer | 1 | | Output #1 |
| N12.12 | Integer | 2 | | Output #2 |
| N12.13 | Integer | 3 | | Output #3 |
| N12.14 | Integer | 4 | | Output #4 |
| N12.15 | Integer | 5 | | Output #5 |
| N12.16 | Integer | 0 | | |

The length limit is:

- Word: 1 ~ 128
- LONG WORD : 1 ~ 64
- 3 Word : 1 ~42
- String: 1~3

| Parameter | Data Table | | | | | | | | | | | | | SFR | | | | | | | | | | Address Mode | | | Address Level | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Y | X | SR | B | TM, CT, CR | N | F | ST | A | L | MG, PD | RP | PS | RTC | HSC | PTO, PWM | STI | EII | BHI | MMI | CS-Comms | IOS-I/O | DLS-DataLog | Immediate | Direct | Indirect | Bit | Word | LongWord | Element |
| Source 1 | √ | √ | | √ | √ | √ | √ | √ | √ | √ | | √ | | | | | | | | | | | | | √ | √ | | | | √ |
| Length | | | | | | | | | | | | | | | | | | | | | | | | √ | | | | | | |
| Destination | √ | √ | | √ | √ | √ | √ | √ | √ | √ | | √ | | | | | | | | | | | | | √ | √ | | | | √ |

# SWAP - Swap Word

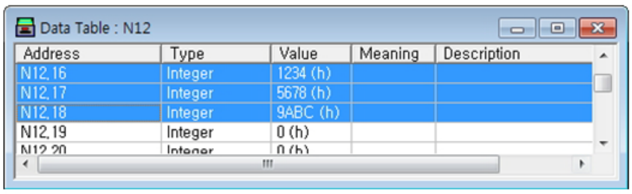



Instruction Type: output

The SWAP instruction swaps the low and high bytes of a specified number of words for a given length.

You can check the result of above example to the figures below.

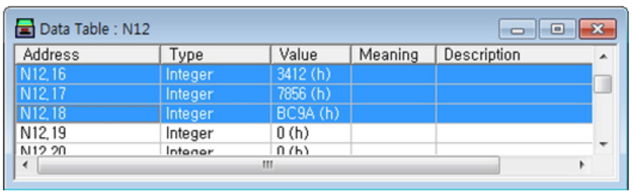
<Before>



Data Table : N12

| Address | Type | Value | Meaning | Description |
|---|---|---|---|---|
| N12.16 | Integer | 1234 (h) | | |
| N12.17 | Integer | 5678 (h) | | |
| N12.18 | Integer | 9ABC (h) | | |
| N12.19 | Integer | 0 (h) | | |
| N12.20 | Integer | 0 (h) | | |

<After>

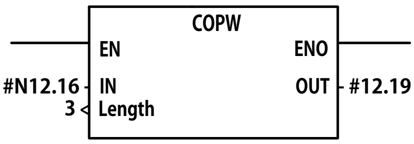


Data Table : N12

| Address | Type | Value | Meaning | Description |
|---|---|---|---|---|
| N12.16 | Integer | 3412 (h) | | |
| N12.17 | Integer | 7856 (h) | | |
| N12.18 | Integer | BC9A (h) | | |
| N12.19 | Integer | 0 (h) | | |
| N12.20 | Integer | 0 (h) | | |

| Parameter | Data Table | | | | | | | | | | | | | SFR | | | | | | | | | | Address Mode | | | Address Level | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Y | X | SR | B | TM, CT, CR | N | F | ST | A | L | MG, PD | RP | PS | RTC | HSC | PTO, PWM | STI | EII | BHI | MMI | CS-Comms | IOS-I/O | DLS-DataLog | Immediate | Direct | Indirect | Bit | Word | LongWord | Element |
| Source 1 | | | | √ | | √ | | √ | √ | | | √ | | | | | | | | | | | | | √ | | | √ | | |
| Destination | | | | | | | | | | | | | | | | | | | | | | | | √ | | | | √ | | |

# COPW - Copy Word




Instruction Type: output

The COPW instruction copies Source data to Destination for a given length.

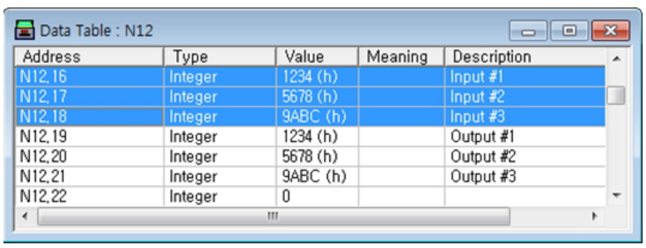
Although similar to the COPT instruction, the COPW instruction allows different source and destination parameters.

- NTEGER to LONG WORD
- LONG WORD to FLOATING POINT
- LONG WORD to INTEGER
- INTEGER to SFR PTO data table

The length of data cannot exceed 128 words.

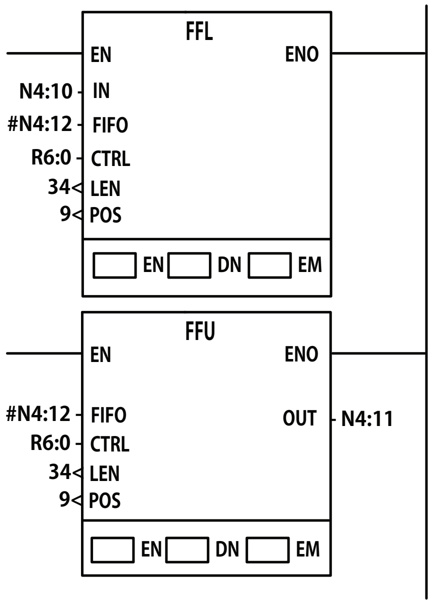
If the data exceed data table size or min. or max value of each data table during execution, the fault will be generated.

You can check the result of the above example to the figure below.




| Address | Type | Value | Meaning | Description |
|---|---|---|---|---|
| N12.16 | Integer | 1234 (h) | | Input #1 |
| N12.17 | Integer | 5678 (h) | | Input #2 |
| N12.18 | Integer | 9ABC (h) | | Input #3 |
| N12.19 | Integer | 1234 (h) | | Output #1 |
| N12.20 | Integer | 5678 (h) | | Output #2 |
| N12.21 | Integer | 9ABC (h) | | Output #3 |
| N12.22 | Integer | 0 | | |

| Parameter | Data Table | | | | | | | | | | | | | SFR | | | | | | | | | | Address Mode | | | Address Level | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Y | X | SR | B | TM, CT, CR | N | F | ST | A | L | MG, PD | RP | PS | RTC | HSC | PTO, PWM | STI | EII | BHI | MMI | CS-Comms | IOS-I/O | DLS-DataLog | Immediate | Direct | Indirect | Bit | Word | LongWord | Element |
| Source 1 | √ | √ | | √ | | √ | | √ | √ | √ | | √ | √ | √ | √ | √ | √ | √ | √ | √ | | | | | √ | √ | | | | √ |
| Length | | | | | | | | | | | | | | | | | | | | | | | | √ | | | | | | |
| Destination | √ | √ | | √ | | √ | √ | √ | √ | √ | | √ | √ | √ | √ | √ | √ | √ | | | | | | | √ | √ | | | | √ |




FFL and FFU Instruction Pair

# Shift and FIFO, LIFO Instructions

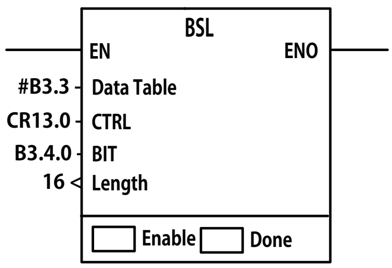
## Introduction

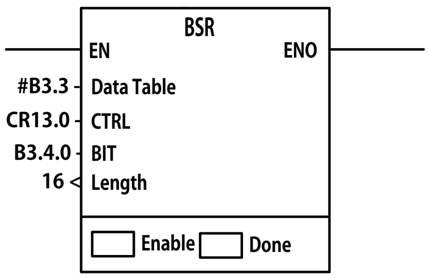
This chapter describes Shift and FIFO, LIFO instructions of the X8 Series PLC.The chapter is organized as follows.

The following table shows bit shift instructions of internal data.

| Instruction | Description |
|---|---|
| BSL | Bit Shift Left<br>Load the spbcified BIT to LSB and shift left a bit array one bit at a time when the rung status is true. |
| BSR | Bit Shift Right<br>Load the spbcified BIT to MSB and shift right a bit array one bit at a time when the rung state is true. |
| FIFOL | FIFO Queue Load<br>Load the specified data to Queue when the rung state is true (PUSH). |
| FIFOU | FIFO Queue Unload<br>Unload the data from the data table specified as Queue when the rung state is true.(POP) |
| LOFOL | LIFO Queue Load<br>Like Stack storage, load the specified data to the data table specified as Stack when the rung state is true.(PUSH) |
| LIFOU | LIFO Queue Unload<br>Like Stack storage, unload the data from the specified stack when the rung state is true.(POP) |

## BSL - Bit Shift Left
## BSR - Bit Shift Righter




Instruction Type: Output

When the rung state is true,

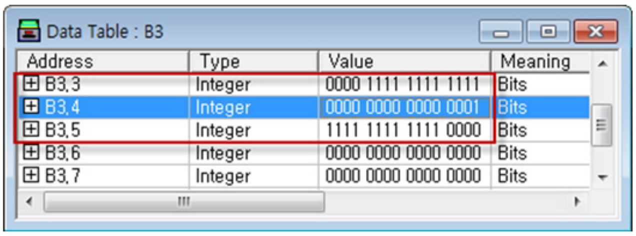
- BSL : Load the specified BIT to LSB and shift left a bit array one bit at a time when the rung status is true.
- BSR : Load the sepcified BIT to MSB and shift right a bit array one bit at a time when the rung state is true.

This instruction uses the following data format.

- Data Table : data table to Shift
- CTRL : CR data table. When the BSL instruction is in run state, this instruction is used to control the internal flags.
- BIT : data bit contains data to load
- LENGTH : bit length of data table

You can check the result of above example to the figures below.

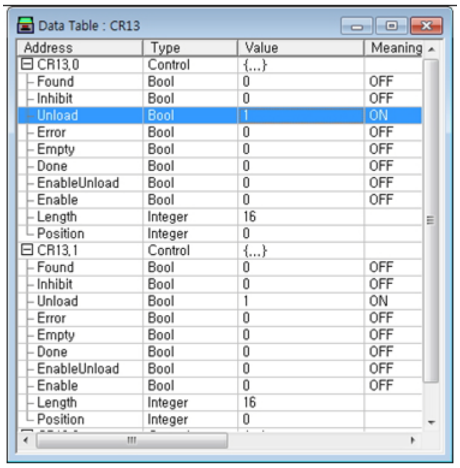
To bit shift, specify B3.3 as data table, and specify bit data as B3.4.0 to load. Then, the B3.3 Bit shift left, and the 3.5 Bit shift right.



Data Table : B3

| Address | Type | Value | Meaning |
|---|---|---|---|
| B3.3 | Integer | 0000 1111 1111 1111 | Bits |
| B3.4 | Integer | 0000 0000 0000 0001 | Bits |
| B3.5 | Integer | 1111 1111 1111 0000 | Bits |
| B3.6 | Integer | 0000 0000 0000 0000 | Bits |
| B3.7 | Integer | 0000 0000 0000 0000 | Bits |

If the bit shift by the BSL or BSR instruction, some values are unloaded from the bit array. You can check the unloaded bit on the Unload flag of the CR data table.



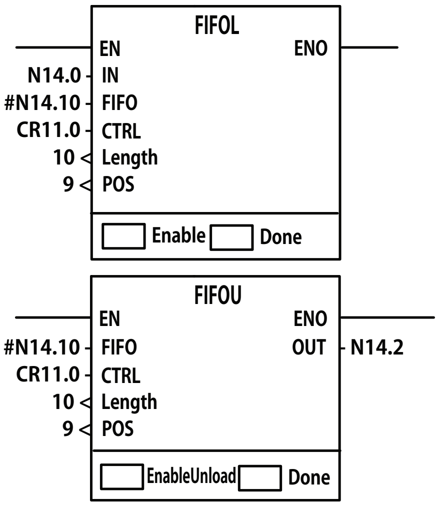
Data Table : CR13

| Address | Type | Value | Meaning |
|---|---|---|---|
| CR13.0 | Control | {...} | |
| Found | Bool | 0 | OFF |
| Inhibit | Bool | 0 | OFF |
| Unload | Bool | 1 | ON |
| Error | Bool | 0 | OFF |
| Empty | Bool | 0 | OFF |
| Done | Bool | 0 | OFF |
| EnableUnload | Bool | 0 | OFF |
| Enable | Bool | 0 | OFF |
| Length | Integer | 16 | |
| Position | Integer | 0 | |
| CR13.1 | Control | {...} | |
| Found | Bool | 0 | OFF |
| Inhibit | Bool | 0 | OFF |
| Unload | Bool | 1 | ON |
| Error | Bool | 0 | OFF |
| Empty | Bool | 0 | OFF |
| Done | Bool | 0 | OFF |
| EnableUnload | Bool | 0 | OFF |
| Enable | Bool | 0 | OFF |
| Length | Integer | 16 | |
| Position | Integer | 0 | |

| Parameter | Data Table | | | | | | | | | | | | | SFR | | | | | | | | | | Address Mode | | | Address Level | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Y | X | SR | B | TM, CT, CR | N | F | ST | A | L | MG, PD | RP | PS | RTC | HSC | PTO, PWM | STI | EII | BHI | MMI | CS-Comms | IOS-I/O | DLS-DataLog | Immediate | Direct | Indirect | Bit | Word | LongWord | Element |
| Data Table | √ | √ | | √ | √ | √ | | | | √ | | √ | | | | | | | | | | | | | √ | √ | | √ | √ | |

| Parameter | Data Table | | | | | | | | | | | | | SFR | | | | | | | | | | Address Mode | | | Address Level | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| CTRL | | | | | | | | | | | | | | | | | | | | | | | | | √ | | | | | √ |
| LENGTH | | | | | | | | | | | | | | | | | | | | | | | | √ | | | | √ | | |
| BIT | √ | √ | | √ | √ | √ | | | | √ | | | | | | | | | | | | | | | √ | √ | √ | | | |

# FIFOL - FIFO QUEUE LOAD FIFOU - FIFO QUEUE UNLOAD




Instruction Type: output

When the rung state is true,

- FIFOL : load the specified data to data table specified as Queue.(Push)
- FIFOU : Unload the data from the specified Queue.(POP)

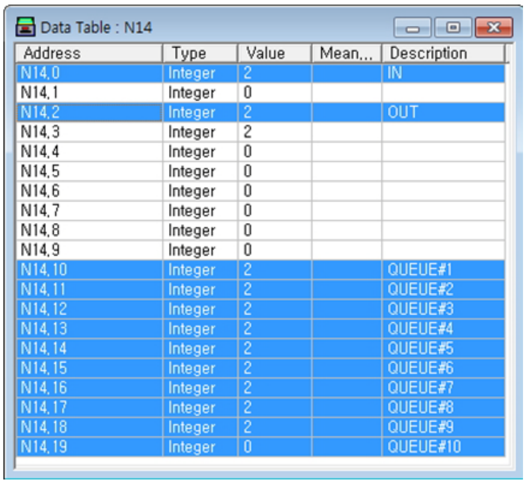
The FIFOL instruction can use following data type.

- IN : Specify the data to load to Queue
- FIFO : data table specified as QUEUE
- CTRL : CR data table. If the FIFOL and FIFOU instructions are in run state, they are used to control the internal flag.
- LENGTH : data table size specified as QUEUE
- POS : data location that is loaded first (Pointer)

The FIFOU instruction use the following data type

- FIFO : data table specified as QUEUE
- CTRL : CR data table. It the BSL instruction is in run state, and used to control the internal flogs.
- LENGTH : bit length of the data table
- POS : data location that is stored first when extracting data (Pointer)
- OUT : data table to store after unload the data from the QUEUE

You can check the result of the above example to the figure below.



Data Table : N14

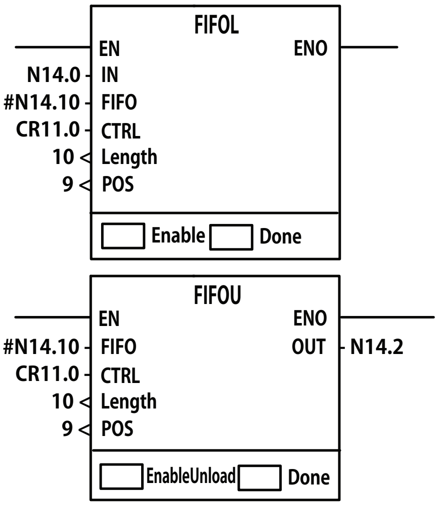
| Address | Type | Value | Mean... | Description |
|---|---|---|---|---|
| N14.0 | Integer | 2 | | IN |
| N14.1 | Integer | 0 | | |
| N14.2 | Integer | 2 | | OUT |
| N14.3 | Integer | 2 | | |
| N14.4 | Integer | 0 | | |
| N14.5 | Integer | 0 | | |
| N14.6 | Integer | 0 | | |
| N14.7 | Integer | 0 | | |
| N14.8 | Integer | 0 | | |
| N14.9 | Integer | 0 | | |
| N14.10 | Integer | 2 | | QUEUE#1 |
| N14.11 | Integer | 2 | | QUEUE#2 |
| N14.12 | Integer | 2 | | QUEUE#3 |
| N14.13 | Integer | 2 | | QUEUE#4 |
| N14.14 | Integer | 2 | | QUEUE#5 |
| N14.15 | Integer | 2 | | QUEUE#6 |
| N14.16 | Integer | 2 | | QUEUE#7 |
| N14.17 | Integer | 2 | | QUEUE#8 |
| N14.18 | Integer | 2 | | QUEUE#9 |
| N14.19 | Integer | 0 | | QUEUE#10 |

When the N14.0 is specified as IN , N14.10 ~ N14.19 are as QUEUE area, and N14.2 is OUT, if the FIFOL instruction is in run state , the N14.0 is stored to the POS in the QUEUE.

If the FIFOU instruction is in run state, unload data from the POS in the QUEUE and stores the data to N4.12.

| Parameter | Data Table | | | | | | | | | | | | | SFR | | | | | | | | | | Address Mode | | | Address Level | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Y | X | SR | B | TM, CT, CR | N | F | ST | A | L | MG, PD | RP | PS | RTC | HSC | PTO, PWM | STI | EII | BHI | MMI | CS-Comms | IOS-I/O | DLS-DataLog | Immediate | Direct | Indirect | Bit | Word | LongWord | Element |
| IN | √ | √ | | √ | √ | √ | | | | √ | | √ | | | | | | | | | | | | √ | √ | √ | | √ | √ | |
| FIFO | √ | √ | | √ | | √ | | | | √ | | √ | | | | | | | | | | | | | √ | √ | | √ | √ | √ |
| CTRL | | | | | | | | | | | | | | | | | | | | | | | | | √ | | | √ | | √ |
| LENGTH | | | | | | | | | | | | | | | | | | | | | | | | √ | | | | √ | | |
| POS | | | | | | | | | | | | | | | | | | | | | | | | √ | | | | √ | | |
| OUT | √ | √ | | √ | √ | | | | | √ | | √ | | | | | | | | | | | | | √ | √ | | √ | √ | |

# LIFOL - LIFO QUEUE LOAD
# LIFOU - LIFO QUEUE UNLOAD




Instruction type: output

If the rung state is true,

- LIFOL: load the specified data to the data table specified as Stack.(Push)
- LIFOU: unload the data from the specified Stack.(POP)

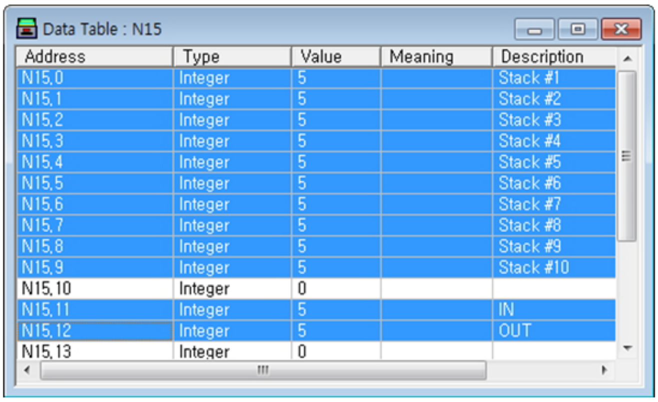
The LIFOL instruction uses the following data types:

- IN: specify the data to load to Stack
- LIFO : data table specified as Stack
- CTRL : CR data table. it the FIFOL and FIFOU instructions are in run state, are used to control the internal flag.
- LENGTH : data table length specified as Stack
- POS : Pointer location to load data (Stack Pointer)

The LIFOU instruction uses the following data types:

- OUT : data output location unloaded from Stack
- LIFO :data table specified as Stack
- CTRL : CR data table. if the FIFOL and FIFOU instructions are in run state, are used to control the internal flags.
- LENGTH : data table length specified as Stack
- POS : pointer location to unload data (Stack Pointer)

You can check the result of above example to the figures below.




| Address | Type | Value | Meaning | Description |
|---|---|---|---|---|
| N15.0 | Integer | 5 | | Stack #1 |
| N15.1 | Integer | 5 | | Stack #2 |
| N15.2 | Integer | 5 | | Stack #3 |
| N15.3 | Integer | 5 | | Stack #4 |
| N15.4 | Integer | 5 | | Stack #5 |
| N15.5 | Integer | 5 | | Stack #6 |
| N15.6 | Integer | 5 | | Stack #7 |
| N15.7 | Integer | 5 | | Stack #8 |
| N15.8 | Integer | 5 | | Stack #9 |
| N15.9 | Integer | 5 | | Stack #10 |
| N15.10 | Integer | 0 | | |
| N15.11 | Integer | 5 | | IN |
| N15.12 | Integer | 5 | | OUT |
| N15.13 | Integer | 0 | | |

When the N15.11 is specified to IN , N15.0 ~ N15.9 are Stack area, and N15.12 is OUT, if the LIFOL instruction is in run state, N15.11 is stored to the POS in the Stack area.

If the LIFOU instruction is in run state, unload the data from the POS in the Stack area, and then stores the data to N15.12.

| **Parameter** | **Data Table** | | | | | | | | | | | | | **SFR** | | | | | | | | | | **Address Mode** | | | **Address Level** | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Y | X | SR | B | TM, CT, CR | N | F | ST | A | L | MG, PD | RP | PS | RTC | HSC | PTO, PWM | STI | EII | BHI | MMI | CS-Comms | IOS-I/O | DLS-DataLog | Immediate | Direct | Indirect | Bit | Word | LongWord | Element |
| IN | √ | √ | | √ | √ | √ | | | | √ | | √ | | | | | | | | | | | | √ | √ | √ | | √ | √ | |
| FIFO | √ | √ | | √ | | √ | | | | √ | | √ | | | | | | | | | | | | | √ | √ | | √ | √ | |
| CTRL | | | | | | | | | | | | | | | | | | | | | | | | | √ | | | | | √ |
| LENGTH | | | | | | | | | | | | | | | | | | | | | | | | √ | | | | √ | | |
| POS | | | | | | | | | | | | | | | | | | | | | | | | √ | | | | √ | | |
| OUT | √ | √ | | √ | √ | √ | | | | √ | | √ | | | | | | | | | | | | | √ | √ | | √ | √ | |

# Sequencing Instructions

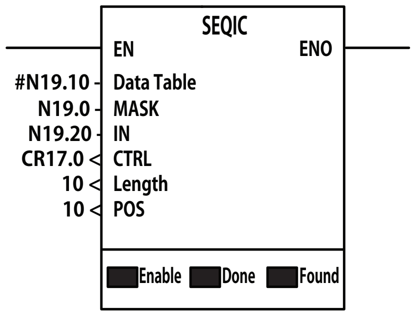
## Introduction

This chapter describes sequencer instructions of the X8 Series PLC.The chapter is organized as follows.

The following table shows Sequencer instructions of internal data.

| Instruction | Description |
|---|---|
| SEQIC | Sequencer Input Compare<br>If rung state is true, compare IN data and masked source at the data table. If the data are equal, it sets Found flag in the CR data table specified as CTRL. |
| SEQO | Sequencer Output<br>If rung state is true, data table value in the POS and masked value are logically ANDed, then, copy the data to the destination data table. If copy is done, Done flag in the data table specified CTRL is set. |
| SEQL | Sequencer Load<br>If rung state is true, load data table values specified IN to the data table specified by POS sequentially. If copy is done for a givek length, it sets DONE flag in the CR data table specified as CTRL. |

## SEQIC - Sequencer Input Compare



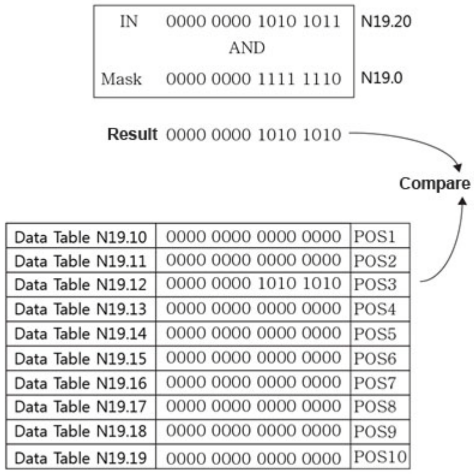

Instruction Type: output

If rung state is true, the SEQIC instruction is used to compare IN data and masked source at the data table. If the data is matched, it sets Found flag in the CR data table specified as CTRL.

When the FOUND flag is set, the FOUND flag indicator turned green on the ladder program.

**IMPORTANT** Data table and Masked source data type must be same.That is, if the data table is an integer(N), Mask and IN must be integer(N).

The following figure explains how the SEQIC instruction works.

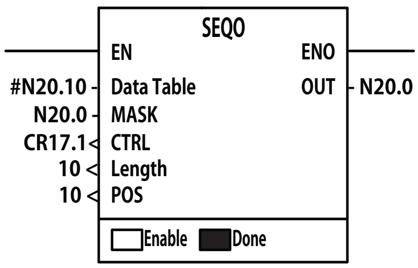



After the Source data in N19.20 and masked data in N19.0 are logically Anded, then compare the result values with the data in the N19.10 ~ N19.19 data table. If the rung state goes true, the POS value increments as +1 automatically to compare with the data in the N19.10 ~ N19.19 data table.

On above figure, same data is in the N19.12. When the instruction detects them, the FOUND flag in the CR data table will be set.

| Parameter | Data Table | | | | | | | | | | | | | SFR | | | | | | | | | Address Mode | | | Address Level | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Y | X | SR | B | TM, CT, CR | N | F | ST | A | L | MG, PD | RP | PS | RTC | HSC | PTO, PWM | STI | EII | BHI | MMI | CS-Comms | DLS-DataLog | Immediate | Direct | Indirect | Bit | Word | LongWord | Element |
| Data Table | √ | √ | | √ | | √ | | | | √ | | √ | | | | | | | | | | | | √ | √ | | √ | √ | |
| MASK | √ | √ | | √ | √ | √ | | | | √ | | √ | | | | | | | | | | | √ | √ | √ | | √ | √ | |
| IN | √ | √ | | √ | √ | √ | | | | √ | | √ | | | | | | | | | | | | √ | √ | | √ | √ | |
| CTRL | | | | | | | | | | | | | | | | | | | | | | | | √ | | | √ | | √ |
| LENGTH | | | | | | | | | | | | | | | | | | | | | | | √ | | | | √ | | |
| POS | | | | | | | | | | | | | | | | | | | | | | | √ | | √ | | √ | | |

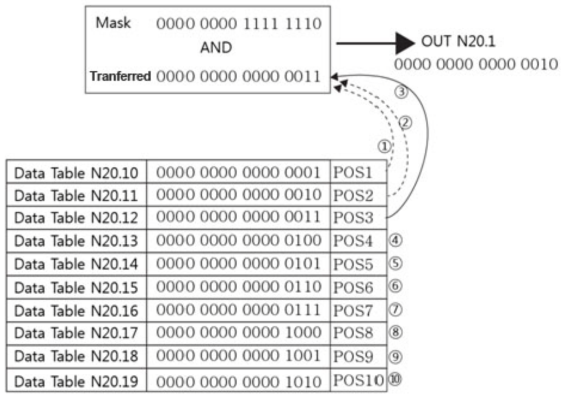
## SEQO - Sequencer Output



Instruction Type: output

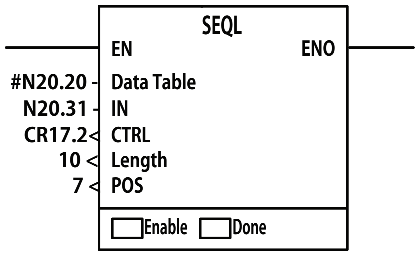
If the rung state is true, the data table value of POS and masked value are logically ANDed, then copies the value to destination data table. When the copy is done for a given length, the DONE flag in the CR data table specified CTRL is set.

**IMPORTANT** Data table and Masked source data type must be same.That is, if the data table is an integer (N), Mask and IN must be integer (N).




| Parameter | Data Table | | | | | | | | | | | | | SFR | | | | | | | | | Address Mode | | | Address Level | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Y | X | SR | B | TM, CT, CR | N | F | ST | A | L | MG, PD | RP | PS | RTC | HSC | PTO, PWM | STI | EII | BHI | MMI | CS-Comms | DLS-DataLog | Immediate | Direct | Indirect | Bit | Word | LongWord | Element |
| Data Table | √ | √ | | √ | | √ | | | | √ | | √ | | | | | | | | | | | | √ | √ | | √ | √ | |
| MASK | √ | √ | | √ | √ | √ | | | | √ | | √ | | | | | | | | | | | √ | √ | √ | | √ | √ | |
| CTRL | | | | | | | | | | | | | | | | | | | | | | | | √ | | | √ | | √ |
| LENGTH | | | | | | | | | | | | | | | | | | | | | | | √ | | | | √ | | |
| POS | | | | | | | | | | | | | | | | | | | | | | | √ | | | | √ | | |
| OUT | √ | √ | | √ | √ | √ | | | | √ | | √ | | | | | | | | | | | | √ | √ | | | √ | √ |

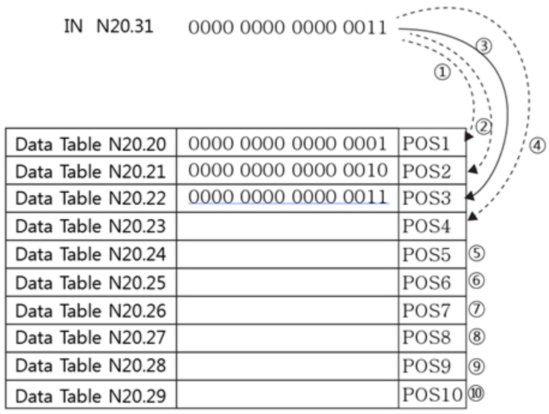
# SEQL - Sequencer Load




Instruction Type: output

If the rung state is true, the SEQL instruction copies the data table values specified IN to the data table specified by POS sequentially. When copy is done for a given length, the DONE flag in the CR data table specified CTRL is set.

| IMPORTANT | Data table and Masked source data type must be same.That is, if the data table is an integer (N), Mask and IN must be integer (N). |
|---|---|




| Parameter | Data Table | | | | | | | | | | | | | SFR | | | | | | | | | Address Mode | | | Address Level | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Y | X | SR | B | TM, CT, CR | N | F | ST | A | L | MG, PD | RP | PS | RTC | HSC | PTO, PWM | STI | EII | BHI | MMI | CS-Comms | DLS-DataLog | Immediate | Direct | Indirect | Bit | Word | LongWord | Element |
| Data Table | √ | √ | | √ | | √ | | | | √ | | √ | | | | | | | | | | | | √ | √ | | √ | √ | |
| IN | √ | √ | | √ | | √ | | | | √ | | √ | | | | | | | | | | | √ | √ | √ | | √ | √ | |
| CTRL | | | | | | | | | | | | | | | | | | | | | | | | √ | | | | | √ |
| LENGTH | | | | | | | | | | | | | | | | | | | | | | | √ | | | | √ | | |
| POS | | | | | | | | | | | | | | | | | | | | | | | √ | | | | √ | | |

# Program Control Instructions

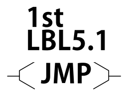
## Introduction

n this chapter, the X8 Series PLC program conφol Instructions are described.This chapter organized as follows

The following table shows the program control instructions of X8 Series PLC ladder program.

| Instructions | Descriptions |
|---|---|
| JUMP | Jump to Label<br>Jump forward/backward to a corresponding label instruction |
| LBL | Label<br>Indicate the location for JUMP instruction. |
| CALL | Call Subroutine<br>Call the subroutine. |
| SBR | Subroutine Label<br>Indicate the beginning of a subroutine and should be paired with RET |
| RET | Return from Subroutine<br>Used at the end of the subroutine and return to next instruction after call the subroutine. |
| SUSP | Suspend<br>Instruction for program debugging test and the PLC enters the IDLE state. |
| ENDT | Temporary End<br>Stop current program temporary.<br>The ENDT instruction enters run state mode during program execution, the PLC aborts ladder scan. |
| END | End of a Ladder Program<br>Indicate the end of a ladder program. The PLC display the End during ladder scan. |
| MCR | Master Control Relay<br>MCR instruction executed in pairs and contorl the intenal ladder program. |

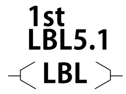
## JMP - Jump to Label




Instruction Type: output

The JMP instruction cause program execution to go to the rung marked label number. The data range for the label is from 0 to 999.

## LBL - Label

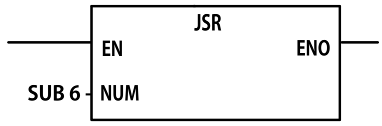



Instruction Type: output

The LBL instruction is used to indicate the location for the JMP instruction to change the order of ladder execution.

The data range for the label is from 0 to 999.

## CALL- Call Subroutine

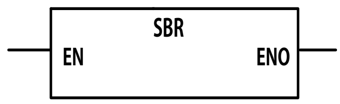



Instruction Type: output

The CALL instruction is used to call a separate subroutine. The NUM of the subroutine is the number of the subroutine on the **XGPC**.

To return from the subroutine during the ladder program execution, use the RET instruction or , use END instruction to return from subroutine after the CALL instruction is executed.

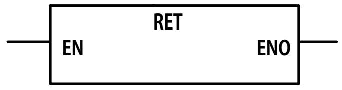
## SBR - Subroutine




Instruction Type: output

The SBR instruction contains codes that is not used in the PLC, and the first subroutine line on the **XGPC**.

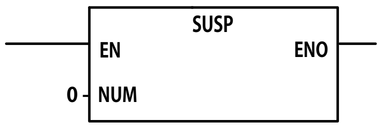
## RET - Return from Subroutine




Instruction Type: output

The RET instruction marks the end of the subroutine execution and returns from subroutine to the next instruction which is call the subroutine.

## SUSP - Suspend




Instruction Type: output

The SUSP instruction is used for program debugging and system troubleshooting. This instruction causes the processor to enter the suspend IDLE mode, causing all outputs to be de-energized.

The NUM data range is from -32768 to 32767.

## ENDT - Temporary End


Instruction Type: output

The ENDT instruction is used to end the ladder program execution temporary.

If this instruction enters RUN state mode during the ladder program execution, the PLC aborts ladder scan.

This instruction can not be used for interrupt service routine, and HSC related subroutine.

# END - End of Ladder Program

—⟨END⟩—

Instruction Type: output

The END instruction marks the end of the ladder program. And the PLC displays the end of ladder program scan.

This instruction causes return from subroutine, interrupt routine at the point where it is called.

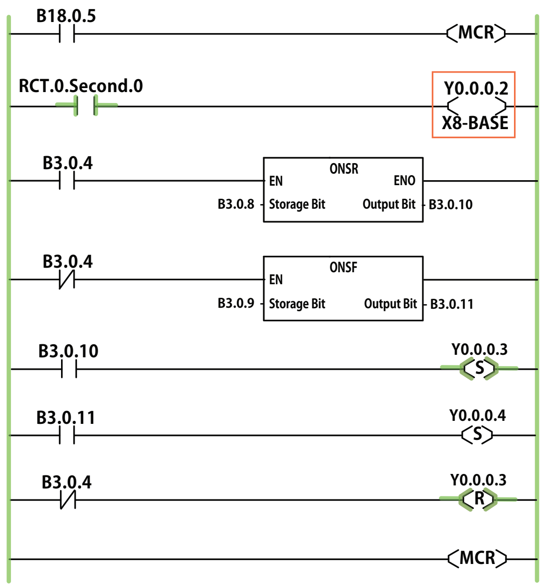
# MCR - Master Contr l l Reply

—⟨MCR⟩—

Instruction Type: output

The MCR instruction works in pairs to control internal ladder program.Rungs within the MCR zone are still scanned, but scan time is reduced due to the false state of non-retentive outputs.

The following example shows that when the start of an MCR rung , Y0.0.02 in the box is output, however when the end of the MCR rung, the Y0.0.02 is not output even the RTC.0.Second.0 state is TRUE.



**IMPORTANT**

If you start with MCR instruction, must end with the MCR instruction.

You cannot nest one MCR within another.

Do not jump into an MCR zone to control logic of the program to be working correctly.

# Input and Output Instructions

## Introduction

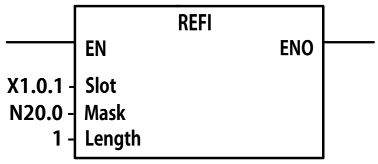
This chapter, the X8 Series PLC input and output Instructions are described.This chapter organized as follows.

The following table shows input and output instruction of X8 Series PLC.

**Application Specific Instr r ctfons**

| Instruction | Used To: |
|---|---|
| REFI | Embedded Input Refresh<br>Update input port state embedded CPU module immediately. |
| REFO | Embedded Output Refresh<br>Update output port state embedded CPU module immediately. |
| EOS | End of Scan<br>Update I/O read, write akd communications. |

## REFI - Embedded Input Refresh




Instruction Type: output

The REFI instruction updates the input port embedded in CPU module and the I/O port is updated when the ladder program scan automatically.

To update input port state immediately, use the REFI instruction.

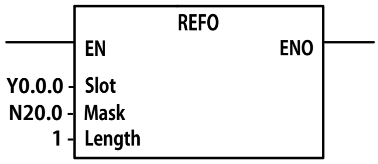
This instruction works on the input port embedded in the CPU module only.

This instruction uses the following operands:

- Slot - defines the slot location of input port. For above example, If slot = X1.0.1, word 1 of slot 0 in input is used.
- Mask - the masked value of the input data or data table contains constants or masked data.
- Length - This is the number of words to input.

| Parameter | Data Table | | | | | | | | | | | | | SFR | | | | | | | | | Address Mode | | | Address Level | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Y | X | SR | B | TM, CT, CR | N | F | ST | A | L | MG, PD | RP | PS | RTC | HSC | PTO, PWM | STI | EII | BHI | MMI | CS-Comms | DLS-DataLog | Immediate | Direct | Indirect | Bit | Word | LongWord | Element |
| SLOT | | √ | | | | | | | | | | | | | | | | | | | | | | √ | √ | | √ | | |
| MASK | √ | √ | √ | √ | √ | √ | | | | | | | | | | | | | | | | | √ | √ | √ | | √ | | |
| LENGTH | | | | | | | | | | | | | | | | | | | | | | | √ | | | | √ | | |

## REFO- Embedded Output Refresh




Instruction Type: output

The REFO instruction updates output port embedded in CPU module and the I/O port is updated when the ladder program scan automatically.

To update output port immediately, use the REFO instruction.

This instruction works on the input port embedded in the CPU module only.

This instruction uses the following operands:

- Slot - defines the slot location of output port. For above example, If slot = Y0.0.1, word 0of slot 0 in output is used.
- Mask - the masked value of the output data or data table contains constants or masked data.
- Length - This is the number of words to output.

| Parameter | Data Table | | | | | | | | | | | | | SFR | | | | | | | | | Address Mode | | | Address Level | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Y | X | SR | B | TM, CT, CR | N | F | ST | A | L | MG, PD | RP | PS | RTC | HSC | PTO, PWM | STI | EII | BHI | MMI | CS-Comms | DLS-DataLog | Immediate | Direct | Indirect | Bit | Word | LongWord | Element |
| SLOT | √ | | | | | | | | | | | √ | | | | | | | | | | | | √ | | | √ | | |
| MASK | √ | √ | | √ | √ | √ | | | | | | | | | | | | | | | | | √ | √ | √ | | √ | | |
| LENGTH | | | | | | | | | | | | | | | | | | | | | | | √ | | | | | | |

## EOS - End of Scan


Instruction Type: output

The EOS instruction updates IO read, write and communications.

The EOS instruction is not required separate the input data because it executes all of ports in the PLC at the same time. When the rung condition is in true, updated all of IO ports and communication status. The scan then resumes at the instruction following the REF instruction.
This instruction updates relatively slow IO port and cannot be executed from time effected STI, HSC routine, EII routine, or a user fault routine because the internal watchdog timer or scan timer is initialized after the EOS instruction execution.

# Using Interrupts

## Introduction

This chapter, the X8 Series PLC interrupts are described.This chapter organized as follows.

The following table shows usage of the interrupt of the X8 Series PLC.

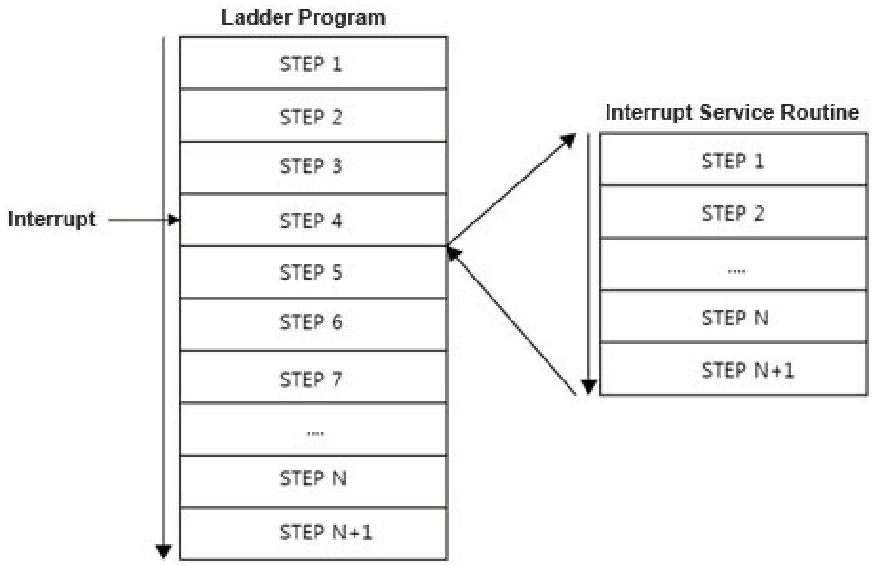
| Instruction | Description |
|---|---|
| INT | Interrupt Subroutine<br>Label to indicate the start of the interrupt. |
| PITS | PIT(Programmable Interrupt Timer) Start<br>Execute interrupt timer. |
| INTD | Interrupt Disable<br>Disable the user interrupt. |
| INTE | Interrupt Enable<br>Enable the user interrupt. |
| INTF | Interrupt Flush<br>Remove the various wafting user interrupt from the system. |

## What is an Interrupt?

The following figure shows how an interrupt works.

Not only PLC, most control devices like PC support interrupt function.

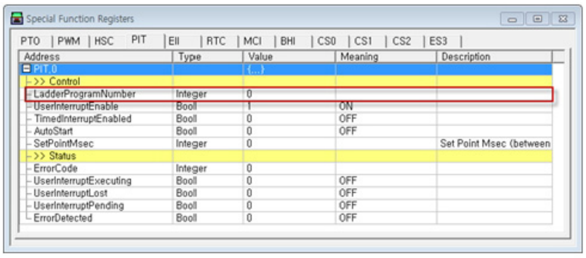
An interrupt is an event that causes the controller to suspend the task it is currently performing, perform a different task, and then return to the next task at the point where it suspended.



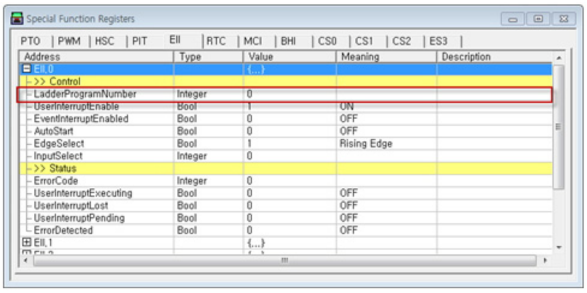

As you can see the above figure, if an interrupt occurs at one point during the ladder program execution, it suspends task it is currently performing, then performs subroutines specified by SFR or ladder program according to the occurred interrupt type.

For example, in the following figures, when an interrupt occurs in the LadderProgramNumber, if you input the subroutine number to perform, it performs following defied subroutines.

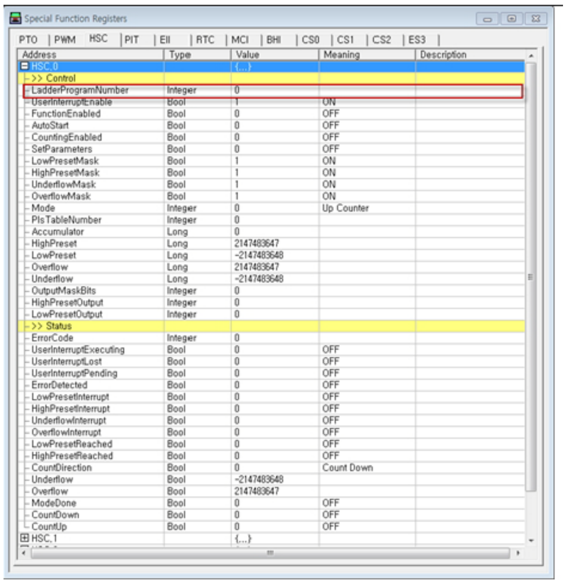
**[Example 1] PIT interrupt service routine configuration window**



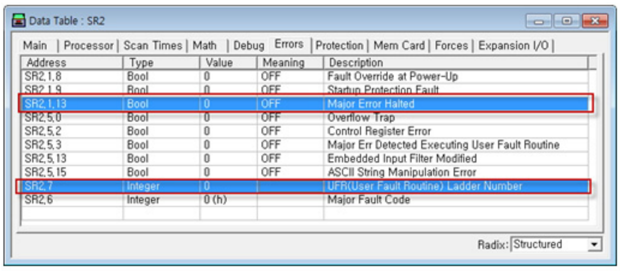

**[Example 2] EII (External Input Interrupt) interrupt service routine configuration window**




**[Example 3] HSC interrupt service routine configuration window**




**[Example 4] UFR (User Fault Routine) interrupt service routine configuration window**




* User Fault Routine (URF) configuration part is located in “Error” of SR2 different with other interrupt configuration.

But, the interrupt service routine execution can affect to PLC scan time directly, interrupt processing must be completed in the shortest time. It is applied to most devices using interrupt.

## Priority of User Interrupts

When multiple interrupts occur, the interrupts are serviced based upon their individual priority.

The priorities from highest to lowest are:

| Priority | Interrupts |
|---|---|
| 1 | UFR (User Fault Routine) Interrupt |
| 2 | External Input Interrupt-0 |
| 3 | External Input Interrupt-1 |
| 4 | HSC Interrupt -0 (HSC0) |
| 5 | External Input Interrupt-2 |
| 6 | External Input Interrupt-3 |
| 7 | HSC Interrupt-1 (HSC1) |
| 8 | External Input Interrupt-4 |
| 9 | HSC Interrupt-2 (HSC2) |
| 10 | External Input Interrupt-5 |
| 11 | HSC Interrupt-3 (HSC3) |
| 12 | Programmable Interrupt Timer |
| 13 | External Input Interrupt-6 |
| 14 | HSC Interrupt-4 (HSC4) |
| 15 | External Input Interrupt-7 |
| 16 | HSC Interrupt-5 (HSC5) |

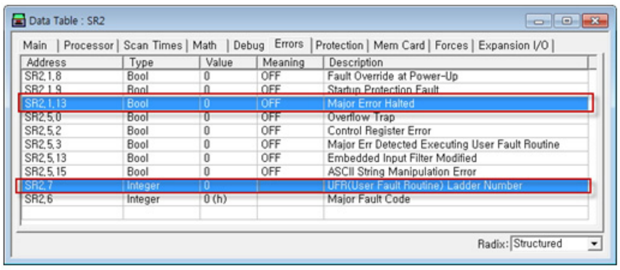
# Interrupt Type and Elements

## User Fault Routine (UFR) Interrupt

The UFR interrupt is an interrupt with the highest priority and recovers User Fault (Error). The fault routine is executed when any recoverable or non-recoverable user fault occurs.

The basic types of faults are described below:

- Recoverable Fault
  Recoverable Faults are caused by the user and may be recovered from by the UFR interrupt. To clear the errors, set the SR2.1.13 flag as reset value in the UFR service routine.

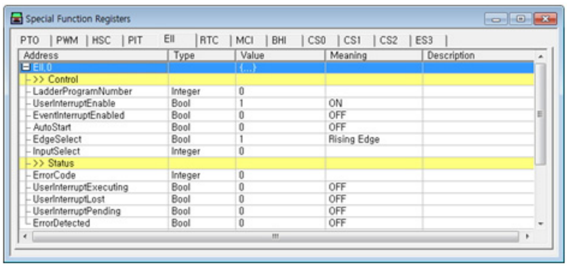



- Non-Recoverable Fault
  Non-Recoverable Faults are covered after Power Cycle (Power On/Off).

## External Input Interrupt (EII)

The EII contains 8 of interrupts from EII 0 to 7 according to the priority.

It is supported X1 input port embedded on PLC CPU only.




Followings are descriptions elements related interrupt of SFR.

It is described based on the EII and each elements are used in HSC and PIT similarly.

## LadderProgramNumber

| Description | Data Format | HSC Mode | Function | User Program Access |
|---|---|---|---|---|
| EII.0. LadderProgramNumber | Word | 6 ~ 1,535 | Control | Read/Write |

The LadderProgramNumber control flag defines EII interrupt service routine.

## UserInterruptEnabled

| Description | Data Format | HSC Mode | Function | User Program Access |
|---|---|---|---|---|
| EII.0.UserInterruptEnable | Bit | 0 or 1 | Control | Read/Write |

The UserInterruptEnabled bit is used to perform whether the LadderProgramNumber service routine allocated to EII event.

## EventInterruptEnabled

| Description | Data Format | HSC Mode | Function | User Program Access |
|---|---|---|---|---|
| EII.0.EventInterruptEnabled | Bit | 0 or 1 | Control | Read/Write |

The EventInterruptEnabled control bit is used to execute whether the EII performs.

## AutoStart

| Description | Data Format | HSC Mode | Function | User Program Access |
|---|---|---|---|---|
| EII.0.AutoStart | Bit | 0 or 1 | Control | Read/Write |

The AutoStart control bit automatically sets the interrupt when the X8 Series PLC enters any executing mode.

## EdgeSelect

| Description | Data Format | HSC Mode | Function | User Program Access |
| --- | --- | --- | --- | --- |
| EII.0.EdgeSelect | Bit | 0 or 1 | Control | Read/Write |

The EdgeSelect control bit is used to select the trigger types when using EII.

- 1: falling edge
- 0: rising edge

## InputSelect

| Description | Data Format | HSC Mode | Function | User Program Access |
| --- | --- | --- | --- | --- |
| EII.0.InputSelect | Word | 0 ~ 7 | Control | Read/Write |

The InputSelect control word is used to define which external input port signal is used.

For example, it uses 0~ 3 among X1.0.0 ~ X1.0.3 (In case of X8-M16DDR).

## ErrorCode

| Description | Data Format | HSC Mode | Function | User Program Access |
| --- | --- | --- | --- | --- |
| EII.0.ErrorCode | Word | | Control | Read/Write |

The following table shows the error codes of EII.

| Error Code | Descriptions |
|---|---|
| 1 | Invalid Program File Number<br>The program file number is not 6 ~ 1535 or not exist. |
| 2 | Invalid Input Selection<br>The input port range is 0,1,2,3,4,5,6,7. |
| 3 | Input Selection Overlap<br>Each EII uses a different input since the EII cannot share the input. |

## UserInterruptExecuting

| Description | Data Format | HSC Mode | Function | User Program Access |
|---|---|---|---|---|
| EII.0.UserInterruptExecuting | Bit | | Status | Read Only |

The UserInterruptExecuting status bit is set when the interrupt serivce routine of EII is executed.

## UserInterruptLost

| Description | Data Format | HSC Mode | Function | User Program Access |
|---|---|---|---|---|
| EII.0.UserInterruptLost | Bit | | Status | Read Only |

The UserInterruptLost bit can process 1 active and maintain up to 2 pending user interrupt conditions before it sets the lost bit.

You can check the interrupt lost by this bit.

## UserInterruptPending

| Description | Data Format | HSC Mode | Function | User Program Access |
|---|---|---|---|---|
| EII.0.UserInterruptPending | Bit | | Status | Read Only |

The UserInterruptPending is a status bit that represents an EII interrupt is pending.

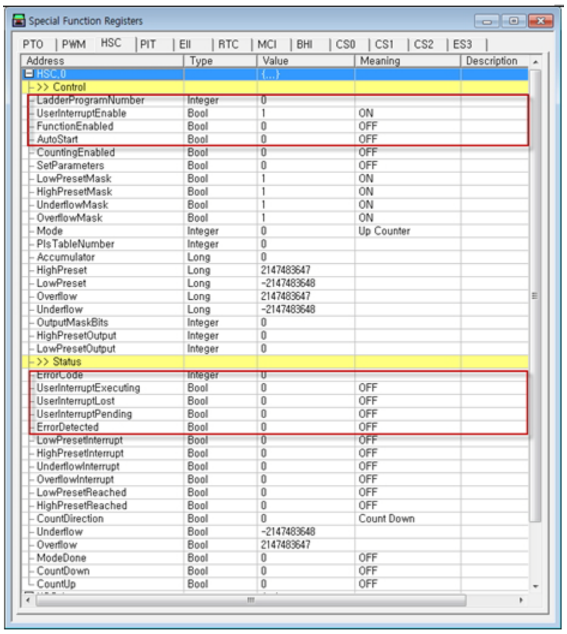
## ErrorDetected

| Description | Data Format | HSC Mode | Function | User Program Access |
|---|---|---|---|---|
| EII.0.ErrorDetected | Bit | | Status | Read Only |

The ErrorDetected flag is a status bit that can be used by the control program to detect if an error is present in the EII configuration.

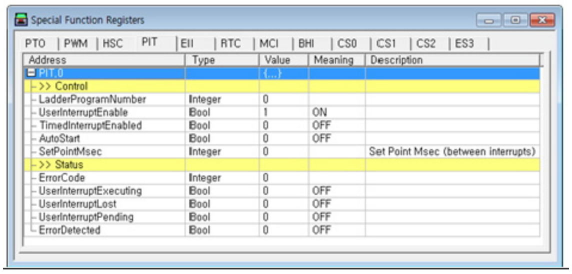
## High Speed Counter (HSC) Interrupt

The following figure shows HSC interrupts in the box among the SFR of the high-speed counter. The usage of the HSC interrupts are similar with EII.




Each elements are used to EII, HSC and PTI similary. Please refer to EII and PIT elements.

## Programmable Interrupt Timer (PIT) Interrupt



Special Function Registers

PTO | PWM | HSC | PIT | EII | RTC | MCI | BHI | CS0 | CS1 | CS2 | ES3

| Address | Type | Value | Meaning | Description |
|---|---|---|---|---|
| PIT.0 | | {...} | | |
| >> Control | | | | |
| LadderProgramNumber | Integer | 0 | | |
| UserInterruptEnable | Bool | 1 | ON | |
| TimedInterruptEnabled | Bool | 0 | OFF | |
| AutoStart | Bool | 0 | OFF | |
| SetPointMsec | Integer | 0 | | Set Point Msec (between interrupts) |
| >> Status | | | | |
| ErrorCode | Integer | 0 | | |
| UserInterruptExecuting | Bool | 0 | OFF | |
| UserInterruptLost | Bool | 0 | OFF | |
| UserInterruptPending | Bool | 0 | OFF | |
| ErrorDetected | Bool | 0 | OFF | |

Following descriptions for elements related to Interrupt of SFR.

It is described based on PIT and ach elemets are used in EII and HSC similarly.

### LadderProgramNumber

| Description | Data Format | HSC Mode | Function | User Program Access |
|---|---|---|---|---|
| PIT.0.LadderProgramNumber | Word | 6 ~ 1,535 | Control | Read / Write |

The LadderProgramNumber control flag defines PIT interrupt service routine.

### UserInterruptEnabled

| Description | Data Format | HSC Mode | Function | User Program Access |
|---|---|---|---|---|
| PIT.0.UserInterruptEnabled | Bit | 0 or 1 | Control | Read / Write |

The UserInterruptEnabled bit is used to perform whether the LadderProgramNumber service routine allocated to PIT event.

## EventInterruptEnabled

| Description | Data Format | HSC Mode | Function | User Program Access |
|---|---|---|---|---|
| PIT.0.EventInterruptEnabled | Bit | 0 or 1 | Control | Read/Write |

The EventInterruptEnabled control bit is used to execute whether the PIT function performs.

## AutoStart

| Description | Data Format | HSC Mode | Function | User Program Access |
|---|---|---|---|---|
| PIT.0.AutoStart | Bit | 0 or 1 | Control | Read/Write |

The AutoStart control bit automatically sets the intterupt when the X8 Series PLC enters any exeruting mode.

## SetPointMsec

| Description | Data Format | HSC Mode | Function | User Program Access |
|---|---|---|---|---|
| PIT.0.SetPointMsec | Word | 0 ~ 65,535 | Control | Read/Write |

The SetPointMsec control flag is set to PIT time section and control the timer interrupt interval.

## ErrorCode

| Description | Data Format | HSC Mode | Function | User Program Access |
|---|---|---|---|---|
| PIT.0.ErrorCode | Word | | Control | Read/Write |

The following table shows the error codes of PIT.

| Error Code | Descriptions |
|---|---|
| 1 | Invalid Program File Number<br>The program file number is not 6 ~ 1535 or not exist. |

## UserInterruptExecuting

| Description | Data Format | HSC Mode | Function | User Program Access |
|---|---|---|---|---|
| PIT.0.UserInterruptExecuting | Bit | | Status | Read Only |

The UserInterruptExecuting status bit is set when the interrupt service routine of PIT is in execution mode.

## UserInterruptLost

| Description | Data Format | HSC Mode | Function | User Program Access |
|---|---|---|---|---|
| PIT.0.UserInterruptLost | Bit | | Status | Read Only |

The UserInterruptLost bit can process 1 active and maintain up to 2 pending user interrupt conditions before it sets the lost bit.

You can check the interrupt lost by this bit.

## UserInterruptPending

| Description | Data Format | HSC Mode | Function | User Program Access |
|---|---|---|---|---|
| PIT.0.UserInterruptPending | Bit | | Status | Read Only |

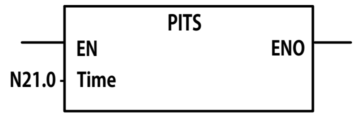
The UserInterruptPending is a status bit that represents an PIT interrupt is pending.

## ErrorDetected

| Description | Data Format | HSC Mode | Function | User Program Access |
|---|---|---|---|---|
| PIT.0.ErrorDetected | Bit | | Status | Read Only |

The ErrorDetected flag is a status bit that can be used by the control program to detect if an error is present in the PIT configuration.

# PITS - Programmable Interrupt Timer Start

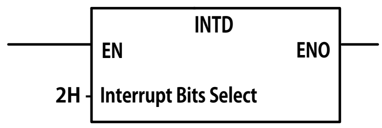



Instruction Type: output

You can set the XGPC internal configration of the Programmable Interrupt through the PTI section of the SFR.

| Parameter | Data Table | | | | | | | | | | | | | SFR | | | | | | | | | Address Mode | | | Address Level | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Y | X | SR | B | TM, CT, CR | N | F | ST | A | L | MG, PD | RP | PS | RTC | HSC | PTO, PWM | STI | EII | BHI | MMI | CS-Comms | DLS-DataLog | Immediate | Direct | Indirect | Bit | Word | LongWord | Element |
| SLOT | √ | √ | | √ | √ | √ | | | | | | √ | | | | | | | | | | | √ | √ | √ | | √ | | |

# INTD - Interrupt Disable



Instruction Type: output

The INTD instruction disables the user interrupts.

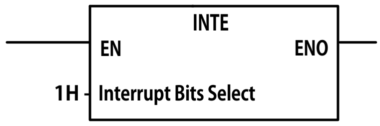
The following table shows Interrupt Bits Select value for disable each interrupts.

EII.1 interrupt is disabled in the above example.

| Interrupt | Bit | hexadecimal |
|---|---|---|
| EII.0 | 0 | 1H |
| EII.1 | 1 | 2H |
| EII.2 | 2 | 4H |
| EII.3 | 3 | 8H |
| EII.4 | 4 | 10H |
| EII.5 | 5 | 20H |
| EII.6 | 6 | 40H |
| EII.7 | 7 | 80H |
| HSC.0 | 16 | 10000H |
| HSC.1 | 17 | 20000H |
| HSC.2 | 18 | 40000H |
| HSC.3 | 19 | 80000H |
| HSC.4 | 20 | 100000H |
| HSC.5 | 21 | 200000H |
| PIT Timer | 31 | 80000000H |

If you need to control 2 or more interrupts at the same time, add above values and input them to the Interrupt Bits Select value.

## INTE - Interrupt Enable




Instruction Type: output

The INTE instruction enables the user interrupts.

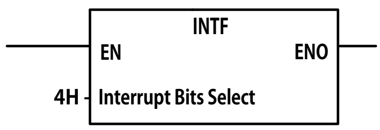
The following table shows Interrupt Bits Select value for enable each interrupts.

EII.0 interrupt is enabled in the above example.

| Interrupt | Bit | hexadecimal |
|---|---|---:|
| EII.0 | 0 | 1H |
| EII.1 | 1 | 2H |
| EII.2 | 2 | 4H |
| EII.3 | 3 | 8H |
| EII.4 | 4 | 10H |
| EII.5 | 5 | 20H |
| EII.6 | 6 | 40H |
| EII.7 | 7 | 80H |
| HSC.0 | 16 | 10000H |
| HSC.1 | 17 | 20000H |
| HSC.2 | 18 | 40000H |
| HSC.3 | 19 | 80000H |
| HSC.4 | 20 | 100000H |
| HSC.5 | 21 | 200000H |
| PIT Timer | 31 | 80000000H |

If you need to control 2 or more interrupts at the same time, add above values and input them to the Interrupt Bits Select value.

## INTF - Interrupt Flush




Instruction Type: output

The INTF instruction removes each pending user interrupts from the system.

| Interrupt | Bit | hexadecimal |
|---|---|---:|
| EII.0 | 0 | 1H |
| EII.1 | 1 | 2H |
| EII.2 | 2 | 4H |
| EII.3 | 3 | 8H |
| EII.4 | 4 | 10H |
| EII.5 | 5 | 20H |
| EII.6 | 6 | 40H |
| EII.7 | 7 | 80H |
| HSC.0 | 16 | 10000H |
| HSC.1 | 17 | 20000H |
| HSC.2 | 18 | 40000H |
| HSC.3 | 19 | 80000H |
| HSC.4 | 20 | 100000H |
| HSC.5 | 21 | 200000H |
| PIT Timer | 31 | 80000000H |

If you need to control 2 or more interrupts at the same time, add above values and input them to the Interrupt Bits Select value.

# PID Control

## Introduction

This chapter describes the X8 Series PLC Proportional Integral Derivative (PID) instructions. This chapter organized as follows.

## The PID Equation

The PID instruction can be expressed like (1) or (2) formula:

The formula adds following three actions:

- Proportional Action (P Action): calculate proportional values to current deviation e.
- Integral Action (I Action): calculate proportional values to accumulated pas deviation
- Derivative Action (D Action): calculate proportional values to the size of deviation e

$$y = K_p\left(e + \frac{1}{T_I}\int e\,dt + T_D\frac{de}{dt}\right) \quad (1)$$

$$= \underbrace{K_p \cdot e}_{\text{P Action}} + \underbrace{\frac{K_p}{T_I}\int e\,dt}_{\text{I Action}} + \underbrace{K_p \cdot T_D\frac{de}{dt}}_{\text{D Action}} \quad (2)$$

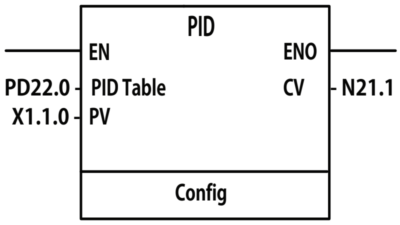
y = Detection
e = Deviation
$K_P$ = Proportional coefficient = 0.01 ~ 327.67
$T_I$ = Integral time = 327.67 ~ 0.01

$T_D$ = Derivative time = 0.01 ~ 327.67

That is, PID control is configured following three elements and the basic meaning of PID control is to find the optimal value considering past, current, and future data.

- Integral Action (I Action): consider past data
- Proportional Action (P Action): consider current data
- Derivative Action (D Action): consider future data

# PID - Proportional Integral Derivative




Instruction Type: output

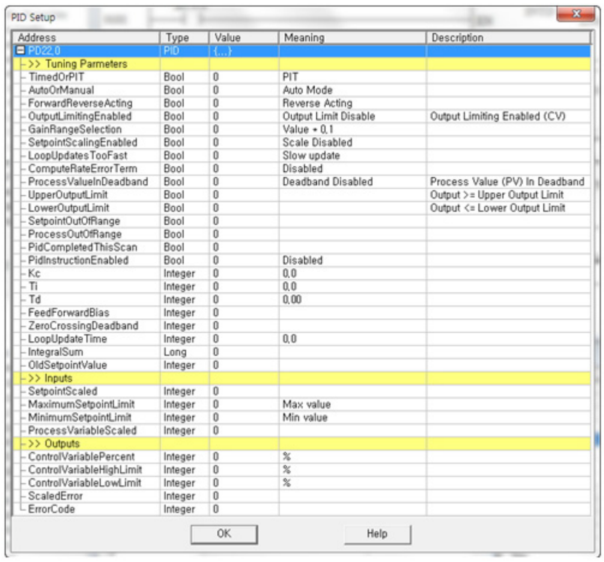
The PID instruction is an instruction to control the PID.

**IMPORTANT**

Analog module data and PID instruction parameter must be same.

It is recommended that you place the PID instruction on a rung without any conditional logic. If conditional logic exists, the Control Variable output remains at its last value, and the CVP CV% term and integral term are both cleared when the rung is false.

When you click the "Config" in the PID instruction, following screen is shown.



PID Setup

| Address | Type | Value | Meaning | Description |
|---|---|---|---|---|
| PD22.0 | PID | {...} | | |
| >> Tuning Parmeters | | | | |
| TimedOrPIT | Bool | 0 | PIT | |
| AutoOrManual | Bool | 0 | Auto Mode | |
| ForwardReverseActing | Bool | 0 | Reverse Acting | |
| OutputLimitingEnabled | Bool | 0 | Output Limit Disable | Output Limiting Enabled (CV) |
| GainRangeSelection | Bool | 0 | Value * 0,1 | |
| SetpointScalingEnabled | Bool | 0 | Scale Disabled | |
| LoopUpdatesTooFast | Bool | 0 | Slow update | |
| ComputeRateErrorTerm | Bool | 0 | Disabled | |
| ProcessValueInDeadband | Bool | 0 | Deadband Disabled | Process Value (PV) In Deadband |
| UpperOutputLimit | Bool | 0 | | Output >= Upper Output Limit |
| LowerOutputLimit | Bool | 0 | | Output <= Lower Output Limit |
| SetpointOutOfRange | Bool | 0 | | |
| ProcessOutOfRange | Bool | 0 | | |
| PidCompletedThisScan | Bool | 0 | | |
| PidInstructionEnabled | Bool | 0 | Disabled | |
| Kc | Integer | 0 | 0,0 | |
| Ti | Integer | 0 | 0,0 | |
| Td | Integer | 0 | 0,00 | |
| FeedForwardBias | Integer | 0 | | |
| ZeroCrossingDeadband | Integer | 0 | | |
| LoopUpdateTime | Integer | 0 | 0,0 | |
| IntegralSum | Long | 0 | | |
| OldSetpointValue | Integer | 0 | | |
| >> Inputs | | | | |
| SetpointScaled | Integer | 0 | | |
| MaximumSetpointLimit | Integer | 0 | Max value | |
| MinimumSetpointLimit | Integer | 0 | Min value | |
| ProcessVariableScaled | Integer | 0 | | |
| >> Outputs | | | | |
| ControlVariablePercent | Integer | 0 | % | |
| ControlVariableHighLimit | Integer | 0 | % | |
| ControlVariableLowLimit | Integer | 0 | % | |
| ScaledError | Integer | 0 | | |
| ErrorCode | Integer | 0 | | |

OK Help

The following table shows the each elements.

Total elements are described based on the data table PD22.

| Address | Data Format | Data Range | Function | User Program Access |
|---|---|---|---|---|
| PD22.0.TimedOrPit | Binary | 0 or 1 | control | Read/Write |
| PD22.0.AutoOrManual | Binary (Bit) | 0 or 1 | control | Read/Write |
| PD22.0.ForwardReverseActing | Binary (Bit) | 0 or 1 | control | Read/Write |
| PD22:0.Kc | Word (INT) | 0 ~32,767 | control | Read/Write |
| PD22:0.Ti | Word (INT) | 0 ~32,767 | control | Read/Write |
| PD22:0.Td | Word (INT) | 0 ~32,767 | control | Read/Write |
| PD22:0.LoopUpdateTime | Word (INT) | 1~1,024 | control | Read/Write |
| PD22:0.ZeroCrossingDeadband | Word (INT) | 0 ~32,767 | control | Read/Write |
| PD22:0.FeedForwardBias | Word (INT) | -16,383~+16,383 | control | Read/Write |
| PD22:0.ScaledError | Word (INT) | -32,768~+32,767 | Status | Read Only |
| PD22:0.ProcessValueInDeadband | Binary (Bit) | 0 or 1 | Status | Read/Write |
| PD22:0.GainRangeSelection | Binary (Bit) | 0 or 1 | control | Read/Write |
| PD22:0.SetpointScalingEnabled | Binary (Bit) | 0 or 1 | control | Read/Write |
| PD22:0.LoopUpdatesTooFast | Binary (Bit) | 0 or 1 | Status | Read/Write |
| PD22:0.ComputeRateErrorTerm | Binary (Bit) | 0 or 1 | control | Read/Write |
| PD22:0.UpperOutputLimit | Binary (Bit) | 0 or 1 | Status | Read/Write |
| PD22:0.LowerOutputLimit | Binary (Bit) | 0 or 1 | Status | Read/Write |
| PD22:0.SetpointOutOfRange | Binary (Bit) | 0 or 1 | Status | Read/Write |
| PD22:0.ProcessOutOfRange | Binary (Bit) | 0 or 1 | Status | Read/Write |
| PD22:0.PidCompletedThisScan | Binary (Bit) | 0 or 1 | Status | Read Only |
| PD22:0.PidInstructionEnabled | Binary (Bit) | 0 or 1 | Status | Read Only |
| PD22:0.IntegralSum | Long Word (32-bit INT) | -2,147,483,648 ~ 2,147,483,647 | Status | Read/Write |
| PD22:0.19 | Long Word (32-bit INT) | -2,147,483,648 ~ 2,147,483,647 | Status | Read Only |

| Address | Data Format | Data Range | Function | User Program Access |
|---|---|---|---|---|
| PD10:0.SetpointScaled | Word (INT) | 0...16,383 | control | Read/Write |
| PD10:0.MaximumSetpointLimit | Word (INT) | -32,768...+32,767 | control | Read/Write |
| PD10:0.MinimumSetpointLimit | Word (INT) | -32,768...+32,767 | control | Read/Write |
| PD10:0.ProcessVariableScaled | Word (INT) | 0...16,383 | control | Read/Write |
| PD10:0.OldSetpointValue | Word (INT) | -32,768...+32,767 | Status | Read Only |

| Address | Data Format | Data Range | Function | User Program Access |
|---|---|---|---|---|
| PD10:0.OutputLimitingEnabled | Binary | 1 =Enable<br>0 = Disable | control | Read/Write |
| PD10:0.ControlVariableHighLimit | Word (INT) | 1~100% | control | Read/Write |
| PD22:0.ControlVariableLowLimit | Word (INT) | 1~100% | control | Read/Write |

## TimedOrPIT

| Description | Data Format | HSC Mode | Function | User Program Access |
|---|---|---|---|---|
| PD.10.TimedOrPIT | Bit | 0 or 1 | Control | Read/Write |

The TimedOrPIT bit determines how update the CV value of the PID.

- 0 :PIT Mode. Update the CV value when PID scan in the PIT interrupt service routine. Therefore, PIT interrupt interval and LoopUpdataTime must be same.
- 1: Timed Mode. update the CV value according to interval specified by the LoopUpdataTime.

## AutoOrManual

| Description | Data Format | HSC Mode | Function | User Program Access |
|---|---|---|---|---|
| PD.10.AutoOrManual | Bit | 0 or 1 | Control | Read/Write |

The AutoOrManual bit determines how control the CV value of the PID.

- 0 :Auto Mode. The PID instruction controls the CV value.
- 1: Manual Mode. The user air program controls the CV value.

## ForwardReverseActing

| Description | Data Format | HSC Mode | Function | User Program Access |
|---|---|---|---|---|
| PD.10.ForwardReverseActing | Bit | 0 or 1 | Control | Read/Write |

The ForwardReverseActing bit define the how run the CV value of the PID.

- 0 :Reverse Acting. The control variable reduced when the SP is greater than the PV.
- 1: Forward Acting. The control variable increased when the PV is greater than the SV.

## OutputLimitingEnabled

| Description | Data Format | HSC Mode | Function | User Program Access |
|---|---|---|---|---|
| PD.10.OutputLimitingEnabled | Bit | 0 or 1 | Control | Read/Write |

The OutputLimitingEnabled bit sets the enable or disable of the CV output limitation of PID.

- 0 : disable the output limitation
- 1: enable the output limitation. The ControlVariableHighLimit and ControlVariableLowLimit values are used for output limiting.

## GainRangeSelection

| Description | Data Format | HSC Mode | Function | User Program Access |
|---|---|---|---|---|
| PD.10.GainRangeSelection | Bit | 0 or 1 | Control | Read/Write |

The GainRangeSelection bit sets the scale of the gain range value.

- 0: Value * 0.1 -> P Gain = P Gain / 10 . D Gain = D Gain / 10
- 1: Value * 0.01 -> P Gain = P Gain / 100

## SetPointScalingEnabled

| Description | Data Format | HSC Mode | Function | User Program Access |
|---|---|---|---|---|
| PD.10.SetPointScalingEnabled | Bit | 0 or 1 | Control | Read/Write |

- 0: disable the Scale.

- 1: enable the Scale.

## LoopUpdateTooFast

| Description | Data Format | HSC Mode | Function | User Program Access |
|---|---|---|---|---|
| PD.10. LoopUpdateTooFast | Bit | 0 or 1 | Status | Read/Write |

The LoopUpdateTooFast flag is set by the PID algorithm when the PID update is not executed cause of the PLC scan time limitation.

To solve this, speed down the PID loop control or use PIT interrupt.

## ProcessValueInDeadBand

| Description | Data Format | HSC Mode | Function | User Program Access |
|---|---|---|---|---|
| PD.10.ProcessValueInDeadBand | Bit | 0 or 1 | Status | Read/Write |

The ProcessValueInDeadBand bit is set when the PV(Process Value) is within the Zero-Crossing Deadband range.

## ZeroCrossingDeadband

| Description | Data Format | HSC Mode | Function | User Program Access |
|---|---|---|---|---|
| PD.10.ZeroCrossingDeadband | Word | 0 ~ 32,767 | Control | Read/Write |

The deadband extends above and below the setpoint by the value entered and is entered at the zero crossing of the process variable and the setpoint.

This means that the deadband is in effect only after the process variable enters the deadband and passes through the setpoint.

The valid range is 0 to the scaled maximum, or 0 to 16,383 when no scaling exists.

## Kc (Controller Gain)

| Description | Data Format | HSC Mode | Function | User Program Access |
|---|---|---|---|---|
| PD.10.Kc | Word | 0 ~ 32,767 | Control | Read/Write |

The KC flag is used to enter the gain value of the X8 Series PLC. Kc is proportional gain value of PID control and has the following values.

- GainRangeSelection = 0 : 0 ~ 3276.7
- GainRangeSelection = 1 : 0 ~ 327.67

Set to 50% of the value when the vibration occurs if the Ti and Td is 0.

**TIP**

Gain = output/ input : Input to output ratio

Pb= 1/Gain * 100(%) : Constant that determines the size of the control deviation (e) for the control output

*Pb = Proportional Band

## Ti (Integral Time)

| Description | Data Format | HSC Mode | Function | User Program Access |
|---|---|---|---|---|
| PD.10.Ti | Word | 0 ~ 32,767 | Control | Read/Write |

The Ti is I, that is, integral time for I Action of the PID control.

That is,

$$= K_p \cdot e \quad + \quad \frac{K_p}{T_I}\int e\,dt \quad + \quad K_p \cdot T_D \frac{de}{dt}$$

In above formula

$$\frac{K_p}{T_I}\int e\,dt$$

This part is a major parameter that is divided accumulated error in the integral time (Ti).

- GainRangeSelection = 0 : 0 ~ 3276.7 min.
- GainRangeSelection = 1 : 0 ~ 327.67 min.

## Td (Derivative Time)

| Description | Data Format | HSC Mode | Function | User Program Access |
|---|---|---|---|---|
| PD.10.Td | Word | 0 ~ 32,767 | Control | Read/Write |

The Td is D, that is, derivative time for D Action of the PID control.

That is,

$$= K_p \cdot e \quad + \quad \frac{K_p}{T_I} \int e\,dt \quad + \quad K_p \cdot T_D \frac{de}{dt}$$

In above formula:

$$K_p \cdot T_D \frac{de}{dt}$$

part is a major parameter that is multiplied derivative value by derivative time (Td). The range is 0 ~327.67 min.

Generally Td is set to 1/2 of the Td.

## LoopUpdateTime

| Description | Data Format | HSC Mode | Function | User Program Access |
|---|---|---|---|---|
| PD.10.LoopUpdateTime | Word | 0 ~ 32,767 | Control | Read/Write |

The LoopUpdateTime is PID operating cycle of the PID control. The operating cycle is 0.01sec scaling.

Enter the LoopUpdateTime data up to 10 times faster than the actual cycle. When the PID operating cycle is within the PIT mode, enter the same data with timer interrupt occur cycle in PIT configuration part.

## FeedForwardBias

| Description | Data Format | HSC Mode | Function | User Program Access |
|---|---|---|---|---|
| PD.10.FeedForwardBias | Word | -16,383 ~ 16,383 | Control | Read/Write |

The FeedForwardBias is used to compensate for disturbances that may affect the CV.

## ScaledError

| Description | Data Format | HSC Mode | Function | User Program Access |
|---|---|---|---|---|
| PD.10.ScaledError | Word | -32,767 ~ 32,767 | Status | Read Only |

The ScaledError is the difference between the process variable and the setpoint.

## ErrorCode

| Description | Data Format | HSC Mode | Function | User Program Access |
|---|---|---|---|---|
| PD.10.ErrorCode | Word | -32,767 ~ 32,767 | Status | Read Only |

The ErrorCode displays the PID Error Code.

| Error Code | Description |
|---|---|
| 0x01 | Kc Error |
| 0x02 | Ti Error |
| 0x03 | Td Error |
| 0x04 | Bias Error |
| 0x05 | Loop Update Error |
| 0x11 | CV Low Limit Error |
| 0x12 | CV High Limit Error |
| 0x13 | CV Low Limit greater than High Limit |
| 0x21 | Set Point Minimum greater than Maximum |
| 0x31 | Set Point out of range |
| 0x41 | Deadband Error |

# ASCII Instructions

## Introduction

This chapter describes the X8 Series PLC ASCII instructions. This chapter organized as follows.

The X8 Series PLC supports ASCII typed data and contains instructions for processing characters, string and ASCII communications. The following table shows the ASCII instructions.

**Instructions for ASCII character (string)**

| Instructions | Descriptions |
|---|---|
| ATOI | Convert ASCII String to Integer |
| ITOA | Convert Integer to ASCII String |
| ACN | ASCII String Concatenate<br>Concatenate two of characters (strings). |
| AEX | ASCII String Extract<br>Extract part of the string data. |
| ASCH | Search a string.<br>Search a specific string. |
| ACMP | ASCII String Compare<br>Compare two strings. |

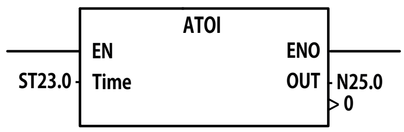
**Instructions for ASCII communications**

| Instructions | Descriptions |
|---|---|
| ARNL | Number of Characters for Line<br>Get the number of characters contains End of Line character in buffer. |
| ARNC - | Number of Characters in Buffer<br>Get number of characters in buffer. |
| ACB - | ASCII Clear Buffer<br>Clear sending and receiving communications buffer. |
| AHS | Handshake<br>On/OFf the handshake control signal in the modem communications. |
| ARC | Read Characters<br>Read the received character in buffer and store the data to String Data Table. |
| AWA | Write String with Append<br>Add two characters specified in Channel Configuration and write the data to port. |
| AW | Write a string.<br>Write a string to port. |

The following table shows Error code of the ASCII instructions.

| Error Code | Descriptions |
|---|---|
| 0 | Success |
| 2 | Illegal parameter |
| 3 | Not supported |
| 4 | Channel is shutdown |
| 5 | Protocol contention |
| 6 | Transmit is in progress |
| 7 | CTS Signal lost |
| 10 | Source Control Data Table invalid |
| 11 | Source String Length invalid |
| 12 | Request String Length invalid |
| 13 | Unload bit in Control Data Table is set |
| 14 | ACB Instruction deletion |
| 15 | Channel Configuration was changed |

## ATOI - Convert ASCII String to Integer




Instruction Type: output

The ATOI instruction converts data in the STRING data table to an integer. Only 0~9 characters supported.

For example:

- ST23.0 = if there is "12" character, 12 is output.
- ST23.0 = if there is "ABCDEF12FFFF" character, 12 is output.

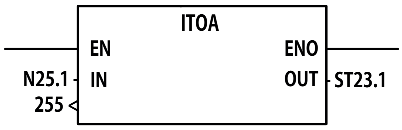
To support second example, the ATOI instruction allows number converting only. Therefore, If you enter ST23.0 = "AA", this instruction does not output any data since there is no numbers.

The data range is

- WORD : -32,768 ~ 32,767
- LONG OWRD : -2,147,483,648 ~ 2,147,483,647

| Parameter | Data Table | | | | | | | | | | | | | SFR | | | | | | | | | Address Mode | | | Address Level | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Y | X | SR | B | TM, CT, CR | N | F | ST | A | L | MG, PD | RP | PS | RTC | HSC | PTO, PWM | STI | EII | BHI | MMI | CS-Comms | DLS-DataLog | Immediate | Direct | Indirect | Bit | Word | LongWord | Element |
| Source | | | | | | | | √ | | | | | | | | | | | | | | | | √ | | | | | √ |
| Destination | √ | √ | | √ | √ | | | | √ | | | √ | | | | | | | | | | | | | | | √ | √ | |

# ITOA - Convert ASCII String to Integer




Instruction Type: output

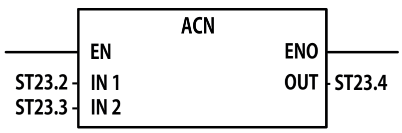
The ITOA instruction converts an ASCII integer data to string. Only 0~9 characters are supported.

The input data range is:

- LONG OWRD : -2,147,483,648 ~ 2,147,483,647

| Parameter | Data Table | | | | | | | | | | | | | SFR | | | | | | | | | Address Mode | | | Address Level | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Y | X | SR | B | TM, CT, CR | N | F | ST | A | L | MG, PD | RP | PS | RTC | HSC | PTO, PWM | STI | EII | BHI | MMI | CS-Comms | DLS-DataLog | Immediate | Direct | Indirect | Bit | Word | LongWord | Element |
| Source | √ | √ | | √ | √ | √ | | | | √ | | √ | | | | | | | | | | | √ | √ | | | √ | √ | |
| Destination | | | | | | | | | | | | | | | | | | | | | | | | √ | | | | | √ |

# ACN - ASCII String Concatenate




Instruction Type: output

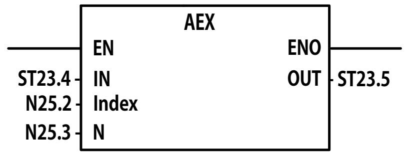
The ACN instruction combines two strings.

[For example]

- if Source 1="AB", Source 2="CD", "ABCD" is stored in Destination.

| Parameter | Data Table | | | | | | | | | | | | | SFR | | | | | | | | | Address Mode | | | Address Level | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Y | X | SR | B | TM, CT, CR | N | F | ST | A | L | MG, PD | RP | PS | RTC | HSC | PTO, PWM | STI | EII | BHI | MMI | CS-Comms | DLS-DataLog | Immediate | Direct | Indirect | Bit | Word | LongWord | Element |
| Source 1 | | | | | | | | √ | | | | | | | | | | | | | | | | √ | | | | | √ |
| Source 2 | | | | | | | | √ | | | | | | | | | | | | | | | | √ | | | | | √ |
| Destination | | | | | | √ | | √ | | | | | | | | | | | | | | | | √ | | | | | √ |

## AHX - ASCII String Extract




Instruction Type: output

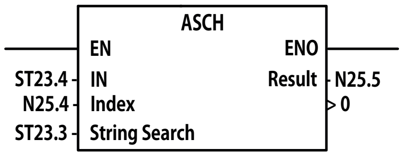
The AHX instruction extracts N characters from the Index position in a string.

[For example]

If Source 1="NEW  X8 PLC ", INDEX=5, and N=6, "X8 PLC" is stored to destination.

| Parameter | Data Table | | | | | | | | | | | | | SFR | | | | | | | | | Address Mode | | | Address Level | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Y | X | SR | B | TM, CT, CR | N | F | ST | A | L | MG, PD | RP | PS | RTC | HSC | PTO, PWM | STI | EII | BHI | MMI | CS-Comms | DLS-DataLog | Immediate | Direct | Indirect | Bit | Word | LongWord | Element |
| Source 1 | | | | | | | | √ | | | | | | | | | | | | | | | | √ | | | | | √ |
| INDEX | √ | √ | | √ | √ | √ | | | √ | | | √ | | | | | | | | | | | √ | | | | | | |
| N | √ | √ | | √ | √ | √ | | | √ | | | | | | | | | | | | | | √ | | | | | | |
| Destination | | | | | | | | √ | | | | | | | | | | | | | | | | √ | | | | | √ |

# ASCH - Searching String

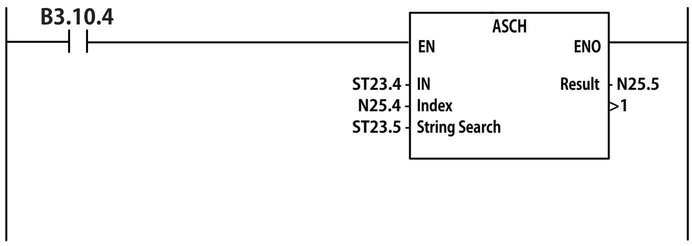



Instruction Type: output

The ASCH instruction is an instruction to search a string you want from the source string data table.

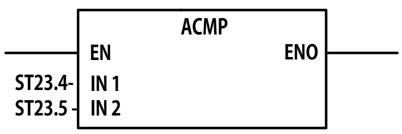
- IN: Source String
- Index: Start point to search (The first character is 1)
- String To Search: String to search

The following ladder example shows that search the string in ST23.5 starting at the Index, for the string found in ST23.4. the position result is stored in "1".




| Parameter | Data Table | | | | | | | | | | | | | SFR | | | | | | | | | Address Mode | | | Address Level | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Y | X | SR | B | TM, CT, CR | N | F | ST | A | L | MG, PD | RP | PS | RTC | HSC | PTO, PWM | STI | EII | BHI | MMI | CS-Comms | DLS-DataLog | Immediate | Direct | Indirect | Bit | Word | LongWord | Element |
| Source 1 | | | | | | | | √ | | | | | | | | | | | | | | | | √ | | | | | √ |
| INDEX | | | | | | | | √ | √ | | | √ | | | | | | | | | | | | √ | | | | | √ |
| String Search | √ | √ | | √ | √ | √ | | | √ | | | | | | | | | | | | | | √ | √ | | | √ | | |
| Destination | √ | √ | | √ | √ | √ | | | √ | | | √ | | | | | | | | | | | | √ | | | √ | | |

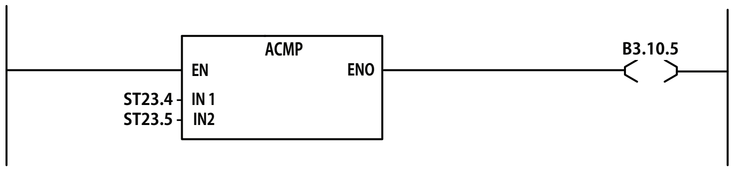
# ACMP - Compare String




Instruction Type: output

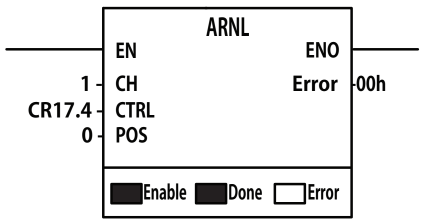
The ACMP instruction is used to compare two strings in Source 1 and 2. If the strings are identical, the rung is true.

The following example shows that compare two string in the Source 1 and 2. If the strings are identical, the rung is true and store the result to destination.




| Parameter | Data Table | | | | | | | | | | | | | SFR | | | | | | | | | Address Mode | | | Address Level | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Y | X | SR | B | TM, CT, CR | N | F | ST | A | L | MG, PD | RP | PS | RTC | HSC | PTO, PWM | STI | EII | BHI | MMI | CS-Comms | DLS-DataLog | Immediate | Direct | Indirect | Bit | Word | LongWord | Element |
| Source 1 | | | | | | | | √ | | | | | | | | | | | | | | | | √ | | | | | √ |
| Source 2 | | | | | | | | √ | | | | | | | | | | | | | | | | √ | | | | | √ |
| Destination | | | | | | √ | | | | | | | | | | | | | | | | | | √ | | | | | √ |

## ARNL - Number of Characters for Line




Instruction Type: output

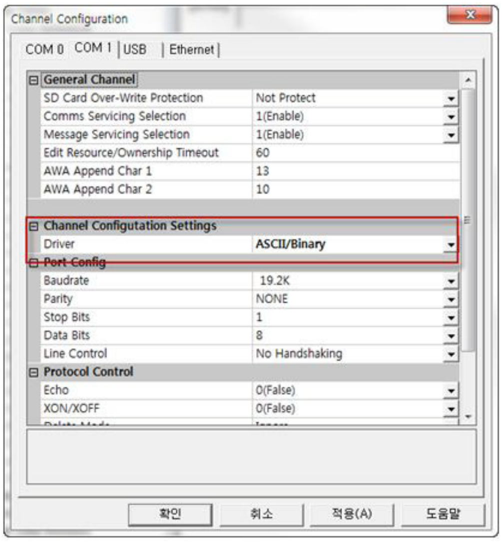
The ARNL instruction gets number of characters include End Of Line in the buffer.

The result data gets from buffer is stored to POS or Position in the Control Data Table (CR).

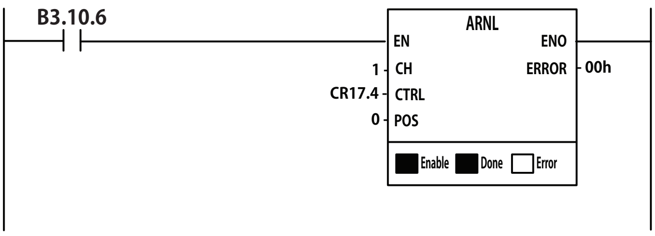
The parameters are:

- CH : serial port number ( 0:COM1, 1:COM2, 2:USB)
- CTRL : CR (Control) data table

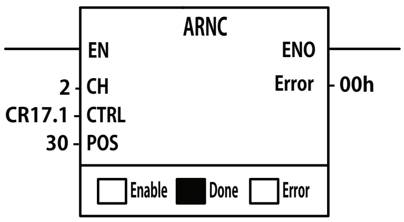
**IMPORTANT** Driver configuration in the Channel Configuration Settings should be set to ASCII/BIN. Following figure shows the Channel 1 configuration part. If ARNL instruction is executed without Driver configuration, Error=3 is generated.




Following ladder example reflects the CH 1 configuration.




## ARNC - Number of Characters in Buffer




Instruction Type: output

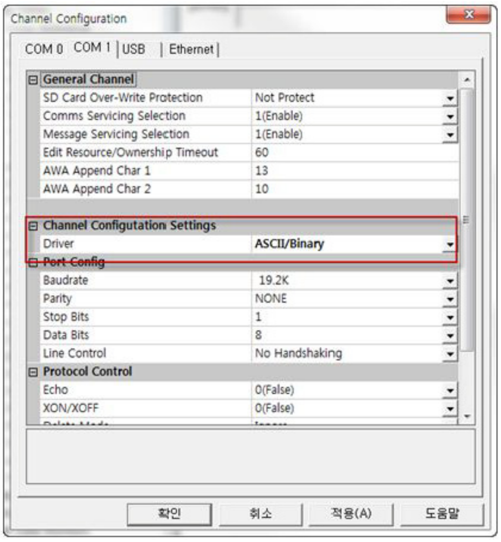
The ARNC instruction gets the number of the characters in butter.

The result data gets from buffer is stored to POS or Position in the Control Data Table (CR).

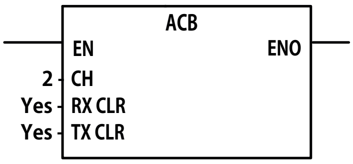
The parameters are:

- CH : serial port number ( 0:COM1, 1:COM2, 2:USB)
- CTRL : CR (Control) data table

**IMPORTANT** Driver configuration in the Channel Configuration Settings should be set to ASCII/BIN. Following figure shows the Channel 1 configuration part. If ARNC instruction is executed without Driver configuration, Error=3 is generated.




## ACB - Clear Buffer

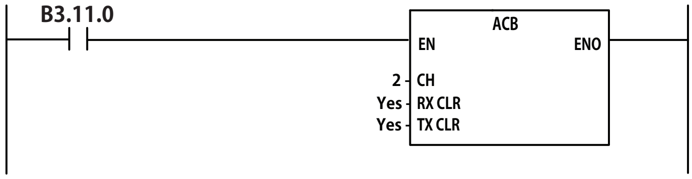



Instruction Type: output

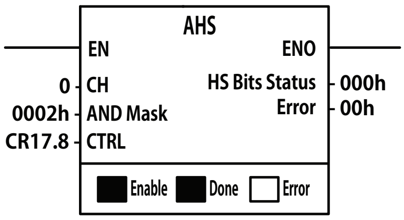
The ABC instruction clears sending and receiving communications buffer.

The parameters are:

- CH : serial port number ( 0:COM1, 1:COM2, 2:USB)
- RX CLR : Yes or No
- TX CLR : Yes or No




## AHS - Handshake

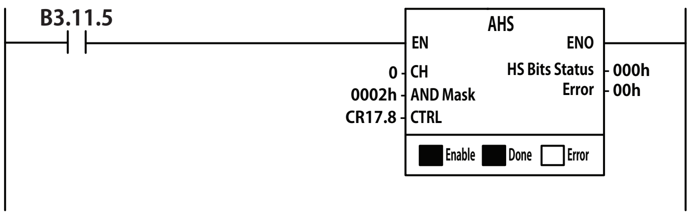



Instruction Type: output

The AHS instruction is used to on/off the RTS signal, handshake control signal.

The parameters are:

- CH : serial port number ( 0:COM1, 1:COM2, 2:USB)
- AND Mask : reset the RTS control signal. Bit 1 corresponds to the RTS, a value of "2" in the AND Mask resets the RTS, a value of "0" remains the RTS signal. The following examples shows that the value of 2 in the AND Mask resets the RTS.
- OR Mask : sets the RTS control signal. Bit 1 corresponds to the RTS , a value of "2" in the OR Mask sets the RTS, a value of "0" remains the RTS signal.
- CTRL : CR (Control) data table



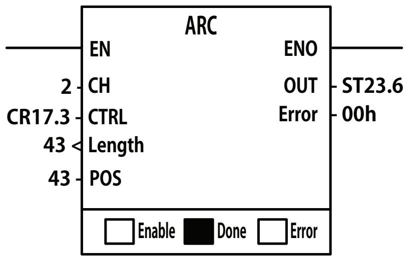
**TIP**

If 2 (USB) is set to CH, Error 3 is generated since the USB is not support Handshake.

Handshake control is only possible in ASCII/BIN communications. Other protocols are controlled in PLC autonomously.

| Error Code | Descriptions |
|---|---|
| 0 | Success |
| 2 | Illegal parameter |
| 3 | Not supported |
| 4 | Channel is shutdown |
| 5 | Protocol contention |
| 6 | Transmit is in progress |
| 7 | CTS Signal lost |
| 10 | Source Control Data Table invalid. |
| 11 | Source String Length invalid. |
| 12 | Request String Length invalid. |
| 13 | Unload bit in Control Data Table is set. |
| 14 | ACB Instruction deletion |
| 15 | Channel Configuration was changed. |

# ARC - Read Characters



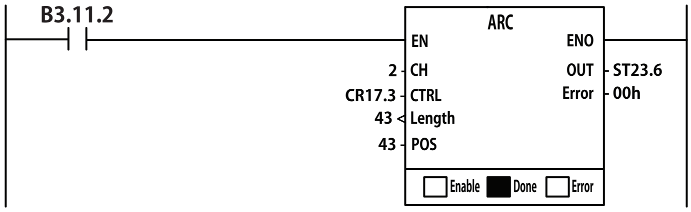

Instruction Type: output

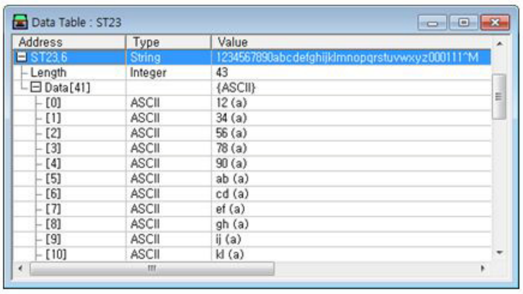
The ARC instruction is used to read characters from the buffer, and store them in the String Data Table.

The parameters are:

- CH : serial port number ( 0:COM1, 1:COM2, 2:USB)
- CTRL : CR (Control) data table
- Length : data length to read

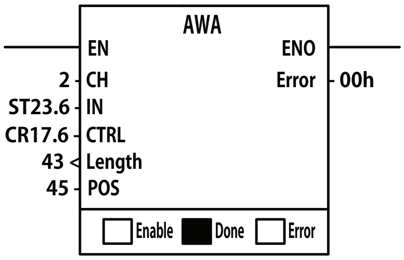
The following example shows that read data from a buffer through USB port and store them to ST23.6.





Data Table : ST23

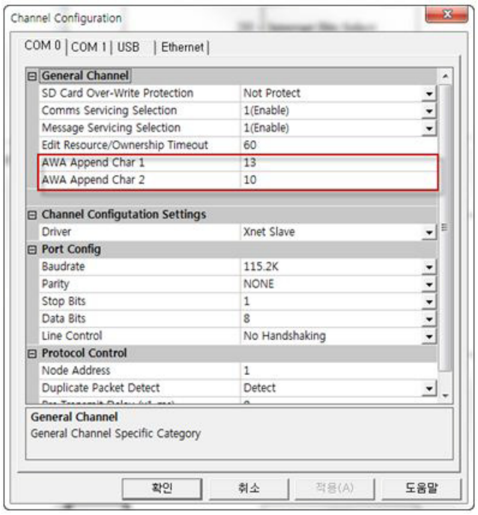
| Address | Type | Value |
|---|---|---|
| ST23.6 | String | 1234567890abcdefghijklmnopqrstuvwxyz000111^M |
| Length | Integer | 43 |
| Data[41] | | {ASCII} |
| [0] | ASCII | 12 (a) |
| [1] | ASCII | 34 (a) |
| [2] | ASCII | 56 (a) |
| [3] | ASCII | 78 (a) |
| [4] | ASCII | 90 (a) |
| [5] | ASCII | ab (a) |
| [6] | ASCII | cd (a) |
| [7] | ASCII | ef (a) |
| [8] | ASCII | gh (a) |
| [9] | ASCII | ij (a) |
| [10] | ASCII | kl (a) |

## AWA - Write String with Append




Instruction Type: output

The AWA instruction adds two characters from the Channel Configuration and writes them to the port. The default is CR(0DH) and LF(0AH).

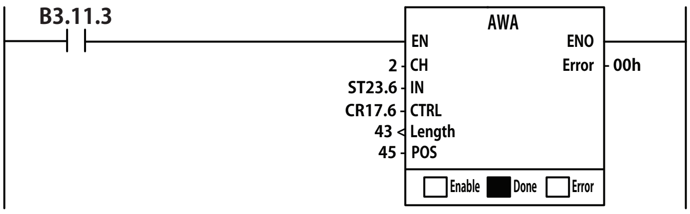



The input parameters are:

- CH : serial port number ( 0:COM1, 1:COM2, 2:USB)
- IN : STRING data table to write to destination
- CTRL : CR (Control) data table
- Length : data length to write to port

The output parameter is:

- POS : number of data complete write



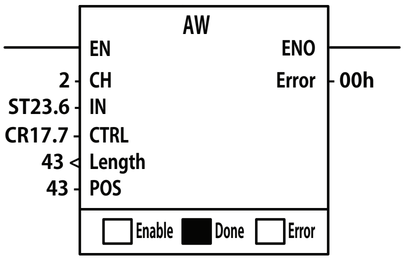

**IMPORTANT**

It is set when the rung condition goes false to true. If the rung condition is changed true to false during the AWA instruction is in run state, the instruction complete the communication output.

Set the writing interval to more 0.5 seconds.

| Parameter | Data Table | | | | | | | | | | | | | SFR | | | | | | | | | Address Mode | | | Address Level | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Y | X | SR | B | TM, CT, CR | N | F | ST | A | L | MG, PD | RP | PS | RTC | HSC | PTO, PWM | STI | EII | BHI | MMI | CS-Comms | DLS-DataLog | Immediate | Direct | Indirect | Bit | Word | LongWord | Element |
| CH | | | | | | | | | | | | | | | | | | | | | | | √ | | | | √ | | |
| IN | | | | | | | | √ | | | | | | | | | | | | | | | | √ | | | | | √ |
| CTRL | | | | | √ | | | | | | | | | | | | | | | | | | | √ | | | | | √ |

# AW - Write String




Instruction Type: output

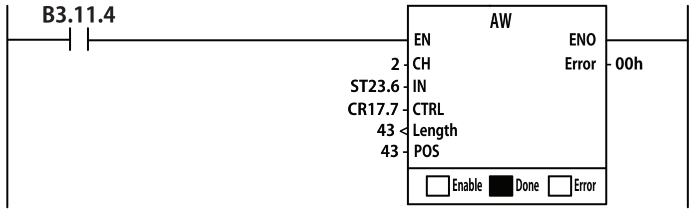
The AW instruction to write strings from a source string to the port.

The input parameters are:

- CH : serial port number( 0:COM1, 1:COM2, 2:USB)
- IN : STRING data table to write to destination
- CTRL : CR (Control) data table
- Length : data length to write to port

The output parameter is:

- POS : number of data completing writing




| **IMPORTANT** | t is set when the rung condition goes false to true. If the rung condition is changed true to false during the AW instruction is in run state, the instruction complete the communication output.<br>Set the writing interval to more 0.5 seconds. |
|---|---|

| **Parameter** | **Data Table** | | | | | | | | | | | | | **SFR** | | | | | | | | | **Address Mode** | | | **Address Level** | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Y | X | SR | B | TM, CT, CR | N | F | ST | A | L | MG, PD | RP | PS | RTC | HSC | PTO, PWM | STI | EII | BHI | MMI | CS-Comms | DLS-DataLog | Immediate | Direct | Indirect | Bit | Word | LongWord | Element |
| CH | | | | | | | | | | | | | | | | | | | | | | | √ | | | | √ | | |
| IN | | | | | | | | √ | | | | | | | | | | | | | | | | √ | | | | | √ |
| CTRL | | | | | √ | | | | | | | | | | | | | | | | | | | √ | | | | | √ |

# Communications Instructions

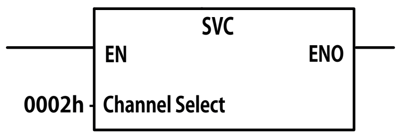
## Introduction

This chapter describes the X8 Series PLC communications instructions. This chapter organized as follows.

The following table shows communication instructions of the X8 Series PLC.

| Instruction | Description |
|---|---|
| SVC | Interrupt the program scan to execute the service communications part of the operating cycle. The scan then resumes at the instruction following the SVC instruction. |
| MSG | Transfer data from one device to another. |

## SVC - Service Communication




Instruction Type: output

Under normal PLC ladder program operation the controller processes communications once every time it scans the control program. if the ladder scan is long, it can cause problems in the overall system performance under the situation that communication is done in the shortest time. The relatively time-consuming serial communication also can cause problems in the PLC performance since the timer interrupt service routine will need a lot of time on communications.

To solve theses problems, you can use the SVC instruction.

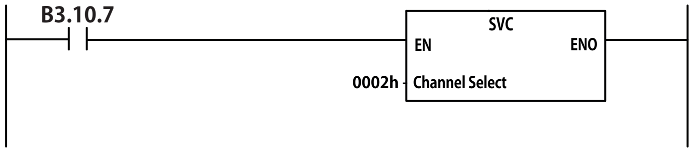
That is, The SVC instruction is used to improve communications performance and throughput, but also causes the ladder scan to be longer.

The input parameters are:

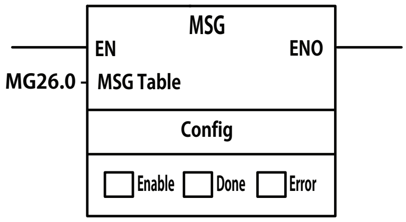
- Channel Select : serial port number ( 0:COM1, 1:COM2, 2:USB)
  If you execute the SVC instruction for 2 or more ports at the same time, logically ORed 3 ports and then substitute them.

| Channel Select | Channel(s) Serviced |
|---|---|
| 1h(0 bit) | Serial Channel 0 |
| 2h(1 bit) | Serial Channel 1 |
| 4h(2 bit) | Serial Channel 2(USB) |
| 8h(3 bit) | Ethernet Channel 3 |

The following is an example of the SVC instruction.




## MSG - Message Communication




Instruction Type: output

The MSG instruction send the message to the specified protocol.

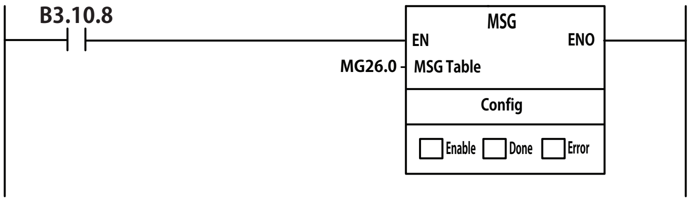
The supported protocols are:

- Xnet
  - Xnet X8CPU Read
  - Xnet X8CPU Write
  - Modbus
  - Modbus Read Coil Status (01)
  - Modbus Input Status (02)
  - Modbus Read Holding Registers (03)
  - Modbus Read Input Registers (04)
  - Modbus Write Single Coil (05)
  - Modbus Write Single Register (06)
  - Modbus Write Multiple Coils (0F)
  - Modbus Write Multiple Registers (10)
- PCCC
  - PCCC 500CPU Read
  - PCCC 500CPU Write
  - PCCC 485CIF Read
  - PCCC 485CIF Write
  - PCCC PLC5 Read
  - PCCC PLC5 Write
- NX Protocol
  - NX-Plus Read Bits (21)
  - NX-Plus Write Bits (22)

- NX-Plus Read Words (23)
- NX-Plus Write Words (24)

- CIP Generic ( Add when you choose EterNet Port as communications port)

This MSG instruction, like other instructions, is set when the rung state is true. The following is an example of the MSG instruction.

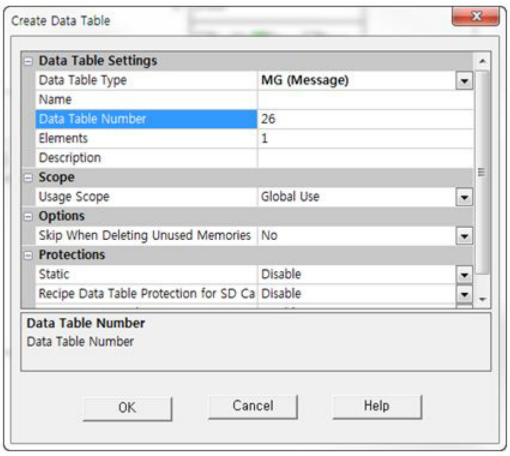



How to use the MSG instruction

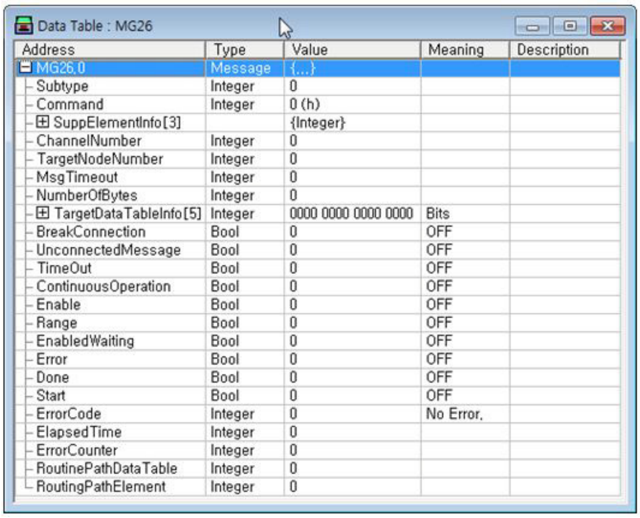
The MSG instruction is input parameter and needs MG data table. Therfore, the MG data table should be created.

Right click on the Data Tables menu to create new data table and select MG (Message) for Data Table Type.

Though the Data Table Number is 26 in the following figure, use an unallocated number . And set the number of internal Element. ( 1 in the figure (Default setting) )




The following figure shows elements of MG data table.



Data Table : MG26

| Address | Type | Value | Meaning | Description |
|---|---|---|---|---|
| MG26.0 | Message | {...} | | |
| Subtype | Integer | 0 | | |
| Command | Integer | 0 (h) | | |
| SuppElementInfo[3] | | {Integer} | | |
| ChannelNumber | Integer | 0 | | |
| TargetNodeNumber | Integer | 0 | | |
| MsgTimeout | Integer | 0 | | |
| NumberOfBytes | Integer | 0 | | |
| TargetDataTableInfo[5] | Integer | 0000 0000 0000 0000 | Bits | |
| BreakConnection | Bool | 0 | OFF | |
| UnconnectedMessage | Bool | 0 | OFF | |
| TimeOut | Bool | 0 | OFF | |
| ContinuousOperation | Bool | 0 | OFF | |
| Enable | Bool | 0 | OFF | |
| Range | Bool | 0 | OFF | |
| EnabledWaiting | Bool | 0 | OFF | |
| Error | Bool | 0 | OFF | |
| Done | Bool | 0 | OFF | |
| Start | Bool | 0 | OFF | |
| ErrorCode | Integer | 0 | No Error. | |
| ElapsedTime | Integer | 0 | | |
| ErrorCounter | Integer | 0 | | |
| RoutinePathDataTable | Integer | 0 | | |
| RoutingPathElement | Integer | 0 | | |

## Subtype

| Address | Data Format | HSC Mode | Function | User Program Access |
|---|---|---|---|---|
| MGn.0.Subtype | Word | -32,767~ 32,767 | Status | Read Only |

The Subtype instruction is used to set the Message type ( Communication method).

- 0: Xnet Master
- 1: Modbus Master
- 2: PCCC
- 3: CIP Generic
- 4: NX-Plus Master

## Command

| Address | Data Format | HSC Mode | Function | User Program Access |
|---|---|---|---|---|
| MGn.0.Command | Word | -32,767~ 32,767 | Status | Read Only |

The Command instruction is used to store the specified instruction code.

## SuppElementInfo[0] ~ [2]

| Address | Data Format | HSC Mode | Function | User Program Access |
|---|---|---|---|---|
| MGn.0.SuppElementInfo[0] ~[2] | Word | -32,767~ 32,767 | Status | Read Only |

The SuppElementInfo instruction is used to store the additional Element information.

## ChannelNumber

| Address | Data Format | HSC Mode | Function | User Program Access |
|---|---|---|---|---|
| MGn.0.ChannelNumber | Word | -32,767~ 32,767 | Status | Read Only |

The ChannelNumber instruction is used to store the channel number to send this message.

## MsgTimeout

| Address | Data Format | HSC Mode | Function | User Program Access |
|---|---|---|---|---|
| MGn.0.MsgTimeout | Word | -32,767~ 32,767 | Status | Read Only |

This is the Message Timeout word. The value is 10m sec unit.

## NumberOfByte

| Address | Data Format | HSC Mode | Function | User Program Access |
|---|---|---|---|---|
| MGn.0.NumberOfByte | Word | -32,767~ 32,767 | Status | Read Only |

The NumberOfByte instruction stores the number of Data Byte to send. When the subtype is Modbus Master of NX-Plus, it stores the number of Element.

## TargetDataTableInfo[0] ~ [4]

| Address | Data Format | HSC Mode | Function | User Program Access |
|---|---|---|---|---|
| MGn.0.TargetDataTableInfo[0] ~ [4] | Word | -32,767~ 32,767 | Status | Read Only |

The TargetDataTableInfor instructions is used to store the data table information of the Target node.

## BreakConnection

| Address | Data Format | HSC Mode | Function | User Program Access |
|---|---|---|---|---|
| MGn.0.BreakConnection | Bit | 0 or 1 | Status | Read / Write |

When the BreakConnection bit is set, the Ethernet TCP connection will be closed after the MSG instruction has completed.

## UnConnectedMessage

| Address | Data Format | HSC Mode | Function | User Program Access |
|---|---|---|---|---|
| MGn.0.UnConnectedMessage | Bit | 0 or 1 | Status | Read Only |

When the UnConnectedMessage bit is set, an unconnected type message will be generated whenever that EtherNet/IP MSG instruction is triggered.

## Timeout

| Address | Data Format | HSC Mode | Function | User Program Access |
|---|---|---|---|---|
| MGn.0.TimeOut | Bit | 0 or 1 | Status | Read Only |

The Timeout instruction is set when it has not receive any response for Timeout. If the Timeout is set by the user, the MSG instruction is cleared during execution.

## ContinuousOperation

| Address | Data Format | HSC Mode | Function | User Program Access |
|---|---|---|---|---|
| MGn.0.ContinuousOperation | Bit | 0 or 1 | Status | Read Only |

When this bit is set, the message is triggered after the message transmission.

## Enable

| Address | Data Format | HSC Mode | Function | User Program Access |
|---|---|---|---|---|
| MGn.0.Enable | Bit | 0 or 1 | Status | Read/Write |

When this bit is set, the message is triggered.

The Enable bit is set when rung conditions go true and the MSG is enabled.

The enable conditions are:

- when the command packet is built and put into one of the MSG buffers
- the request is put in the MSG queue

You may clear this bit in order to re-trigger a MSG instruction or set the ContinuousOperation to automatic execution.

There are two method to clear this bit.

Set the Done or Error bit in the MSG data table or use RST instruction to clear the Enable bit.

**IMPORTANT** Do not set this bit from the ladder program.


## Range

| Address | Data Format | HSC Mode | Function | User Program Access |
|---|---|---|---|---|
| MGn.0.Range | Bit | 0 or 1 | Status | Read Only |

When this bit is cleared, the MSG instruction is used for Local communication. When this is set, the MSG instruction is used for Remote communication. This is used only in the PCCC protocol.

## EnabledWaiting

| Address | Data Format | HSC Mode | Function | User Program Access |
|---|---|---|---|---|
| MGn.0.EnabledWaiting | Bit | 0 or 1 | Status | Read Only |

When this bit is set, the message is enabled and waiting for message transmission.

## Error

| Address | Data Format | HSC Mode | Function | User Program Access |
|---|---|---|---|---|
| MGn.0.Error | Bit | 0 or 1 | Status | Read Only |

The Error bit is set when message transmission has failed.

## Done

| Address | Data Format | HSC Mode | Function | User Program Access |
|---|---|---|---|---|
| MGn.0.Eone | Bit | 0 or 1 | Status | Read Only |

The Done bit is set when the message is transmitted successfully. The Done bit is cleared the next time the associated rung goes from false to true.

## Start

| Address | Data Format | HSC Mode | Function | User Program Access |
|---|---|---|---|---|
| MGn.0.Start | Bit | 0 or 1 | Status | Read Only |

The Start bit is set when the processor receives acknowledgment from the target device. The Start bit is cleared when the Done and ErrorTimeout bit is set.

## ErrorCode

| Address | Data Format | HSC Mode | Function | User Program Access |
|---|---|---|---|---|
| MGn.0.ErrorCode | Bit | 0 or 1 | Status | Read Only |

When this bit is set, the error code is displayed to check.

## ElapsedTIme

| Address | Data Format | HSC Mode | Function | User Program Access |
|---|---|---|---|---|
| MGn.0.ElapsedTIme | Bit | 0 or 1 | Status | Read Only |

This bit ise cleared when the message is enabled.

## ErrorCounter

| Address | Data Format | HSC Mode | Function | User Program Access |
|---|---|---|---|---|
| MGn.0.ErrorCount | Word | -32,768 ~ 32,767 | Status | Read Only |

This bit is used to calculate the number of errors. If the error count exceeds the error count configuration value, this message is not triggered.

- Low Byte - Error Counter Configuration
- High Byte - Error Counter Status

This feature also applied to the Countinuous Operation bit.

## RoutiongPathDataTblNumber

| Address | Data Format | HSC Mode | Function | User Program Access |
|---|---|---|---|---|
| MGn.0.RoutiongPathDataTblNumber | Word | -32,768 ~ 32,767 | Status | Read Only |

This word displays the Routing Path Data Table Number specified by the MSG configuration.

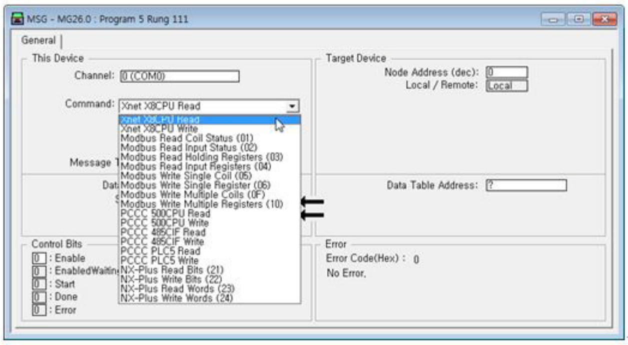
## RoutiongPathDataTable

| Address | Data Format | HSC Mode | Function | User Program Access |
|---|---|---|---|---|
| MGn.0.RoutiongPathDataTable | Word | -32,768 ~ 32,767 | Status | Read Only |

This word displays the Routing Path Data Table Elements specified by the MSG configuration.

The MSG instruction configuration process:

- Configure the port and protocol to use
- Configure nod address of the target device and Local or Remote communication method.
- Enter data table addresses of both sides


MSG - MG26.0 : Program 5 Rung 111
General
This Device
Channel:
0 (COM0)
Command:
Xnet X8CPU Read
Xnet X8CPU Read
Xnet X8CPU Write
Modbus Read Coil Status (01)
Modbus Read Input Status (02)
Modbus Read Holding Registers (03)
Modbus Read Input Registers (04)
Modbus Write Single Coil (05)
Modbus Write Single Register (06)
Modbus Write Multiple Coils (0F)
Modbus Write Multiple Registers (10)
PCCC 500CPU Read
PCCC 500CPU Write
PCCC 485CIF Read
PCCC 485CIF Write
PCCC PLC5 Read
PCCC PLC5 Write
NX-Plus Read Bits (21)
NX-Plus Write Bits (22)
NX-Plus Read Words (23)
NX-Plus Write Words (24)
Target Device
Node Address (dec):
0
Local / Remote:
Local
Data Table Address:
?
Control Bits
0 : Enable
0 : Start
0 : Done
0 : Error
Error
Error Code(Hex) : 0
No Error.

# System Error Code

## System Error Code

| Controller Fault Code (Hex) | Description | Level |
|---|---|---|
| 00 | 1.1.1 No Error | 1.1.2 Recoverable |
| 1.1.3 System Hard Faults | | |
| 01 | System Error - Unexpected Reset or System Watchdog Error | Non-User |
| 02 | System Error - Controller ASIC Error | Non-User |
| 03 | System Error - NVRAM Memory corrupted | Non-User |
| 04 | System Error - Hardware Error | Non-User |
| 05 | System Error - Firmware Error | Non-User |
| 07 | | |
| System Soft Faults | | |
| 08 | System Error - Background User Program Integrity failed. | Non-User |
| 09 | | |
| 0A | | |
| 0B | | |
| 0C | | |
| 0D | | |
| 0E | | |
| 0F | | |
| Ladder Program Faults | | |
| 10 | Ladder Program Error - Integrity check at start executing | Non-User |
| 11 | Ladder Program Error - Ladder Memory currupted | Non-User |
| 12 | Ladder Program Error - User Program Functional Type is incompatible. | Non-User |
| 13 | Ladder Program Error - Online Edit failed. | Non-User |
| 14 | | |
| 15 | | |
| 16 | | |
| 17 | | |

| Controller Fault Code (Hex) | Description | Level |
|---|---|---|
| 18 | Ladder Program Error - Duplicated Label detected. | Non-Recoverable |
| 19 | Ladder Program Error - Start Up Protection faulted. | Recoverable |
| 1A | | |
| 1B | | |
| 1C | | |
| 1D | | |
| 1E | | |
| 1F | Ladder Program Error - Minor Error Bit detected at the end of scan. | Recoverable |
| 20 | Ladder Program Error - User Fault Routine program number is incorrect. | Non-Recoverable |
| 22 | | |
| 23 | | |
| 24 | | |
| 25 | | |
| 26 | | |
| 27 | | |
| 28 | Ladder Program Error - Address Range Check failed during ladder execution. | Recoverable |
| 29 | | |
| 2A | Ladder Program Error - Invalid Bit Address detected. | Non-User |
| 2B | Ladder Program Error - Invalid Word Address detected | Non-User |
| 2C | | |
| 2D | | |
| 2E | PIT Error - See PIT Special Function Register for specific error code. | Recoverable |
| 2F | EII Error - See EII Special Function Register for specific error code. | Recoverable |
| Instruction Specific Faults | | |
| 30 | Instruction Error - Unsupported Instruction detected. | Non-User |
| 31 | Instruction Error - Unsupported Operand Type detected. | Non-User |
| 32 | Instruction Error - Negative Timer Value entered. | Recoverable |
| 33 | Instruction Error - Invalid Parameter detected in PID instruction. | Recoverable |
| 34 | | |
| 35 | Instruction Error - ENDT, SVC, EOS instructions detected in User Fault Routine. | Non-Recoverable |
| 36 | | |

| Controller Fault Code (Hex) | Description | Level |
|---|---|---|
| 37 | Reserved. Not for Use. | |
| 38 | Instruction Error - CALL instruction underflowed. | Non-User |
| 39 | Instruction Error - Subroutine nesting limit exceed. | Non-User |
| 3A | Instruction Error - Address Range Error in SEQx instrucitons. | Recoverable |
| 3B | Instruction Error - Address Range Error in FIFO instructions. | Recoverable |
| 3C | Instruction Error - Invalid sequence length in SEQx instrucitons. | Recoverable |
| 3D | Instruction Error - Invalid length position in FIFO instructions. | Recoverable |
| 3E | Instruction Error - Address Range Error in COPT, COPW, FILT instrucitons. | Recoverable |
| 3F | | |
| 40 | Instruction Error - Invalid Recipe number entered. | Recoverable |
| 41 | Instruction Error - Invalid data write to RTC. | Recoverable |
| 42 | | |
| 43 | | |
| 44 | | |
| 45 | | |
| 46 | | |
| 47 | | |
| 48 | | |
| 49 | | |
| 4A | | |
| 4B | | |
| 4C | | |
| 4D | HSC Error - See HSC Special Function Register for specific error code. | Recoverable |
| 4E | PTO Error - See PTO Special Function Register for specific error code. | Non-User |
| 4F | PWM Error - See PWM Special Function Register for specific error code. | Non-User |
| Embedded I/O Faults | | |
| 50 | | Non-User |
| 51 | Embedded IO Error - Configuration error. | Non-User |
| 52 | Embedded IO Error - Bad checksum. | Non-User |
| 53 | Embedded IO Error - Incompatible base. | Non-User |

# Appendix B

## MSG Instruction Error Code

### MSG Instruction Error Code

| MSG Error Code Mapping | | |
|---|---|---|
| **Internal Fail Code (Hex)** | | **Description** |
| **High byte** | **Low byte** | |
| **Identifier Code** | **Error Code group** | |
| No Error | | |
| 00 | 00 | No Error |
| Link Layer Qualification<br>Error Confirmation from Link Layer | | |
| 01 | 11 | NAK No Memory retries by link layer exhausted. |
| 02 | | Target node cannot respond because message is too large. |
| 03 | | Target node cannot respond because it does not understand the command parameters. |
| 04 | | Network is not active or duplicate node detected. |
| 05 | | Target node cannot respond because requested function is not available. |
| 06 | | Target node does not respond. |
| 07 | | Message retries exhausted. |
| 08 | | Local (modem) communication has been lost. |
| | | |
| 09 | | Received a Master Link reset. |
| | | |
| 01 | 21 for Ethernet | Invalid Local IP address configured. |
| 02 | | Multi-Hop messaging cannot route request. |
| 03 | | Maximum connections used - no connections available. |
| 04 | | Invalid Target IP address or host name. |
| 05 | | Cannot communicate with the name server. |
| 06 | | Connection not completed before user-specified timeout. |
| 07 | | Connection timed out by the network. |
| 08 | | Connection refused by destination host. |
| 09 | | Connection was broken. |
| 0A | | Reply not received before user-specified timeout. |
| 0B | | No network buffer space available. |
| | | |
| | | |
| | | |
| XX | 22 for Xnet | Xnet TCP specific error codes. |

| MSG Error Code Mapping | | |
|---|---|---|
| **Internal Fail Code (Hex)** | | **Description** |
| **High byte** | **Low byte** | |
| **Identifier Code** | **Error Code group** | |
| XX | 23 for Modbus | Modbus TCP specific error codes. |
| | 24 for EtherNet/IP | EtherNet/IP specific error codes. |
| | | |
| | | |
| | | |
| XX | 25 for CIP Response | CIP status code returned by CIP reply. |
| 01 | 26 for Socket MSG | No socket. |
| 02 | | No buffer available. |
| 03 | | Max connection. |
| 04 | | Illegal sequence. |
| 05 | | Connection broken |
| 06 | | Address in use. |
| 07 | | DNS error. |
| 08 | | Force buffer return. |
| 10 | | Invalid parameter - send data size. |
| 11 | | Invalid parameter - service code. |
| 12 | | Invalid parameter - socket type. |
| 13 | | Invalid parameter - server type. |
| 14 | | Invalid parameter - type code. |
| 15 | | Invalid parameter - family. |
| 16 | | Invalid parameter - port. |
| 17 | | Invalid parameter - address. |
| 18 | | Invalid parameter - address length. |
| 19 | | Invalid parameter - data length. |
| 1A | | Invalid parameter - timeout. |
| 20 | | Socket error - create. |
| 21 | | Socket error - listen. |
| 22 | | Socket error - bind. |
| 23 | | Socket error - accept. |
| 24 | | Socket error - connect. |
| 25 | | Socket error - send. |

| MSG Error Code Mapping | | |
|---|---|---|
| **Internal Fail Code (Hex)** | | **Description** |
| **High byte** | **Low byte** | |
| **Identifier Code** | **Error Code group** | |
| 26 | 26 for Socket MSG | Socket error - receive. |
| 27 | | Socket error - unlisten. |
| 28 | | Socket error - unbind. |
| 29 | | Socket error - unaccept. |
| 2A | | Socket error - disconnect. |
| 2B | | Socket error - delete. |
| 30 | | Open connection timeout. |
| 31 | | Accept connection timeout. |
| 32 | | Read timeout. |
| 33 | | Write timeout. |
| 22 | | Socket error - bind. |
| 27 | | Socket error - unlisten. |
| 28 | | Socket error - unbind. |
| 29 | | Socket error - unaccept. |
| 2A | | Socket error - disconnect. |
| 2B | | Socket error - delete. |
| 30 | | Open connection timeout. |
| 31 | | Accept connection timeout. |
| 32 | | Read timeout. |
| 33 | | Write timeout. |
| 01 | 27 for SMTP | SMTP mail server IP Address not configured. |
| 02 | | To (destination) Address not configured or invalid. |
| 03 | | From (reply) Address not configured or invalid. |
| 04 | | Unable to connect to SMTP mail server. |
| 05 | | Communication error with SMTP server. |
| 06 | | Authentication required. |
| 07 | | Authentication failed. |
| | | |
| 10 | | SMTP Configuration does not exist. |
| | | |
| XX | 35 for Xbus MSG | Xbus messaging failed. |

| MSG Error Code Mapping | | |
|---|---|---|
| Internal Fail Code (Hex) | | Description |
| High byte | Low byte | |
| Identifier Code | Error Code group | |
| 01 | 41 | RP CFG, Invalid subtype. |
| 02 | | RP CFG, Invalid IP Address. |
| | | |
| 00 | 42 | RP_CFG, No Data Table defined or Incorrect RP Data Table defined. |
| 00 | 4E | Target node does not accept this type of MSG instruction. |
| 00 | 4F | Local communication channel is shutdown. |
| Application Layer Qualification<br>Bad MG Data Table parameter for building MSG | | |
| 01 | 51 | Channel does not exist. |
| 02 | | rw.byte.cmd is invalid. |
| 03 | | targetDataTable.type for PLC5 Read/Write is invalid. |
| 04 | | rw.byte.fnc is invalid. |
| 05 | | localAddr is not within user data space. |
| 06 | | numBytesRW is invalid. |
| 07 | | targetDataTable.subele is invalid. |
| 08 | | localAddr + numBytesRW - 2 is invalid. |
| 09 | | PLC type not supported. |
| 0A | | subtype is invalid. |
| 0B | | Improper number of elements specified to adequately fill a set of larger elements. |
| 0C | | Protocol not supported on channel. |
| 0D | | Source target mismatch. |
| 0E | | Supplemental Routing Path Error. |
| 0F | | Illegal service code. Bit 7 must be clear. |
| 10 | | Slot is invalid. |
| Link Layer Qualification<br>Bad parameter in command packet for current communications protocol | | |

| MSG Error Code Mapping | | |
|---|---|---|
| **Internal Fail Code (Hex)** | | **Description** |
| **High byte** | **Low byte** | |
| **Identifier Code** | **Error Code group** | |
| 01 | 61 | Format Unsupported. |
| 02 | | Current communications protocol does not support MSG initiation. |
| 03 | | Channel is shutdown or reconfiguration in progress. |
| 04 | | Current communications protocol does not support the packet type the MSG command is using.<br>DF1 FD - Not SDA or SDN Broadcast.<br>DF1 HD - Not SDA.<br>DH485 - Not SDA or SDN Broadcast. |
| 05 | | Local Target Node conflict.<br>DF1 FD - Not applicable.<br>DF1 HD - target node equals channel's node number.<br>DH485 - local packet and target node equals channel's node number or remote packet and local bridge node equals channel's node number. |
| 07 | | Source node is not valid (not within the range 0-254). |
| 08 | | Current communications protocol does not support the MSG address type<br>DF1 FD & DF1 HD - remote MSG not supported.<br>DH485 - remote broadcast MSG not supported. |
| 09 | | Amount of data is too large to be supported by link layer or expansion I/O communications module. |
| 00 | 62 | Unexpected link layer error. |
| Network Layer Qualification<br>Bad network address | | |
| 01 | 71 | Internet-to-Internet Remote MSG (remote bridge non-zero) and the targetNode is not valid (not within the range 0-254). |
| 02 | | Internet-to-NonInternet Remote MSG (remote bridge zero) and the targetNode is not valid (not within the range 0-254). |
| 03 | | Internet-to-NonInternet Remote MSG (remote bridge zero) and the remoteBridgeNode is not valid (not within the range 0-254). |
| 00 | 72 | Unexpected network error. |
| Application Layer Qualification<br>MSG Reply Failures | | |
| STS | 81 | Reply received with error. STS not 0x00, nor 0xF0. |
| EXT STS | 82 | Reply received with error. STS=0xF0, EXT STS. |
| FNC | 83 | Function code unknown. |

| MSG Error Code Mapping | | |
|---|---|---|
| **Internal Fail Code (Hex)** | | **Description** |
| **High byte** | **Low byte** | |
| **Identifier Code** | **Error Code group** | |
| 01 | 88 | Insufficient data. |
| 02 | | Improper Amount of data. |
| 03 | | Command code mismatched. |
| 04 | | Packet type mismatched. |
| 05 | | Rejected SDN packet. |
| 06 | | Bad data type. |
| 07 | | Buffer format code mismatched. |
| 08 | | Service code mismatched. |
| 09 | | Modbus Write Reply - bytes 2-5 don't match outgoing MSG. |
| 0A | | NX-Plus Write with fail. |

Application Layer Qualification
MSG Timing Failures

| MSG Error Code Mapping | | |
|---|---|---|
| **Internal Fail Code (Hex)** | | **Description** |
| **High byte** | **Low byte** | |
| **Identifier Code** | **Error Code group** | |
| 01 | 91 | MSG waiting for buffer (MSG timed out while on overflow queue) after timeout. |
| 02 | | MSG waiting for buffer (MSG timed out while on overflow queue) before timeout. |
| 03 | | MSG waiting for link layer (MSG timed out while link layer transmitting the command or waiting for target to acknowledge receipt), command buffer #0. |
| 13 | | MSG waiting for link layer (MSG timed out while link layer transmitting the command or waiting for target to acknowledge receipt), command buffer #1. |
| 23 | | MSG waiting for link layer (MSG timed out while link layer transmitting the command or waiting for target to acknowledge receipt), command buffer #2. |
| 33 | | MSG waiting for link layer (MSG timed out while link layer transmitting the command or waiting for target to acknowledge receipt), command buffer #3. |
| 04 | | MSG waiting for reply (MSG timed out waiting for target to reply), command buffer #0. |
| 14 | | MSG waiting for reply (MSG timed out waiting for target to reply), command buffer #1. |
| 24 | | MSG waiting for reply (MSG timed out waiting for target to reply), command buffer #2. |
| 34 | | MSG waiting for reply (MSG timed out waiting for target to reply), command buffer #3. |
| 05 | | MSG waiting in transmit queue (MSG timed out while waiting for transmission by link layer), command buffer #0. |
| 15 | | MSG waiting in transmit queue (MSG timed out while waiting for transmission by link layer), command buffer #1. |
| 25 | | MSG waiting in transmit queue (MSG timed out while waiting for transmission by link layer), command buffer #2. |
| 35 | | MSG waiting in transmit queue (MSG timed out while waiting for transmission by link layer), command buffer #3. |

Application Layer Qualification
Communications Protocol Reset - Active MSGs flushed.

| MSG Error Code Mapping | | |
|---|---|---|
| **Internal Fail Code (Hex)** | | **Description** |
| **High byte** | **Low byte** | |
| **Identifier Code** | **Error Code group** | |
| 01 | A1 | MSG was on confirmation queue. |
| 02 | | MSG was waiting for reply, command buffer #0. |
| 12 | | MSG was waiting for reply, command buffer #1. |
| 22 | | MSG was waiting for reply, command buffer #2. |
| 32 | | MSG was waiting for reply, command buffer #3. |
| 03 | | MSG was on overflow queue. |
| 04 | | MSG was on transmit queue. |

# ASCII Character Set

## ASCII Character Set

| Column 1 | | | | | Column 2 | | | | Column 3 | | | | Column 4 | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Ctrl | DEC | HEX | OCT | ASC | DEC | HEX | OCT | ASC | DEC | HEX | OCT | ASC | DEC | HEX | OCT | ASC |
| ^@ | 00 | 00 | 000 | NUL | 32 | 20 | 040 | SP | 64 | 40 | 100 | @ | 96 | 60 | 140 | \ |
| ^A | 01 | 01 | 001 | SOH | 33 | 21 | 041 | ! | 65 | 41 | 101 | A | 97 | 61 | 141 | a |
| ^B | 02 | 02 | 002 | STX | 34 | 22 | 042 | ” | 66 | 42 | 102 | B | 98 | 62 | 142 | b |
| ^C | 03 | 03 | 003 | ETX | 35 | 23 | 043 | # | 67 | 43 | 103 | C | 99 | 63 | 143 | c |
| ^D | 04 | 04 | 004 | EOT | 36 | 24 | 044 | $ | 68 | 44 | 104 | D | 100 | 64 | 144 | d |
| ^E | 05 | 05 | 005 | ENQ | 37 | 25 | 045 | % | 69 | 45 | 105 | E | 101 | 65 | 145 | e |
| ^F | 06 | 06 | 006 | ACK | 38 | 26 | 046 | & | 70 | 46 | 106 | F | 102 | 66 | 146 | f |
| ^G | 07 | 07 | 007 | BEL | 39 | 27 | 047 | ’ | 71 | 17 | 107 | G | 103 | 67 | 147 | g |
| ^H | 08 | 08 | 010 | BS | 40 | 28 | 050 | ( | 72 | 48 | 110 | H | 104 | 68 | 150 | h |
| ^I | 09 | 09 | 011 | HT | 41 | 29 | 051 | ) | 73 | 49 | 111 | I | 105 | 69 | 151 | i |
| ^J | 10 | 0A | 012 | LF | 42 | 2A | 052 | * | 74 | 4A | 112 | J | 106 | 6A | 152 | j |
| ^K | 11 | 0B | 013 | VT | 43 | 2B | 053 | + | 75 | 4B | 113 | K | 107 | 6B | 153 | k |
| ^L | 12 | 0C | 014 | FF | 44 | 2C | 054 | ’ | 76 | 4C | 114 | L | 108 | 6C | 154 | l |
| ^M | 13 | 0D | 015 | CR | 45 | 2D | 055 | - | 77 | 4D | 115 | M | 109 | 6D | 155 | m |
| ^N | 14 | 0E | 016 | SO | 46 | 2D | 056 | . | 78 | 4E | 116 | N | 110 | 6E | 156 | n |
| ^O | 15 | 0F | 017 | SI | 47 | 2F | 057 | / | 79 | 4F | 117 | O | 111 | 6F | 157 | o |
| ^P | 16 | 10 | 020 | SLE | 48 | 30 | 060 | 0 | 80 | 50 | 120 | P | 112 | 70 | 160 | p |
| ^Q | 17 | 11 | 021 | DC1 | 49 | 31 | 061 | 1 | 81 | 51 | 121 | Q | 113 | 71 | 161 | q |
| ^R | 18 | 12 | 022 | DC2 | 50 | 32 | 062 | 2 | 82 | 52 | 122 | R | 114 | 72 | 162 | r |
| ^S | 19 | 13 | 023 | DC3 | 51 | 33 | 063 | 3 | 83 | 53 | 123 | S | 115 | 73 | 163 | s |
| ^T | 20 | 14 | 024 | DC4 | 52 | 34 | 064 | 4 | 84 | 54 | 124 | T | 116 | 74 | 164 | t |
| ^U | 21 | 15 | 025 | NAK | 53 | 35 | 065 | 5 | 85 | 55 | 125 | U | 117 | 75 | 165 | u |
| ^V | 22 | 16 | 026 | SYB | 54 | 36 | 066 | 6 | 86 | 56 | 126 | V | 118 | 76 | 166 | v |
| ^W | 23 | 17 | 027 | ETB | 55 | 37 | 067 | 7 | 87 | 57 | 127 | W | 119 | 77 | 167 | w |
| ^X | 24 | 18 | 030 | CAN | 56 | 38 | 070 | 8 | 88 | 58 | 130 | X | 120 | 78 | 170 | x |
| ^Y | 25 | 19 | 031 | EM | 57 | 39 | 071 | 9 | 89 | 59 | 131 | Y | 121 | 79 | 171 | y |
| ^Z | 26 | 1A | 032 | SUB | 58 | 3A | 072 | : | 90 | 5A | 132 | Z | 122 | 7A | 172 | z |
| ^[ | 27 | 1B | 033 | ESC | 59 | 3B | 073 | ; | 91 | 5B | 133 | [ | 123 | 7B | 173 | { |

| Column 1 | | | | | Column 2 | | | | Column 3 | | | | Column 4 | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Ctrl | DEC | HEX | OCT | ASC | DEC | HEX | OCT | ASC | DEC | HEX | OCT | ASC | DEC | HEX | OCT | ASC |
| ^\ | 28 | 1C | 034 | FS | 60 | 3C | 074 | < | 92 | 5C | 134 | \ | 124 | 7C | 174 | \| |
| ^] | 29 | 1D | 035 | GS | 61 | 3D | 075 | = | 93 | 5D | 135 | ] | 125 | 7D | 175 | } |
| ^^ | 30 | 1E | 036 | RS | 62 | 3E | 076 | > | 94 | 5E | 136 | ^ | 126 | 7E | 176 | ~ |
| ^_ | 31 | 1F | 037 | US | 63 | 3F | 077 | ? | 95 | 5F | 137 | _ | 127 | 7F | 177 | DEL |

Publication number :X8 -RM001A-EN-September 2012

**RS Automation Co., Ltd.**
**www.rsautomation.biz**

RS Automation Building, 348-2, Jinwi Industrial Complex, Cheongho-ri, Jinwi-myeon,
Pyeongtaek-si, Gyeonggi-do, Korea, zip code : 451-862

T 82-31-685-9300, F 82-31-685-9500

RS Automation Global Business Support
rsagbs@rsautomation.biz

韩国京畿道平泽市振威面清湖里振威工业园348-2RS自动化大厦 邮编：451-862

T 82-31-685-9300, F 82-31-685-9500

RS自动化全球商户支持
rsagbs@rsautomation.biz