Fuji Electric

WSZ-Series Controller

User's Manual - I

Hardware & Instruction

Preface, Contents

MEHT322a

Attention related to safety (Read carefully before application)

◎ For the purpose of ensuring your personal safety and protecting this product and its peripheral equipment, read this manual regarding the safety functions carefully before installation and operation of the WSZ controller. The three safety categories of this manual are classified into Danger, Warning, Caution grades according to the danger level and are preceded by the " ! " symbol. Following is their description:

Indicates casualty or serious damage or property loss will result if the correct instruction was not followed.

Casualty or serious damage or property loss may result if the correct instruction was not followed.

Indicates minor damage or property loss will result if the correct instruction was not followed.

◎ This manual is a guideline for qualified personnel on how to install the WSZ controller correctly and use it safely. The qualified personnel stated here means professional electromechanical engineering personnel who is familiar with safety specifications and methods of grounding, circuit wiring, peripheral equipment system etc. and possesses practical experience.

⊙ Keep in mind before using the controller

Abnormality in the external power supply or failure of the controller itself will result in the controller or the complete system emerging in an unsafe status, and may cause unpredictable actions. Such actions may cause human injury, death, or serious damage of the unit itself. Thus, please design a separate external safety protection circuit, such as emergency stop circuit, machine replacement device, or redundant safety protection circuit in your application with the following safety considerations:

1. Emergency stop circuit, safety protection circuit, motor positive/reverse interlock circuit, upper/lower limit destruction prevention circuit of position control, etc. These circuits should be hardwired and external to the controller.
2. The controller is unable to detect the abnormality of the input signal circuit (such as overload or interrupted controller input circuit. The controller interprets them as OFF, then erroneous output may result in the controller and may cause major safety problems, thus external detection and protection circuits should be provided in addition to the controller.
3. The output components of the controller such as transistor, relay are possible to cause permanent ON or OFF and resulting in serious accidents, thus protection by additional external circuits or mechanisms is necessary for the output points to avoid major safety problems.

WSZ-Controller User's Manual Ⅰ

【Hardware】

Contents

Chapter 1: Introduction of FUJI WSZ Series Controller

Chapter 2: System Architecture

Chapter 3: Expansion of WSZ Controller

Chapter 4: Installation Guide

Chapter 5: Power Supply Wiring, Power Consumption Calculation, and Power Sequence Requirements

Chapter 6: Digital Input (DI) Circuit

Chapter 7: Digital Output (DO) Circuit

Chapter 8: Test Run, Monitoring and Maintenance

WSZ-Controller User's Manual Ⅰ
【Instruction】

Contents

Chapter 5: Descriptions of Function Instructions

Chapter 6: Basic Function Instructions

Chapter 7: Advanced Function Instructions

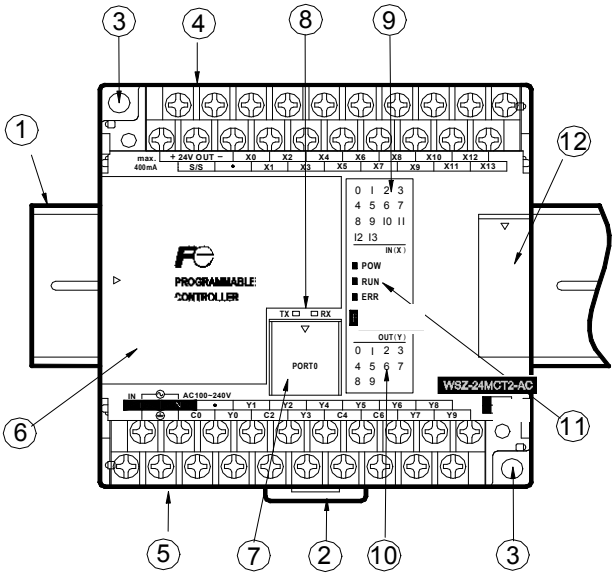
Chapter 8: Step Instruction Description

Chapter 1 Introduction of WSZ Series Controller

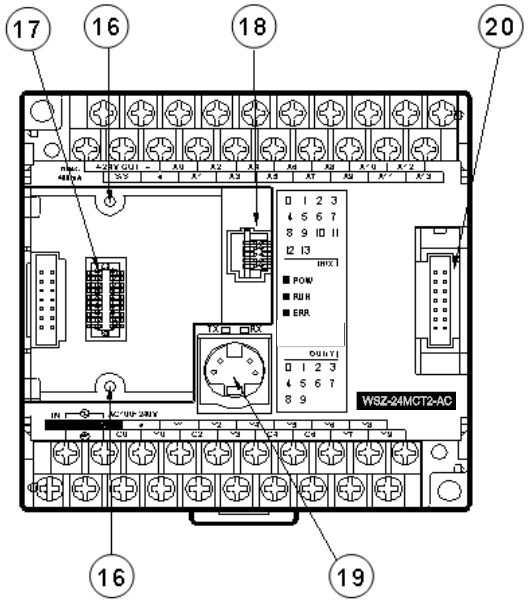
The FUJI WSZ Series controller is a new generation of controller equipped with excellent functions comparable to medium or large controller, with up to three communication ports. The maximum I/O numbers are 256 points for Digital Input (DI) and Digital Output (DO), 64 words for Numeric Input (NI) and Numeric Output (NO). The Main Units of WSZ are available in MC type (High-Performance Type). With the combination of I/O point ranges from 24 to 60, a total of 4 models are available. 14DI/10DO (WSZ-24XYT-AC), 16DO (WSZ-16YR), 6ch AI (WSZ-6AD), 2ch AO (WSZ-2DA), 4ch AI+2ch AO (WSZ-4A2D), 16ch temperature measurement (WSZ-16TC) models are available for Expansion Units/Modules. With interface options in RS232 and RS485, the communication peripherals are available with one board. The various models are described in the following:

1.1 Appearance of Main Unit

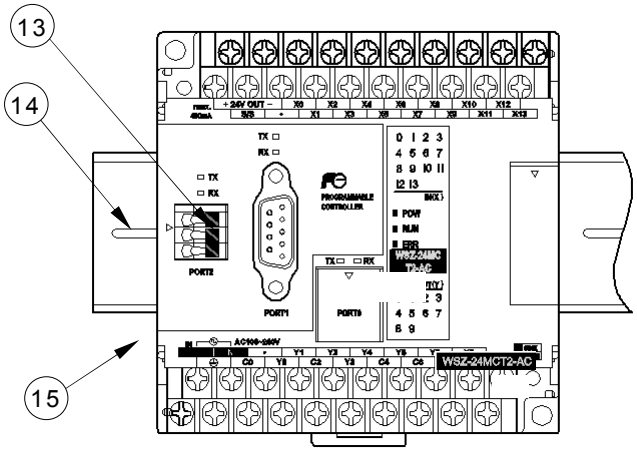
All the Main Units of WSZ controller have the same physical structure. The only difference is the case width. There are three different case sizes, which are 90mm, 130mm, and 175mm. The figure below will use the Main Unit case of the WSZ-24MCT2-AC as an example for illustration:



(Front view without Communication Board)



(Front view with cover plate removed)



(Front view with CB-25 Board installed)

① 35mm-width DIN RAIL
② DIN RAIL tab
③ Hole for screw fixation（ψ4.5x2）
④ Terminal of 24VDC power output and digital input (Pitch 7.62mm)
⑤ Terminal of main power input and digital output (Pitch 7.62mm)
⑥ Standard cover plate (Without communication board)
⑦ Cover plate of build-in communication port (Port0)

⑧ Indicators for transmit (TX) and receive (RX) status of build-in communication port (Port0)

⑨ Indicators for Digital Input (Xn)

⑩ Indicators for Digital Output (Yn)

⑪ Indicators for system status (POW, RUN, ERR)

⑫ I/O output expansion header cover [units of 20 points or beyond only], with esthetic purpose and capable of securing expansion cable

⑬ RS232 communication port (Port1)

⑭ RS485 communication port (Port2)

⑮ WSZ-CB25 Communication Board (CB)

⑯ Screw holes of communication board

⑰ Connector for communication board

⑱ Connector for Memory Pack

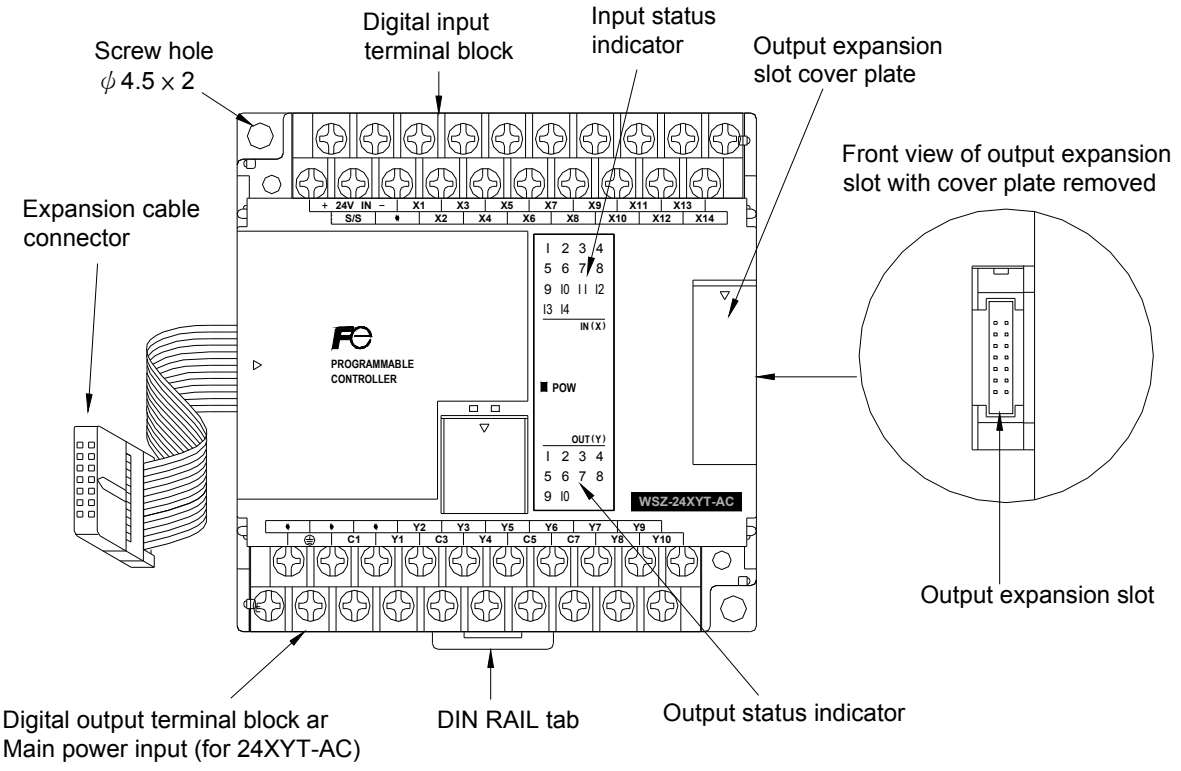
⑲ Connector for build-in RS232 communication port (Port0)

⑳ I/O output expansion header, for connecting with cables from expansion units/modules

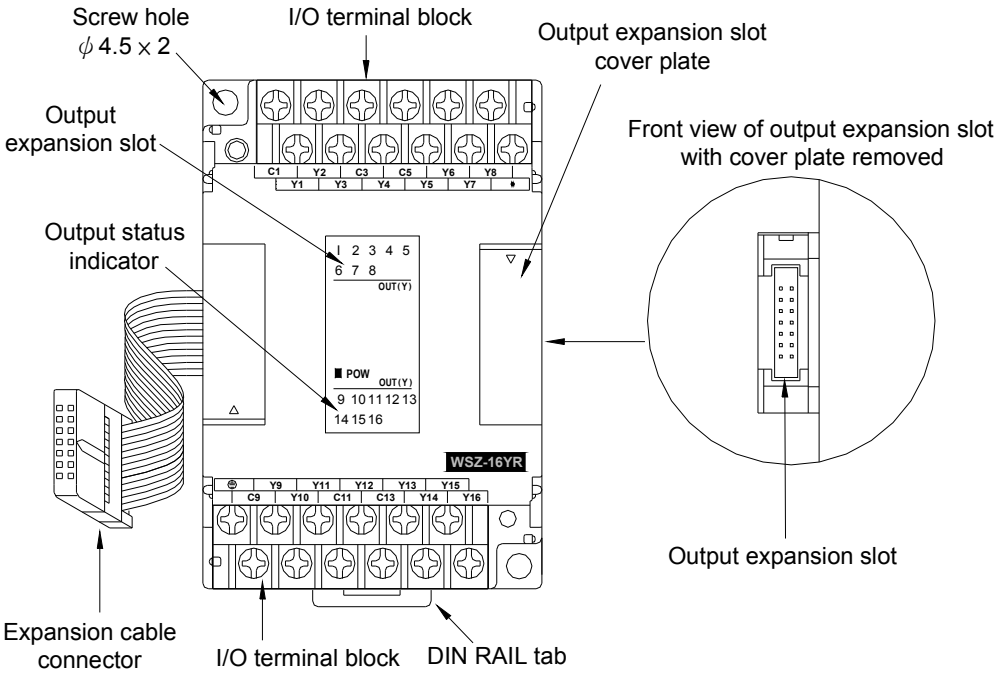
1.2 Appearance of Expansion Unit/Module

There are two types of cases for expansion unit/modules. One type uses the same case as main unit that of the 90mm, while the other have thinner 40mm case, which are for expansion modules. All expansion cables (left) of expansion units/modules are flat ribbon cables (5cm long), which were soldered directly on the PCB, and the expansion header (right) is a 14Pin Header, with this to connect the right adjacent expansion units/modules. In the following, each of the two types of expansion units/modules is described as an example:

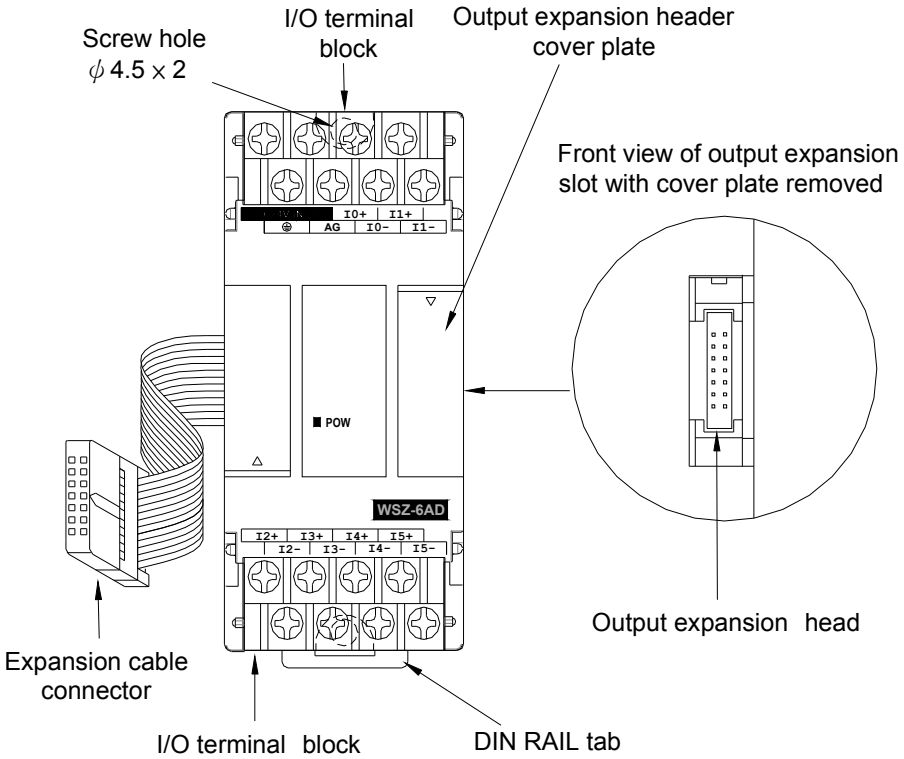
- Expansion unit with 90mm width case: [-24XYT-AC, -16TC]



- Expansion module with 60mm width case: [-16YR]



- Expansion module with 40mm width case: [-6AD, -2DA, -4A2D]



1.3 List of WSZ Models

Item Name	Model	Specifications
Main Units	WSZ-24MCT2-AC	100~240VAC power supply ; 14 points 24VDC digital input (4 points 200kHz high speed, 4 points 20kHz medium speed, 6 points medium speed total 5kHz); 10 points transistor sink output (4 points 200kHz high speed, 4 points 20kHz medium speed); RS232 port ; build-in RTC; detachable terminal block
	WSZ-32MCT2-AC	100~240VAC power supply ; 20 points 24VDC digital input (6 points 200kHz high speed, 2 points 20kHz medium speed, 8 points medium speed total 5kHz); 12 points transistor sink output (6 points 200kHz high speed, 2 points 20kHz medium speed); RS232 port ; build-in RTC; detachable terminal block
	WSZ-40MCT2-AC	100~240VAC power supply ; 24 points 24VDC digital input (6 points 200kHz high speed, 2 points 20kHz medium speed, 8 points medium speed total 5kHz); 16 points transistor sink output (6 points 200kHz high speed, 2 points 20kHz medium speed); RS232 port ; build-in RTC; detachable terminal block
	WSZ-60MCT2-AC	100~240VAC power supply ; 36 points 24VDC digital input (8 points 200kHz high speed, 8 points medium speed total 5kHz); 24 points transistor sink output (8 points 200kHz high speed); RS232 port ; build-in RTC; detachable terminal block
Digital(DIO) Expansion Unit	WSZ-24XYT-AC	14 points 24VDC digital input, 10 points transistor sink output, build-in power supply
Digital(DIO) Expansion Module	WSZ-16YR	16 points relay output
Analog Expansion Modules	WSZ-2DA	2 channels, 14-bit analog output module (-10V~0V~+10V or -20mA~0mA~+20mA)
	WSZ-6AD	6 channels, 14-bit analog input module (-10V~0V~+10V or -20mA~0mA~+20mA)
	WSZ-4A2D	4 channels, 14-bit analog input (same specification as WSZ-6AD) + 2 channels, 14-bit analog output (same specification as WSZ-2DA) combo module
Temperature Measurement Module	WSZ-16TC	16 channels, thermocouple temperature input module with 0.1℃ resolution
Communication Expansion Board	WSZ-CB25	1 port RS232 (Port 1) + 1 port RS485 (Port 2) communication board
Communication Cables	WSZ-232P0-9F-150	WSZ main unit port 0 RS232 to 9 pins female D-Sub communication cable, length 150cm
	WSZ-232P0-9M-400	WSZ main unit port 0 RS232 to 9 pins male D-Sub communication cable, length 400cm
Memory Pack	WSZ-PACK	WSZ program memory pack with 20K words program, 20K words register, write protection switch
Programming Device	Winproladder for WSZ	WSZ Winproladder Programming software for Windows XP and 7 (32 bits)

1.4 Specifications of Main Unit

"*" Default Settings

Item					Specification	Note
Execution Speed					0.33us per Sequence Command	
Space of Control Program					20K Words	
Program Memory					SRAM + Lithium Battery for Back-up	Option: FLASH ROM
Sequence Command					36	
Application Command					326 (126 types)	Include Derived Commands
Flow Chart (SFC) Command					4	
Single Point《BIT Status》	X	Input Contact (DI)			X0~X255 (256)	Corresponding to External Digital Input Point
	Y	Output Relay (DO)			Y0~Y255 (256)	Corresponding to External Digital Output Point
	TR	Temporary Relay			TR0~TR39 (40)	
	M	Internal Relay		Non-Retentive	M0~M799 (800)*	Can be configured as retentive type
					M1400~M1911 (512)	
				Retentive	M800~M1399 (600)*	Can be configured as non-retentive type
		Special Relay			M1912~M2001 (90)	
	S	Step Relay		Non-Retentive	S0~S499 (500)*	S20~S499 can be configured as retentive type
				Retentive	S500~S999 (500)*	Can be configured as non-retentive type
	T	Timer "Time Up" Status Contact			T0~T255 (256)	
	C	Counter "Count Up" Status Contact			C0~C255 (256)	
Register《WORD Data》	TMR	Current Time Value Register	0.01s Time base		T0~T49 (50)*	T0~T255 Numbers for each time base can be flexibly adjusted
			0.1s Time base		T50~T199 (150)*	
			1s Time base		T200~T255 (56)*	
	CTR	Current Counter Value Register	16-Bit	Retentive	C0~C139 (140)*	Can be configured as non-retentive type
				Non-Retentive	C140~C199 (60)*	Can be configured as retentive type
			32-Bit	Retentive	C200~C239 (40)*	Can be configured as non-retentive type
				Non-Retentive	C240~C255 (16)*	Can be configured as retentive type
	HR DR	Data Register		Retentive	R0~R2999 (3000)*	Can be configured as non-retentive type
					D0~D3999 (4000)	
				Non-Retentive	R3000~R3839 (840)*	Can be configured as retentive type
	HR ROR			Retentive	R5000~R8071 (3072)*	When not configured as ROR, it can serve as normal register (for read/write)
				Read-Only Register	R5000~R8071 can be configured as ROR, default setting is (0)*	ROR is stored in special ROR area and not consume program space
				File Register	F0~F8191 (8192)*	Must save/retrieved via special commands
	IR	Input register			R3840~R3903 (64)	Corresponding to external numeric input
	OR	Output Register			R3904~R3967 (64)	Corresponding to external numeric output
	SR	Special System Register			R3968~R4167 (197) D4000~D4095 (96)	Except R4152~4154
	〈Special Register〉	0.1ms High Speed Timer register			R4152~R4154 (3)	
		High Speed Counter Register	Hardware(4 sets)		DR4096~DR4110 (4x4)	
			Software (4 sets)		DR4112~DR4126 (4x4)	
		Real Time Calendar Register			R4128 (sec) / R4129 (min) / R4130 (hour) / R4131 (day) / R4132 (month) / R4133 (year) / R4134 (week)	Accuracy: ±20 seconds per day
	XR	Index Register			V, Z (2), P0~P9 (10)	
Interrupt Control		External Interrupt Control			32 (16 points input positive/negative edges)	
		Internal Interrupt Control			8 (1, 2, 3, 4, 5, 10, 50, 100ms)	
0.1ms High Speed Timer (HST)					1 (16bits), 4 (32bits, derived from HHSC)	

<table>
<tr><td rowspan="5">High Speed Counter</td><td rowspan="3">Hardware High Speed Counter (HHSC)/32 Points</td><td rowspan="2">Channels</td><td colspan="2">Total 4</td><td rowspan="5">• Total number of HHSC and SHSC is 8
• HHSC can change into High Speed Timer with 32 bits/0.1ms Time base</td></tr>
<tr><td>Up to 4</td><td>Up to 2</td></tr>
<tr><td>Counting frequency</td><td>Up to 200KHz (single-end input)</td><td>Up to 20KHz (single-end input)</td></tr>
<tr><td>Counting modes</td><td colspan="2">8 modes (U/D, U/Dx2, P/R, P/R x2, A/B, A/B x2, A/B x3, A/B x4)</td></tr>
<tr><td rowspan="3">Software High Speed Counter (SHSC)/32 Points</td><td>Channels</td><td colspan="2">Up to 4</td></tr>
<tr><td>Counting modes</td><td colspan="2">3 modes (U/D, P/R, A/B)</td></tr>
<tr><td>Counting frequency</td><td colspan="2">Maximum sum up to 5KHz</td></tr>
<tr><td rowspan="3">Communication Interface</td><td colspan="2">Port0 (RS232)</td><td colspan="2">Communication Speed 4.8Kbps~921.6Kbps (9.6Kbps)*</td><td></td></tr>
<tr><td colspan="2">Port1 (RS232), Port2 (RS485)</td><td colspan="2">Communication Speed 4.8Kbps~921.6Kbps (9.6Kbps)*</td><td>Port1~2 talk original or Modbus RTU Master/Slave Communication Protocol</td></tr>
<tr><td colspan="2">Maximum Connections</td><td colspan="2">254</td><td></td></tr>
<tr><td rowspan="4">NC Positioning Output (PSO)</td><td rowspan="2">Number of Axes</td><td colspan="2">Total 4</td><td rowspan="2"></td></tr>
<tr><td>Up to 4</td><td>Up to 2</td></tr>
<tr><td>Output Frequency</td><td colspan="2">200KHz single-end transistor output</td><td>20KHz single-end transistor output</td></tr>
<tr><td>Output Pulse Mode</td><td colspan="2">3 modes (U/D, P/R, A/B)</td><td></td></tr>
<tr><td>Positioning Language</td><td colspan="2">Special Positioning Programming Language</td><td></td></tr>
<tr><td rowspan="2">HSPWM Output</td><td>Number of Points</td><td colspan="2">Up to 4</td><td></td></tr>
<tr><td>Output Frequency</td><td colspan="2">72Hz~18.432KHz (with 0.1% resolution)
720Hz~184.32KHz (with 1%resolution)</td><td></td></tr>
<tr><td colspan="2" rowspan="4">Captured input</td><td rowspan="2">Points</td><td colspan="2">Max. 36 points (all of main units have the feature)</td><td></td></tr>
<tr><td colspan="2">> 10 μs (super high speed/high speed input)</td><td></td></tr>
<tr><td rowspan="2">Captured pulse width</td><td colspan="2">> 47 μs (medium speed input)</td><td></td></tr>
<tr><td colspan="2">> 470 μs (mid/low speed input)</td><td></td></tr>
<tr><td colspan="2" rowspan="3">Setting of Digital Filter</td><td rowspan="2">X0~X15</td><td colspan="2">Frequency 14KHz~1.8MHz</td><td>Chosen by frequency at high frequencies</td></tr>
<tr><td colspan="2">Time constant 0~1.5ms/0~15ms, adjustable by step of 0.1ms/1ms</td><td>Chosen by time constant at low frequencies</td></tr>
<tr><td>X16~X35</td><td colspan="2">Time constant 1ms~15ms, adjustable by step of 1ms</td><td></td></tr>
<tr><td colspan="3">Maximum expandable modules</td><td colspan="2">32</td><td></td></tr>
</table>

1.5 Environmental Specifications

Item			Specification	Note
Operation Ambient Temperature	Enclosure equipment	Minimum	5°C	Permanent Installation
		Maximum	40°C	
	Open equipment	Minimum	5°C	
		Maximum	55°C	
Storage Temperature			-25°C~+70°C	
Relative Humidity (non-condensing, RH-2)			5%~95%	
Pollution Level			Degree II	
Corrosion Resistance			Base on IEC-68 Standard	
Altitude			≦2000m	
Vibration	Fixated by DIN RAIL		0.5G, for 2 hours each along the 3 axes	
	Secured by screws		2G, for 2 hours each along the 3axes	
Shock			10G, 3 times each along the 3 axes	
Noise Suppression			1500Vp-p, width 1us	
Withstand Voltage			1500VAC, 1 minute	L, N to any terminal

⚠ Warning

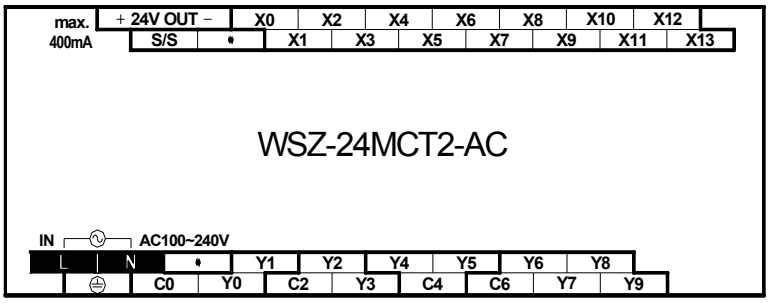
The listed environmental specifications are for WSZ controller under normal operation. Any operation in not conform to above conditions should be consulted with FUJI.

1.6 Connection Diagrams of Various Models

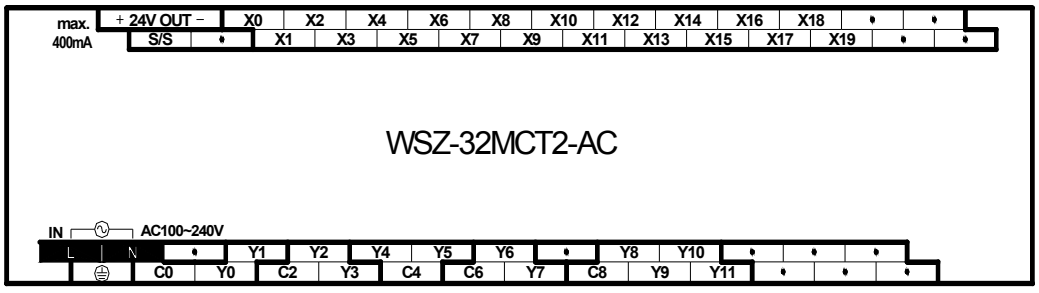
1.6.1 Main Unit

[7.62mm Terminal Block, detachable in model]

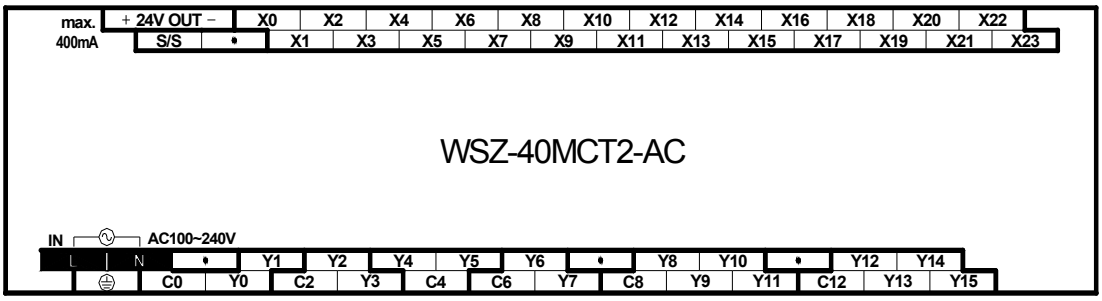
- 24 point digital I/O main unit (14 points IN, 10 points OUT)



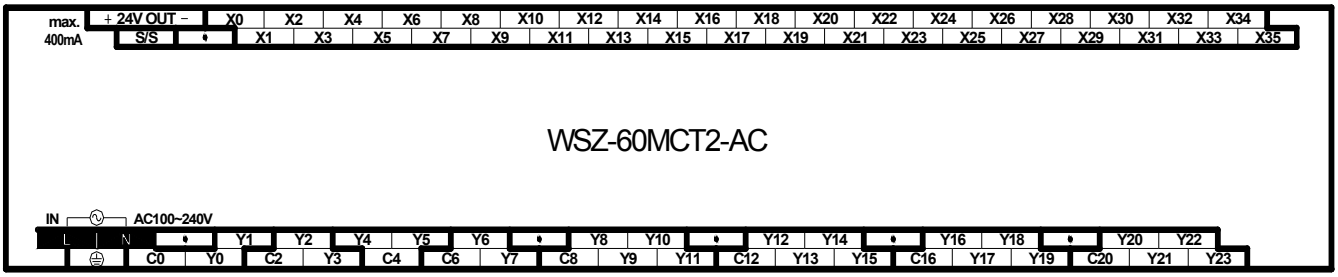
- 32 point digital I/O main unit (20 points IN, 12 points OUT)



- 40 point digital I/O main unit (24 points IN, 16 points OUT)

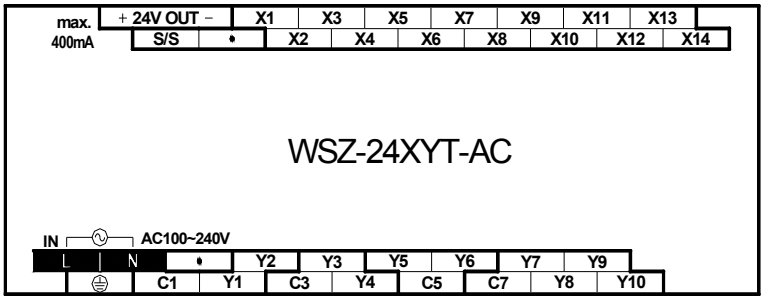


- 60 point digital I/O main unit (36 points IN, 24 points OUT)



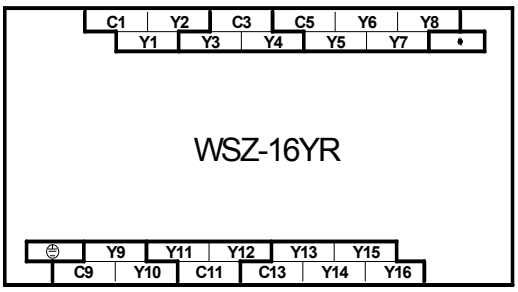
1.6.2 Digital I/O Expansion Unit [7.62mm fixed terminal block]

- 24 point digital I/O expansion unit (14 points IN, 10 points OUT)



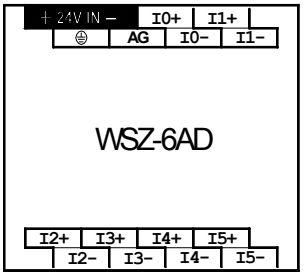
1.6.3 Digital I/O Expansion Module [7.62mm fixed terminal block]

- 16 point digital I/O expansion module (16 points OUT)

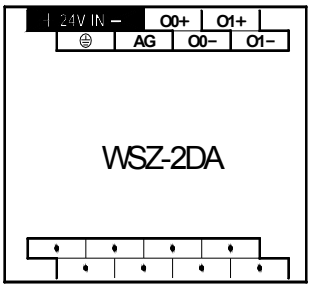


1.6.4 Analog I/O Expansion Module [7.62mm fixed terminal block]

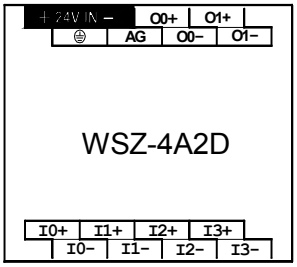
- 6 channel A/D analog input module



- 2 channel D/A output module

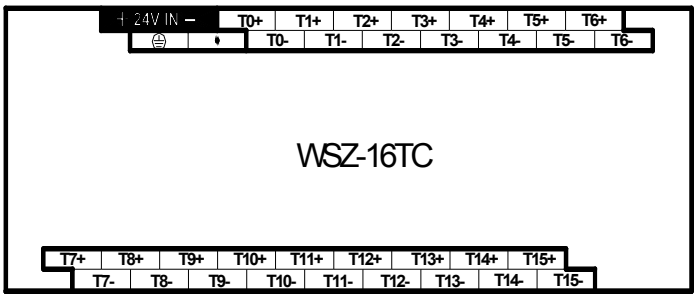


- 4 channel A/D input, 2 channel D/A output module



1.6.5 Temperature Input Module [7.62mm fixed terminal block]

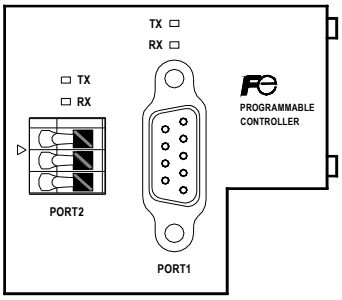
- 16 channel thermocouple input module



1.6.6 Communication Board (CB)

[DB9F/3Pin plug able terminal block](Below is outlook of CB and the corresponding cover plate)

- 1 RS232 + 1 RS485 ports

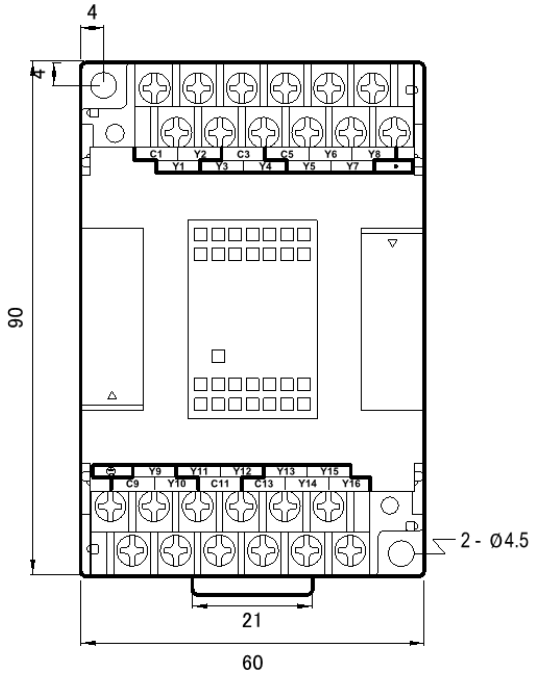


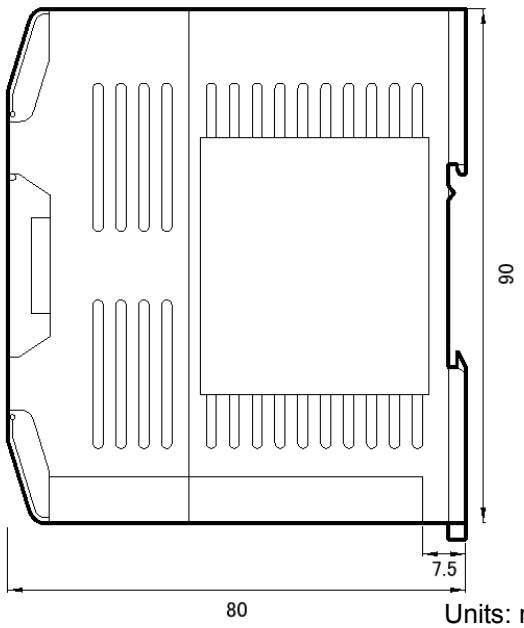
WSZ-CB25

1.7 Drawings with External Dimensions

(1) Outlook I:

Expansion Module: WSZ-16YR



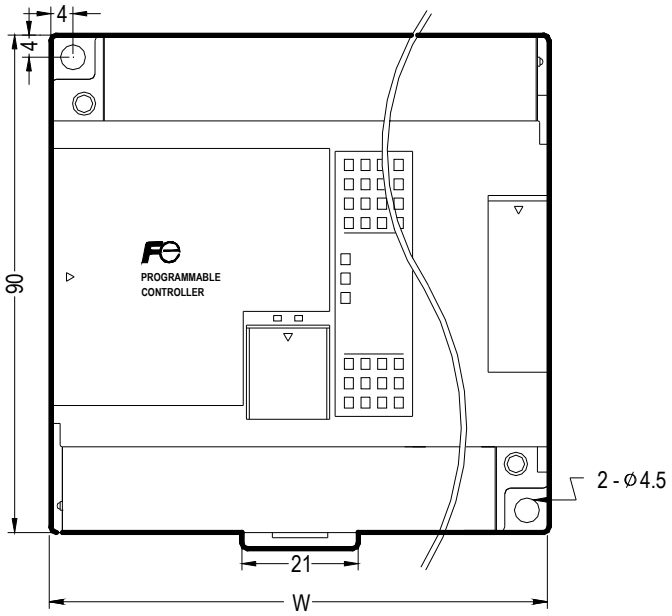


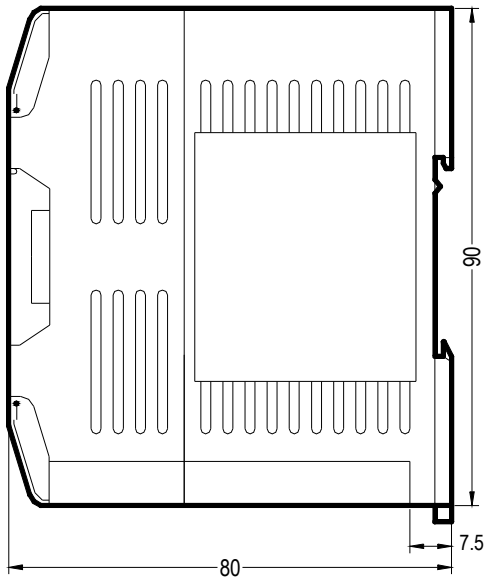
Units: mm

(2) Outlook II:

Main Unit: WSZ-24MCT2-AC, WSZ-32MCT2-AC, WSZ-40MCT2-AC, WSZ-60MCT2-AC

Expansion Unit/Module: WSZ-24XYT-AC, WSZ-16TC



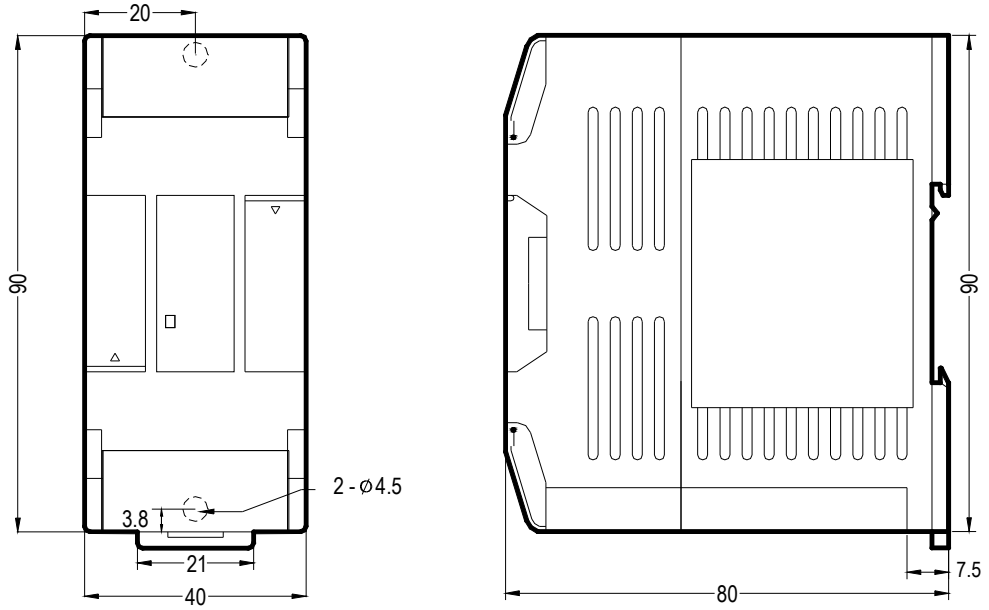


Units: mm

W	Model
90mm	WSZ-24MCT2-AC, WSZ-24XYT-AC, WSZ-16TC
130mm	WSZ-32MCT2-AC, WSZ-40MCT2-AC
175mm	WSZ-60MCT2-AC

(3) Outlook III:

Expansion Module: WSZ-6AD, WSZ-2DA, WSZ-4A2D

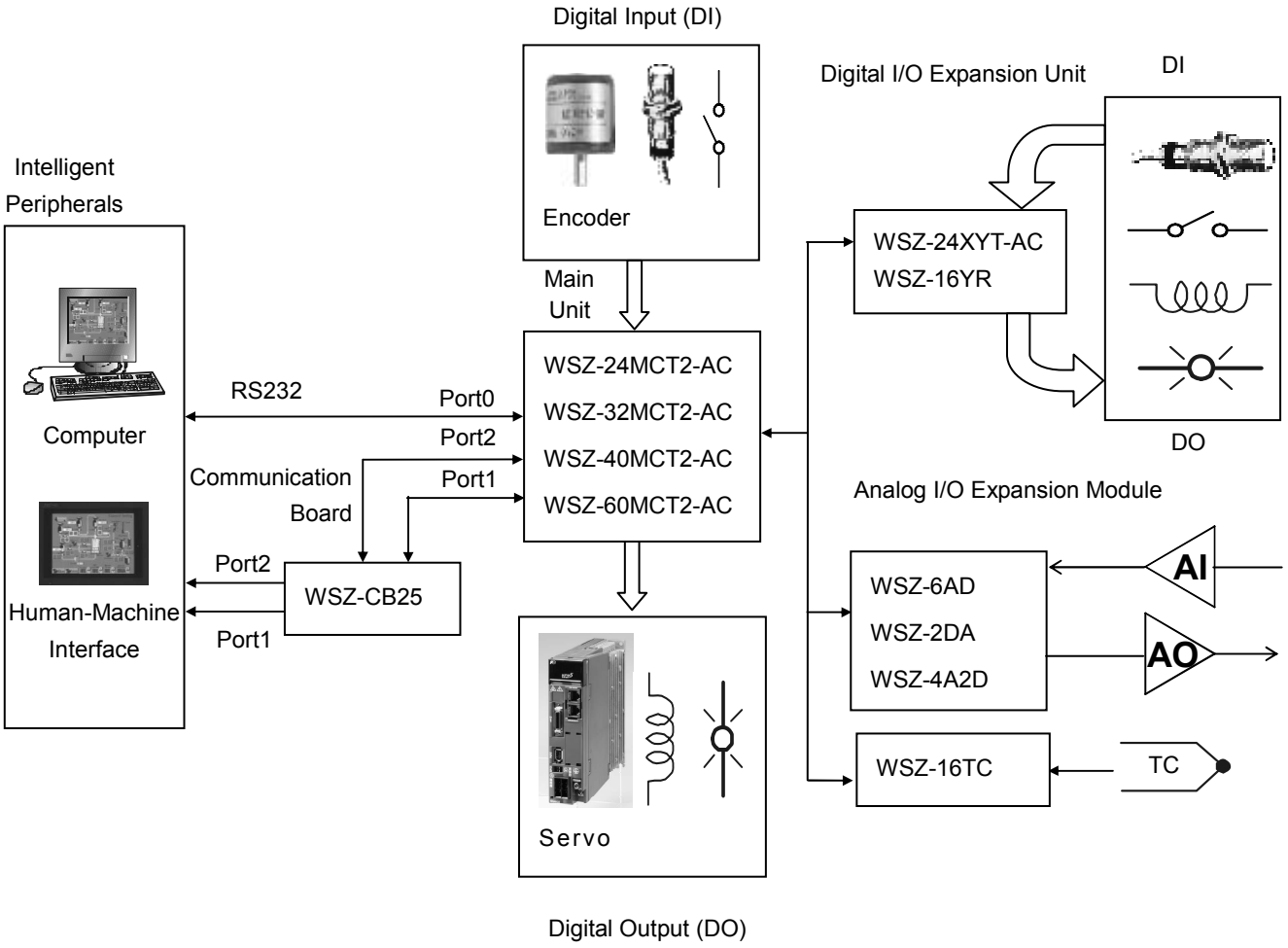


Units: mm

Chapter 2 System Architecture

2.1 Single-WSZ Controller

The Single-Unit system means a system built only by a single WSZ controller and its expansion unit/modules and communication board. Such system has a limited capability, incorporate CPU communication via LINK function for expansions. The figure below shows the block diagram of the Single-Unit system of WSZ controller, where, besides the available main units, the available communication peripherals resources and I/O expansion resources are depict on the left and the right respectively.



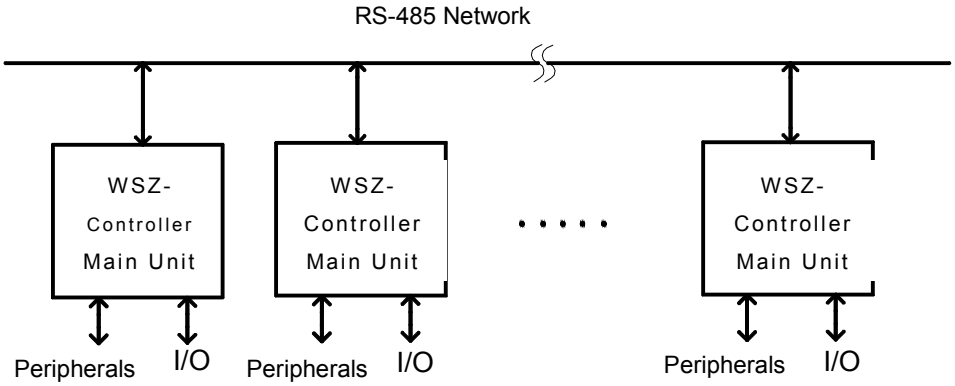
For the I/O of WSZ controller, it can achieve a maximum of 256 point digital input (DI), 256 point digital output (DO), 64 word numeric inputs (NI), and 64 word numeric outputs (NO).

Regarding communication resources, the WSZ-controller hardware can accommodate up to 3 communication ports (with a maximum speed of 921.6Kbps). In addition to providing the original communication protocol, it also supports the Modbus master/slave protocol or any user-defined protocol.

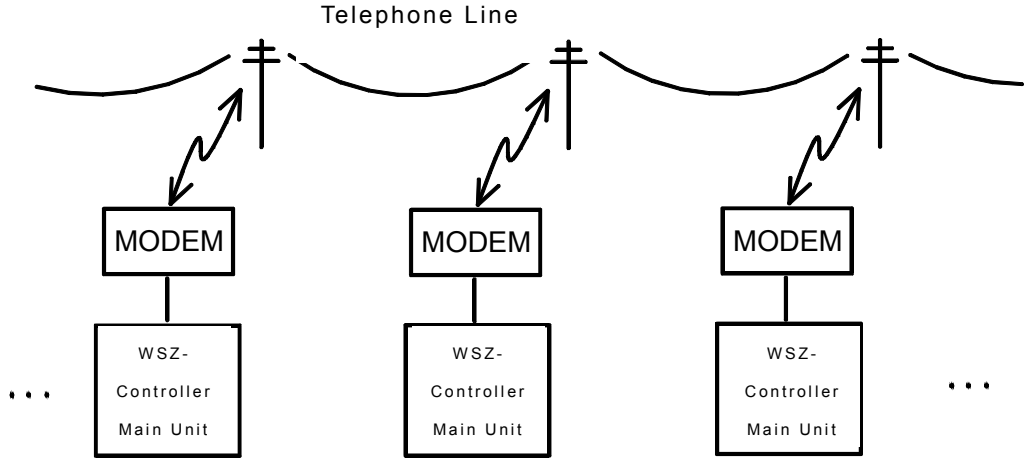
2.2 Formation of Multi-Unit System

By connections through communication ports and specific communication drivers, multiple Single-Unit Controller systems can be integrated to achieve resources sharing among multiple Controller or Controllers and its host computer. It is described as follows:

2.2.1 Connection of Multiple WSZ-Controller (CPU Link)



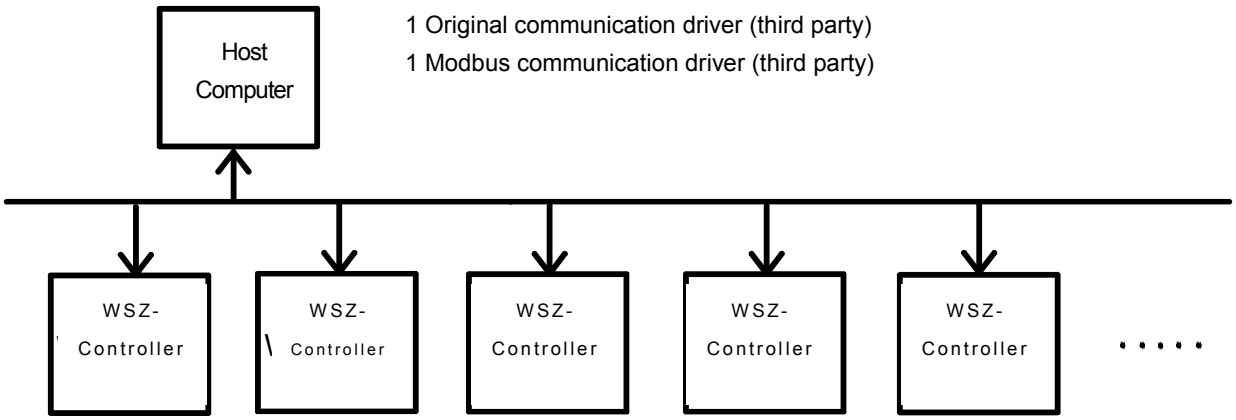
As shown in the figure, through the usage of high-speed RS-485 network, can easily establish the connections of 2~254 main units (each Controller with its own station number). All need to do is to write and execute CPU Link commands in one of the main units, which makes it the Master of the CPU Link network. No other command is necessary for other Slave units. The Master CPU will automatically collect the information or data in the specific areas of all units (including the Master) and put it into the Common Data areas (CDM) of all units. Thus all the units connected by network can share the data for each other and turning the finite Single-Unit system with limited I/O into a huge system.



Besides the above area network connection, WSZ-Controller can also be connected using MODEM via the phone line (either leased line or public phone line) to form remote multiple Controller Link. (When using a public phone line, the Master Controller will perform consecutive dialing for all its Slave Controller.)

2.2.2 Connection WSZ-Controller with Host Computer or Intelligent Peripherals

Any one of the two communication ports on WSZ-Controller can be used to connect to an upper-level computer or other systems, with this architecture, the WSZ-Controller is playing the Slave role. WSZ-Controller supports the Original and Modbus protocol. Connection can be established as long as the upper-level computer or intelligent peripherals use either one of the two protocols. The following is the block diagram.



Chapter 3 Expansion of WSZ-Controller

If the I/O point of WSZ Main unit is not enough for a specific application, it can be expanded with the additional expansion unit/modules. Besides I/O point there also have the requirements to expand the communication port in some occasions.

3.1 I/O Expansion

The expansion of WSZ-controller I/O consists of Digital I/O (DI/O, which status is represented by a single bit) and the Numeric I/O (NI/O, which status is represented by a 16-bit Word). Either the DI/O or the NI/O expansion is realized through expansion units or modules cascaded thru the usage of the "I/O Output Expansion Connector" located at the right side of WSZ main unit or expansion unit/module.

The I/O points of WSZ controller system are limited to 512 points of DI/O (256 points for DI and DO, respectively), 128 words of NI/O (64 words for NI and NO, respectively). Besides this there are two limits imposed by hardware: ① A maximum number of 32 units or modules can be used in the expansion. ② The total length of the expansion cables cannot exceed 5 meters.

⚠ Caution

1. If the DI or DO points of the application system exceed 256 points, while startup the main unit of WSZ controller will treat this as an illegal I/O configuration, which in return will flag as an error situation by turn on the "ERR" LED and put the error code in Y0~Y3 LED (refer the page H8-2, Chapter 8). The corresponding error code will also be indicated in the CPU status register (R4049).
2. If the NI or NO points of the application system exceed 64 points, while startup the main unit of WSZ controller will treat this as an illegal I/O configuration, which in return will flag as an error situation by turn on the "ERR" LED and put the error code in Y0~Y3 LED (refer the page H8-2, Chapter 8). The corresponding error code will also be indicated in the CPU status register (R4049).
3. The maximum number of expansion unit/modules of WSZ controller is 32. Beyond this numbers will be treated as an invalid I/O configuration and the main unit will stop its operation, which in return will flag as an error situation by turn on the "ERR" LED and put the error code in Y0~Y3 LED (refer the page H8-2, Chapter 8). The corresponding error code will also be indicated in the CPU status register (R4049).

⚠ Warning

1. The maximum length of the I/O expansion cable for WSZ controller is 5 meters. Cables longer than that will cause incorrect I/O operation because of excess signal delay in hardware or noise pickup, resulting in damage to equipment or posing hazard to operating personnel. Since this kind of situation cannot be detected by the main unit, users are advised to take extra cautions and necessary measures.

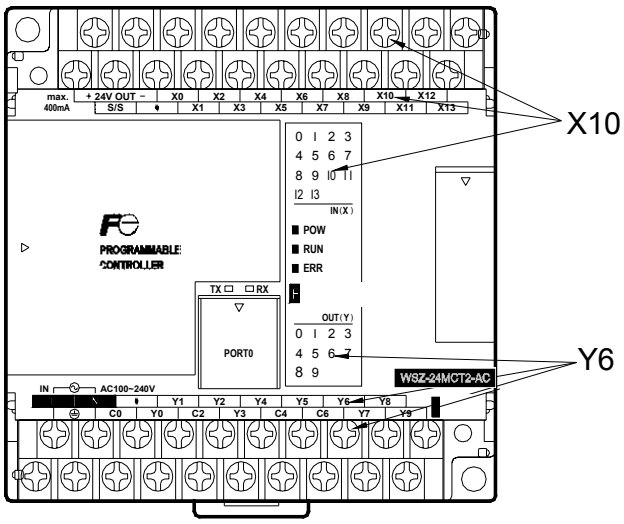
3.1.1 Digital I/O Expansion and I/O Numbering

Digital I/O means I/O with the discrete type status, including digital input (with initial X in DI numbering) and digital output (with initial with Y in DO numbering). The DI and DO of WSZ controller can both be expanded up to 256 points (numbered as X0~X255 and Y0~Y255, each with 256 points).

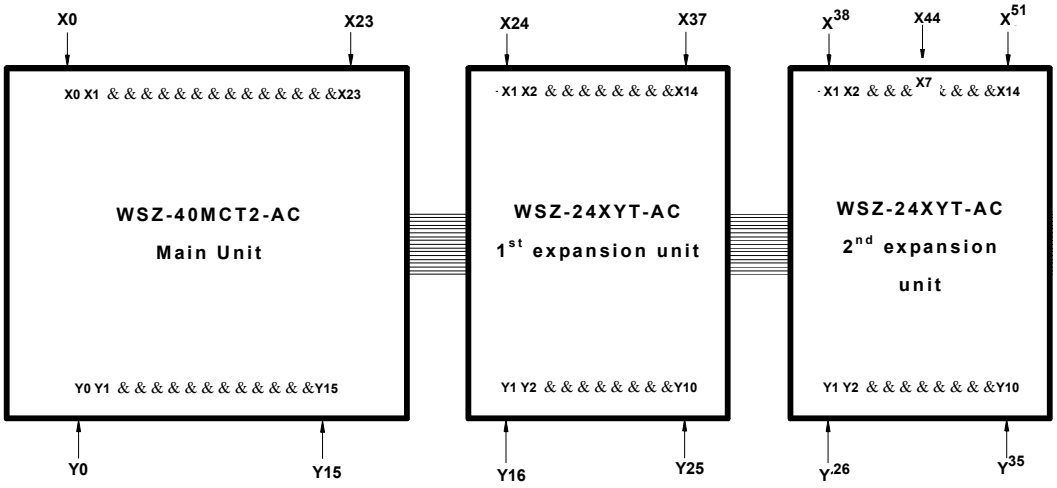
The status of input contacts (X0~X255) of controller come from the input signal connected to the digital input terminal block on main unit or expansion unit/module; while the status appears at digital output terminal block of main unit and expansion unit/module reflects the digital output relay (Y0~Y255) status inside controller.

On main unit, at the position below the digital input terminal block and the position above the output terminal block, there have labels indicate the corresponding signal name. They label each terminal with numbers representing the corresponding digital input contact Xn and digital output relay Yn. In the example of the main unit in WSZ-24MCT2-AC the corresponding digital input contacts on the input terminal block are labeled X0~X13, and the corresponding digital

output relays on the output terminal block Y0~Y9. Users only need to locate the printed label for each terminal to find out its I/O number. The LED status display region also indicates the ON/OFF status for all DI (X0~X13) and DO (Y0~Y9) on the main unit. Users can easily find each terminal with its I/O number and LED status indication, as shown in the figure below using X10 and Y6 as an example:



While the various expansion unit/modules other than the main units have the same printed labels on the input/output terminals as the main units do, these labels are only relative I/O numbers, different from the absolute I/O numbers on main units. The number of a terminal only represents its order on the expansion unit/module. For example, the first contact is X1 or Y1, the second X2 or Y2, etc. All numbers on the expansion unit/module begin with 1. The actual number of digital input contact or the output replay, however, is determined by summing the numbers on all previous expansion units/modules and the main unit. See the following figure and its calculation.



As shown in the above figure, because the top X numbers of the previous two units are 23 and 14, respectively, the number of input contact X12 on second expansion unit should be:

X (23 + 14 + 12) = X49

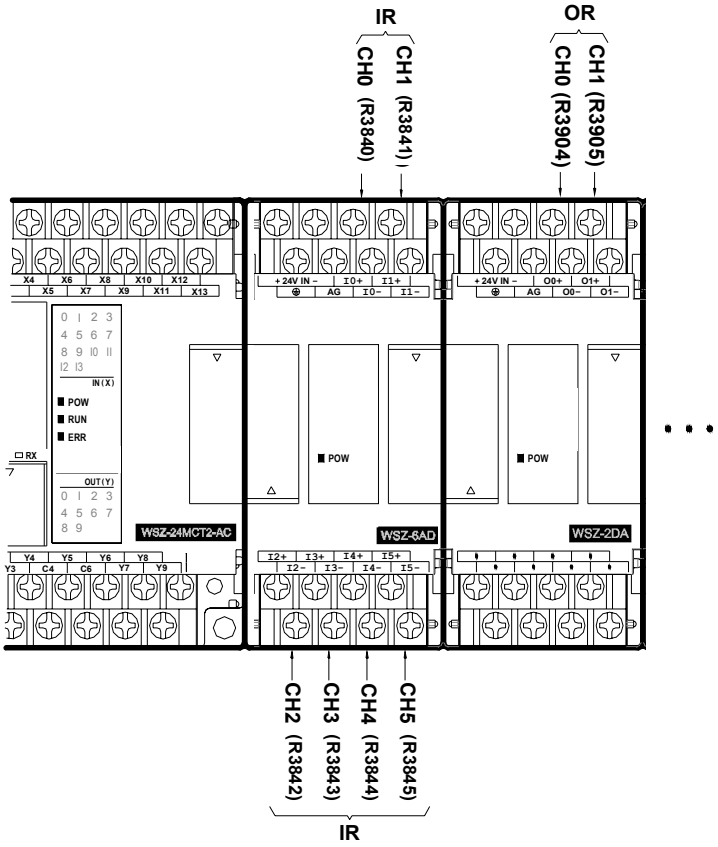
3.1.2 Numeric I/O Expansion and I/O Channel Mapping

The numeric I/O in WSZ-controller treat 16 single-bit data as one 16-bit numeric data (Word) ranging from the 0~65535. Since all numeric data of WSZ controller are stored in the register inside controller (16-bit width), therefore numeric I/O is also called register I/O. The Input Register (IR) has 64 Word (R3840~R3903) for inputs from external numeric input (NI) module, and the Output Register (OR) also has 64 Word (R3904~R3967) for outputs to external numeric output (NO) module.

Analog Input Module is of Numeric input (NI) modules which use input register (IR) to convey the status. Analog Output Module is of Numeric output (NO) modules which output are directly from the Output register (OR). The Analog Input and Analog Output are of analog voltage or current. Either the magnitude of voltage or current is represented by the 16-bit value of the corresponding register. The corresponding current/voltage signal of any IR or OR on the NI/O module is named as a Channel (CH). The channels on the NI module are called numeric input channels (NI channels) and those on NO module numeric output channels (NO channels). The number of IR/OR used by NI and NO channels on each module varies depending on the module type or working mode. The following table lists the numbers of IR and OR used by NI and NO channels on each NI/O module:

NI/O Module Name	NI Channel Label	NO Channel Label	Number of IR occupied (Word)	Number of OR occupied (Word)	Note
WSZ-6AD	CH0		1		
	CH1		1		
	CH2		1		
	CH3		1		
	CH4		1		
	CH5		1		
WSZ-2DA		CH0		1	
		CH1		1	

The corresponding IR or OR number calculation of the NI/O module starts from the first expansion unit/module (main unit itself does not have any NI/O). The first NI channel corresponds to the first IR register (R3840). Adding the IR number of the second NI channel with the number of IR used by the second NI channel gives the IR number of the third NI channel. All other numbers can be obtained accordingly. Similarly, the first NO channel corresponds to the first OR (R3904). Adding R3904 with the number of OR used by the first NO channel gives the OR number of the second NO channel. (In the cumulative calculation of NI channels, care only for NI channels and disregard DI/O and NO. Similarly, in the case of NO channels, disregard DI/O and NI channels.) The following figure helps users find out the relation between NI/O channels and controller's IR and OR.



During the startup stage, WSZ controller will automatically detect the types and CH numbers of expansion units. While operation, the WSZ controller will read the CH input values from the NI module and stores them into corresponding IR (R3840~R3903) and outputs OR values (R3904~R3967) to channels on the NO module. No pre-configuration or setting by users is required.

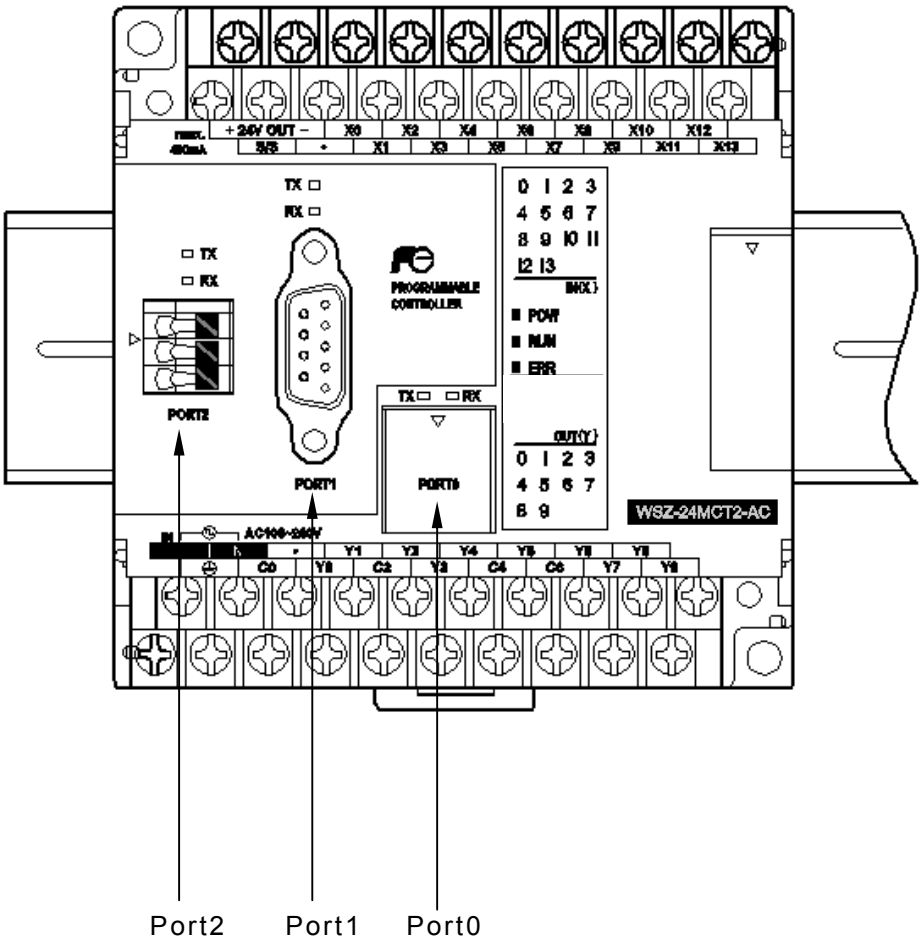
3.2 Expansion of Communication Port

The main unit has one build-in communication port (port 0, with RS232 interface). Expansion of communication ports can be achieved by employing Communication Board (CB). The available model of CB for WSZ is:

	Model Number	Specifications
Communication Board (CB)	WSZ-CB25	1 RS232 (port1) + 1 RS485 (port2) communication board

Communication board, which can be directly installed on main units, is employed for expansion of communication ports port1 and port2. The labels of communication ports are marked on the cover plate of communication board, from which users can easily identify each port. Except that the build-in communication port (Port 0) can only be used for RS232 interface, all the other ports (Port1~2) can be used for RS232 or RS485 interface in CB. The following figure shows an example of expansion of 3 (maximum allowed number) communication ports (CB25):

The most expansion of communication port diagram



Port2 Port1 Port0

Chapter 4 Installation Guide

Danger

1. Turn off all power during installation of WSZ controller or related equipments to prevent electric shock or damage to equipment.
2. Upon completion of all installation wiring, put the protective cover back on the terminal block before turning on the power to avoid electrical shock.
3. During installation, never remove the dust cover sheet that were surrounded the controller before wiring is completed to avoid complications such as fire hazards, breakdown, or malfunction caused by drill dust or wire shreds falling inside controller.
4. Upon completion of installation and wiring, remember to remove the dust cover sheet to avoid fire, breakdown or malfunction, caused by overheating.

4.1 Installation Environment

Caution

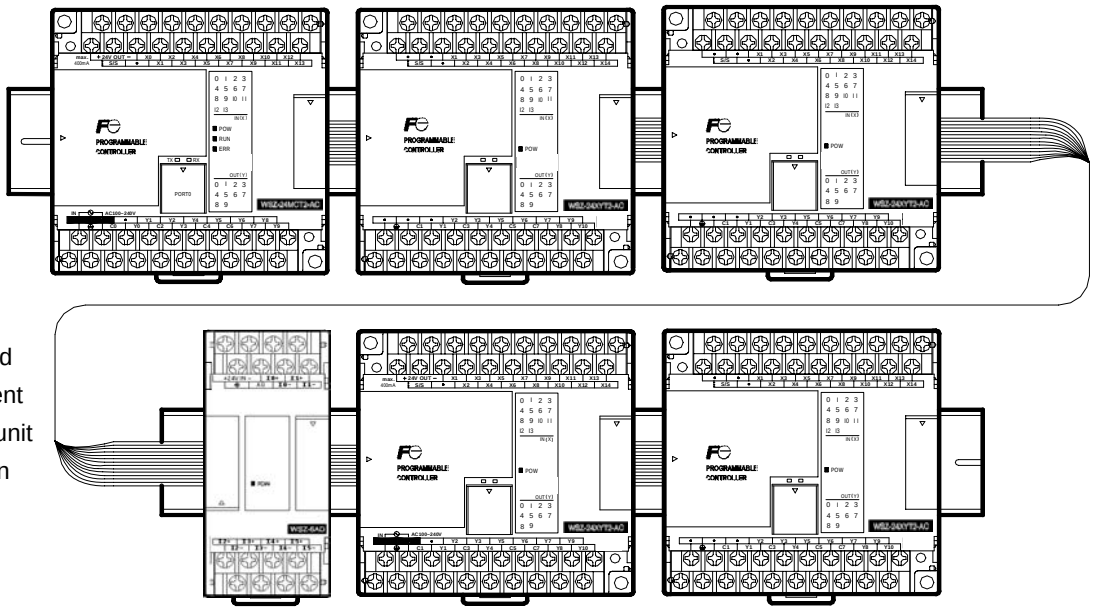
1. Environmental specifications of WSZ controller cannot exceed those listed in this manual. In addition, do not operate this equipment in environments with oil smoke, conductive dust, high temperatures, high humidity, corrosion gases, inflammable gases, rain or condensation, and high vibrations and shock.
2. This product has to be housed appropriately whether it's used in a system or standalone. The choice and installation of housing must comply with local national standards.

4.2 Precautions of Controller Installation

To avoid interference, the controller should be installed to keep from noise sources such as high-voltage or high-current lines and high power switches. Other precautions are:

4.2.1 Placement of Controller

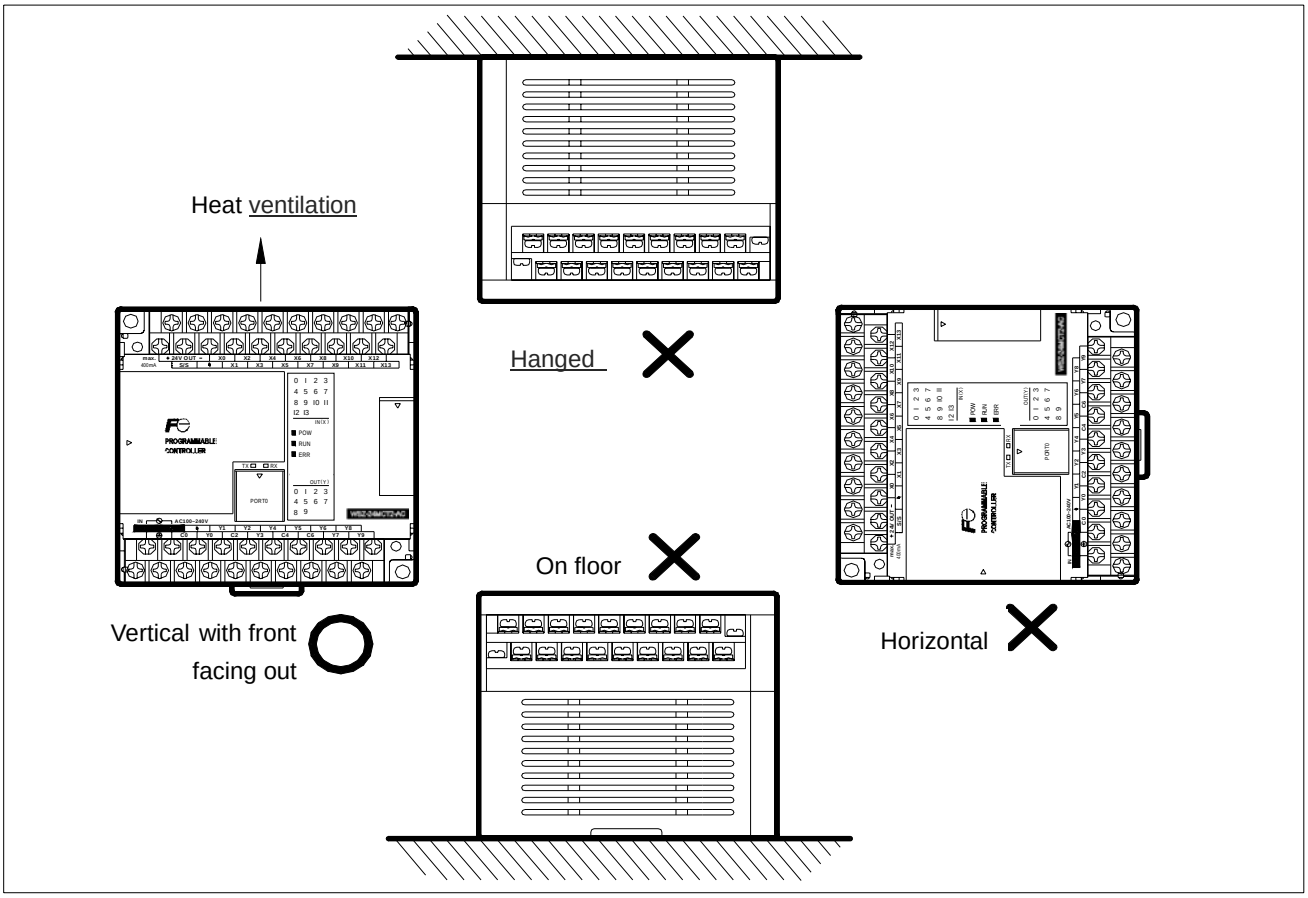
Fixation of WSZ controller, which can be fixed by DIN RAIL or screws, should place vertically and start from the main unit on the left to the expansion unit on the right. A typical figure of placement is shown below:

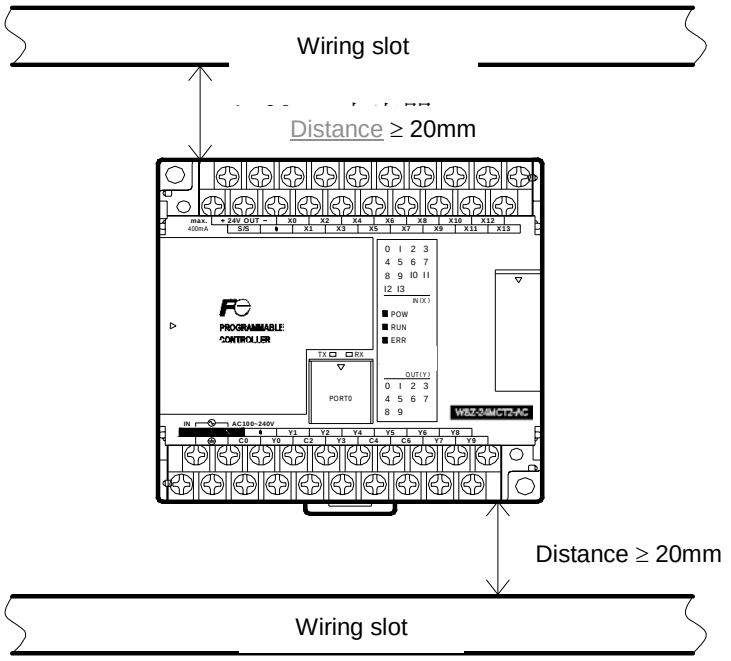


Suggested arrangement of multiple unit expansion

4.2.2 Ventilation Space

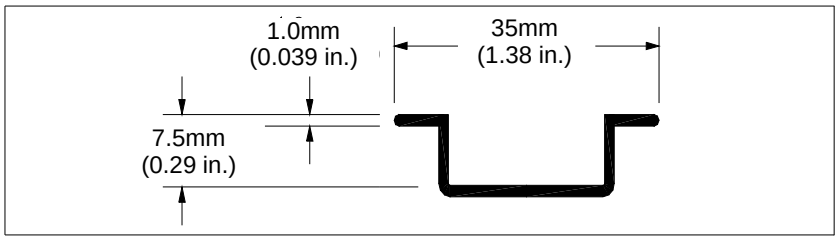
The heat in WSZ controller is ventilated via air circulation. There should reserve more than 20mm space, both below and above controller, and with vertical installation, for ventilation. As shown in the figure below:



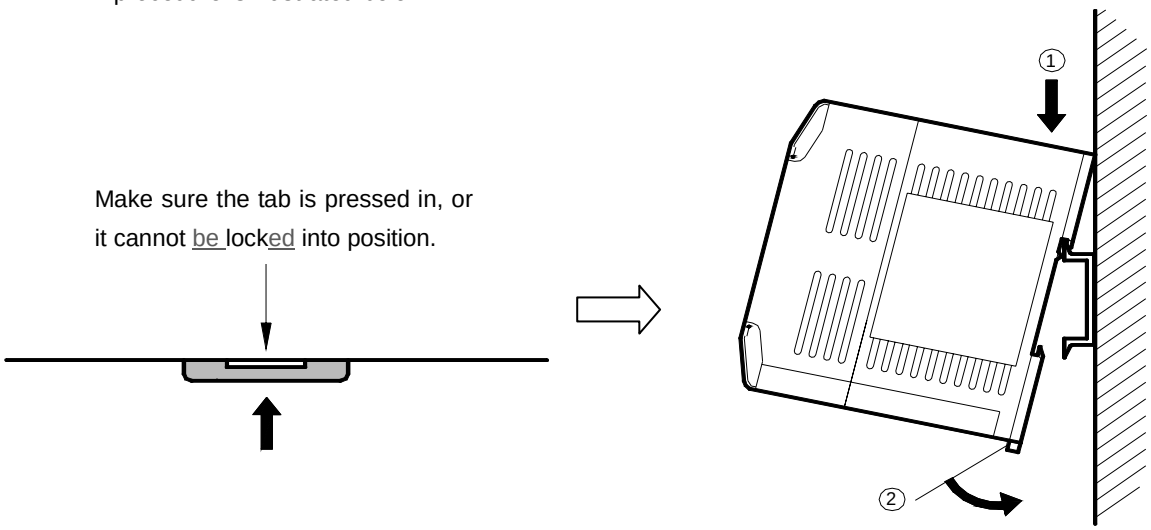


4.3 Fixation by DIN RAIL

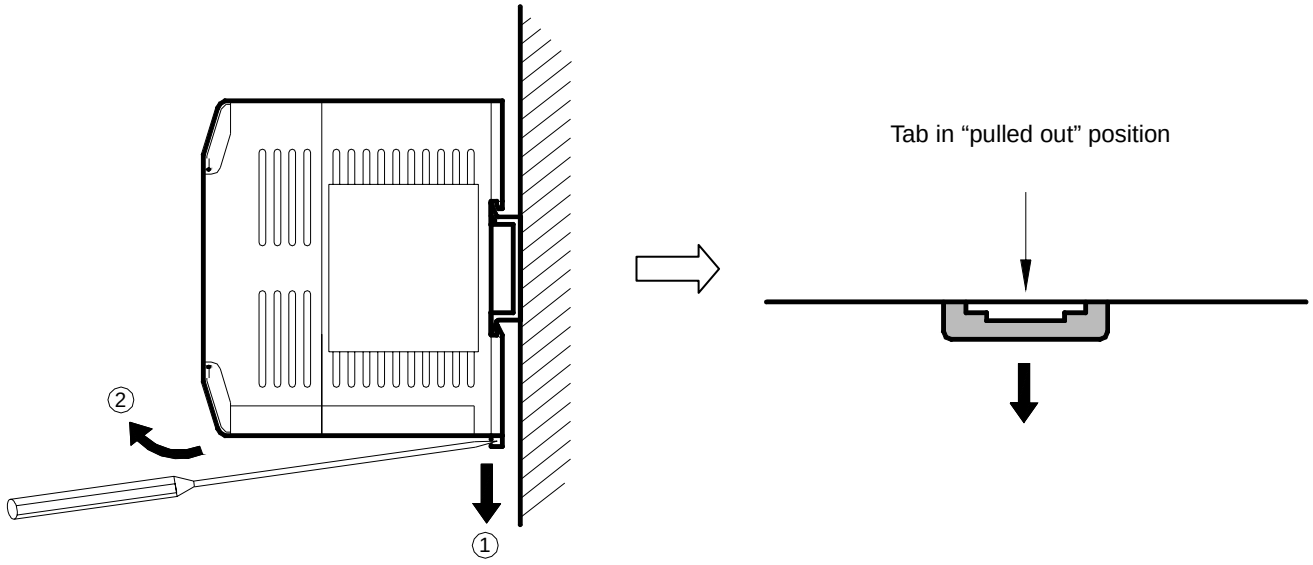
In an environment with slight vibration (less than 0.5G), this is the most convenient way of fixation and is easy for maintenance. Please use DIN EN50022 DIN RAIL, as shown in the figure below.



Mount ⇨ Hold controller facing its front, press it down with a 15 degree tilt onto the DIN RAIL. Swing it down until the upper edge of DIN RAIL groove on controller back touches the upper tab of DIN RAIL. Then use this locked-in point as a pivot to press the controller forward on the bottom and lock it in position. The procedure is illustrated below:



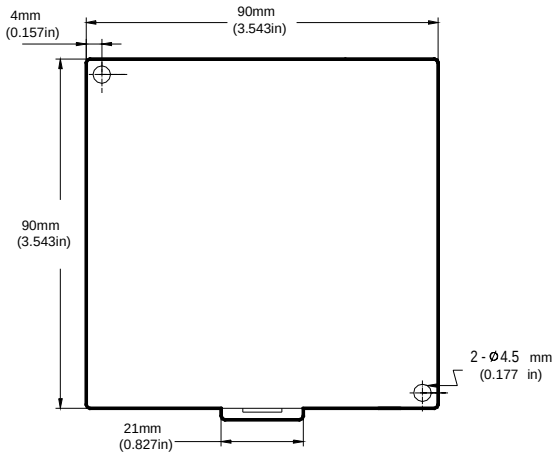
Dismount ⇨ Use a long screwdriver to reach in the hole on the DIN RAIL tab. Pull out the tab to “pulled out” position to remove controller, as shown in the figure below.



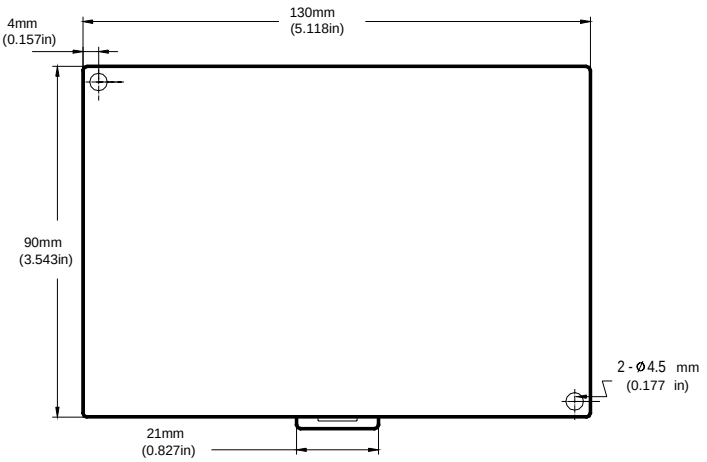
4.4 Fixation by Screws

In environments with larger vibration (more than 0.5G), the unit must be secured by M3 or M4 screws. Positions and sizes of screw holes in various models of WSZ controller are illustrated in the following:

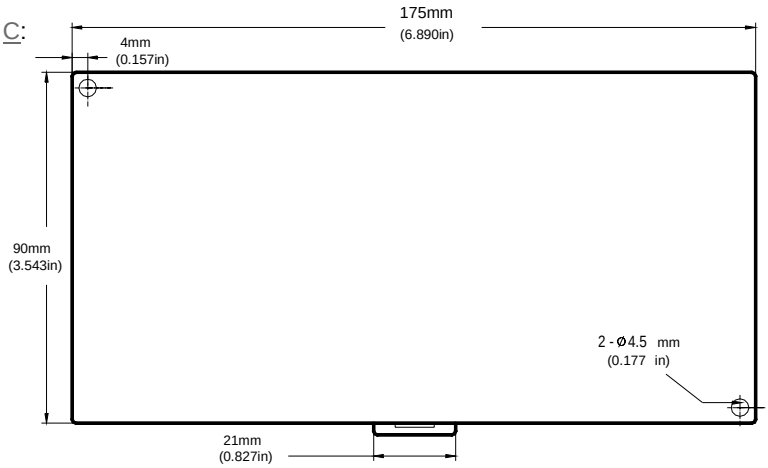
Size A:



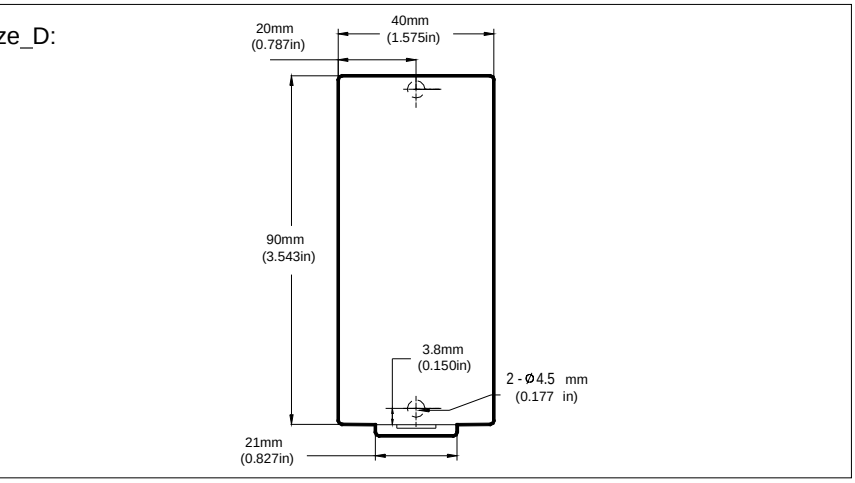
Size B:



Size C:



Size_D:



4.5 Precautions on Construction and Wiring

1. During the wiring of WSZ controller, please follow local national standards or regulations for installation.
2. WSZ-Controller's I/O wiring restricts the use of AWG24~AWG12 copper wire, please note that I/O wiring to be in accordance with the current load and select the appropriate wire diameter of the wiring.
3. Shorter wires are preferred. It is advised that the length of I/O wiring does not exceed 100m (10m for high-speed input).
4. Input wiring should be separated from output or power wiring (at least 30~50mm apart). In case separation is not possible, adopt vertical crossing, no parallel wiring is allow.
5. The pitch of WSZ controller terminal block is 7.62mm. The torque for screw and suggested terminal is shown below:

7.62 mm terminal block		torque: 6~8kg-cm

Chapter 5 Power Supply Wiring, Power Consumption Calculation, and Power Sequence Requirements

WSZ controller internally has three kinds of circuit: a 5VDC logic circuit, a 24VDC output circuit, and a 24VDC input circuit. They are all powered by the built-in power supply of main/expansion units. Expansion modules other than main/expansion units do not contain any power supply and are powered by the power supply inside the main/expansion units.

⚠ Caution

In industrial environments, main power may irregularly experience a surge current or high voltage pulse caused by the start or shut down of high power equipment. Users are advised to take necessary measures (for example, the use of isolation transformer or other MOV suppression devices) for the protection of controller and its peripherals.

5.1 Specifications of AC Power Sourced Power Supply and Wiring

The available AC power supplies of WSZ controller are the 24 Watt (SPW24-AC) supply for 20~60PTs main/expansion unit. SPW24-AC is to be installed on a main unit or inside an expansion unit, where their appearances are invisible. The following table lists the specifications:

Spec / Model / Item		SPW24-AC
Input Range	Voltage	100 ~ 240VAC -15% / +10%
	Frequency	50 / 60HZ -5% / +5%
Max. Power Consumption		36W / 24W
Inrush Current		20A@264VAC
Allowable Power Interrupt		< 20ms(min.)
Fuse Spec.		2A , 250VAC
Isolation Type		Transformer/Photo-coupler Isolation, 1500VAC/minute
Power Output *1	5VDC(logic circuit)	5V , ±5% , 1A(max)
	24VDC(output circuit)	24V , ±10% , 400mA(max)
	24VDC(input circuit)	24V , ±10% , 400mA(max)

Note *1 : The 5VDC (for logic circuit) output power and the 24VDC (for output circuit) power can be accessed from the "I/O expansion output header" located on the right side of the main/expansion units for expansion modules. And the 5VDC power is also used by communication board (CB25). The 24VDC power for input circuits is provided from the farthest 2 upper left terminals (labeled "+24V OUT-") on the input terminal block of main/expansion unit to input circuit in expansion module or other sensors.

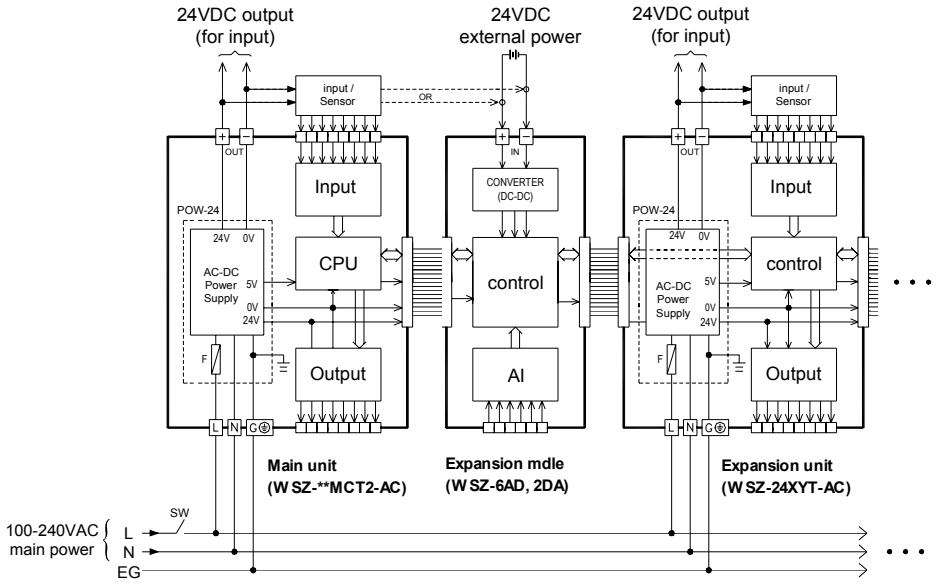
Caution

The schematic diagram of AC power supply wiring in main/expansion units is shown below. Also be cautious about the following:

1. Please follow the wiring schemes regulated by local national standards to use single-pole switch（break hot wire〝L〞）, or double-pole switch（break both〝L〞 and〝N〞）, to turn on or off the AC input power.
2. In wiring, hot wire〝L〞must be connected to the L terminal on unit, while the ground line〝N〞connected to the N terminal. Please use wires with diameters 1mm^2～2mm^2.
3. All G⏚ terminals on main unit and expansion unit/module have to be connected to the EG (Earth Ground) terminal of main power system as shown in the figure below, with wire diameters larger than 2mm^2.

Warning

Output of power for sensor cannot be connected in parallel with other powers, in which the conflict between two sets of power will decrease their lifetime or cause immediate damage. This will induce unexpected malfunction of controller and cause serious or even deadly damage to people or equipment.



5.2 Residual Capacity of Main/Expansion Unit & Current Consumption of Expansion Module

Besides its own circuit usage, the residual capacities of three sets of built-in power supply of main/expansion unit are big enough for other expansion modules usage. As each model of the main/expansion unit has different residual capacity, various models of expansion modules also consume different amounts of current. In practice, one has to consider the match between the two to avoid overload in any of the three sets of output power. In the following, the worst case of the available residual capacity in each main/expansion unit and the maximum power consumption of expansion modules are described below spare.

5.2.1 Residual Capacity of Main/Expansion Unit

Model \ Extra Capacity		Output Power		
		5VDC(logic circuit)	24VDC(output circuit)	24VDC(input circuit)
Main Unit	WSZ-24MCT2-AC	722 mA	325mA	295mA
	WSZ-32MCT2-AC	712 mA	315mA	262mA
	WSZ-40MCT2-AC	688 mA	295mA	244mA
	WSZ-60MCT2-AC	644 mA	255mA	190mA
Expansion Unit	WSZ-24XYT-AC	948 mA	337mA	337mA

- In the above table, the residual capacity is calculated according to the most power-consuming model of in each main/expansion unit by its I/O point number, under the maximum load condition (with both DI and DO ON). The basic units for calculation are 7.5mA /PT for high/medium speed DI, 4.5mA/PT for low speed DI, 10mA/PT for high speed DO, 7.5mA/PT for medium speed DO, and 5mA for low speed DO.

⚠ Warning

Either for the built-in power supply of the main/expansion unit or the expansion power supply for the expansion unit, the total amount of current cannot exceed the value listed in the above table. Any violation will cause a voltage drop by overloading the power supply, or intermittent powered with the supply in protection mode, either of which will result in unexpected action of controller and cause harm to people or damage to equipment.

5.2.2 Maximum Current Consumption of Expansion Module

Without its own power supply, expansion modules must be supported by the main/expansion unit, expansion power, or external power supply (24VDC input circuit alone). The following table lists the maximum consumption current of each expansion module.

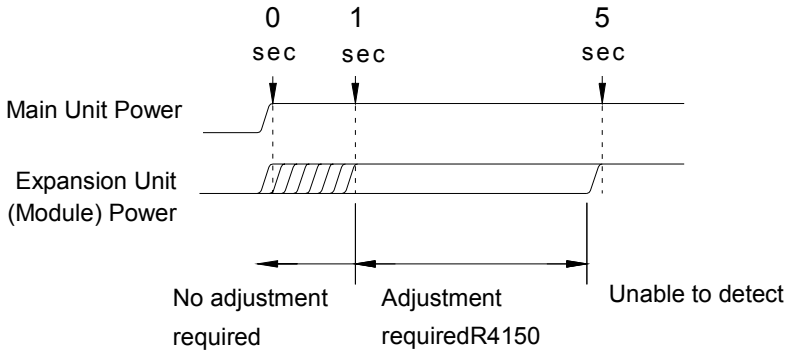
Model \ Current		5VDC Logic Circuit	24VDC Output Circuit	24VDC Input Circuit
Expansion Module	WSZ-6AD	25 mA	—	53 mA
	WSZ-2DA	33 mA	—	90 mA
	WSZ-4A2D	25mA	—	80mA
	WSZ-16YR	39mA	80mA	—
	WSZ-16TC	33mA	—	30mA
Communication Board (CB)	WSZ-CB25	55 mA	—	—

- The above table lists the required current for the maximum consumption in each expansion module. The 24VDC input circuit consumes 4.5mA less per point of OFF state DI in DI/O module. The effects of power consumption variation regarding the ON/OFF state of DI/DO of expansion modules other than DI/DO are less significant and can be neglected.

- The effect of residual capacity variation regarding the ON/OFF state of DI/DO for 5VDC logic circuit can be neglected.

5.3 Requirement of Power Sequence in Main Unit & Expansion Unit/Module

When the power is on, the main unit first detects the type and number of expansion unit/module attached to its expansion interface and get the actual I/O configuration. Therefore, while the main unit performs detection, the power in expansion unit/module should be already UP, otherwise, the detected I/O configuration will not correct. Namely, the power of expansion unit/module should be ON simultaneously or even earlier. There will be no time sequence error when main unit/expansion unit/module is connected together to one power. If the expansion unit and main unit powered by different powers (or the same power but different switches), or external power supply is used for expansion modules, time sequence of both powers should be considered. To solve the problem of the expansion unit/module power not get ready before main unit power does, WSZ controller provides a special R4150 register which can delay the detection time of I/O configuration. The time base of R4150 is 0.01sec with a default value of 100 (namely a 1sec delay), which can be set from 100~500 (1~5sec), as shown in the figure below. If the expansion unit power cannot be UP within 1sec after main unit power is ON, the R4150 time needs to be set longer to delay the detection by CPU. It cannot exceed 5sec; however, otherwise the configuration of expansion interface cannot be detected.



Chapter 6 Digital Input (DI) Circuit

The WSZ controller provides the single-end 24VDC inputs which use the common terminal to save terminals. The response speeds of single-end input circuits are available in high, medium and low. The single-end input circuit is set to SINK or SOURCE (we will use the term SRCE) by the wiring of the common terminals S/S inside controller and external common wire of input circuits (see Sec. 6.2 for details).

6.1 Specifications of Digital Input (DI) Circuit

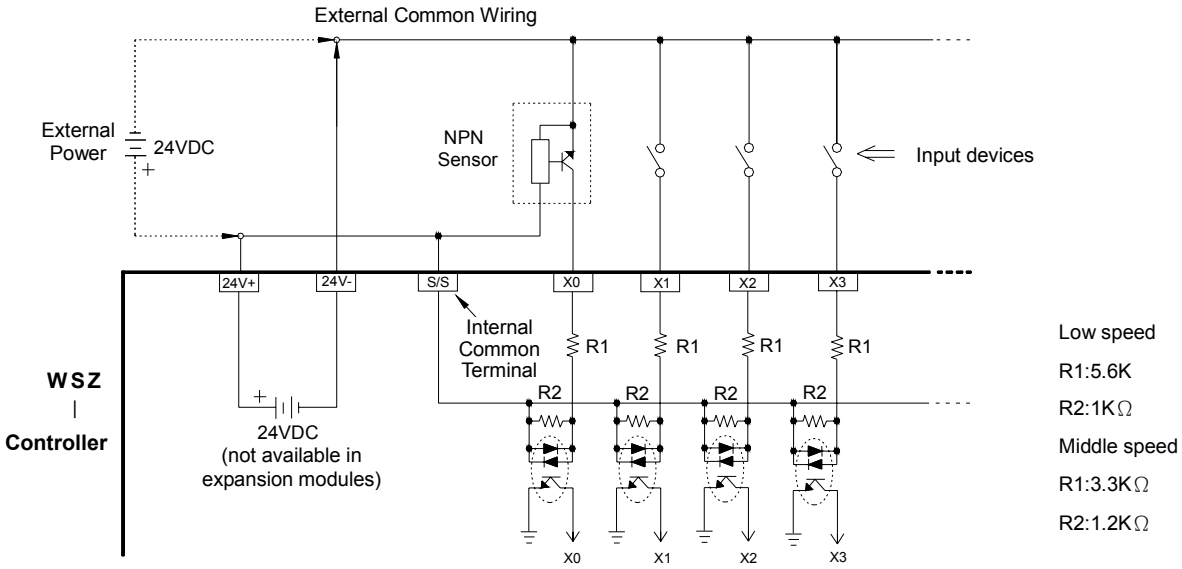
<table>
<tr><th colspan="3" rowspan="2">Item / Specifications</th><th colspan="5">24VDC Single-end Input</th><th rowspan="2">Note</th></tr>
<tr><th>High Speed (HSC)</th><th colspan="2">Medium Speed (HSC)</th><th>Medium Low Speed (Capture input)</th><th>Low Speed</th></tr>
<tr><td colspan="3">Maximum input frequency
Accumulated time</td><td>200kHz</td><td>20kHz (HHSC)</td><td>Total 5kHz (SHSC)</td><td>0.47ms</td><td>4.7ms</td><td rowspan="11">Half of maximum frequency while A/B phase input</td></tr>
<tr><td colspan="3">Input Signal Voltage</td><td colspan="5">24VDC±10%</td></tr>
<tr><td rowspan="2">Input Current Threshold</td><td>ON Current</td><td></td><td>> 8mA</td><td colspan="3">> 4mA</td><td>> 2.3mA</td></tr>
<tr><td>OFF Current</td><td></td><td>< 2mA</td><td colspan="3">< 1.5mA</td><td>< 0.9mA</td></tr>
<tr><td colspan="3">Maximum Input current</td><td>10.5mA</td><td colspan="3">7.6mA</td><td>4.5mA</td></tr>
<tr><td colspan="3">Input Status Indication</td><td colspan="5">Displayed by LED: Lit when “ON”, dark when “OFF”</td></tr>
<tr><td colspan="3">Isolation Type</td><td colspan="5">Photo-coupler signal isolation, 500VAC, 1 minute</td></tr>
<tr><td colspan="3">SINK/SRCE Wiring</td><td colspan="5">Via variation of internal common terminal S/S and external common wiring</td></tr>
<tr><td rowspan="5">List of Input Response Speed for Various Models</td><td>WSZ-24MCT2-AC</td><td></td><td>X0,1,4,5</td><td>X8,9, 12,13</td><td>X2,3,6, 7,10,11</td><td>-</td><td>-</td></tr>
<tr><td>WSZ-32MCT2-AC</td><td></td><td>X0,1,4,5,8,9</td><td>X12,13</td><td>X2,3,6, 7,10,11 ,14,15</td><td>X16~19</td><td>-</td></tr>
<tr><td>WSZ-40MCT2-AC</td><td></td><td>X0,1,4,5,8,9</td><td>X12,13</td><td>X2,3,6, 7,10,11 ,14,15</td><td>X16~23</td><td>-</td></tr>
<tr><td>WSZ-60MCT2-AC</td><td></td><td>X0,1,4,5,8,9,12, 13</td><td>-</td><td>X2,3,6, 7,10,11 ,14,15</td><td>X16~35</td><td>-</td></tr>
<tr><td>Expansion Unit</td><td></td><td>-</td><td>-</td><td>-</td><td>-</td><td>All Input Points</td></tr>
<tr><td colspan="3">Noise Filtering Time Constant</td><td>DHF(0ms ~ 15ms) + AHF(0.47μ s)</td><td colspan="2">DHF(0ms ~ 15ms) + AHF(4.7μ s)</td><td>DHF(1ms ~ 15ms) + AHF(0.47ms)</td><td>AHF(4.7ms)</td><td>DHF：Digital Hardware Filter
AHF：Analog Hardware Filter</td></tr>
</table>

HSC: High Speed Counter
HHSC: Hardware High Speed Counter
SHSC: Software High Speed Counter

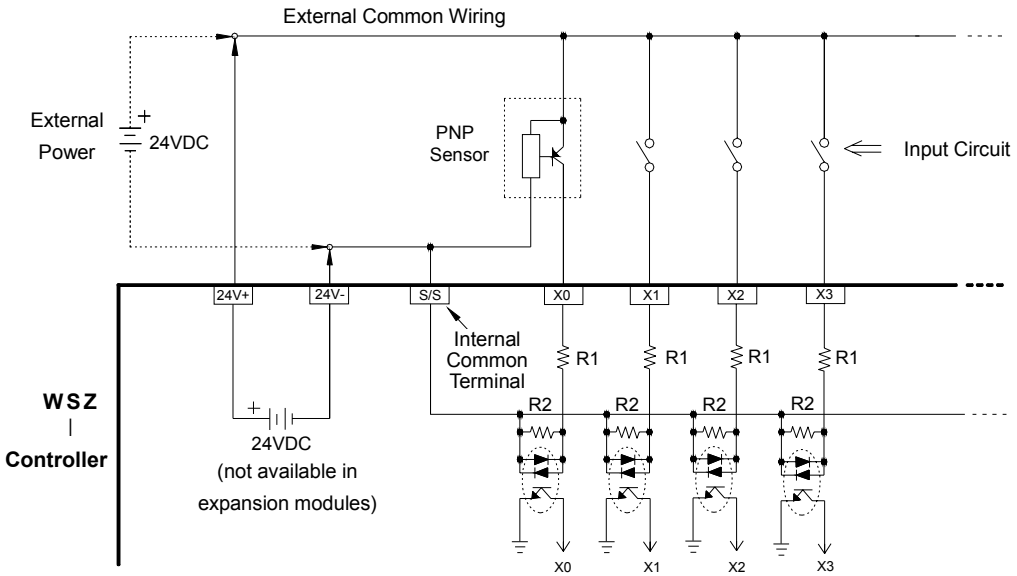
6.2 24VDC Single-End Input Circuit and Wiring for SINK/SRCE Input

The 24VDC single-end digital input circuits are available for high, medium and low speed. They all have the similar circuit structures but with different response speeds. To save input terminals, the circuit of single-end input is implemented by connecting one end of all input points (photo-coupler) inside the controller to the same internal common point labeled as S/S. The other end of each input circuit is connected to corresponding terminals such as X0, X1, X2, etc. The S/S common terminal and N single-end inputs comprise of N digital inputs (i.e., only N+1 terminals are used for N terminals). Therefore, we call this type of input structure the single-end input. The user also needs to do the same thing when making the connection of external digital input devices. Namely, the one end of all input devices (e.g., buttons, switches) are connected together and called the external common wire, while the other ends of input circuits are connected to the input terminals X0, X1, X2, etc., of controller. Then finish it by connecting the external common wiring and internal common terminal S/S to the positive/negative terminals of the 24VDC power. When connect the internal common terminal S/S to 24V+(positive) and the external common wire to 24V－ (negative), then the circuit serve as SINK input. On the contrary, while exchange the wiring of the above internal and external common will serve as a SRCE input. The above wiring schemes can illustrated below:

- Wiring of single-end common SINK input (internal common terminal S/S → 24V+, external common wiring → 24V－)



- Wiring of single-end common SRCE input(internal common terminal S/S → 24V－, external common wiring →24V+)

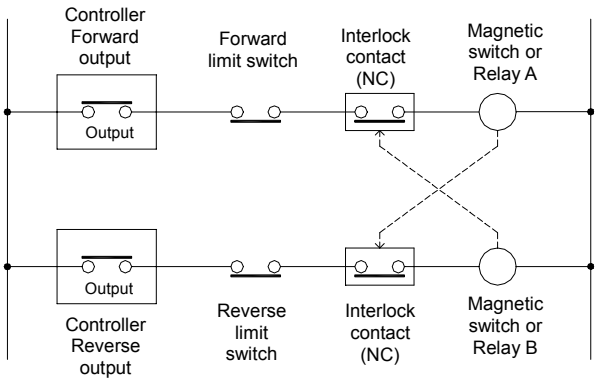


Chapter 7 Digital Output (DO) Circuit

The digital outputs of WSZ-controller are available in the single-end output circuit for saving terminal. There are two kinds of output device for the single-end output, which are relays and transistors. Since the relay is bilateral, even when used in single-end output, it can serve as SINK or SOURCE output. The transistor, however, because of its polarities, is needed to be careful about connection with the power (common point Cn of SINK output must connect to negative end of DC power).

⚠ Warning

1. No over current protection is available in the WSZ series controller. All other output circuits have to be added with over current or short circuit protections externally, such as fuses, in applications with safety concern.
2. Terminals labeled by 〝●〞 on the terminal block are empty contacts, which cannot be connected with any wire to maintain the required safety clearance and to avoid damage to the unit.
3. In situations where simultaneous operations of outputs(such as reverse/forward action of motor) pose safety concerns, besides the interlock in controller programs, additional interlock circuits are needed outside controller, as shown below：



7.1 Specifications of Digital Output Circuit

Item / Specification			Single-End Transistor Output			Single-End Relay Output
			High Speed	Medium Speed	Low Speed	
Maximum switching (working) Frequency*			200KHz	20KHz	On/Off	For On/Off, not for frequent exchange
Working Voltage			5~30VDC			<250VAC,30VDC
Maximum Load Current		Resistive	0.5A		0.5A/ 0.1A (24XYT)	2A/single, 4A/common
		Inductive				80VA(AC)/24VA(DC)
Maximum Voltage Drop (@ maximum load)			0.6V	2.2V		0.06V(initial)
Maximum load			-			2mA/DC power
Leakage Current			< 0.1 mA/30VDC			-
Maximum Output Delay Time		ON→ OFF	2μ s	15μ s		10ms
		OFF→ ON		30μ s		
Output Status Indication			LED is bit when 〝ON〞, dark when 〝OFF〞			
Over Current Protection			N/A			
Isolation Type			Photo-coupler Isolation, 500VAC, 1 minute			Electromagnetic Isolation, 1500VAC, 1 minute
Output Type			Transistor (SINK)			Bilateral device, can be arbitrarily set to SINK/SOURCE output
For Various Models Response Speed List of output	WSZ-24MCT2-AC		Y0~3	Y4~7	Y8, 9	-
	WSZ-32MCT2-AC		Y0~5	Y6, 7	Y8~11	-
	WSZ-40MCT2-AC		Y0~5	Y6, 7	Y8~15	-
	WSZ-60MCT2-AC		Y0~7	-	Y8~23	-
	Expansion Units/Modules		-	-	All output points	All output points

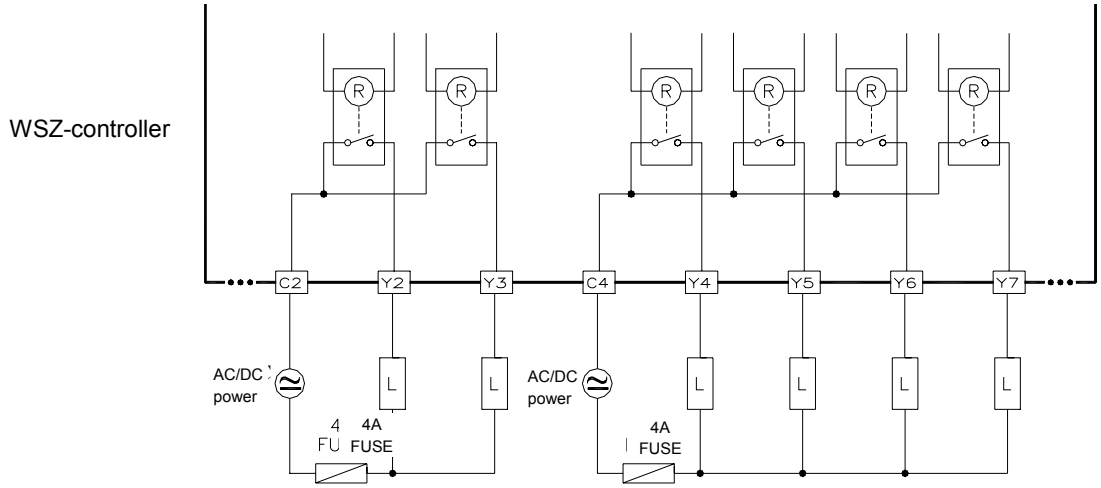
Note* : Half of the maximum frequency while A/B phase output

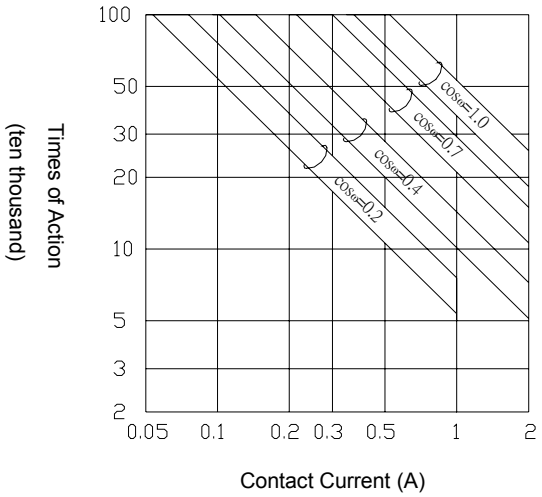
7.2 Single-End Output Circuit

Transistor output circuit is single-end output structure. A single-end output in each digital output (DO) takes up only one terminal. But since any output device has two ends, the one end of several output devices have to be connected together to one common point (called output common) for single-end output. Then each output point can output via this common point. The more output device share a same common points, the more terminals are saved, while relatively increasing the current running through the common point. Combination of any output common with its individual single-end outputs are called a Common Output Block, which is available in 2, 4 and 8PTs (high-density module) in WSZ controller. Each Common Output Block is separated from one another. The common terminal has a label initiated with letter “C”, while its numbering is determined by the minimum Yn number which comprises the output block. In the example of the figure below, the number of common terminal of output block Y2 and Y3 is C2, while the number of common terminal of output Block Y4, Y5, Y6 and Y7 is C4. The various single-end common output circuits are described below：

7.2.1 Structure and Wiring of Single-End Relay Output Circuit

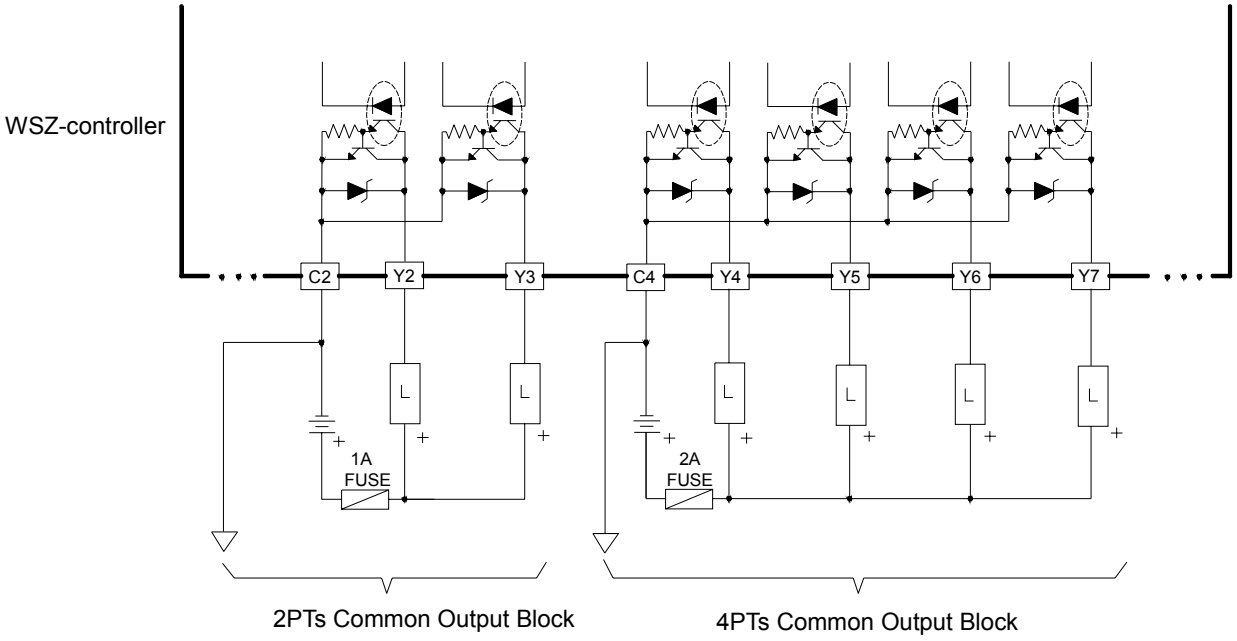
Because relay contacts have no polarity, it can be applied for AC or DC load power. Each relay can provide current up to 2A. The maximum rated current in all output commons of FBs-PLC is 4A. Its mechanical lifetime can reach up to 2 million times, while the contacts have a shorter lifetime. The lifetime also varies depending on working voltage, load type (power factor $\cos\phi$) and contact current. The relation between them is plotted in the figure below. In the case of pure resistive load ($\cos\phi$=1.0) at 120VAC and 2A, the lifetime of contacts is about 250 thousand times. While for high inductive or capacitive load with $\cos\phi$ up to 0.2 and current within 1A, the lifetime decreases rapidly to about 50 thousand times (AC200V) or 80 thousand times (AC120V).





7.2.2 Structure and Wiring of Single-End Transistor SINK Output Circuit

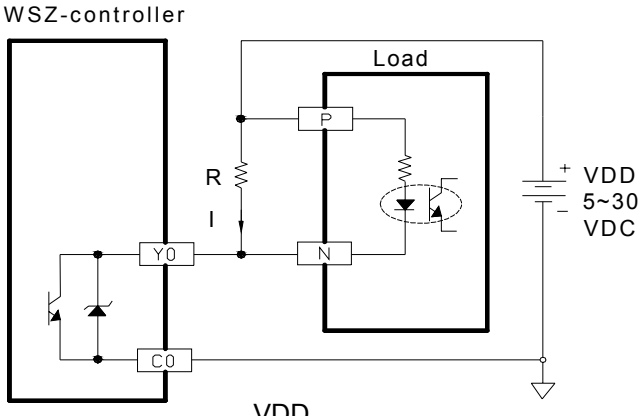
A. Transistor Single-End SINK Output



The figure above uses output block s of 2PTs common and 4PTs common as an example to explain the differences in structural and wiring for SINK output circuits, respectively.(8PTs common has the same block structure and wiring, except with different point number)

7.3 Speed Up the Single-End Transistor Output Circuit

With the SINK structure in single-end output transistor circuit, when the transistor switches from ON to OFF, the junction capacitor between transistor CE electrodes should be charged to near the load voltage VDD before it can stop the current running through the photo-coupler inside the load, which increase the OFF time and decrease the response speed. This problem can be solved by adding a Dummy load to accelerate charging rate and speed up the working frequency of transistor output. For the transistor output, Dummy load that are added to the high- and medium-speed transistor output and generate a load current of 20~50mA is adequate. For low speed transistor where its driving capability (0.5A) but speed is concerned, adding a Dummy load only decreases its driving capability without any significant improvement and hence is not recommended. The following diagram shows how to add a Dummy load to SINK transistor output.



$$I = \frac{VDD}{R} = 20 \sim 50mA$$

7.4 Output Device Protection and Noise Suppression in DO Circuit

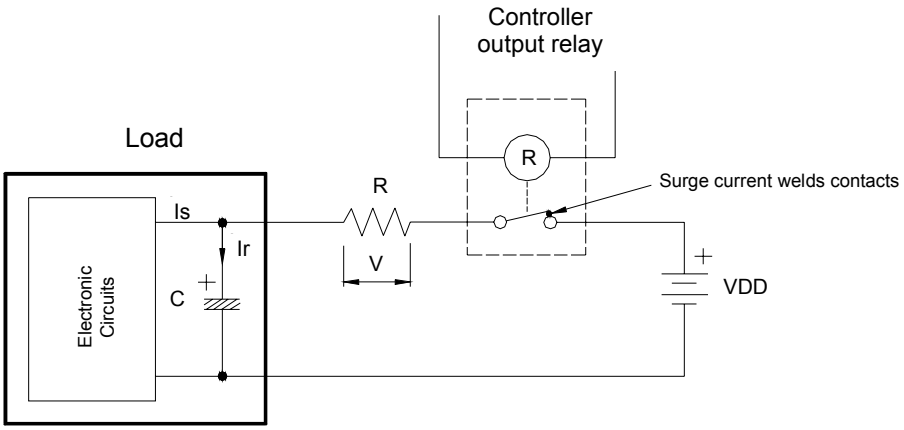
Since the digital output circuit is mainly used for the ON/OFF switching operation, the output components such as transistors can be deemed as kinds of switch components. Normally, counter-electromotive force voltages are generated during the ON/OFF operation of these switch components. The effect of surge counter-electromotive force voltages is particularly serious when heavy capacitive or inductive loads are incorporated, which may cause damage to the output components or generate noises in other electronic circuits and equipment. Special consideration should be given to transistors when they are used in high power applications or connected with capacitive or inductive loads and are described in the following:

7.4.1 Protection of Relay Output and Noise Suppression

Because the relay contacts are used to contact switch components having extremely low resistance, the surge current IR generated instantly upon turning on the relay is normally pretty strong (even if the steady load current is very small). Under such strong surge, the contact tends to melt and stick due to extreme temperature in such a way that the relay cannot trip when it is disconnected. In addition, when the relay connections are OFF, large di/dt is generated because of the instantaneous change from low resistance to open circuit (∞) soon after following the tripping of contact. As a result, an extremely strong counter-electromotive force voltage is induced, which creates sparks between the electrodes of two relay contacts and results in poor contact due to carbon deposits. Among those three output components, either in ON or OFF state, very serious interference can be caused by the surge current or the counter-electromotive force of the relay. The solutions to this problem are listed as follows:

A. Suppression of Surge Current

Connect a small resistor R in series to lower the surge current, but note that too large R will affect the driving capability or cause too much voltage drop.

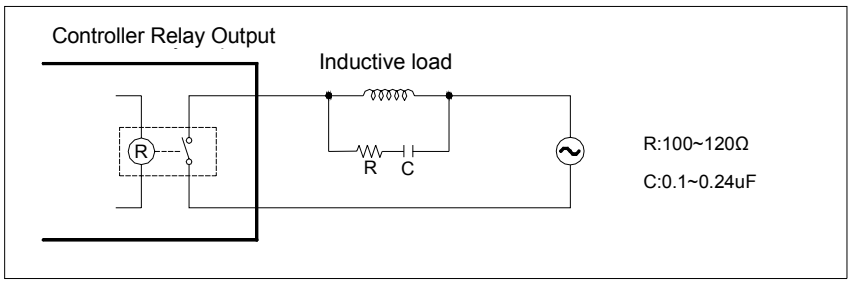


$$R \geqq \frac{VDD}{Ir\ max}$$ (note power dissipation $P=Is^2R$ and voltage drop $V=IsR$)

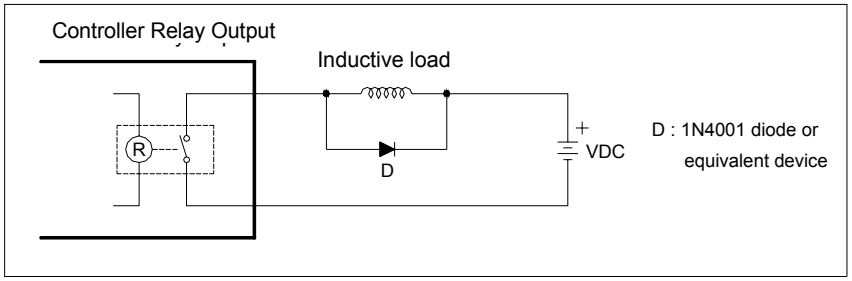
Ir max of relay in WSZ-Controller =5A

B. Suppression of Counter-Electromotive Force

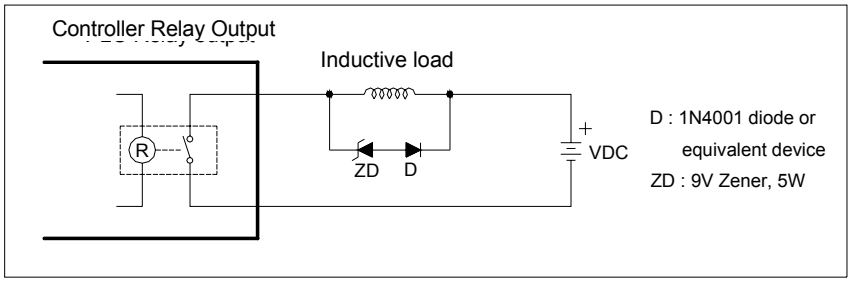
For the inductive load, whether in AC or DC power, suppression devices must be connected in parallel to both its ends to protect the relay contacts and lower noise interference. The schematic diagrams for AC and DC powers are shown below, respectively:



Scheme of AC power load



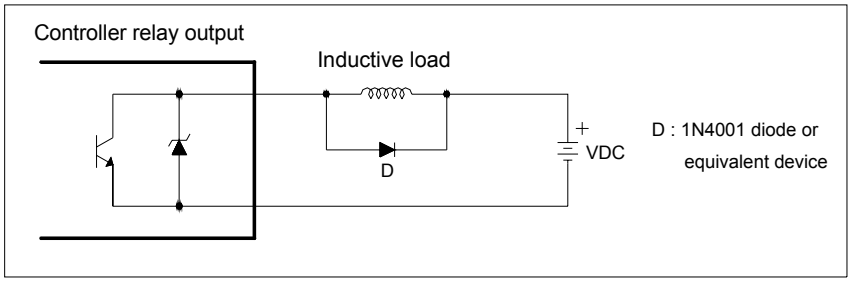
Suppress by a diode in DC power load (for low power)



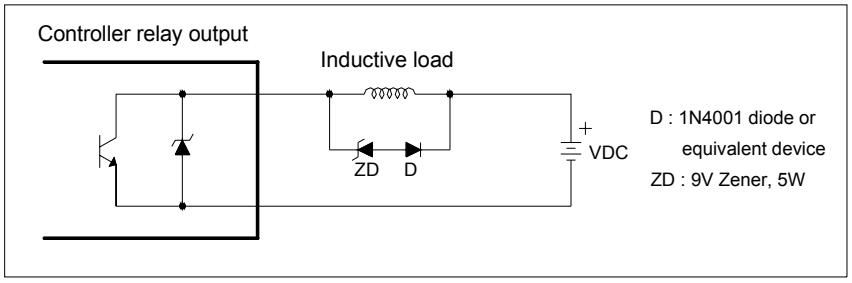
Suppress by a diode + Zener in DC power load (for high power and frequent ON/OFF)

7.4.2 Protection of Transistor Output and Noise Suppression

The transistor output already includes Zener diode for counter-electromotive force, which is sufficient for low power inductive load and medium frequency of ON/OFF application. In conditions of high power or frequent ON/OFF, please construct another suppression circuit to lower noise interference and prevent voltage from exceeding the limit or overheating that may damage the transistor output circuit.



Suppress by a diode (for low power)



Suppress by a diode + Zener (high power and frequent ON/OFF)

Chapter 8 Test Run, Monitoring and Maintenance

⚠ Warning

During maintenance, be sure to turn off the input power of controller in case the actions to touch any terminal on controller, or insert and extract accessories (e.g., expansion ribbon cables) are required. Otherwise, electric shock, short circuit, damaged controller or controller malfunction will be caused if the power is on.

8.1 Inspection after Wiring and before First Time Power on

1. Before power on, clean all unnecessary objects such as iron chippings and screws, and remove the dust cover sheet that surround the WSZ-controller.
2. Make sure that the input power and controller required power is of the same type. When input power is AC power, please pay attention to connect the hot wire (L) to the “L” terminal on controller and the ground wire (N) to the “N” terminal. Mistakenly connect to terminals other than “L” and “N” will result in electric shock, serious damage or malfunction.
3. Make sure the load power and controller output circuits are consistent. Connection of AC power to transistor output will damage controller or result in malfunction.
4. Make sure the 24VDC input and polarities of SINK in transistor output are consistent with those of your existing wiring. Any mismatch will result in failure of controller input and damage to the output circuit.

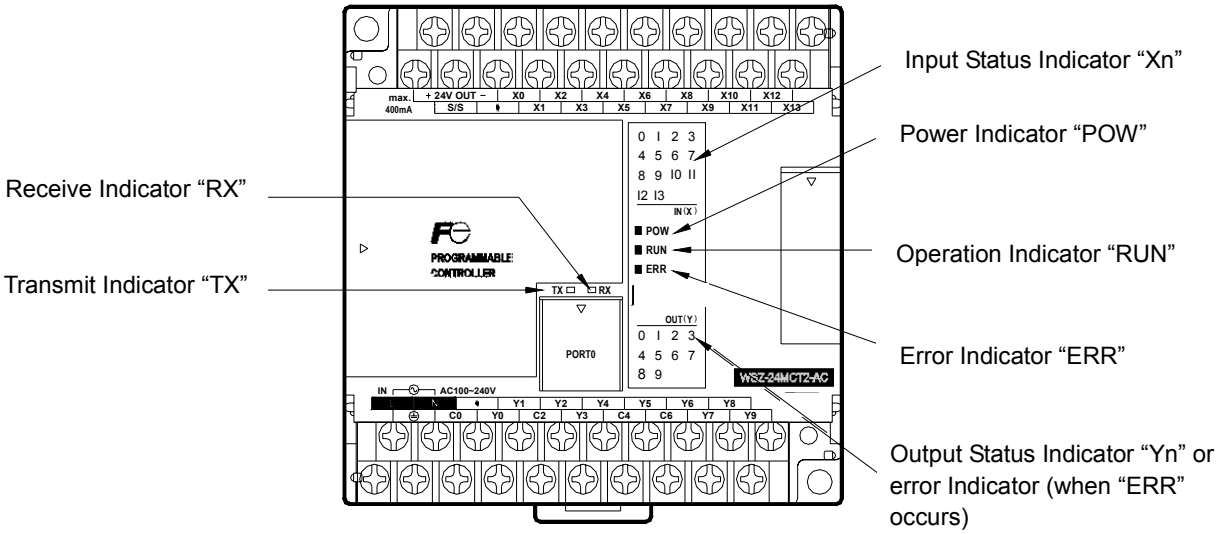
8.2 Test Run and Monitoring

The WSZ controller provides a convenient feature to Disable/Enable the I/O points by whole or individually. Namely, while controller performs the normal logic scan operation and I/O refreshment, it does not update the status of the disabled input points according to the actual external input. For the disabled output points, the result of logic scan can’t override the disable status of outputs, only the user can force the state to ‘on’ or ‘off’ in order to simulate its operation. The user only needs to utilize the disable function combined with Monitor to achieve simulating the input or WinProladder and observe the result. Upon the finish the simulation, revert all the inputs or outputs to enable state will bring back normal operation. Refer the instructions of WinProladder for the operation of RUN/STOP controller, Disable/Enable I/O and monitoring of I/O status and content of register.

⚠ Warning

The disable function is to let the input or output status out of program control and switched to the control of the user (tester) to freely set the disabled input or output to be ON or OFF. In normal controller operation, when dealing with input or output with safety issues (such as upper/lower limit of detected input or output emergency stop), the user must make sure whether it can be disabled or overridden to ON/OFF before starting the disable or override control, to avoid damage to equipment or harm to people.

8.3 LED Indicators on Controller Main Unit and Troubleshooting



Power Indicator〝POW〞

1. After the controller is power on, with correct power source and wiring, the〝POW〞LED indicator in the middle of the controller nameplate will turn on, indicating that power supply is normal. If the indicator is not on, please try to temporarily remove the wiring of 24VDC output power for Sensor. If the controller is back to normal, it means that the load on the power for the 24VDC input circuit is too large so that controller enters overload low voltage protection mode. (When controller enters overload low voltage protection mode, “POW” LED is off and there are slight and intermittent low frequency hissing sounds, from which one can tell if the 24VDC power is overloaded or shorted.)
2. When the above method still cannot turn on the 〝POW〞 LED, if it is confirmed that correct power input exists between controller power input L/N terminals or +/- (DC power), please send the unit to your local distributor for repair.

Operation Indicator〝RUN〞

As long as the CPU is working properly, in the STOP state, this indicator will go on and off for 2 seconds, respectively. When it’s in the RUN state, the indicator will go on and off for 0.25 seconds, respectively. To make controller enter into Run state, or switch from RUN to STOP state, it has to be done through the programmer (WinProladder). Once controller is set to RUN or STOP, it will keep that state even after power off. The only exception is, when using the ROM PACK, no matter if it’s running or stopped before power off, controller will automatically enter RUN state (with correct ROM PACK syntax check) when power is back. In normal operation of controller upon errors (e.g., errors in WDT timer and program), controller will automatically switch to STOP state and light the〝ERR〞error indicator. If it is a minor error, the RUN state can be resumed as long as the power is back after an outage. In case of serious errors, the controller cannot be operated again with the programmer until the problem is solved. If controller cannot be resumed to RUN state after all, please send it to your local distributor for repair.

Error Indicator〝ERR〞

In normal controller operation, either in RUN or STOP state, this indicator will not show any signal (off). If it is on, it means that the system has an error (e.g., WDT time-out, program error, communication error, etc.)
If it is constantly on, please reset the power. If the situation is still the same, it implies a hardware failure in CPU and has to be sent to the distributor for repair.
When the ERR indicator flashes with a 0.5 sec interval, it means that some anomaly occurs to controller. At the same time, status indicators Y0~Y3 switch to serve as indications of 15 error codes (the corresponding outputs are disabled), which are described in the following：

Y3	Y2	Y1	Y0	Error Code	Description
0	0	0	1	1	Application program contains the functions not supported by this CPU
0	0	1	0	2	Mismatch of PLC ID VS. program ID
0	0	1	1	3	Check sum error in LADDER program
0	1	0	0	4	System STACK abnormal
0	1	0	1	5	Watch-Dog occurs
0	1	1	0	6	Exceed main unit I/O
0	1	1	1	7	Syntax check error occurs
1	0	0	0	8	Expansion I/O modules over limit
1	0	0	1	9	Expansion I/O points over limit
1	0	1	0	10	System FLASH ROM CRC error
1	0	1	1	11	Reserved
1	1	0	0	12	Reserved
1	1	0	1	13	Reserved
1	1	1	0	14	Reserved
1	1	1	1	15	Reserved

Indicator on Transmit/Receive of Built-In Communication Port (Port0) “TX”、“RX”

These two LED indicators are used for the status of transmit/receive of the built-in communication port (Port0). The RX indicator (green) is for indication when controller receives external signals, while the TX indicator (red) is for indication when controller transmits signals, both of which are very helpful in monitoring communication condition and debugging. When controller communicates with external equipment (computer, programmer, intelligent peripherals, etc.), Port0 can only be used in slave mode (Port1~2 can be used in master mode). Therefore, during its operation, controller must first receive external signals (RX on) before it can transmit signals back to external equipment (TX on now). When the communication is fail, one can tell if it’s controller is not receiving signals or controller is not replying by looking at the these two indicators. The currents in these two LED are constant and their lighting duration is proportional to the reception or transmission time. The more received/transmitted data or the slower (bps) reception/transmission, the longer the reception/transmission time and so is the indication time (brighter visually). If in high speed but small amount of data, only short and dim brightness is observed. Therefore, the communication condition can be easily distinguished by these two indicators.

Indicator of Input Status〝Xn〞

When external input Xn is ON, the corresponding LED indicator Xn will be on, otherwise it will be off. If it fails to respond to external input, please check if the terminal wiring is securely connected, or measure the voltage between 〝Xn〞 and common 〝C〞 to see if it has a change of 0V/22V with ON/OFF of input. If it does, it means that an error

occurs in controller input circuit or LED indicator. Or you can locate the problem by using the monitor mode of the programmer to check if this input status works correspondingly with the external input state.

Indicator of Output Status〝Yn〞

When the Yn output of controller is ON, its corresponding output indicator will also be on and its external load will be ON. If ON/OFF condition of external load is inconsistent with output indicator, please check the wiring of the load, power, and terminal for secure connection. If the connection is good, then it should be the controller output component failure. The main reasons to cause the output component failure are:

1. Overload or short circuit that burns output component and results in permanent open or short circuit.
2. Not overloaded, but Inrush current from capacitive load welds relay contacts at〝ON〞, resulting in permanent ON, or burns transistor, resulting in permanent ON or OFF.
3. Not overloaded, but the inductive load without proper snubber circuit causes high voltage sparks between relay contact at〝OFF〞and generate carbon deposition, which separates contacts and causes permanent OFF or intermittent ON/OFF, or punches through transistor with high voltage, resulting in permanent ON or OFF.

8.4 Maintenance

WSZ controller itself has no user serviceable parts and all maintenance has to be conducted by professional personnel. During use, in case of any defective unit, please first try to find out the defect from the above error codes on the main unit, followed by performing maintenance over the entire unit or on the Board level. Send the unit that is still not functioning well to local distributors.

8.5 The Charge of Battery & Recycle of Used Battery

Every WSZ controller main units have inside one re-chargeable lithium battery to safely maintain program and data during main power shut down. Each lithium battery was fully charged when the WSZ controller ship out from the factory capable to retain program and data at least 3 months. There is risk to miss program and data when battery exhaust over 3 months, the users should mind the date marked on each WSZ controller.

In case exceeding 3 months, users can do battery re-charging by themselves through keeping WSZ controller be powered for over 12 hours then more 3 months can work smoothly on the data saving.

⚠ Warning

Any recharge, disassembly, heating, burning on defective or discarded battery is prohibited. Otherwise may cause danger of explosion or fire. The chemical material of battery will lead to environment pollution, easily throw away or treat as normal garbage is prohibited. Please follow after the local or government's regulation to make proper treatment on discard battery.

【*Instruction*】

Chapter 1 PLC Ladder Diagram

In this chapter, we would like to introduce you the basic principles of ladder diagram. If you are familiar with PLC Ladder Diagram, you may skip this chapter.

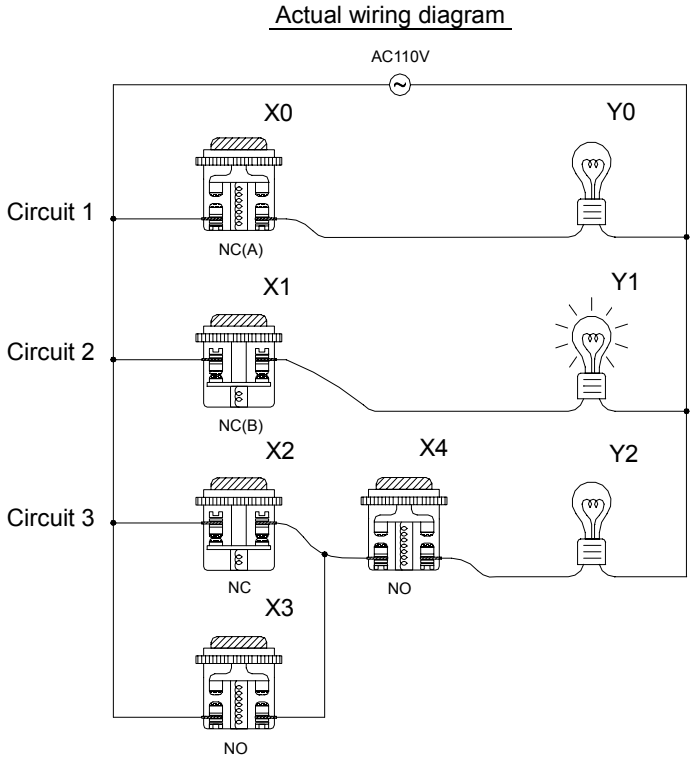
1.1 The Operation Principle of Ladder Diagram

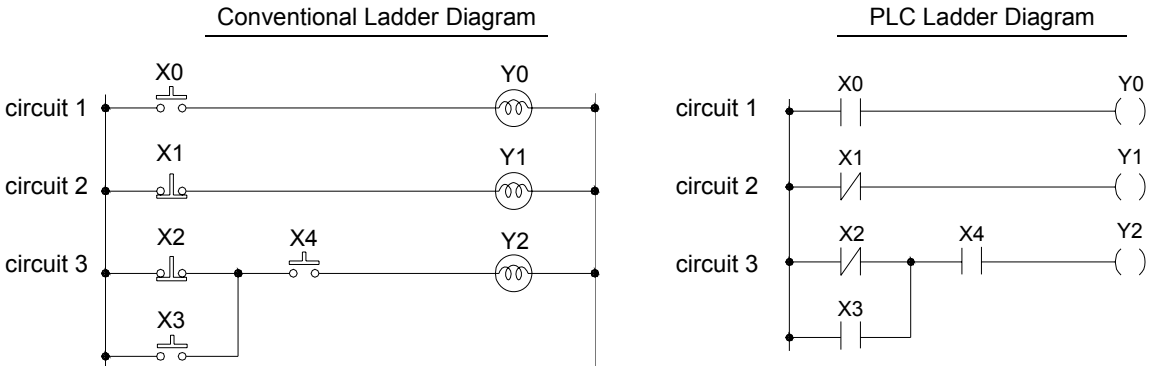
Ladder Diagram is a type of graphic language for automatic control systems it had been used for a long period since World War II. Until today, it is the oldest and most popular language for automatic control systems. Originally there are only few basic elements available such as A-contact (Normally OPEN), B contact (Normally CLOSE), output Coil, Timers and Counters. Not until the appearance of microprocessor based PLC, more elements for Ladder Diagram, such as differential contact, retentive coil (refer to page 1-6) and other instructions that a conventional system cannot provide, became available.

The basic operation principle for both conventional and PLC Ladder Diagram is the same. The main difference between the two systems is that the appearance of the symbols for conventional Ladder Diagram are more closer to the real devices, while for PLC system, symbols are simplified for computer display. There are two types of logic system available for Ladder Diagram logic, namely combination logic and sequential logic. Detailed explanations for these two logics are discussed below.

1.1.1 Combination Logic

Combination logic of the Ladder Diagram is circuits that combines one or more input elements in series or parallel and then send the results to the output elements, such as Coils, Timers/Counters, and other application instructions.





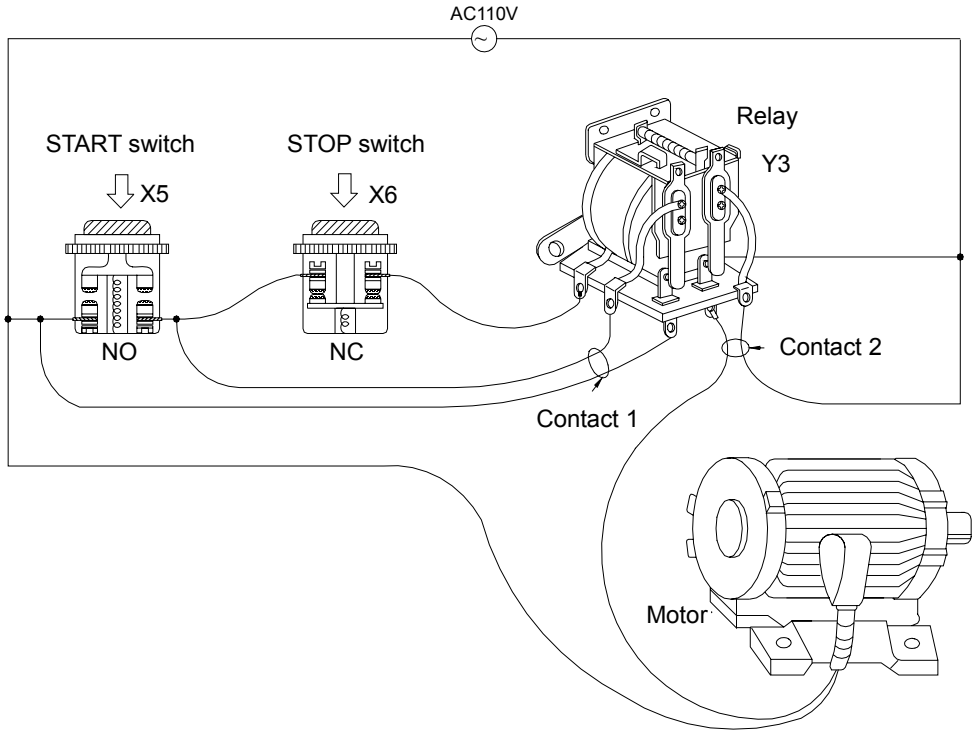
The above example illustrated the combination logic using the actual wiring diagram, conventional Ladder Diagram, and PLC Ladder Diagram. Circuit 1 uses a NO (Normally Open) switch that is also called "A" switch or contact. Under normal condition (switch is not pressed), the switch contact is at OFF state and the light is off. If the switch is pressed, the contact status turns ON and the light is on. In contrast, circuit 2 uses a NC (Normally Close) switch that is also called "B" switch or contact. Under normal condition, the switch contact is at ON state and the light is on. If the switch is pressed, the contact status turns OFF and the light also turns off.

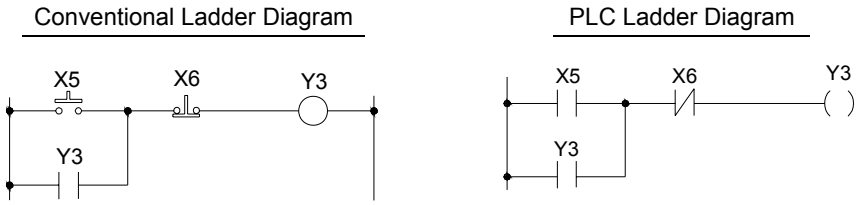
Circuit 3 contains more than one input element. Output Y2 light will turn on under the condition when X2 is closed or X3 switches to ON, and X4 must switch ON too.

1.1.2 Sequential Logic

The sequential logic is a circuit with feedback control; that is, the output of the circuit will be feedback as an input to the same circuit. The output result remains in the same state even if the input condition changes to the original position. This process can be best explained by the ON/OFF circuit of a latched motor driver as shown in below.

Actual wiring diagram





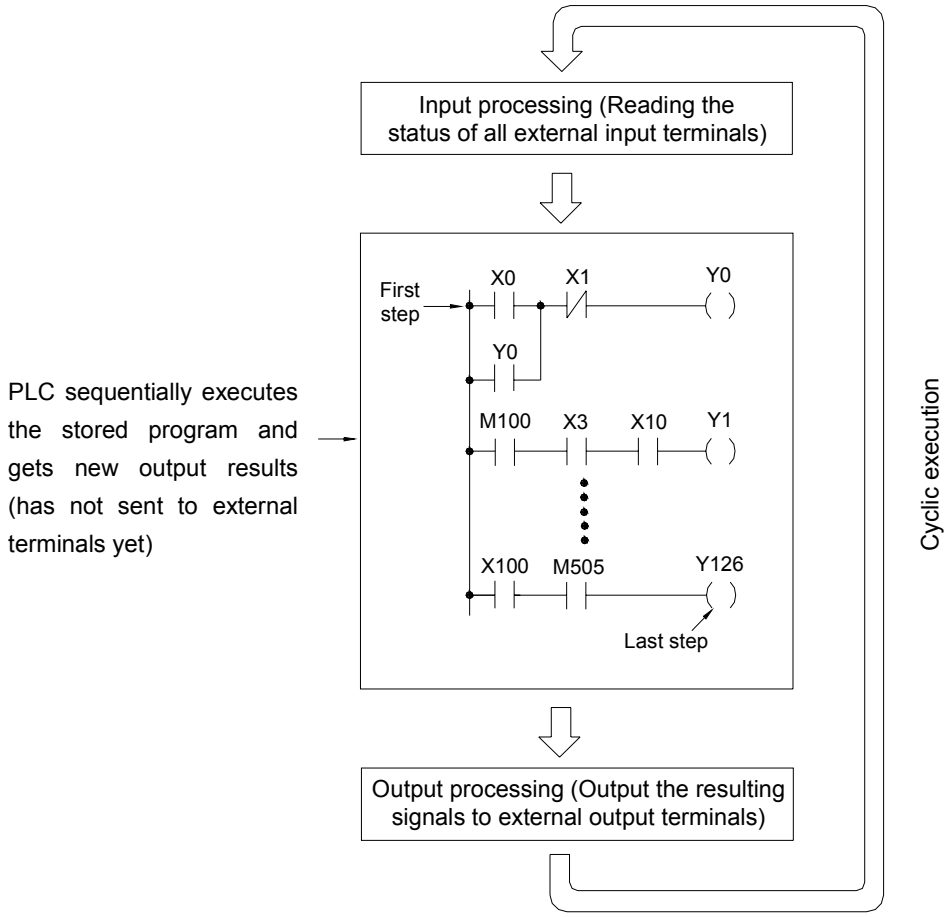
When we first connect this circuit to the power source, X6 switch is ON but X5 switch is OFF, therefore the relay Y3 is OFF. The relay output contacts 1and 2 are OFF because they belong to A contact (ON when relay is ON). Motor does not run. If we press down the switch X5, the relay turns ON as well as contacts 1and 2 are ON and the Motor starts. Once the relay turns ON, if we release the X5 switch (turns OFF), relay can retain its state with the feedback support from contact 1 and it is called Latch Circuit. The following table shows the switching process of the example we have discussed above.

	X5 switch (NO)	X6 switch (NC)	Motor（Relay）status
①	Released	Released	OFF
↓			
②	Pressed	Released	ON
↓			
③	Released	Released	ON
↓			
④	Released	Pressed	OFF
↓			
⑤	Released	Released	OFF

From the above table we can see that under different stages of sequence, the results can be different even the input statuses are the same. For example, let's take a look at stage ① and stage ③ , X5 and X6 switches are both released, but the Motor is ON (running) at stage ③ and is OFF (stopped) at stage ① . This sequential control with the feedback of the output to the input is a unique characteristic of Ladder Diagram circuit. Sometimes we call the Ladder Diagram a "Sequential Control Circuit" and the PLC a "Sequencer". In this section, we only use the A/B contacts and output coils as the example. For more details on sequential instructions please refer to chapter 4 - "Sequential Instructions."

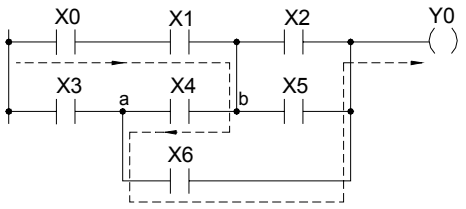
1.2 Differences Between Conventional and PLC Ladder Diagram

Although the basic operation principle for both conventional and PLC Ladder Diagram are the same, but in reality, PLC uses the CPU to emulate the conventional Ladder Diagram operations; that is, PLC uses scanning method to monitor the statuses of input elements and output coils, then uses the Ladder Diagram program to emulate the results which are the same as the results produced by the conventional Ladder Diagram logic operations. There is only one CPU, so the PLC has to sequentially examine and execute the program from its first step to the last step, then returns to the first step again and repeats the operation (cyclic execution). The duration of a single cycle of this operation is called the scan time. The scan time varies with the program size. If the scan time is too long, then input and output delay will occur. Longer delay time may cause big problems in controlling fast response systems. At this time, PLCs with short scan time are required. Therefore, scan time is an important specification for PLCs. Due to the advance in microcomputer and ASIC technologies nowadays the scan speed has been enhanced a great deal. For example, in the case of WSZ-Controller, the scan process speed of the contact of 1k step is 0.33ms. The following diagram illustrates the scanning process of a PLC Ladder Diagram.



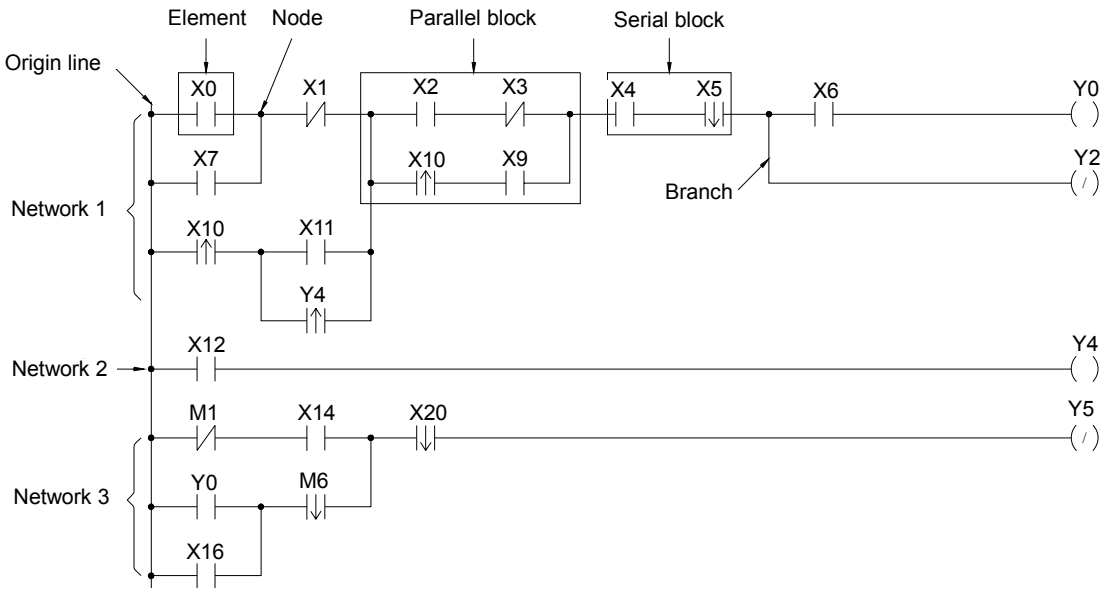
Besides the time scan difference mentioned above, the other difference between the conventional and PLC Ladder Diagram is “Reverse flow” characteristic. As shown in the diagram below, if X0, X1, X4 and X6 are ON, and the remaining elements are OFF. In a conventional Ladder Diagram circuit, a reverse flow route for output Y0 can be defined by the dashed line and Y0 will be ON. While for PLC, Y0 is OFF because the PLC Ladder Diagram scans from left to right, if X3 is off then CPU believes node “a” is OFF, although X4 and node “b” are all ON, since the PLC scan reaches X3 first. In other words, the PLC ladder can only allow left to right signal flow while conventional ladder can flow bi-directional.

Reverse flow of conventional Ladder diagram



1.3 Ladder Diagram Structure and Terminology

Sample Ladder Diagram



(Remark：The maximum size of WSZ-controller network is 16 rows X 22 columns)

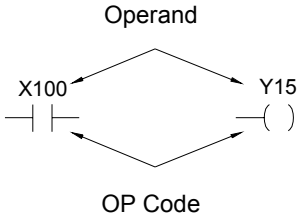
As shown above, the Ladder Diagram can be divided into many small cells. There are total 88 cells (8 rows X 11 columns) for this example Ladder Diagram. One cell can accommodate one element. A completed Ladder Diagram can be formed by connecting all the cells together according to the specific requirements. The terminologies related to Ladder Diagram are illustrated below.

① Contact

Contact is an element with open or short status. One kind of contact is called "Input contact"(reference number prefix with X) and its status reference from the external signals (the input signal comes from the input terminal block).The relation between the reference number and the contact status depends on the contact type. The contact elements provided by WSZ series controller include: A contact, B contact, up/down differential contacts and Open/Short contacts. Please refer to ③for more details.

② Origin-line: The starting line at the left side of the Ladder Diagram.

③ Element: Element is the basic unit of a Ladder Diagram. An element consists of two parts as shown in the diagram below. One is the element symbol which is called “OP Code” and another is the reference number part which is called “Operand”.



Element type	Symbol	Remark
A Contact （Normally OPEN）	□△△△△ ⊣ ⊢	□ can be X、Y、M、S、T、C（please refer to section 3.2）
B Contact （Normally CLOSE）	□△△△△ ⊣/⊢	
Up Differential Contact	□△△△△ ⊣↑⊢	□ can be X、Y、M、S
Down Differential Contact	□△△△△ ⊣↓⊢	
Open Circuit Contact	—○ ○—	
Short Circuit Contact	•———	
Output Coil	□△△△△ —( )	□ can be Y、M、S
Inverse Output Coil	□△△△△ —(/)	
Latching Output Coil	Y△△△ —(L)	

Remark：please refer to section 2.2 for the ranges of X、Y、M、S、T and C contacts. Please refer to section 4.2 for the characteristics of X、Y、M、S、T and C contacts.

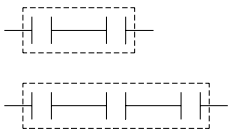
There are three special sequential instructions, namely OUT TRn, LD TRn and FOn, which were not displayed on the Ladder Diagram. Please refer to section 5.1.4 “Function Output FO”.

④ Block: a circuit consists of two or more elements.

There are two basic types of blocks:

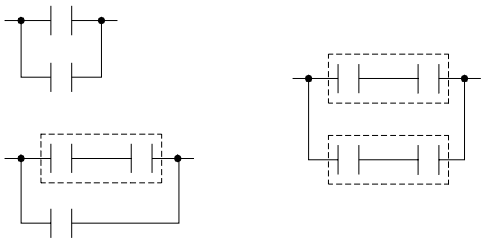
- Serial block：Two or more elements are connected in series to form a single row circuit.

Example:



- Parallel block: Parallel block is a type of a parallel closed circuit formed by connecting elements or serial blocks in parallel.

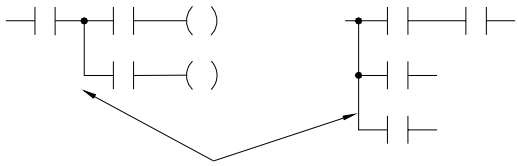
Example:



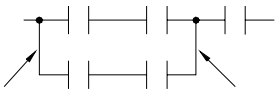
Remark: Complicated block can be formed by the combination of the single element, serial blocks and parallel blocks.

⑤ Branch: In any network, branch is obtained if the right side of a vertical line is connected with two or more rows of circuits.

Example:

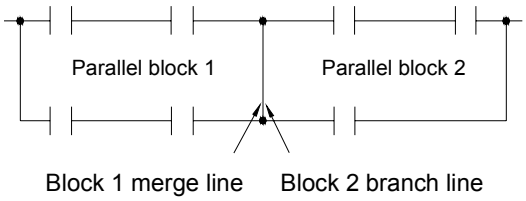


Merge line is defined as another vertical line at the right side of a branch line that merges the branch circuits into a closed circuit (forming a parallel block). This vertical line is called “Merge line”.



If both the right and the left sides of the vertical line are connected with two or more rows of circuits, then it is both a branch line and a merge line as shown in the example below.

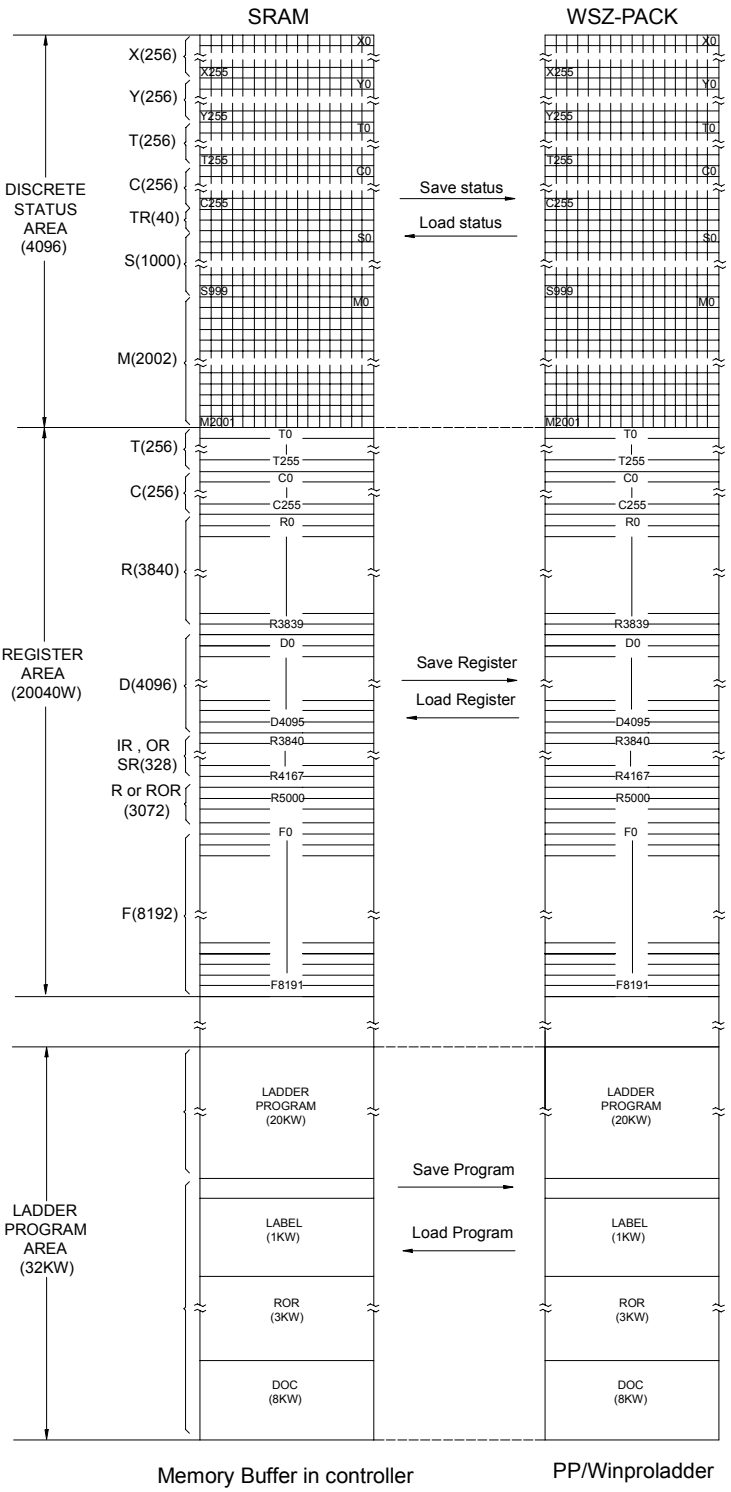
Example:



⑥ Network: Network is a circuit representing a specified function. It consists of the elements, branches, and blocks. Network is the basic unit in the Ladder Diagram which is capable of executing the completed functions, and the program of Ladder Diagram is formed by connecting networks together. The beginning of the network is the origin line. If two circuits are connected by a vertical line, then they belong to the same network. If there is no vertical line between the two circuits, then they belong to two different networks. Figure 1, shows three (1～3) networks.

Chapter 2 WSZ-Controller Memory Allocation

2.1 WSZ-Controller Memory Allocation



Remark:

1. When the Read Only Register (ROR) has been configured by the user, the contents of R5000～R8071 (depends on the quantity of configuration) will be loaded from the ROR's during each time of power up or changing from STOP to RUN mode.
   The user can access the ROR through the corresponding R5000～R8071. Write operation of function instructions are prohibited in this ROR area of corresponding R5000～R8071. The others of R5000～R8071 that have not been configured for ROR, they can work as general purpose registers.
2. There is a dedicated area of program memory to store the contents of Read Only Register.
   ROR can be configured up to 3072 words in maximum.

2.2 Digital and Register Allocations

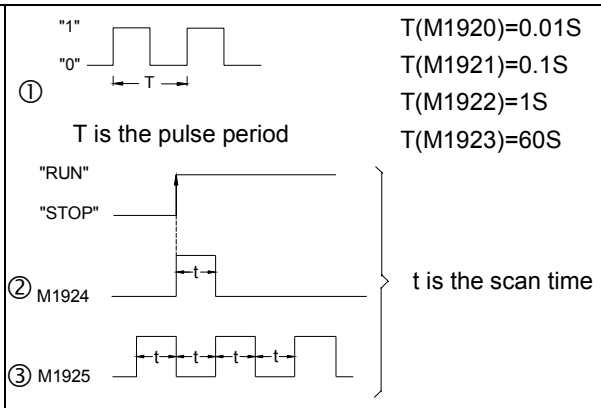
〝*〞 is default, user configurable

Type	Symbol	Item			Range		Remarks
Digital 《 Bit Status 》	X	Digital Input (DI)			X0～X255 (256)		Mapping to external digital I/O
	Y	Digital Output (DO)			Y0～Y255 (256)		
	TR	Temporary Relay			TR0～TR39 (40)		For branched points
	M	Internal Relays	Non-Retentive		M0～M799 (800)* M1400～M1911 (512)		M0～M1399 configurable as Non-retentive or Retentive, M1400～M1911 are fixed to Non-retentive
			Retentive		M800～M1399 (600)*		
		Special Relay			M1912～M2001 (90)		
	S	Step Relays	Non-Retentive		S0～S499 (500)*		S20～S499 configurable as Retentive
			Retentive		S500～S999 (500)*		S500～S999 configurable as Non-retentive
	T	Timer contact status			T0～T255 (256)		
	C	Counter contact status			C0～C255 (256)		
Register 《 Word Data 》	TMR	CV of Timer Register	0.01STime Base		T0～T49 (50)*		The quantity of each time base can be configured
			0.1S Time Base		T50～T199 (150)*		
			1S Time Base		T200～T255 (56)*		
	CTR	CV of Counter Register	16-bit	Retentive	C0～C139 (140)*		Configurable as Non-retentive
				Non-Retentive	C140～C199 (60)*		Configurable as Retentive
			32-bit	Retentive	C200～C239 (40)*		Configurable as Non-retentive
				Non-Retentive	C240～C255 (16)		Configurable as Retentive
	HR or DR	Data Registers		Retentive	R0～R2999 (3000)* D0～D3999 (4000)		R0～R3839 configurable as Non-retentive or Retentive, D0～D3999 are fixed to Retentive
				Non-Retentive	R3000～R3839 (840)*		
	HR or ROR			Retentive	R5000～R8071(3072)*		As general purpose registers if ROR not been configured.
				Read Only Registers	R5000～R8071(0)*		Configurable as ROR for recipe like application
	IR	Input Registers			R3840～R3903 (64)		Map to external AI Register input
	OR	Output Registers			R3904～R3967 (64)		Map to external AO /Register output
	SR	System Special Registers			R3968～R4167 (200) D4000～D4095 (96)		
	Special Register	0.1ms High-Speed Timer Register			R4152～R4154 (3)		
		HSC Registers	Hardware (4sets)		DR4096～DR4110 (4x4)		
			Software(4sets)		DR4112～DR4126 (4x4)		
		Calendar Registers	Minute	Second	R4129	R4128	
			Day	Hour	R4131	R4130	
			Year	Month	R4133	R4132	
				Week		R4134	
	FR	File Registers			F0～F8191(8192)		Need dedicated instruction to access
	XR	Index Registers			V,Z (2), P0～P9 (10)		

Remark: During power up or changing operation mode from STOP→RUN, all contents in non-retentive relays or registers will be cleared to 0; the retentive relays or registers will remain the same state as before.

2.3 Special Relay Details

Relay No.	Function	Description
1. Stop, Prohibited Control		
M1912	Emergency Stop control	• If ON, controller will be stopped and all outputs OFF. This bit will be cleared when power up or changing operation mode from STOP→RUN.
M1913	Disable external outputs control	• All external outputs are turn off but the status of Y0～Y255 inside the controller will not be affected.
M2001	Disable/Enable status retentive control	•If M2001 is 0 or enabled, the Disable/Enable status of all contacts will be reset to enable during power up or changing operation mode from STOP→RUN. •If M2001 is disabled and force ON, the Disable/Enable status & ON/OFF state of all contacts will remain as before during power up or changing operation mode from STOP→RUN. While testing, it may disable and force ON M2001 to keep the ON/OFF state of disabled contacts, but don't forget to enable the M2001 after testing.
2. CLEAR Control		
M1914	Clear Non-Retentive Relays	• Cleared When at 1
M1915	Clear Retentive Relays	• Cleared When at 1
M1916	Clear Non-Retentive Registers	• Cleared When at 1
M1917	Clear Retentive Registers	• Cleared When at 1
M1918	Master Control (MC) Selection	• If 0, the pulse activated functions within the master control loop will only be executed once at first 0→1 of master control loop. If 1, the pulse activated functions within the master control loop will be executed every time while changing 0→1 of master control loop.
M1919	Function output control	•If 0, the functional outputs of some function instructions will memory the output state, even these instructions not been executed. If 1, the functional output of some function instructions without the memory ability.
※ M1918/M1919 can be set to 0 or 1 at will around the whole program to meet the control requirements.		

Relay No.	Function	Description
3. Pulse Signals		
◤M1920 ◤M1921 ◤M1922 ◤M1923 ◤M1924 ◤M1925 ◤M1926	0.01S Clock pulse 0.1S Clock pulse 1S Clock pulse 60S Clock pulse ① Initial pulse (first scan) ② Scan clock pulses ③ =0, Controller is working at STOP mode =1, Controller is working at RUN mode	 T(M1920)=0.01S T(M1921)=0.1S T(M1922)=1S T(M1923)=60S
◤M1927	CTS input status of communication port 1	• 0：CTS True (ON) • 1：CTS False (OFF) • When communication port 1 is used to connect with the printer or modem, it can use this signal and a timer to detect whether the printer or the modem is ready.
4. Error Messages		
◤M1928 ◤M1929 ◤M1930 ◤M1931 ◤M1932 ◤M1933 ◤M1934 | ◤M1935	Reserved Reserved No expansion unit or exceed the limit on number of I/O points Immediate I/O not in the main unit range Unused System stack error Reserved	• 1: Indicating no expansion unit or exceed the limit on number of I/O points • 1: Indicating that Immediate I/O not in the main unit range and the main unit cannot RUN • 1: Indicating that system stack error
5. Port3～Port4 Controls（Not used）		
M1936 M1937 M1938 M1939	Port 3 busy indicator Port 3 finished indicator Port 4 busy indicator Port 4 finished indicator	• 0：Port 3 Busy • 1：Port 3 Ready • 1：Port 3 finished all communication transactions • 0：Port 4 Busy • 1：Port 4 Ready • 1：Port 4 finished all communication transactions

Relay No.	Function	Description
6. HSC0/HSC1 Controls		
M1940	HSC0 software Mask	• 1: Mask
M1941	HSC0 software Clear	• 1: Clear
M1942	HSC0 software Direction	• 0: Count-up, 1: Count-down
M1943	Reserved	
M1944	Reserved	
M1945	Reserved	
M1946	HSC1 software Mask	• 1: Mask
M1947	HSC1software Clear	• 1: Clear
M1948	HSC1 software Direction	• 0: Count-up, 1: Count-down
M1949	Reserved	
M1950	Port 3 status	• 1: Port 3 has received and transmitted a message（Not used）
M1951	Port 4 status	• 1: Port 4 has received and transmitted a message（Not used）
7. RTC Controls		
M1952	RTC setting	
M1953	±30 second Adjustment	
◤M1954	RTC installation checking	
◤M1955	Set value error	
8. Communication/Timing/Counting Controls		
M1956	Selection of Message Fame Interval Detection Time	• 0：Use system default value as Message Fame Interval Detection Time for Modbus RTU communication protocol • 1：Use the high byte value of R4148 as Message Fame Interval Detection Time for Modbus RTU protocol
M1957	The CV value control after the timer "Time Up"	• 0: The CV value will continue timing until the upper limit is met after “Time Up” • 1: The CV value will stop at the PV value after “Time Up” (User may control M1957 within the program to control the individual timer)
M1958	Communication port 2 High Speed Link mode selection	• 0: Set Port 2 to Normal Speed Link • 1: Set Port 2 to High Speed CPU Link ※M1958 is only effective at slave station
M1959	Modem dialing signal selection	• 0: Dialing by TONE when Port 1 connecting with Modem. • 1: Dialing by PULSE when Port 1 connecting with Modem.
M1960	Port 1 busy indicator	• 0：Port 1 Busy • 1：Port 1 Ready
M1961	Port 1 finished indicator	• 1：Port 1 finished all communication transactions
M1962	Port 2 busy indicator	• 0：Port 2 Busy • 1：Port 2 Ready
M1963	Port 2 finished indicator	• 1：Port 2 finished all communication transactions
M1964	Modem dialing control	• If Port 1 is connected with Modem, when signal 0→1 will dial the phone number; when signal 1→0 will hang-up the phone.

Relay No.	Function	Description
M1965	Dialing success flag	• 1: Indicating that dialing is successful (when Port 1 is connected with Modem).
M1966	Dialing fail flag	• 1: Indicating that dialing has failed (when Port 1 is connected with Modem).
M1967	Port 2 High Speed Link working mode selection	• 0: Continuous cycle. • 1: One cycle only. It will stop when the last communication transaction is completed (only effective at the master station).
M1968	Step program status	• 1: Indicating that there are more than 16 active steps in the step program at the same time.
M1969	Indirect addressing illegal write flag	• 1: Indicating that a function with index addressing attempts to write cross over the boundary of different type of data.
M1970	Port 0 status	• 1: Port 0 has received and transmitted a message
M1971	Port 1 status	• 1: Port1 has received and transmitted a message
M1972	Port 2 status	• 1: Port2 has received and transmitted a message
M1973	The CV value control after counting “Count-Up”	• 0: Indicating that the CV value will continue counting up to the upper limit after “Time-Up”. • 1: Indicating that the CV value will stop at the PV value after “Count-Up”（User may control M1973 within the program to control the individual counter）
M1974	RAMP function (FUN95) slope control	• 0: Time control for ramping • 1: Equivalent slope control for ramping
M1975	CAM function (FUN112) selection	• 1: For the circular applications where the electric CAM switch (FUN112) can support the wrap around situation like the angle from 359° cross to 0°
9. HSC2～HSC7 Controls		
M1976	HSC2 software Mask	• 1: Mask
M1977	HSC2 software Clear	• 1: Clear
M1978	HSC2 software Direction	• 0: Count-up, 1: Count-down
M1979	HSC3 software Mask	• 1: Mask
M1980	HSC3 software Clear	• 1: Clear
M1981	HSC3 software Direction	• 0: Count-up, 1: Count-down
M1982	HSC4 software Mask	• 1: Mask
M1983	HSC4 software Direction	• 0: Count-up, 1: Count-down
M1984	HSC5 software MASK	• 1: Mask
M1985	HSC5 software Direction	• 0: Count-up, 1: Count-down
M1986	HSC6 software Mask	• 1: Mask
M1987	HSC6 software Direction	• 0: Count-up, 1: Count-down
M1988	HSC7 software Mask	• 1: Mask
M1989	HSC7 software Direction	• 0: Count-up, 1: Count-down
M1990	Reserved	

Relay No.	Function	Description
10. PSO0～PSO3 Controls		
M1991	Selection of stopping the pulse output (FUN140, FUN147)	• 0：Immediately stop while stopping pulse output • 1：Slow down stop while stopping pulse output
M1992	PSO0 Busy indicator	• 0：PSO0 Busy • 1：PSO0 Ready
M1993	PSO1 Busy indicator	• 0：PSO1 Busy • 1：PSO1 Ready
M1994	PSO2 Busy indicator	• 0：PSO2 Busy • 1：PSO2 Ready
M1995	PSO3 Busy indicator	• 0：PSO3 Busy • 1：PSO3 Ready
M1996	PSO0 Finished indicator	• 1：PSO0 finished the last step of motion
M1997	PSO1 Finished indicator	• 1：PSO1 finished the last step of motion
M1998	PSO2 Finished indicator	• 1：PSO2 finished the last step of motion
M1999	PSO3 Finished indicator	• 1：PSO3 finished the last step of motion
M2000	Selection of Multi-Axis synchronization for High Speed Pulse Output (FUN140, FUN147)	• 1: Synchronized Multi-Axis

2.4 Special Registers Details

Register No.	Function	Description
R3840 | R3903	Input Registers CH0 : R3840 | | CH63 : R3903	For Analog inputs
R3904 | R3967	Output Registers CH0 : R3904 | | CH63 : R3967	For Analog outputs
R3968 | R3980	Define stimulate Modbus equipment	
R3981 | R3999	Reserved	
R4000	Reserved	
R4001	Reserved	
R4002	Reserved	
R4003 | R4004	Define FUN86 temperature reading value at starting/end address	
R4005	High Byte：Period of PWM =0, 2 seconds =1, 4 seconds =2, 8 seconds =3, 1 second =4, 16 seconds ≥5, 32 seconds Low Byte：Period of PID calculation =0, 2 seconds =1, 4 seconds =2, 8 seconds =3, 1 second =4, 16 seconds ≥5, 32 seconds	For PID temperature control
R4006	Threshold value of output ratio for heating/cooling loop abnormal detecting (Unit in %)	For PID temperature control
R4007	Threshold value of continuous time for heating/cooling loop abnormal detecting (Unit in second)	For PID temperature control
R4008	Maximum temperature for heating loop abnormal detecting	For PID temperature control
R4009	Temperature display in Celsius/Fahrenheit	=0, Celsius ;=1,Fahrenhelt

Register No.	Function	Description
R4010 | R4011	Installed temperature sensor flag	Each bit represents 1 sensor, if bit value = 1 means installed.
R4012 | R4013	PID Temperature control flag	Each bit represents 1 temperature point, if bit value = 1 means enable control.
R4014	Reserved	
R4015	Averaging of temperature value =0, no average on temperature =1, average by two readings =2, average by four readings =3, average by eight readings	
R4016	Reserved	
R4017	Reserved	
R4018	Reserved	
R4019	Number of PASSWORD Retry	
R4020	Control FUN148 instruction forbid to clockwise/anti-clockwise.	
R4021 | R4024	Reserved	
R4025 R4026 R4027 R4028	Total Expansion Input Registers Total Expansion Output Registers Total Expansion Digital Inputs Total Expansion Digital Outputs	
R4029	Reserved for system	
R4030 | R4039	Tables to save or read back the data registers into or from ROM Pack	When the ROM Pack being used to save the ladder program and data registers, these tables describes which registers will be written into the ROM Pack. The addressed registers will be initialized from ROM Pack while power up.
R4040	Reply delay time settings for Port 0 and Port 1	Low Byte：For Port 0 (Unit in ms) High Byte：For Port 1 (Unit in ms)
R4041	Reply delay time settings for Port 2 and Port 3	Low Byte：For Port 2 (Unit in ms) High Byte：For Port 3 (Unit in ms, Not used)
R4042	Reply delay time settings for Port 4	Low Byte：For Port 4 (Unit in ms, Not used) High Byte：Reserved for system
R4043	Port 3 Communication Parameters Register	Set Baud Rate, Data bit…of Port 3 (Not used)
R4044	Port 4 Communication Parameters Register	Set Baud Rate, Data bit…of Port 4 (Not used)
R4045	Transmission Delay & Receive Time-out interval time Setting, while Port 3 being used as the master of FUN151 or FUN150	Low Byte：Port 3 Receive Time-out interval time (Unit in 10ms, Not used) High Byte：Port 3 Transmission Delay (Unit in 10ms, Not used)

Register No.	Function	Description
R4046	Power up initialization mode selection of data registers that has been written into ROM Pack.	=5530H: Don't initialize the addressed data registers been written into ROM Pack while power up =Others : initialize the addressed data registers been written into ROM Pack while power up
R4047	Communication protocol setting for Port1～Port2	Set the Modbus RTU communication protocol
R4048	Transmission Delay & Receive Time-out interval time Setting, while Port 4 being used as the master of FUN151 or FUN150	Low Byte：Port 4 Receive Time-out interval time (Unit in 10ms, Not used) High Byte：Port 4 Transmission Delay (Unit in 10ms, Not used)
R4049	CPU Status Indication	=A55AH, Force CPU RUN =0, Normal Stop =1, Function(s) existed that CPU does not support =2, PLC ID not matched with Program ID =3, Ladder checksum error =4, System STACK error =5, Watch-Dog error =6, Immediate I/O over the CPU limitation =7, Syntax not OK =8, Qty of expansion I/O modules exceeds =9, Qty of expansion I/O points exceeds =10, CRC error of system FLASH ROM
R4050	Port 0 Communication Parameters Register	Set Baud Rate of Port 0
R4051	Reserved	
R4052	Indicator while writing ROM Pack	
R4053	Reserved	
R4054	Define the master station number of the High-Speed CPU Link network (FUN151 Mode 3)	If the master station number is 1,it can ignore this register. To set the master station number other than 1 should: Low Byte : Station number High Byte: 55H
R4055	Controller station number	• If high byte is not equal 55H, R4055 will show the station number of this controller • If want to set controller station number then R4055 should set to: Low Byte : Station number High Byte: 55H
R4056	High Byte :Reserved Low Byte: High speed pulse output frequency dynamic control	Low Byte: =5AH, can dynamically change the output frequency of High Speed Pulse Output
R4057	Power off counter	The value will be increased by 1 while power up
R4058	Error station number while Port 2 in High Speed CPU Link	Used by FUN151 Mode 3 of Port 2

Register No.	Function	Description
R4059	Error code while Port 2 in High Speed CPU LINK mode	Used by FUN151 Mode 3 of Port 2 High byte Low Byte R4059 | Err code | Err count | H Error code： 0AH, No response 01H, Framing Error 02H, Over-Run Error 04H, Parity Error 08H, CRC Error
R4060	Error code of PSO 0	The error codes are: 1: Parameter 0 error 2: Parameter 1 error 3: Parameter 2 error 4: Parameter 3 error 5: Parameter 4 error 6: Parameter 5 error 7: Parameter 6 error 8: Parameter 7 error 9: Parameter 8 error 10: Parameter 9 error 13: Parameter 12 error 14: Parameter 13 error 15: Parameter 14 error 30: Speed setting reference number error 31: Speed value error 32: Stroke setting reference number error 33: Stroke value error 34: Illegal positioning program 35: Step over 36: Step number exceeds 255 37: Highest frequency error 38: Idle frequency error 39: Movement compensation value too large 40: Movement value exceeds range 41: DRVC instruction not allow ABS addressing 42: DRVZ can't follow DRVC 43: Drive command error 50: Illegal operation mode of DRVZ 51: Illegal DOG input number 52: Illegal PG0 input number 53: Illegal CLR output number 60: Illegal linear interpolation command
R4061	Error code of PSO 1	Same as above
R4062	Error code of PSO 2	Same as above
R4063	Error code of PSO 3	Same as above
R4064 R4065	Being completed step number of positioning program	PSO 0 PSO 1

Register No.	Function	Description
R4066	Being completed step number of positioning program	PSO 2
R4067		PSO 3
R4068 R4069	FUN147 GP0 vector speed	
R4070 R4071	FUN147GP1 vector speed	
R4072	Pulse count remaining for output	Low Word of PSO 0
R4073		High Word of PSO 0
R4074		Low Word of PSO 1
R4075		High Word of PSO 1
R4076		Low Word of PSO 2
R4077		High Word of PSO 2
R4078		Low Word of PSO 3
R4079		High Word of PSO 3
R4080	Current output frequency	Low Word of PSO 0
R4081		High Word of PSO 0
R4082		Low Word of PSO 1
R4083		High Word of PSO 1
R4084		Low Word of PSO 2
R4085		High Word of PSO 2
R4086		Low Word of PSO 3
R4087		High Word of PSO 3
R4088	Current pulse position	Low Word of PSO 0
R4089		High Word of PSO 0
R4090		Low Word of PSO 1
R4091		High Word of PSO 1
R4092		Low Word of PSO 2
R4093		High Word of PSO 2
R4094		Low Word of PSO 3
R4095		High Word of PSO 3

Register No.	Function	Description
R4096	HSC0 current value Low Word	
R4097	HSC0 current value High Word	
R4098	HSC0 preset value Low Word	
R4099	HSC0 preset value High Word	
R4100	HSC1 current value Low Word	
R4101	HSC1 current value High Word	
R4102	HSC1 preset value Low Word	
R4103	HSC1 preset value High Word	
R4104	HSC2 current value Low Word	
R4105	HSC2 current value High Word	
R4106	HSC2 preset value Low Word	
R4107	HSC2 preset value High Word	
R4108	HSC3 current value Low Word	
R4109	HSC3 current value High Word	
R4110	HSC3 preset value Low Word	
R4111	HSC3 preset value High Word	
R4112	HSC4 current value Low Word	
R4113	HSC4 current value High Word	
R4114	HSC4 preset value Low Word	
R4115	HSC4 preset value High Word	
R4116	HSC5 current value Low Word	
R4117	HSC5 current value High Word	
R4118	HSC5 preset value Low Word	
R4119	HSC5 preset value High Word	
R4120	HSC6 current value Low Word	
R4121	HSC6 current value High Word	
R4122	HSC6 preset value Low Word	
R4123	HSC6 preset value High Word	
R4124	HSC7 current value Low Word	
R4125	HSC7 current value High Word	
R4126	HSC7 preset value Low Word	
R4127	HSC7 preset value High Word	
R4128	Second of calendar	
R4129	Minute of calendar	
R4130	Hour of calendar	
R4131	Day of calendar	
R4132	Month of calendar	
R4133	Year of calendar	
R4134	Day of week of calendar	
R4135	month ┼ minute	
◤ R4136	Current scan time	• Error < ±1ms
◤ R4137	Maximum scan time	• Re-calculate when controller changes from STOP to RUN
◤ R4138	Minimum scan time	

Register No.	Function	Description
R4139	CPU Status	Bit0 =0, Controller STOP =1, Controller RUN Bit1 , Reserved Bit2 =1, Ladder program checksum error Bit3 =0, Without ROM Pack =1, With ROM Pack Bit4 =1, Watch-Dog error Bit5 =1, MA model main unit (Not used) Bit6 =1, With ID protection Bit7 =1, Emergency stop Bit8 =1, Immediate I/O over range Bit9 =1, System STACK error Bit10 =1, ASIC failed Bit11 =1, Function not allowed (Use the instruction of non-support) Bit12 , Reserved Bit13 , Reserved Bit14 =1, With calendar Bit15 =1, MC main unit
R4140 R4141 R4142 R4143 R4144 R4145	Telephone Number	

Register No.	Function	Description
R4146	Port 1 Communication Parameters Register	Set Baud Rate, Data bit… of Port 1
R4147	Transmission Delay & Receive Time-out interval time Setting, while Port 1 being used as the master of FUN151 or FUN150	Low Byte：Port 1 Receive Time-out interval time (Unit in 10ms) High Byte：Port 1 Transmission Delay (Unit in 10ms)
R4148	Message Frame Detection Time Interval	.While the communication port being used as the master or slave of Modbus RTU protocol, the system will give the default time interval to identify each packet of receiving message; except this, the user can set this time interval through the high byte setting of R4148 and let M1956 be 1, to avoid the overlap of different packet of message frame. M1956=1, High Byte of R4148 is used to set the new message detection time interval for Port 1～Port 2 (Unit in ms) .While the communication port being used to communicate with the intelligent peripherals through FUN151 instruction, if the communication protocol without the end of text to separate each packet of message frame, it needs message detection time interval to identify the different packet. High byte of R4148 is used for this setting for Port 1～Port 2. (Unit in ms)
R4149	Modem Interface Setting & Port0 without checking of station number for Original external communication protocol	• High Byte of R4149: =55H, Remote-Diagnosis/Remote-CPU-Link by way of Port 1 through Modem connection, it supports user program controlled dial up function =AAH, Remote diagnosis by way of Port 1 through Modem connection, it supports Passive receiving & Active dialing operation mode =Others, without above function • Low Byte of R4149: =1, Port 0 without checking of station number for Original external communication protocol (communicating with HMI/SCADA) =Others, Port 0 checks station number, it allows multi-drop network for data acquisition.
R4150	Power on I/O service delay time setting	• Controller is ready for I/O service after this delay time while power up. The unit is in 0.01s. The default value is 100.
R4151	Circular 1mS time base timer	• The content of R4151 will be increased by 1 every 1ms. It can be used for a more precise timing application.
R4152 R4153 R4154	Low word of HSTA CV register High word of HSTA CV register PV register of HSTA	HSTA is high speed timer in 0.1 ms resolution. The HSTA can act as 32-bit cyclic timer or fixed time interrupt timer.

Register No.	Function	Description
R4155	Port 1 & Port 2 without station number checking for original external communication protocol	• Low Byte of R4155: =1, Port 1 without station number checking for original external communication protocol (communicating with HMI/SCADA) =Others, Port 1 checks station number, it allows multi-drop network for data acquisition • High Byte of R4155: =1, Port 2 without station number checking for original external communication protocol (communicating with MMI/SCADA) =Others, Port 2 checks station number, it allows multi-drop network for data acquisition
R4156	Port 3 & Port 4 without station number checking for original external communication protocol	• Low Byte of R4156 (Not used): =1, Port 3 without station number checking for original external communication protocol (communicating with MMI/SCADA) ≠1,Port 3 checks station number, it allows multi-drop network for data acquisition • High Byte of R4156 (Not used): =1, Port 4 without station number checking for original external communication protocol (communicating with MMI/SCADA) ≠1,Port 4 checks station number, it allows multi-drop network for data acquisition
R4157	Controller OS Version	
R4158	Port 2 Communication Parameters Register (Not for High Speed CPU Link)	Set Baud Rate, Data bit…of Port 2
R4159	Transmission Delay & Receive Time-out interval time Setting, while Port 2 being used as the master of FUN151 or FUN150	Low Byte：Port 2 Receive Time-out interval time (Unit in 10ms) High Byte：Port 2 Transmission Delay (Unit in 10ms)
R4160	Port2 RX/TX time out setting for High Speed CPU Link	High Byte of R4160 : =56H, User setting mode if the system default works not well, Low Byte of R4160 is used for this setting (Not suggest) =Others, system will give the default value according to the setting of R4161
R4161	Port 2 Communication Parameters Register (For High Speed CPU Link)	•Set Baud Rate, Parity…of Port 2 • Data bit is fixed to 8-bit • Baud Rate≧38400 bps
R4162	Fixed time interrupt enable/disable control	B7 B6 B5 B4 B3 B2 B1 B0 100ms 50ms 10ms 5ms 4ms 3ms 2ms 1ms Bit=0, interrupt enabled Bit=1, interrupt disabled

Register No.	Function	Description
R4163	Modem dialing control setting	• Low Byte of R4163 : =1, Ignore the dialing tone and the busy tone when dialing. =2, Wait the dialing tone but ignore the busy tone when dialing. =3, Ignore the dialing tone but detect the busy tone when dialing. =4, Wait the dialing tone and detect the busy tone when dialing. =Any other value treated as value equal 4. • High Byte of R4163 : The Ring count setting for Modem auto answer
R4164	V index register	
R4165	Z index register	
R4166	System used	
R4167	Model of main unit	• Low Byte of R4167: =3, 14I + 10O (WSZ-24MCT2-AC) =4, 20I + 12O (WSZ-32 MCT2-AC) =5, 24I + 16O (WSZ-40 MCT2-AC) =6, 36I + 24O (WSZ-60 MCT2-AC) • High Byte of R4167: =1, MC

Register No.	Function	Description
D4000	Port 1 User-defined Baud Rate Divisor (R4146 must be 56XFH)	Port 1 user-defined Baud Rate (1125~1152000 bps) D4000 = (18432000/Baud Rate) - 1
D4001	Port 2 User-defined Baud Rate Divisor (R4158 must be 56XFH)	Port 2 user-defined Baud Rate (1125~1152000 bps) D4001 = (18432000/Baud Rate) - 1
D4002	Port 3 User-defined Baud Rate Divisor (R4043 must be 56XFH)	Port 3 user-defined Baud Rate (1125~1152000 bps, Not used) D4002 = (18432000/Baud Rate) - 1
D4003	Port 4 User-defined Baud Rate Divisor (R4044 must be 56XFH)	Port 4 user-defined Baud Rate (1125~1152000 bps, Not used) D4003 = (18432000/Baud Rate) - 1
D4004	FUN30 PID resolution of analog input	=0, 14-bit format but valid 12-bit resolution =1,14-bit format and valid 14-bit resolution
D4005	FUN30 PID gain constant	KC=D4005/Pb
D4006 | D4042	Analog input valid bit and set times of average	
D4043 | D4045	Communication function setting	
D4046 | D4052	Reserved	
D4053 D4054	RTC chip RTC time setup	RTC chip is ISL1208, is able through D4054 to do time setup
D4055 | D4059	Reserved	
D4060 D4061 D4062 D4063	FUN147 GP0 error code FUN147 GP1 error code FUN147 the step number (positioning point) which has been completed of GP0 FUN147 the step number (positioning point) which has been completed of GP1	
D4064 | D4079	Reserved	

Register No.	Function	Description
D4080	P0 index register	
D4081	P1 index register	
D4082	P2 index register	
D4083	P3 index register	
D4084	P4 index register	
D4085	P5 index register	
D4086	P6 index register	
D4087	P7 index register	
D4088	P8 index register	
D4089	P9 index register	
D4090 | D4095	Reserved	

Remark: All the special relays or registers attached with "◤ " symbol shown in the above table are write prohibited.

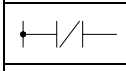
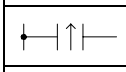
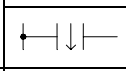
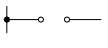
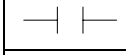
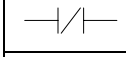
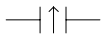
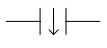
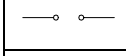
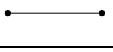
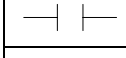
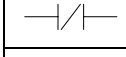
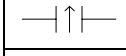
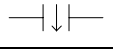
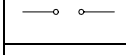
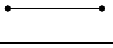
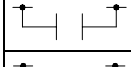
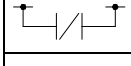
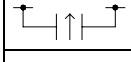
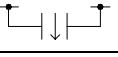
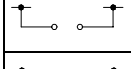
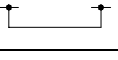
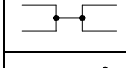
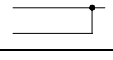
For the special relays attached with "◤ " symbol also has following characteristics

. Forced and Enable/Disable operation is not allowed.

. Can't be referenced by TU/TD transitional contact (contact will always open)

Chapter 3 WSZ-Controller Instruction Lists

3.1 Sequential Instructions

Instruction	Operand	Symbol	Function Descriptions	Execution Time	Instruction type
ORG	X,Y,M, S,T,C		Starting a network with a normally open (A) contact	0.33us	Network starting instructions
ORG NOT			Starting a network with a normally closed (B) contact		
ORG TU			Starting a network with a differential up (TU) contact	0.54us	
ORG TD			Starting a network with a differential down (TD) contact		
ORG OPEN			Starting a network with a open circuit contact	0.33us	
ORG SHORT			Starting a network with a short circuit contact		
LD	X,Y,M, S,T,C		Starting a relay circuit from origin or branch line with a normally open contact	0.33us	Origin or branch line starting instructions
LD NOT			Starting a relay circuit from origin or branch line with a normally closed contact		
LD TU			Starting a relay circuit from origin or branch line with a differential up contact	0.54us	
LD TD			Starting a relay circuit from origin or branch line with a differential down contact		
LD OPEN			Starting a relay circuit from origin or branch line with a open circuit contact	0.33us	
LD SHORT			Starting a relay circuit from origin or branch line with a short circuit contact		
AND	X,Y,M, S,T,C		Serial connection of normally open contact	0.33us	Serial connection instructions
AND NOT			Serial connection of normally closed contact		
AND TU			Serial connection of differential up contact	0.54us	
AND TD			Serial connection of differential down contact		
AND OPEN			Serial connection of open circuit contact	0.33us	
AND SHORT			Serial connection of short circuit contact		
OR	X,Y,M, S,T,C		Parallel connection of normally open contact	0.33us	Parallel connection instructions
OR NOT			Parallel connection of normally closed contact		
OR TU			Parallel connection of differential up contact	0.54us	
OR TD			Parallel connection of differential down contact		
OR OPEN			Parallel connection of open circuit contact	0.33us	
OR SHORT			Parallel connection of short circuit contact		
ANDLD			Serial connection of two circuit blocks	0.33us	Blocks merge instructions
ORLD			Parallel connection of two circuit blocks		

Instruction	Operand	Symbol	Function Descriptions	Execution Time	Instruction type
OUT	Y,M,S	—( )	Send result to coil	0.33us | 1.09us	Coil output instruction
OUT NOT		—(/)	Send inverted result to coil		
OUT L	Y	—(L)	Send result to an external output coil and appoint it as of retentive type		
OUT	TR		Save the node status to a temporary relay	0.33us	Node operation instruction
LD			Load the temporary relay		
TU		—↑—	Take the transition up of the node status	0.33us	
TD		—↓—	Take the transition down of the node status	0.33us	
NOT		—/—	Invert the node status	0.33us	
SET		—(S)	Set a coil	0.33us | 1.09us	
RST		—(R)	Reset a coil	0.33us | 1.09us	

● The 36 sequential instructions listed above are all applicable to every models of WSZ-controller.

3.2 Function Instructions

There are more than 100 different WSZ-Controller function instructions. If put the “D” and “P” derivative instructions into account, the total number of instructions is over 300. On top of these, many function instructions have multiple input controls (up to 4 inputs) which can have up to 8 different types of operation mode combinations. Hence, the size of WSZ-controller instruction sets is in fact not smaller than that of a large controller. Having powerful instruction functions, though may help for establishing the complicated control applications, but also may impose a heavy burden on those users of small type controller’s. For ease of use, WSZ controller function instructions are divided into two groups, the Basic function group which includes 26 commonly used function instructions and 4 SFC instructions and the advanced function group which includes other more complicated function instructions, such as high-speed counters and interrupts. This will enable the beginners and the non-experienced users to get familiar with the basic function very quickly and to assist experienced users in finding what they need in the advanced set of function instructions.

The instructions attached with “🗐” symbol are basic functions which amounts to 26 function instructions and 4 SFC instructions. All the basic functions will be explained in next chapter. The details for the reset of functions please refer advanced manual.

■ General Timer/Counter Function Instructions

FUN No.	Name	Operand	Derivative Instruction	Function descriptions
	T nnn	PV		General timer instructions (“nnn” range 0～255)
	C nnn	PV		General counter instructions (“nnn” range 0～255)
7	UDCTR	CV,PV	D	16-Bit or 32-Bit up/down counter

■ Single Operand Function Instructions

4	DIFU	D		To get the up differentiation of a D relay and store the result to D
5	DIFD	D		To get the down differentiation of a D relay and store the result to D
10	TOGG	D		Toggle the status of the D relay

■ Setting/Resetting

	SET	D	DP	Set all bits of register or a discrete point to 1
	RST	D	DP	Clear all bits of register or a discrete point to 0
114	Z-WR	D	P	Zone set or clear

■ SFC Instructions

	STP	Snnn		STEP declaration
	STPEND			End of the STEP program
	TO	Snnn		STEP divergent instruction
	FROM	Snnn		STEP convergent instruction

■ Mathematical Operation Instructions

11	(+)	Sa,Sb,D	DP	Perform addition of Sa and Sb and then store the result to D
12	(−)	Sa,Sb,D	DP	Perform subtraction of Sa and Sb and then store the result to D
13	(*)	Sa,Sb,D	DP	Perform multiplication of Sa and Sb and then store the result to D
14	(／)	Sa,Sb,D	DP	Perform division of Sa and Sb and then store the result to D
15	(+1)	D	DP	Adds 1 to the D value
16	(−1)	D	DP	Subtracts 1 from the D value
23	DIV48	Sa,Sb,D	P	Perform 48 bits division of Sa and Sb and then store the result to D
24	SUM	S,N,D	DP	Take the sum of the successive N values beginning from S and store it in D
25	MEAN	S,N,D	DP	Take the mean average of the successive N values beginning from S and store it in D
26	SQRT	S,D	DP	Take the square root of the S value and store it in D
27	NEG	D	DP	Take the 2's complement (negative number) of the D value and store it back in D
28	ABS	D	DP	Take the absolute value of D and store it back in D
29	EXT	D	P	Take the 16 bit numerical value and extend it to 1 32 bit numerical value (value will not change)
30	PID	TS,SR,OR, PR,WR		PID operation
31	CRC	MD,S,N,D	P	CRC16 checksum calculation
32	ADCNV	PL,S,N,D		Offset and full scale conversion

FUN No.	Name	Operand	Derivative Instruction	Function descriptions
33	LCNV	Md,S,Ts,D,L	P	Linear Conversion
34	MLC	Rs,SI,Tx,Ty, TI,D	P	Multiple Linear Conversion
200	I→F	S,D	DP	Integer to floating point number conversion
201	F→I	S,D	DP	Floating point number to integer conversion
202	FADD	Sa,Sb,D	P	Addition of floating point number
203	FSUB	Sa,Sb,D	P	Subtraction of floating point number
204	FMUL	Sa,Sb,D	P	Multiplication of floating point number
205	FDIV	Sa,Sb,D	P	Division of floating point number
206	FCMP	Sa,Sb	P	Comparison of floating point number
207	FZCP	S,SU,SL	P	Zone comparison of floating point number
208	FSQR	S,D	P	Square root of floating point number
209	FSIN	S,D	P	SIN trigonometric function
210	FCOS	S,D	P	COS trigonometric function
211	FTAN	S,D	P	TAN trigonometric function
212	FNEG	D	P	Change sign of floating point number
213	FABS	D	P	Take absolute value of floating point number
214	FLN	S,D	P	Floating point napierian logarithm
215	FEXP	S,D	P	Floating point nature power function
216	FLOG	S,D	P	Floating point logarithm
217	FPOW	Sy, Sx,D	P	Floating point power function
218	FASIN	S,D	P	Floating point arc sine function
219	FACOS	S,D	P	Floating point arc cosine function
220	FATAN	S,D	P	Floating point arc tangent function

■ Logical Operation Instructions

18	AND	Sa,Sb,D	DP	Perform logical AND for Sa and Sb and store the result to D
19	OR	Sa,Sb,D	DP	Perform logical OR for Sa and Sb and store the result to D
35	XOR	Sa,Sb,D	DP	Take the result of the Exclusive OR logical operation made between Sa and Sb, and store it in D
36	XNR	Sa,Sb,D	DP	Take the result of the Exclusive OR logical operation made between Sa and Sb, and store it in D

■ Comparison Instructions

17	CMP	Sa,Sb	DP	Compare the data at Sa and data at Sb and output the result to function outputs (FO)
37	ZNCMP	S,S_U,S_L	DP	Compare S with the zones formed by the upper limit S_U and lower limit S_L, and set the result to FO0~FO2

■ In Line Comparison Instructions

FUN No.	Name	Operand	Derivative instruction	Function descriptions
170	＝	Sa,Sb	D	Equal to compare
171	＞	Sa,Sb	D	Greater than compare
172	＜	Sa,Sb	D	Less than compare
173	＜＞	Sa,Sb	D	Not equal to compare
174	＞＝	Sa,Sb	D	Greater than or equal to compare
175	＝＜	Sa,Sb	D	Less than or equal to compare

■ Data Movement Instructions

8	MOV	S,D	DP	Transfer the W or DW data specified at S to D
9	MOV/	S,D	DP	Invert the W or DW data specified at S, and then transfers the result to D
40	BITRD	S,N	DP	Read the status of the bits specified by N within S, and send it to FO0
41	BITWR	D,N	DP	Write the INB input status into the bits specified by N within D
42	BITMV	S,Ns,D,Nd	DP	Write the status of bit specified by N within S into the bit specified by N within D
43	NBMV	S,Ns,D,Nd	DP	Write the Ns nibble within S to the Nd nibble within D
44	BYMV	S,Ns,D,Nd	DP	Write the byte specified by Ns within S to the byte specified by Nd within D
45	XCHG	Da,Db	DP	Exchange the values of Da and Db
46	SWAP	D	P	Swap the high-byte and low-byte of D
47	UNIT	S,N,D	P	Take the nibble 0 (NB0) of the successive N words starting from S and combine the nibbles sequentially then store in D
48	DIST	S,N,D	P	De-compose the word into successive N nibbles starting from nibble 0 of S, and store them in the NB0 of the successive N words starting from D
49	BUNIT	S,N,D	P	Low byte of words re-unit
50	BDIST	S,N,D	P	Words split into multi-byte
160	RW-FR	Sa,Sb,Pr,L	DP	File register access
161	WR-MP	S, Bk,Os, Pr,L,WR		Write memory pack
162	RD- MP	Bk,Os,Pr L,D PR,WR	P	Read memory pack

■ Shifting/Rotating Instructions

FUN No.	Name	Operand	Derivative instruction	Function descriptions
6	BSHF	D	DP	Shift left or right 1 bit of D register
51	SHFL	D,N	DP	Shift left the D register N bits and move the last shifted out bits to OTB. The empty bits will be replaced by INB input bit
52	SHFR	D,N	DP	Shift right the D register N bits and move the last shifted out bits to OTB, The empty bits will be replaced by INB input bit
53	ROTL	D,N	DP	Rotate left the D operand N bits and move the last rotated out bits to OTB
54	ROTR	D,N	DP	Rotate right the D operand N bits and move the last rotated out bits to OTB

■ Code Conversion Instruction

20	→BCD	S,D	DP	Convert binary data of S into BCD data and store the result to D
21	→BIN	S,D	DP	Convert BCD data of S into binary data and store the result to D
55	B→G	S,D	DP	Binary to Gray code conversion
56	G→B	S,D	DP	Gray code to Binary conversion
57	DECOD	S,Ns,N_L,D	P	Decode the binary data formed by N_L bits starting from Ns bit within S, and store the result in the register starting from D
58	ENCOD	S,Ns,N_L,D	P	Encoding the N_L bits starting from the Ns bit within S, and store the result in D
59	→7SG	S,N,D	P	Convert the N+1 number of nibble data within S, into 7 segment code, then store in D
60	→ASC	S,D	P	Write the constant string S (max. 12 alpha-numeric or symbols) into the registers starting from D
61	→SEC	S,D	P	Convert the time data (hours, minutes, seconds) of the three successive registers starting from S into seconds data then store to D
62	→HMS	S,D	P	Convert the seconds data of S into time data (hours, minutes, seconds) and store the data in the three successive registers starting from D
63	→HEX	S,N,D	P	Convert the successive N ASCII data starting from S into hexadecimal data and store them to D
64	→ASCⅡ	S,N,D	P	Convert the successive N hexadecimal data starting from S into ASCII codes and store them to D

■ Flow Control Instructions

FUN No.	Name	Operand	Derivative instruction	Function descriptions
0	MC	N		The start of master control loop
1	MCE	N		The end of master control loop
2	SKP	N		The start of skip loop
3	SKPE	N		The end of skip loop
	END			End of Program
22	BREAK		P	Exit from FOR-NEXT loop
65	LBL	1～6 alphanumeric		Define the label with 1~6 alphanumeric characters
66	JMP	LBL	P	Jump to LBL label and continues the program execution
67	CALL	LBL	P	Call the sub-program begin with LBL label
68	RTS			Return to the calling main program from sub-program
69	RTI			Return to interrupted main program from sub-program
70	FOR	N		Define the starting point of the FOR Loop and the loop count N
71	NEXT			Define the end of FOR loop

■ I/O Function Instructions

74	IMDIO	D,N	P	Update the I/O signal on the main unit immediately
76	TKEY	IN,D,KL	D	Convenient instruction for 10 numeric keys input
77	HKEY	IN,OT,D,KL, WR	D	Convenient instruction for 16 keys input
78	DSW	IN,OT,D	D	Convenient instruction for digital switch input
79	7SGDL	S,OT,N, WR	D	Convenient instruction for multiplexing 7-segment display
80	MUXI	IN,OT,N,D, WR		Convenient instruction for multiplexing input instruction
81	PLSO	MD, Fr, PC UY,DY,HO	D	Pulse output function (for bi-directional drive of step motor)
82	PWM	TO,TP,OT		Pulse width modulation output function
83	SPD	S,TI,D		Speed detection function
84	TDSP	MD,S,Ns, NI,D,Nd		7/16-segment LED display control
86	TPCTL	Md,Yn,Sn, Zn,Sv,Os, PR,IR,DR, OR,WR		PID Temperature control
139	HSPWM	PW,OP,RS, PN,OR,WR		Hardware PWM pulse output

■ Cumulative Timer Function Instructions

FUN No.	Name	Operand	Derivative instruction	Function descriptions
87	T.01S	CV,PV	D	Cumulative timer using 0.01s as the time base
88	T.1S	CV,PV	D	Cumulative timer using 0.1s as the time base
89	T1S	CV,PV	D	Cumulative timer using 1s as the time base

■ Watch Dog Timer Control Function Instructions

90	WDT	N	P	Set the WDT timer time out time to N ms
91	RSWDT		P	Reset the WDT timer to 0

■ High Speed Counter Control Function Instructions

92	HSCTR	CN	P	Read the current CV value of the hardware HSC, HSC0～HSC3, or HST on SoC to the corresponding CV register in the controller respectively
93	HSCTW	S,CN,D	P	Write the CV or PV register of HSC0～HSC3 or HST in the controller to CV or PV register of the hardware HSC or HST on SoC respectively

■ Report Function Instructions

94	ASCWR	MD,S,Pt		Parse and generate the report message based on the ASCII formatted data starting from the address S. Then report message will send to port1

■ Ramp Function Instructions

95	RAMP	Tn,PV,SL, SU,D		Ascending/Descending convenient instruction
98	RAMP2	Om,Ta Td,Rt Rc,WR		Tracking type ramp function for D/A output

■ Communication Function Instructions

150	M-Bus	MD,S,Pt		Modbus protocol communication
151	CLINK	Pt,MD,SR, WR		Original/Generic protocol communication

■ Table Function Instructions

FUN No.	Name	Operand	Derivative instruction	Function descriptions
100	R→T	Rs,Td,L,Pr	DP	Store the Rs value into the location pointed by the Pr in Td
101	T→R	Ts,L,Pr,Rd	DP	Store the value at the location pointed by the Pr in Ts into Rd
102	T→T	Ts,Td,L,Pr	DP	Store the value at the location pointed by the Pr in Ts into the location pointed by the Pr in Td
103	BT_M	Ts,Td,L	DP	Copy the entire contents of Ts to Td
104	T_SWP	Ta,Tb,L	DP	Swap the entire contents of Ta and Tb
105	R-T_S	Rs,Ts,L,Pr	DP	Search the table Ts to find the location with data different or equal to the value of Rs. If found store the position value into the Pr
106	T-T_C	Ta,Tb,L,Pr	DP	Compare two tables Ta and Tb to search the entry with different or same value. If found store the position value into the Pr
107	T_FIL	Rs,Td,L	DP	Fill the table Td with Rs
108	T_SHF	IW,Ts,Td, L,OW	DP	Store the result into Td after shift left or right one entry of table Ts. The shift out data is send to OW and the shift in data is from IW
109	T_ROT	Ts,Td,L	DP	Store the result into Td after shift left or right one entry of table Ts.
110	QUEUE	IW,QU,L, Pr,OW	DP	Push IW into QUEUE or get the data from the QUEUE to OW (FIFO)
111	STACK	IW,ST,L, Pr,OW	DP	Push IW into STACK or get the data from the STACK to OW (LIFO)
112	BKCMP	Rs,Ts,L,D	DP	Compare the Rs value with the upper/lower limits of L, constructed by the table Ts, then store the comparison result of each pair into the relay designated by D (DRUM)
113	SORT	S,D,L	DP	Sorting the registers starting from S length L and store the sorted result to D

■ Matrix Instructions

120	MAND	Ma,Mb,Md,L	P	Store the results of logic AND operation of Ma and Mb into Md
121	MOR	Ma,Mb,Md,L	P	Store the results of logic OR operation of Ma and Mb into Md
122	MXOR	Ma,Mb,Md,L	P	Store the results of logic Exclusive OR operation of Ma and Mb into Md
123	MXNR	Ma,Mb,Md,L	P	Store the results of logic Exclusive OR operation of Ma and Mb into Md
124	MINV	Ms,Md ,L	P	Store the results of inverse Ms into Md
125	MCMP	Ma,Mb,L Pr	P	Compare Ma and Mb to find the location with different value, then store the location into Pr
126	MBRD	Ms,L,Pr	P	Read the bit status pointed by the Pr in Ms to the OTB output
127	MBWR	Md,L,Pr	P	Write the INB input status to the bits pointed by the Pr in Ms
128	MBSHF	Ms,Md,L	P	Store the results to Md after shift one bit of the Ms. Shifted out bit will appear at OTB and the shift in bits comes from INB
129	MBROT	Ms,Md,L	P	Store the results to Md after rotate one bit of the Ms. Rotated out bit will appear at OTB.
130	MBCNT	Ms,L,D	P	Calculate the total number of bits that are 0 or 1 in Ms, then store the results into D

■ NC Positioning Instruction

FUN No.	Name	Operand	Derivative instruction	Function descriptions
140	HSPSO	Ps,SR,WR		HSPSO instruction of NC positioning control
141	MPARA	Ps,SR		Parameter setting instruction of NC positioning control
142	PSOFF	Ps	P	Stop the pulse output of NC positioning control
143	PSCNV	Ps,D	P	Convert the Ps positions of NC positioning to mm, Inch or Deg
147	MHSPO	Gp,SR,WR,		Multi-Axis high speed pulse output
148	MPG	Sc,Ps,Fo,Mr, WR		Manual pulse generator for positioning

■ Disable/Enable Control of Interrupt or Peripheral

145	EN	LBL	P	Enable HSC, HST, external INT or peripheral operation
146	DIS	LBL	P	Disable HSC, HST, external INT or peripheral operation

■ System Instruction

190	STAT	Gp, D		Read system status

Chapter 4 Sequential Instructions

The sequential instructions of WSZ-controller shown in this chapter are also listed in section 3.1. Please refer to Chapter 1, "PLC Ladder diagram", for the coding rules in applying those instructions. In this chapter, we only introduce the applicable operands, ranges and element characteristics, functionality.

4.1 Valid Operand of Sequential Instructions

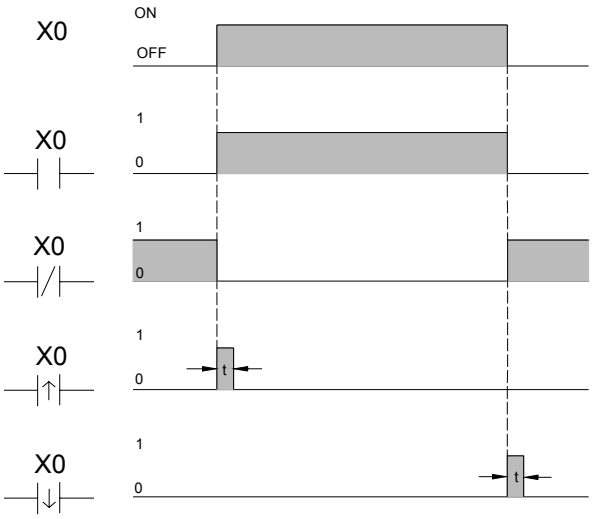
Operand / Ranges / Instruction	X	Y	M	SM	S	T	C	TR	OPEN	SHORT
	X0 | X255	Y0 | Y255	M0 | M1911	M1912 | M2001	S0 | S999	T0 | T255	C0 | C255	TR0 | TR39	—	—
ORG	○	○	○	○	○	○	○		○	○
ORG NOT	○	○	○	○	○	○	○			
ORG TU	○	○	○	○※	○	○	○			
ORG TD	○	○	○	○※	○	○	○			
LD	○	○	○	○	○	○	○	○	○	○
LD NOT	○	○	○	○	○	○	○			
LD TU	○	○	○	○※	○	○	○			
LD TD	○	○	○	○※	○	○	○			
AND	○	○	○	○	○	○	○		○	○
AND NOT	○	○	○	○	○	○	○			
AND TU	○	○	○	○※	○	○	○			
AND TD	○	○	○	○※	○	○	○			
OR	○	○	○	○	○	○	○		○	○
OR NOT	○	○	○	○	○	○	○			
OR TU	○	○	○	○※	○	○	○			
OR TD	○	○	○	○※	○	○	○			
OUT		○	○	○※	○			○		
OUT NOT		○	○	○※	○					
OUT L		○								
ANDLD	—									
ORLD	—									
TU	—									
TD	—									
NOT	—									
SET		○	○	○※	○					
RST		○	○	○※	○					

※For the relays marked with a '◤' symbol in the special relay table（please refer to section 2.3）is write prohibited. In addition, TU and TD contacts are not supported for those relays as well. The operands marked with a '*' symbol in the table shown above should exclude those special relays.

4.2 Element Description

4.2.1 Characteristics of A,B,TU and TD Contacts

- Input X0 from the input terminal block
- A contact Element status
- B contact Element status
- TU contact Element status
- TD contact Element status

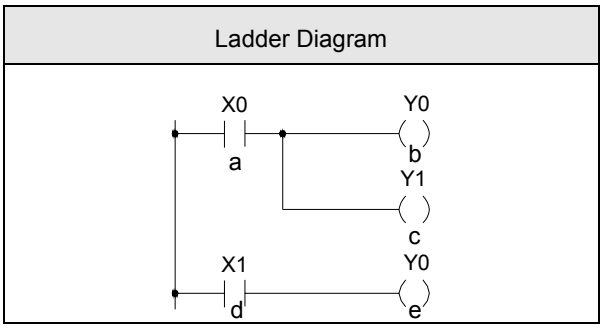


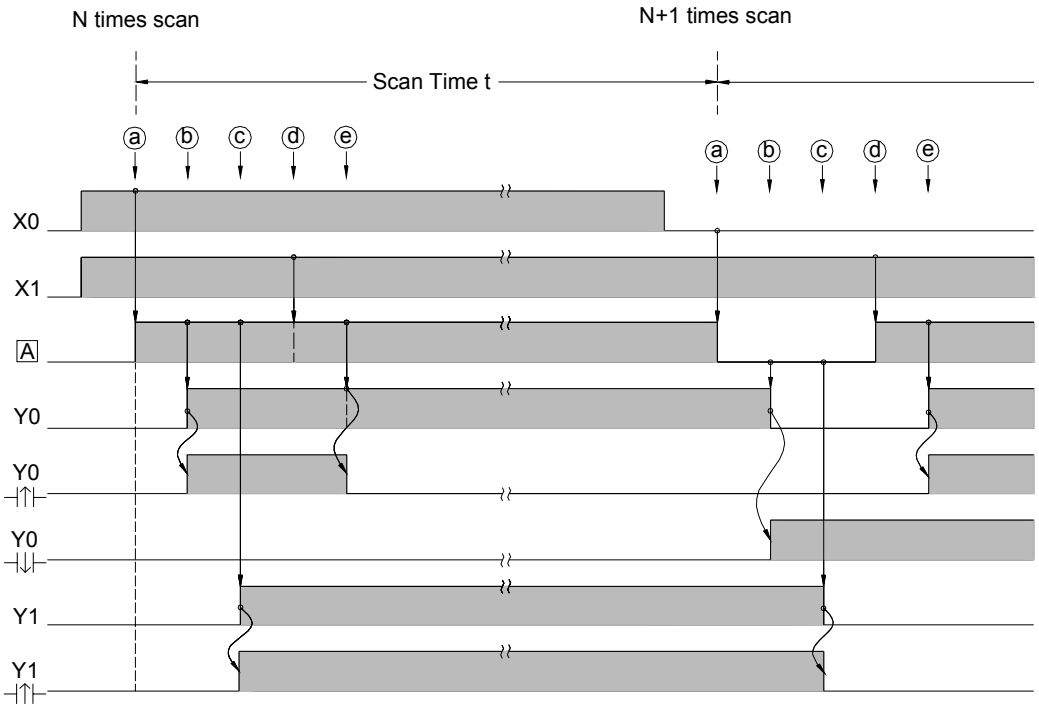
t: scan time

The waveform shown above reveals the function of A, B, TU and TD elements by exercising the external input X0 form OFF to ON then OFF.

- TU (Transition Up): This is the "Transition Up Contact". Only a rising edge (0→1) of the referenced signal will turn on this element for one scan time.
- TD (Transition Down): This is the "Transition Down Contact". Only a falling edge (1→0) of the referenced signal will turn on this element for one scan time.
- TU and TD contact will work normally as described above if the change of the status of the valid referenced operands listed in the "Valid Range of the Operand of Sequential instructions" table are not driven by the function instructions.

Remark: For TU(TD) elements which operand is of relay will turn on after the first time the corresponding relay get driven from 0 to 1(1 to 0). When the next time the corresponding relay get driven from 1 to 1(0 to 0) the TD(TU) element will turn OFF. Care should be taken while there is a multiple coil usage situation existed in the ladder program. This situation can be best illustrated at below. In the waveform we can see Y0 TU element only turn on between ⓑ and ⓔ time which only the Y0 TU elements existed between rung 1 and rung 2 can detect the Y0 rising edge, while other Y0 TU elements out side these two ladder rungs will never aware the occurrence of the rising edge. For the relays do not have the multiple coil usage in ladder program, The ON status of corresponding TU or TD element can be sustained for one scan time, but for relays which contrary to above, the turn on time will shorter than 1 scan time as illustrated at below.



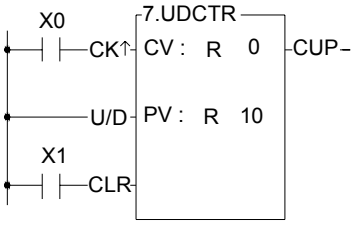


A: The internal accumulator of controller

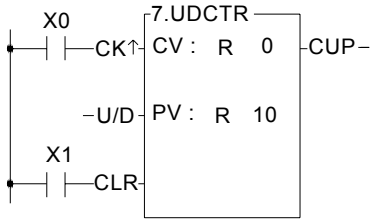
- Besides the TU/TD instructions which can detect the status change of reference operand, WSZ-controller also provides the instructions to detect the change of node status (power flow). For details please refer the descriptions of FUN4 (DIFU) and FUN5 (DIFD) instructions at chapter 6.

4.2.2 OPEN and SHORT Contact

The status of OPEN and SHORT contact are fixed and can't be changed by any ladder instructions. Those two contacts are mainly used in the places of the Ladder Diagram where fixed contact statuses are required, such as the place where the input of an application instruction is used to select the mode. The sample program shown below gives an example of configuring an Up/Down counter (UDCTR) to an Up counter by using the SHORT contact.

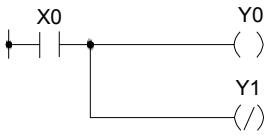


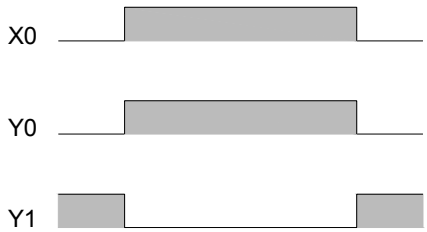
FUN7 is the UDCTR function. While rising edge of CK input occur, FUN7 will count up if the U/D status is 1 or count down if the U/D status is 0. The example shown above, U/D status is fixed at 1 since U/D is directly connected from the origin-line to a SHORT contact, therefore FUN7 becomes an Up counter. On the contrary, if the U/D input of FUN7 is connected with an OPEN contact from the origin-line, the FUN7 becomes a DOWN counter.



4.2.3 Output Coil and Inverse Output Coil

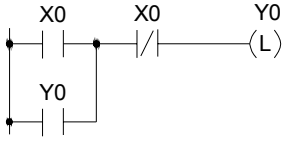
Output Coil writes the node status into an operand specified by the coil instruction. Invert Output Coil writes the complement status of node status into an operand specified by the coil instruction. The characteristics depicts at below.





4.2.4 Retentive Output Coil

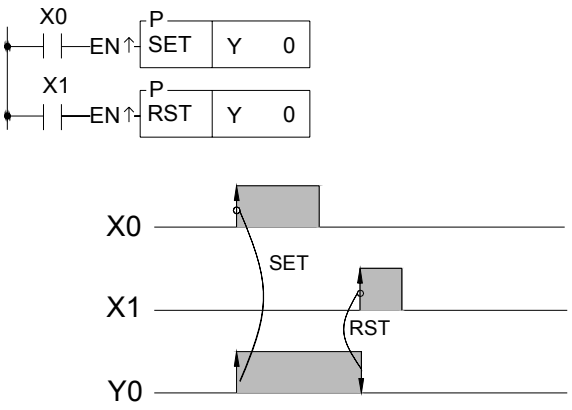
The coil element can be categorized into two types, namely Retentive and Non Retentive. For example, M0~M799 can be specified as the Retentive coils and M800~M1399 can be specified as the Non Retentive coils. One way to categorize the relay type is to divide the relays into groups. Though this method is simple but for the most applications the coils needed to be retentive may be in a random order.



From the above example, if turn the X0 "ON" then "OFF", Y0 will keep at "ON". When change the controller state from RUN to STOP then RUN or turn the power off then on, the Y0 still keep at ON state. But if use the OUT Y0 instruction instead of the OUT L Y0 , Y0 status will be OFF.

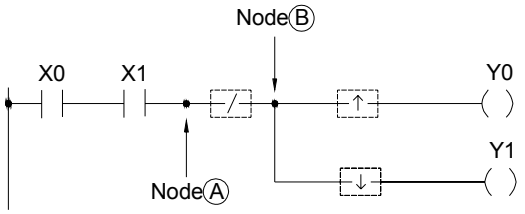
4.2.5 Set Coil and Reset Coil

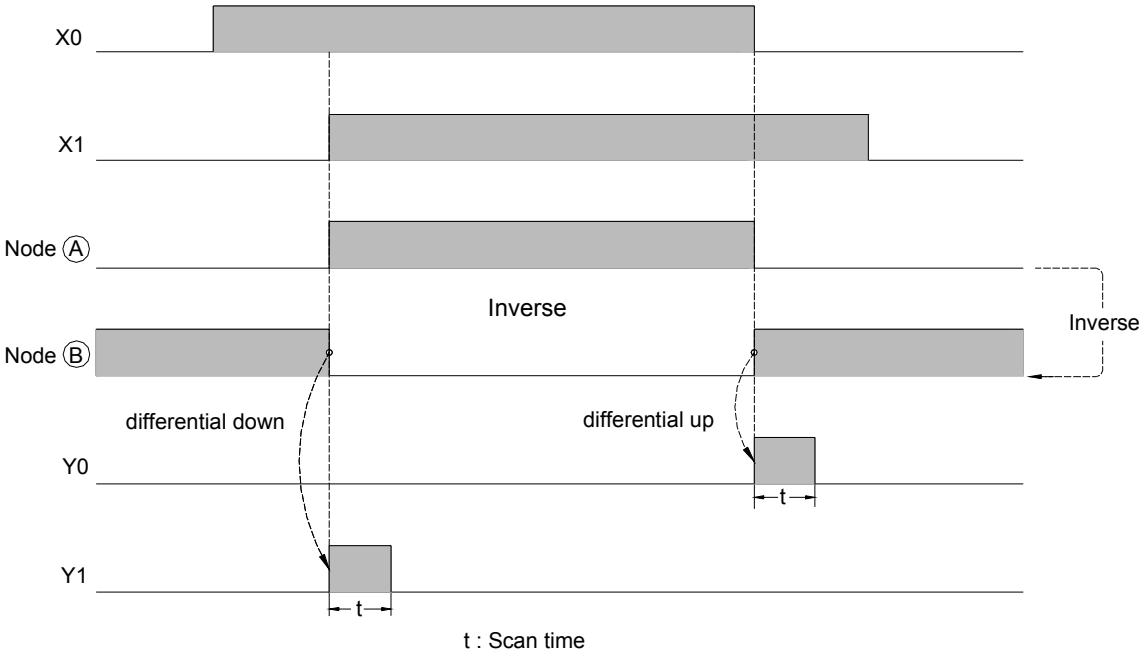
Set Coil writes 1 into an operand specified. Reset Coil writes 0 into an operand specified. The characteristics depicts at below.



4.3 Node Operation Instructions

A node is the connection between elements in a ladder diagram consisting of sequential instruction elements (please refer to Section 1.2). There are four instructions dedicated for node status operation in WSZ-controller. Using the diagram below, the three node operation instructions NOT, TU and TD, are illustrated.

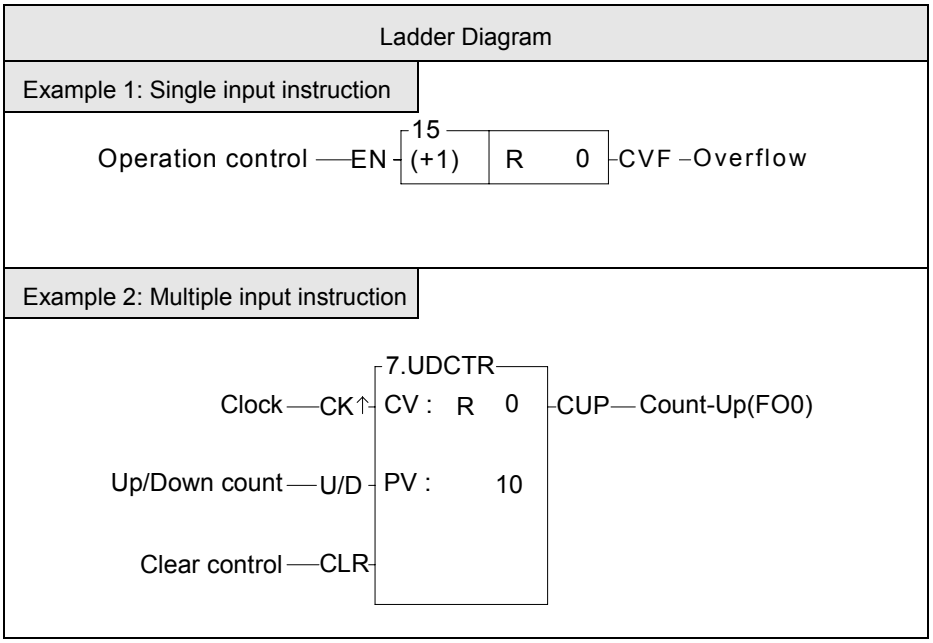



X0
X1
Node Ⓐ
Inverse
Inverse
Node Ⓑ
differential down
differential up
Y0
t
Y1
t
t : Scan time

Chapter 5 Descriptions of Function Instructions

5.1 The Format of Function Instructions

In this chapter we will introduce the function instructions of WSZ-controller in details. All the explanations for each function will be divided into four parts including input control, instruction number/name, operand and function output. An example is shown in below.



5.1.1 Input Control

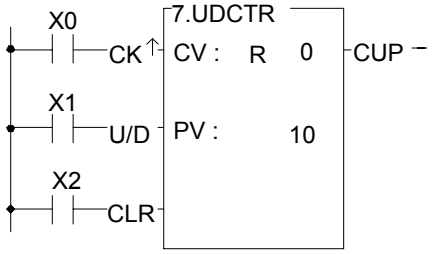
Except for the seven function instructions that do not have input control, the number of the input control of other WSZ-controller function instructions can be ranged from one to four. Execution of the instructions and operations is dependent on the input control signal or the combinations of the several input control signals. The ladder programming software for WSZ controller - WinProladder can help user to complete the complex design and document works. In the ladder program window we can see all the function instructions were displayed by blocks surrounded with abbreviated words for ease of comprehension, include inputs, outs, function name, and parameter names. As shown in example 2 above, the first input mark "CK ↑ " indicates when the "CK ↑ " input changes from 0 to 1 (rising edge) the counter will be increased or decreased by 1 (depending on the "U/D" status). The second input mark "U/D" with a status of 1 represents the word above slash ("U") and the status 0 represents the word under slash ("D"), that is second input "U/D" states =1, the counter will be increased by 1 when "CK ↑ " input from 0 to 1, and when "U/D"=0, the counter will be decreased by 1. The third input mark "CLR" indicates when this input is 1, the counter will be cleared to 0. Chapter 6~7 give the descriptions of input control of each function instruction.

> Remark: There are total of seven instructions whose input control should be directly connected to the origin-line those are MCE, SKPE, LBL, RTS, RTI, FOR, and NEXT. Please refer to chapter 6 and 7 for more detailed explanations.

All input controls of the function instructions should be connected by the corresponding elements, otherwise a syntax error will occur. As shown in example 3 below, the function instruction FUN7 has three inputs and three elements before FUN7. ORG X0, LD X1 and LD X2 corresponds to the first input CK ↑ , second input U/D and third input CLR.

Example 3:

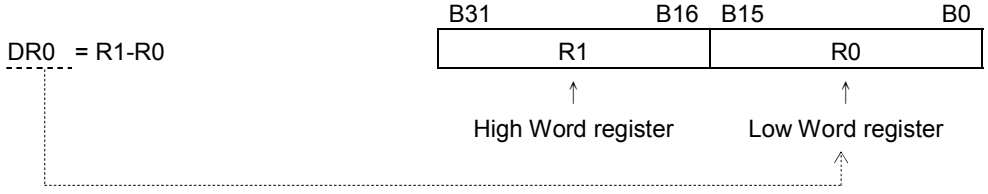
Ladder Diagram

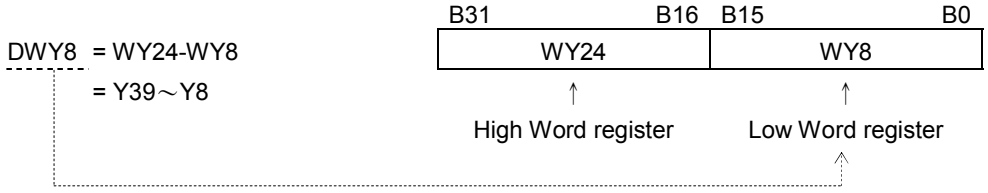


5.1.2 Instruction Number and Derivative Instructions

As mentioned before, except for the nine instructions that can be entered using the dedicated keys on the keyboard, other function instructions must be entered using the "instruction number". Follow the instruction number there are postfixes D, P, DP can be added which can derive three additional function instructions.

D: Indicates a Double Word (32-bit). The 16-bit word is the basic unit of the registers in WSZ-controller. The data length of R, T and C (except C200~C255) registers are 16-bit. If a register with 32-bit data length is required, then it is necessary to combine two consecutive 16-bit registers together such as R1-R0, R3-R2 etc. and those registers are represented by prefix a D letter before register name such as DR0 represents R1-R0 and DR2 represents R3-R2.



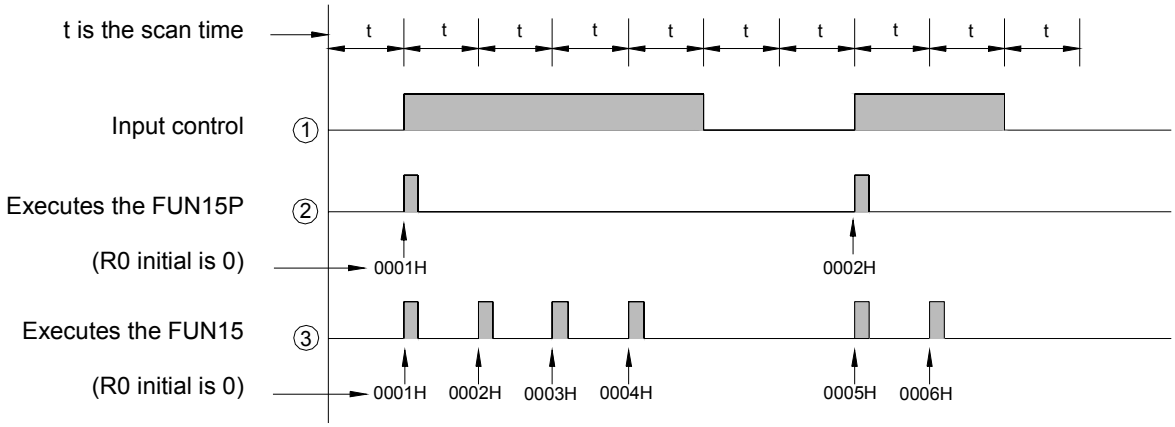


Remark: In order to differentiate between 16-bit and 32-bit instructions while using the ladder diagram, we add the postfix letter D after the "Instruction number" to represent 32-bit instructions and the size of their operand are 32-bit as shown in example 2 on P.6-6. The instruction FUN 11D has a postfix letter D, therefore the source and destination operands need to prefix a letter D as well, such as the augend Sa : R0 is actually Sa=DR0=R1-R0 and Sb=DR2=R3-R2. Please also pay special attention to the length of the other operands except source and destination are only one word whether 16-bit or 32-bit instructions are used.

P: indicates the pulse mode instruction. The instruction will be executed when the status of input control changes from 0 to1 (rising edge). As shown in example 1, if a postfix letter P is added to the instruction (FUN 15P), the instruction FUN 15P will only be executed when the status of input control signal changes from 0 to 1. The execution of the instruction is in level mode if it does not have a P postfix, this means the instruction will be executed for every scan until the status of input control changes from 1 to 0. The pulse input is indicated by a symbol " ↑ ", such as CK ↑ , EN ↑ , TG ↑ etc.. In this operation manual, an example of the operation statement of a function instruction is shown below.

● When the operation control〝EN〞=1 or〝EN↑〞（P instruction）from 0→1, ……

The first one indicates the execution requirement for non-P instruction (level mode) and the second one indicates the execution requirement for P instruction (pulse mode). The following waveform shows the result (R0) of FUN15 and FUN15P under the same input condition.



DP: Indicates the instruction is a 32-bit instruction operating with pulse mode.

Remark: P instruction is much more time saving than level instruction in program scanning, So user should use P instruction as much as possible.

5.1.3 Operand

The operand is used for data reference and storage. The data of source (S) operand are only for reference and will not be changed with the execution of the instruction. The destination (D) operand is used to store the result of operation and its data may be changed after the execution of the instruction. The following table illustrates the names and functions of WSZ controller function instruction's operands and types of contacts, coils, or registers that can be used as an operand.

■ The names and functions of the major operands:

Abbreviation	Name	Descriptions
S	Source	The data of source (S) operand are only for reading and reference and will not be changed with the execution of the instruction. If there are more than one source operands, each operand will be identified by the footnote such as Sa and Sb.

Abbreviation	Name	Descriptions
D	Destination	The destination (D) operand is used to store the result of operation. The original data will be changed after operation. Only the coils and registers which are not writing prohibited can be the destination operand.
L	Length	Indicates the data size or the length of the table, usually are constants.
N	Number	A constant most often used as numbers and times. If there are more than one constant, each constant will be identified by the footnotes such as Na, Nb, Ns etc..
Pr	Pointer	Used to point to a specific a block of data or a specific data or register in a table. Generally the Pr value can be varied, therefore cannot be constant or input register.)
CV	Current value	Used in T and C instruction to store the current value of T or C
PV	Set value	Used in T and C instructions for reference and comparison
T	Table	A combination of a set of consecutive registers forms a table. The basic operation units are word and double word. If there is more than one table, each table will be identified by footnotes such as Ta, Tb, Ts and Td etc...
M	Matrix	A combination of a set of consecutive registers forms a matrix. The basic operation unit is bit. If there is more than one matrix, each matrix will be identified by footnotes such as Ma, Mb, Ms and Md etc...

Besides the major operands mentioned above, there are other operands which are used for certain special purposes such as the operand Fr for frequency, ST for stack, QU for Queue etc.. Please refer to the instruction descriptions for more details.

- The types of the operand and their range: The types of operand for the function instructions are discrete, register and constant.

  a) Discrete operand:

  There are total five function instructions that reference the discrete operand, namely SET, RST, DIFU, DIFD and TOGG. Those five instructions can only be used for operations of Y△△△(external output), M△△△△ (internal and special) and S△△△(step) relays. The table shown below indicates the operands and ranges of the five function instructions.

Range / Ope-rand	Y	M	SM	S
	Y0 | Y255	M0 | M1911	M1912 | M2001	S0 | S999
D	○	○	○*	○

Symbol "O" indicates the D (Destination operand) can use this type of coils as operands. The "*" sign above the "O" shown in SM column indicates that should exclude the write prohibited relays as operands. Please refer to page 2-3 for introduction of the special relays.

  b) Register operand :

  The major operand for function instructions is register operand. There are two types of register operands: the native registers which already is of Words or Double Words data such as R, D, T, C. The other is derivative registers (WX, WY, WM, WS) which are formed by discrete bits. The types of registers that can be used as instruction operands and their ranges are all listed in the following table:

Range / Ope-rand	WX	WY	WM	WS	TMR	CTR	HR	IR	OR	SR	ROR	DR	K	XR
	WX0 | WX240	WY0 | WY240	WM0 | WM1896	WS0 | WS984	T0 | T255	C0 | C255	R0 | R3839	R3840 | R3903	R3904 | R3967	R3968 | R4167	R5000 | R8071	D0 | D4095	16/32-bit +/- number	V、Z P0~P9
S	○	○	○	○	○	○	○	○	○	○	○*	○	○	○
D		○	○	○	○	○	○		○	○*	○*	○		○
⋮														

The "○" symbol in the table indicates can apply this kind of data as operand. The "○*" symbol indicates can apply this kind of data except the write prohibited registers as operand. To learn more about write prohibited registers please refer to page 2-8 for introduction of the special register.

When R5000～R8071 are not set to be read only registers, can used as normal registers (read, and write)

Remark 1: The registers with a prefix W, such as WX, WY, WM and WS are formed by 16 bits. For example, WX0 means the register is formed by X0(bit 0)~X15(bit 15). WY144 means the register is formed by Y144(bit 0)~Y159(bit 15). Please note that the discrete number must be the multiple of 8 such as 0, 8, 16, 24....

Remark 2: The last register (Word) in a table can not be represented as a 32-bit operand in the function because 2 Words are required for a 32-bit operand.

Remark 3: TMR (T0 ~ T255) and CTR (C0 ~ C255) are the registers of timers and counters respectively. Although they can be used as general registers, they also complicate the systems and make debugging more difficult. Therefore you should avoid writing anything into the TMR or CTR registers.

Remark 4: T0 ~ T255 and C0 ~ C199 are 16-bit register. But C200~C255 are 32-bit register, therefore can't be used as 16-bit operands.

Remark 5: Apart from being directly appointed by register's number (address) as the foregoing discussions, the register's operand in the range of R0 ~ R8071 can be combined with pointer register V、Z or P0~P9 to make indirect addressing. Please refer to the example in the next section (Section 5.2) for the description of using pointer register (XR) to make indirect addressing.

c) Constant operands :

The range of 16-bit constant is between -32768~32767. The range of 32-bit constant is between -2147483648~2147483647. The constant for several function instructions can only be a positive constant. The ranges of 16-bit and 32-bit constants are listed in the table shown below.

Classification	Range
16-bit signed number	-32768～32767
16-bit un-signed number	0～32767
32-bit signed number	-2147483648～2147483647
32-bit un-signed number	0～2147483647
16/32-bit signed number	-32768～32767 or -2147483648～2147483647
16/32-bit un-signed number	0～32767 or 0～2147483647

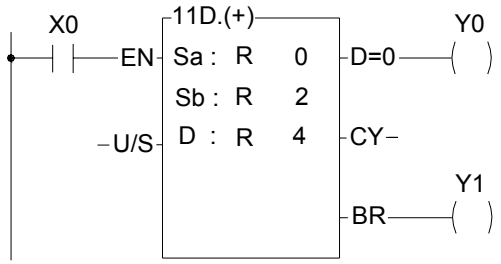
It is possible that the length and size of a specific operand, such as L, bit size, N etc.., are different, and the differences are all directly marked at the operand column. Please refer to the explanations of function instructions.

5.1.4 Functions Output (FO)

The “Function Output” (FO) is used to indicate the operation result of the function instruction. Like control input, each function outputs shown in the screen of programming software are all attached with a word which comes from the abbreviation of the output functionality. Such as CY derived from Carry. The maximum number of function outputs is 4 and those are denoted as FO0, FO1, FO2, FO3 respectively. The FO status must be taken out by FO instruction. The unused FO may be left without connecting to any elements, such as FO1 (CY) shown in Example 4 below.

Example 4 :

Ladder Diagram



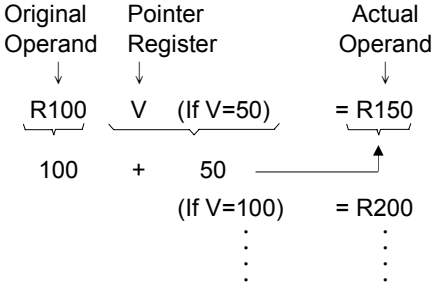
When M1919=0, the FO status will only be updated if the instruction is executed. It will keep the same status until a new FO status is generated after the instruction is executed again (memory keeping).

When M1919=1, the FO status will be reset to 0 (no memory keeping) if the instruction is not executed.

5.2 Use Index Register (XR) for Indirect Addressing

In the WSZ-controller function instructions, there are some operands that can be combined with pointer register (V、Z、P0~P9) to make indirect addressing (will be shown in the operand table if it applicable). However, only the registers in the range R0～R8071 can be combined with an pointer register to perform indirect addressing (other operands such as discrete, constant and D0～D3071 cannot be used for indirect addressing).

There are twelve pointer registers XR (V、Z、P0~P9). The V register in fact is the R4164 of special registers (R3840～R4167) , the Z register is the R4165 and the P0~P9 register is the (D4080~D4089). The actual addressed register by index addressing is just offset the original operand with the content of the index register.



As shown in the above diagram, you only need to change the V value to change the operand address. After combining the index addressing with the WSZ-controller function instructions, a powerful and highly efficient control application can be achieved by using very simple instructions. Using the program shown in the diagram below as an example, you only need to use a block move instruction (BT_M) to achieve a dynamic block data display, such as a parking management system.

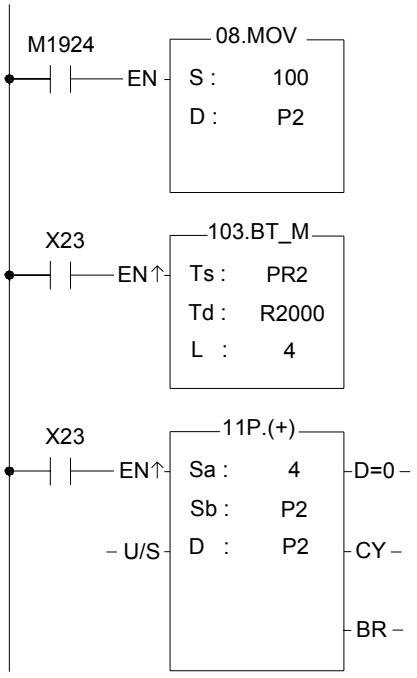
Index Register(P0~P9) Introduction

In indirect addressing application, Rxxxx register can combine V、Z & P0～P9 for index addressing; Dxxxx register can't combine V、Z for index addressing, but P0～P9 are allowed.

When V、Z index register being combined with the Rxxxx register,
for example, R0 with V、Z, the instruction format is R0V(where V=100, it means R100) or R0Z(where Z=500, it means R500); when P0～P9 index register being combined with the Rxxxx register, the instruction format is RPn (n=0～9) or RPmPn (m,n=0～9), for example RP5 (where P5=100, it means R100) or RP0P1(where P0= 100, P1=50, it means150).

When P0～P9 index register being combined with the Dxxxx register, the instruction format is DPn (n=0～9) or DPmPn (m,n=0～9), for example DP3 (where P3=10, it means D10) or DP4P5 (where P4=100, P5=1, it means D101).

It can combine both P0～P9 index register, for example P2=20, P3=30, when Rxxxx or Dxxxx register combines both index register, RP2P3 will point to R50, DP2P3 will point to D50, it means the summation of both index register for indirect addressing.

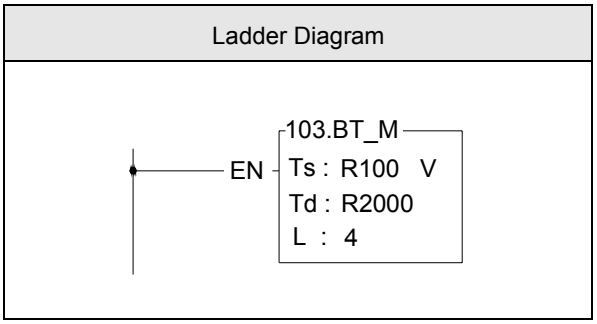


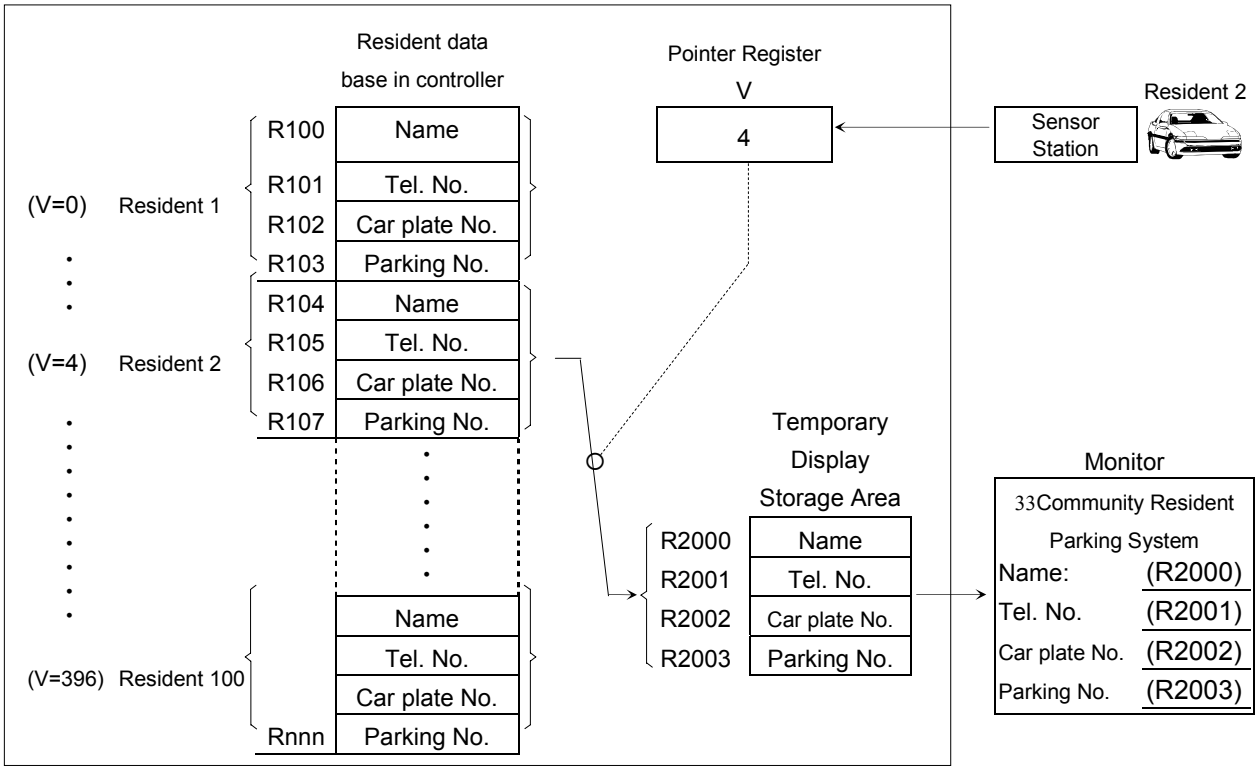
1. Index register P2=100 while power up or first run.

2. When X23 changes from 0→1, FUN103 will perform the table movement, the source starts from R100 (P2=100), the destination starts from R2000, the amount is 4. Coping the content of R100~R103 for R2000~R2003 at first execution, coping the content of R104~R107 for R2000~R2003 at second execution…

3. Increasing the P2 index register by 4 to point to next 4.

Indirect addressing program example





Description

Suppose that there are 100 resident parking spaces available in a parking management system for community residents. Each resident has a set of basic information including name, telephone number, number plate and parking number, that occupy four consecutive controller registers as shown in the above diagram. A total of 400 registers (R100～R499) are occupied. Each resident is given a card with a unique card number (the number is 0 for resident 1, 4 for resident 2 etc..) for the sensing pass of the main entrance and parking lot. The card number will be sensed by the controller and stored into the pointer register “V”. The attendant’s monitor (LCD or CRT) will only display the data grasped by R2000～R2003 in the controller. For example, the card of residence 2 with the card number 4 is sensed, then the register V=4 and the controller will immediately move the data in R104～R107 to the temporary display storage area (R2000～R2003). Hence, the attendant’s monitor can display the data of residence 2 as soon as its card is sensed.

⚠ Warning

1. Although using pointer register for indirect addressing application is powerful and flexible, but changing the V and Z values freely and carelessly may cause great damages with erroneous writing to the normal data areas. The user should take special caution during operation.
2. In the data register range that can be used for indirect addressing application (R0～R8071), the 328 registers R3840～R4167 (i.e. IR, OR and SR) are important registers reserved for system or I/O usage. Writing at-will to these registers may cause system or I/O errors and may result in a major disaster. Due to the fact that users may not easily detect or control the flexible register address changes made by the V and Z values, WSZ-controller will automatically check if the destination address is in the R3840～R4067 range. If it is, the write operation will not be executed and the M1969 flag “Illegal write of Indirect addressing” will be set as 1. In case it is necessary to write to the registers R3840～R4067, please use the direct addressing.

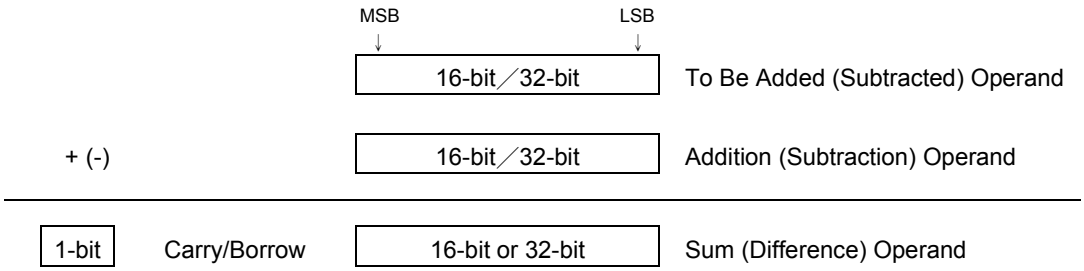
5.3 Overflow and Underflow of Increment (+1) or Decrement (-1) (Beginners should skip this section)

The maximum positive values that can be represented by 16-bit and 32-bit operands are 32767 and 2147483647, respectively. While the minimum negative values that can be represented by 16-bit and 32-bit operands are –32768 and –2147483648, respectively. When increase or decrease an operand (e.g. when Up/Down Count of a counter or the register value is +1 or -1), and the result exceeds the value of the positive limit of the operand, then “Overflow” (OVF) occurs. This will cause the value to cycle to its negative limit (e.g. add 1 to the 16-bit positive limit 32767 will change it to –32768). If the result is smaller than the negative limit of the operand, then “Underflow” (UDF) occurs. This will cause the value to cycle to its positive limit (e.g. deducting 1 from the negative limit –32768 will change it to 32767) as shown in the table below. The flag output of overflow or underflow exists in the FO of WSZ-controller and can be used in cascaded instructions to obtain over 16-bit or 32-bit operation results.

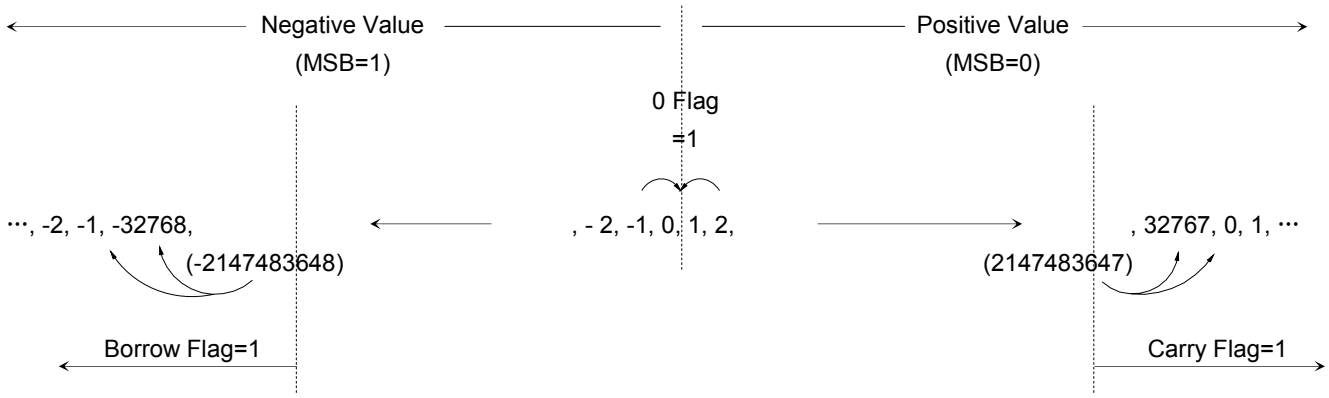
Increase (Decrease) Result / Overflow/ Underflow	16-bit Operand	32-bit Operand
Increase	OVF=1 → -32767 → -32768 → 32767 → 32766 → 32765	OVF=1 → -2147483646 → -2147483647 → -2147483648 → 2147483647 → 2147483646
Decrease	UDF=1 → -32767 → -32768 → 32767 → 32766 → 32765	UDF=1 → -2147483647 → -2147483648 → 2147483647 → 2147483646 → 2147483645

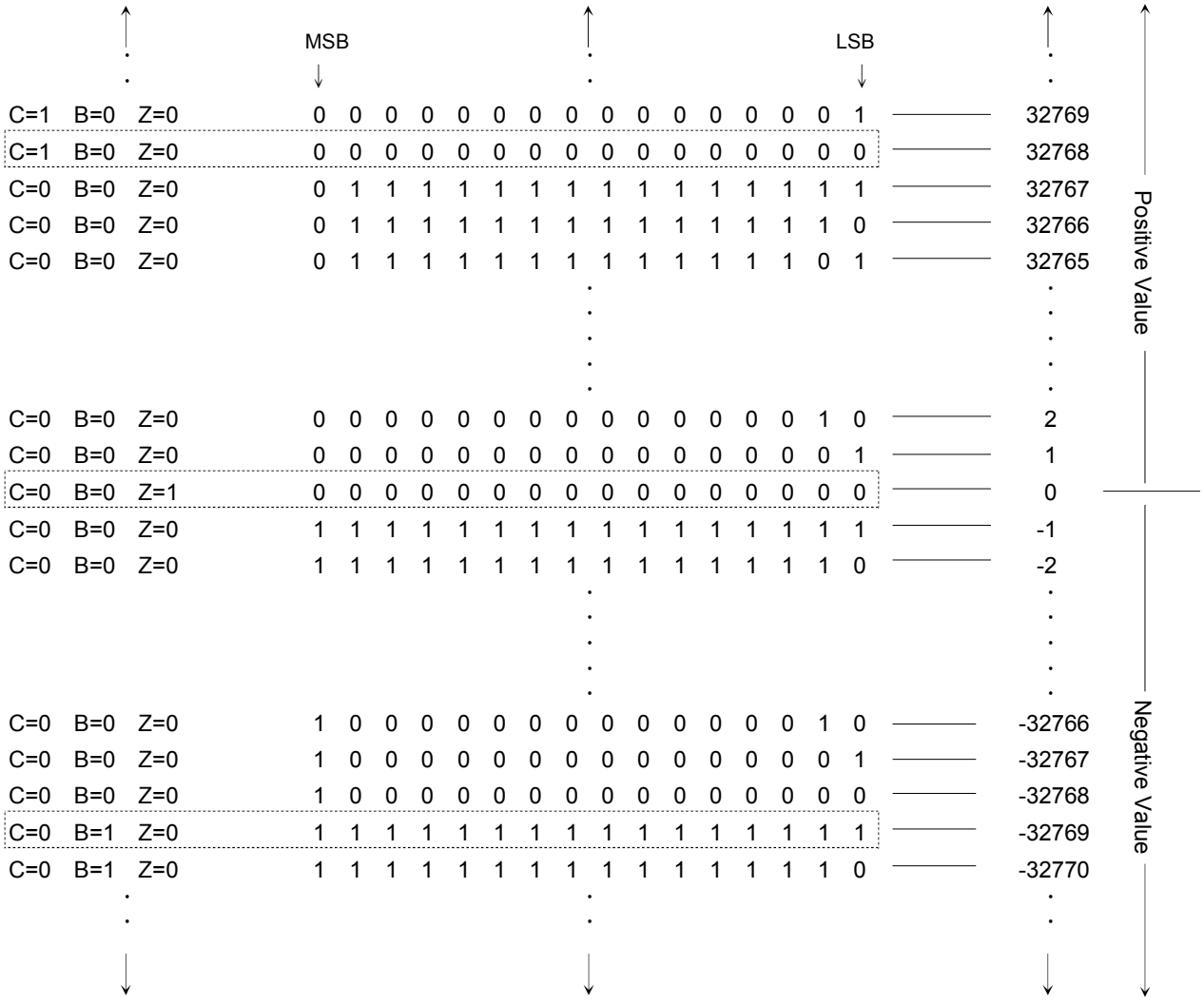
5.4 Carry and Borrow in Addition/Subtraction

Overflow/Underflow takes place when the operation of increment/decrement causes the value of the operand to exceed the positive/negative limit that can be represented in the controller; consequently a flag of overflow/underflow is introduced. Carry/Borrow flag is different from overflow/underflow. At first, there must be two operands making addition (subtraction) where a sum (difference) and a flag of carry/borrow will be obtained. Since the number of bits of the numbers to be added (subtracted), to add (subtract) and of sum (difference) are the same (either 16-bit or 32-bit), the result of addition (subtraction) may cause the value of sum (difference) to exceed 16-bit or 32-bit. Therefore, it is necessary to use carry/borrow flag to be in coordination with the sum (difference) operand to represent the actual value. The carry flag is set when the addition (subtraction) result exceeds the positive limit (32767 or 2147483647) of the sum (difference) operand. The borrow flag is set when addition (subtraction) result exceeds the negative limit (-32768 or -2147483648) of the sum (difference) operand. Hence, the actual result after addition (subtraction) is equal to the carry/borrow plus the value of the sum (difference) operand. The FO of WSZ controller addition/subtraction instructions has both carry and borrow flag outputs for obtaining the actual result.



While all WSZ-controller numerical operations use 2'S Complement, the representation of the negative value of the sum (difference) obtained from addition (subtraction) is different from the usual negative number representation. When the operation result is a negative value, 0 can never appear in the MSB of the sum (difference) operand. The carry flag represents the positive value 32768 (2147483648) and the borrow flag represents the negative value -32768 (-2147483648).





C = Carry B = Borrow Z = Zero

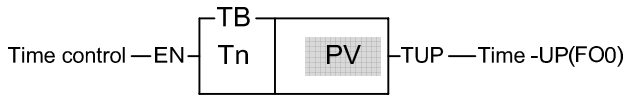
Chapter 6 Basic Function Instructions

T	TIMER	T

Symbol

Operand

Ladder symbol



Tn: Timer Number.

PV: Preset value of the timer.

TB: Time Base (0.01s, 0.1s, 1s)

Range / Operand	WX	WY	WM	WS	TMR	CTR	HR	IR	OR	SR	ROR	DR	K
	WX0 | WX240	WY0 | WY240	WM0 | WM1896	WS0 | WS984	T0 | T255	C0 | C255	R0 | R3839	R3840 | R3903	R3904 | R3967	R3968 | R4167	R5000 | R8071	D0 | D4095	0 | 32767
Tn					○								
PV	○	○	○	○	○	○	○	○	○	○	○	○	○

- The total number of timers is 256 (T0～T255) with three different time bases, 0.01s, 0.1s and 1s.The default number and allocation of timers is shown as below (Can be adjusted according to user's actual requirements by the "Configuration" function):

  T0～T49：0.01s timer（default as 0.00～327.67s）

  T50～T199：0.1s timer（default as 0.0～3276.7s）

  T200～T255：1s timer（default as 0～32767s）

- WSZ programming tool will lookup the timer's time base automatically according to the "Memory Configuration" after the timer number is keyed in. Timer's time = Time base x Preset value. In the example 1 below, the time base T0 = 0.01s and the PV value = 1000, therefore the T0 timer's time = 0.01s x 1000 = 10.00s.

- If PV is a register, then Timer's time = Time base x register content. Therefore, you only need to change the register content to change the timer's time. Please refer to Example 2.

※ The maximum error of a timer is a time base plus a scan time. In order to reduce the timing error in the application, please use the timer with a smaller time base.

Description

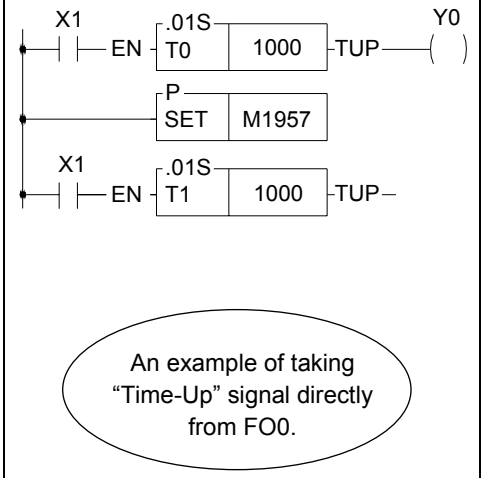
- When the time control "EN" is 1, the timer will start timing (the current value will accumulate from 0) until "Time Up" (i.e. CV≧PV), then the Tn contact and TUP (FO0) will change to 1. As long as the timer control "EN" input is kept as 1, even the CV of Tn has reached or exceeded the PV, the CV of the timer will continue accumulating (with M1957 = 0) until it reaches the maximum limit (32767). The Tn contact status and flag will remain as 1 when CV≧PV, unless the "EN" input is 0. When "EN" input is 0, the CV of Tn will be reset to 0 immediately and the Tn contact and "Time Up" flag TUP will also change to 0 (please refer to the diagram ① below).

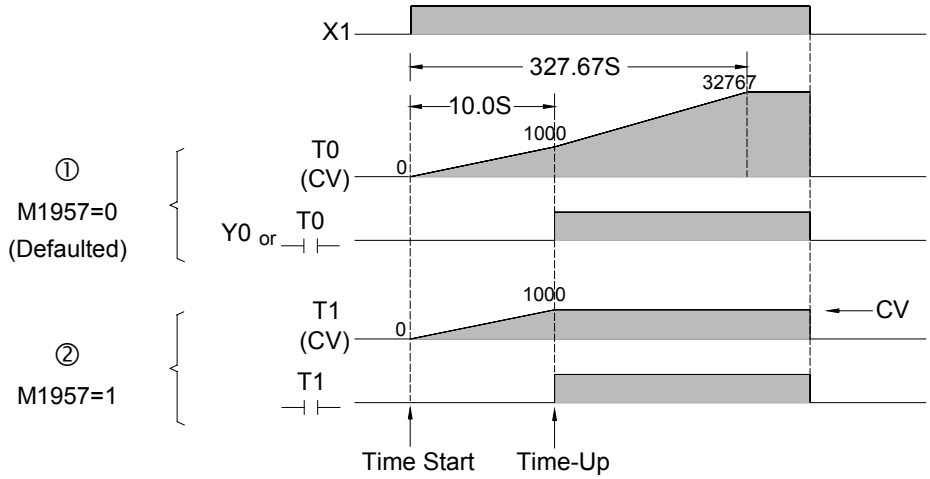
- If the WSZ OS version is higher than V3.0 (inclusive), the M1957 can be set to 1 so the CV will not accumulate further after "Time Up" and stops at the PV value. The default value of the M1957 is 0, therefore the status of M1957 can be set before executing any timer instruction in the program to individually set the timer CV to continue accumulating or stop at the PV after "Time Up" (please refer to the diagram ② below).

T	TIMER	T

Example 1 Constant preset value

Ladder diagram



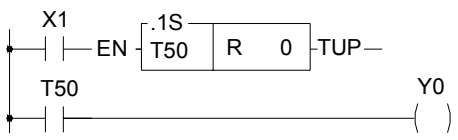


Example 2 Variable PV

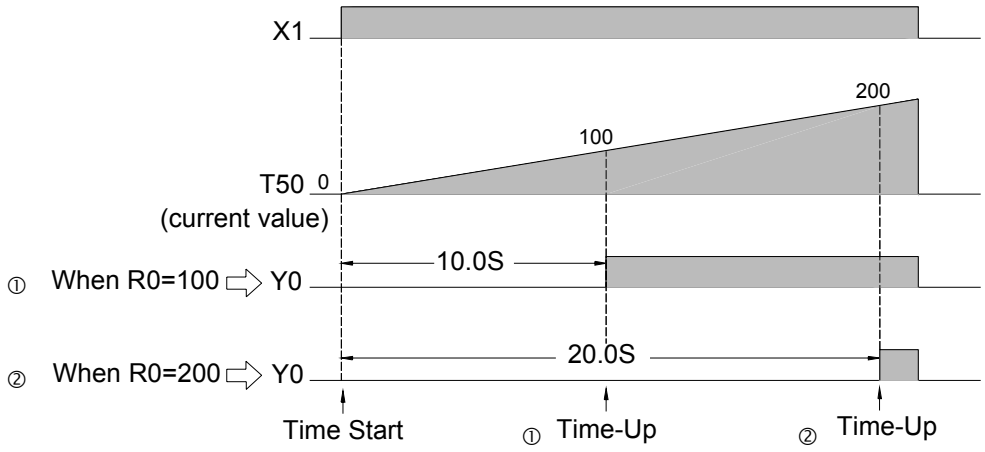
The preset value (PV) shown in example 1 is a constant which is equal to 1000. This value is fixed and can not be changed once programmed. In many circumstances, the preset time of the timers needs to be varied while controller running. In order to change the preset time of a timer, can first use a register as the PV operand (R or WX, WY...) and then the preset time can be varied by changing the register content. As shown in this example, if set R0 to100, then T becomes a 10s Timer, and hence if set R0 to 200, then T becomes a 20s Timer.

T	TIMER	T

Ladder diagram



An example of applying the “time-up” status by using the T50 contact.

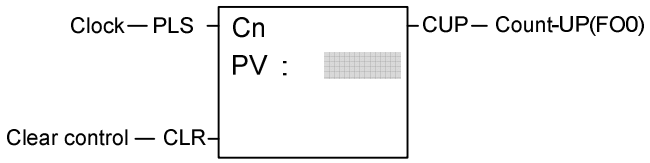


Remark: If the preset value of the timer is equal to 0, then the timer's contact status and FO0 (TUP) become 1 ("EN" input must be at 1) immediately after the controller finishes its first scan because "Time-Up" has occurred. (TUP) stays at 1 until "EN" input changes to 0.

C	COUNTER （16-Bit: C0～C199，32-Bit: C200～C255）	C

Symbol

Ladder symbol



Operand

Cn: The Counter number

PV: Preset value

Range / Operand	WX	WY	WM	WS	TMR	CTR	HR	IR	OR	SR	ROR	DR	K
	WX0 | WX240	WY0 | WY240	WM0 | WM1896	WS0 | WS984	T0 | T255	C0 | C255	R0 | R3839	R3840 | R3903	R3904 | R3967	R3968 | R4167	R5000 | R8071	D0 | D4095	0 | 2147483647
Cn						○							
PV	○	○	○	○	○	○	○	○	○	○	○	○	○

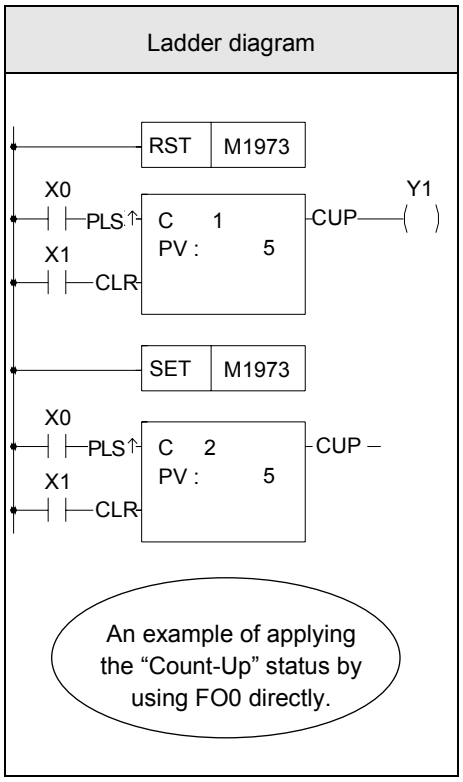
- There are total 200 16-Bit counters (C0~C199). The range of preset value is between 0~32767. C0~C139 are Retentive Counters and the CV value will be retained when the controller turns on or RUN again after a power failure or a controller STOP. For Non Retentive Counters, if a power failure or controller STOP occurs, the CV value will be reset to 0 when the controller turns on or RUN again.
- There are total 56 32-Bit counters (C200~C255). The range of the preset value is between 0~2147483647. C200~C239 are Retentive Counters and C240~C255 are Non Retentive Counters.
- The default number and assignment of the counters are shown below, if necessary can use the "CONFIGURATION" function to change the settings.
- To insure the proper counting, the sustain time of input status of PLS should greater than 1 scan time.
- The max. counting frequency with this instruction can only up to 20Hz, for higher frequency please use the high-speed soft/hardware counter.

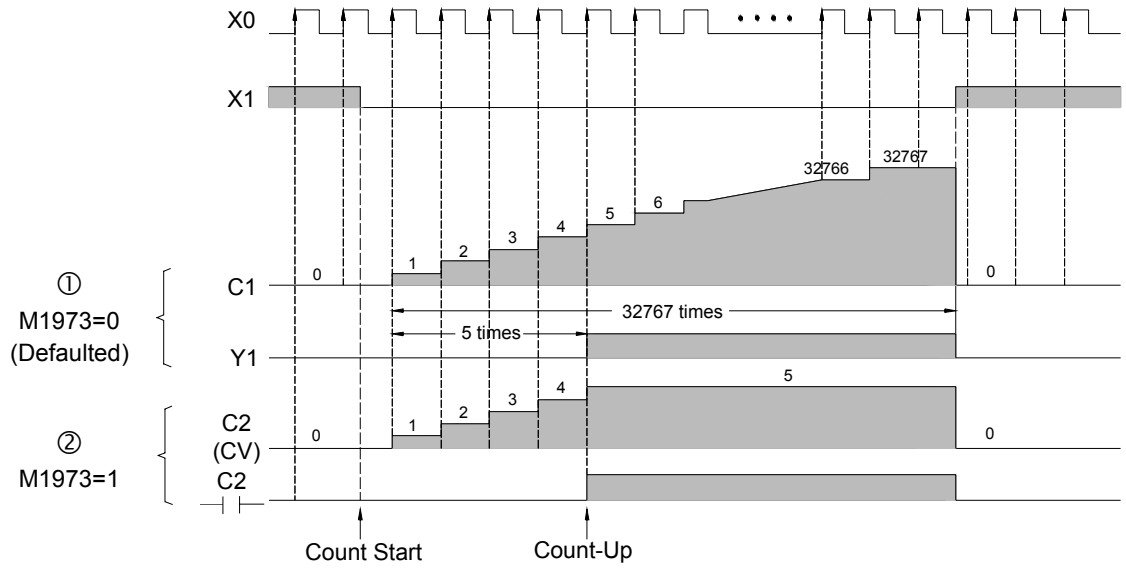
Description

- When "CLR" is at 1, all of the contact Cn, FO0 (CUP), and CV value of the counter CV are cleared to 0 and the counter stops counting.
- When "CLR" is at 0, the counter is allowed to count up. The Counter counts up every time the clock "PLS" changes from 0 to 1 (adds 1 to the CV) until the cumulative current value is equal to or greater than the preset value (CV>=PV), the counter "Count-Up" and the contact status of the counter Cn and FO0 (CUP) changes to 1. If the input status of clock continues to change, even the cumulative current value is equal and greater than the preset value, the CV value will still accumulate until it reaches the up limit at 32767 or 2147483647. The contact Cn and FO0 (CUP) stay at 1 as long as CV>=PV unless the "CLR" input is set to 1.（please refer the diagram ① below）
- If the WSZ OS version is higher than V3.0 (inclusive), the M1973 can set to 1 so the CV will not accumulate further after “Count Up” and stops at the PV. M1973 default value is 0, therefore the status of M1973 can be set before executing any counter instruction in the program to individually set the counter CV to continue accumulating or stops at the PV after “Count Up” (please refer to the diagram ② below).

C	COUNTER （16-Bit: C0～C199, 32-bit: C200～C255）	C

Example 1　16-Bit Fixed Counter



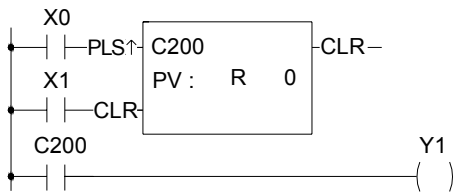


Example 2　32-Bit counter with variable preset value

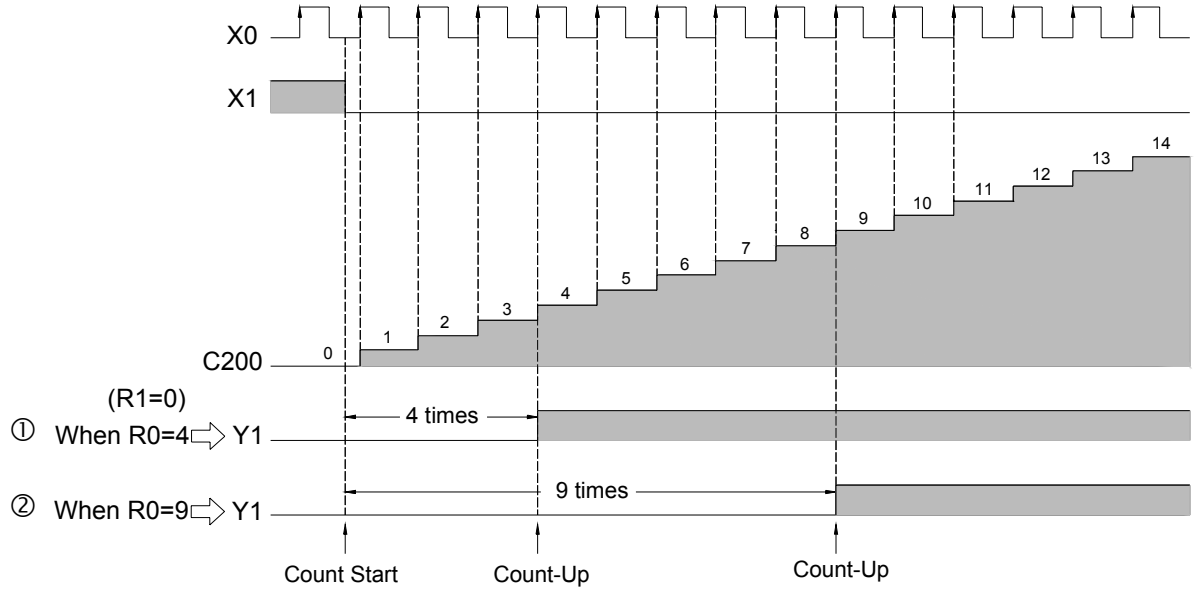
Like a timer, if the PV of a counter is changed to a register (such as R, D, and so on), the counter will use the register contents as the counting PV. Therefore, only need to change the register contents to change the PV of the counter while controller is running. Below is an example of a 32-bit counter that uses the data register R0 as the PV (in fact it is the 32-bit PV formed by R1 and R0).

C	COUNTER （16-Bit: C0～C199, 32-Bit: C200～C255）	C

Ladder diagram



An example of applying the “time-up” status by using the C200 contact.

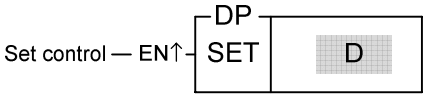


Remark: If the preset value of the counter is 0 and "CLR" input also at 0, then the Cn contact status and FO0 (CUP) becomes 1 immediately after the controller finishes its first scan because the "Count-Up" has occurred. It will stay at 1 regardless how the CV value varies until "CLR" input changes to 1.

SET D P	SET (Set coil or all the bits of register to 1)	SET D P

Symbol

Ladder symbol



Operand

D: destination to be set
(the number of a coil or a register)

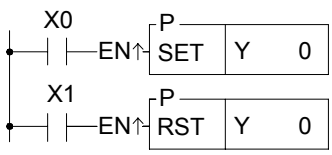
Range / Operand	Y	M	SM	S	WY	WM	WS	TMR	CTR	HR	OR	SR	ROR	DR
	Y0 | Y255	M0 | M1911	M1912 | M2001	S0 | S999	WY0 | WY240	WM0 | WM1896	WS0 | WS984	T0 | T255	C0 | C255	R0 | R3839	R3904 | R3967	R3968 | R4167	R5000 | R8071	D0 | D4095
D	○	○	○*	○	○	○	○	○	○	○	○	○*	○*	○

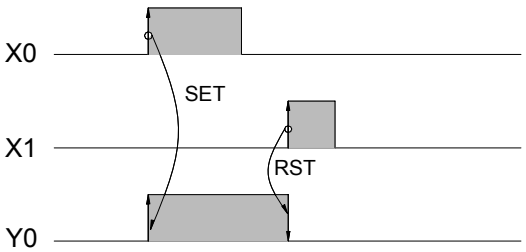
Description

● When the set control "EN" =1 or "EN↑" (P instruction) changes from 0 to 1, sets the bit of a coil or all bits of a register to 1.

Example 1 Single Coil Set

Ladder Diagram

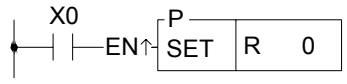


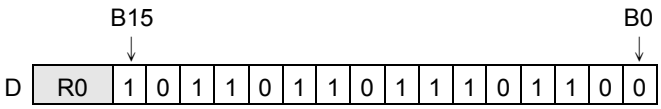


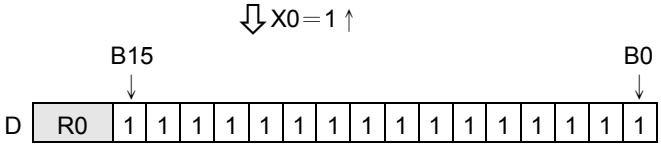
SET D P	SET (Set coil or all the bits of register to 1)	SET D P

Example 2 Set 16-Bit Register

Ladder Diagram

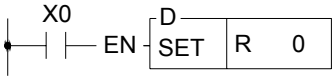


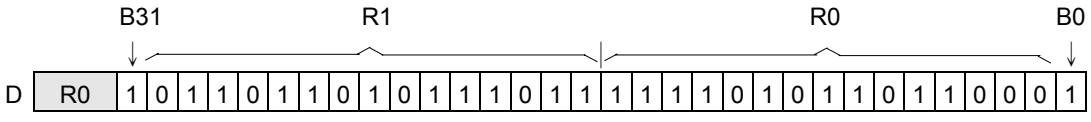


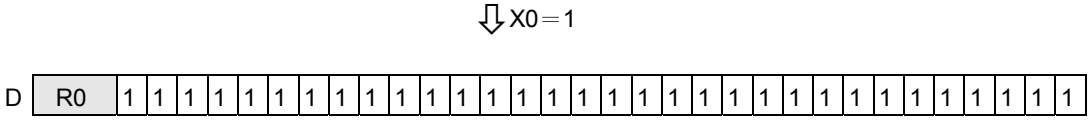


Example 3 32-Bit Register Set

Ladder Diagram



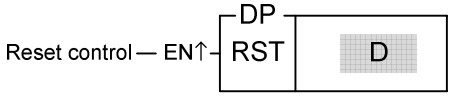




RST D P	RESET (Reset the coil or the register to 0)	RST D P

Symbol

Ladder symbol



Operand

D: Destination to be reset
(the number of a coil or a register)

Range / Operand	Y	M	SM	S	WY	WM	WS	TMR	CTR	HR	OR	SR	ROR	DR
	Y0 | Y255	M0 | M1911	M1912 | M2001	S0 | S999	WY0 | WY240	WM0 | WM1896	WS0 | WS984	T0 | T255	C0 | C255	R0 | R3839	R3904 | R3967	R3968 | R4167	R5000 | R8071	D0 | D4095
D	○	○	○*	○	○	○	○	○	○	○	○	○*	○*	○

Description

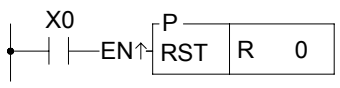
● When the reset control "EN" =1 or "EN↑" (P instruction) changes from 0 to 1, resets the coil or register to 0.

Example 1 Single Coil Reset

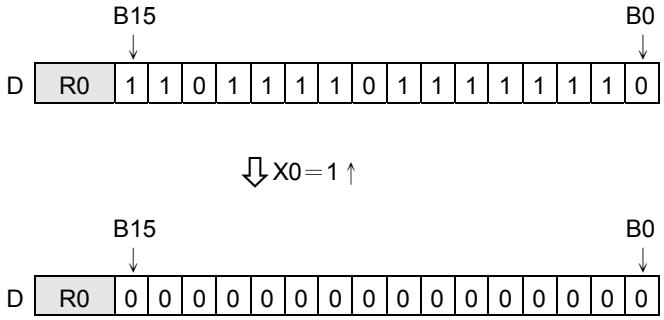
Please refer to example 1 for the SET instruction shown in page 6-8.

Example 2 16-Bit Register Reset

Ladder Diagram

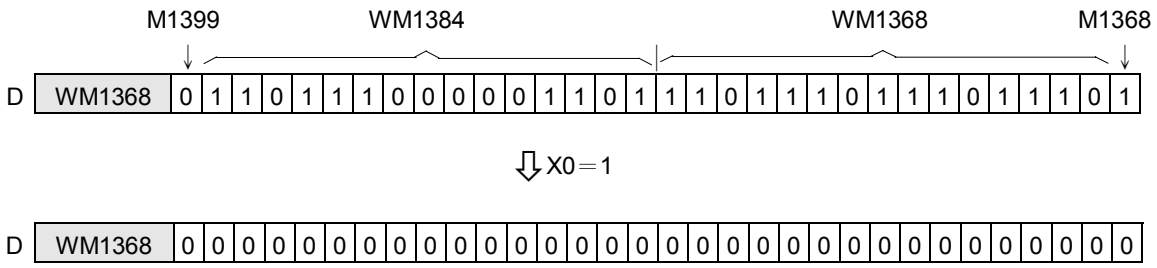


RST D P	RESET (Reset the coil or register to 0)	RST D P



Example 3 | 32-Bit Register Reset

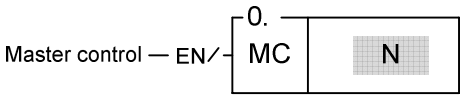
Ladder Diagram



FUN 0 MC	MASTER CONTROL LOOP START	FUN 0 MC

Symbol

Ladder symbol



Operand

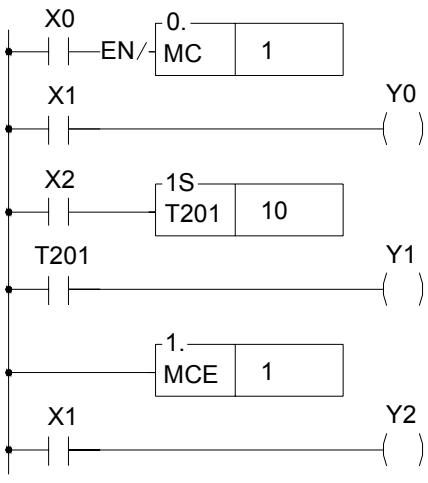
N: Master Control Loop number (N=0~127)
the number N cannot be used repeatedly.

Description

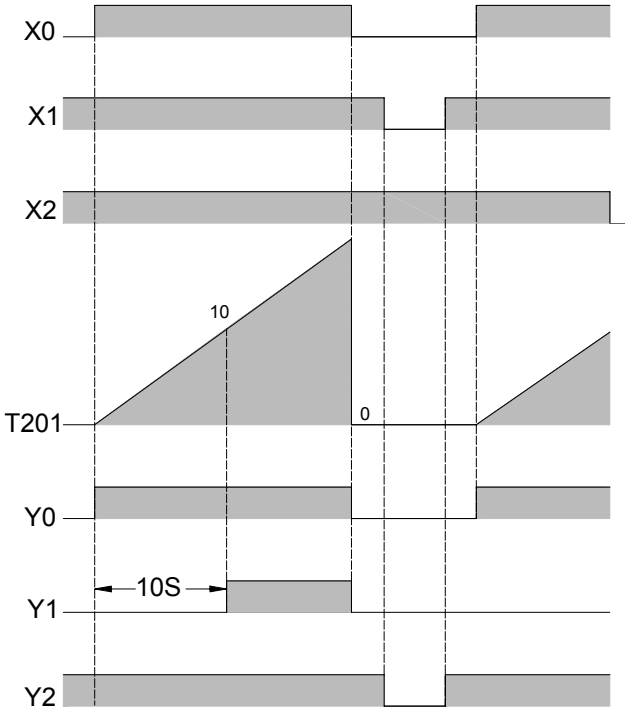
- There are a total of 128 MC loops (N=0~127). Every Master Control Start instruction, MC N, must correspond to a Master Control End instruction, MCE N, which has the same loop number as MC N. They must always be used in pairs and you should also make sure that the MCE N instruction is after the MC N instruction.
- When the Master Control input "EN/" is 1, then this MC N instruction will not be executed, as it does not exist.
- When the Master Control input "EN/" is 0, the master control loop is active, the area between the MC N and MCE N is called the Master Control active loop area. All the status of OUT coils or Timers within Master Control active loop area will be cleared to 0. Other instructions will not be executed.

Example

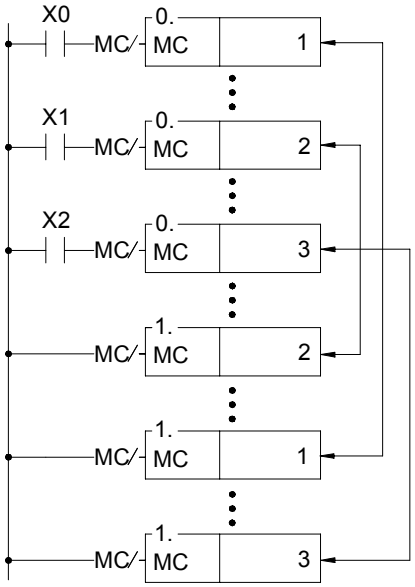
Ladder Diagram



FUN 0 MC	MASTER CONTROL LOOP START	FUN 0 MC



Remark1: MC/MCE instructions can be used in nesting or interleaving as shown to the right:

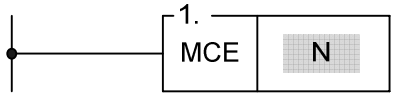


Remark2:
- When M1918=0 and the master input changes from 0 to 1, and if pulse type function instructions exist in the master control loop, then these instructions will have a chance to be executed only once (when the first time the master control input changes from 0 to 1). Afterwards, no matter how many times the master control input changes from 0 to 1, the pulse type function instructions will not be executed again.
- When M1918=1 and the master control input changes from 0 to 1, and if pulse type function instructions exist in the master control loop, then each time the master control input changes from 0 to 1 the pulse type function instructions in the master control loop will be executed as long as the action conditions are satisfied.
- When a counting instruction exists in the master control loop, set M1918 to 0 can avoid counting error.
- When the pulse type function instructions in the master control loop must act upon the 0 to 1 input change by the master control, the flag M1918 should be set to 1.

FUN 1 MCE	MASTER CONTROL LOOP END	FUN 1 MCE

Symbol

Ladder symbol



Operand

N: Master Control End number (N=0~127) N can not be used repeatedly.

Description

- Every MCE N must correspond to a Master Control Start instruction. They must always be used as a pair and you should also make sure that the MCE N instruction is after the MC N instruction. After the MC N instruction has been executed, all output coil status and timers will be cleared to 0 and no other instructions will be executed. The program execution will resume until a MCE instruction which has the same N number as MC N instruction appears.
- MCE instruction does not require an input control because the instruction itself forms a network which other instructions can not connect to it. If the MC instruction has been executed then the master control operation will be completed when the execution of the program reaches the MCE instruction. If MC N instruction has never been executed then the MCE instruction will do nothing.

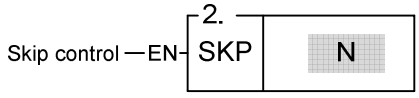
Description

- Please refer to the example and explanations for MC instruction.

FUN 2 SKP	SKIP START	FUN 2 SKP

Symbol

Ladder symbol



Operand

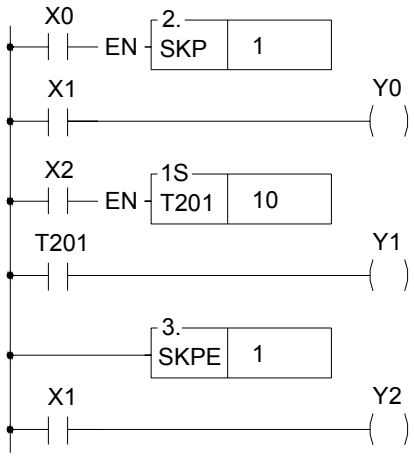
N: Skip loop number (N=0~127),
N can not be used repeatedly.

Description

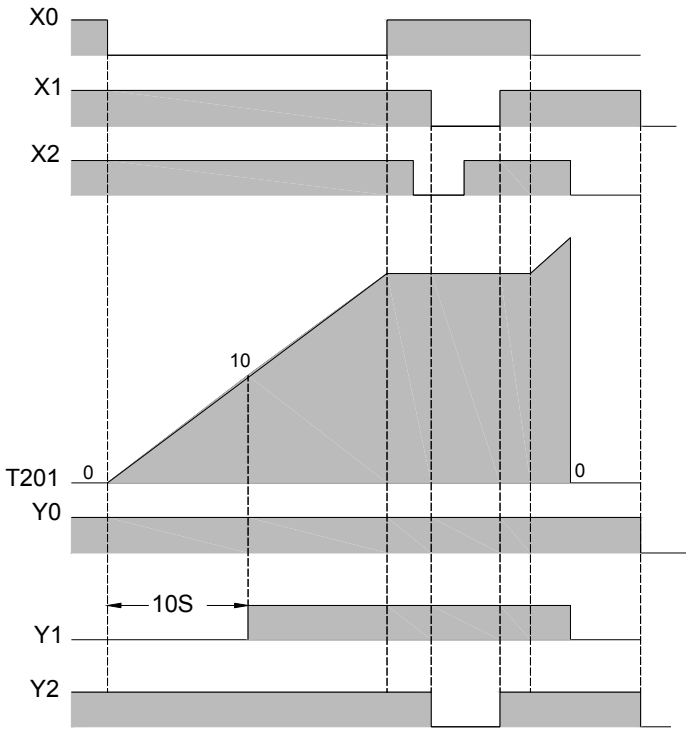
- There are total 128 SKP loops (N=0~127). Every skip start instruction, SKP N, must correspond to a skip end instruction, SKPE N, which has the same loop number as SKP N. They must always be used as a pair and you should also make sure that the SKPE N instruction is after the SKP N instruction.
- When the skip control "EN" is 0, then the Skip Start instruction will not be executed.
- When the skip control "EN" is 1, the range between the SKP N and SKPE N which is so called the Skip active loop area will be skipped, that is all the instructions in this area will not be executed. Therefore the statuses of the discrete or registers in this Skip active loop area will be retained.

Example

Ladder Diagram



FUN 2 SKP	SKIP START	FUN 2 SKP



FUN 3 SKPE	SKIP END	FUN 3 SKPE

Symbol

Ladder symbol

Operand

N： SKIP END Loop number (N=0~127) N can not be used repeatedly.

Description

- Every SKPE N must correspond to a SKP N instruction. They must always be used as a pair and you should also make sure that the SKPE N instruction is behind the SKP N instruction.
- SKPE instruction does not require an input control because the instruction itself forms a network which other instructions can not connect to it. If the SKP N instruction has been executed then the skip operation will be completed when the execution of the program reaches the SKPE N instruction. If SKP N instruction has never been executed then the SKPE instruction will do nothing.

Example

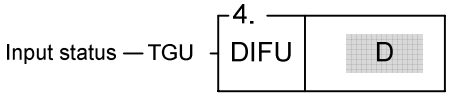
- Please refer to the example and explanations for SKP N instruction.

Remark: SKP/SKPE instructions can be used by nesting or interleaving. The coding rules are the same as for the MC/MCE instructions. Please refer to the section of MC/MCE instructions.

FUN 4 DIFU	DIFFERENTIAL UP	FUN 4 DIFU

Symbol

Ladder symbol



Operand

D: a specific coil number where the result of the Differential Up operation is stored.

Range / Operand	Y	M	SM	S
	Y0 | Y255	M0 | M1911	M1912 | M2001	S0 | S999
D	○	○	○*	○

Description

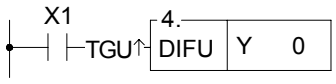
- The DIFU instruction is used to output the up differentiation of a node status (status input to "TGU↑") and the pulse signal resulting from the status change at the rising edge of the "TG↑" for one scan time is stored to a coil specified by D.
- The functionality of this instruction can also be achieved by using a TU contact.

Example

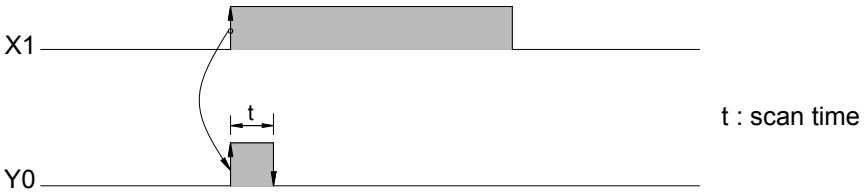
The results of the following two samples are exactly the same

Ladder Diagram

Example 1



Example 2

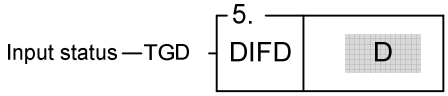


t : scan time

FUN 5 DIFD	DIFFERENTIAL DOWN	FUN 5 DIFD

Symbol

Ladder symbol



Operand

N: a specific coil number where the result of the Differential Down operation is stored.

Operand \ Range	Y	M	SM	S
	Y0 | Y255	M0 | M1911	M1912 | M2001	S0 | S999
D	○	○	○*	○

Description

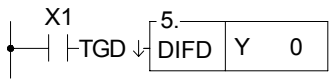
- The DIFD instruction is used to output the down differentiation of a node status (status input to "TGD↓") and the pulse signal resulting from the status change at the falling edge of the "TGD↓" for one scan time is stored to a coil specified by D.
- The functionality of this instruction can also be achieved by using a TD contact.

Example

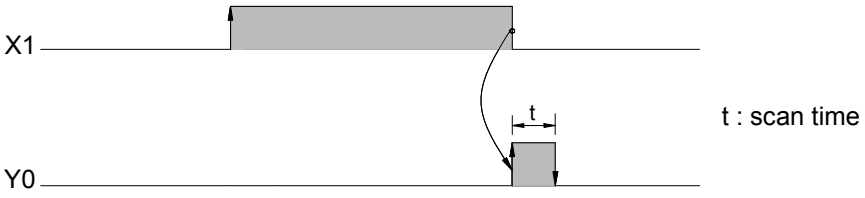
The results of the following two samples are exactly the same.

Ladder Diagram

Example 1



Example 2

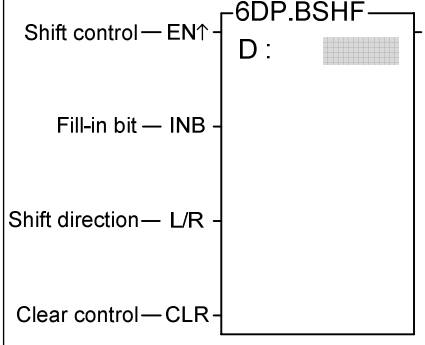


t : scan time

FUN 6 D P BSHF	BIT SHIFT (Shifts the data of the 16-bit or 32-bit register to left or to right by one bit)	FUN 6 D P BSHF

Symbol

Ladder symbol



Operand

D: The register number for shifting

Range / Operand	WY	WM	WS	TMR	CTR	HR	OR	SR	ROR	DR
	WY0 | WY240	WM0 | WM1896	WS0 | WS984	T0 | T255	C0 | C255	R0 | R3839	R3904 | R3967	R3968 | R4167	R5000 | R8071	D0 | D4095
D	○	○	○	○	○	○	○	○*	○*	○

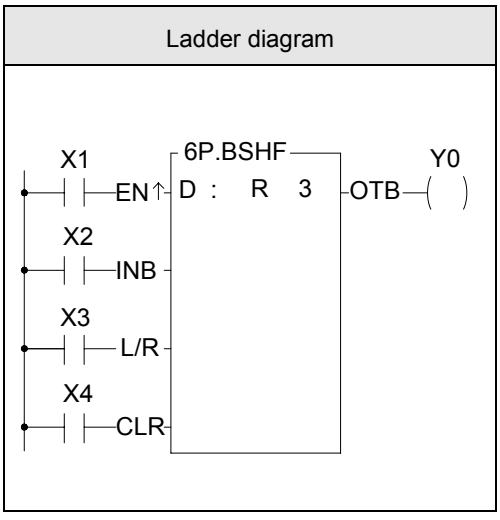
Description

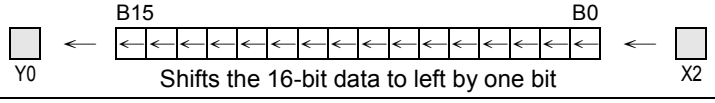
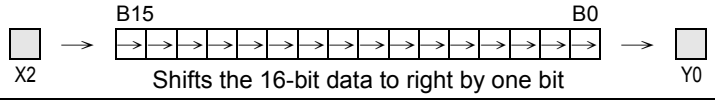
- When the status of clear control "CLR" is at 1, then the data of register D and FO0 will all be cleared to 0. Other input signals are all in effect.
- When the status of clear control is "CLR" at 0, then the shift operation is permissible. When the shift control "EN" =1 or "EN↑" (P instruction) changes from 0 to 1, the data of the register will be shifted to right (L/R=0) or to left (L/R=1) by one bit. The shifted-out bit (MSB when shift to left and LSB when shift to right) for both cases will be sent to FO0. The vacated bit space (LSB when shift to left and MSB when shift to right) due to shift operation will be filled in by the input status of fill-in bit "INB".

Example

Shifts the 16-bit register data

Ladder diagram

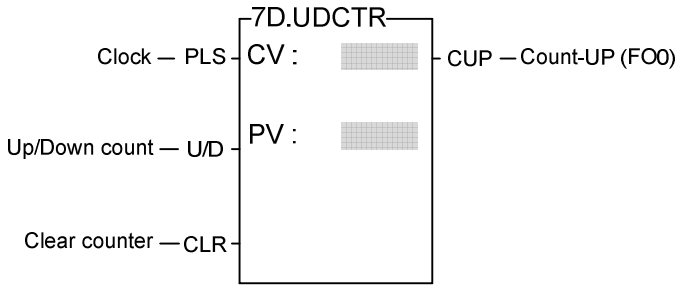


X3=1 (Left shift)	
X3=0 (Right shift)	

FUN 7 D UDCTR	UP/DOWN COUNTER (16-bit or 32-bit up and down 2-phase Counter)	FUN 7 D UDCTR

Symbol

Ladder symbol



Operand

CV: The number of the Up/Down Counter
PV: Preset value of the counter or it's register number

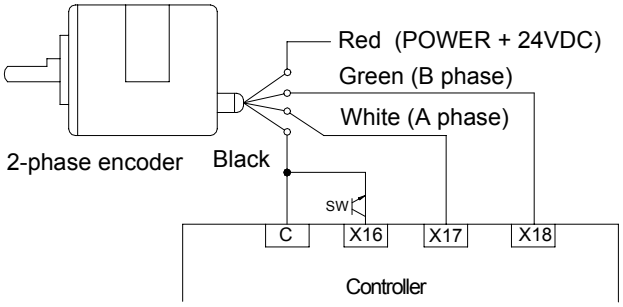
Range / Operand	WX	WY	WM	WS	TMR	CTR	HR	IR	OR	SR	ROR	DR	K
	WX0 | WX240	WY0 | WY240	WM0 | WM1896	WS0 | WS984	T0 | T255	C0 | C255	R0 | R3839	R3840 | R3903	R3904 | R3967	R3968 | R4167	R5000 | R8071	D0 | D4095	16/32-bit +/– number
CV		○	○	○	○	○	○		○	○*	○*	○	
PV	○	○	○	○	○	○	○	○	○	○	○	○	○

Description

- When the clear counter “CLR” is 1, the counter’s CV will be reset to 0 and the counter will not be able to count.
- When the clear counter “CLR” is 0, counting will then be allowed. The nature of the instruction is a P instruction. Therefore, when the clock “PLS” is 0 to 1 (rising edge), the CV will increased by 1 (if U/D=1) or decreased by 1 (if U/D=0).
- When CV=PV, FO0 (“Count-Up) will change to 1”. If there are more clocks input, the counter will continue counting which cause CV≠PV. Then, FO0 will immediately change to 0. This means the “Count-Up” signal will only be equal to 1 if CV=PV, or else it will be equal to 0 (Care should be taken to this difference from the “Count-Up” signal of the general counter).
- The upper limit of up count value is 32767 (16-bit) or 2147483647 (32-bit). After the upper limit is reached, if another up count clock is received, the counting value will become –32768 or -2147483648 (the lower limit of down count).
- The lower limit of down count value is -32767 (16-bit) or -2147483647 (32-bit). After the lower limit is reached, if another down count clock is received, the counting value will become 32768 or 2147483648 (the upper limit of up count).
- If U/D is fixed as 1, the instruction will become a single-phase up count counter. If U/D is fixed as 0, the instruction will become a single-phase down count counter.

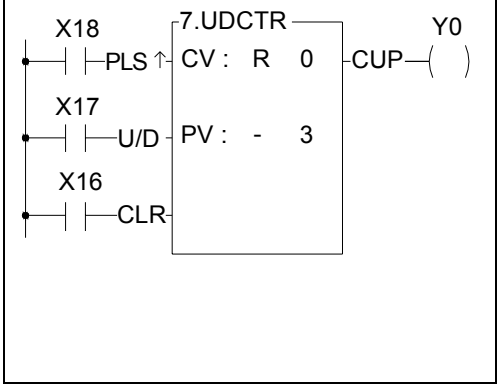
Example

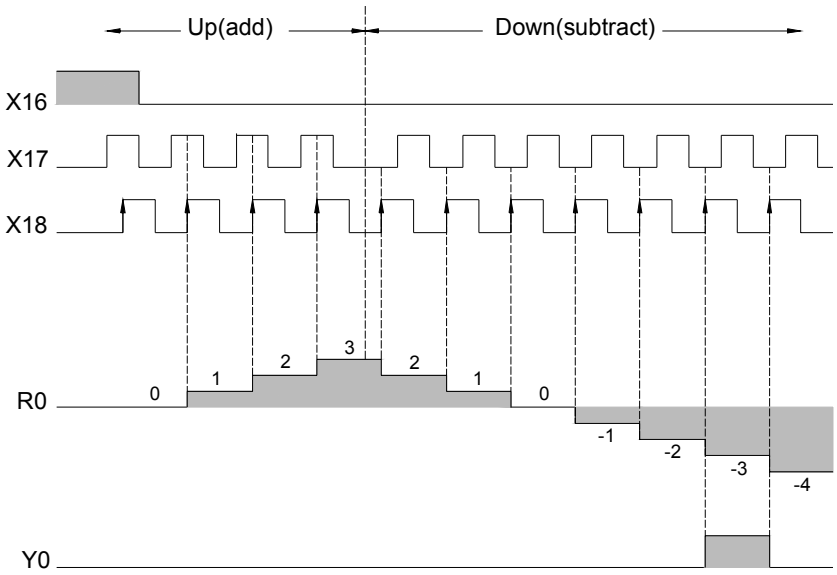
The diagram below is an application example of UDCTR instruction being applied to an encoder.



FUN 7 D UDCTR	UP/DOWN COUNTER (16-bit or 32-bit up/down 2-phase Counter)	FUN 7 D UDCTR

Ladder Diagram





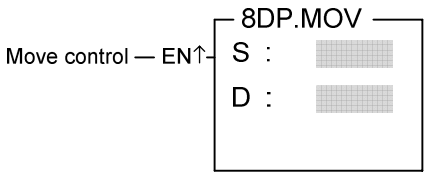
Remark 1: Since the counting operation of UDCTR is implemented by software scanning, therefore if the clock speed is faster than the scan speed, lose count may then happen (generally the clock should not exceed 20Hz depending on the size of the program). Please use the software or hardware high-speed counter in the controller. Refer to the “High Speed Counter Application” in the Advanced Manual.

Remark 2: In order to ensure the proper counting, the sustain time of the status of clock input should greater than 1 scan time.

FUN 8 D P MOV	MOVE (Moves data from S to D)	FUN 8 D P MOV

Description

Ladder symbol



Operand

S: Source register number
D: Destination register number
The S, N, D may combine with V, Z, P0~P9 to serve indirect addressing

Range / Operand	WX	WY	WM	WS	TMR	CTR	HR	IR	OR	SR	ROR	DR	K	XR
	WX0 | WX240	WY0 | WY240	WM0 | WM1896	WS0 | WS984	T0 | T255	C0 | C255	R0 | R3839	R3840 | R3903	R3904 | R3967	R3968 | R4167	R5000 | R8071	D0 | D4095	16/32-bit +/− number	V、Z P0~P9
S	○	○	○	○	○	○	○	○	○	○	○	○	○	○
D		○	○	○	○	○	○		○	○*	○*	○		○

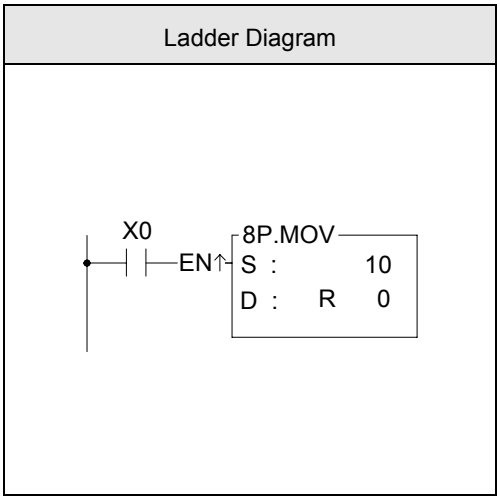
Description

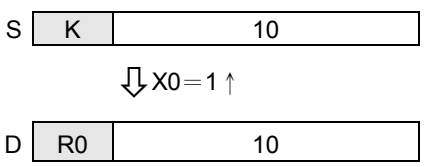
- Move (write) the data of S to a specified register D when the move control input "EN" =1 or "EN↑" (P instruction) changes from 0 to 1.

Example

Writes a constant data into a 16-bit register.

Ladder Diagram

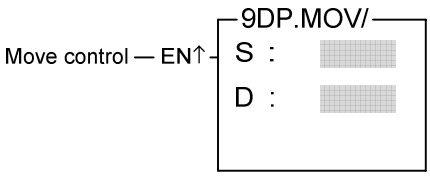




FUN 9 D P MOV/	MOVE INVERSE (Inverts the data of S and moves the result to a specified device D)	FUN 9 D P MOV/

Symbol

Ladder symbol



Operand

S: Source register number
D: Destination register number
S, N, D may combine with V, Z, P0~P9 to serve indirect addressing

Range / Operand	WX	WY	WM	WS	TMR	CTR	HR	IR	OR	SR	ROR	DR	K	XR
	WX0 | WX240	WY0 | WY240	WM0 | WM1896	WS0 | WS984	T0 | T255	C0 | C255	R0 | R3839	R3840 | R3903	R3904 | R3967	R3968 | R4167	R5000 | R8071	D0 | D4095	16/32-bit +/– number	V、Z P0~P9
S	○	○	○	○	○	○	○	○	○	○	○	○	○	○
D		○	○	○	○	○	○		○	○*	○*	○		○

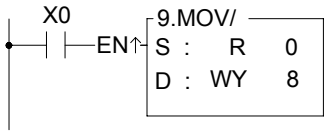
Description

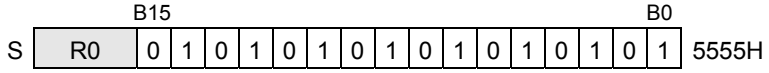
- Inverts the data of S (changes the status from 0 to 1 and from 1 to 0) and moves the results to a specified register D when the move control input "EN" =1 or "EN↑" (P instruction) changes from 0 to1.

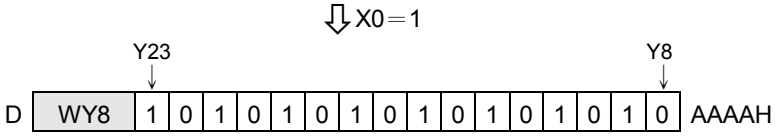
Example

Moves the inverted data of a 16-bit register to another 16-bit register.

Ladder Diagram



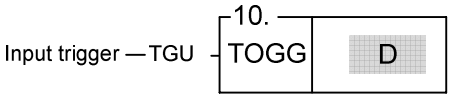




FUN 10 TOGG	TOGGLE SWITCH (Changes the output status when the rising edge of control input occur)	FUN 10 TOGG

Symbol

Ladder symbol



Operand

D: the coil number of the toggle switch

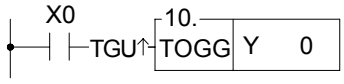
Range / Operand	Y	M	SM	S
	Y0 | Y255	M0 | M1911	M1912 | M2001	S0 | S999
D	○	○	○*	○

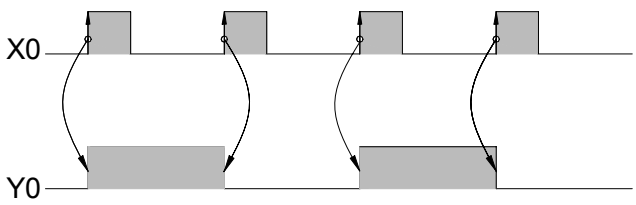
Description

- The coil D changes its status (from 1 to 0 and from 0 to 1) each time the input "TGU" is triggered from 0 to 1 (rising edge).

Example

Ladder Diagram

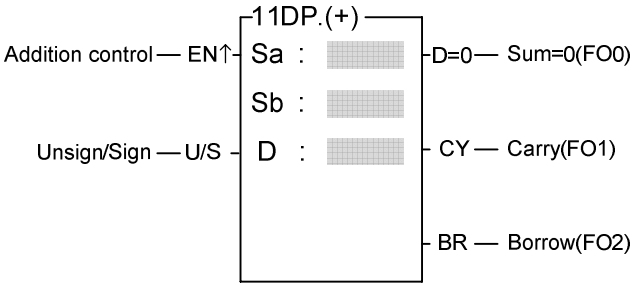




FUN 11 D P (+)	ADDITION (Performs addition of the data specified at Sa and Sb and stores the result in D)	FUN 11 D P (+)

Symbol

Ladder symbol



Operand

Sa: Augend
Sb: Addend
D : Destination register to store the results of the addition
Sa, Sb, D may combine with V, Z, P0~P9 to serve indirect addressing

Range / Operand	WX	WY	WM	WS	TMR	CTR	HR	IR	OR	SR	ROR	DR	K	XR
	WX0 | WX240	WY0 | WY240	WM0 | WM1896	WS0 | WS984	T0 | T255	C0 | C255	R0 | R3839	R3840 | R3903	R3904 | R3967	R3968 | R4167	R5000 | R8071	D0 | D4095	16/32-bit +/− number	V、Z P0~P9
Sa	○	○	○	○	○	○	○	○	○	○	○	○	○	○
Sb	○	○	○	○	○	○	○	○	○	○	○	○	○	○
D		○	○	○	○	○	○		○	○*	○*	○		○

Description

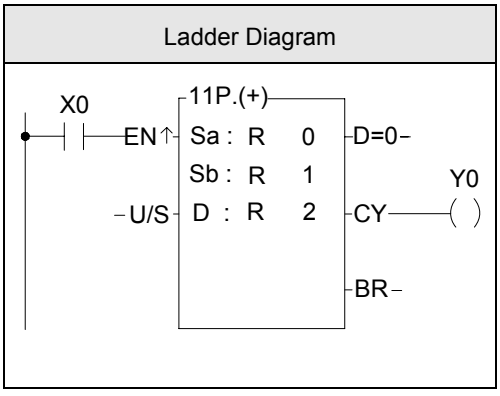
- Performs the addition of the unsigned data specified at Sa and Sb and writes the results to a specified register D when "U/S" =0 and the addition control "EN" =1 or "EN↑" (P instruction) changes from 0 to 1. If the result of addition is equal to 0 then set FO0 to 1. If carry occurs (the result exceeds 32767 or 2147483647) then set FO1 to 1. If borrow occurs (adding negative numbers resulting in a sum less than -32768 or -2147483648), then set the FO2 to 1. All the FO statuses are retained until this instruction is executed again and overwritten by a new result.

- Performs the addition of the signed data specified at Sa and Sb and writes the results to a specified register D when "U/S" =1 and the addition control "EN" =1 or "EN↑" (P instruction) changes from 0 to 1. If the result of addition is equal to 0 then set FO0 to 1. If carry occurs (the result exceeds 65535 or 4294967295) then set FO1 to 1.

Example

16-bit addition

Ladder Diagram



Sa	R0	12345	R0＋R1＝32770
Sb	R1	20425	

⇩X0＝1 ↑

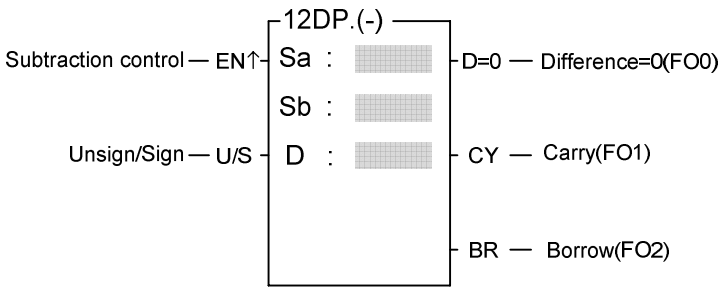
D	R2	2	32768+2=32770

Y0＝1 (carry 1 represents ＋32768)

FUN 12 D P (－)	SUBTRACTION (Performs subtraction of the data specified at Sa and Sb and stores the result in D)	FUN 12 D P (－)

Symbol

Ladder symbol



Operand

Sa: Minuend
Sb: Subtrahend
D : Destination register to store the results of the subtraction
Sa, Sb, D may combine with V, Z, P0~P9 to serve indirect addressing

Range / Operand	WX	WY	WM	WS	TMR	CTR	HR	IR	OR	SR	ROR	DR	K	XR
	WX0 | WX240	WY0 | WY240	WM0 | WM1896	WS0 | WS984	T0 | T255	C0 | C255	R0 | R3839	R3840 | R3903	R3904 | R3967	R3968 | R4167	R5000 | R8071	D0 | D4095	16/32-bit +/– number	V、Z P0~P9
Sa	○	○	○	○	○	○	○	○	○	○	○	○	○	○
Sb	○	○	○	○	○	○	○	○	○	○	○	○	○	○
D		○	○	○	○	○	○		○	○*	○*	○		○

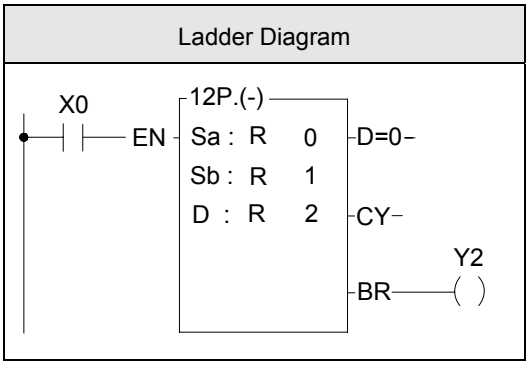
Description

- Performs the subtraction of the unsigned data specified at Sa and Sb and writes the results to a specified register D when "U/S" =0 and the subtraction control "EN" =1 or "EN↑" (P instruction) changes from 0 to 1. If the result of subtraction is equal to 0 then set FO0 to 1. If carry occurs (subtracting a negative number from a positive number and the result exceeds 32767 or 2147483647), then set FO1 to 1. If borrow occurs (subtracting a positive number from a negative number and the resulted difference is less than -32768 or -2147483648), then set FO2 to 1. All the FO statuses are retained until this instruction is executed again and overwritten by a new result.
- Performs the subtraction of the signed data specified at Sa and Sb and writes the results to a specified register D when "U/S" =1 and the subtraction control "EN" =1 or "EN↑" (P instruction) changes from 0 to 1. If the result of subtraction is equal to 0 then set FO0 to 1. If borrow occurs (Sa – Sb < 0), then set FO2 to 1.

Example

16-bit subtraction

Ladder Diagram



Sa	R0	－5	R0－R1＝－32772
Sb	R1	32767	

⇩ X0＝1 ↑

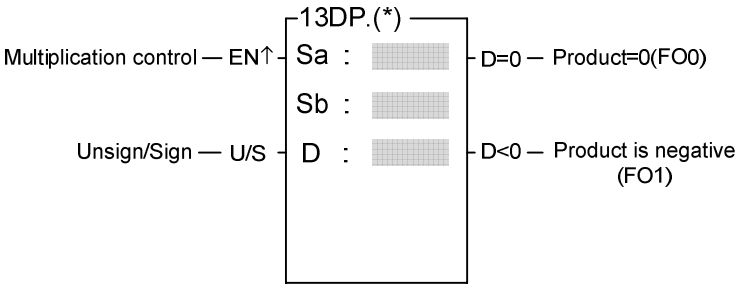
D	R2	－4	－32768－4＝－32772

Y2＝1（borrow 1 represents －32768）Please refer to section 5.4

FUN 13 D P (＊)	MULTIPLICATION (Performs multiplication of the data specified at Sa and Sb and stores the result in D)	FUN 13 D P (＊)

Symbol

Ladder symbol



Operand

Sa: Multiplicand
Sb: Multiplier
D : Destination register to store the results of the multiplication.
Sa, Sb, D may combine with V, Z, P0~P9 to serve indirect addressing

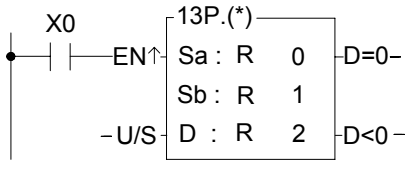
Range / Operand	WX	WY	WM	WS	TMR	CTR	HR	IR	OR	SR	ROR	DR	K	XR
	WX0 | WX240	WY0 | WY240	WM0 | WM1896	WS0 | WS984	T0 | T255	C0 | C255	R0 | R3839	R3840 | R3903	R3904 | R3967	R3968 | R4167	R5000 | R8071	D0 | D4095	16/32-bit +/– number	V、Z P0~P9
Sa	○	○	○	○	○	○	○	○	○	○	○	○	○	○
Sb	○	○	○	○	○	○	○	○	○	○	○	○	○	○
D		○	○	○	○	○	○		○	○*	○*	○		○

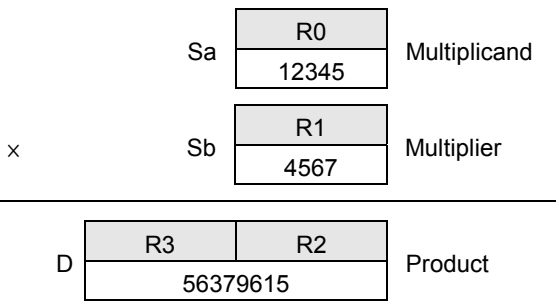
Description

- Performs the multiplication of the unsigned data specified at Sa and Sb and writes the results to a specified register D when "U/S" =0 and the multiplication control "EN" =1 or "EN↑" (P instruction) changes from 0 to 1. If the product of multiplication is equal to 0 then set FO0 to 1. If the product is a negative number, then set FO1 to 1.
- Performs the multiplication of the signed data specified at Sa and Sb and writes the results to a specified register D when "U/S" =1 and the multiplication control "EN" =1 or "EN↑" (P instruction) changes from 0 to 1. If the product of multiplication is equal to 0 then set FO0 to 1.

Example 1 16-bit multiplication

Ladder Diagram

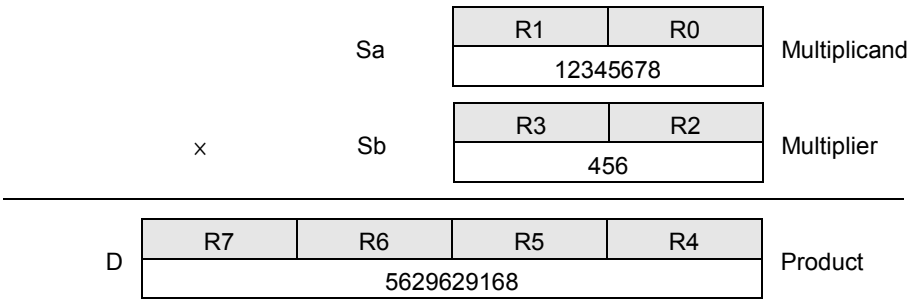




FUN 13 D P (＊)	MULTIPLICATION (Performs multiplication of the data specified at Sa and Sb and stores the result in D)	FUN 13 D P (＊)

Example 2 32-bit multiplication

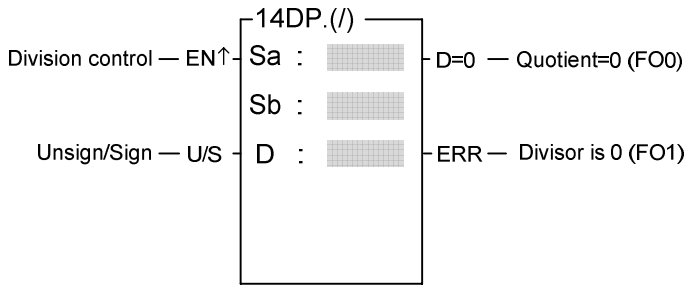
Ladder Diagram



FUN 14 D P (／)	DIVISION (Performs division of the data specified at Sa and Sb and stores the result in D)	FUN 14 D P (／)

Symbol

Ladder symbol



Operand

Sa: Dividend
Sb: Divisor
D : Destination register to store the results of the division.
Sa, Sb, D may combine with V, Z, P0~P9 to serve indirect addressing

Range / Operand	WX	WY	WM	WS	TMR	CTR	HR	IR	OR	SR	ROR	DR	K	XR
	WX0 | WX240	WY0 | WY240	WM0 | WM1896	WS0 | WS984	T0 | T255	C0 | C255	R0 | R3839	R3840 | R3903	R3904 | R3967	R3968 | R4167	R5000 | R8071	D0 | D4095	16/32-bit +/– number	V、Z P0~P9
Sa	○	○	○	○	○	○	○	○	○	○	○	○	○	○
Sb	○	○	○	○	○	○	○	○	○	○	○	○	○	○
D		○	○	○	○	○	○		○	○*	○*	○		○

Description

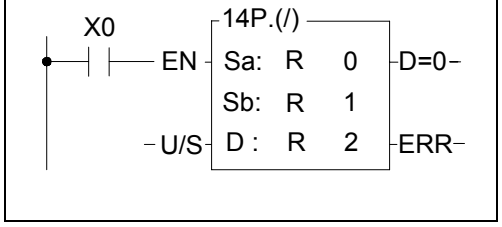
- Performs the division of the unsigned data specified at Sa and Sb and writes the quotient and remainder to registers specified by register D when "U/S" =0 and the division control "EN" =1 or "EN↑" (P instruction) changes from 0 to 1. If the quotient of division is equal to 0 then set FO0 to 1. If the divisor Sb=0 then set the error flag FO1 to 1 without executing the instruction.

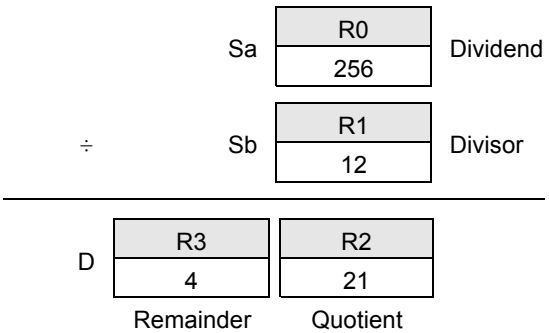
- Performs the division of the signed data specified at Sa and Sb and writes the quotient and remainder to registers specified by register D when "U/S" =1 and the division control "EN" =1 or "EN↑" (P instruction) changes from 0 to 1. If the quotient of division is equal to 0 then set FO0 to 1. If the divisor Sb=0 then set the error flag FO1 to 1 without executing the instruction.

Example 1

16-bit division

Ladder Diagram

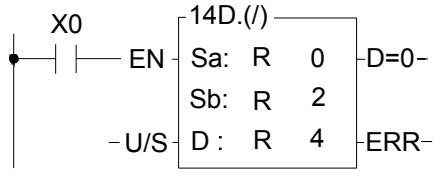




FUN 14 D P (／)	DIVISION (Performs division of the data specified at Sa and Sb and stores the result in D)	FUN 14 D P (／)

Example 2 32-bit division

Ladder Diagram



	Sa	R1	R0	Dividend
		2147483647		
÷	Sb	R3	R2	Divisor
		1234567		

D	R7	R6	R5	R4
	571634		1739	
	Remainder		Quotient	

FUN 15 D P (+1)	INCREMENT (Adds 1 to the D value)	FUN 15 D P (+1)

Symbol

Ladder symbol

Operand

D : The register to be increased
D may combine with V, Z, P0~P9 to serve indirect addressing

Range / Operand	WY	WM	WS	TMR	CTR	HR	OR	SR	ROR	DR	XR
	WY0 | WY240	WM0 | WM1896	WS0 | WS984	T0 | T255	C0 | C255	R0 | R3839	R3904 | R3967	R3968 | R4167	R5000 | R8071	D0 | D4095	V、Z P0~P9
D	○	○	○	○	○	○	○	○*	○*	○	○

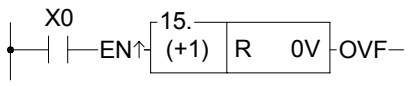
Description

- Adds 1 to the register D when the increment control "EN" =1 or "EN↑" (P instruction) changes from 0 to 1. If the value of D is already at the upper limit of positive number 32767 or 2147483647, adding one to this value will change it to the lower limit of negative number -32768 or -2147483648. At the same time, the overflow flag FO0 (OVF) is set to 1.

Example

16-bit increment register

Ladder diagram



When V＝100，0＋100＝100

D	R100	1

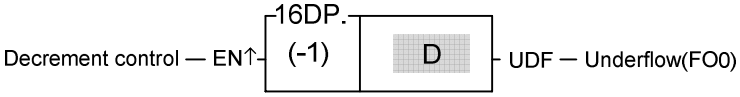
⇩X0＝1

D	R100	2

FUN 16 D P (－1)	DECREMENT (Subtracts 1 from the D value)	FUN 16 D P (－1)

Symbol

Ladder symbol



Operand

D : The register to be decreased
D may combine with V, Z, P0~P9 to serve indirect addressing

Range / Operand	WY	WM	WS	TMR	CTR	HR	OR	SR	ROR	DR	XR
	WY0 | WY240	WM0 | WM1896	WS0 | WS984	T0 | T255	C0 | C255	R0 | R3839	R3904 | R3967	R3968 | R4167	R5000 | R8071	D0 | D4095	V、Z P0～P9
D	○	○	○	○	○	○	○	○*	○*	○	○

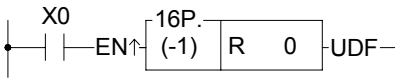
Description

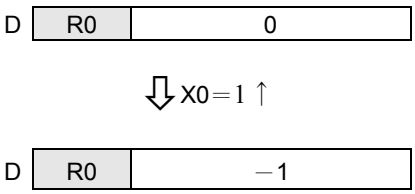
- Subtracts 1 from the register D when the decrement control "EN" =1 or "EN↑" (P instruction) changes from 0 to 1. If the value of D is already at the lower limit of negative number -32768 or -2147483648, subtracting one from this value will change it to the upper limit of positive number 32767 or 2147483647. At the same time, the underflow flag FO0 (UDF) is set to 1.

Example

16-bit decrement register

Ladder diagram

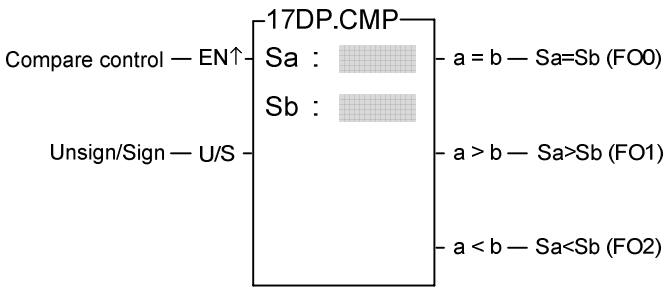




FUN 17 D P CMP	COMPARE (Compares the data of Sa and Sb and outputs the results to function Outputs)	FUN 17 D P CMP

Symbol

Ladder symbol



Operand

Sa: The register to be compared

Sb: The register to be compared

Sa, Sb may combine with V, Z, P0~P9 to serve indirect addressing

Range / Ope-rand	WX	WY	WM	WS	TMR	CTR	HR	IR	OR	SR	ROR	DR	K	XR
	WX0 | WX240	WY0 | WY240	WM0 | WM1896	WS0 | WS984	T0 | T255	C0 | C255	R0 | R3839	R3804 | R3903	R3904 | R3967	R3968 | R4167	R5000 | R8071	D0 | D4095	16/32 bit +/-number	V 、 Z P0 ~ P9
Sa	○	○	○	○	○	○	○	○	○	○	○	○	○	○
Sb	○	○	○	○	○	○	○	○	○	○	○	○	○	○

Description

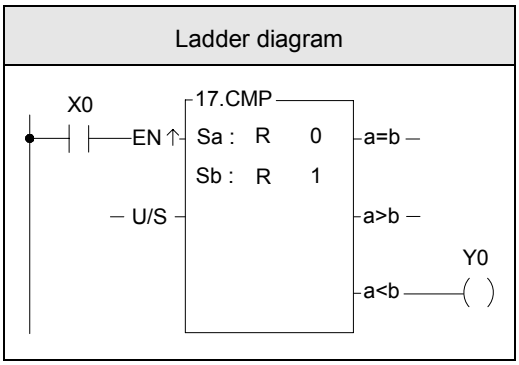
- Compares the unsigned data of Sa and Sb when "U/S" =0 and the compare control "EN" =1 or "EN↑" (P instruction) changes from 0 to 1. If the data of Sa is equal to Sb, then set FO0 to 1. If the data of Sa>Sb, then set FO1 to 1. If the data of Sa<Sb, then set FO2 to 1.

- Compares the signed data of Sa and Sb when "U/S" =1 and the compare control "EN" =1 or "EN↑" (P instruction) changes from 0 to 1. If the data of Sa is equal to Sb, then set FO0 to 1. If the data of Sa>Sb, then set FO1 to 1. If the data of Sa<Sb, then set FO2 to 1.

Example

Compares the data of 16-bit register

Ladder diagram

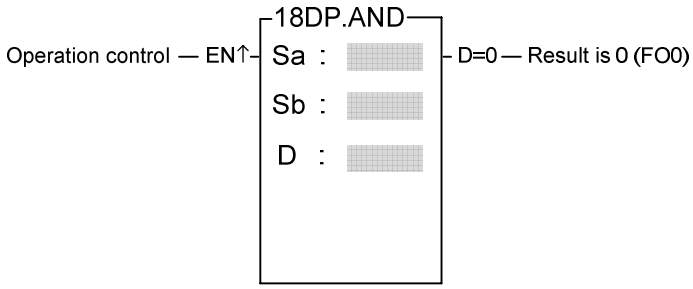


- From the above example, we first assume the data of R0 is 1 and R1 is 2, and then compare the data by executing the CMP instruction. The FO0 and FO1 are set to 0 and FO2 (a<b) is set to 1 since a<b.
- If you want to have the compound results, such as≧、≦、< > etc., please send =、< and > results to relay first and then combine the result from the relays.
- M1919=0, when this command in not executed, FO0, FO1, FO2 will remain in the status at last execution.
- M1919=1, when this command in not executed, FO0, FO1, FO2 are all cleared to 0.
- Control M1919 properly to obtain memory-holding function for functional command output.

FUN 18 D P AND	LOGICAL AND	FUN 18 D P AND

Symbol

Ladder symbol



Operand

Sa: The register to be ANDed
Sb: The register to be ANDed
D : The register to store the result of AND
The Sa, Sb, D may combine with V, Z, P0~P9 to serve indirect addressing application

Range / Operand	WX	WY	WM	WS	TMR	CTR	HR	IR	OR	SR	ROR	DR	K	XR
	WX0 | WX240	WY0 | WY240	WM0 | WM1896	WS0 | WS984	T0 | T255	C0 | C255	R0 | R3839	R3804 | R3903	R3904 | R3967	R3968 | R4167	R5000 | R8071	D0 | D4095	16/32 bit +/-number	V 、 Z P0 ~ P9
Sa	○	○	○	○	○	○	○	○	○	○	○	○	○	○
Sb	○	○	○	○	○	○	○	○	○	○	○	○	○	○
D		○	○	○	○	○	○		○	○*	○*	○		

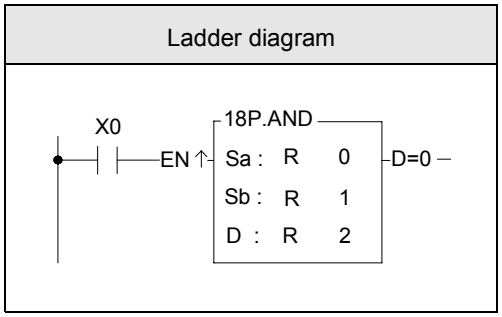
Description

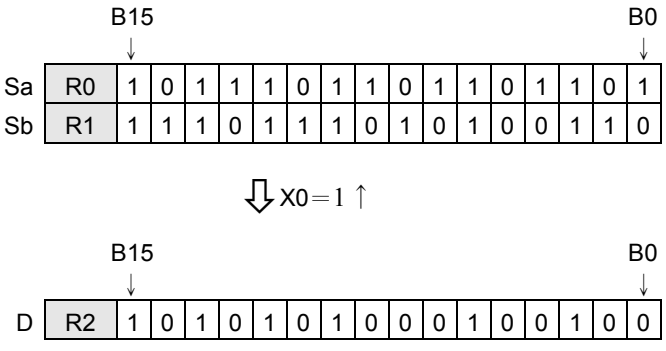
- Performs logical AND operation for the data of Sa and Sb when the operation control "EN" =1 or "EN↑" (P instruction) changes from 0 to 1. This operation compares the corresponding bits of Sa and Sb (B0~B15 or B0~B31). The bit in the D is set to 1 if both of the corresponding bits data of Sa and Sb is 1. The bit in the D is set to 0 if one of the corresponding bits is 0.

Example

Operation of 16-bit logical AND

Ladder diagram

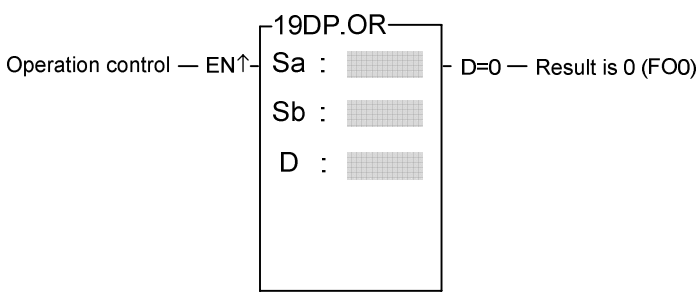




FUN 19 D P OR	LOGICAL OR	FUN 19 D P OR

Symbol

Ladder symbol



Operand

Sa: The register to be ORed
Sb: The register to be ORed
D : The register to store the result of OR
The Sa, Sb, D may combine with V, Z, P0~P9 to serve indirect addressing

Range / Operand	WX	WY	WM	WS	TMR	CTR	HR	IR	OR	SR	ROR	DR	K	XR
	WX0 | WX240	WY0 | WY240	WM0 | WM1896	WS0 | WS984	T0 | T255	C0 | C255	R0 | R3839	R3804 | R3903	R3904 | R3967	R3968 | R4167	R5000 | R8071	D0 | D4095	16/32 bit +/-number	V、Z P0~P9
Sa	○	○	○	○	○	○	○	○	○	○	○	○	○	○
Sb	○	○	○	○	○	○	○	○	○	○	○	○	○	○
D		○	○	○	○	○	○		○	○*	○*	○		

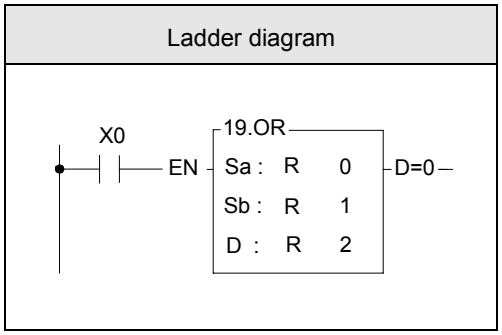
Description

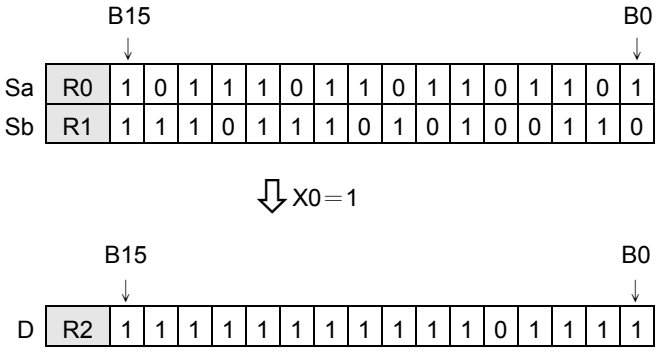
- Performs logical OR operation for the data of Sa and Sb when the operation control "EN" =1 or "EN↑" (P instruction) changes from 0 to 1. This operation compares the corresponding bits of Sa and Sb (B0~B15 or B0~B31). The bit in the D is set to 1 if one of the corresponding of Sa or Sb is 1. The bit in the D is set to 0 if both of the corresponding bits of Sa and Sb is 0.

Example

Operation of 16-bit logical OR

Ladder diagram

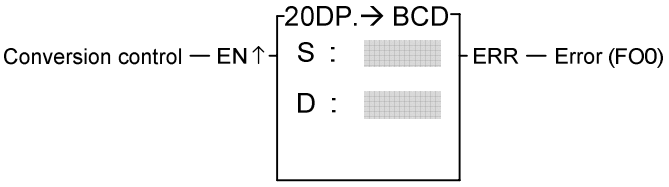




FUN 20 D P →BCD	BIN TO BCD CONVERSION (Converts BIN data of the device specified at S into BCD and stores the result in D)	FUN 20 D P →BCD

Symbol

Ladder symbol



Operand

S : The register to be converted
D : The register to store the converted data (BCD code)
The S, D may combine with V, Z, P0~P9 to serve indirect addressing

Range / Operand	WX	WY	WM	WS	TMR	CTR	HR	IR	OR	SR	ROR	DR	K	XR
	WX0 | WX240	WY0 | WY240	WM0 | WM1896	WS0 | WS984	T0 | T255	C0 | C255	R0 | R3839	R3804 | R3903	R3904 | R3967	R3968 | R4167	R5000 | R8071	D0 | D4095	16/32 bit +/- number	V、Z P0~P9
S	○	○	○	○	○	○	○	○	○	○	○	○	○	○
D		○	○	○	○	○	○		○	○*	○*	○		○

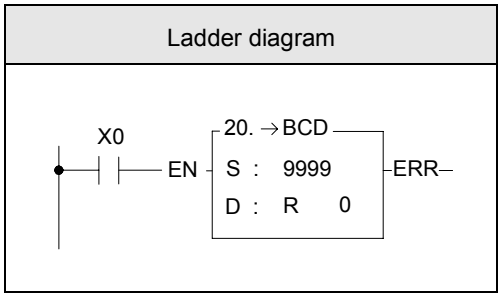
Description

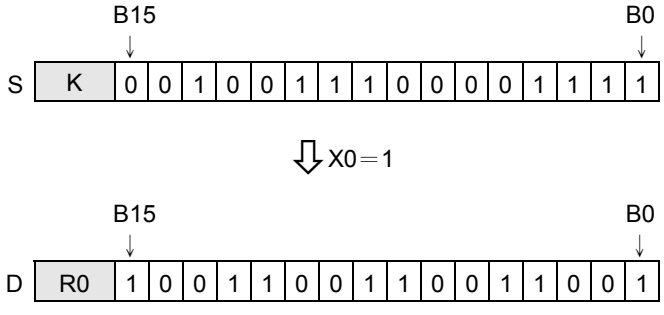
- WSZ-controller uses binary code to store and to execute calculations. If want to send the internal controller data to the external displays such as seven-segment displays, it is more convenient for us to read the result on screen by converting the BIN data to BCD data. For example, it is clearer for us to read the reading "12" instead of the binary code "1100."
- Converts BIN data of the device specified at S into BCD and writes the result in D when the conversion control "EN" =1 or "EN↑" (P instruction) changes from 0 to 1. If the data in S is not a BCD value (0~9999 or 0~9999999), then the error flag FO0 is set to 1 and the old data of D are retained.

Example

16-bit BIN to BCD conversion

Ladder diagram

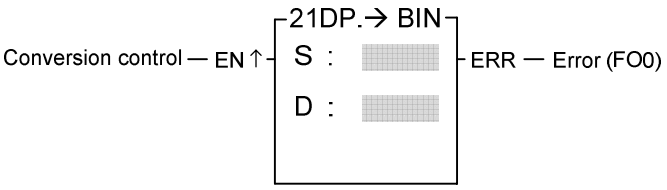




FUN 21 D P →BIN	BCD TO BIN CONVERSION (Converts BCD data of the device specified at S into BIN and stores the result in D)	FUN 21 D P →BIN

Symbol

Ladder symbol



Operand

S : The register to be converted

D : The register to store the converted data (BIN code)

The S, D may combine with V, Z, P0~P9 to serve indirect addressing

Range / Operand	WX	WY	WM	WS	TMR	CTR	HR	IR	OR	SR	ROR	DR	XR
	WX0 | WX240	WY0 | WY240	WM0 | WM1896	WS0 | WS984	T0 | T255	C0 | C255	R0 | R3839	R3840 | R3903	R3904 | R3967	R3968 | R4167	R5000 | R8071	D0 | D4095	V、Z P0~P9
S	○	○	○	○	○	○	○	○	○	○	○	○	○
D		○	○	○	○	○	○		○	○*	○*	○	○

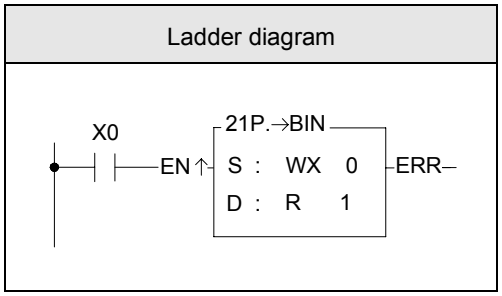
Description

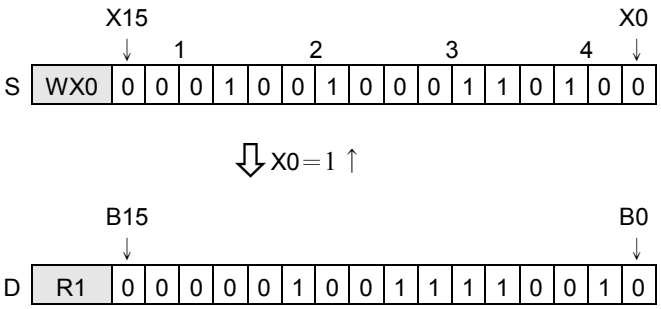
- The decimal (BCD) data must be converted to binary (BIN) data first in order for controller to accept the data which is originally in decimal unit (BCD code) inputted from external device such as digital switch because the BCD data can not be accepted by controller for its operations.
- Converts BCD data of the device specified at S into BIN and writes the result in D when the operation control "EN" =1 or "EN↑" (P instruction) changes from 0 to 1. If the data in S is not in BCD, then the error flag FO0 is set to 1 and the old data of D are retained.
- Constant is converted to BIN automatically when store in program and can not be used as a source operand of this function.

Example

16-bit BCD to BIN conversion

Ladder diagram

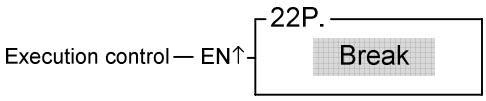




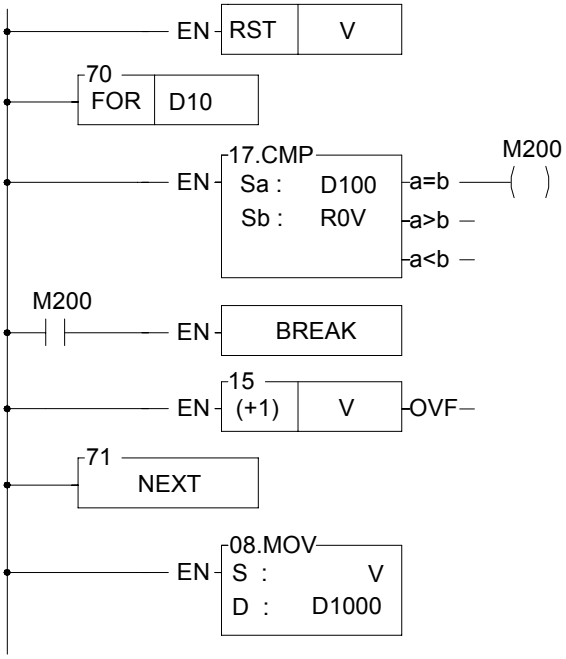
Chapter 7 Advanced Function Instructions

FUN22 P BREAK	BREAK FROM FOR AND NEXT LOOP （BREAK）	FUN22 P BREAK

Ladder symbol



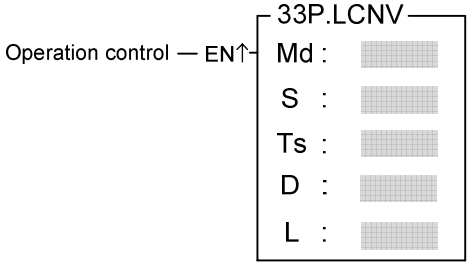
- When execution control "EN"=1 or "EN↑" （P instruction) changes from 0 to 1, it will terminate the FOR and NEXT program loop.
- The program within the FOR and NEXT loop will be executed N times (N is assigned by FOR instruction) successively, but if it is necessary to terminate the execution loop less than N times, the BREAK instruction is necessary to apply.
- The BREAK instruction must be located within the FOR and NEXT program loop.



Description：The loop count used to execute the FOR and NEXT program loop is assigned by register D10；the program within the FOR and NEXT loop is designed to search the same data storing in D100 from the register table starting at R0.If it finds, the searching loop will be terminated and then it goes to execute the program after the NEXT instruction；If it doesn't find, the searching loop will be executed N times (N is the content of D10) and then it goes to execute the program after the NEXT instruction.
M200 tells the status and D1000 is the pointer of searching.

FUN33 P LCNV	Linear Conversion (LCNV)	FUN33 P LCNV

Ladder symbol



Md: Operation mode, 0～3

S: Starting address of the source data

Ts: Starting address of the parameter table for conversion

D: Starting address to store the result

L: Quantity of conversion entry, 1～64

Range / Operand	HR	IR	ROR	DR	K
	R0 | R3839	R3840 | R3903	R5000 | R8071	D0 | D3999	
Md					0～3
S	○	○	○	○	
Ts	○		○	○	
D	○		○*	○	
L	○		○	○	1～64

- When the analog input module being used for the analog measurement, the raw reading value of the analog input can be converted into the engineering range through this instruction for display or for proceeding control operation.
- For process measurement calibration, making the linear conversion for the engineering process variable, which the measurement value from the Controller's can be corrected by the value from the standard meter's through this instruction.
- When operation control "EN"=1or "EN↑" (P instruction) changes from 0 to 1, this instruction will perform the linear conversion operation according to the mode selection, where S is the starting address of the source data, Ts is the starting address of the conversion parameter table, D is the starting address to store the converted result, and L is the quantity of conversion entry.
- There are two expressions to meet the suitable application:

Expression 1: Two points calibration method

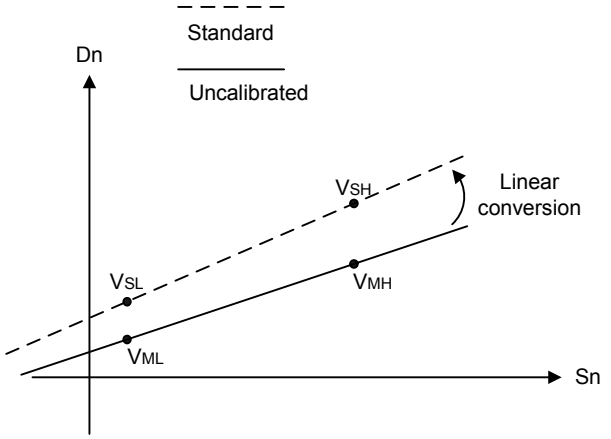
Fill the conversion parameter table with the low value of measurement(VML), high value of measurement(VMH), and the corresponding low value of standard (VSL), high value of standard(VSH); the converted result(Dn) will be generated from the source data(Sn) through the formula shown below:

A = (VSL－VSH／VML－VMH)×10000

B = VSL－(VML×A／10000)

Dn = (Sn×A／10000)＋B

- The range of operands VSL,VSH, VML,VMH,Sn and Dn are between -32768 ~ 32767
- For analog input scaling, where
  VML=Minimum of analog input
  VMH=Maximum of analog input
  VSL=Minimum of engineering range
  VSH=Maximum of engineering range



FUN33 P LCNV	Linear Conversion (LCNV)	FUN33 P LCNV

Expression 2: Multiplication + Offset method

Fill the conversion parameter table with the values of multiplier (A), divisor (B) and offset(C);
The converted result (Dn) will be generated from the source data (Sn) through the formula shown below:

Dn =[(Sn×A)／B]＋C

The range of each operand as below:

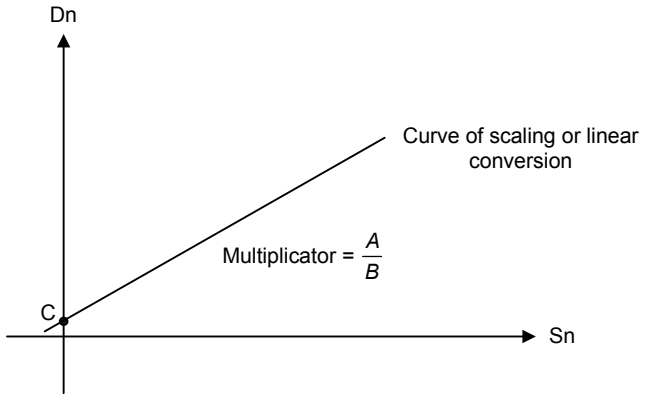
A = 1 ～ 65535

B = 1 ～ 65535

C = -32768 ～ 32767

Sn = 0 ～ 65535

Dn = -32768 ～ 32767

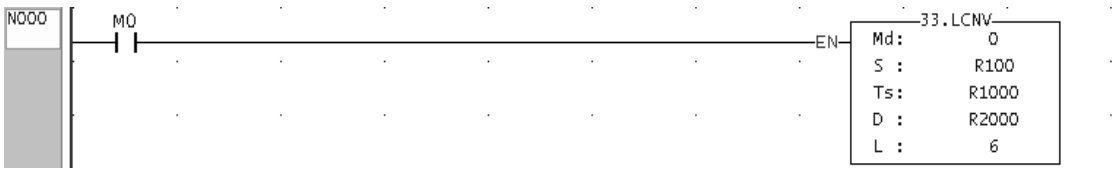


Description of operation mode:

1. When Md =0, the linear conversion works by expression 1, and all source data share the same parameters VML, VMH, VSL and VSH for conversion.
2. When Md =1, the linear conversion works by expression 1, and each source data has the independent corresponding parameters VML, VMH, VSL, VSH for conversion; if there are N entries of source data, the conversion parameter table should have N groups of VML, VMH, VSL, VSH for working, there are N×4 registers in the conversion parameter table.
3. When Md =2, the linear conversion works by expression 2, and all source data share the same parameters A, B and C for conversion.
4. When Md =3, the linear conversion works by expression 2, and each source data has the independent corresponding parameters A, B, C for conversion; if there are N entries of source data, the conversion parameter table should have N groups of A, B, C for working, there are N×3 registers in the conversion parameter table.

FUN33 P LCNV	Linear Conversion (LCNV)	FUN33 P LCNV

Example program 1: Mode0 of linear conversion



Description : When M0 =1, it will perform the mode 0 operation of linear conversion, where R100 is the starting address of the source data, R1000 is the starting address of the table of the conversion parameters VML, VMH, VSL, VSH, the quantity is 6, and R2000～R2005 will store the converted results.

Ts

R1000	282	VML
R1001	3530	VMH
R1002	260	VSL
R1003	3650	VSH

S		⇨	D	
R100	282		R2000	260
R101	3530		R2001	3650
R102	1906		R2002	1955
R103	0		R2003	-34
R104	5000		R2004	5184
R105	-115		R2005	-154

FUN33 P LCNV	Linear Conversion (LCNV)	FUN33 P LCNV

Example program 2: Mode1 of linear conversion

Description : When M0 =1, it will perform the mode 1 operation of linear conversion, where R100 is the starting address of the source data, R1000 is the starting address of the table of the conversion parameters VML, VMH, VSL, VSH, the quantity is 3, and R2000～R2002 will store the converted results.

	Ts	
R1000	282	VML_0
R1001	3530	VMH_0
R1002	260	VSL_0
R1003	3650	VSH_0
R1004	-52	VML_1
R1005	1208	VMH_1
R1006	-38	VSL_1
R1007	1101	VSH_1
R1008	235	VML_2
R1009	4563	VMH_2
R1010	264	VSL_2
R1011	4588	VSH_2

	S			D
R100	282		R2000	260
R101	1208	⇨	R2001	1100
R102	2399		R2002	2426

FUN33 P LCNV	Linear Conversion （LCNV）	FUN33 P LCNV

Example program 3: Mode2 of linear conversion

Description : When M0 =1, it will perform the mode 2 operation of linear conversion, where R100 is the starting address of the source data, R1000 is the starting address of the table of the conversion parameters A, B, C, the quantity is 6, and R2000～R2005 will store the converted results.

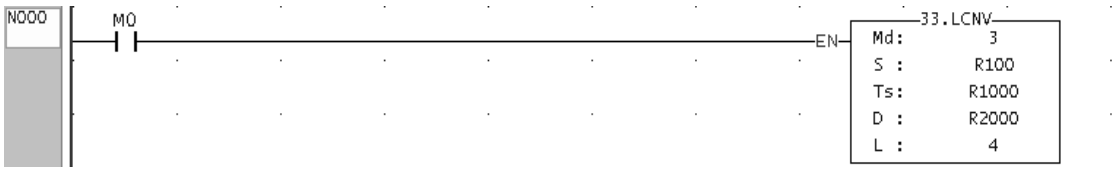
Ts

R1000	985	A
R1001	1000	B
R1002	22	C

S		⇨	D	
R100	1000		R2000	1005
R101	2345		R2001	2329
R102	3560		R2002	3526
R103	401		R2003	414
R104	568		R2004	579
R105	2680		R2005	2659

FUN33 P LCNV	Linear Conversion (LCNV)	FUN33 P LCNV

Example program 4: Mode3 of linear conversion



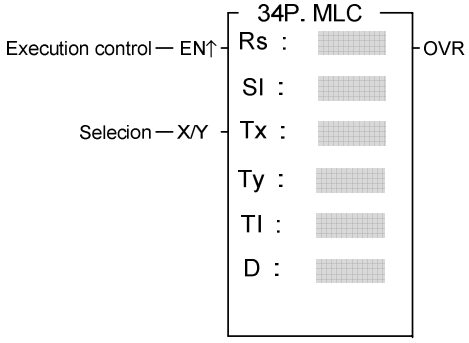
Description : When M0 =1, it will perform the mode 3 operation of linear conversion, where R100 is the starting address of the source data, R1000 is the starting address of the table of the conversion parameters A, B, C, the quantity is 4, and R2000～R2003 will store the converted results.

	Ts
R1000	5000
R1001	16380
R1002	0
R1003	10000
R1004	16383
R1005	0
R1006	2200
R1007	16380
R1008	-200
R1009	1600
R1010	16383
R1011	-100

	S			D
R100	8192		R2000	2501
R101	16383	⇨	R2001	10000
R102	8190		R2002	900
R103	0		R2003	-100

FUN34 P MLC	Multiple Linear Conversion (MLC)	FUN34 P MLC

Ladder symbol



Rs : Starting address of the source data
SI : Quantity of source data, 1~64
Tx : Starting address of X table
Ty : Starting address of Y table
TI : Quantity of table, 2~255
D : Starting address to store the result

Range / Operand	HR	IR	ROR	DR	K
	R0 | R3839	R3840 | R3903	R5000 | R8071	D0 | D4095	
Rs	○	○	○	○	
SI	○		○	○	1~64
Tx	○		○	○	
Ty	○		○*	○	
TI	○		○	○	2～255
D	○		○	○	

- When the analog input module being used for the analog measurement, the raw reading value of the analog input can be converted into the engineering range through this instruction for display or for proceeding control operation.
- For process measurement calibration, making the linear conversion for the engineering process variable, which the measurement value from the Controller's can be corrected by the value from the standard meter's through this instruction.
- When execution control "EN"=1or "EN↑" (P instruction) changes from 0 to 1, this instruction will perform the multiple linear conversion operation according to the selection of X/Y input; where Rs is the starting address of the source data, SI is the quantity of source data for conversion, Tx is the starting address of X conversion parameter table, Ty is the starting address of Y conversion parameter table, TI is the quantity of X/Y table, D is the starting address to store the converted result.
- When executing and selection X/Y=0, it will compare the source data with the entities of Tx table to find the corresponding location in Tx table (The entities in Tx table must be in ascending sequence), and then calculate the linear conversion according to the located section of Tx and Ty table;
  When executing and selection X/Y=1, it will compare the source data with the entities of Ty table to find the corresponding location in Ty table (The entities in Ty table can either be in ascending or descending sequence), and then calculate the linear conversion according to the located section of Ty and Tx table.
- When the source data is out of all entities of table, OVR=1.
- It wouldn't execute this instruction if illegal SI or TI.

FUN34 P MLC	Multiple Linear Conversion (MLC)	FUN34 P MLC

Expression:

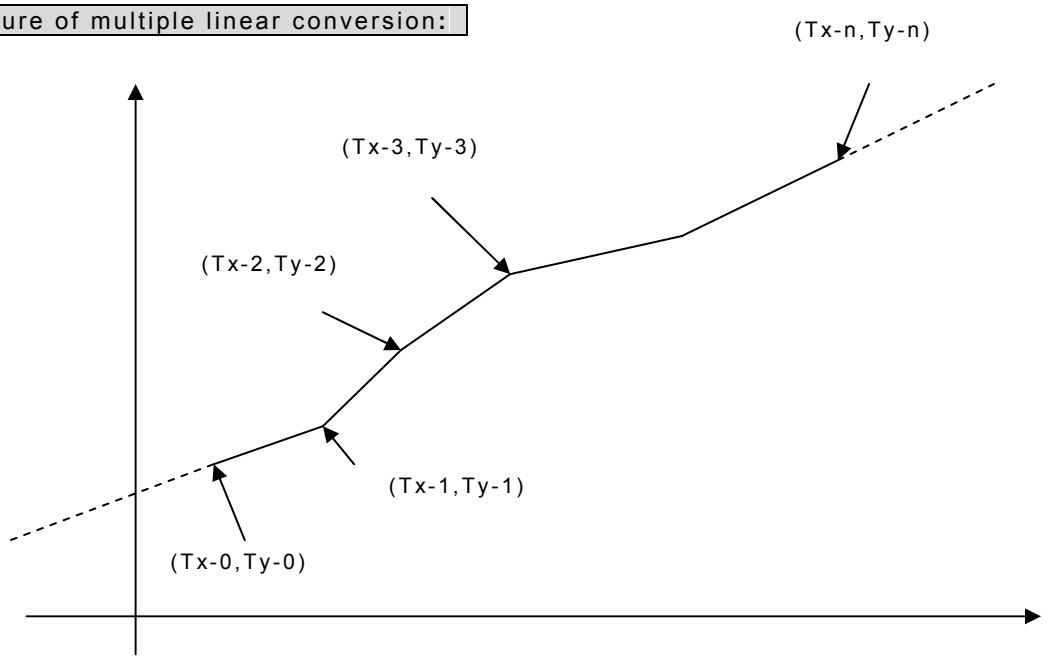
. The entities of Tx conversion parameter table must be in ascending sequence to have correct linear conversion; the entities of Ty conversion parameter table can either be in ascending or descending sequence. When executing this instruction, it will search the located section by comparing entities of the table with source data, and then calculate the linear conversion according to the following expression:

Vy = (Vx－Tx_n) × (Ty_n+1－Ty_n／Tx_n+1－Tx_n) ＋ Ty_n if X/Y=0

Vx = (Vy－Ty_n) × (Tx_n+1－Tx_n／Ty_n+1－Ty_n) ＋ Tx_n if X/Y=1

.Value of Vy、Vx, Tx_n, Tx_n+1, Ty_n, Ty_n+1 must be -32768～32767

Figure of multiple linear conversion:



FUN34 P MLC	Multiple Linear Conversion (MLC)	FUN34 P MLC

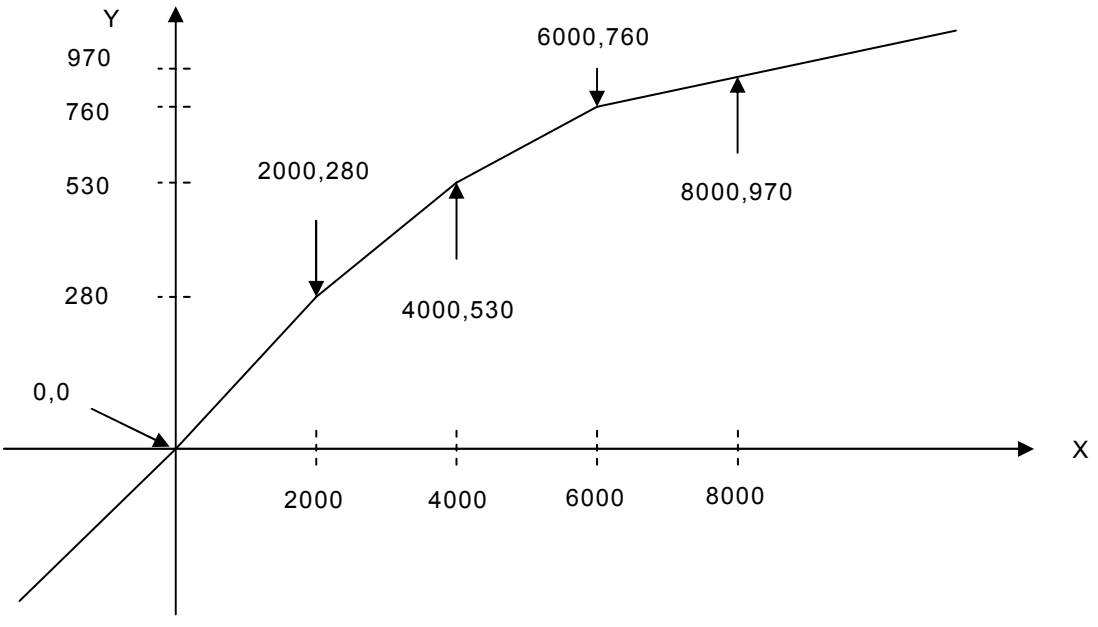
Example 1 :

Description：When M10=1, M11=0, R0 is the starting address of source data、R99 is the quantity of source data, R1000 is the starting address of Tx conversion parameter table, R2000 is the starting address of Ty conversion parameter table, R199 is the quantity of table; the source data R0~R5 will be calculated the linear conversion according to Tx and Ty table between four sections, then store the results into D0～D5.

Status Monitoring

Ref. No.	Status	Data	Ref. No.	Status	Data	Ref. No	Status	Data	Ref. No	Status	Data
R1000	Decimal	0	R2000	Decimal	0	R0	Decimal	1000	D0	Decimal	140
R1001	Decimal	2000	R2001	Decimal	280	R1	Decimal	2500	D1	Decimal	342
R1002	Decimal	4000	R2002	Decimal	530	R2	Decimal	5600	D2	Decimal	714
R1003	Decimal	6000	R2003	Decimal	760	R3	Decimal	7500	D3	Decimal	917
R1004	Decimal	8000	R2004	Decimal	970	R4	Decimal	8000	D4	Decimal	970
R199	Decimal	5				R5	Decimal	10000	D5	Decimal	1180
M10	Enable	ON	M11	Enable	OFF	R99	Decimal	6			

StatusPage0 / StatusPage01 / StatusPage2



FUN34 P MLC	Multiple Linear Conversion (MLC)	FUN34 P MLC

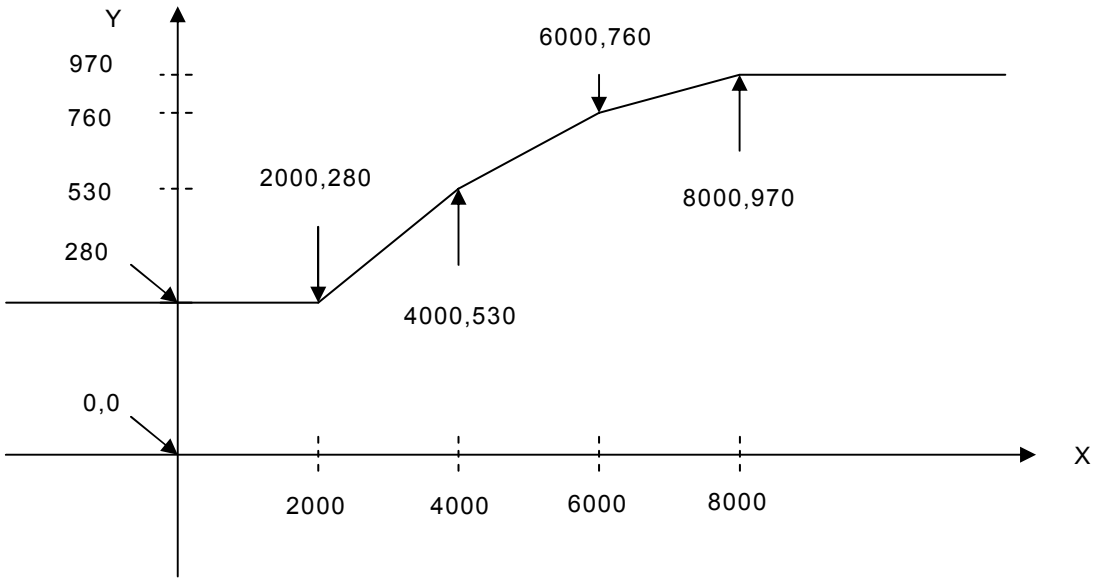
Example 2 :

Description：When M10=1, M11=0, R0 is the starting address of source data, R99 is the quantity of source data, R1000 is the starting address of Tx conversion parameter table, R2000 is the starting address of Ty conversion parameter table, R199 is the quantity of table; the source data R0~R5 will be calculated the linear conversion according to Tx and Ty table between five sections, then store the results into D0～D5.The result value is 280 if source data ≦ 2000; the result value is 970 if source data ≧ 8000.

Status Monitoring

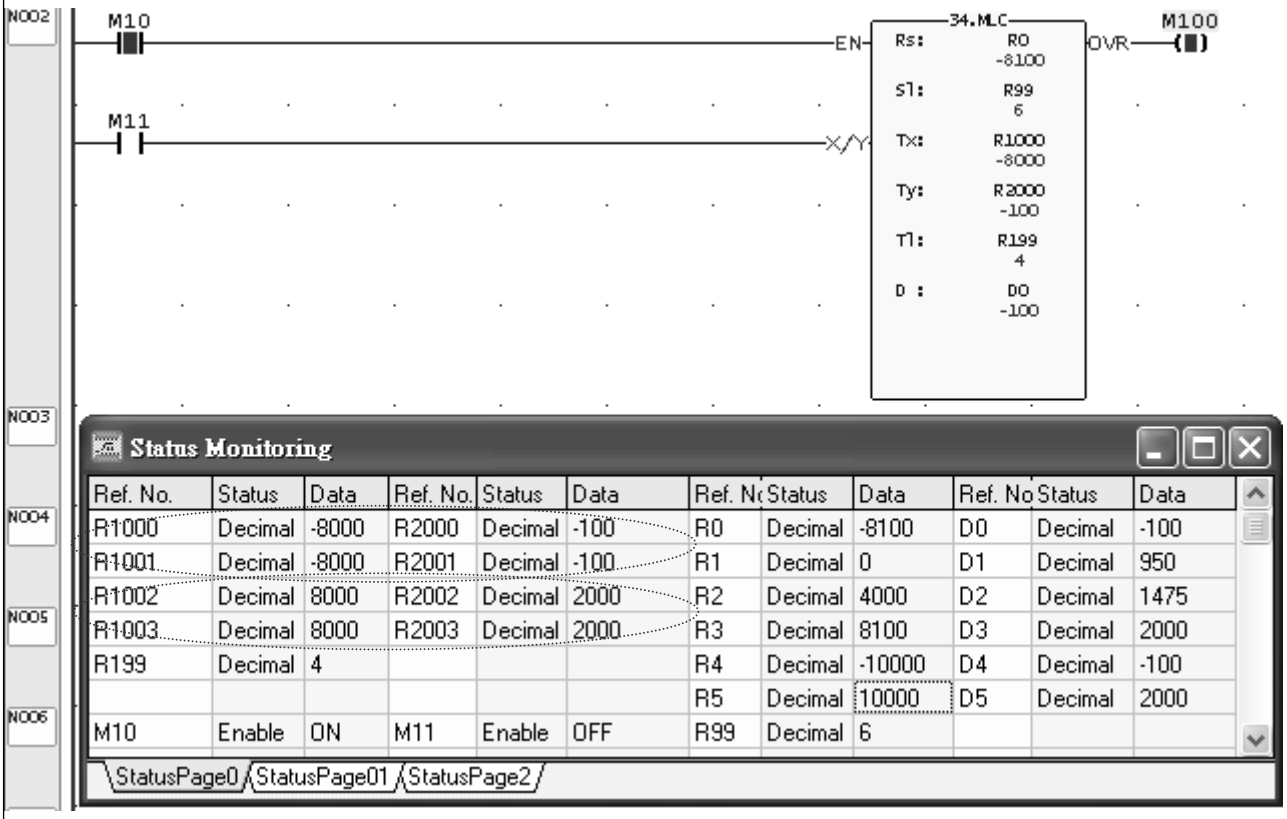
Ref. No.	Status	Data	Ref. No.	Status	Data	Ref. No	Status	Data	Ref. No	Status	Data
R1000	Decimal	2000	R2000	Decimal	280	R0	Decimal	1000	D0	Decimal	280
R1001	Decimal	2000	R2001	Decimal	280	R1	Decimal	2000	D1	Decimal	280
R1002	Decimal	4000	R2002	Decimal	530	R2	Decimal	3800	D2	Decimal	505
R1003	Decimal	6000	R2003	Decimal	760	R3	Decimal	7500	D3	Decimal	917
R1004	Decimal	8000	R2004	Decimal	970	R4	Decimal	8000	D4	Decimal	970
R1005	Decimal	8000	R2005	Decimal	970	R5	Decimal	10000	D5	Decimal	970
R199	Decimal	6	R99	Decimal	6	M10	Enable	ON	M11	Enable	OFF

StatusPage0 StatusPage01 StatusPage2



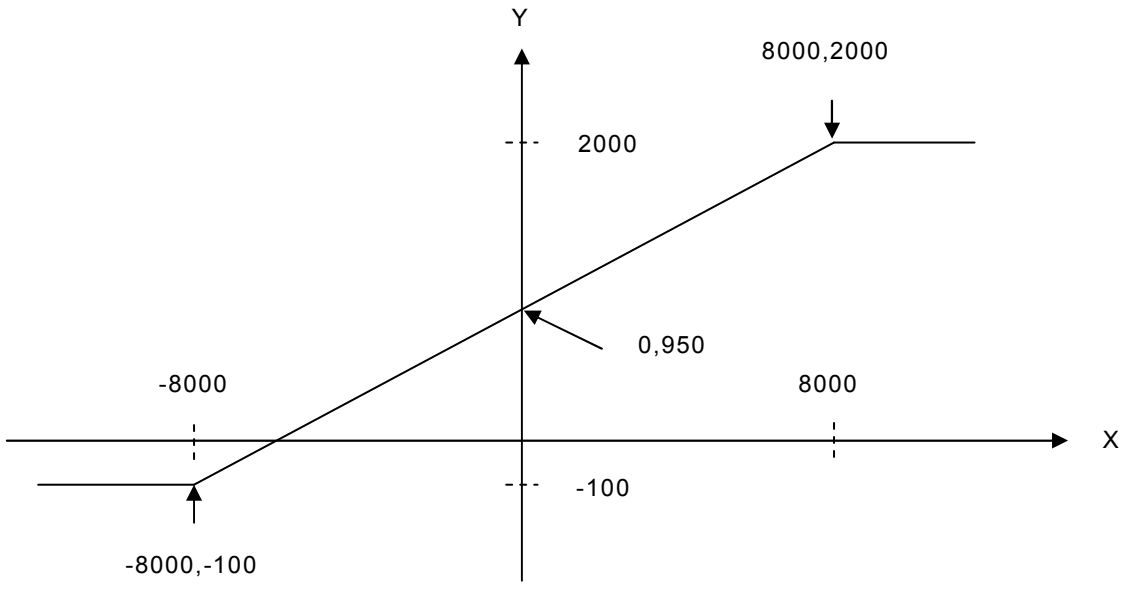
FUN34 P MLC	Multiple Linear Conversion (MLC)	FUN34 P MLC

Example 3 :



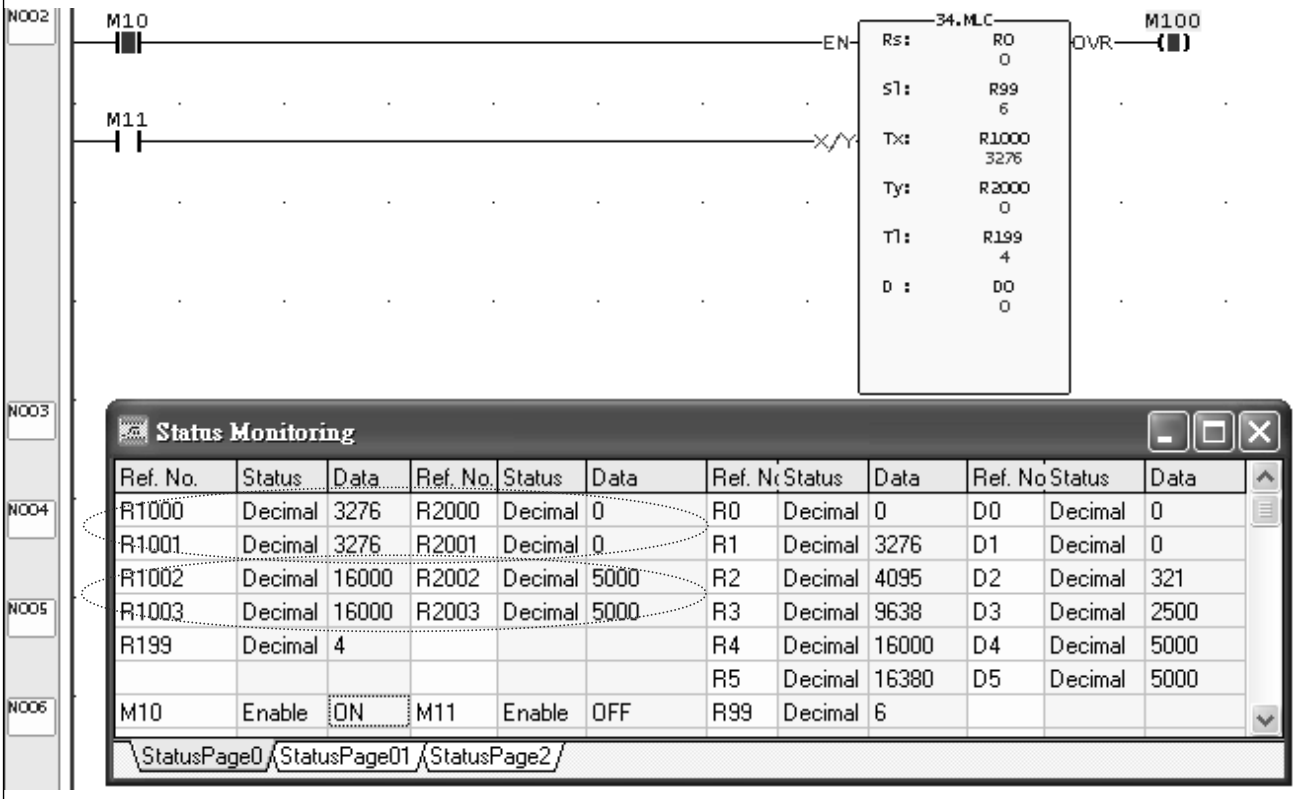
Ref. No.	Status	Data	Ref. No.	Status	Data	Ref. No	Status	Data	Ref. No	Status	Data
R1000	Decimal	-8000	R2000	Decimal	-100	R0	Decimal	-8100	D0	Decimal	-100
R1001	Decimal	-8000	R2001	Decimal	-100	R1	Decimal	0	D1	Decimal	950
R1002	Decimal	8000	R2002	Decimal	2000	R2	Decimal	4000	D2	Decimal	1475
R1003	Decimal	8000	R2003	Decimal	2000	R3	Decimal	8100	D3	Decimal	2000
R199	Decimal	4				R4	Decimal	-10000	D4	Decimal	-100
						R5	Decimal	10000	D5	Decimal	2000
M10	Enable	ON	M11	Enable	OFF	R99	Decimal	6			

Description : When M10=1, M11=0, R0 is the starting address of source data, R99 is the quantity of source data, R1000 is the starting address of Tx conversion parameter table, R2000 is the starting address of Ty conversion parameter table, R199 is the quantity of table; the source data R0~R5 will be calculated the linear conversion according to Tx and Ty table between three sections, then store the results into D0～D5.The result value is -100 if source data ≦ -8000; the result value is 2000 if source data ≧ 8000.



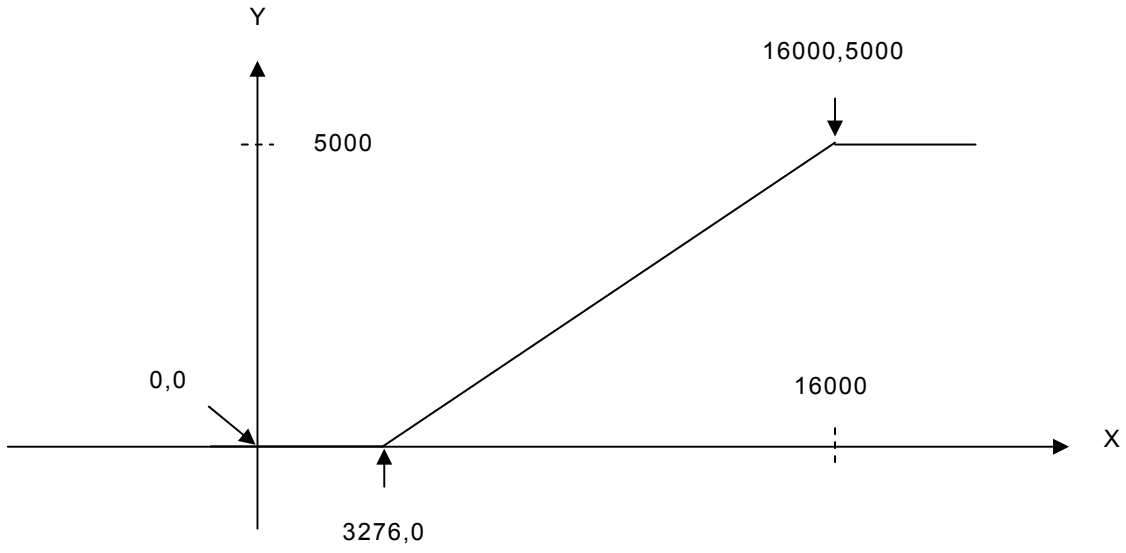
FUN34 P MLC	Multiple Linear Conversion (MLC)	FUN34 P MLC

Example 4 :



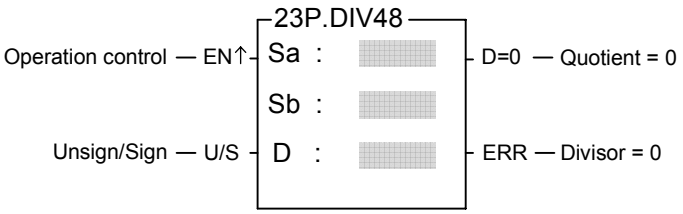
Ref. No.	Status	Data	Ref. No.	Status	Data	Ref. No	Status	Data	Ref. No	Status	Data
R1000	Decimal	3276	R2000	Decimal	0	R0	Decimal	0	D0	Decimal	0
R1001	Decimal	3276	R2001	Decimal	0	R1	Decimal	3276	D1	Decimal	0
R1002	Decimal	16000	R2002	Decimal	5000	R2	Decimal	4095	D2	Decimal	321
R1003	Decimal	16000	R2003	Decimal	5000	R3	Decimal	9638	D3	Decimal	2500
R199	Decimal	4				R4	Decimal	16000	D4	Decimal	5000
						R5	Decimal	16380	D5	Decimal	5000
M10	Enable	ON	M11	Enable	OFF	R99	Decimal	6			

Description：When M10=1, M11=0, R0 is the starting address of source data, R99 is the quantity of source data, R1000 is the starting address of Tx conversion parameter table, R2000 is the starting address of Ty conversion parameter table, R199 is the quantity of table; the source data R0~R5 will be calculated the linear conversion according to Tx and Ty table between three sections, then store the results into D0～D5.The result value is 0 if source data ≦ 3276; the result value is 5000 if source data ≧ 16000.



FUN 23 P DIV48	48-BIT DIVISION	FUN 23 P DIV48

Ladder symbol



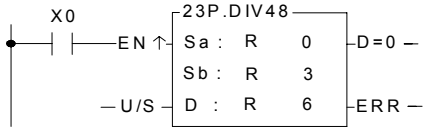
Sa：Starting register of dividend
Sb：Starting register of divisor
D : Starting register for storing the division result (quotient)
Sa, Sb, can combine V, Z, P0~P9 for index addressing.

Range / Operand	HR	OR	SR	ROR	DR	XR
	R0 | R3839	R3904 | R3967	R3968 | R4167	R5000 | R8071	D0 | D4095	V, Z P0~P9
Sa	○	○	○	○	○	○
Sb	○	○	○	○	○	○
D	○	○	○*	○*	○	○

- When “U/S”=0 and operation control “EN”=1 or “EN↑” (P instruction) changes from 0 to 1, will perform the unsigned 48 bits division operation. Dividend and divisor are each formed by three consecutive registers starting by Sa and Sb respectively. If the result is zero, ‘D=0’ output will be set to 1. If divisor is zero then the ‘ERR’ will be set to 1 and the resultant register will keep unchanged.
- When “U/S”=1 and operation control “EN”=1 or “EN↑” (P instruction) changes from 0 to 1, will perform the signed 48 bits division operation. Dividend and divisor are each formed by three consecutive registers starting by Sa and Sb respectively. If the result is zero, ‘D=0’ output will be set to 1. If divisor is zero then the ‘ERR’ will be set to 1 and the resultant register will keep unchanged.
- All operands involved in this function are all 48 bits, so Sa, Sb and D are all comprised by 3 consecutive registers.

Example: 48-bit division

In this example dividend formed by register R2, R1, R0 will be divided by divisor formed by register R5, R4, R3. The quotient will store in R8, R7, and R6.



Sa

R2	R1	R0
2147483647		

÷ Sb

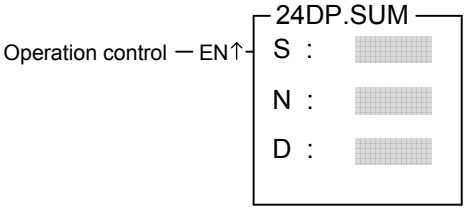
R5	R4	R3
1234567		

R8	R7	R6
1739		

Quotient

FUN 24 D P SUM	SUM (Summation of block data)	FUN 24 D P SUM

Ladder symbol



S : Starting number of source register

N : Number of registers to be summed
(successive N data units starting from S)

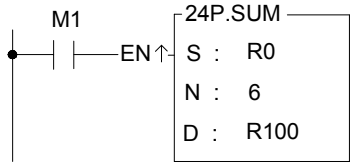
D : The register which stored the result (summation)

S, N, D, can associate with V, Z, P0~P9 index register to serve the indirect addressing application.

Range / Operand	WX	WY	WM	WS	TMR	CTR	HR	IR	OR	SR	ROR	DR	K	XR
	WX0 | WX240	WY0 | WY240	WM0 | WM1896	WS0 | WS984	T0 | T255	C0 | C255	R0 | R3839	R3840 | R3903	R3904 | R3967	R3968 | R4167	R5000 | R8071	D0 | D4095	1 | 511	V, Z P0~P9
S	○	○	○	○	○	○	○	○	○	○	○	○		○
N	○	○	○	○	○	○	○	○	○	○	○	○	○	○
D		○	○	○	○	○	○		○	○*	○*	○		○

- When operation control “EN”=1 or “EN↑” (P instruction) changes from 0 to 1, it puts the successive N units of 16 bits or 32 bits (D instruction) registers for addition calculation to get the summation, and stores the result into the register which is designated by D.
- When the value of N is 0 or greater than 511, the operation will not be performed.
- Communication port1 or port2 can be used to serve as a general purpose ASCII communication interface. If the data error detecting method is Check-Sum, this instruction can be used to generate the sum value for sending data or not use this instruction to check if the received data is error or not.

〈Example 1〉When M1 changes from OFF to ON, following instruction will calculates the summation for 16-bit data.

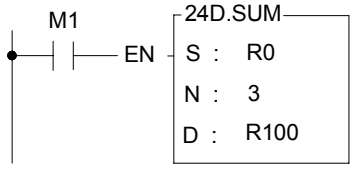


- The left illustrates that 6 16-bit registers starting from R0 is calculated for summation, and the result is stored into the R100 register.

R0=0030H
R1=0031H
R2=0032H
R3=0033H
R4=0034H
R5=0035H

➔ R100=012FH

〈Example 2〉When M1 is ON, it calculates the summation for 32-bit data.



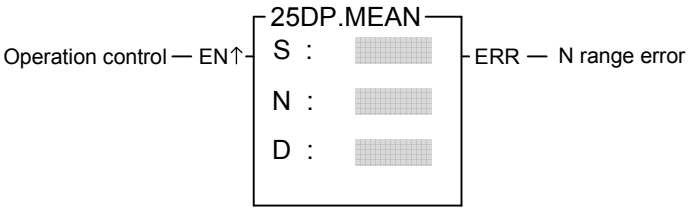
- The left illustrates that three 32-bit registers starting from DR0, is calculated for their summation, and the result is stored into the DR100 register.

R1, R0=00310030H
R3, R2=00330032H
R5, R4=00410039H

➔ R101, R100=00A5009BH

FUN 25 D P MEAN	MEAN (Average of the block data)	FUN 25 D P MEAN

Ladder symbol



S : Source register number

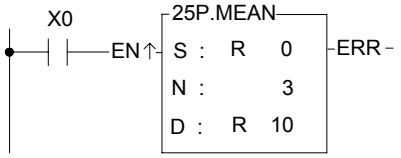
N : Number of registers to be averaged
(N units of successive registers starting from S)

D : Register number for storing result (mean value)

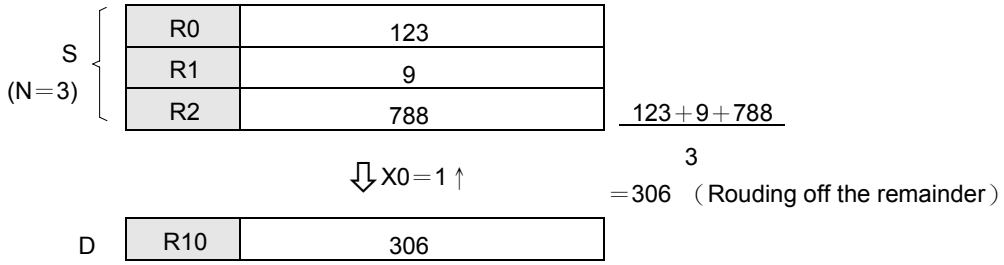
The S, N, D may combine with V, Z, P0~P9 to serve indirect address application

Range / Operand	WX	WY	WM	WS	TMR	CTR	HR	IR	OR	SR	ROR	DR	K	XR
	WX0 | WX240	WY0 | WY240	WM0 | WM1896	WS0 | WS984	T0 | T255	C0 | C255	R0 | R3839	R3840 | R3903	R3904 | R3967	R3968 | R4167	R5000 | R8071	D0 | D4095	2 | 256	V, Z P0~P9
S	○	○	○	○	○	○	○	○	○	○	○	○		○
N	○	○	○	○	○	○	○	○	○	○	○	○	○	○
D		○	○	○	○	○	○		○	○*	○*	○		○

- When operation control "EN" =1 or "EN↑" (P instruction) changes from 0 to 1, add the N successive 16-bit or 32-bit (D instruction) numerical values starting from S, and then divided by N. Store this mean value (rounding off numbers after the decimal point) in the register specified by D.
- While the N value is derived from the content of the register, if the N value is not between 2 and 256, then the N range error "ERR" will be set to 1, and do not execute the operation.

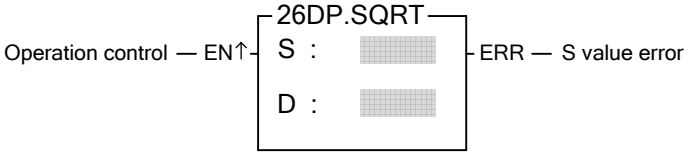


- At left, the example program gets the mean value of the 3 successive 16-bit registers starting from R0, and stores the results into the 16-bit register R10



FUN 26 D P SQRT	SQUARE ROOT	FUN 26 D P SQRT

Ladder symbol



S : Source register to be taken square root

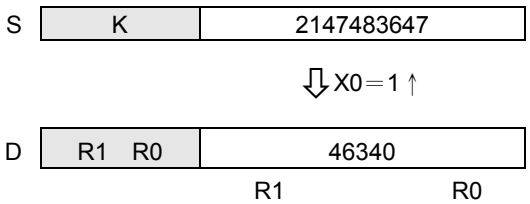
D : Register for storing result (square root value)

S, D may combine with V, Z, P0~P9 to serve indirect address application

Range / Operand	WX	WY	WM	WS	TMR	CTR	HR	IR	OR	SR	ROR	DR	K	XR
	WX0 | WX240	WY0 | WY240	WM0 | WM1896	WS0 | WS984	T0 | T255	C0 | C255	R0 | R3839	R3840 | R3903	R3904 | R3967	R3968 | R4167	R5000 | R8071	D0 | D4095	16-bit + number	V, Z P0~P9
S	○	○	○	○	○	○	○	○	○	○	○	○	○	○
D		○	○	○	○	○	○		○	○*	○*	○		○

- When operation control "EN" =1 or "EN↑" (P instruction) changes from 0 to 1, take the square root (rounding off numbers after the decimal point) of the data specified by the S field, and store the result into the register specified by D.
- While the S value is derived from the content of the register, if the value is negative, then the S value error flag "ERR" will be set to 1, and do not execute the operation.

- The instruction at left calculates the square root of the constant 2147483647, and stores the result in R0.



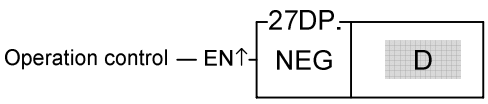
$$\sqrt{2147483647} = 46340.\underline{\underline{95}}$$

↑

Rounding off

FUN 27 D P NEG	NEGATION (Take the negative value)	FUN 27 D P NEG

Ladder symbol

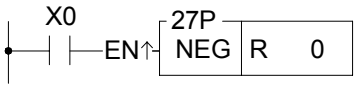


D : Register to be negated

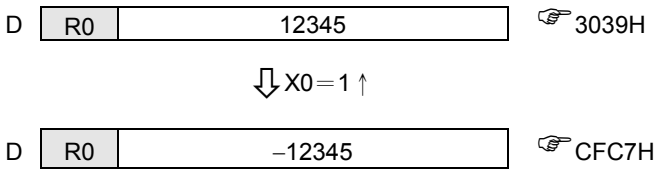
D may combine with V, Z, P0~P9 to serve indirect address application

Operand \ Range	WY	WM	WS	TMR	CTR	HR	OR	SR	ROR	DR	XR
	WY0 | WY240	WM0 | WM1896	WS0 | WS984	T0 | T255	C0 | C255	R0 | R3839	R3904 | R3967	R3968 | R4167	R5000 | R8071	D0 | D4095	V, Z P0~P9
D	○	○	○	○	○	○	○	○*	○*	○	○

- When operation control "EN" =1 or "EN↑" (P instruction) changes from 0 to 1, negate (ie. calculate 2's complement) the value of the content of the register specified by D, and store it back in the original D register.
- If the value of the content of D is negative, then the negation operation will make it positive.

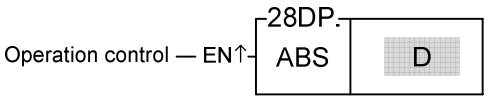


- The instruction at left negates the value of the R0 register, and stores it back to R0.



FUN 28 D P ABS	ABSOLUTE (Take the absolute value)	FUN 28 D P ABS

Ladder symbol



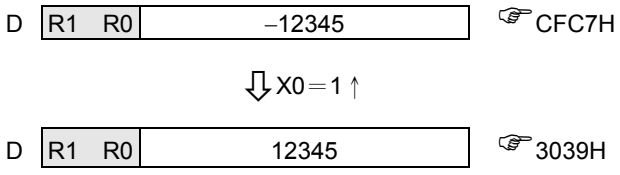
D : Register to be taken absolute value

D may combine with V, Z, P0~P9 to serve indirect address application

Operand \ Range	WY	WM	WS	TMR	CTR	HR	OR	SR	ROR	DR	XR
	WY0 | WY240	WM0 | WM1896	WS0 | WS984	T0 | T255	C0 | C255	R0 | R3839	R3904 | R3967	R3968 | R4167	R5000 | R8071	D0 | D4095	V, Z P0~P9
D	○	○	○	○	○	○	○	○*	○*	○	○

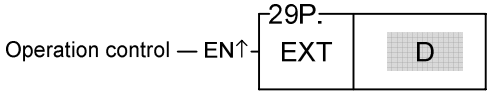
- When operation control "EN" =1 or "EN↑" (P instruction) changes from 0 to 1, calculate the absolute value of the content of the register specified by D, and write it back into the original D register.

- The instruction at left calculates the absolute value of the R0 register, and stores it back in R0.



FUN 29 P EXT	SIGN EXTENSION	FUN 29 P EXT

Ladder symbol



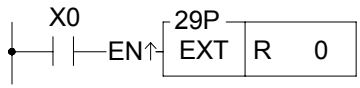
D : Register to be taken sign extension

D may combine with V, Z, P0~P9 to serve indirect address application

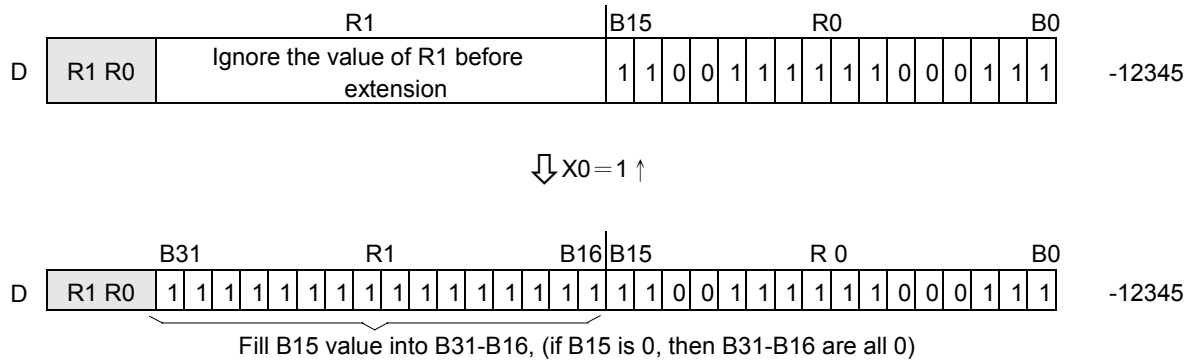
Range / Operand	WY	WM	WS	TMR	CTR	HR	OR	SR	ROR	DR	XR
	WY0 | WY240	WM0 | WM1896	WS0 | WS984	T0 | T255	C0 | C255	R0 | R3839	R3904 | R3967	R3968 | R4167	R5000 | R8071	D0 | D4095	V, Z P0~P9
D	○	○	○	○	○	○	○	○*	○*	○	○

- When operation control "EN" =1 or "EN↑" (P instruction) changes from 0 to 1, this instruction will sign extension the 16 bits numerical value specified by D to 32-bit value and store it into the 32-bit register comprised by the two successive words, D + 1 and D. (Both values are the same, only it was originally formated as a 16 bits numerical value, and was then extended to be formated as a 32 bits numerical value.)

- This instruction extent the numerical value of a 16-bit register into an equivalent numerical value in a 32-bit register (for example 33FFH converts to 000033FFH), Its main function is for numerical operations (+,-,*,/,CMP.....) which can take the 16 bits or 32 bits numerical values as operand. Before operation all the operand should be adjusted to the same length for proper operation.



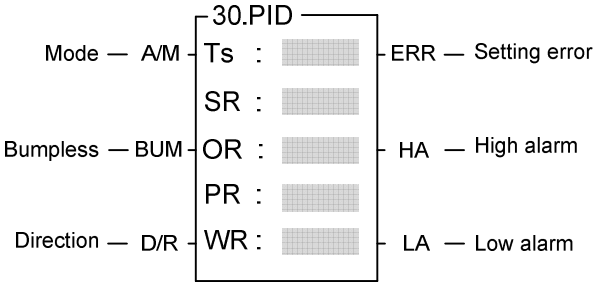
- The instruction at left takes a 16 bits numerical value R0, and extends it to an equivalent value in 32 bits, then stores it into a 32 bits register (DR0=R1R0) comprised R0 and R1.



Before extension（16 bits） R0= CFC7H=-12345
After extension（32 bits）R1R0=FFFFCFC7H=-12345
} The two numerical values are actually the same.

FUN 30 PID	GENERAL PURPOSE PID OPERATION (Brief description)	FUN 30 PID

Ladder symbol



Ts : PID Operation time interval

SR : Starting register of process control parameter table comprised by 8 consecutive registers.

OR : PID output register

PR : Starting register of the process parameter table comprised by 7 consecutive registers.

WR : Starting register of working variable for PID internal operation. It requires 7 registers and can't be re-used in other part of the ladder program.

Range / Ope-rand	HR R0 | R3839	ROR R5000 | R8071	DR D0 | D4095	K
Ts	○	○	○	1～3000
SR	○	○*	○	
OR	○	○*	○	
PR	○	○*	○	
WR	○	○*	○	

- PID function according to the current value of process variable (PV) derived from the external analog signal and the setting value (SP) of process performs the calculation, which base on the PID formula. The result of calculation is the control output for the controlled process, which can feed directly to the AO module or other output interface or leaved for further process. The usage of PID control for process if properly can achieve a fast and smooth result of PV tracking toward SP change or be immune to the disturbance of process.

- The PID formula in digital form:

$$Mn = [(D4005/Pb)\ xEn] + \sum_{0}^{n} [(D4005/Pb) \times Ti \times Ts \times En] - [(D4005/Pb) \times Td \times (PVn - PVn\text{-}1)/Ts] + Bias$$

Mn : Control output at time"n"

D4005 : The gain constant, the default is 1000, it's range is 1～5000.

Pb : Proportional band (range: 2~5000, unit 0.1%. Kc (gain) =1000/ Pb)

Ti : Integral time constant (range: 0~9999 corresponds to 0.00~99.99 Repeats/Minute)

Td : Differential time constant (range: 0~9999 corresponds to 0.00~99.99 Minutes)

PVn : Process value at time"n"

PV n-1 : Process value at time"n-1"

En : Error at time"n" =set value (SP) – process value at time"n" (PVn)

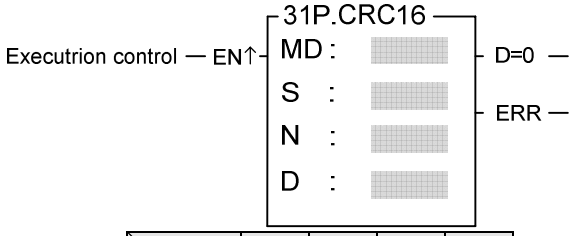
Ts : Interval time of PID calculation (range: 1~3000, unit: 0.01s)

Bias : Control output offset (range: 0~16380)

- For detail description of this function, please refer chapter 18.

FUN31 P CRC16	CRC16 CALCULATION （CRC16）	FUN31 P CRC16

Ladder symbol



Operand \ Range	HR	ROR	DR	K
	R0 | R3839	R5000 | R8071	D0 | D4095	
MD				0~1
S	○	○	○	
N	○	○	○	1~256
D	○	○*	○	

MD：0, Not including higher byte of registers to be calculated the CRC16

：1, Including higher byte of registers to be calculated the CRC16

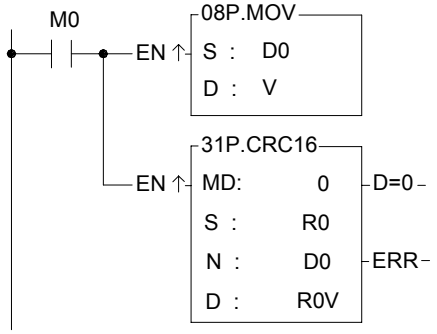
S：Starting address of CRC16 calculation

N：Length of CRC16 calculation (In Byte)

D：The destination register to store the calculation of CRC16,
Register D stores the Upper Byte of CRC16
Register D＋1 stores the Lower Byte of CRC16

S, N, D may associate with V, Z, P0~P9 index register to serve the indirect addressing application.

- When execution control "EN"=1 or "EN↑" (P instruction) changes from 0 to 1, it will start the CRC16 calculation from the lower byte of S and by the length of N, the result of calculation will be stored into register D and D+1.
- The output indication "D=0" will be ON if the value of calculation is 0.
- It will not execute the calculation and the output indication "ERR" will be ON if the length is invalid.
- When communicating with the intelligent peripheral in binary data format, the CRC16 error detection is used very often; the well known Modbus RTU communication protocol uses this method for error detection of message frame.
- CRC16 is the check value of a Cyclical Redundancy Check calculation performed on the message contents.
- Perform the CRC16 calculation on the received message data and error check value, the result of the calculation value must be 0, it means no error within this message frame.



Description：When M0 changes from 0 to 1, it will execute the CRC16 calculation starting from lower byte of R0, the length is assigned by D0, and then stores the CRC value into register R0+V and R0+V+1.

It is supposed D0=10, the registers R10 and R11 will store the CRC16 value.

S

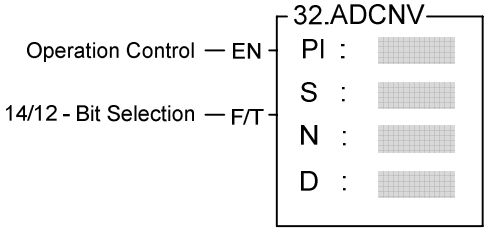
	High Byte	Low Byte
R0	Don't care	Byte-0
R1	Don't care	Byte-1
R2	Don't care	Byte-2
R3	Don't care	Byte-3
R4	Don't care	Byte-4
R5	Don't care	Byte-5
R6	Don't care	Byte-6
R7	Don't care	Byte-7
R8	Don't care	Byte-8
R9	Don't care	Byte-9

D

	High Byte	Low Byte
R10	00	CRC-Hi
R11	00	CRC-Lo

FUN32 ADCNV	CONVERTING THE RAW VALUE OF 4～20MA ANALOG INPUT （ADCNV）	FUN32 ADCNV

Ladder symbol



Range / Ope-rand	HR	IR	ROR	DR	K
	R0 | R3839	R3840 | R3903	R5000 | R8071	D0 | D4095	
PI					0~1
S	○	○	○	○	
N	○		○	○	1~64
D	○		○*	○	

PI：0, the polarity setting of analog input module is at unipolar position

：1, the polarity setting of analog input module is at bipolar position

S：Starting address of source registers

N：Quantity of conversion (In Word)

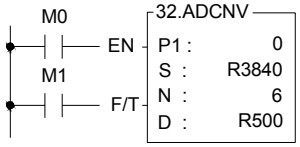
D：Starting address of destination registers

S, N, D may associate with V, Z, P0~P9 index register to serve the indirect addressing application.

- When the analog input is 4～20mA, the analog input module is one of the solution to get this kind of signal, but the input span of the analog input module is 0～20mA (Setting at 10V, Unipolar), however there will exist the offset of the raw reading value; this instruction is applied to eliminate the offset and convert the raw reading value into the range of 0～4095(12-bit) or 0～16383(14-bit), it is more convenient for following operation.
- When execution control "EN" =1, it will execute the conversion starting from S, length by N, and then store the results into the D registers.
- This instruction will not act if invalid length of N.
- When the input "F/T" =0, it assigns the 12-bit analog input module; while "F/T" =1, it assigns the 14-bit analog input module.
- The reading value of the analog input must be in -2048 ~ 2047 or -8192 ~ 8191 formats that the conversion will have the correct correspondence. Otherwise, if the reading value is in 0~4095 or 0~16383 format that the conversion will have the wrong correspondence.

FUN32 ADCNV	CONVERTING THE RAW VALUE OF 4～20MA ANALOG INPUT （ADCNV）	FUN32 ADCNV

Example：



Description：When M0 is ON and M1 is OFF, it will perform 6 points of conversion starting from R3840, where the offset of 4～20mA raw reading value will be eliminated, and the corresponding value 0～4095 will be stored into R500～R505.

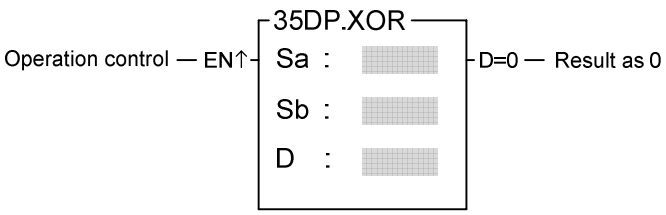
	S		D	
R3840	－1229	R500	0	(4 mA)
R3841	409	R501	2047	(12 mA)
R3842	2047	R502	4095	(20 mA)
R3843	－2048	R503	0	(0 mA)
R3844	－2048	R504	0	(0 mA)
R3845	－2048	R505	0	(0 mA)

When M0 is ON and M1 is ON, it will perform 6 points of conversion starting from R3840, where the offset of 4~20mA raw reading value will be eliminated, and the corresponding value 0~16383 will be stored into R500~R505.

	S		D	
R3840	－4916	R500	0	(4 mA)
R3841	1637	R501	8191	(12 mA)
R3842	8191	R502	16383	(20 mA)
R3843	－8192	R503	0	(0 mA)
R3844	－8192	R504	0	(0 mA)
R3845	－8192	R505	0	(0 mA)

FUN 35 D P XOR	EXCLUSIVE OR	FUN 35 D P XOR

Ladder symbol



Sa : Source data a for exclusive or operation

Sb :Source data b for exclusive or operation

D : Register storing XOR results

Sa, Sb, D may combine with V, Z, P0~P9 to serve indirect address application

Range / Operand	WX	WY	WM	WS	TMR	CTR	HR	IR	OR	SR	ROR	DR	K	XR
	WX0 | WX240	WY0 | WY240	WM0 | WM1896	WS0 | WS984	T0 | T255	C0 | C255	R0 | R3839	R3840 | R3903	R3904 | R3967	R3968 | R4167	R5000 | R8071	D0 | D4095	16/32-bit +/- number	V, Z P0~P9
Sa	○	○	○	○	○	○	○	○	○	○	○	○	○	○
Sb	○	○	○	○	○	○	○	○	○	○	○	○	○	○
D		○	○	○	○	○	○		○	○*	○*	○		○

- When operation control "EN" =1 or "EN↑" (P instruction) changes from 0 to 1, will perform the logical XOR (exclusive or) operation of data Sa and Sb. The operation of this function is to compare the corresponding bits of Sa and Sb (B0~B15 or B0~B31), and if bits at the same position have different status, then set the corresponding bit within D as 1, otherwise as 0.
- After the operation, if all the bits in D are all 0, then set the 0 flag "D =0" to 1.

- The instruction at left makes a logical XOR operation using the R0 and R1 registers, and stores the result in R2.

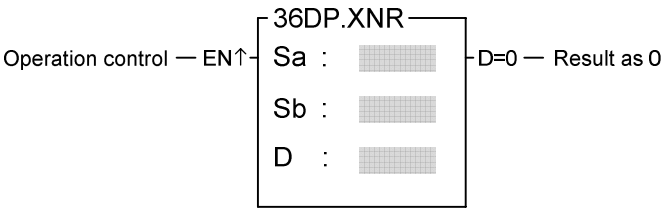
Sa	R0	1	0	1	1	1	0	1	1	0	1	1	0	1	1	0	1
Sb	R1	1	1	1	0	1	1	1	0	1	0	1	0	0	1	1	0

⇩X0＝1↑

D	R2	0	1	0	1	0	1	0	1	1	1	0	0	1	0	1	1

FUN 36 D P XNR	EXCLUSIVE NOR	FUN 36 D P XNR

Ladder symbol



Sa : Data a for XNR operation

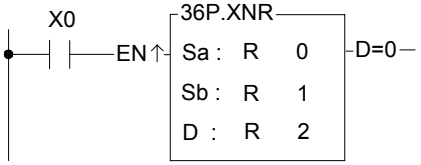
Sb : Data b for XNR operation

D : Register storing XNR results

Sa, Sb, D may combine with V, Z, P0~P9 to serve indirect address application

Range / Operand	WX	WY	WM	WS	TMR	CTR	HR	IR	OR	SR	ROR	DR	K	XR
	WX0 | WX240	WY0 | WY240	WM0 | WM1896	WS0 | WS984	T0 | T255	C0 | C255	R0 | R3839	R3840 | R3903	R3904 | R3967	R3968 | R4167	R5000 | R8071	D0 | D4095	16/32-bit ± number	V, Z P0~P9
Sa	○	○	○	○	○	○	○	○	○	○	○	○	○	○
Sb	○	○	○	○	○	○	○	○	○	○	○	○	○	○
D		○	○	○	○	○	○		○	○*	○*	○		○

- When operation control "EN" =1 or "EN↑" (P instruction) changes from 0 to 1, will perform the logical XNR (inclusive or) operation of data Sa and Sb. The operation of this function is to compare the corresponding bits of Sa and Sb (B0~B15 or B1~B31), and if the bit has the same value, then set the corresponding bit within D as 1. If not then set it to 0.
- After the operation, if the bits in D are all 0, then set the 0 flag "D=0" to 1.



- The instruction at left makes a logical XNR operation of the R0 and R1 registers, and the results are stored in the R2 register.

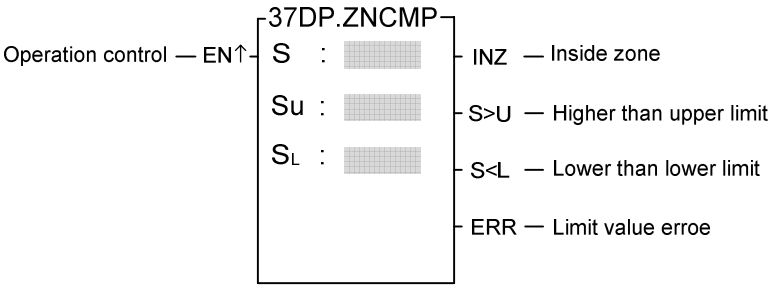
Sa	R0	1	0	1	1	1	0	1	1	0	1	1	0	1	1	0	1
Sb	R1	1	1	1	0	1	1	1	0	1	0	1	0	0	1	1	0

⇩X0＝1↑

D	R2	1	0	1	0	1	0	1	0	0	0	1	1	0	1	0	0

FUN 37 D P ZNCMP	ZONE COMPARE	FUN 37 D P ZNCMP

Ladder symbol



S : Register for zone comparison

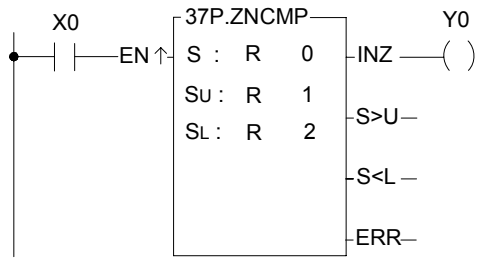
SU : The upper limit value

SL : The lower limit value

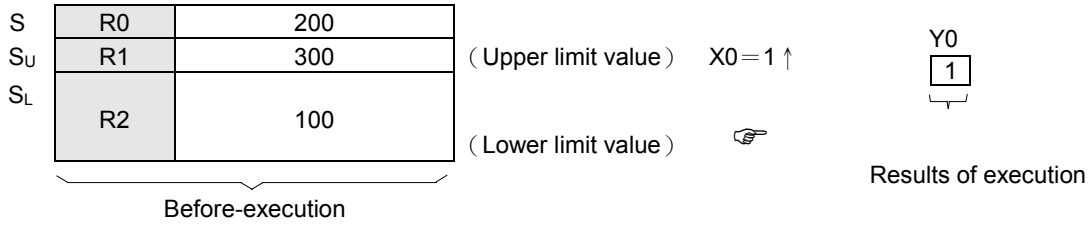
S, SU, SL may combine with V, Z, P0~P9 to serve indirect address application

Range / Operand	WX	WY	WM	WS	TMR	CTR	HR	IR	OR	SR	ROR	DR	K	XR
	WX0 | WX240	WY0 | WY240	WM0 | WM1896	WS0 | WS984	T0 | T255	C0 | C255	R0 | R3839	R3840 | R3903	R3904 | R3967	R3968 | R4167	R5000 | R8071	D0 | D4095	16/32-bit +/- number	V, Z P0~P9
S	○	○	○	○	○	○	○	○	○	○	○	○		○
SU	○	○	○	○	○	○	○	○	○	○	○	○	○	○
SL	○	○	○	○	○	○	○	○	○	○	○	○	○	○

- When operation control "EN" =1 or "EN↑" (P instruction) changes from 0 to 1, compares S with upper limit SU and lower limit SL. If S is between the upper limit and the lower limit ($S_L \leqq S \leqq S_U$), then set the inside zone flag "INZ" to 1. If the value of S is greater than the upper limit S_U, then set the higher than upper limit flag "S>U" to 1. If the value of S is smaller then the lower limit S_L, then set the lower than lower limit flag "S<L" as 1.
- The upper limit S_U should be greater than the lower limit S_L. If $S_U < S_L$, then the limit value error flag "ERR" will set to 1, and this instruction will not carry out.

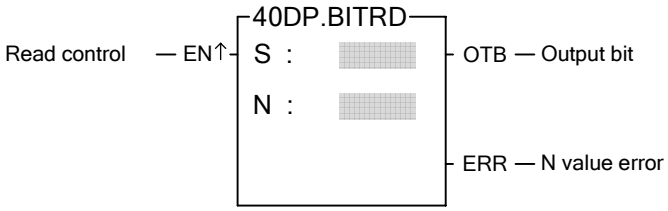


- The instruction at left compares the value of R0 with the upper and lower limit zones formed by R1 and R2. If the values of R0~R2 are as shown in the diagram at bottom left, then the result can then be obtained as at the right of this diagram.
- If want to get the status of out side the zone, then OUT NOT Y0 may be used, or an OR operation between the two outputs S>U and S<L may be carried out, and move the result to Y0.



FUN 40 D P BITRD	BIT READ	FUN 40 D P BITRD

Ladder symbol



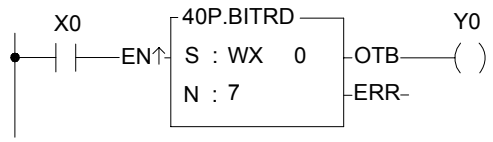
S : Source data to be read

N : The bit number of the S data to be read out.

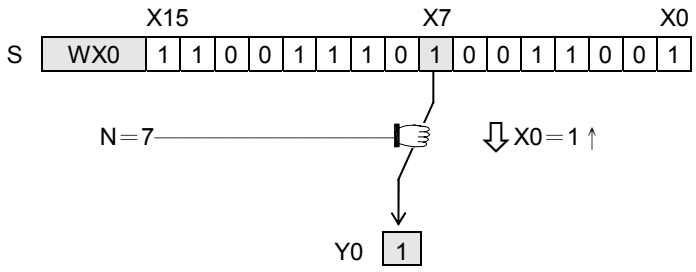
S, N may combine with V, Z, P0~P9 to serve indirect address application.

Operand \ Range	WX	WY	WM	WS	TMR	CTR	HR	IR	OR	SR	ROR	DR	K	XR
	WX0 | WX240	WY0 | WY240	WM0 | WM1896	WS0 | WS984	T0 | T255	C0 | C255	R0 | R3839	R3840 | R3903	R3904 | R3967	R3968 | R4167	R5000 | R8071	D0 | D4095	16/32-bit +/- number	V, Z P0~P9
S	○	○	○	○	○	○	○	○	○	○	○	○	○	○
N	○	○	○	○	○	○	○	○	○	○	○	○	0～31	○

- When read control "EN" =1 or "EN↑" (P instruction) changes from 0 to 1, take the Nth bit of the S data out , and put it to the output bit "OTB".
- When read control "EN" =0 or "EN↑" (P instruction) is not change from 0 to 1, The output “OTB” can be selected to keep at the last state(if M1919=0) or set to zero (if M1919=1).
- When the operand is 16 bits, the effective range for N is 0~15. For 32 bits operand (D instruction) it is 0~31. N beyond this range will set the N value error flag "ERR" to 1, and do not carry out this instruction.

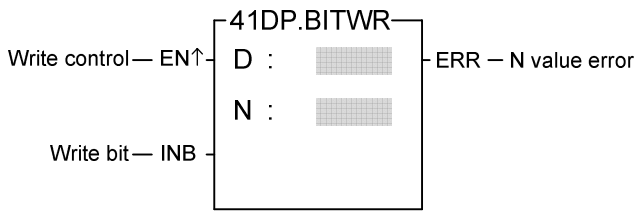


- The instruction at left reads the 7th bit (X7) status from WX0 (X0~X15) and output to Y0. The results are as follows:



FUN 41 D P BITWR	BIT WRITE	FUN 41 D P BITWR

Ladder symbol



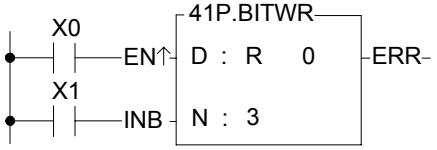
D : Register for bit write

N : The bit number of the D register to be written.

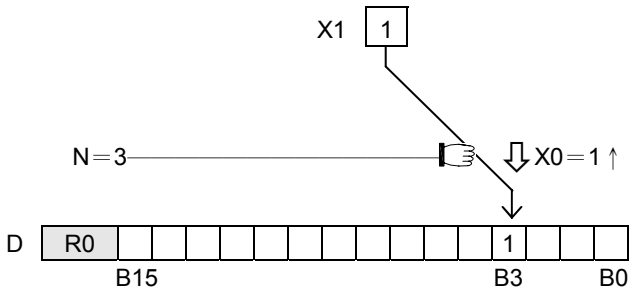
D, N may combine with V, Z , P0~P9 to serve indirect address application.

Range / Operand	WX	WY	WM	WS	TMR	CTR	HR	IR	OR	SR	ROR	DR	K	XR
	WX0 | WX240	WY0 | WY240	WM0 | WM1896	WS0 | WS984	T0 | T255	C0 | C255	R0 | R3839	R3840 | R3903	R3904 | R3967	R3968 | R4167	R5000 | R8071	D0 | D4095	0 or 15 | 0 or 31	V, Z | P0~P9
D		○	○	○	○	○	○		○	○*	○*	○		○
N	○	○	○	○	○	○	○	○	○	○	○	○	○	○

- When write control "EN" =1 or "EN↑" (P instruction) changes from 0 to 1, will write the write bit (INB) into the Nth bit of register D.
- When the operand is 16 bits, the effective range of N is 0~15. For 32 bits (D instruction) operand it is 0~31. N beyond this range, will set the N value error flag "ERR" to 1, and do not carry out this instruction.



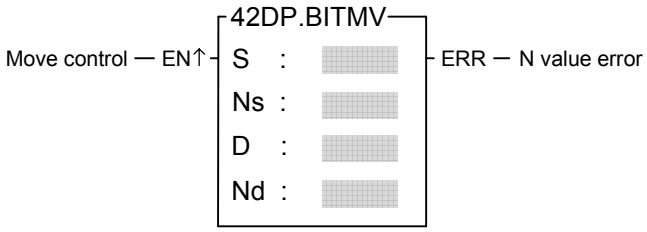
- The instruction at left writes the status of the write bit INB into B3 of R0. Assuming X1 = 1, the result will be as follows:



Bits other than B3 remain unchanged

FUN 42 D P BITMV	BIT MOVE	FUN 42 D P BITMV

Ladder symbol



S : Source data to be moved

Ns : Assign Ns bit within S as source bit

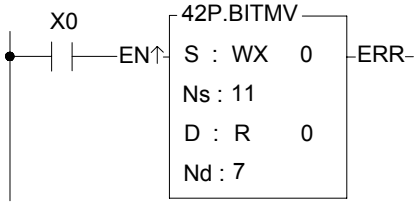
D : Destination register to be moved

Nd : Assign Nd bit within D as target bit

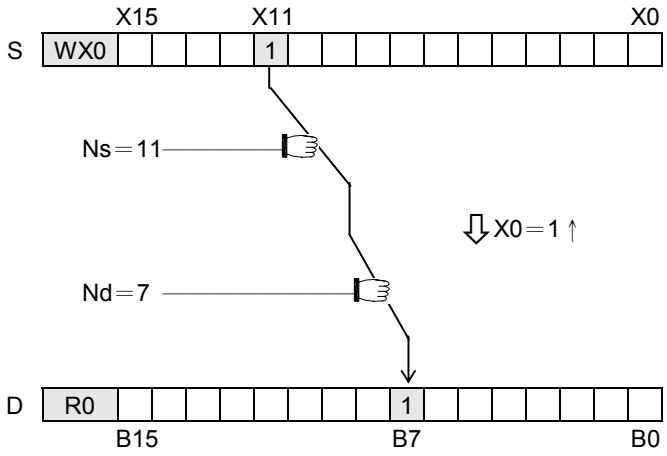
S, Ns, D, Nd may combine with V, Z, P0~P9 to serve indirect address application.

Range / Operand	WX	WY	WM	WS	TMR	CTR	HR	IR	OR	SR	ROR	DR	K	XR
	WX0 | WX240	WY0 | WY240	WM0 | WM1896	WS0 | WS984	T0 | T255	C0 | C255	R0 | R3839	R3840 | R3903	R3904 | R3967	R3968 | R4167	R5000 | R8071	D0 | D4095	16/32-bit +/- number	V, Z P0~P9
S	○	○	○	○	○	○	○	○	○	○	○	○	○	○
Ns	○	○	○	○	○	○	○	○	○	○	○	○	0～31	○
D		○	○	○	○	○	○		○	○*	○*	○		○
Nd	○	○	○	○	○	○	○	○	○	○	○	○	0～31	○

- When move control "EN" =1 or "EN↑" (P instruction) changes from 0 to 1, will move the bit status specified by Ns within S into the bit specified by Nd within D.
- When the operand is 16 bits, the effective range of N is 0~15. For 32 bits (D instruction) operand the effective range is 0~31. N beyond this range will set the N value error flag "ERR" to 1, and do not carry out this instruction.

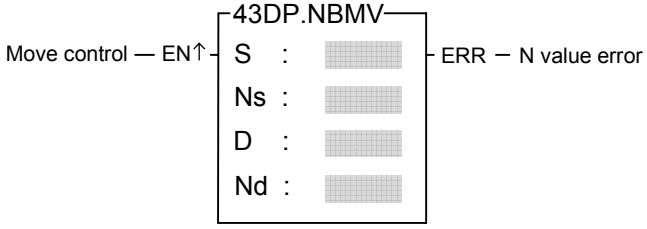


- The instruction at left moves the status of B11 (X11) within S into the B7 position within D. Except bit B7, other bits within D does not change.



FUN 43 D P NBMV	NIBBLE MOVE	FUN 43 D P NBMV

Ladder symbol



S : Source data to be moved

Ns: Assign Ns nibble within S as source nibble

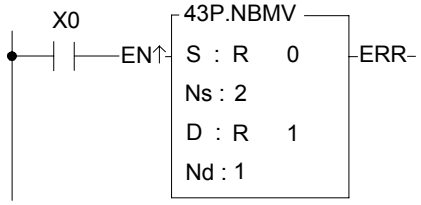
D : Destination register to be moved

Nd: Assign Nd nibble within D as target nibble

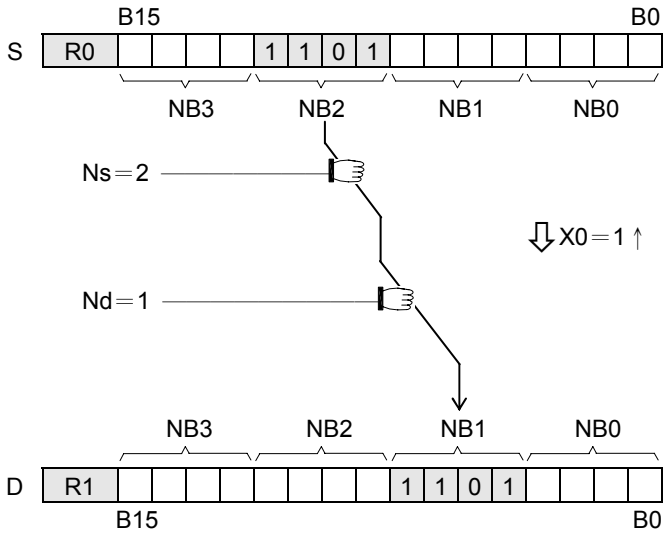
S, Ns, D, Nd may combine with V, Z, P0~P9 to serve indirect address application.

Range / Operand	WX	WY	WM	WS	TMR	CTR	HR	IR	OR	SR	ROR	DR	K	XR
	WX0 | WX240	WY0 | WY240	WM0 | WM1896	WS0 | WS984	T0 | T255	C0 | C255	R0 | R3839	R3840 | R3903	R3904 | R3967	R3968 | R4167	R5000 | R8071	D0 | D4095	16/32-bit +/- number	V, Z P0~P9
S	○	○	○	○	○	○	○	○	○	○	○	○	○	○
Ns	○	○	○	○	○	○	○	○	○	○	○	○	0～7	○
D		○	○	○	○	○	○		○	○*	○*	○		○
Nd	○	○	○	○	○	○	○	○	○	○	○	○	0～7	○

- When move control "EN" =1 or "EN↑" (P instruction) changes from 0 to 1, will move the Ns'th nibble from within S to the nibble specified by Nd within D. (A nibble is comprised by 4 bits. Starting from the lowest bit of the register, B0, each successive 4 bits form a nibble, so B0~B3 form nibble 0, B4~B7 form nibble 1, etc...)
- When the operand is 16 bits, the effective range of Ns or Nd is 0~3. For 32 bits (D instruction) operand the range is 0~7. Beyond this range, will set the N value error flag "ERR" to 1, and do not carry out this instruction.

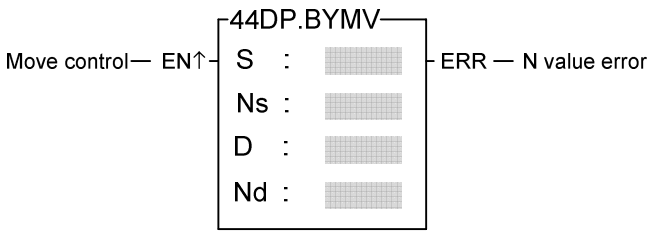


- The instruction at left moves the third nibble NB2 (B8~B11) within S to the first nibble NB1 (B4~B7) within D. Other nibbles within D remain unchanged.



FUN 44 D P BYMV	BYTE MOVE	FUN 44 D P BYMV

Ladder symbol



S : Source data to be moved

Ns : Assign Ns byte within S as source byte

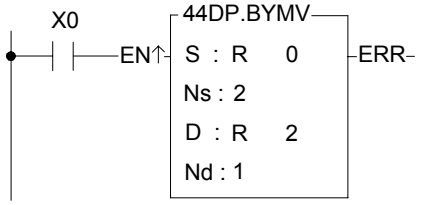
D : Destination register to be moved

Nd : Assign Nd byte within D as target byte

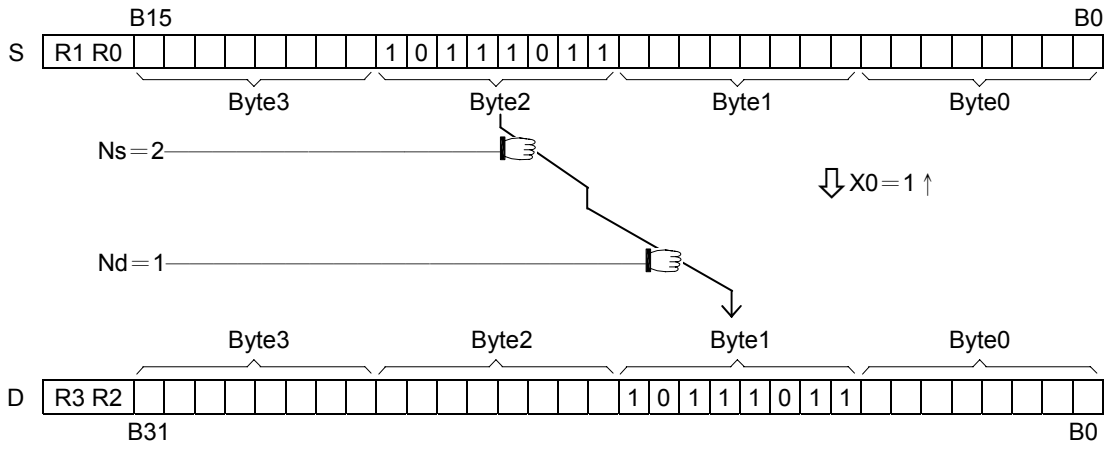
S, Ns, D, Nd may combine with V, Z, P0~P9 to serve indirect address application.

Operand \ Range	WX	WY	WM	WS	TMR	CTR	HR	IR	OR	SR	ROR	DR	K	XR
	WX0 | WX240	WY0 | WY240	WM0 | WM1896	WS0 | WS984	T0 | T255	C0 | C255	R0 | R3839	R3840 | R3903	R3904 | R3967	R3968 | R4167	R5000 | R8071	D0 | D4095	16/32-bit +/- number	V, Z P0~P9
S	○	○	○	○	○	○	○	○	○	○	○	○	○	○
Ns	○	○	○	○	○	○	○	○	○	○	○	○	0～3	○
D		○	○	○	○	○	○		○	○*	○*	○		○
Nd	○	○	○	○	○	○	○	○	○	○	○	○	0～3	○

- When move control "EN" =1 or "EN↑" (P instruction) changes from 0 to 1, move Nsth byte within S to Ndth byte position within D. (A byte is comprised of 8 bits. Starting from the lowest bit of the register, B0, each successive eight bits form a byte, so B0~B7 form byte 0, B8~B15 form byte 1, etc...)
- When the operand is 16 bits, the effective range of Ns or Nd is 0~1. For 32 bits (D instruction) operand, the range is 0~3. Beyond this range, will set the N value error flag "ERR" to 1, and do not carry out this instruction.

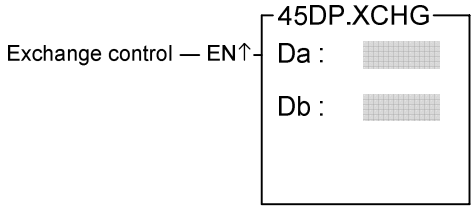


- The instruction at left moves the third byte (B16~B23) within S (32 bits register composed of R1R0), to the second byte within D (32 bits register composed of R3R2). Other bytes within D remain unchanged.



FUN 45 D P XCHG	EXCHANGE	FUN 45 D P XCHG

Ladder symbol



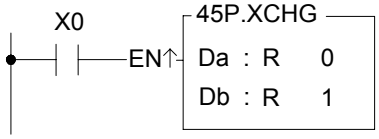
Da : Register a to be exchanged

Db : Register b to be exchanged

Da, Db may combine with V, Z, P0~P9 to serve indirect address application.

Range / Operand	WY	WM	WS	TMR	CTR	HR	OR	SR	ROR	DR	XR
	WY0 | WY240	WM0 | WM1896	WS0 | WS984	T0 | T255	C0 | C255	R0 | R3839	R3904 | R3967	R3968 | R4167	R5000 | R8071	D0 | D4095	V, Z P0~P9
Da	○	○	○	○	○	○	○	○*	○*	○	○
Db	○	○	○	○	○	○	○	○*	○*	○	○

- When exchange control "EN" =1 or "EN↑" (P instruction) changes from 0 to 1, will exchanges the contents of register Da and register Db in 16 bits or 32 bits (D instruction) format.



- The instruction at left exchanges the contents of the 16-bit R0 and R1 registers.

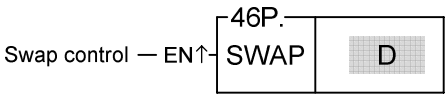
		B15															B0
Da	R0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Db	R1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

⇩X0＝1 ↑

		B15															B0
Da	R0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
Db	R1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

FUN 46 P SWAP	BYTE SWAP	FUN 46 P SWAP

Ladder symbol

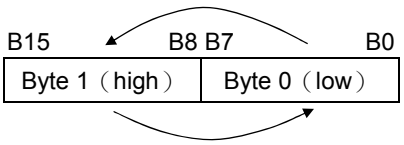


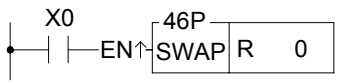
D : Register for byte data swap

D may combine with V, Z, P0~P9 to serve indirect address application.

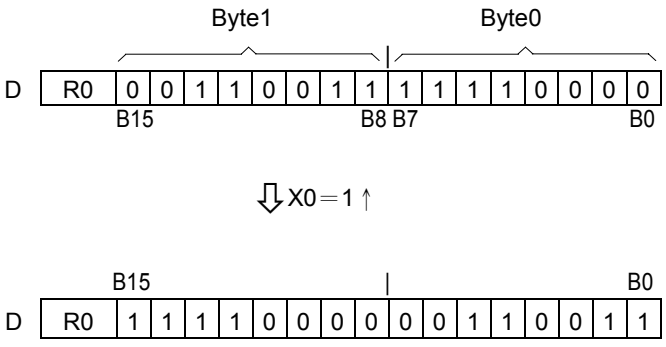
Range / Operand	WY	WM	WS	TMR	CTR	HR	OR	SR	ROR	DR	XR
	WY0 | WY240	WM0 | WM1896	WS0 | WS984	T0 | T255	C0 | C255	R0 | R3839	R3904 | R3967	R3968 | R4167	R5000 | R8071	D0 | D4095	V, Z P0~P9
D	○	○	○	○	○	○	○	○*	○*	○	○

- When swap control "EN" =1 or "EN↑" (P instruction) changes from 0 to 1, swap the data of the low byte, Byte 0 (B0~B7), and the high byte, Byte 1 (B8~B15), in the 16 bits register specified by D.



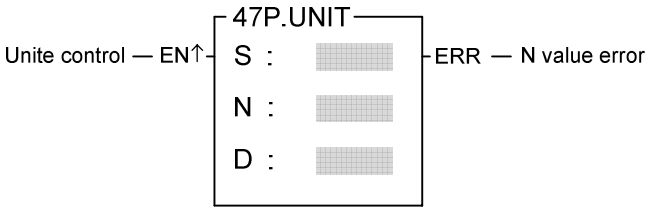


- The instruction at left swaps the data of the low byte (B0~B7) and the high byte (B8~B15) in R0. The results are as follows:



FUN 47 P UNIT	NIBBLE UNITE	FUN 47 P UNIT

Ladder symbol



S : Starting source register to be united

N : Number of nibbles to be united

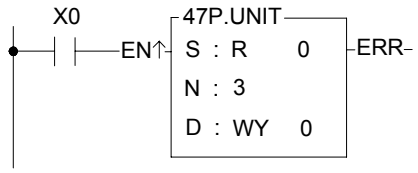
D : Registers storing united data

S, N, D may combine with V, Z, P0~P9 to serve indirect address application.

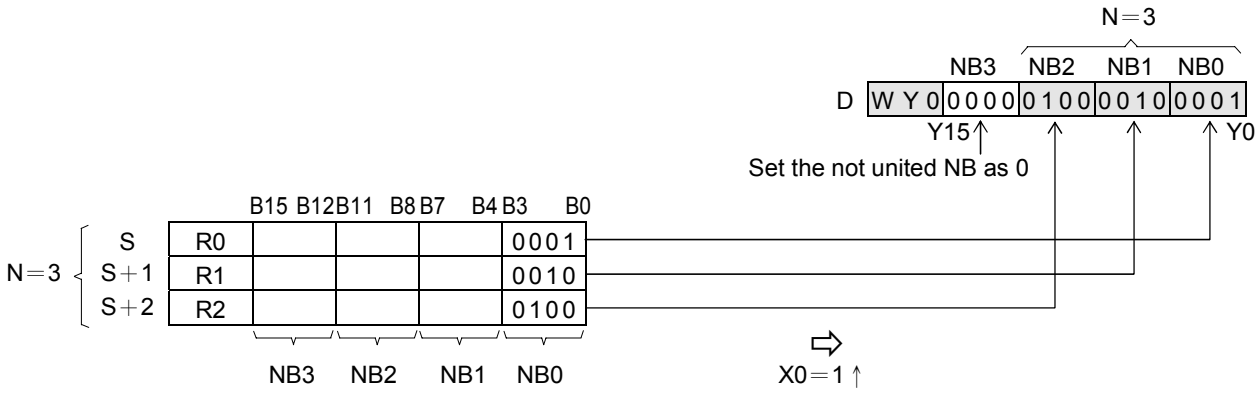
Range / Operand	WX	WY	WM	WS	TMR	CTR	HR	IR	OR	SR	ROR	DR	K	XR
	WX0 | WX240	WY0 | WY240	WM0 | WM1896	WS0 | WS984	T0 | T255	C0 | C255	R0 | R3839	R3840 | R3903	R3904 | R3967	R3968 | R4167	R5000 | R8071	D0 | D4095	1 | 4	V, Z P0~P9
S	○	○	○	○	○	○	○	○	○	○	○	○		○
N	○	○	○	○	○	○	○	○	○	○	○	○	○	○
D		○	○	○	○	○	○		○	○*	○*	○		○

- When unite control "EN" =1 or "EN↑" (P instruction) changes from 0 to 1, take out the lowest nibbles NB0, of N successive registers starting from S, and fill them into NB0, NB1, ...NBn-1 of D in ascending order. Nibbles not yet filled in D (when N is odd) are filled with 0. (A nibble is comprised by 4 bits. Starting from the lowest bit in the register, B0, each successive four bits form a nibble, so B0~B3 form nibble 0, B4~B7 form nibble 1, etc...).

- This instruction only provides WORD (16 bits) operand. Because of this, there are usually only 4 nibbles can be involved. Therefore the effective range of N is 1~4. Beyond this range, will set the N value error flag "ERR" to 1, and do not carry out this instruction.

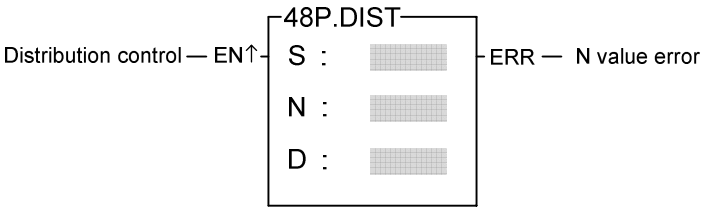


- The instruction at left takes out NB0 from 3 registers, R0, R1 and R2, and fills them into NB0~NB2 within WY0 register.



FUN 48 P DIST	NIBBLE DISTRIBUTE	FUN 48 P DIST

Ladder symbol



S : Source data to be distributed

N : Number of nibbles to be distributed

D : Starting register storing distribution data

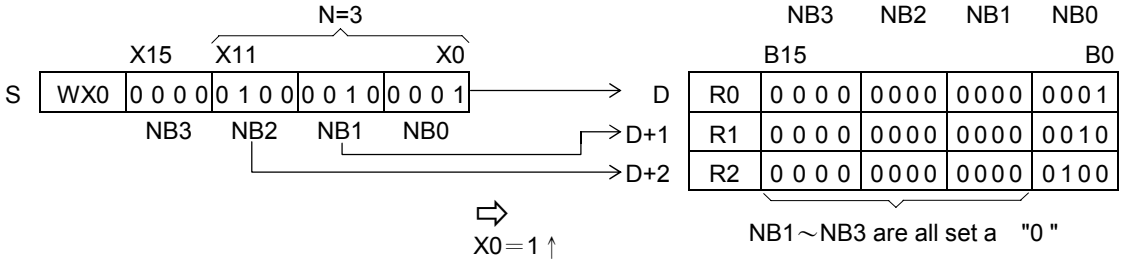
S, N, D may combine with V, Z, P0~P9 to serve indirect address application.

Range / Operand	WX	WY	WM	WS	TMR	CTR	HR	IR	OR	SR	ROR	DR	K	XR
	WX0 | WX240	WY0 | WY240	WM0 | WM1896	WS0 | WS984	T0 | T255	C0 | C255	R0 | R3839	R3840 | R3903	R3904 | R3967	R3968 | R4167	R5000 | R8071	D0 | D4095	16-bit +/- number	V, Z P0~P9
S	○	○	○	○	○	○	○	○	○	○	○	○	○	○
N	○	○	○	○	○	○	○	○	○	○	○	○	1~4	○
D		○	○	○	○	○	○		○	○*	○*	○		○

- When distribution control "EN" =1 or "EN↑" (P instruction) changes from 0 to 1, will take N successive nibbles starting from the lowest nibble NB0 within S, and distribute them in ascending order into the 0 nibbles of N registers starting from D. The nibbles other than NB0 in each of the registers within D are all set to zero. (A nibble is comprised by 4 bits. Starting from the lowest bit in a register, B0, each successive 4 bits form a nibble, so B0~B3 form nibble 0, B4~B7 form nibble 1, etc...)

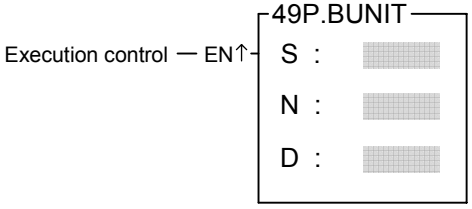
- This instruction only provides WORD (16 bits) operand. Therefore there are usually only 4 nibbles can be involved, so the effective value of N is 1~4. Beyond this range, will set the N value error flag "ERR" to 1, and do not carry out this instruction.

- The instruction at left writes NB0~NB2 from the WX0 register into the NB0 of the 3 consecutive registers R0~R2.



FUN49 P BUNIT	BYTE UNITE	FUN49 P BUNIT

Ladder symbol



S ：Starting address of source register to be united

N ：Number of bytes to be united

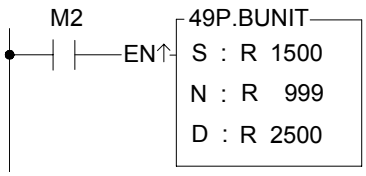
D ：Registers to store the united data

S, N, D may associate with V, Z, P0~P9 index register to serve the indirect addressing application.

Operand \ Range	HR R0 ~ R3839	ROR R5000 ~ R8071	DR D0 ~ D4095	K
S	○	○	○	
N	○	○	○	1~256
D	○	○*	○	

- When execution control "EN"=1 or "EN↑"（P instruction）changes from 0 to 1, it will perform the byte combination starting from S, length by N, and then store the results into D registers.
- This instruction will not act if invalid range of length.
- When communicating with intelligent peripheral in binary data format, this instruction may be applied to do byte combination for following word data processing.

Example：



Description：When M2 changes from 0→1, it will perform the byte combination starting from R1500, the length is assigned by R999, and then store the results into registers starting from R2500.
It is supposed R999=10, the results of combination will store into R2500～R2504.

S

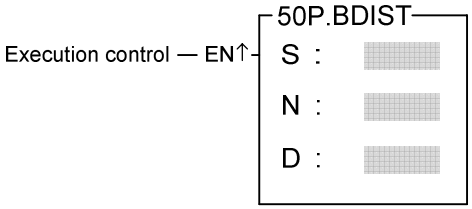
	High Byte	Low Byte
R1500	Don't care	Byte-0
R1501	Don't care	Byte-1
R1502	Don't care	Byte-2
R1503	Don't care	Byte-3
R1504	Don't care	Byte-4
R1505	Don't care	Byte-5
R1506	Don't care	Byte-6
R1507	Don't care	Byte-7
R1508	Don't care	Byte-8
R1509	Don't care	Byte-9

D

	High Byte	Low Byte
R2500	Byte-0	Byte-1
R2501	Byte-2	Byte-3
R2502	Byte-4	Byte-5
R2503	Byte-6	Byte-7
R2504	Byte-8	Byte-9

FUN50 P BDIST	BYTE DISTRIBUTE	FUN50 P BDIST

Ladder symbol



S ：Starting address of source register to be distributed

N ：Number of bytes to be distributed

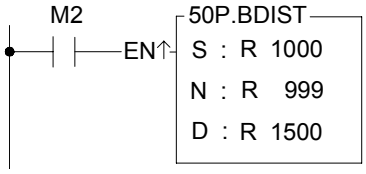
D ：Registers to store the distributed data

S, N, D may associate with V, Z, P0~P9 index register to serve the indirect addressing application.

Range / Operand	HR R0 | R3839	ROR R5000 | R8071	DR D0 | D4095	K
S	○	○	○	
N	○	○	○	1~256
D	○	○*	○	

- When execution control "EN" =1 or "EN↑" （P instruction） changes from 0 to 1, it will perform the byte distribution starting from S, length by N, and then store the results into D registers.
- This instruction will not act if invalid range of length.
- When communicating with intelligent peripheral in binary data format, this instruction may be applied to do byte distribution for data transmission.

Example：



Description：When M2 changes from 0→1, it will perform the byte distribution starting from R1000, the length is assigned by R999, and then store the results into registers starting from R1500.
It is supposed R999=9, the results of distribution will store into R1500～R1508.

S

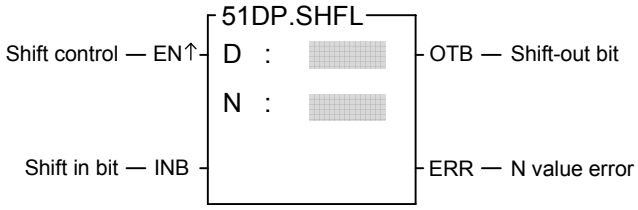
	High Byte	Low Byte
R1000	Byte-0	Byte-1
R1001	Byte-2	Byte-3
R1002	Byte-4	Byte-5
R1003	Byte-6	Byte-7
R1004	Byte-8	Don't care

D

	High Byte	Low Byte
R1500	00	Byte-0
R1501	00	Byte-1
R1502	00	Byte-2
R1503	00	Byte-3
R1504	00	Byte-4
R1505	00	Byte-5
R1506	00	Byte-6
R1507	00	Byte-7
R1508	00	Byte-8

FUN 51 D P SHFL	SHIFT LEFT	FUN 51 D P SHFL

Ladder symbol



D : Register to be shifted

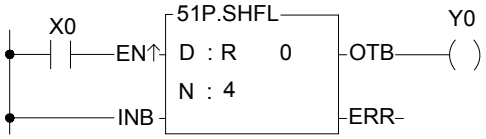
N : Number of bits to be shifted

N, D may combine with V, Z, P0~P9 to serve indirect address application.

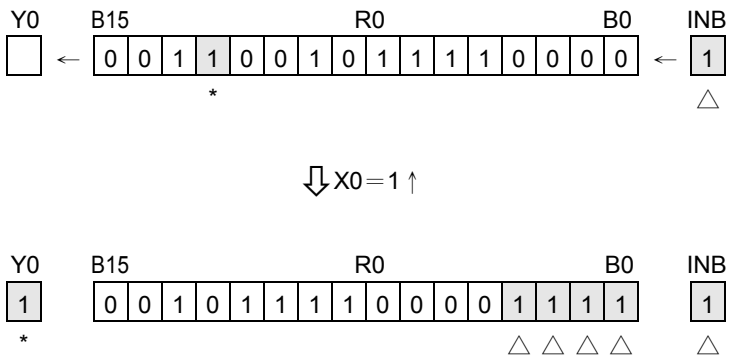
Range / Operand	WX	WY	WM	WS	TMR	CTR	HR	IR	OR	SR	ROR	DR	K	XR
	WX0 | WX240	WY0 | WY240	WM0 | WM1896	WS0 | WS984	T0 | T255	C0 | C255	R0 | R3839	R3840 | R3903	R3904 | R3967	R3968 | R4167	R5000 | R8071	D0 | D4095	1 | or | 16 32	V, Z P0~P9
D		○	○	○	○	○	○		○	○*	○*	○		○
N	○	○	○	○	○	○	○	○	○	○	○	○	○	○

- When shift control "EN" =1 or "EN↑" (P instruction) changes from 0 to 1, will shift the data of the D register towards the left by N successive bits (in ascending order). After the lowest bit B0 has been shifted left, its position will be replaced by shift-in bit INB, while the status of shift-out bits B15 or B31 (D instruction) will appear at shift-out bit "OTB".

- If the operand is 16 bits, the effective range of N is 1~16. For 32 bits (D instruction) operand, it is 1~32. Beyond this range, will set the N value error flag "ERR" to 1, and do not carry out this instruction.

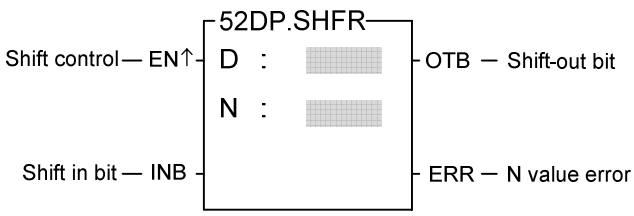


- The instruction at left shifts the data in register R0 towards the left by 4 successive bits. The results are shown below.



FUN 52 D P SHFR	SHIFT RIGHT	FUN 52 D P SHFR

Ladder symbol



D : Register to be shifted

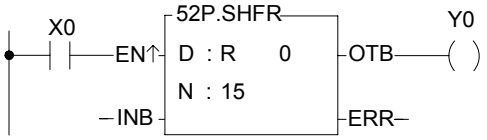
N : Number of bits to be shifted

D, N may combine with V, Z, P0~P9 to serve indirect address application.

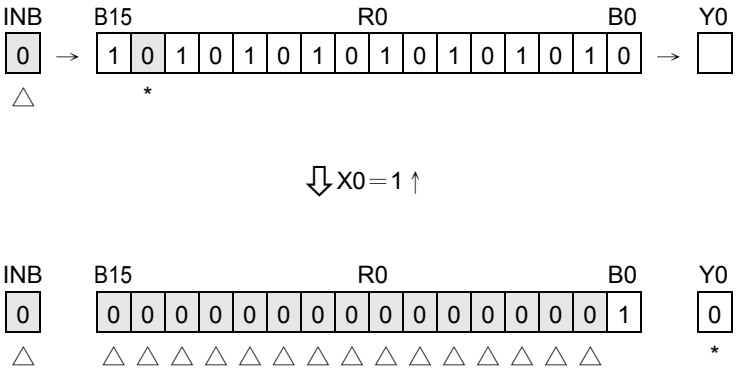
Operand \ Range	WX	WY	WM	WS	TMR	CTR	HR	IR	OR	SR	ROR	DR	K	XR
	WX0 | WX240	WY0 | WY240	WM0 | WM1896	WS0 | WS984	T0 | T255	C0 | C255	R0 | R3839	R3840 | R3903	R3904 | R3967	R3968 | R4167	R5000 | R8071	D0 | D4095	1 | or | 16 32	V, Z | P0~P9
D		○	○	○	○	○	○		○	○*	○*	○		○
N	○	○	○	○	○	○	○	○	○	○	○	○	○	○

- When shift control "EN" =1 or "EN↑" (P instruction) changes from 0 to 1, will shift the data of D register towards the right by N successive bits (in descending order). After the highest bits, B15 or B31 (D instruction) have been shifted right, their positions will be replaced by the shift-in bit INB, while shift-out bit B0 will appear at shift-out bit "OTB".

- If the operand is 16 bits, the effective range of N is 1~16. For 32 bits (D instruction) operand, it is 1~32. Beyond this range, will set the N value error flag "ERR" to 1, and do not carry out this instruction.

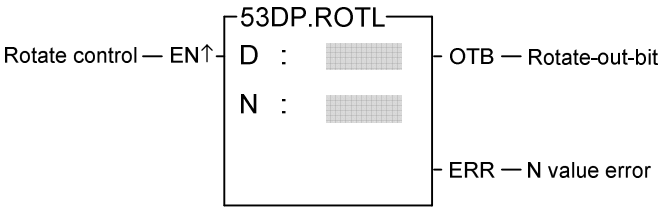


- The instruction at left shifts the data in R0 register towards the right by 15 successive bits. The results are shown below.



FUN 53 D P ROTL	ROTATE LEFT	FUN 53 D P ROTL

Ladder symbol



D : Register to be rotated

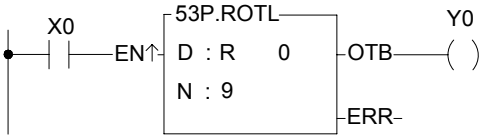
N : Number of bits to be rotated

D, N may combine with V, Z, P0~P9 to serve indirect address application.

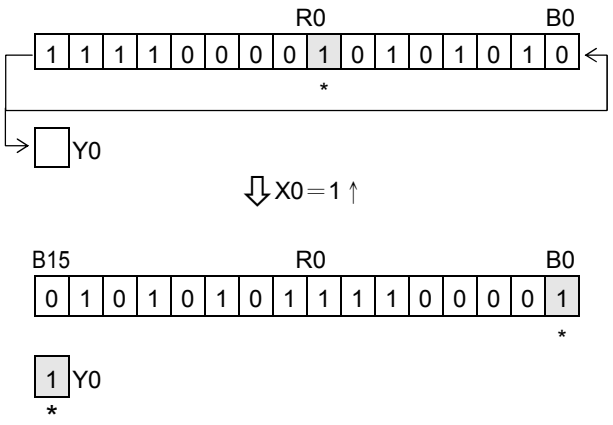
Range / Operand	WX	WY	WM	WS	TMR	CTR	HR	IR	OR	SR	ROR	DR	K	XR
	WX0 | WX240	WY0 | WY240	WM0 | WM1896	WS0 | WS984	T0 | T255	C0 | C255	R0 | R3839	R3840 | R3903	R3904 | R3967	R3968 | R4167	R5000 | R8071	D0 | D4095	1 | or | 16 32	V, Z P0~P9
D		○	○	○	○	○	○		○	○*	○*	○		○
N	○	○	○	○	○	○	○	○	○	○	○	○	○	○

- When rotate control "EN" =1 or "EN↑" (P instruction) changes from 0 to 1, will rotate the data of D register towards the left by N successive bits (in ascending order, i.e. in a 16-bit instruction, B0→B1, B1→B2, ... , B14→B15, B15→B0. In a 32-bit instruction, B0→B1, B1→B2, ... , B30→B31, B31→B0). At the same time, the status of the rotated out bits B15 or B31 (D instruction) will appear at rotate-out bit "OTB".

- If the operand is 16 bits, the effective range of N is 1~16. For 32 bits (D instruction) operand, it is 1~32. Beyond this range, will set the N value error flag "ERR" to 1, and do not carry out this instruction.

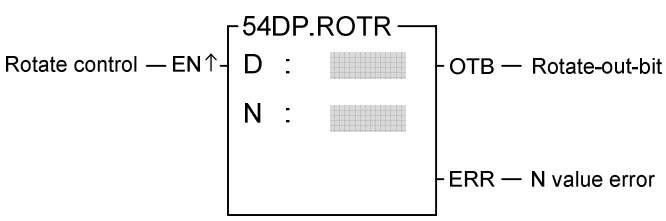


- The instruction at left rotates data from the R0 register towards the left 9 successive bits. The results are shown below.



FUN 54 D P ROTR	ROTATE RIGHT	FUN 54 D P ROTR

Ladder symbol



D : Register to be rotated

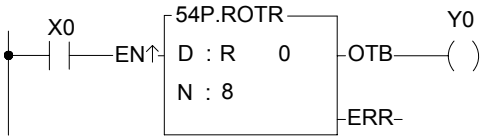
N : Number of bits to be rotated

D, N may combine with V, Z, P0~P9 to serve indirect address application.

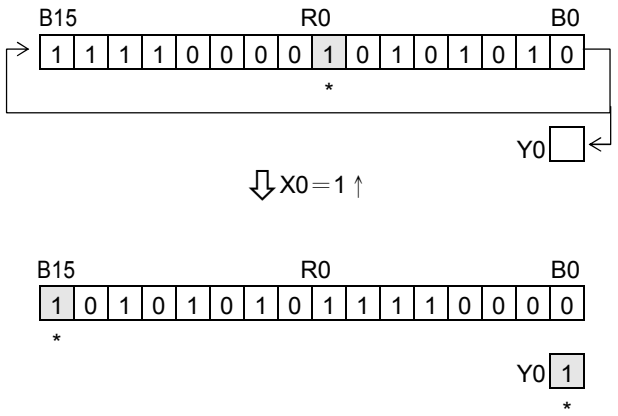
Operand \ Range	WX	WY	WM	WS	TMR	CTR	HR	IR	OR	SR	ROR	DR	K	XR
	WX0 | WX240	WY0 | WY240	WM0 | WM1896	WS0 | WS984	T0 | T255	C0 | C255	R0 | R3839	R3840 | R3903	R3904 | R3967	R3968 | R4167	R5000 | R8071	D0 | D4095	1 | or | 16 32	V, Z P0~P9
D		○	○	○	○	○	○		○	○*	○*	○		○
N	○	○	○	○	○	○	○	○	○	○	○	○	○	○

- When rotate control "EN" =1 or "EN↑" (P instruction) changes from 0 to 1, will rotate the bit data of D register towards the right by N successive bits (in descending order, i.e. in a 16-bit instruction, B15→B14, B14→B13, ... , B1→B0, B0→B15. In a 32-bit instruction, B31→B30, B30→B29, ... , B1→B0, B0→B31). At the same time, the status of the rotated out B0 bits will appear at the rotate-out bit "OTB".

- If the operand is 16 bit, the effective range of N is 1~16. For 32 bits (D instruction) operand, it is 1~32. Beyond this range, will set the N value error flag "ERR" to 1, and do not carry out this instruction.

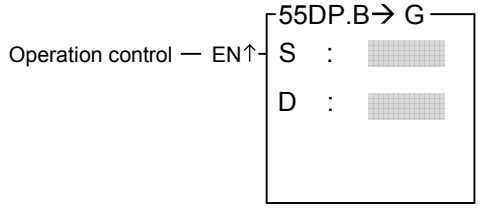


- The instruction at left rotates data from R0 register towards the right 8 successive bits. The results are shown below.



FUN55 D P B→G	BINARY-CODE TO GRAY-CODE CONVERSION	FUN55 D P B→G

Ladder symbol



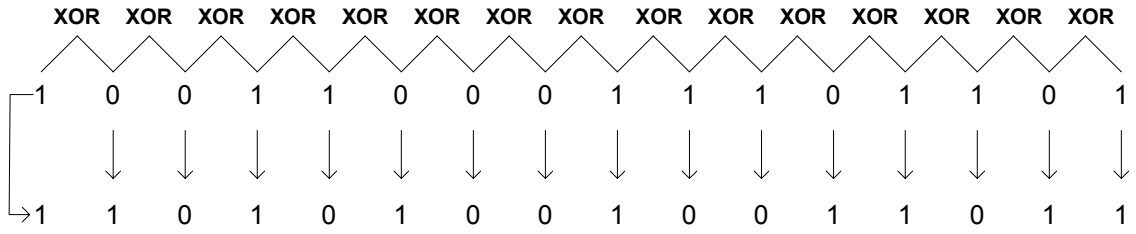
S ：Starting of source

D ：Starting address of destination

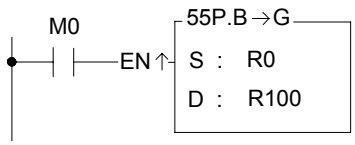
S，D operand can combine V, Z, P0~P9 for index addressing.

Range / Ope-rand	WX	WY	WM	WS	TMR	CTR	HR	IR	OR	SR	ROR	DR	K	XR
	WX0 | WX240	WY0 | WY240	WM0 | WM1896	WS0 | WS984	T0 | T255	C0 | C255	R0 | R3839	R3840 | R3903	R3904 | R3967	R3968 | R4167	R5000 | R8071	D0 | D4095	16-bit +/- number	V, Z P0~P9
S	○	○	○	○	○	○	○	○	○	○	○	○	○	○
D		○	○	○			○				○*	○		○

- When operation control "EN"=1 or "EN↑"(P instruction) changes from 0 to 1, it will perform the code conversion; where S is the source (Binary code), and D is the destination (Gray code) for storing the result.
- If the operand is 16 bits, the conversion result is stored into one register. If the operand is 32 bits, the conversion result is stored into two registers.
- The conversion method shown as below



Example 1: When M0 changes from 0→1, it will perform the 16-bit code conversion.

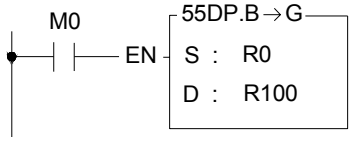


- Converting the 16-bit Binary-code in R0 into Gray-code, and then storing the result into R100.

R0＝1001010101010011B ➔ R100＝1101111111111010B

FUN55 D P B→G	BINARY-CODE TO GRAY-CODE CONVERSION	FUN55 D P B→G

Example 2: When M0 =1, it will perform the 32-bit code conversion.

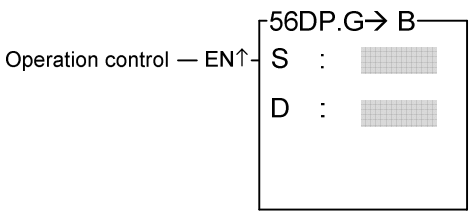


- Converting the 32-bit Binary-code in DR0 into Gray-code, and then storing the result into DR100.

DR0＝00110111001001000010111100010100B ➔ DR100＝00101100101101100011100010011110B

FUN56 D P G→B	GRAY-CODE TO BINARY-CODE CONVERSION	FUN56 D P G→B

Ladder symbol



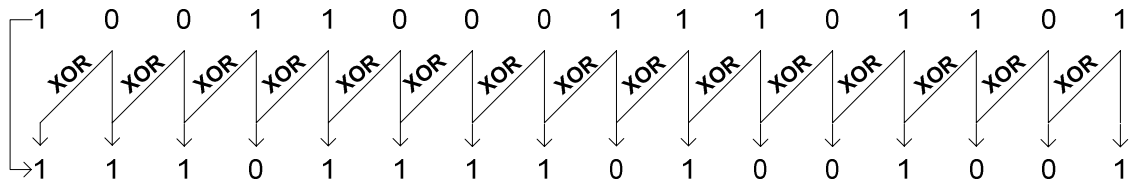
S ：Starting of source

D ：Starting address of destination

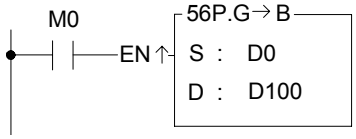
S ，D operand can combine V, Z, P0~P9 for index addressing.

Range / Ope-rand	WX	WY	WM	WS	TMR	CTR	HR	IR	OR	SR	ROR	DR	K	XR
	WX0 | WX240	WY0 | WY240	WM0 | WM1896	WS0 | WS984	T0 | T255	C0 | C255	R0 | R3839	R3840 | R3903	R3904 | R3967	R3968 | R4167	R5000 | R8071	D0 | D4095	16-bit +/- number	V, Z P0~P9
S	○	○	○	○	○	○	○	○	○	○	○	○	○	○
D		○	○	○			○				○*	○		○

- When operation control "EN"=1 or "EN↑"(P instruction) changes from 0 to 1, it will perform the code conversion; where S is the source (Gray code), and D is the destination (Binary code) for storing the result.
- If the operand is 16 bits, the conversion result is stored into one register. If the operand is 32 bits, the conversion result is stored into two registers.
- The conversion method shown as below :



Example 1: When M0 changes from 0→1, it will perform the 16-bit code conversion.

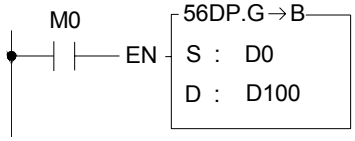


˙Converting the 16-bit Gray-code in D0 into Binary-code, and then storing the result into D100.

D0＝1001010101010011B ➔ D100＝1110011001100010B

FUN56 D P G→B	GRAY-CODE TO BINARY-CODE CONVERSION	FUN56 D P G→B

Example 2: When M0 =1, it will perform the 32-bit code conversion.

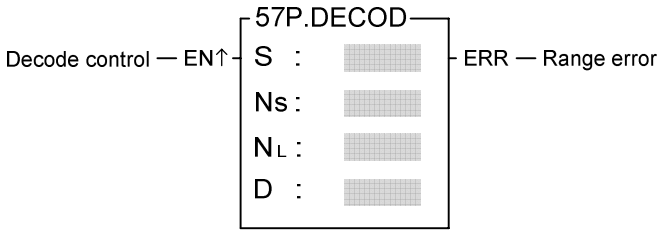


˙Converting the 32-bit Gray-code in DD0 into Binary-code, and then storing the result into DD100.

DD0＝00110111001001000010111100010100B ➔ DD100＝00100101110001111100101000011000B

FUN 57 P DECOD	DECODE	FUN 57 P DECOD

Ladder symbol



S : Source data register to be decoded (16 bits)

N_S : Starting bits to be decoded within S

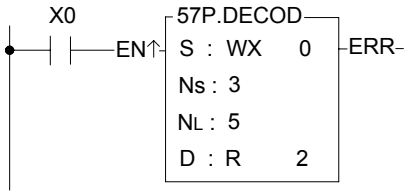
N_L : Length of decoded value (1~8 bits)

D : Starting register storing decoded results (2~256 points = 1~16 words).

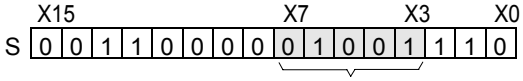
S, N_S, N_L, D may combine with V, Z, P0~P9 to serve indirect address application.

Range / Operand	WX	WY	WM	WS	TMR	CTR	HR	IR	OR	SR	ROR	DR	K	XR
	WX0 | WX240	WY0 | WY240	WM0 | WM1896	WS0 | WS984	T0 | T255	C0 | C255	R0 | R3839	R3840 | R3903	R3904 | R3967	R3968 | R4167	R5000 | R8071	D0 | D4095	16-bit +/- number	V, Z P0~P9
S	○	○	○	○	○	○	○	○	○	○	○	○	○	○
N_S	○	○	○	○	○	○	○	○	○	○	○	○	0～15	○
N_L	○	○	○	○	○	○	○	○	○	○	○	○	1～8	○
D		○	○	○	○	○	○		○	○*	○*	○		○

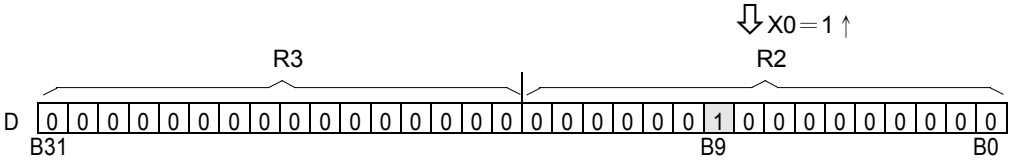
- This instruction, will set a single bit among the total of 2^{NL} discrete points (D) to 1 and the others bit are set to 0. The bit number to be set to 1 is specified by the value comprised by BN_S～BN_S+N_L-1 of S（which is called the decode value, BN_S is the starting bit of the decode value, and BN_S+N_L-1 is the end value）.
- When decode control "EN" =1 or "EN↑" (P instruction) changes from 0 to 1, will take out the value BN_S～BN_S+N_L-1 from S. And with this value to locate the bit position and set D accordingly, and set all the other bit to zero.
- This instruction only provides 16 bits operand, which means S only has B0~B15. Therefore the effective range of Ns is 0~15, and the N_L length of the decode value is limited to 1~8 bits. Therefore the width of the decoded result D is $2^{1\sim8}$ points = 2~256 points = 1~16 words (if 16 points are not sufficient, 1 word is still occupied). If the value of N_S or N_L is beyond the above range, will set the range-error flag "ERR" to 1, and do not carry out this instruction.
- If the end bit value exceeds the B15 of S, then will extend toward B0 of S + 1. However if this occurs then S+1 can't exceed the range of specific type of operand (ie. If S is of D type register then S+1 can't be D+1). If violate this, then this instruction only takes out the bits from starting bit BNs to its highest limit as the decode value.



- The instruction at left takes out the data of five successive bits from X3 to X7 within the WX0 register and decodes it. The results are then stored in the 32-bit register starting at R2.



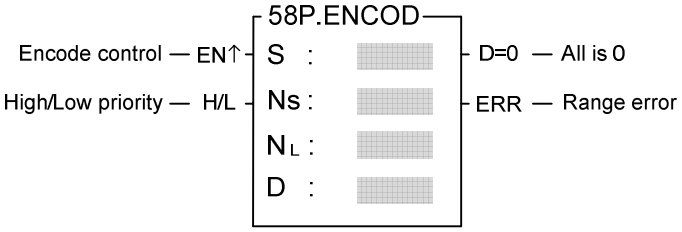
Length of decode value N_L=5,so bit value is formed by X7~X3 (equal 9)



Because N_L=5,the width of D is 2^5= 32 points = 2 words. That is, D is formed by R3R2, and the decoded value is 01001=9, therefore B9 (the 10th point) within D is set to 1, and all other points are 0.

FUN 58 P ENCOD	ENCODE	FUN 58 P ENCOD

Ladder symbol



S : Starting register to be encoded

N_S : Bit position within S as the encoding start point

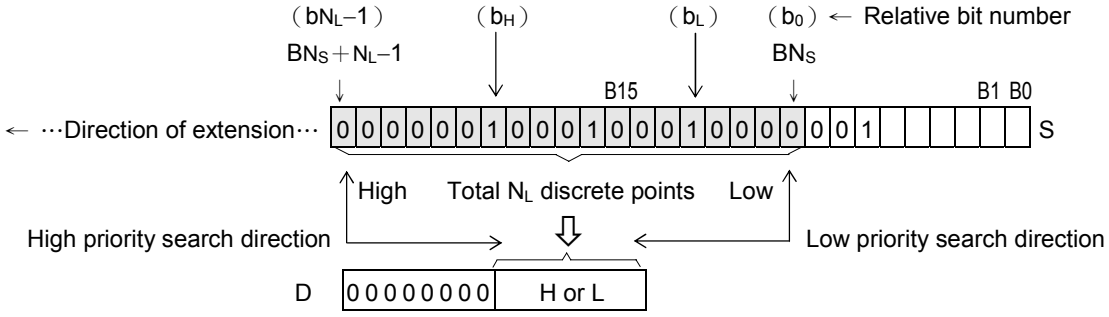
N_L : Number of encoding discrete points (2~256)

D : Number of register storing encoding results (1 word)

S, N_S, N_L, D may combine with V, Z, P0~P9 to serve indirect address application.

Range / Operand	WX	WY	WM	WS	TMR	CTR	HR	IR	OR	SR	ROR	DR	K	XR
	WX0 | WX240	WY0 | WY240	WM0 | WM1896	WS0 | WS984	T0 | T255	C0 | C255	R0 | R3839	R3840 | R3903	R3904 | R3967	R3968 | R4167	R5000 | R8071	D0 | D4095	16-bit +/- number	V, Z P0~P9
S	○	○	○	○	○	○	○	○	○	○	○	○		○
N_S	○	○	○	○	○	○	○	○	○	○	○	○	0～15	○
N_L	○	○	○	○	○	○	○	○	○	○	○	○	2～256	○
D		○	○	○	○	○	○		○	○*	○*	○		○

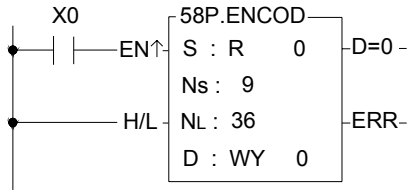
- When encode control "EN" =1 or "EN↑" (P instruction) changes from 0 to 1, will starting from the points specified by Ns within S, take out towards the left (high position direction) N_L number of successive bits BN_S～BN_S+N_L-1 (BN_S is called the encoding start point, and its relative bit number is b0;BN_S+N_L-1 is called the encoding end point, and its relative bit number is BN_L-1). From left to right do higher priority (when H/L=1) encoding or from right to left do lower priority (when H/L=0) encoding (i.e. seek the first bit with the value of 1, and the relative bit number of this point will be stored into the low byte (B0~B7) of encoded resultant register D, and the high byte of D will be filled with 0.



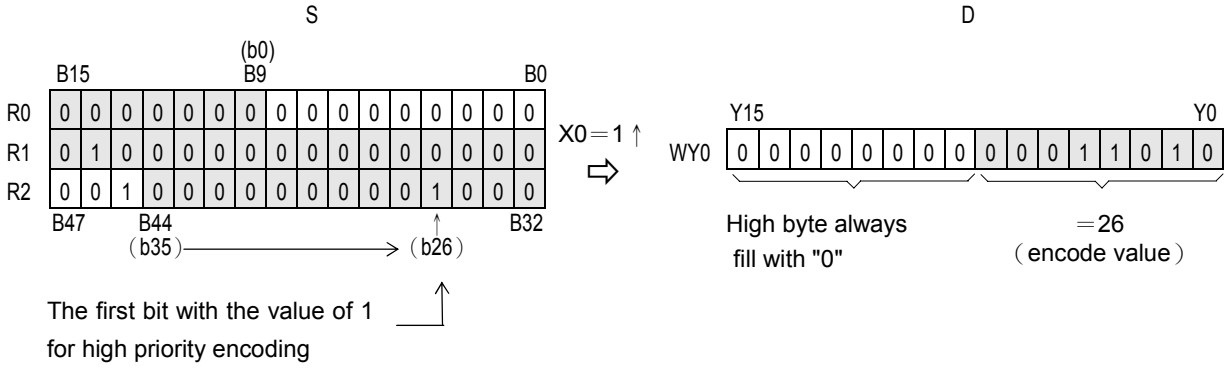
- As shown in the diagram above, for high priority encoding, the bit first to find is b_H (with a value of 12), and for low priority encoding, the bit first to find b_L (with a value of 4). Among the N_L discrete points there must be at least one bit with value of 1. If all bits are 0, will not to carry out this instruction, and the all zero flag "D=0" will set to 1.
- Because S is a 16-bit register, Ns can be 0~15, and is used to assign a point of B0~B15 within S as the encoding start point (b0). The value of N_L can be 2~256, and it is used to identify the encoding end point, i.e. it assigns N_L successive single points starting from the start point (b0) towards the left (high position direction) as the encoding zone (i.e. b0～bN_L-1). If the value of Ns or NL exceeds the above value, then do not carry out this instruction, and set the range-error flag "ERR" as 1.

FUN 58 P ENCOD	ENCODE	FUN 58 P ENCOD

- If the encoding end point (bN_L-1) beyond the B15 of S, then continue extending towards S+1, S+2, but it must not exceed the range of specific type of operand. If it goes beyond this, then this instruction can only take the discrete points between b0 and the highest limit into account for encoding.

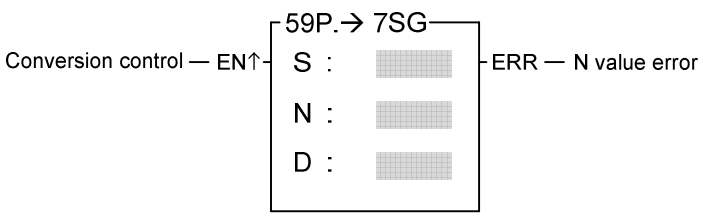


- The instruction at left is a high priority encode example. When X0 goes from 0 to 1, will take out toward left 36 successive bits starting from B9 (b0) specified by Ns within S, and perform high priority encoding (because H/L = 1). That is, starting from b35 (encoding end point), move right to find the first bit with the value of 1. The resultant value of this example is b26, so the value of D is 001AH=26, as shown in the diagram below.



FUN 59 P →7SG	7-SEGMENT CONVERSION	FUN 59 P →7SG

Ladder symbol



S : Source data to be converted

N : The nibble number within S for conversion

D : Register storing 7-segment result

S, N, D may combine with V, Z, P0~P9 to serve indirect address application.

Range / Operand	WX	WY	WM	WS	TMR	CTR	HR	IR	OR	SR	ROR	DR	K	XR
	WX0 | WX240	WY0 | WY240	WM0 | WM1896	WS0 | WS984	T0 | T255	C0 | C255	R0 | R3839	R3840 | R3903	R3904 | R3967	R3968 | R4167	R5000 | R8071	D0 | D4095	16-bit +/- number	V, Z P0~P9
S	○	○	○	○	○	○	○	○	○	○	○	○	○	○
N	○	○	○	○	○	○	○	○	○	○	○	○	0～3	○
D		○	○	○	○	○	○		○	○*	○*	○		○

- When conversion control "EN" =1 or "EN↑" (P instruction) changes from 0 to 1, will convert N+1 number of nibbles (A nibble is comprised by 4 successive bits, so B0~B3 of S form nibble 0, B4~B7 form nibble 1, etc...) within S to 7-segment code, and store the code into a low byte of D (High bytes does not change). The 7 segment within D are put in sequence, with "a" segment placed at B6, "b" segment at B5, ... ,"g" segment at B0. B7 is not used and is fixed as 0.
- Because this instruction is limited to 16 bits, and S only has 4 nibbles (NB0~NB3), the effective range of N is 0~3. Beyond this range, will set the N value flag error "ERR" to 1, and does not carry out this instruction.
- Care should be taken on total nibbles to be converted is N+1. N=0 means one digit to convert, N=1 means two digits to convert etc…

FUN 59 P →7SG	7-SEGMENT CONVERSION	FUN 59 P →7SG

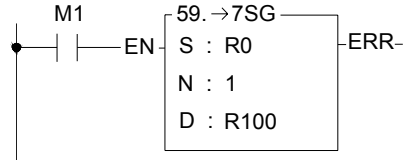
〈Example 1〉When M1 OFF→ON, convert hexadecimal to 7-Segment.

- Figure left shown the conversion of first digit (nibble) of R0 to 7-segment and store in low byte of R100, the high byte of R100 remain unchanged.

Original R100＝0000H

R0＝0001H ➔ R100＝0030H（1）

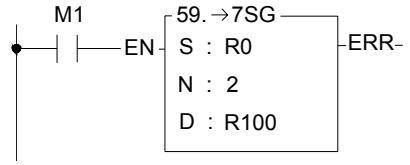
〈Example 2〉When M1 ON, convert the hexadecimal to 7-Segment.



- Instruction at left will convert the first and the second digit of R0 to 7-segment and store in R100.
- The low byte of R100 stores first digit.
- The high byte of R100 stores second digit.

R0＝0056H ➔ R100＝5B5FH（56）

〈Example 3〉When M1 ON, converting hexadecimal to 7-Segment.



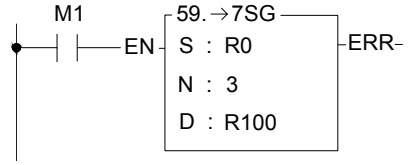
- Instruction at left will convert the first, second and third digit of R0 to 7-segment and store in R100 and R101.
- The low byte of R100 stores first digit.
- The high byte of R100 stores second digit.
- The low byte of R101 stores third digit.
- The high byte of R101 remain unchanged.

Original R101=0000H

R0=0A48H ➔ R100=337FH（48）

R101=0077H（A）

〈Example 4〉When M1 ON, convert hexadecimal to 7-Segment.

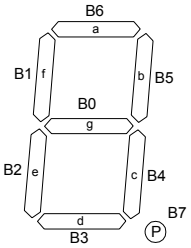


- Instruction at left will convert 1~4 digit of R0 to 7-segment and store in R100 and R101.
- The low byte of R100 stores first digit.
- The high byte of R100 stores second digit.
- The low byte of R101 stores third digit.
- The high byte of R101 stores 4^{th} digit.

R0=2790H ➔ R100=7B7EH（90）

R101=6D72H（27）

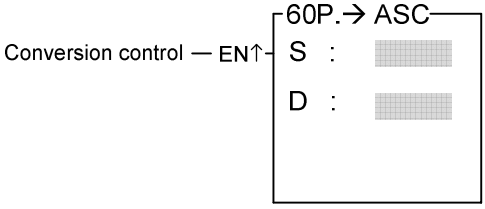
FUN 59 P →7SG	7-SEGMENT CONVERSION	FUN 59 P →7SG

Nibble data of S		7-segment display format	Low byte of D								Display pattern
Hexadecimal number	Binary number		B7 ●	B6 a	B5 b	B4 c	B3 d	B2 e	B1 f	B0 g	
0	0000		0	1	1	1	1	1	1	0	
1	0001		0	0	1	1	0	0	0	0	
2	0010		0	1	1	0	1	1	0	1	
3	0011		0	1	1	1	1	0	0	1	
4	0100		0	0	1	1	0	0	1	1	
5	0101		0	1	0	1	1	0	1	1	
6	0110		0	1	0	1	1	1	1	1	
7	0111		0	1	1	1	0	0	1	0	
8	1000		0	1	1	1	1	1	1	1	
9	1001		0	1	1	1	1	0	1	1	
A	1010		0	1	1	1	0	1	1	1	
B	1011		0	0	0	1	1	1	1	1	
C	1100		0	1	0	0	1	1	1	0	
D	1101		0	0	1	1	1	1	0	1	
E	1110		0	1	0	0	1	1	1	1	
F	1111		0	1	0	0	0	1	1	1	

7-segment display pattern table

FUN 60 P →ASC	ASCII CONVERSION	FUN 60 P →ASC

Ladder symbol

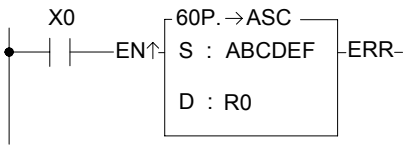


S : Alphanumeric to be converted into ASCII code

D : Starting register storing ASCII results

Range / Operand	WY	WM	WS	TMR	CTR	HR	OR	SR	ROR	DR	Alphanumeric
	WY0 | WY240	WM0 | WM1896	WS0 | WS984	T0 | T255	C0 | C255	R0 | R3839	R3904 | R3967	R3968 | R4167	R5000 | R8071	D0 | D4095	1～12 alphanumeric
S											○
D	○	○	○	○	○	○	○	○*	○*	○	

- When conversion control "EN" =1 or "EN↑" (P instruction) changes from 0 to 1, will convert alphabets and numbers stored in S (S has a maximum of 12 alphanumeric characters) into ASCII and store it into registers starting from D. Each 2 alphanumeric characters occupy one 16-bit register.
- The application of this instruction, most often, stores alphanumeric information within a program, and waits until certain conditions occur, then converts this alphanumeric information into ASCII and conveys it to external display devices which can accept ASCII code.



- The instruction at left converts the 6 alphabets -ABCDEF into ASCII then stores it into 3 successive registers starting from R0.

S

Alphabet
ABCDEF

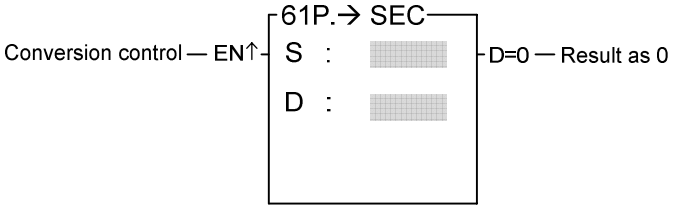
X0＝1 ↑ ⇨

D

	High Byte	Low Byte
R0	42（B）	41（A）
R1	44（D）	43（C）
R2	46（F）	45（E）

FUN 61 P →SEC	HOUR : MINUTE : SECOND TO SECONDS CONVERSION	FUN 61 P →SEC

Ladder symbol

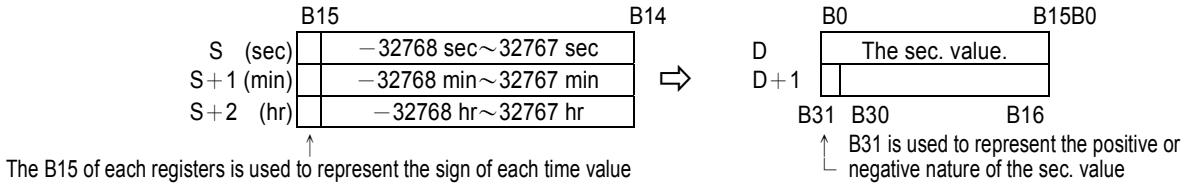


S : Starting calendar data register to be converted

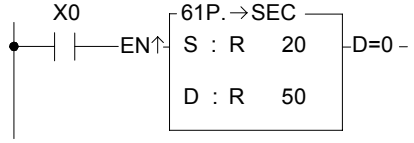
D : Starting register storing results

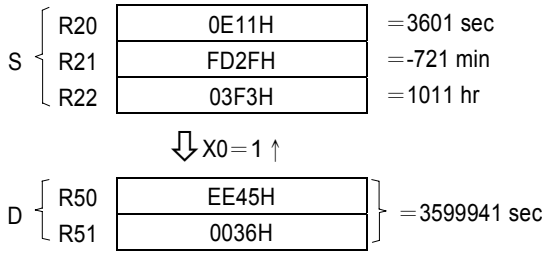
Range / Operand	WX	WY	WM	WS	TMR	CTR	HR	IR	OR	SR	ROR	DR
	WX0 | WX240	WY0 | WY240	WM0 | WM1896	WS0 | WS984	T0 | T255	C0 | C255	R0 | R3839	R3840 | R3903	R3904 | R3967	R3968 | R4167	R5000 | R8071	D0 | D4095
S	○	○	○	○	○	○	○	○	○	○	○	○
D		○	○	○	○	○	○		○	○*	○*	○

- When conversion control "EN" =1 or "EN↑" (P instruction) changes from 0 to 1, will convert the hour: minute: second data of S~S+2 into an equivalent value in seconds and store it into the 32-bit register formed by combining D and D+1. If the result = 0, then set the "D = 0" flag as 1.
- Among the WSZ-controller instructions, the hour: minute: second time related instructions (FUN61 and 62) use 3 words of register to store the time data, as shown in the diagram below. The first word is the second register, the second word is the minute register, and finally the third word is the hour register, and in the 16 bits of each register, only B14~B0 are used to represent the time value. While bit B15 is used to express whether the time values are positive or negative. When B15 is 0, it represents a positive time value, and when B15 is 1 it represents a negative time value. The B14~B0 time value is represented in binary, and when the time value is negative, B14~B0 is represented with the 2's complement. The number of seconds that results from this operation is the result of summation of seconds from the three registers representing hours: minutes: seconds.



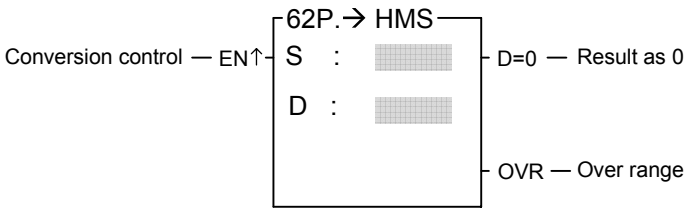
- Besides FUN61 or 62 instructions which treat hour: minute: second registers as an integral data, other instructions treat it as individual registers.
- The example program at below converts the hour: minute: second data formed by R20~R22 into their equivalent value in seconds then stored in the 32-bit register formed by R50~R51. The results are shown below.





FUN 62 P →HMS	SECOND→HOUR：MINUTE：SECOND	FUN 62 P →HMS

Ladder symbol

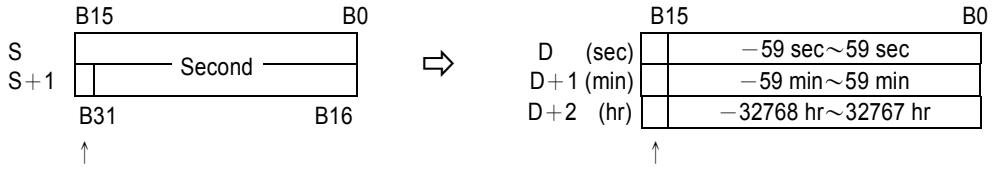


S :Starting register of second to be converted

D :Starting register storing result of conversion (hour : minute : second)

Range / Operand	WX	WY	WM	WS	TMR	CTR	HR	IR	OR	SR	ROR	DR	K
	WX0 | WX240	WY0 | WY240	WM0 | WM1896	WS0 | WS984	T0 | T255	C0 | C255	R0 | R3839	R3840 | R3903	R3904 | R3967	R3968 | R4167	R5000 | R8071	D0 | D4095	-117968399 | 117964799
S	○	○	○	○	○	○	○	○	○	○	○	○	○
D		○	○	○	○	○	○		○	○*	○*	○	

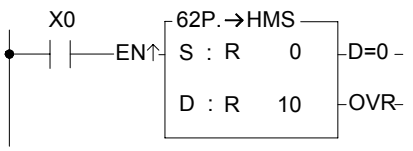
- When conversion control "EN" =1 or "EN↑" (P instruction) changes from 0 to 1, will convert the second data from the S~S+1 32-bit register into the equivalent hour : minute : second time value and store it in the three successive registers D~D+2. All the data in this instruction is represented in binary. (if there is a negative value it is represented using the 2's complement.)

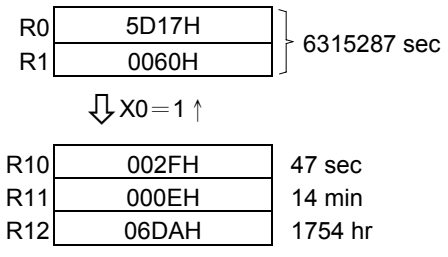


The bit B31 of the second register is used as the sign bit of the second value.

The bits B15 of each register are used as the sign bit of the hour: minute: second value.

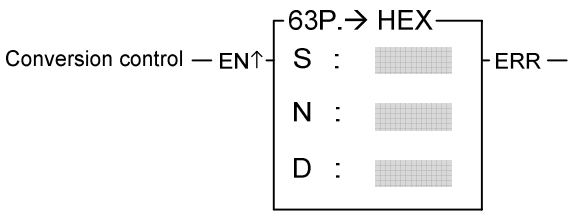
- As shown in the diagram above, after convert to hour : minute : second value, the minute : second value can only be in the range of -59 to 59, and the hour number can be in the range of -32768 to 32767 hours. Because of this, the maximum limit of D is -32768 hours, -59 minutes, -59 seconds to 32767 hours, 59 minutes, 59 seconds, the corresponding second value of S which is in the range of -117968399 to 117964799 seconds. If the S value exceeds this range, this instruction cannot be carried out, and will set the over range flag "OVR" to 1. If S = 0 then result is 0 flag "D = 0" will be set to 1.
- The program in the diagram below is an example of this instruction. Please note that the content of the registers are denoted by hexadecimal, and on the right is its equivalent value in decimal notation.





FUN 63 P →HEX	CONVERSION OF ASCII CODE TO HEXADECIMAL VALUE	FUN 63 P →HEX

Ladder symbol



S : Starting source register.

N : Number of ASCII codes to be converted to hexadecimal values.

D : The starting register that stores the result (hexadecimal value).

S, N, D, can associate with V, Z, P0~P9 to do the indirect addressing application.

Range / Operand	WX	WY	WM	WS	TMR	CTR	HR	IR	OR	SR	ROR	DR	K	XR
	WX0 | WX240	WY0 | WY240	WM0 | WM1896	WS0 | WS984	T0 | T255	C0 | C255	R0 | R3839	R3840 | R3903	R3904 | R3967	R3968 | R4167	R5000 | R8071	D0 | D4095	16-bit +number	V, Z P0~P9
S	○	○	○	○	○	○	○	○	○	○	○	○		○
N	○	○	○	○	○	○	○	○	○	○	○	○	1～511	○
D		○	○	○	○	○	○		○	○*	○*	○		○

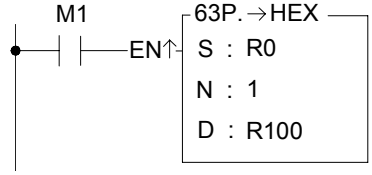
- When conversion control “EN” =1 or “EN↑” (P instruction) changes from 0 to 1, it will convert the N successive hexadecimal ASCII character(‘0’~’9’,’A’~’F’) convey by 16 bits registers (Low Byte is effective) into hexadecimal value, and store the result into the register starting with D. Every 4 ASCII code is stored in one register. The nibbles of register, which does not involve in the conversion of ASCII code will remain unchanged.

- The conversion will not be performed when N is 0 or greater than 511.

- When there is ASCII error (neither 30H～39H nor 41H～46H), the output “ERR” is ON.

- The main purpose of this instruction is to convert the hexadecimal ASCII character (‘0’~’9’,’A’~’F’), which is received by communication port1 or communication port2 from the external ASCII peripherals, to the hexadecimal values that the CPU can process directly.

〈Example 1〉 When M1 from OFF→ON, ASCII code converted to hexadecimal value.



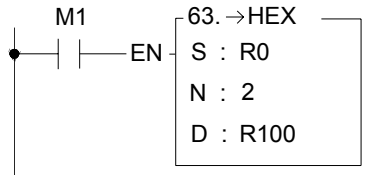
- Converts the ASCII code of R0 into hexadecimal value and store to nibble0 (nibble1~nibble3 remain unchanged) of R100.

Originally R100＝0000H

R0＝0039H（9）➔ R100＝0009H

FUN 63 P →HEX	CONVERSION OF ASCII CODE TO HEXADECIMAL VALUE	FUN 63 P →HEX

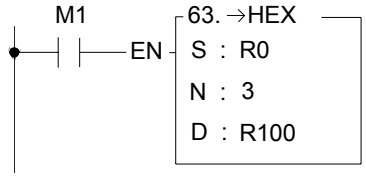
〈Example 2〉When M1 is ON, ASCII code converted to hexadecimal value.



- Converts the ASCII code of R0 and R1 into hexadecimal value and store to low byte (high byte remain unchanged) of R100.

R0＝0039H（9） Originally R100＝0000H
R1＝0041H（A）➔ R100＝009AH

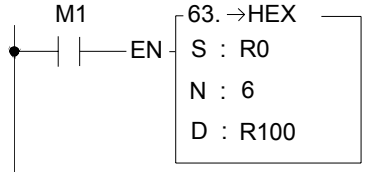
〈Example 3〉When M1 is ON, ASCII code converted to hexadecimal value.



- Converts the ASCII code of R0 and R1 into hexadecimal value and store result into R100 (nibble 3 remain unchanged).

R0＝0039H （9） Originally R100＝0000H
R1＝0041H （A）
R2＝0045H （E）➔ R100＝09AEH

〈Example 4〉When M1 is ON, ASCII code converted to hexadecimal value.

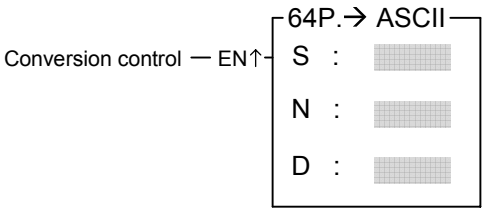


- Converts the ASCII code of R0~R5 into hexadecimal value and store it to R100~R101.

R0＝0031H（1） Originally R100＝0000H
R1＝0032H（2） R101＝0000H
R2＝0033H（3）
R3＝0034H（4）
R4＝0035H（5）➔ R100＝3456H
R5＝0036H（6） R101＝0012H

FUN 64 P →ASCII	CONVERSION OF HEXADECIMAL VALUE TO ASCII CODE	FUN 64 P →ASCII

Ladder symbol

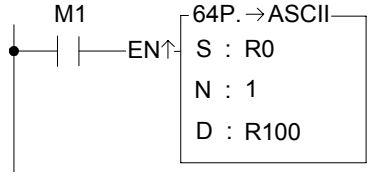


S : Starting source register
N : Number of hexadecimal digit to be converted to ASCII code
D : The starting register storing result.
S, N, D, can associate with V, Z, P0~P9 to do the indirect addressing application.

Range / Operand	WX	WY	WM	WS	TMR	CTR	HR	IR	OR	SR	ROR	DR	K	XR
	WX0 | WX240	WY0 | WY240	WM0 | WM1896	WS0 | WS984	T0 | T255	C0 | C255	R0 | R3839	R3840 | R3903	R3904 | R3967	R3968 | R4167	R5000 | R8071	D0 | D4095	16-bit + number	V, Z P0~P9
S	○	○	○	○	○	○	○	○	○	○	○	○		○
N	○	○	○	○	○	○	○	○	○	○	○	○	1~511	○
D		○	○	○	○	○	○		○	○*	○*	○		○

- When conversion control “EN” =1 or “EN↑” (P instruction) changes from 0 to 1, will convert the N successive nibbles of hexadecimal value in registers start from S into ASCII code, and store the result to low byte (high byte remain unchanged) of the registers which start from D.
- The conversion will not be performed when the value of N is 0 or greater than 511.
- The main purpose of this instruction is to convert the numerical value data, which controller has processed, to ASCII code and transmit to ASCII peripherals by communication port1 or communication port 2.

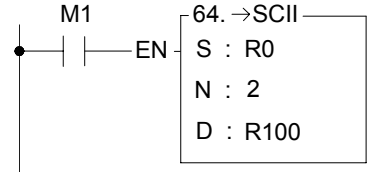
〈Example 1〉When M1 changes from OFF→ON, it converts hexadecimal value to ASCII code.



• Converts the Nibble 0 of R0 to ASCII code and stores it into R100 (High byte does not change).

R0＝0009H ➔ R100＝0039H（9）

〈Example 2〉When M1 is ON, it converts hexadecimal value to ASCII code.

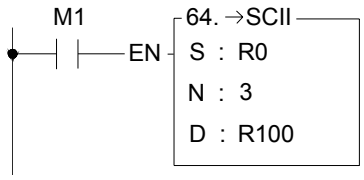


• Converts the NB0～NB1 of R0 to ASCII code and stores it into R100 ～ R101 (high bytes remain unchanged).

R0＝009AH ➔ R100＝0039H（9）
R101＝0041H（A）

FUN 64 P →ASCII	CONVERSION OF HEXADECIMAL VALUE TO ASCII CODE	FUN 64 P →ASCII

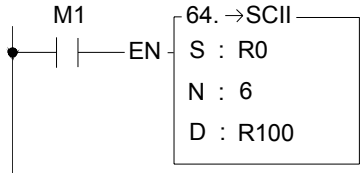
〈Example 3〉When M1 is ON, it converts hexadecimal value to ASCII code.



- Converts the NB0～NB2 of R0 to ASCII code and stores it into R100～R102.

R0＝0123H ➔ R100＝0031H（1）
R101＝0032H（2）
R102＝0033H（3）

〈Example 4〉When M1 is ON, it converts hexadecimal value to ASCII code.

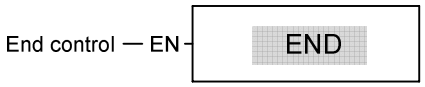


- Converts the NB0～NB5 of R0～R1 to ASCII code and stores it into R100～R105.

R0＝3456H ➔ R100＝0031H（1）
R1＝0012H R101＝0032H（2）
R102＝0033H（3）
R103＝0034H（4）
R104＝0035H（5）
R105＝0036H（6）

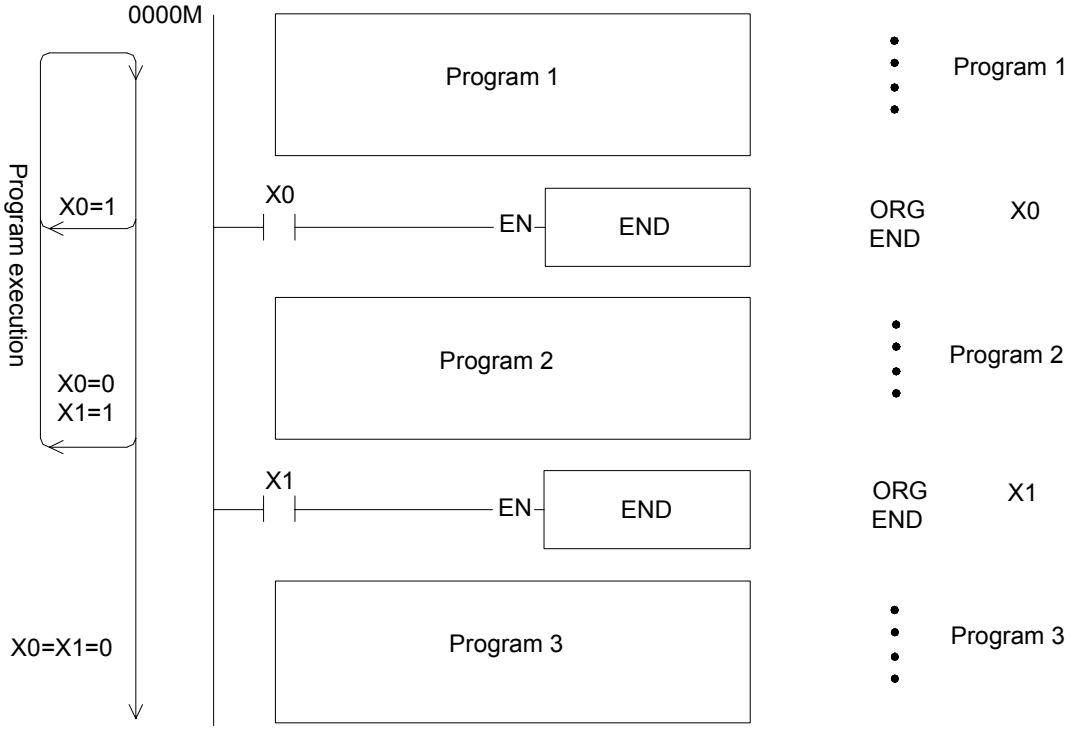
END	PROGRAM END	END

Ladder symbol



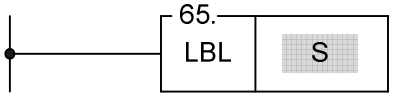
No operand

- When end control "EN" =1, this instruction is activated. Upon executing the END instruction and "EN" =1, the program flow will immediately returns to the starting point (0000M) to restart the next scan – i.e. all the programs after the END instruction will not be executed. When "EN" =0, this instruction is ignored, and programs after the END instruction will continue to be executed as the END instruction is not exist.
- This instruction may be placed more than one point within a program, and its input (end control "EN") controls the end point of program execution. It is especially useful for debugging and for testing.
- It's not necessary to put any END instructions in the main program, CPU will automatic restart from start point when reach the end of main program.



FUN 65 LBL	LABEL	FUN 65 LBL

Ladder symbol



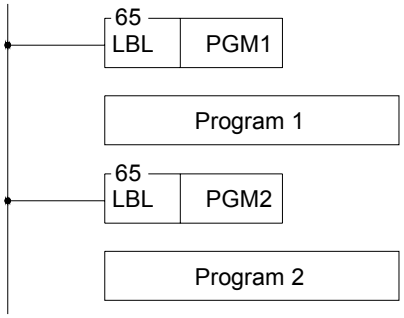
S : Alphanumeric, 1~6 characters

- This instruction is used to make a tag on certain address within a program, to provide a target address for execution of JUMP, CALL instruction and interrupt service. It also can be used for document purpose to improve the readability and interpretability of the program.
- This instruction serves only as the program address marking to provide the control of procedure flow or for remark. The instruction itself will not perform any actions; whether the program contains this instruction or not, the result of program execution will not be influenced by this instruction.
- The label name can be formed by any 1～6 alphanumeric characters and can't be duplicate in the same program. The following label names are reserved for interrupt function usage. These "reserved words", can't be used for normal program labels.

Reserved words	Description
X0+I～X15+I（INT0～INT15） X0–I～X15–I（INT0-～INT15-）	labels for external input (X0～X15) interrupt service routine.
HSC0I～HSC7I	labels for high speed counter HSC0～HSC7 interrupt service routine.
1MSI（1MS）, 2MSI（2MS）, 3MSI（3MS）, 4MSI（4MS）, 5MSI（5MS）, 10MSI（10MS）, 50MSI（50MS）, 100MSI（100MS）	Labels for 8 kinds of internal timer interrupt service routine.
HSTAI（ATMRI）	Label for High speed fixed timer interrupt service routine.
PSO0I～PSO3I	Labels for the pulse output command finished interrupt service routine.

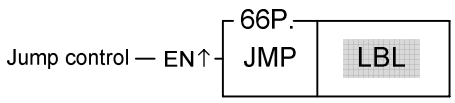
Only the interrupt service routine can use the label names listed on above table, if mistaken on using the reserved label on the normal subroutine can cause the CPU fail or unpredictable operation.

The label of following diagram illustration served only as program remarks (it is not treated as a label for call or jump target). For the application of labeling in jump control, please refer to JMP instruction for explanation. As to the labeling serves as subroutine names, please refer to CALL instruction for details.



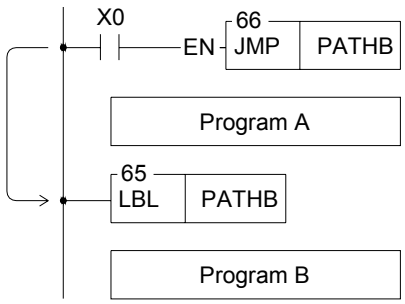
FUN 66 P JMP	JUMP	FUN 66 P JMP

Ladder symbol



LBL : The program label to be jumped

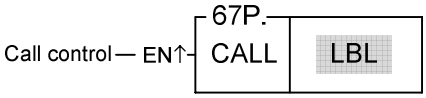
- When jump control "EN"=1 or "EN↑" (P instruction) changes from 0 to 1, controller will jump to the location behind the marked label and continuous to execute the program.
- This instruction is especially suit for the applications where some part of the program will be executed only under certain condition. This can shorter the scan time while not executes the whole program.
- This instruction allows jump backward (i.e. the address of LBL is comes before the address of JMP instruction). However, care should be taken if the jump action cause the scan time exceed the limit set by the watchdog timer, the WDT interrupt will be occurred and stop executing.
- The jump instruction allows only for jumping among main program or jumping among subroutine area, it can't jump across main/subroutine area.



• In the left diagram, when X0=1, the program will jump directly to the LBL position named PATHB and continuing to execute program B. Therefore it will skip the program A and none of the instructions of program A will be executed. The status of registers and the coils associated with program A will keep unchanged (as if there is no program section A).

FUN 67 P CALL	CALL	FUN 67 P CALL

Ladder symbol

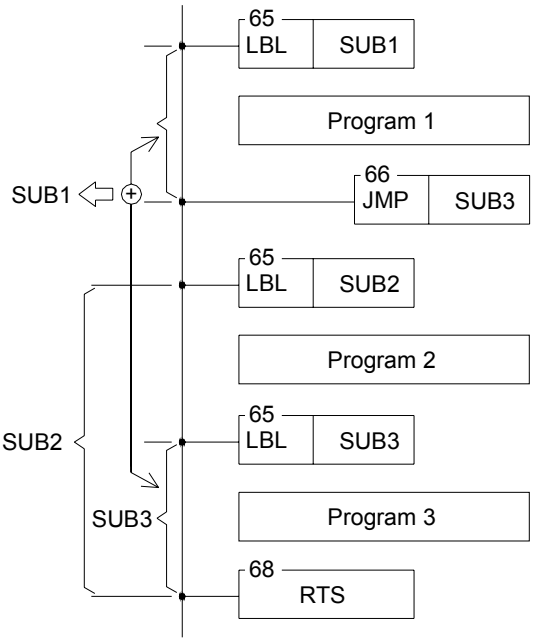


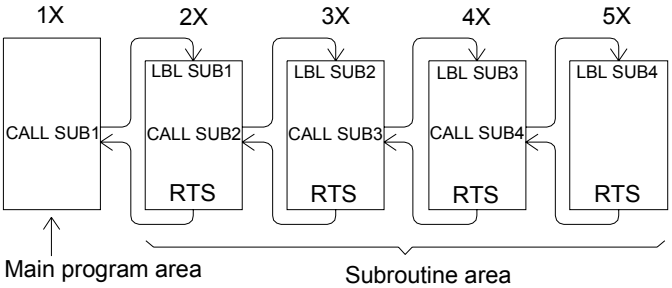
LBL : The subroutine label name to be called.

- When call control "EN"=1 or "EN↑" (P instruction) changes from 0 to 1, controller will call (perform) the subroutine bear the same label name as the one being called. When execute the subroutine, the program will execute continuous as normal program does but when the program encounter the RTS instruction then the flow of the program will return back to the address immediately after the CALL instruction.

- All the subroutines must end with one "return from subroutine instruction RTS" instruction; otherwise it will cause executing error or CPU shut down. Nevertheless, an RTS instruction can be shared by subroutines (so called as multiple entering subroutines; even though the entry points are different, they have a same returning path) as illustrated in the right diagram subroutine SUB1～3.

- When main program called a subroutine, the subroutine also can call the other subroutines (so called the nested subroutines) for up to 5 levels at the most (include the interrupt routine).

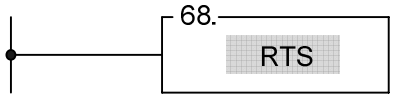




- Interrupt service programs （HSC0I～HSC7I, PSO0I～PSO3I, X0+I～X15+I / INT0～INT15, X0-I～X15-I / INT0-～INT15-, HSTAI / ATMRI, 1MS / 1MS, 2MSI / 2MS, 3MSI / 3MS, 4MSI / 4MS, 5MSI / 5MS, 10MSI / 10MS, 50MSI / 50MS, 100MSI / 100MS） are also a kind of subroutine. It is also placed in sub program area. However, the calling of interrupt service program is triggered off by the signaling of hardware to make the CPU perform the corresponding interrupt service program (which we called as the calling of the interrupt service program). The interrupt service program can also call subroutine or interrupted by other interrupts with higher priority. Since it is also a subroutine (which occupied one level), it can only call or interrupted by 4 levels of subroutine or interrupt service program. Please refer to RTI instruction for explanation.

FUN 68 RTS	RETURN FROM SUBROUTINE	FUN 68 RTS

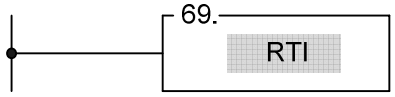
Ladder symbol



- This instruction is used to represent the end of a subroutine. Therefore it can only appear within the subroutine area. Its input side has no control signal, so there is no way to serially connect any contacts. This instruction is self sustain, and is directly connected to the power line.
- When controller encounter this instruction, it means that the execution of a subroutine is finished. Therefore it will return to the address immediately after the CALL instruction, which were previously executed and will continue to execute the program.
- If this instruction encounters any of the three flow control instructions MC, SKP, or JMP, then this instruction may not be executed (it will be regarded as not exist). If the above instructions are used in the subroutine and causing the subroutine not to execute the RTS instruction, then controller will halt the operation and set the M1933 (flow error flag) to 1. Therefore, no matter what the flow is going, it must always ensure that any subroutine must be able to execute a matched RTS instruction.
- For the usage of the RTS instruction please refer to instructions for the CALL instruction.

FUN 69 RTI	RETURN FROM INTERRUPT	FUN 69 RTI

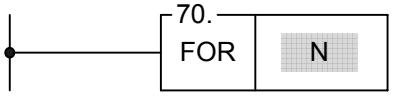
Ladder symbol



- The function of this instruction is similar to RTS. Nevertheless, RTS is used to end the execution of sub program, and RTI is used to end the execution of interrupt service program. Please refer to the explanation of RTS instruction.
- A RTI instruction can be shared by more than one interrupt service program. The usage is the same as the sharing of an RTS by many subroutines. Please refer to the explanation of CALL instruction.
- The difference between interrupts and call is that the sub program name (LBL) of a call is defined by user, and the label name and its call instruction are included in the main program or other sub program. Therefore, when controller performs the CALL instruction and the input “EN”=1, the controller will call (execute) this sub program. For the execution of interrupt service program, it is directly used with hardware signals to interrupt CPU to pause the other less important works, and then to perform the interrupt service program corresponding to the hardware signal (we call it the calling of interrupt service program). In comparing to the call instruction that need to be scanned to execute, the interrupt is a more real time in response to the event of the outside world. In addition, the interrupt service program cannot be called by label name; therefore we preserve the special “reserved words” label name to correspond to the various interrupts offered by controller (check FUN65 explanation for details). For example, the reserved word X0+I is assigned to the interrupt occurred at input point X0; as long as the sub program contains the label of X0+I, when input point X0 interrupt is occurred (X0: ↑), the controller will pause the other lower priority program and jump to the subroutine address which labeled as X0+I to execute the program immediately.
- If there is an interrupt occurred while CPU is handling the higher priority (such as hardware high speed counter interrupt) or same priority interrupt program (please refer to Chapter 9-3 for priority levels), the controller will not execute the interrupt program for this interrupt until all the higher priority programs were finished.
- If the RTI instruction cannot be reached and performed in the interrupt service routine, may cause a serious CPU shut down. Consequently, no matter how you control the flow of program, it must be assured that the RTI instruction will be executed in any interrupt service program.
- For the detailed explanation and example for the usage of interrupts, please refer to Chapter 9 for explanation.

FUN 70 FOR	FOR	FUN 70 FOR

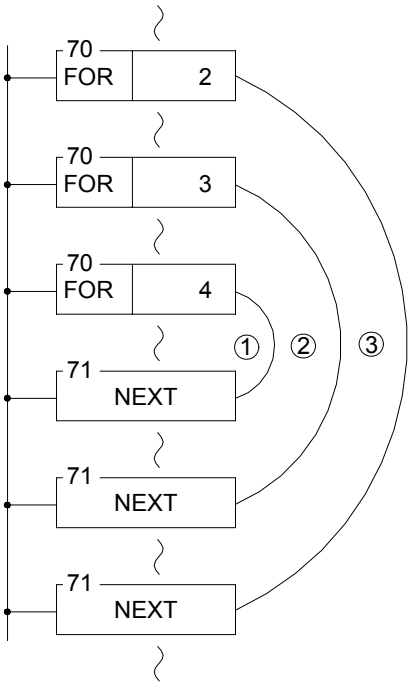
Ladder symbol



N : Number of times of loop execution

Range / Operand	WX	WY	WM	WS	TMR	CTR	HR	IR	OR	SR	ROR	DR	K
	WX0 | WX240	WY0 | WY240	WM0 | WM1896	WS0 | WS984	T0 | T255	C0 | C255	R0 | R3839	R3840 | R3903	R3904 | R3967	R3968 | R4167	R5000 | R8071	D0 | D4095	1 | 16383
N	○	○	○	○	○	○	○	○	○	○	○	○	○

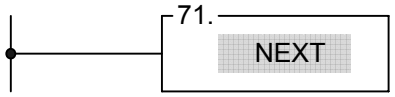
- This instruction has no input control, is connected directly to the power line, and cannot be in series with any conditions.
- The programs within the FOR and NEXT instructions form a program loop (the start of the loop program is the next instruction after FOR, and the last is the instruction before NEXT). When controller executes the FOR instruction, it first records the N value after that instruction (loop execution number), then for N times successively execution from start to last of the programs in the loop. Then it jumps out of the loop, and continues executes the instruction immediately after the NEXT instruction.
- The loop can have a nested structure, i.e. the loop includes other loops, like an onion. 1 loop is called a level, and there can be a maximum of 5 levels. The FOR and NEXT instructions must be used in pairs. The first FOR instruction and the last NEXT instruction are the outermost (first) level of a nested loop. The second FOR instruction and the second last NEXT instruction are the second level, the last FOR instruction and the first NEXT instruction form the loop's innermost level.



- In the example in the diagram at left, loop ① will be executed 4 x3 x2 = 24 times, loop ② will be executed 3 x2 = 6 times, and loop ③ will be executed 2 times.
- If there is a FOR instruction and no corresponding NEXT instruction, or the FOR and NEXT instructions in the nested loop have not been used in pairs, or the sequence of FOR and NEXT has been misplaced, then a syntax error will be generated and this program may not be executed.
- In the loop, the JMP instruction may be used to jump out of the loop. However, care must be taken that once the loop has been entered (and executed to the FOR instruction), no matter how the program flow jumps, it must be able to reach the NEXT instruction before reaching the END instruction or the bottom of the program. Otherwise WSZ-controller will halt the operation and show an error message.
- The effective range of N is 1~16383 times. Beyond this range WSZ-controller will treat it as 1. Care should be taken , if the amount of N is too large and the loop program is too big, a WDT may occur.

FUN 71 NEXT	LOOP END	FUN 71 NEXT

Ladder symbol



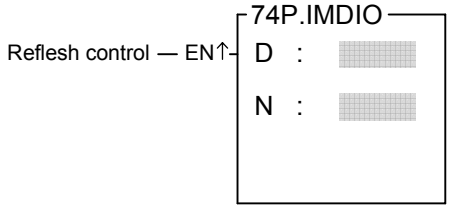
- This instruction and the FOR instruction together form a program loop. The instruction itself has no input control, is connected directly to the power line, and cannot be in series with any conditions.

- When controller has not yet entered the loop (has not yet executed to the FOR instruction, or has executed but then jumped out), but the NEXT instruction is reached, then controller will not take any action, just as if this instruction did not exist.

- For the usage of this instruction please refer to the explanations for the FOR instruction on the preceding page.

FUN 74 P IMDIO	IMMEDIATE I/O	FUN 74 P IMDIO

Ladder symbol



D : Starting number of I/O points to be refreshed

N : Number of I/O points to be refreshed

Range / Ope-rand	X	Y	K
	Xn of Main Unit.	Yn of Main Unit.	1 | 36
D	○	○	
N			○

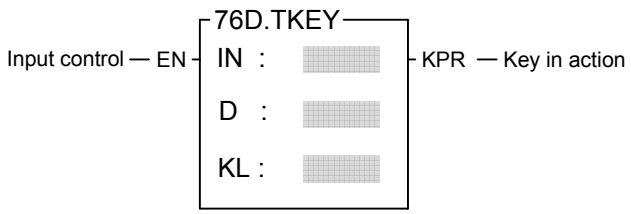
- For normal controller scan cycle, the CPU gets the entire input signals before the program is executed, and then performs the executing of program based on the fresh input signals. After finished the program execution the CPU will update all the output signals according to the result of program execution. Only after the complete scan has been finished will all the output results be transferred all at once to the output. Thus for the input event to output responses, there will be a delay of at least 1 scan time (maximum of 2 scan time). With this instruction, the input signals or output signals specified by this instruction can be immediately refresh to get the faster input to output response without the limitation imposed by the scan method.
- When refresh control "EN" =1 or "EN↑" (P instruction) changes from 0 to 1, then the status of N input points or output points (D~D+N-1) will be refreshed.
- The I/O points for WSZ-controller's immediate I/O are only limited to I/O points on the main unit. The table below shows permissible I/O numbers for 24, 32, 40 and 60 point main units:

Main-unit type / Permissible numbers	24 points	32 points	40 points	60 points
Input signals	X0～X13	X0～X19	X0～X23	X0～X35
Output signals	Y0～Y9	Y0～Y11	Y0～Y15	Y0～Y23

- If the intended refresh I/O signals of this instruction are beyond the range of I/O points specified on above table then controller will be unable to operate and the M1931 error flag will be set to 1. (For example, if in a program, D=X11, N=10, which means X11 to X20 are to be immediately retrieved. Supposing the main unit is WSZ-32MCT2, then its biggest input point is X19, and clearly X20 has already exceeded the main unit's input point number so under such case M1931 error flag will be set to 1).
- With this instruction, controller can immediately refresh input/output signals. However, the delay of the hardware or the software filter impose on the I/O signals still exist. Please pay attention on this.

FUN 76 D TKEY	DECIMAL- KEY INPUT	FUN 76 D TKEY

Ladder symbol



IN : Key input point

D : register storing key-in numerals

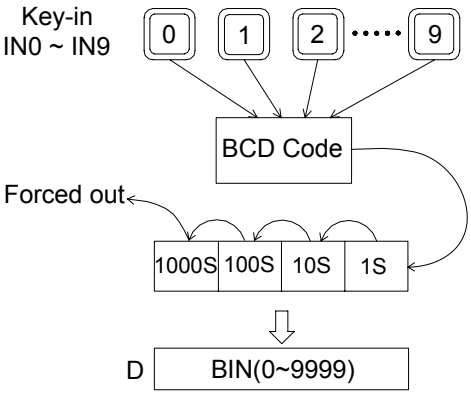
KL: starting coil to reflect the input status

D may combine with V, Z, P0~P9 to serve indirect address application.

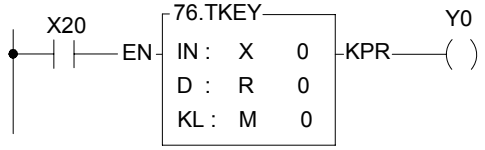
Range / Operand	X	Y	M	S	WY	WM	WS	TMR	CTR	HR	OR	SR	ROR	DR	XR
	X0 | X240	Y0 | Y240	M0 | M1896	S0 | S984	WY0 | WY240	WM0 | WM1896	WS0 | WS984	T0 | T255	C0 | C255	R0 | R3839	R3904 | R3967	R3968 | R4167	R5000 | R8071	D0 | D4095	V, Z | P0~P9
IN	○														
D					○	○	○	○	○	○	○	○*	○*	○	○
KL		○	○	○											

- This instruction has designated 10 input points IN~IN+9 (IN0~IN9) to one decimal number entry (IN->0, IN+1->1…). According to the key-in sequence (ON) of these input points, it is possible to enter 4 or 8 decimal numbers into the registers specified by D.

- When input control "EN" =1, this instruction will monitor the 10 input points starting from IN and put the corresponding number into D register while the key were depressed. It will wait until the input point has released, then monitor the next "ON" input point, and shift in the new number into D register (high digit is older than low digit) . For the 16-bit operand, D register can store up to 4 digits, and for the 32-bit operand 8 digits may be stored. When the key numbers fulfill the D register, new key-in number will kick out the oldest key number of the D register. The key-in status of the 10 input points starting from IN will be recorded on the 10 corresponding coil starting from KL. These coils will set to 1 while the corresponding key is depressed and remain unchanged even if the corresponding key is released. If other key is depressed, it will return to zero. As long as any input point is depressed (ON), then the key-in flag KPR will set to 1. Only one of IN0~IN9 key can be depressed at the same time. If more than one is pressed, then the first one is the only one taken. Below is a schematic diagram of the function with 16-bit operand.



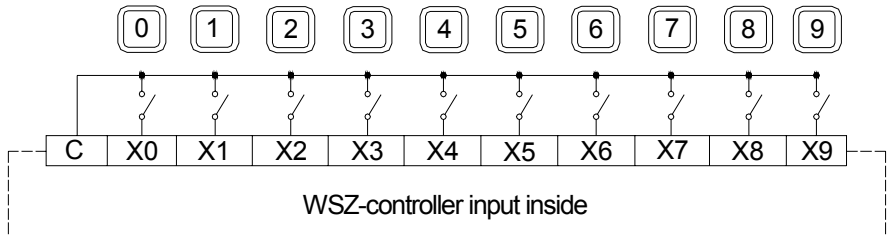
- When input control "EN" = 0, this instruction will not be executed. KPR output and KL coil status will be 0. However, the numerical values of D register will remain unchanged.



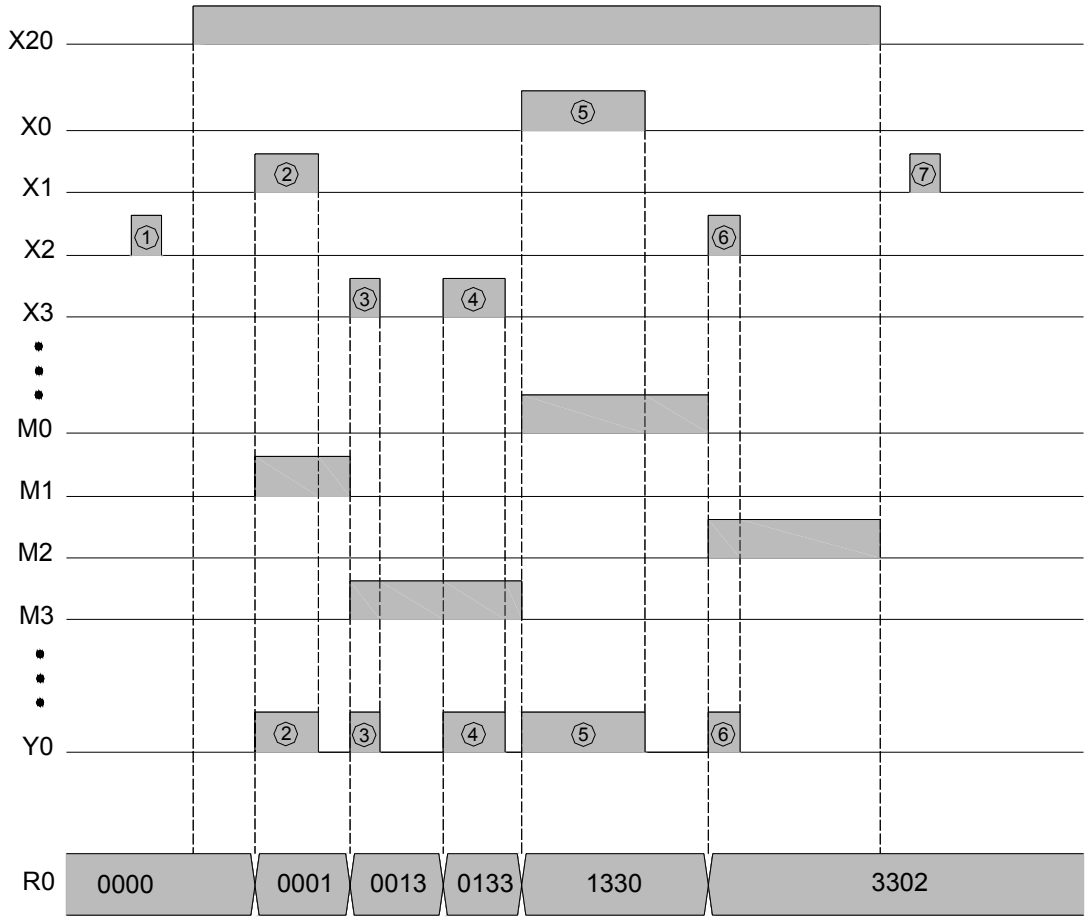
• The instruction at left represents the input point X0 with the number "0", X1 is represented by 1, ... , M0 records the action of X0, M1 records the action of X1 ... , and the input numerical values are stored in the R0 register.

FUN 76 D TKEY	DECIMAL- KEY INPUT	FUN 76 D TKEY

The following diagram is the input wiring schematic for this example:

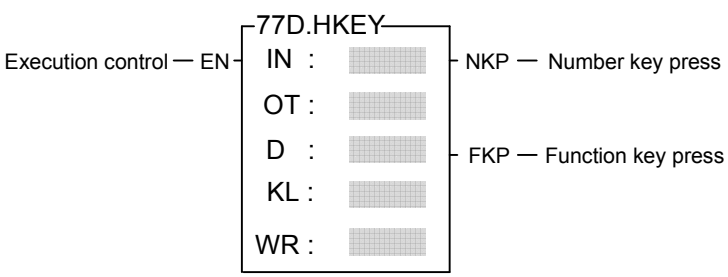


- If the X0~X3 key-in sequence follows the ①②③④⑤⑥⑦ sequence in the following diagram. At step ① and ⑦ the X20 is 0, so there was no generated key, only steps ②③④⑤⑥ are effective. Because the register can only hold 4 key numbers, the first key of these 5 steps was kick out. The key strokes 3302 of the steps ③④⑤⑥ are entered in the R0 register.



FUN 77 D HKEY	HEX-KEY INPUT	FUN 77 D HKEY

Ladder symbol



IN : Starting of digital input for key scan

OT: Starting of digital output for multiplexing key scan (4 points)

D : Register to store key-in numbers

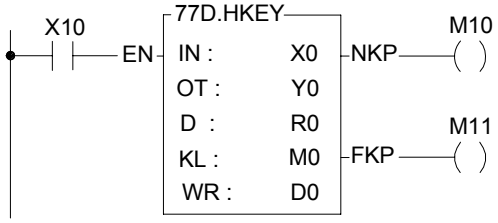
KL : Starting relay for key status

WR: Working register, it can't repeat in use

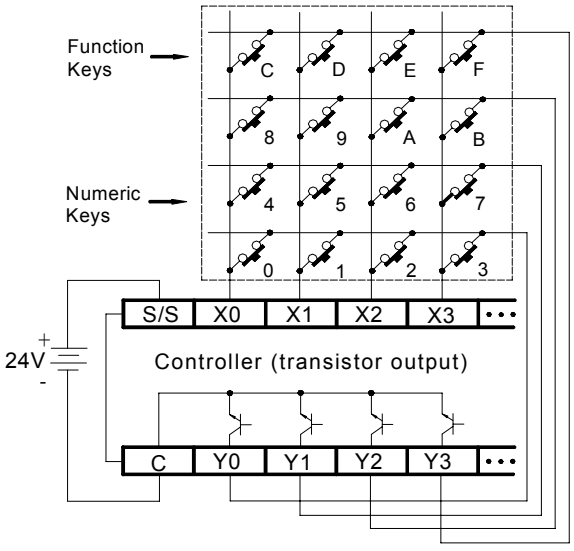
D may combine with V, Z, P0~P9 to serve indirect addressing application.

Operand \ Range	X	Y	M	S	WY	WM	WS	TMR	CTR	HR	OR	SR	ROR	DR	XR
	X0 | X240	Y0 | Y240	M0 | M1896	S0 | S984	WY0 | WY240	WM0 | WM1896	WS0 | WS984	T0 | T255	C0 | C255	R0 | R3839	R3904 | R3967	R3968 | R4167	R5000 | R8071	D0 | D4095	V, Z | P0~P9
IN	○														
OT		○													
D					○	○	○	○	○	○	○	○*	○*	○	○
KL		○	○	○											
WR										○			○*	○	

- The numeric (0~9) key function of this instruction is similar as for the TKEY instruction. The hardware connection for TKEY and HKEY is different. For TKEY instruction each key have one input point to connect, while HKEY use 4 input points and 4 output points to form a 4x4 multiplex 16 keys input. 4x4 means that there can be 16 input keys, so in addition to the 10 numeric keys, the other 6 keys can be used as function keys (just like the usual discrete input). The actions of the numeric keys and the function keys are independent and have no effect on each other.
- When execution control "EN" =1, this instruction will scan the numeric keys and function keys in the matrix formed by the 4 input points starting from IN and the 4 output points starting from OT. For the function of the numeric keys and "NKP" output please refer to the TKEY instruction. The function keys maintain the key-in status of the A~F keys in the last 6 relays specified by KL (the first 10 store the key-in status of the numeric keys). If any one of the A~F keys is depressed, FKP (FO1) will set to 1. The OT output points for this instruction must be transistor outputs.
- The biggest number for a 16-bit operand is 4 digits (9999), and for 32-bit operand is 8 digits (99999999). However, there are only 6 function keys (A~F), no matter whether it is a 16-bit or 32-bit operand.

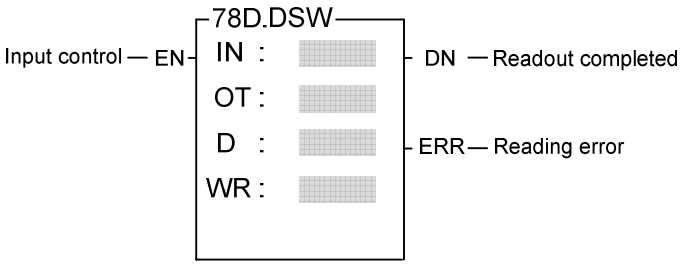


- The instruction in the diagram above uses X0~X3 and Y0~Y3 to form a multiplex key input. It can input numeric values of 8 digits and stores the results in R1R0. The input status of the function keys is stored in M10(A)~M15(F).



FUN 78 D DSW	DIGITAL SWITCH INPUT	FUN 78 D DSW

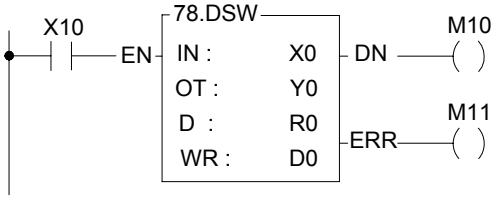
Ladder symbol



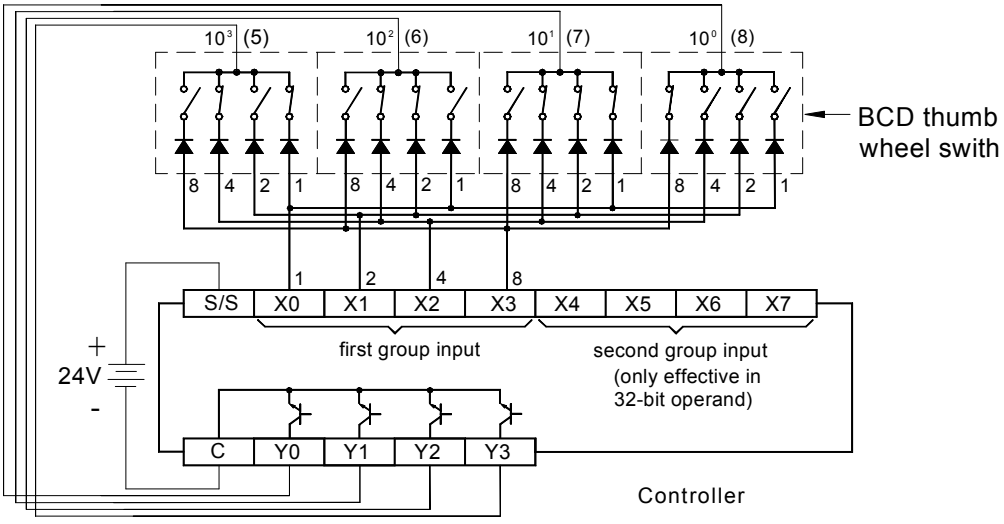
IN : Starting of input for thumb wheel switch
OT: Starting of output for multiplexing scan (4 points)
D : Register to store readout value
WR: Working register, it can't repeat in use (WR & WR+1 for 16-bit operation; WR, WR+1 & WR+2 for 32-bit operation).
D may combine with V, Z, P0~P9 to serve indirect addressing application.

Range / Operand	X	Y	WY	WM	WS	TMR	CTR	HR	OR	SR	ROR	DR	XR
	X0 | X240	Y0 | Y240	WY0 | WY240	WM0 | WM1896	WS0 | WS984	T0 | T255	C0 | C255	R0 | R3839	R3904 | R3967	R3968 | R4167	R5000 | R8071	D0 | D4095	V, Z | P0~P9
IN	○												
OT		○											
D			○	○	○	○	○	○	○	○*	○*	○	○
WR								○			○*	○	

- When input control "EN" =1, this instruction will readout one digit data from the 4 input points starting from IN (IN0~IN3). It takes 4 scans to read out a group of 4-digit BCD values (0000~9999) and store them into D register. With a 32-bit operand, each scan can get 2 digits of data by reading the additional digit from IN4-IN7 and store it in the D+1 register. Each bit of OT0~OT3 will sequentially set to 1 and get the digit data respectively into 10^0(ones), 10^1(tens), 10^2(hundreds), and 10^3(thousands). As long as EN is 1, controller will scan and read out in continuous cycles. When each complete cycle is finished (i.e. the 4 digit readout of 10^0~10^3 is completed), the readout completed flag "DN" is set to 1. However, it is only kept for one scan. If any digital readout value is not within the range of 0~9 (BCD), then reading error "ERR" will be set to 1 and the value of that group of digits will be set to 0000.
- The output points must be transistor outputs.

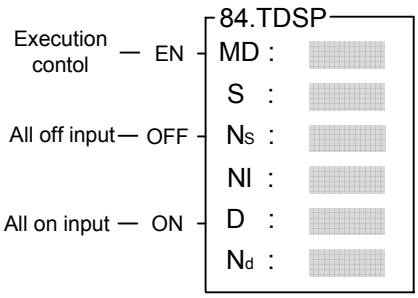


- In this example, when X10 is 1, then the numeric value of the thumb wheel switch (5678 in this example) will be read out and stored into the R0 register.
- The bits (8,4,2,1) with same digit should be connect together and series with a diode (as shown in diagram below).
- With 32-bit operand a set of similar thumb wheel switch may be added to X4~X7 (Y0~Y3 are shared with another group).



FUN 84 TDSP	FBs-7SG DISPLAY MODULE CONVENIENT COMMANDS 7/16-segment display character and number display conversion	FUN 84 TDSP

Ladder symbol



Md: Operation Mode, 0~3
S: Starting address of begin converted characters
Ns: Start of source character, 0~63
Nl: Length of character, 0~64
D: Starting address to store the converted pattern
Nd: Start pointer while storing
S operand can be combined with V, Z, P0~P9 index registers for indirect addressing

Range / Ope-rand	WX	WY	WM	WS	TMR	CTR	HR	IR	OR	SR	ROR	DR	K	XR
	WX0 | WX224	WY0 | WY240	WM0 | WM1896	WS0 | WS984	T0 | T255	C0 | C255	R0 | R3839	R3840 | R3903	R3904 | R3967	R3968 | R4167	R5000 | R8071	D0 | D4095	16/32-bit +/- number	V、Z P0~P9
Md													0~3	
S	○	○	○	○	○	○	○	○	○	○	○	○	○	○
Ns							○	○	○	○*	○*	○	0~63	
Nl							○	○	○	○*	○*	○	1~64	
D		○	○	○	○	○	○		○	○*	○*	○		
Nd							○	○	○	○*	○*	○	0~63	

- This convenient instruction is used to generate the corresponding display pattern for FBs series 7-segment or 16-segment display panel under the control of FBs-7SG1 or FBs-7SG2 modules (It is a module of non-support in WSZ series).
- When execution control “EN” =1, input “OFF” =0, and input “ON” =0, this instruction will perform the display pattern conversion, where S is the starting address storing the being converted characters, Ns is the pointer to locate the starting character, Nl tells the length of being converted characters, and D is the starting address to store the converted result, Nd is the pointer to locate the start of storing.
- There are 4 kinds of operation mode as below:

  Md=0, display pattern conversion for 16-segment display; the source character is the 8-bit ASCII code, the converted result is the 16-bit display pattern. By the control of M1990, it determines the display direction, where M1990=0, right to left display; M1990=1, left to right display

  Md=1, without leading zero display conversion for 16-segment display; the source character is the 8-bit ASCII code, the converted result is the 16-bit display pattern without leading zero.

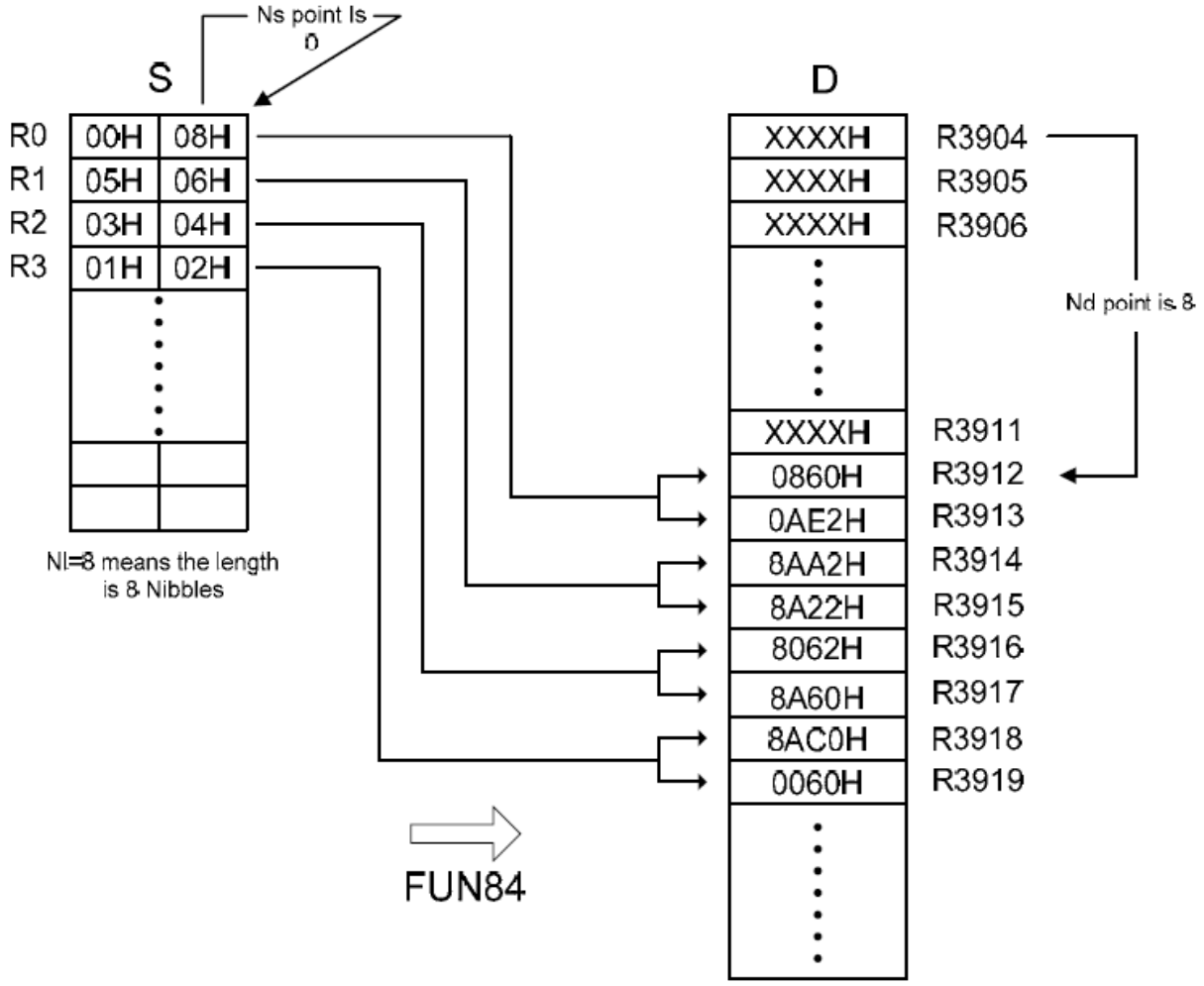
  Md=2, Non-decoded display pattern conversion for 7-segment display; the source character is the 4-bit nibble code, the converted result is the 8-bit display pattern.

  Md=3, without leading zero display conversion for 7-segment decoded display; the source character is the 4-bit nibble code, the converted result is the 4-bit display pattern without leading zero. Byte 0 or nibble 0 of S is the 1st displaying character, Byte 1 or nibble 1 of S is the 2nd displaying character, ……..
- Ns operand is the pointer to tell where the displaying character starts.
- Nl operand is the character quantity for conversion.

FUN 84 TDSP	FBs-7SG DISPLAY MODULE CONVENIENT COMMANDS 7/16-segment display character and number display conversion	FUN 84 TDSP

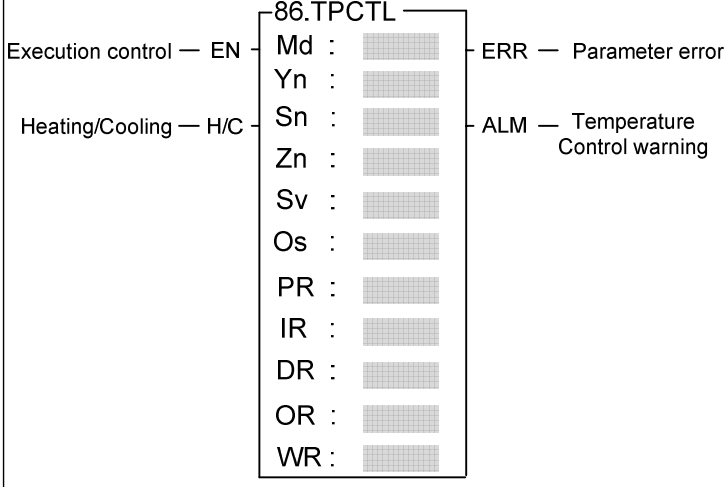
- D operand is the starting address to store the converted display pattern; while Md=0 or 1, one source character of 8-bit ASCII code needs one 16-bit location to store the result; while Md=2, one source character of 4-bit nibble code needs one 8-bit location to store it; while Md=3, one source character of 4-bit nibble code needs one 4-bit location to store it.
- Nd operand is the pointer to tell where is the start to store the converted pattern.
- When inputs “OFF” =1, “ON” =0, and “EN” =0/1, the D operand will be filled with the all OFF pattern according to the operation mode, the Nd pointer, and the quantity of Nl.
- When inputs “ON” =1, “OFF” =0/1, and “EN” =0/1, the D operand will be filled with the all ON pattern according to the operation mode, the Nd pointer, and the quantity of Nl.
- Data will be converted differently based on the selected mode. The description below is based on Example.

  In Example, Md=1; S=R0; Ns=0; Nl=8; D=R3904; and Nd=8. Data conversion is presented below.



FUN 86 TPCTL	PID TEMPERATURE CONTROL INSTRUCTION	FUN 86 TPCTL

Ladder symbol



Range / Operand	Y Y0 | Y255	HR R0 | R3839	ROR R5000 | R8071	DR D0 | D3999	K
Md					0～1
Yn	○				
Sn					0～31
Zn					1～32
Sv		○	○*	○	
Os		○	○*	○	
PR		○	○*	○	
IR		○	○*	○	
DR		○	○*	○	
OR		○	○*	○	
WR		○	○*	○	

Md: Selection of PID method
=0, Modified minimum overshoot method
=1, Universal PID method

Yn: Starting address of PID ON/OFF output; it takes Zn points.

Sn: Starting point of PID control of this instruction; Sn = 0～31.

Zn: Number of the PID control of this instruction; 1≤Sn+Zn≤32 and 1 ≤ Sn+Zn ≤ 32

Sv: Starting register of the set point: it takes Zn registers. (Unit in 0.1°C)

Os: Starting register of the in-zone offset; it takes Zn registers. (Unit in 0.1°C)

PR: Starting register of the gain (Kc): it takes Zn registers.

IR: Starting register of integral tuning constant (Ki); it takes Zn registers.

DR: Starting register of derivative tuning constant (Td); it takes Zn registers.

OR: Starting register of the PID analog output; it takes Zn registers.

WR: Starting of working register for this instruction.
It takes 9 registers and can't be repeated in using.

- By employing the temperature module and table editing method to get the current value of temperature and let it be as so called Process Variable (PV); after the calculation of software PID expression, it will respond the error with an output signal according to the setting of Set Point (SP), the error's integral and the rate of change of the process variable. Through the closed loop operation, the steady state of the process may be expected.
- Convert the output of PID calculation to be the time proportional on/off (PWM) output, and via transistor output to control the SSR for heating or cooling process; this is a good performance and very low cost solution.
- Through the analog output module (D/A module), the output of PID calculation may control the SCR or proportional valve to get more precise process control.
- For detailed functional description and usage, please refer to chapter 20 "Temperature Measurement of WSZ-Controller and PID Control" for explanation.

FUN 86 TPCTL	PID TEMPERATURE CONTROL INSTRUCTION	FUN 86 TPCTL

- PID temperature control (FUN86) instruction supports user defined starting address of temperature reading value for more flexibility in temperature control application.

  · R4003=A55AH, starting address of temperature reading value is defined by R4004

  =other values, starting address of temperature reading value is defined by temperature configuration screen

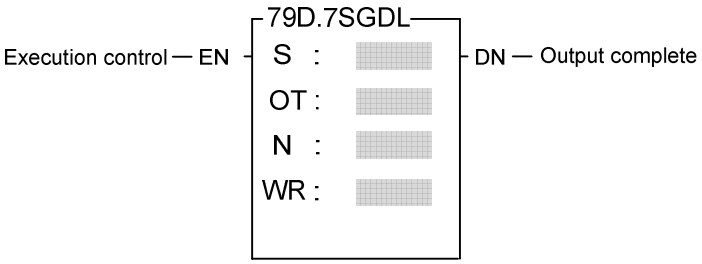
  · R4004=10000～13839, it defines R0~R3839 is the starting address of temperature reading value as the process variables for PID control

  =20000～23999, it defines D0~D3999 is the starting address of temperature reading value as the process variables for PID control

  =other values, starting address of temperature reading value is defined by temperature configuration screen

FUN 79 D 7SGDL	7-SEGMENT OUTPUT WITH LATCH	FUN 79 D 7SGDL

Ladder symbol



S : Register storing the data (BCD) to be displayed

OT : Starting number of scanning output

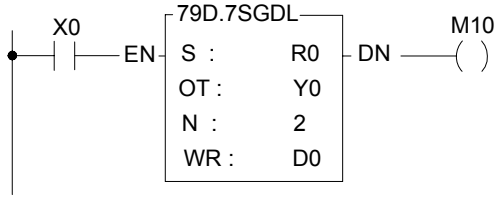
N : Specify signal output and polarity of latch.

WR : Working register, it can't repeat in use

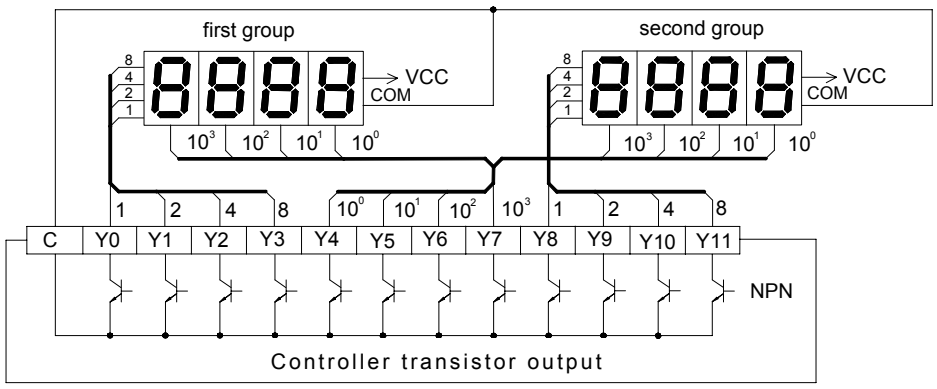
S may combine with V, Z, P0~P9 to serve indirect addressing application.

Range / Operand	Y	WX	WY	WM	WS	TMR	CTR	HR	IR	OR	SR	ROR	DR	K	XR
	Y0 | Y240	WX0 | WX240	WY0 | WY240	WM0 | WM1896	WS0 | WS984	T0 | T255	C0 | C255	R0 | R3839	R3840 | R3903	R3904 | R3967	R3968 | R4167	R5000 | R8071	D0 | D4095	16-bit number	V, Z P0~P9
S		○	○	○	○	○	○	○	○	○	○	○	○	○	○
OT	○														
N														0～3	

- When execution control "EN" =1, the 4 nibbles of the S register, from digit 0 to digit 3, are sequentially sent out to the 4 output points, OT0~OT3. While output the digit data, the latch signal of that digit (OT4 corresponds to digit 0, OT5 corresponds to digit 1, etc...) at the same time is also sent out so that the digital value will be loaded and latched into the 7-segment display respectively.
- When in D (32-bit) instruction, nibbles 0~3 from the S register, and nibbles 0~3 from the S+1 register are transferred separately to OT0~OT3 and OT8~OT11. Because they are transferred at the same time, they can use the same latch signal. 16-bit instructions do not use OT8~OT11.
- As long as "EN" remains 1, controller will execute the transfer cyclically. After each transfer of a complete group of numerical values (nibbles 0~3 or 0~7), the output completed flag "DN" will set to 1. However, it will only be kept for 1 scan.



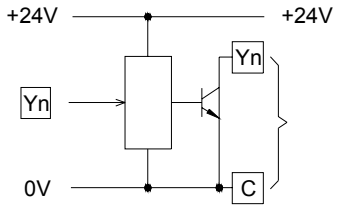
- In this example, when X0=1, the 4 nibbles of R0 will be transferred to the first group 7-segment display in the diagram below. The 4 nibbles of R1 will be transferred to the second group 7-segment display.



FUN 79 D 7SGDL	7-SEGMENT OUTPUT WITH LATCH	FUN 79 D 7SGDL

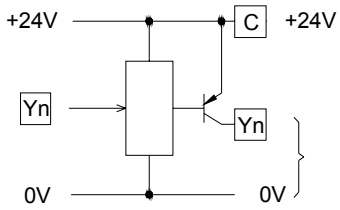
- WSZ transistor output has both a negative logic transistor output (NPN transistor - when the output status is ON, the terminal voltage of the transistor output is low), and a positive logic transistor output (PNP - when the output status is ON, the terminal voltage of the transistor output is high). Their structure is as follows:

WSZ negative logic output (NPN transistor)



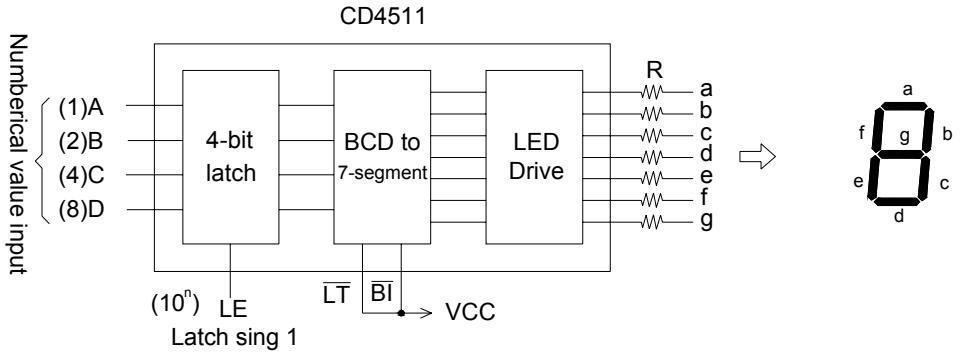
When Yn is "ON", this output voltage is low

WSZ positive logic output (PNP transistor)



When Yn is "ON", Yn's terminal voltage is high

- The data inputs (8,4,2,1) and latch signals of the 7-segment displays on the shelf for positive and negative logic are all available. For example, for numerical value "8", the positive logic input should be 1000, and the negative logic input 0111. Similarly, when the latch signal is 0, the positive logic latch permits the display numerical values to enter through the latch (i.e. be loaded). When the latch signal is 1, the numerical values in the latch are latched (maintained), and with negative logic they are not lached. The following diagram of a CD-4511 7-segment display IC is an example of a positive logic numerical value input with latch.



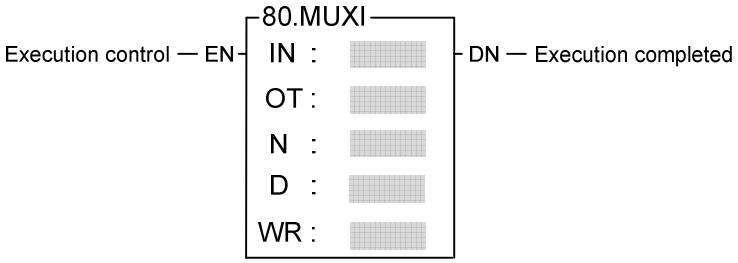
- The controller output and the 7-segment display input polarity can be positive and negative logic. Therefore, the polarities between output and input must be coordinated to get the correct result. This instruction uses N to specify the polarity relation between the controller transistor output, and the 7-segment display. The table below shows all the possibility.

Numerical value input (8~1)	Latch signal (10^0-10^3)	Value of N
Same	Same	0
	Different	1
Different	Same	2
	Different	3

- In the diagram above, CD4511 is used as an example. If use NPN output, the data input polarity is different to controller, and its latch input polarity is the same as controller, so N value should chosen as 2.

FUN 80 MUXI	MULTIPLEX INPUT	FUN 80 MUXI

Ladder symbol



IN : Multiplex input point number

OT: Multiplex output point number (must be transistor output point)

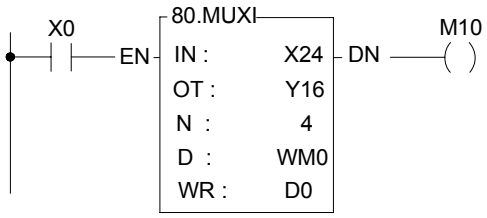
N : Multiplex input lines (2~8)

D : Register for storing results

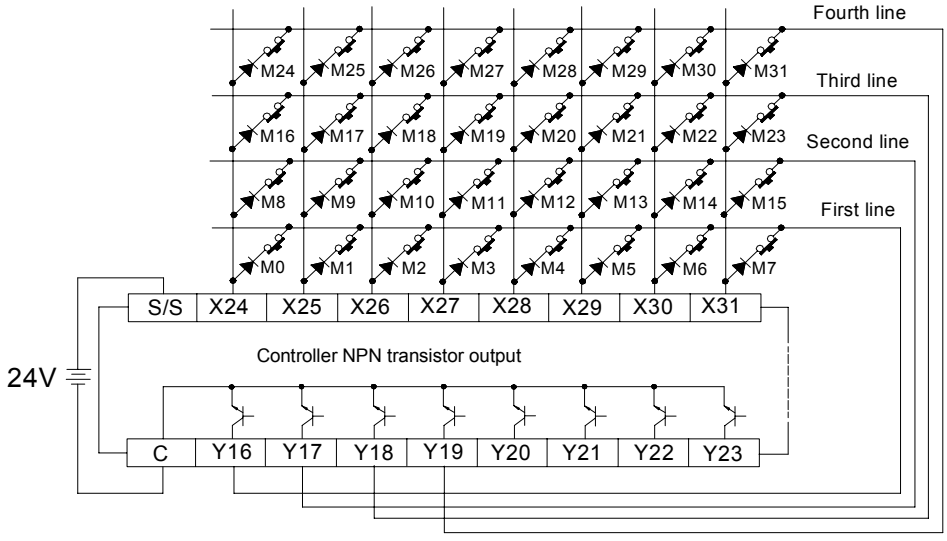
D may combine with V, Z, P0~P9 to serve indirect address application.

Operand \ Range	X	Y	WY	WM	WS	TMR	CTR	HR	OR	SR	ROR	DR	K	XR
	X0 | X240	Y0 | Y240	WY0 | WY240	WM0 | WM1896	WS0 | WS984	T0 | T255	C0 | C255	R0 | R3839	R3904 | R3967	R3968 | R4167	R5000 | R8071	D0 | D4095	2 | 8	V, Z | P0~P9
IN	○													
OT		○												
N													○	
D			○	○	○	○	○	○	○	○*	○*	○		○

- This instruction uses the multiplex method to read out N lines of input status from 8 consecutive input points (IN0~IN7) starting from the input point specified by IN. With this method we can obtain 8×N input status, but only need to use 8 input points and N output points.
- The multiplex scanning method goes through N output points starting from the OT output point. Each scan one of the N bits will set to 1 and the corresponding line will be selected. OT0 responsible for first line, while OT1 responsible for second line, etc. Until it read all the N lines the 8×N status that has been read out is then stored into the register starting at D, and the execution completed flag "DN" is set as 1 (but is only kept for one scanning period).
- With every scan, this instruction retrieves a line for 8 inputs status, so N lines require N scan cycles before they can be completed.

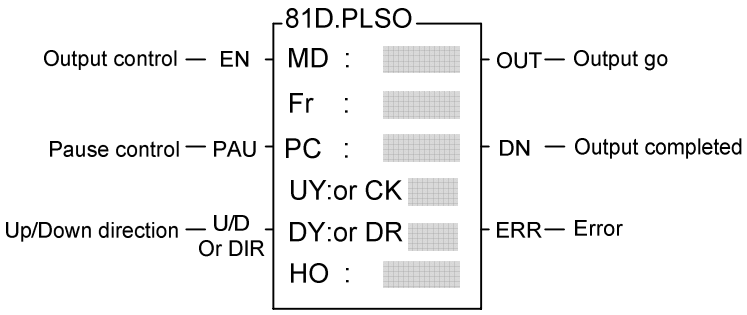


- This example retrieves 4 lines×8 points of input, 32 points status in all. They are stored into the 32-bit register of DWM0 (M0~M31).



FUN 81 D PLSO	PULSE OUTPUT	FUN 81 D PLSO

Ladder symbol



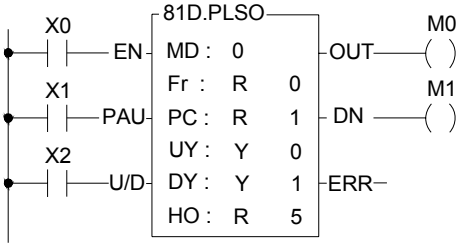
MD : Output mode selection
Fr : Pulse frequency
PC : Output pulse count
UY : Up pulse output point (MD=0).
DY : Down pulse output point (MD=0).
HO : Cumulative output pulse register. (Can be not assigned).
CK : Pulse output point (MD=1).
DR : Up/Down output point (MD=1).
DIR: 1- up; 0- down.

Range / Ope-rand	Y	WX	WY	WM	WS	TMR	CTR	HR	OR	SR	ROR	DR	K
	Yn of Main Unit	WX0 | WX240	WY0 | WY240	WM0 | WM1896	WS0 | WS984	T0 | T255	C0 | C255	R0 | R3839	R3904 | R3967	R3968 | R4167	R5000 | R8071	D0 | D4095	16-bit + number
MD													0～1
Fr		○	○	○	○	○	○	○	○	○	○	○	8～2000
PC		○	○	○	○	○	○	○	○	○	○	○	○
UY，CK	○												
DY，DR	○												
HO			○	○	○	○	○	○	○	○*	○*	○	

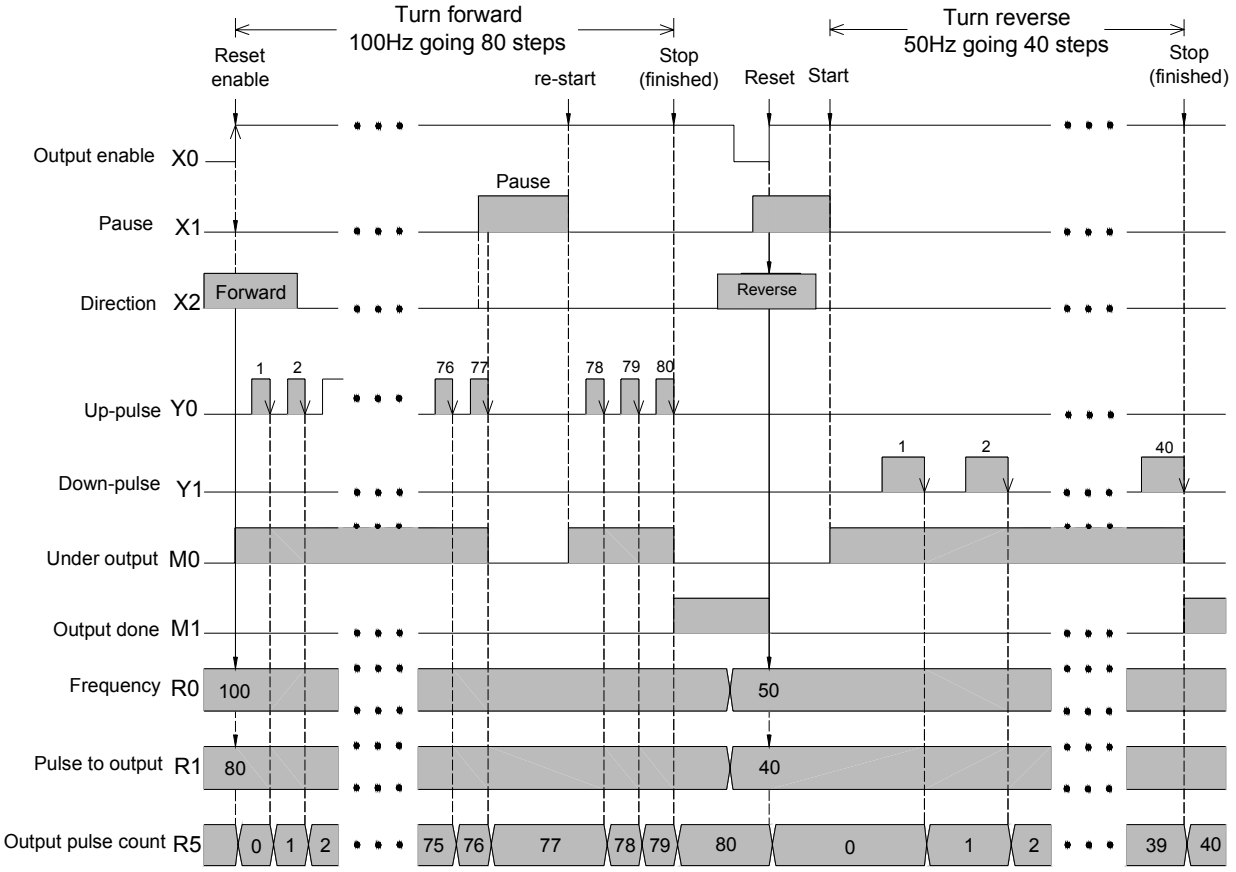
- When MD=0, this instruction performs the pulse output control as following:
- Whenever the output control “EN” changes from 0 to 1, it first performs the reset action, which is to clear the output flag “OUT” and “DN” as well as the pulse out register HO to be 0. It gets the pulse frequency and output pulse count values, and reads status of up and down direction “U/D”, so as to determine the direction to be upward or downward. As the reset finished, this instruction will check the input status of pause output “PAU”. No action will be taken if the pause output is 1 (output pause). If the PAU is 0, it will start to output the ON/OFF pulse with 50% duty at the frequency Fr to the UY(U/D=1) or DY(U/D=0) point. It will increment the value of HO register each time when a pulse is output, and will stop the output when HO register’s pulse count is equal to or greater than the cumulative pulse count of PC register and set the output complete flag “DN” to 1. During the time when output pulse is transmitting the output transmitting flag “OUT” will be set to 1, otherwise it will be 0.
- Once it starts to transmit pulse, the output control “EN” should kept to 1. If it is changed to 0, it will stop the pulse sending (output point become OFF) and the flag “OUT” changes back to 0, but the other status or data will keep unchanged. However, when its “EN” changes again from 0 to 1, it will lead to a reset action and treat as a new start; the entire procedure will be restarted again.
- If you want to pause the pulse output and not to restart the entire procedure, the ‘pause output’ “PAU” input can be used to pause it. When “PAU” =1, this instruction will pause the pulse transmitting (output point is OFF, flag “OUT” change back to 0 and the other status or data keeps unchanged). As it waits until the “PAU” changes back from 1 to 0, this instruction will return to the status before it is paused and continues the pulse transmitting output.
- During the pulse transmission, this instruction will keep monitoring the value of pulse frequency Fr and output pulse count PC. Therefore, as long as the pulse output is not finished, it may allow the changing of the pulse frequency and pulse count. However, the up/down direction “U/D” status will be got only once when it takes the reset action (“EN” changes from 0 to 1), and will keep the status until the pulse output completed or another reset occur. That is to say, except that at the very moment of reset, the change of “U/D” does not influence the operation of this instruction.
- The main purpose of this instruction is to drive the stepping motor with the UY (upward) and DY (downward) two directional pulses control, so as to help you control the forward or reverse rotating of stepping motor. Nevertheless, if you need only single direction revolving, you can assign just one of the UY or DY (which will save one output point), and leaving the other output blank. In such case, the instruction will ignore the up/down input status of “U/D”, and the output pulse will send to the output point you assigned.

FUN 81 D PLSO	PULSE OUTPUT	FUN 81 D PLSO

- When MD=1, the pulse output will reflect on the control output DIR (pulse direction. DIR=1, up; DIR=0, down) and CK (pulse output).
- This instruction can only be used once, and UY (CK) and DY (DR) must be transistor output point on the controller main unit.
- The effective range of output pulse count PC for 16 bits operand is 0～32767. For the 32 bits operand(D instruction), it is 0～2147483647. If the PC value = 0, it is treated as infinite pulse count, and this instruction will transmit pulses without end with HO value and “DN” flag set at 0 all the time. The effective range of pulse frequency (Fr) is 8～2000. If the value PC or Fr exceeds the range, this instruction will not be carried out and the error flag “ERR” will set to 1.

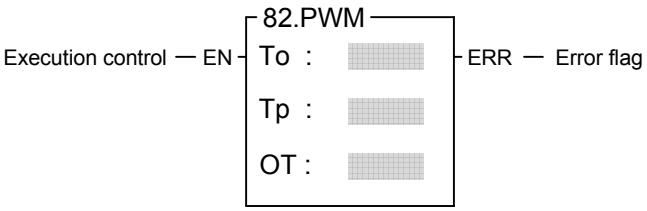


- In this example, the program controls the stepping motor to drive forward for 80 pulses (steps) at the speed of 100Hz first, and then makes it turn reverse for 40 pulses the speed of 50Hz. Make sure that the up/down direction, frequency Fr and the pulse count PC must be set before the reset take action (“EN” changes from 0→1).



FUN 82 PWM	PULSE WIDTH MODULATION	FUN 82 PWM

Ladder symbol



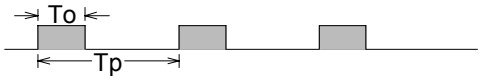
To : Pulse ON width
(0~32767ms)

Tp : Pulse period
(1~32676ms)

OT: Pulse output point

Range / Operand	Y	WX	WY	WM	WS	TMR	CTR	HR	IR	OR	SR	ROR	DR	K
	Yn of main unit	WX0 | WX240	WY0 | WY240	WM0 | WM1896	WS0 | WS984	T0 | T255	C0 | C255	R0 | R3839	R3840 | R3903	R3904 | R3967	R3968 | R4167	R5000 | R8071	D0 | D4095	0 | 32767
To		○	○	○	○	○	○	○	○	○	○	○	○	○
Tp		○	○	○	○	○	○	○	○	○	○	○	○	○
OT	○													

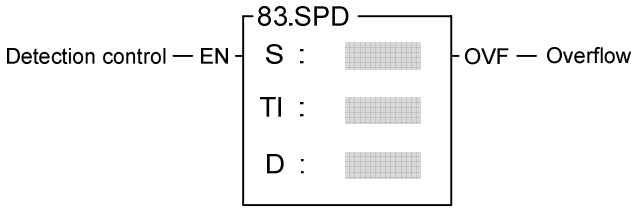
- When execution control "EN" =1, will send the pulse to output point OT with the "ON" state for To ms and period as Tp. OT must be a transistor output point on the main unit. When "EN" is 0, the output point will be OFF.



- The units for Tp and To are ms, resolution is 1 ms. The minimum value for To is 0 (under such case the output point OT will always be OFF), and its maximum value is the same as Tp (under such case the output point OT will always be on). If To > Tp there will be an error, this instruction will not be carried out, and the error flag "ERR" will set to 1.
- This instruction can only be used once.

FUN 83 SPD	SPEED DETECTION	FUN 83 SPD

Ladder symbol

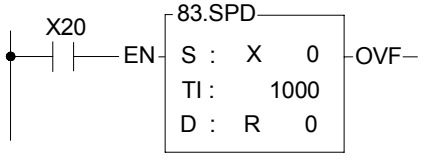


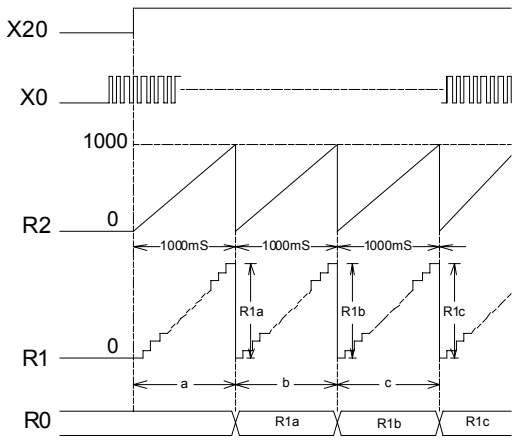
S : Pulse input point for speed detection
TI : Sampling duration
(units in ms)
D : Register storing results

Range / Operand	X	WX	WY	WM	WS	TMR	CTR	HR	IR	OR	SR	ROR	DR	K
	X0 | X7	WX0 | WX240	WY0 | WY240	WM0 | WM1896	WS0 | WS984	T0 | T255	C0 | C255	R0 | R3839	R3840 | R3903	R3904 | R3967	R3968 | R4167	R5000 | R8071	D0 | D4095	1 | 32767
S	○													
TI		○	○	○	○	○	○	○	○	○	○	○	○	○
D			○	○	○	○	○	○	○	○	○*	○*	○	

- This instruction uses the interrupt feature of the 8 high speed input points (X0~X7) on the WSZ main unit to detect the frequency of the input signal. Within a specific sampling time (TI), it will calculate the input pulse count for S input point, and indirectly find the revolution speed of rotating devices (such as motors).
- While use this instruction to detect the rotating speed of devices, The application should design to generate more pulse per revolution in order to get better result, but the sum of input frequency of all detected signals should under 5KHz, otherwise the WDT may occur.
- The D register for storing results uses 3 successive 16-bit registers starting from D (D0~D2). Besides D0 which is used to store counting results, D1 and D2 are used to store current counting values and sampling duration.
- When detection control "EN" =1, it starts to calculate the pulse count for the S input point, which can be shown in D1 register. Meanwhile the sampling timer (D2) is switched on and keeps counting until the value of D2 is reach to the sampling period (TI). The final counted value is stored into the D0 register, and then a new counting cycle is started again. The sampling counting will go on repeating until "EN" = 0.
- Because D0 only has 16 bits, so the maximum count is 32767. If the sampling period is too long or the input pulse is too fast then the counted value may exceed 32767, under that case the overflow flag will set to 1, and the counting action will stop.
- Because the sampling period TI is already known and if every revolution of attached rotating device produces "n" pulses, then the following equation can be used to get the revolution.

$$\text{speed}: N = \frac{(D0) \times 60}{n \times TI} \times 10^3 \quad \text{(rpm)}$$



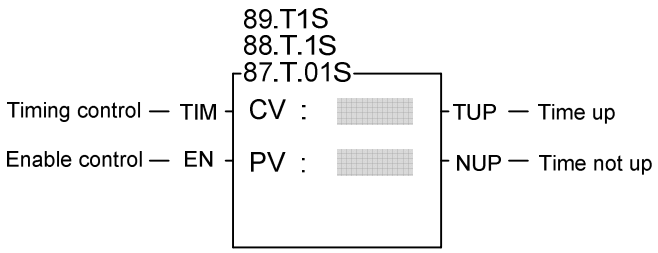


- In the above example, if every revolution of the rotating device produces 20 pulses (n = 20), and the R0 value is 200, then the revolution per minute speed "N" is as follows.

$$\text{follows}: N = \frac{(200) \times 60}{60 \times 1000} \times 10^3 = 200 \text{ rpm}$$

FUN89 D T1S FUN88 D T.1S FUN87 D T.01S	ACCUMULATIVE TIMER	FUN89 D T1S FUN88 D T.1S FUN87 D T.01S

Ladder symbol



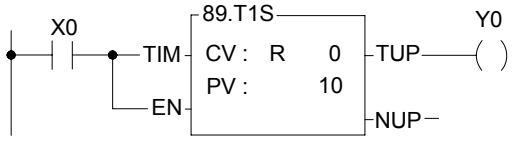
CV : Register storing elapse time (current value)

PV : Preset value of timer

Range / Operand	WX	WY	WM	WS	TMR	CTR	HR	IR	OR	SR	ROR	DR	K
	WX0 | WX240	WY0 | WY240	WM0 | WM1896	WS0 | WS984	T0 | T255	C0 | C199	R0 | R3839	R3840 | R3903	R3904 | R3967	R3968 | R4167	R5000 | R8071	D0 | D4095	0～32767
CV		○	○	○	○	○	○	○	○	○*	○*	○	
PV	○	○	○	○	○	○	○	○	○	○	○	○	○

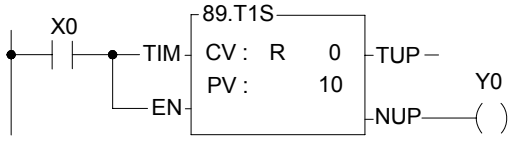
- The operation for this instruction is the same as that for the basic timer (T0~T255), except that the basic timer only has a "timing control" input - when its input is 1 it starts timing, and when input is 0 it get clear. Every time the input changes, it starts timing again and is unable to accumulate. Timing with this instruction is only permissible when enable control "EN" =1. With this instruction, when timing control "TIM" is 1, it is the same as a basic timer, but when "TIM" is 0, it does not clear, but keeps the current value. If the timer needs to clear, then change enable control "EN" to 0. When timing control "TIM" is once again to be 1, it will continue to accumulate from the previous value when the timer last paused. In addition, this instruction also has two outputs, "Time up TUP" (when time up it is 1, usually it is 0) and "Time not up" (usually it is 1, when time is up it is 0). Users can utilize input and output combinations to produce timers with various different functions. For example:

- On delay energizing timer:



- This timer's output (Y0 in this example) is normally not energized. When this timer's input control (X0 in this example) is activated (ON), only after delay by 10 sec will output Y0 become energized (ON).

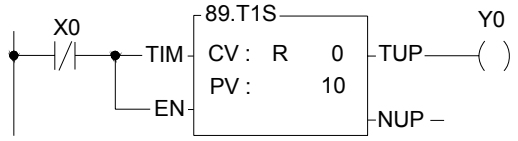
- On delay de-energizing timer:



- The output Y0 of this timer is usually energized. When this timer's input control X0 is on, only after delay by 10 sec will the output become de-energized (OFF).

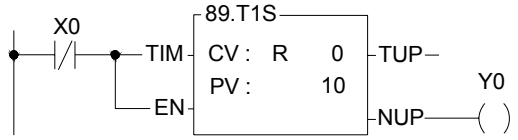
FUN89 D T1S FUN88 D T.1S FUN87 D T.01S	ACCUMULATIVE TIMER	FUN89 D T1S FUN88 D T.1S FUN87 D T.01S

- Off delay energizing timer:



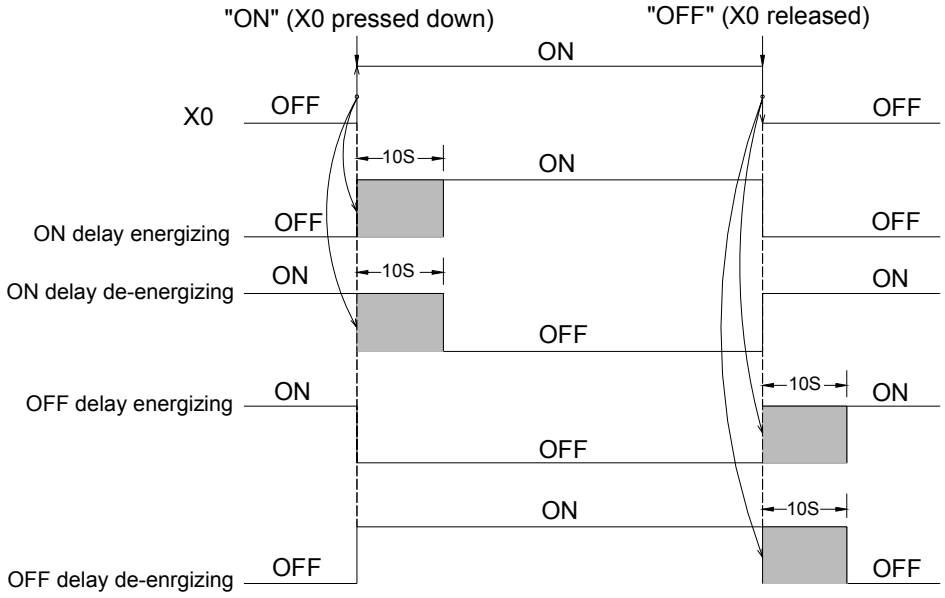
- This timer's output Y0 is usually de-energized. When this timer's input control X0 is off, only after delay by 10 sec will output Y0 become energized (ON).

- Off delay de-energizing timer:



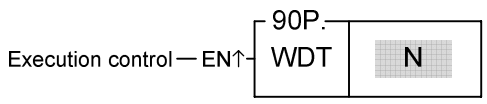
- This timer's output Y0 is usually energized. When this timer's timing control X0 is off, only after delay by 10 sec will output Y0 become de-energized (OFF).

- The diagram below shows the relation on input and output for the above 4 kinds of timers.



FUN 90 P WDT	WATCHDOG TIMER	FUN 90 P WDT

Ladder symbol

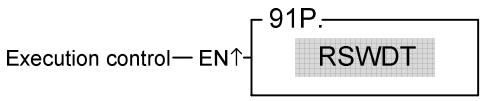


N : The watchdog time. The range of N is 5~120, unit in 10ms (i.e. 50ms~1.2 sec) .

- When execution control "EN" =1 or "EN↑" (P instruction) changes from 0 to 1, will set the watchdog time to Nx10ms. If the scan time exceeds this preset time, controller will shut down and not execute the application program.
- The WDT feature is designed mainly as a safety consideration from the system view for the application. For example, if the CPU of controller is suddenly damaged, and there is no way to execute the program or refresh I/O, then after the WDT time expired, the WDT will automatically switch off all the I/Os, so as to ensure safety. In certain applications, if the scan time is too long, it may cause safety problems or problems of non-conformance with control requirements. This instruction can used to establish the limitation of the scan time that you require.
- Once the WDT time has been set it will always be kept, and there is no need to set it again on each scan. Therefore, in practice this instruction should use the P instruction.
- Default WDT time is 0.25 sec.
- For the operation principles of WDT please refer to the RSWDT (FUN 91) instruction.

FUN 91 P RSWDT	RESET WATCHDOG TIMER	FUN 91 P RSWDT

Ladder symbol



This instruction has no operand.

- When execution control "EN" =1 or "EN↑" (P instruction) changes from 0 to 1, the WDT timer will be reset (i.e. WDT will start timing again from 0).

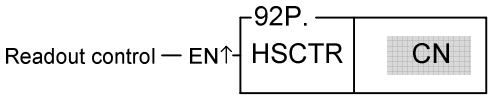
- The functions of WDT have already been described in FUN90 (WDT instruction).
  The operation principles of watch dog timer are as follows:

  The watchdog timer is normally implemented by a hardware one-shot timer (it can not be software, otherwise if CPU fails, the timer becomes ineffective, and safeguards are quite impossible). "One-shot" means that after triggered the timer once, the timing value will immediately be reset to 0 and timing will restart. If WDT has begun timing, and never triggered it again, then the WDT timing value will continue accumulating until it reach the preset value of N, at that time WDT will be activated, and controller will be shut down. If trigger the WDT once every time before the WDT time N has been reached, then WDT will never be activated. Controller can use this feature to ensure the safety of the system. Each time when controller enters into system housekeeping after finished the program scanning and I/O refresh, it will usually trigger WDT once, so if the system functions normally and scan time does not exceed WDT time then WDT is never activated. However, if CPU is damaged and unable to trigger WDT, or the scan time is too long, then there will not be enough time to trigger WDT within the period N, WDT will be activated and will shut off controller.

- In some applications, when you set the WDT time (FUN90) to desire, the scan time of your program in certain situations may temporarily exceed the preset time of WDT. This situation can be anticipated and allowed for, and you naturally do not wish controller to shut down for this reason. You can use this instruction to trigger WDT once and avoid the activation of WDT. This is the main purpose of this instruction.

FUN 92 P HSCTR	HARDWARE HIGH SPEED COUNTER CURRENT VALUE (CV) ACCESS	FUN 92 P HSCTR

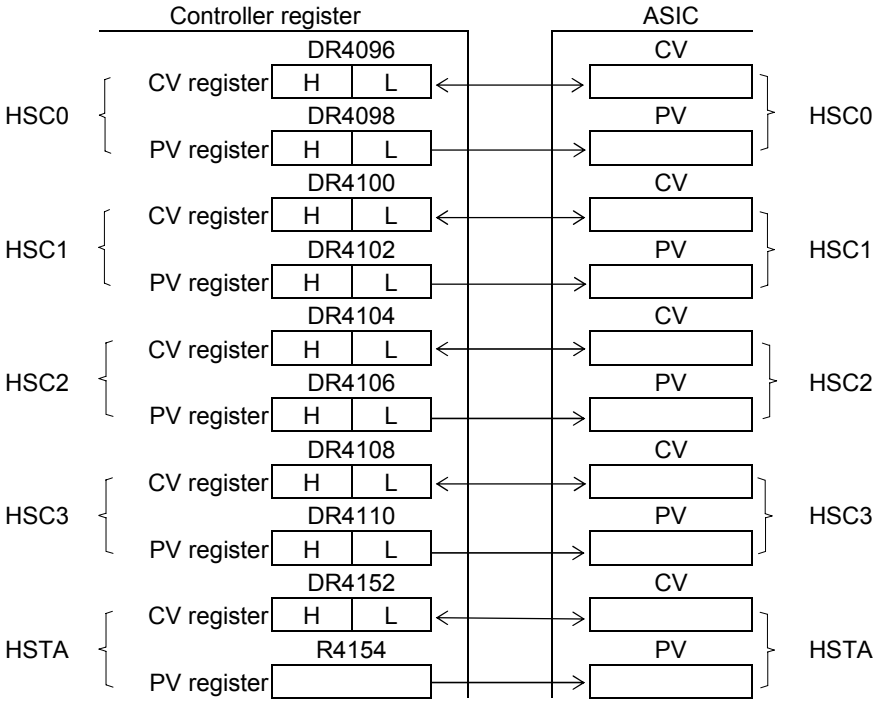
Ladder symbol



CN : Hardware high speed counter number

0: HSC0 or HST0
1: HSC1 or HST1
2: HSC2 or HST2
3: HSC3 or HST3
4: HSTA

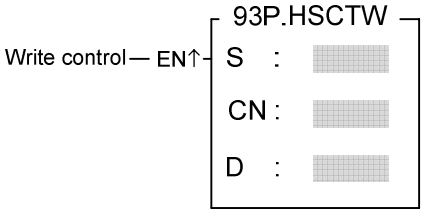
- The HSC0～HSC3 counters of WSZ are 4 sets of 32-bit high speed counter with the variety counting modes such as up/down pulse, pulse-direction, AB-phase. All the 4 high speed counters are built in the ASIC hardware and could perform count, compare, and send interrupt independently without the intervention of the CPU. In contrast to the software high speed counters HSC4～HSC7, which employ interrupt method to request for CPU processing, hence if there are many counting signals or the counting frequency is high, the controller performance (scanning speed) will be degraded dramatically. Since the current values CV of HSC0 ～HSC3 are built in the internal hardware circuits of ASIC, the user control program (ladder diagram) cannot retrieve them directly from ASIC. Therefore, it must employ this instruction to get the CV value from hardware HSC and put it into the register which control program can access. The following is the arrangement of CV, PV in ASIC and their corresponding CV, PV registers of controller for HSC0~HSC3.



- When readout control “EN” =1 or “EN↑” (P instruction) changes from 0 to 1, will gets the CV value of HSC designated by CN from ASIC and puts into the HSC corresponding CV register (i.e. the CV of HSC0 will be read and put into DR4096 or the CV of HSC1 will be read and put into DR4100).
- Although the PV within ASIC has a corresponding PV register in CPU, but it is not necessary to access it (actually it can’t be) for that the PV value within ASIC comes from the PV register in CPU.
- HSTA is a timer, which use 0.1ms as its time base. The content of CV represents elapse time counting at 0.1ms tick.
- For detailed applications, please refer to Chapter 10 “WSZ-Controller High-Speed Counter and Timer”.

FUN 93 P HSCTW	HARDWARE HIGH SPEED COUNTER CURRENT VALUE AND PRESET VALUE WRITING	FUN 93 P HSCTW

Ladder symbol

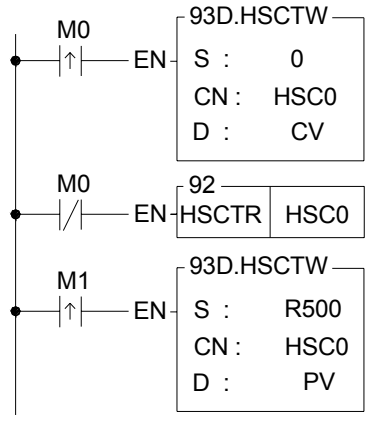


S : The source data for writing

CN : Hardware high speed counter to be written
- 0: HSC0 or HST0
- 1: HSC1 or HST1
- 2: HSC2 or HST2
- 3: HSC3 or HST3
- 4: HSTA

D : Write target (0 represents CV, 1 represents PV)

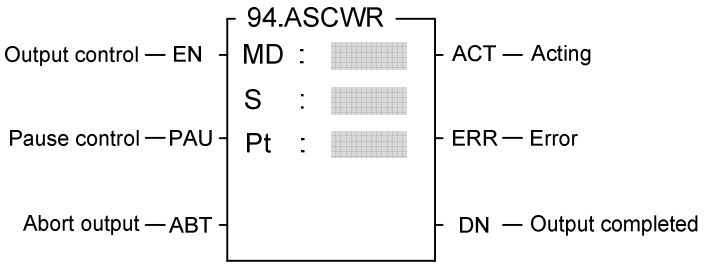
- Please refer first to FUN92 for the relation between the CV or PV value of HSC0～HSC3 and HSTA within ASIC and their corresponding CV and PV registers in CPU.
- When write control “EN”=1 or “EN↑” (P instruction) changes from 0 to 1, it writes the content of CV or PV register of high speed counter designed by CN of CPU, to the corresponding CV or PV of HSC within ASIC.
- It is quit often to set the PV value for most application program. When the count value reaches the preset value, the counter will send out interrupt signal immediately. By way of the interrupt service program, you can implement different kinds of precision counting or positioning control.
- When there is an interrupt of power supply for WSZ-controller, the values of current value registers CV of HSC0～HSC3 within ASIC will be read out and wrote into the HSC0～HSC3 CV registers (with power retentive function) of CPU automatically. When power comes up, these CV values will be restored to ASIC. However, if your application demands that when power is on, the values should be cleared to 0 or begin counting from a certain value, then you have to use this instruction to write in the CV value for HSC in ASIC.
- When a non-zero value is written into the PV register of HSTA, it will cause the HSTAI interrupt subroutine to be executed for every PV x 0.1ms.
- For detailed applications, please refer Chapter 10 “WSZ-Controller High-Speed Counter and Timer”.



- As the program in the left diagram, when M0 changes from 0→1, it clears the current value of HSC0 to 0, and writes into ASIC hardware through FUN93.
- When M0 is 0, it reads out the current counting value.
- When M1 changes from 0→1, it moves DR500 to DR4098, and writes the preset value into ASIC hardware through FUN93.
- Whenever the current value equals to the DR500, The HSC0I interrupt sub program will be executed.

FUN 94 ASCWR	ASCII FILE WRITE	FUN 94 ASCWR

Ladder symbol



MD: Output mode
=0, output to communication port1.
others, reserved for future usage.

S : Starting register of file data.

Pt : Starting working register for this instruction instance. It taken up 8 registers and can't be reused in other part of program.

Operand \ Range	HR	ROR	DR	K
	R0 | R3839	R5000 | R8071	D0 | D4095	0 | 1
MD				○
S	○	○	○	
Pt	○	○*	○	

- When MD=0 and output control "EN" changes from 0 to 1, it transmits the ASCII data which starting from S to the communication port 1 (Port1), until reach end of file.
- S file data can be edited with the programming software WinProladder (please refer to the explanation of chapter 14 "The Application of ASCII file output function"). If necessary the user can also edit the ASCII file directly by change the value of data registers. However, the edited data must be follow the ASCII file format (the details described in chapter 14), otherwise, this instruction will halt the transmission and set the error flag "ERR" to 1. If the entire file is correctly and successfully transmitted, then the output is completed and "DN" is set to 1.
- The control input of this instruction is of positive edge triggered. Once "EN" changes from 0 to 1 then this instruction starts the execution, until finished the transmission of the entire file then the execution is completed. During the transmission, the action flag "ACT" will be kept at 1 all the time. Only when output pause, error, or abort occurs, will it change back to 0.
- This instruction can be repeatedly used, but only one will be executed (transmit data) at any certain time. It is the obligation of user to make sure the right execution sequence.
- While this instruction is in execution, if the pause "PAU" is 1, this instruction will pause the transmission of file data. It will resume transmission when the pause "PAU" backs to 0.
- While this instruction is in execution, if the abort "ABT" is 1, this instruction will abandon the transmission of file data, and then it is able to take next instruction for execution.
- This instruction can only be used for the communication port1.
- For detail applications, please refer to chapter 14 "The Application of ASCII file output function".

FUN 94 ASCWR	ASCII FILE WRITE	FUN 94 ASCWR

- Interface signals:

  M1927: This signal is control by CPU, it is applied in ASCWR MD:0

  : ON, it represents that the RTS (connect to the CTS of controller) of the printer is “False”.
  I.e. the printer is not ready or abnormal.

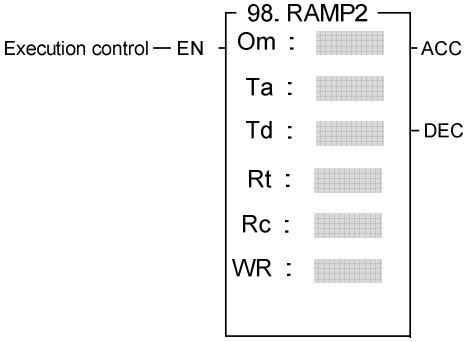
  : OFF, it represents that the RTS of the Printer is “True”; Printer is ready.

  Note: Using the M1927 associates with timer can detect if the printer is abnormal or not.

  R4146: The setting of communication parameters

FUN98 RAMP2	TRACKING TYPE RAMP FUNCTION FOR D/A OUTPUT	FUN98 RAMP2

Ladder symbol



Om: Maximum output; range from 0~65535

Ta: The acceleration time for the output from 0 up to maximum; Range from 0~65000, unit is in ms

Td: The deceleration time for the output from maximum down to 0; Range from 0~65000, unit is in ms

Rt: Register of target output; Range from 0~65535

Rc: Register of current output, it is used for analog output

WR: Starting address of working registers, it needs 4 registers

Range / Ope-rand	HR R0 | R3839	OR R3904 | R3967	ROR R5000 | R8071	DR D0 | D4095	K 16bit
Om	○		○	○	0~65535
Ta	○		○	○	0~65000
Td	○		○	○	0~65000
Rt	○		○	○	
Rc	○	○	○	○	
WR	○		○*	○	

- When execution control "EN"=0, current output value (Rc) will be 0 immediately; the output indicators ACC=0 and DEC=0.
- When execution control "EN"=1, this instruction being executed; it will output current value (Rc) first, and then compare the target output value (Rt) with current output value (Rc) every scan; if the target output value is greater than current output value, the current output will be increased according to the rate, which is decided by the settings of acceleration time (Ta) and maximum output (Om), till current output value is equal to the target output value (ACC=1 during this time); if the target output value is less than current output value, the current output will be decreased according to the rate, which is decided by the settings of deceleration time (Td) and maximum output (Om), till current output value is equal to the target output value (DEC=1 during this time).
- If the setting value of target output (Rt) is greater than maximum output (Om), the output value will be clamped by the maximum value.
- It can have smooth activity for acceleration and deceleration control via the execution of this instruction by using current output value (Rc) for analog output (R3904~R3967).
- The setting value of target output (Rt) needs to stay two scan times at least for proper operation.
- It needs 4 registers for working, they can not be repeated in use。
- This instruction is for positive value operation, but it also can have negative output by short and easy application program for help. Please see example 2.

FUN98 RAMP2	TRACKING TYPE RAMP FUNCTION FOR D/A OUTPUT	FUN98 RAMP2

Example 1：Positive output for ACC/DEC control

D10 ：Setting of maximum output, it is 16383

D0 ：The acceleration time for the output from 0 up to maximum, it is 30000ms

D1 ：The deceleration time for the output from maximum down to 0, it is 20000ms

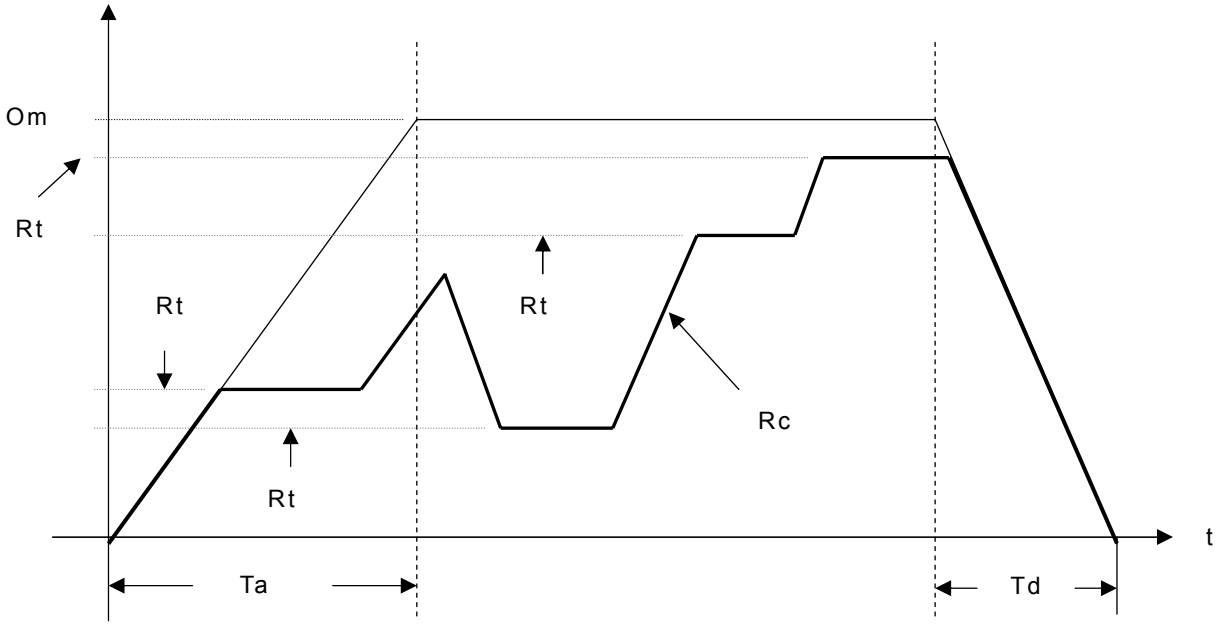
D100： Setting of target output value, it is 8192

R3904：Register of current output, it is used for analog output

D1000~D1003：Working registers

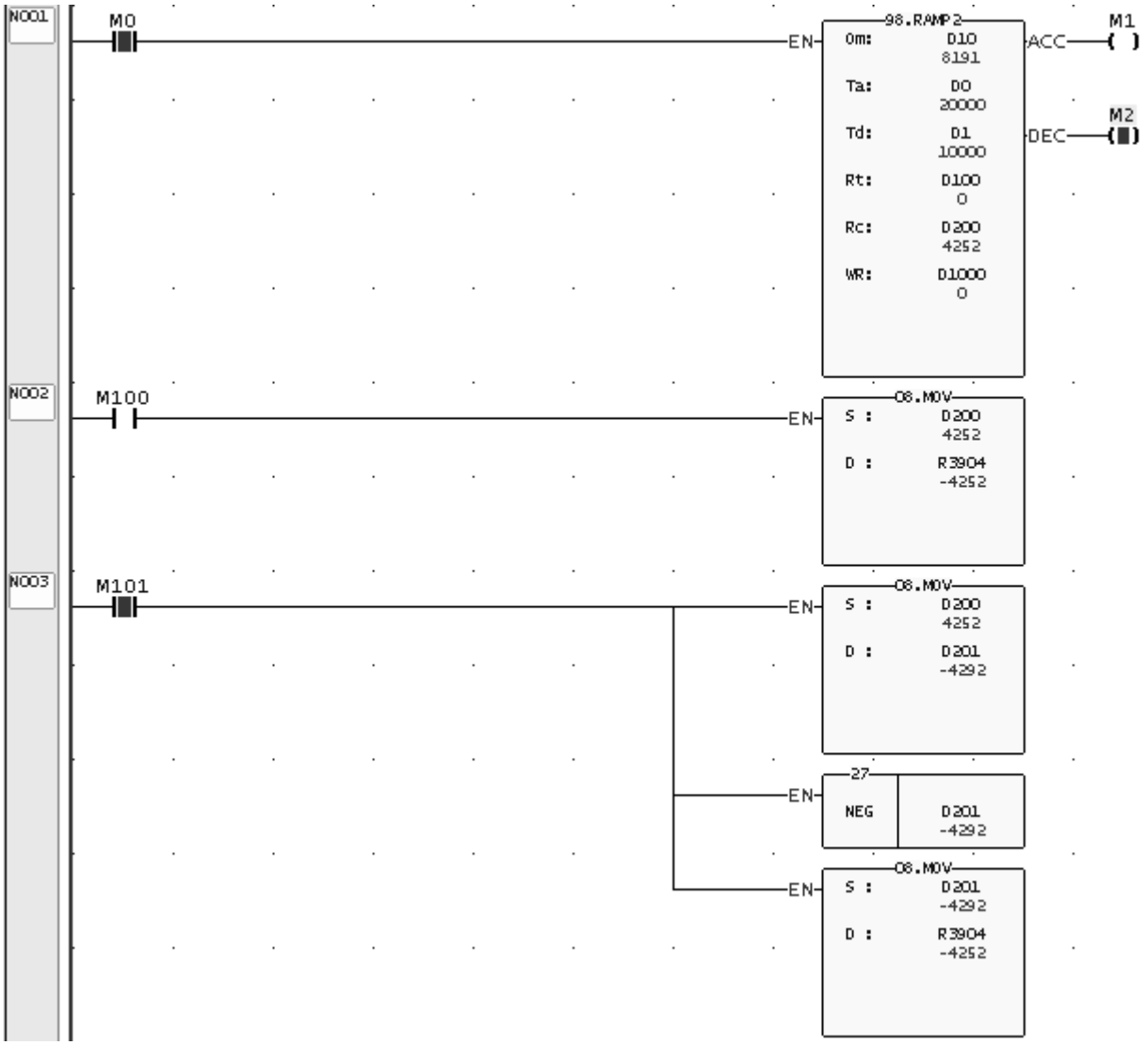
Description：When M0=0, current output value is 0 immediately (No ramp).

When M0=1, it will output the value of R3904 first; and then compare the target output value (D100) with current output value (R3904) every scan; if D100＞R3904, the current output value of R3904 will be increased according to the rate of 16383/30000 (Om=16383, Ta=30000), till R3904=D100 (ACC=1 during this time); if D100＜R3904, the current output value of R3904 will be decreased according to the rate of 16383/20000 (Om=16383, Td=20000), till R3904=D100 (DEC=1 during this time).



FUN98 RAMP2	TRACKING TYPE RAMP FUNCTION FOR D/A OUTPUT	FUN98 RAMP2

Example 2：Both positive and negative output for ACC/DEC control



D10 ：Setting of maximum output, it is 8191

D0 ：The acceleration time for the output from 0 up to maximum, it is 20000ms

D1 ：The deceleration time for the output from maximum down to 0, it is 10000ms

D100： Setting of target output value, it is 0

D200： Register of current output, it is used for analog output

D1000~D1003：Working registers

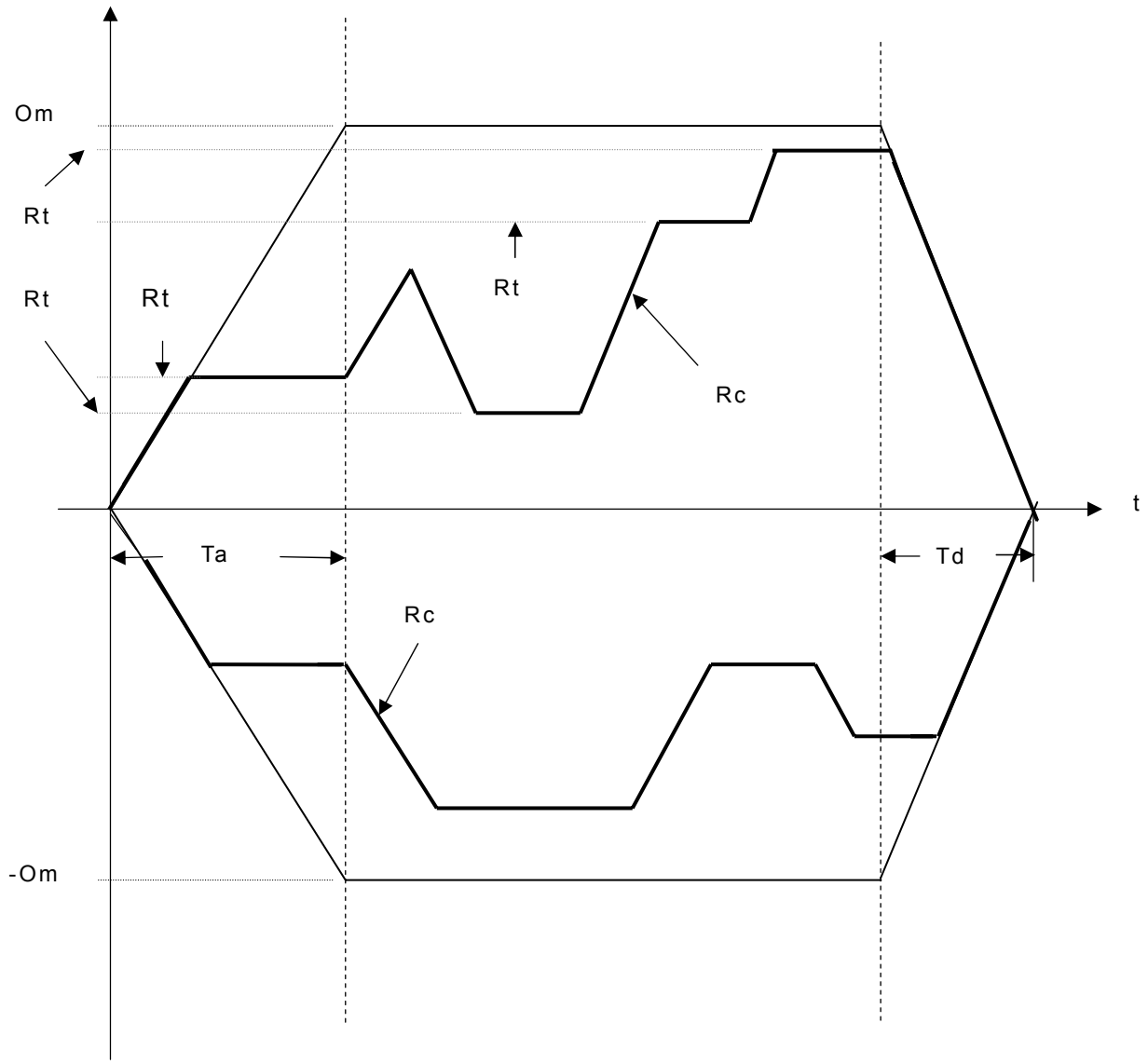
Description：When M0=0, current output value is 0 immediately (No ramp).

When M0=1, it will output the value of D200 first; and then compare the target output value (D100) with current output value (D200) every scan; if D100＞D200, the current output value of D200 will be increased according to the rate of 8191/20000 (Om=8191, Ta=20000), till D200=D100 (ACC=1 during this time); if D100＜D200, the current output value of D200 will be decreased according to the rate of 8191/10000 (Om=8191, Td=10000), till D200=D100 (DEC=1 during this time).

M100=1, positive output control; M101=1, negative output control.

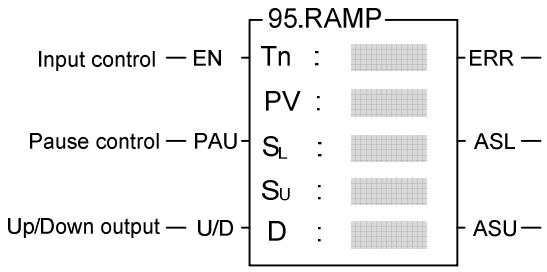
The target output (D100) is always positive value from 0~65535.

FUN98 RAMP2	TRACKING TYPE RAMP FUNCTION FOR D/A OUTPUT	FUN98 RAMP2



FUN 95 RAMP	RAMP FUNCTION FOR D/A OUTPUT	FUN 95 RAMP

Ladder symbol



Tn : Timer for ramp function
PV : Preset value of ramp timer (the unit is 0.01 sec) or the increment / decrement value of every 0.01 sec
S_L : Lower limit value (ramp floor value of up or ramp ceiling value of down).
S_U : Upper limit value (ramp floor value of down or ramp ceiling value of up).
D : Register storing current ramping value.
D+1 : Working register
S_U, S_L could be positive or negative value when incorporate with AO module application.

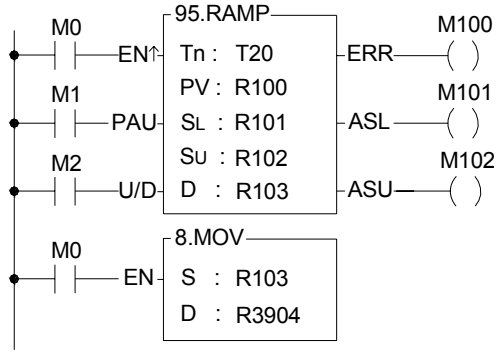
Operand \ Range	WX	WY	WM	WS	TMR	CTR	HR	IR	OR	SR	ROR	DR	K
	WX0 | WX240	WY0 | WY240	WM0 | WM1896	WS0 | WS984	T0 | T255	C0 | C255	R0 | R3839	R3840 | R3903	R3904 | R3967	R3968 | R4167	R5000 | R8071	D0 | D4095	16-bit +/- number
Tn					○								
PV	○	○	○	○	○	○	○	○	○	○	○	○	○
S_L	○	○	○	○	○	○	○	○	○	○	○	○	○
S_U	○	○	○	○	○	○	○	○	○	○	○	○	○
D		○	○	○	○	○	○		○	○*	○*	○	

Description

- Tn must be a 0.01 sec time base timer and never used in other part of program.
- When M1974=0, PV is the preset value of ramp timer. Its unit is 10ms (0.01 sec).
- When M1974=1, PV is the ramp value to increment / decrement every 10ms.
- When input control “EN” changes from 0 to 1, it first reset the timer Tn to 0.
  When “U/D”=1 it will load the value of SL to register D. And when M1974 =0 it will be increased by $S_U–S_L$ / PV every 0.01 sec or when M1974 =1 it will increase by PV every 0.01 sec. When the D value reaches the S_U value the output “ASU” =1.
  When “U/D”=0 it will load the value of S_U to register D. When M1974 =0 it will be decreased by $S_U–S_L$ / PV every 0.01 sec or when M1974 = 1 it will be decreased by PV every 0.01 sec. When the D value reaches the S_L value the output “ASL” =1.
- The ramping direction(U/D) is determined at the time when input control “EN” changes from 0 to 1. After the output D start to ramp, the change of U/D is no effect.
- If it is required to pause the ramping action, it must let the input control “PAU” =1; when “PAU”=0, and the ramping action is not completed, it will continue to complete the ramping action.
- The value of S_U must be larger than S_L, otherwise the ramp function will not be performed, and the output “ERR” will set to 1.
- This instruction use the register D to store the output ramping value; if the application use the D/A module to send the speed command, then speed command can be derived from the RAMP function to get a more smooth movement.
- In addition to use register D to store the ramping value, this instruction also used the register D+1 to act as internal working register; therefore the other part of program can not use the register D+1.

FUN 95 RAMP	RAMP FUNCTION FOR D/A OUTPUT	FUN 95 RAMP

Program example



Move the ramping value to AO output register R3904.

T20: Ramp timer (timer with 0.01 sec time base)

R100: preset value of ramp timer (M1974=0, the unit is 0.01 sec).
ramp value to increment / decrement every 10ms (M1974=1)

R101: Lower limit value. (Ramp floor value of up or ramp ceiling value of down)

R102: Upper limit value. (Ramp floor value of down or ramp ceiling value of up)

R103: Register storing current ramp value.

R104: Working register

- If M1974=0, When input control M0 changes from 0→1, it first reset the timer T20 to 0. If M2=1, it will load the R101 (lower limit) value into the R103, and it will increase the output with fixed value (R102-R101 / R100) for every 0.01 second and stores it to register R103. When the T20 timer going up to the preset value R100, the output value equals to R102, and the output M102 will set to 1. If M2=0, will load the R102 (upper limit) value into the R103, and it will decrease the output amount with fixed ratio (R102-R101 / R100) for every 0.01 second and store it to register R103. The T20 timer going up to the preset value R100, the output value equals to R101, and the output M101 will set to 1.
- M1=1, pause the ramping action.
- The value of R102 must be greater than R101, otherwise the ramp action will not be performed, and the output M100 will set to 1.

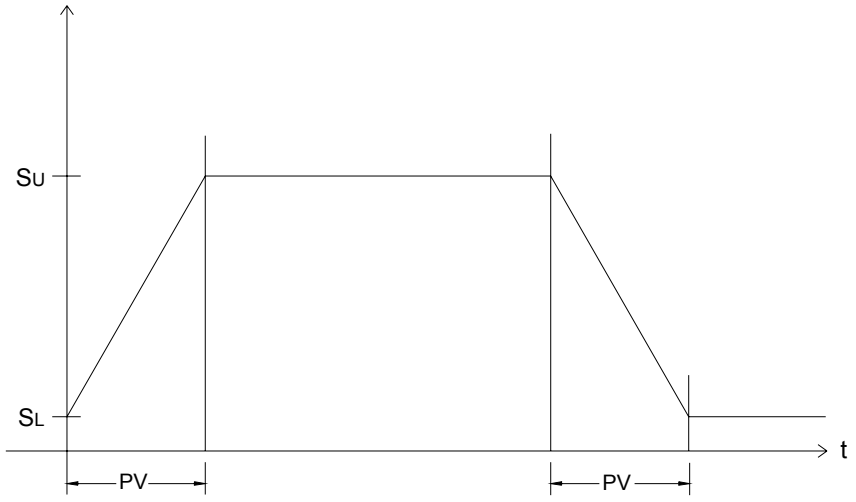


Table Instructions

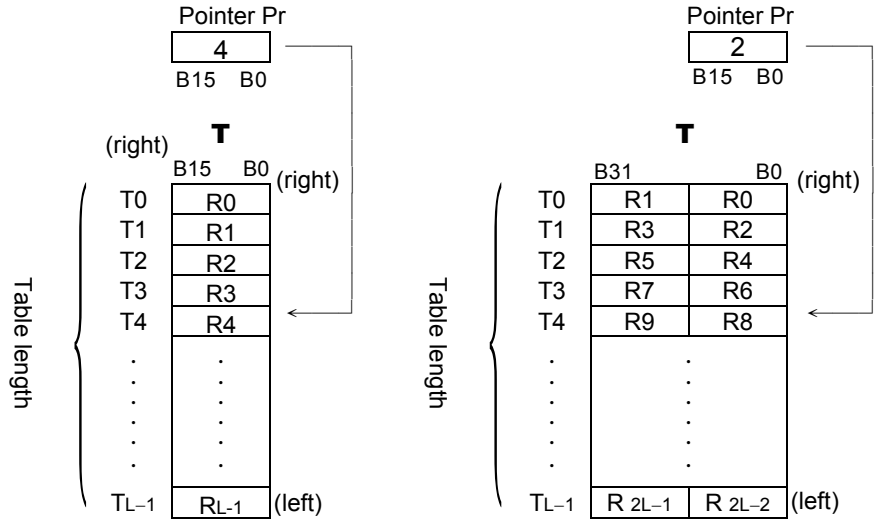
Fun No.	Functionality	Fun No.	Functionality
100	Register to table data move	107	Table fill
101	Table to register data move	108	Table shift
102	Table to table data move	109	Table rotate
103	Block table move	110	Queue
104	Block table swap	111	Stack
105	Register to table search	112	Block compare
106	Table to table compare	113	Data Sort

- A table consists of 2 or more consecutive registers (16 or 32 bits). The number of registers that comprise the table is called the table length (L). The operation object of the table instructions always takes the register as unit (i.e. 16 or 32 bits data).

- The operation of table instructions are used mostly for data processing such as move, copy, compare, search etc, between tables and registers, or between tables. These instructions are convenient for application.

- Among the table instructions, most instructions use a pointer to specify which register within a table will be the target of operation. The pointer for both 16 and 32-bit table instructions will always be a 16-bit register. The effective range of the pointer is 0 to L-1, which corresponds to registers T_0 to T_{L-1} (a total of L registers). The table shown below is a schematic diagram for 16-bit and 32-bit tables.

- Among the table operations, shift left/right, rotate left/right operations include a movement direction. The direction toward the higher register is called left, while the direction toward the lower register is called right, as shown in the diagram below.

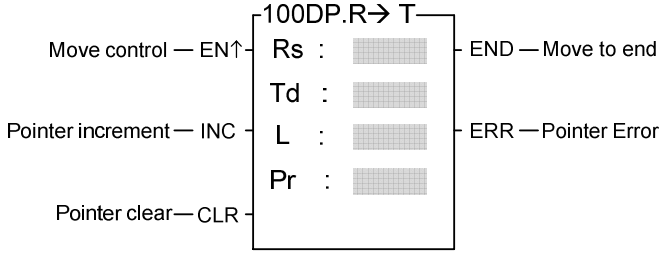


16-bit table

32-bit table

FUN100 D P R→T	REGISTER TO TABLE MOVE	FUN100 D P R→T

Ladder symbol



Rs : Source data can be constant or register.

Td : Source register for destination table

L : Length of destination table

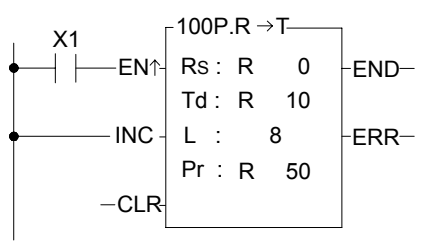
Pr : Pointer register

Rs, Td can associate with V, Z, P0~P9 index register as indirect addressing.

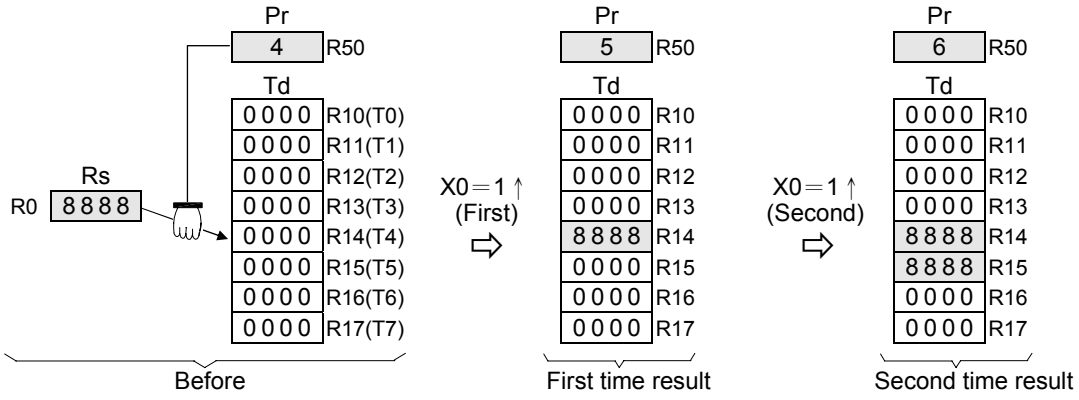
Range / Operand	WX	WY	WM	WS	TMR	CTR	HR	IR	OR	SR	ROR	DR	K	XR
	WX0 | WX240	WY0 | WY240	WM0 | WM1896	WS0 | WS984	T0 | T255	C0 | C255	R0 | R3839	R3840 | R3903	R3904 | R3967	R3968 | R4167	R5000 | R8071	D0 | D4095	16/32-bit +/- number	V, Z P0~P9
Rs	○	○	○	○	○	○	○	○	○	○	○	○	○	○
Td		○	○	○	○	○	○		○	○*	○*	○		○
L							○				○*	○	2~2048	
Pr		○	○	○	○	○	○		○	○*	○*	○		

- When move control "EN" =1 or "EN↑" (P instruction) changes from 0 to 1, the contents of the source register Rs will be written onto the register Tdpr indicated by the pointer Pr within the destination table Td (length is L). Before executing, this instruction will first check the pointer clear "CLR" input signal. If "CLR" is 1, it will first clear the pointer Pr, and then carry out the move operation. After the move has been completed, it will then check the Pr value. If the Pr value has already reached L-1 (point to the last register in the table) then it will only set the move-to-end flag "END" to 1, and finish execution of this instruction. If the Pr value is less than L-1, then it must again check the pointer increment "INC" input signal. If "INC" is 1, then Pr value will be also increased. Besides, pointer clear "CLR" is able to operate independently, without being influenced by other input.

- The effective range of the pointer is 0 to L-1. Beyond this range, the pointer error "ERR" will be set to 1, and this instruction will not be performed.

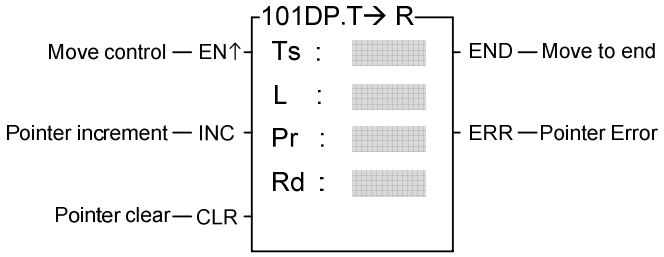


- The example at left at the very beginning pointer Pr =4, the entire content of table Td is 0, and the Rs value is 8888. The diagram below shows the operation results when X1 have the transition of 0 to 1 twice.
- Because INC is 1, Pr will increase by 1 each time the instruction is executed.



FUN101 D P T→R	TABLE TO REGISTER MOVE	FUN101 D P T→R

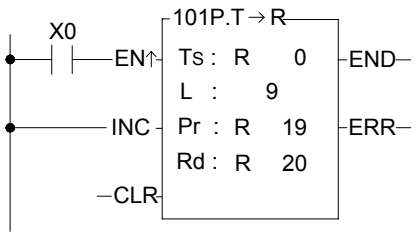
Ladder symbol



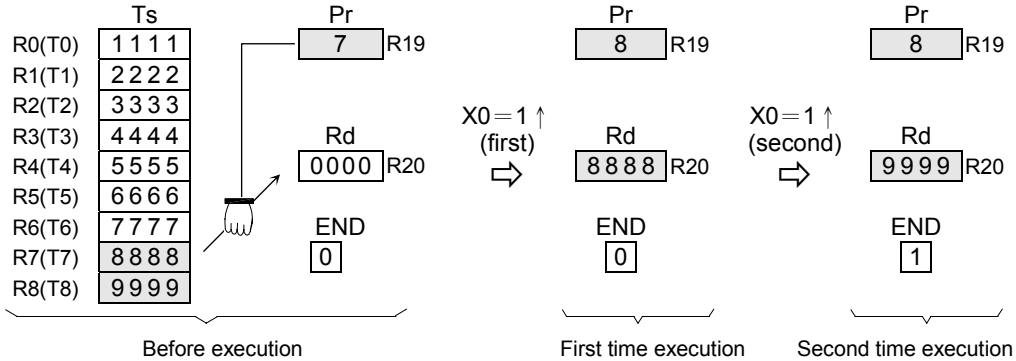
Ts : Source table starting register
L : Length of source table
Pr : Pointer register
Rd : Destination register
Ts, Rd may combine with V, Z, P0~P9 to serve indirect address application.

Range / Ope-rand	WX	WY	WM	WS	TMR	CTR	HR	IR	OR	SR	ROR	DR	K	XR
	WX0 | WX240	WY0 | WY240	WM0 | WM1896	WS0 | WS984	T0 | T255	C0 | C255	R0 | R3839	R3840 | R3903	R3904 | R3967	R3968 | R4167	R5000 | R8071	D0 | D4095	16/32bit +/- number	V, Z P0~P9
Ts	○	○	○	○	○	○	○	○	○	○	○	○		○
L							○				○*	○		
Pr		○	○	○	○	○	○		○	○*	○*	○	2~2048	
Rd		○	○	○	○	○	○		○	○*	○*	○		○

- When move control "EN" =1 or "EN↑" (P instruction) changes from 0 to 1, the value of the register Tspr specified by pointer Pr within source table Ts (length is L) will be written into the destination register Rd. Before executing, this instruction will first check the input signal of pointer clear "CLR". If "CLR" is 1, it will first clear Pr and then carry out the move operation. After completing the move operation, it will then check the value of Pr. If the Pr value has already reached L-1 (point to the last register in the table), then it sets the move-to-end flag to 1, and finishes executing of this instruction. If Pr is less than L-1, it check the status of "INC". If "INC" is 1, then it will increase Pr and finish the execution of this instruction. Besides, pointer clear "CLR" can execute independently and is not influenced by other inputs.
- The effective range of the pointer is 0 to L-1. Beyond this range the pointer error "ERR" will be set to 1 and this instruction will not be carried out.

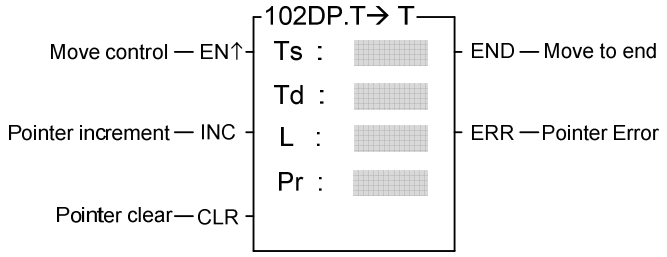


- In the example at left, at the very beginning Pr =7 and Ts and Rd are as shown at left in the diagram below. When X0 have a transition from 0 to 1 twice, the results are shown at right in the diagram below.
- At the second time execution, the pointer has already reached to the end so there will be no increment.



FUN102 D P T→T	TABLE TO TABLE MOVE	FUN102 D P T→T

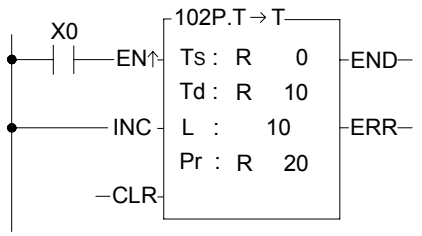
Ladder symbol



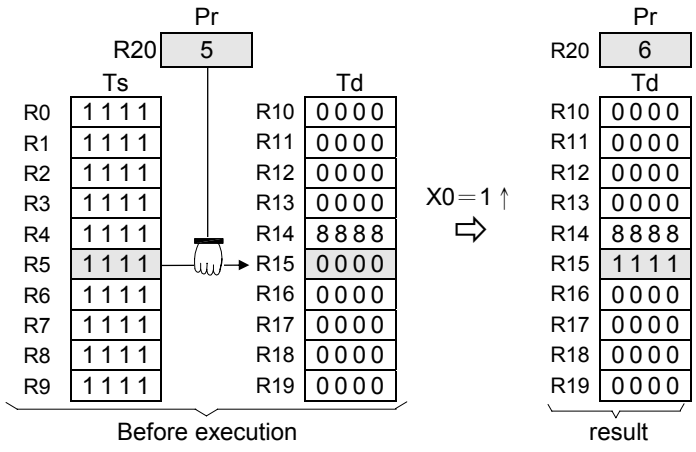
Ts : Starting number of source table register
Td : Starting number of destination table register
L : Table (Ts and Td) length
Pr : Pointer register
Ts, Td may combine with V, Z, P0~P9 to serve indirect address application.

Operand \ Range	WX	WY	WM	WS	TMR	CTR	HR	IR	OR	SR	ROR	DR	K	XR
	WX0 | WX240	WY0 | WY240	WM0 | WM1896	WS0 | WS984	T0 | T255	C0 | C255	R0 | R3839	R3840 | R3903	R3904 | R3967	R3968 | R4167	R5000 | R8071	D0 | D4095	2 | 2048	V, Z | P0~P9
Ts	○	○	○	○	○	○	○	○	○	○	○	○		○
Td		○	○	○	○	○	○		○	○*	○*	○		○
L							○				○*	○	○	
Pr		○	○	○	○	○	○		○	○*	○*	○		

- When move control "EN" =1 or "EN↑" (P instruction) changes from 0 to 1, the register Tspr pointed by pointer Pr within the source table will be moved to a register Tdpr, which also pointed by the pointer Pr in the destination table. Before execution, it will first check the input signal of pointer clear "CLR". If "CLR" is 1, it will first clear Pr to 0 and then do the move (in this case Ts0→Td0). After the move action has been completed it will then check the value of pointer Pr. If the Pr value has already reached L-1 (point to the last register on the table), then it will set the move-to-end flag "END" to 1 and finish executing of this instruction. If the Pr value is less than L-1, it will check the status of "INC". If "INC" is 1, then the Pr value will be increased by 1 before execution. Besides, pointer clear "CLR" can execute independently, and will not be influenced by other input.
- The effective range of the pointer is 0 to L-1. Beyond this range, the pointer error flag "ERR" will be set to 1, and this instruction will not be carried out.

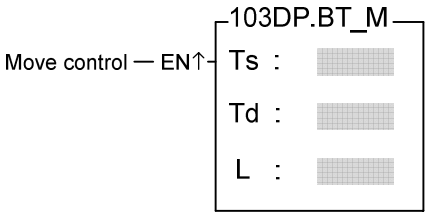


- The diagram at left below is the status before execution. When X0 from 0 to 1, the content of R5 in Ts table will copy to R15 and pointer R20 will be increased by 1.



FUN103 D P BT_M	BLOCK TABLE MOVE	FUN103 D P BT_M

Ladder symbol



Ts :Starting register for source table

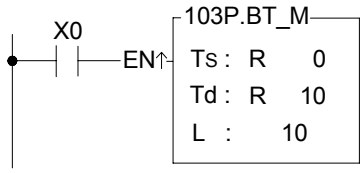
Td : Starting register for destination table

L: Lengths of source and destination tables

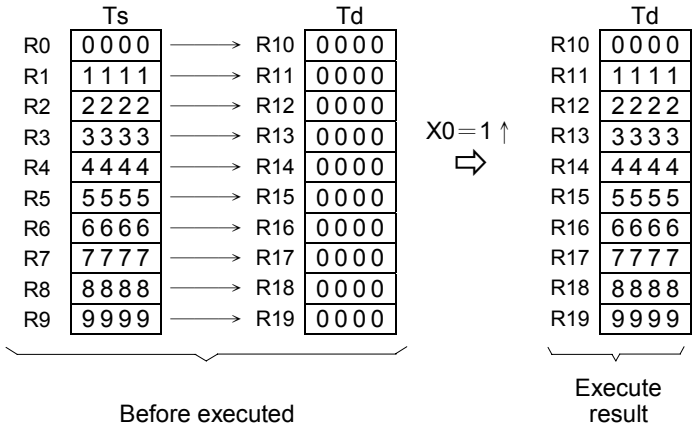
Ts, Td may combine with V, Z, P0~P9 to serve indirect address application.

Operand \ Range	WX	WY	WM	WS	TMR	CTR	HR	IR	OR	SR	ROR	DR	K	XR
	WX0 | WX240	WY0 | WY240	WM0 | WM1896	WS0 | WS984	T0 | T255	C0 | C255	R0 | R3839	R3840 | R3903	R3904 | R3967	R3968 | R4167	R5000 | R8071	D0 | D4095	2 | 256	V, Z | P0~P9
Ts	○	○	○	○	○	○	○	○	○	○	○	○		○
Td		○	○	○	○	○	○		○	○*	○*	○		○
L							○				○*	○	○	

- In this instruction the source table and destination table are the same length. When this instruction was executed all the data in the Ts table is completely copied to Td. No pointer is involved in this instruction.
- When move control "EN" =1 or "EN↑" (P instruction) changes from 0 to 1, all the data from source table Ts (length L) is copied to the destination table Td, which is the same length.
- One table is completely copied every time this instruction is executed, so if the table length is long, it will be very time consuming. In practice, P modifier should be used to avoid time waste caused by each scan repeating the same movement action.

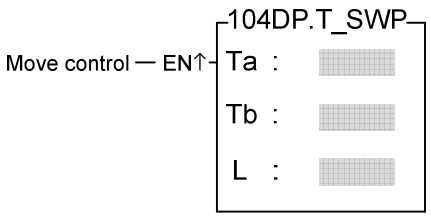


- The diagram at left below is the status before execution. When X0 from 0 to 1, the content of R0~R9 in Ts table will copy to R10~R19.



FUN104 D P T_SWP	BLOCK TABLE SWAP	FUN104 D P T_SWP

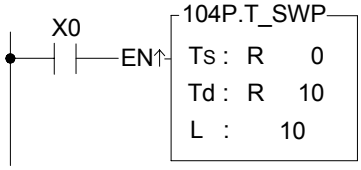
Ladder symbol



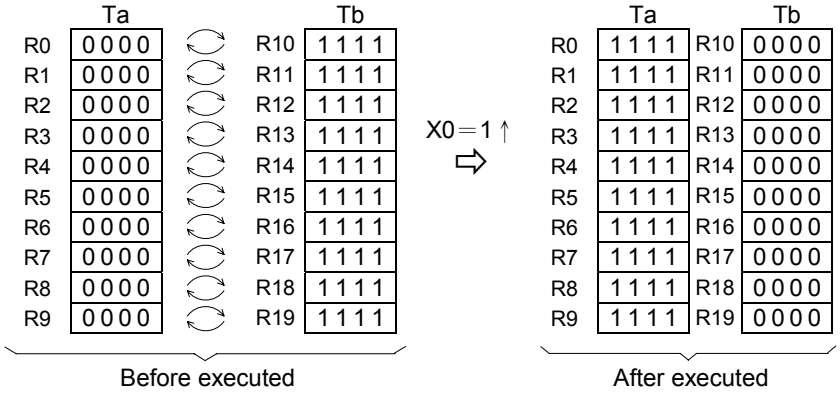
Ta : Starting register of Table a
Tb : Starting register of Table b
L : Lengths of Table a and b
Ts, Tb may combine with V, Z, P0~P9 to serve indirect address application.

Range / Operand	WY	WM	WS	TMR	CTR	HR	OR	SR	ROR	DR	K	XR
	WY0 | WY240	WM0 | WM1896	WS0 | WS984	T0 | T255	C0 | C255	R0 | R3839	R3904 | R3967	R3968 | R4167	R5000 | R8071	D0 | D4095	2 | 256	V, Z P0~P9
Ta	○	○	○	○	○	○	○	○*	○*	○		○
Tb	○	○	○	○	○	○	○	○*	○*	○		○
L						○			○*	○	○	

- This instruction swaps the contents of Tables a and b, so the table must be the same length, and the registers in the table must of write able. Since a complete swap is done with each time the instruction is executed, no pointer is needed.
- When move control "EN" =1 or "EN↑" (P instruction) changes from 0 to 1, the contents of Table a and Table b will be completely swapped.
- This instruction will swap all the registers specified in L each time the instruction is executed, so if the table length is big, it will be very time consuming, therefore P instruction should be used.

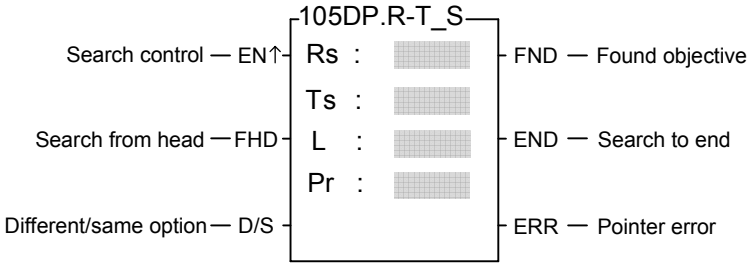


- The diagram at left below is the status before execution. When X0 from 0 to 1, the contents of R0~R9 in Ts table will swap with R10~R19.



FUN105 D P R-T_S	REGISTER TO TABLE SEARCH	FUN105 D P R-T_S

Ladder symbol

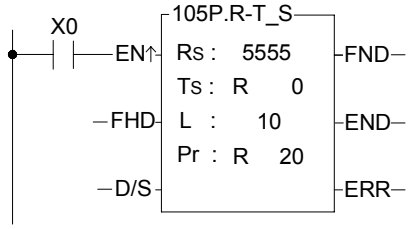


Rs : Data to search, It can be a constant or a register
Ts : Starting register of table being searched
L : Label length
Pr : Pointer of table
Rs, Ts may combine with V, Z, P0~P9 to serve indirect address application.

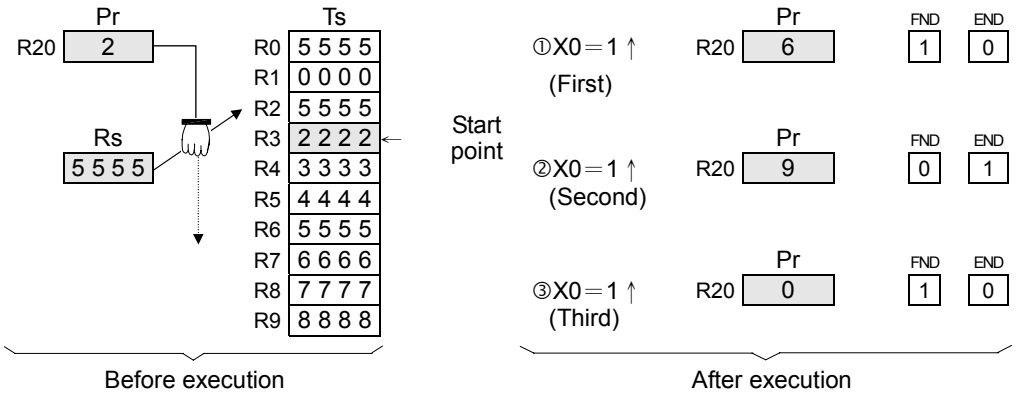
Operand \ Range	WX	WY	WM	WS	TMR	CTR	HR	IR	OR	SR	ROR	DR	K	XR
	WX0 | WX240	WY0 | WY240	WM0 | WM1896	WS0 | WS984	T0 | T255	C0 | C255	R0 | R3839	R3840 | R3903	R3904 | R3967	R3968 | R4167	R5000 | R8071	D0 | D4095	16/32-bit +/- number	V, Z | P0~P9
Rs	○	○	○	○	○	○	○	○	○	○	○	○	○	○
Ts	○	○	○	○	○	○	○	○	○	○	○	○		○
L							○				○*	○	2~256	
Pr		○	○	○	○	○	○		○	○*	○*	○		

- When search control "EN" =1 or "EN↑" (P instruction) changes from 0 to 1, will search from the first register of Table Ts (when "FHD" =1 or Pr value has reached L-1), or from the next register (Tspr + 1) pointed by the pointer within the table ("FHD" =0, while Pr value is less than L-1) to find the first data different with Rs(when D/S =1) or find the first data the same with Rs (when D/S =0). If it find a data match the condition it will immediately stop the search action, and the pointer Pr will point to that data and found objective flag "FND" will set to 1. When the searching has searched to the last register of the table, the execution of the instruction will stop, whether it was found or not. In that case the search-to-end flag "END" will be set to 1 and the Pr value will stop at L-1. When this instruction next time is executed, Pr will automatically return to the head of the table (Pr =0) before the search begin.

- The effective range of Pr is 0 to L-1. If the value exceeds this range then the pointer error flag "ERR" will change to 1, and this instruction will not be carried out.

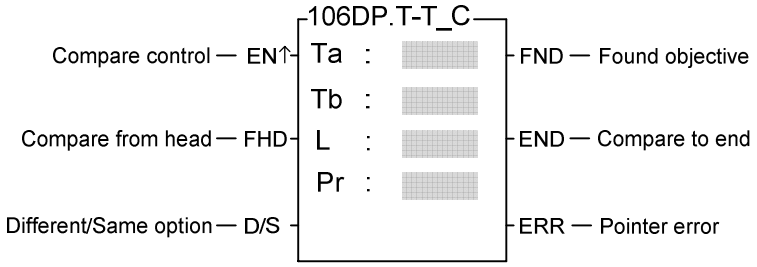


- The instruction at left is searching the table for a register with the value 5555 (because D/S =0, it is searching for same value). Before execution, the pointer point to R2, but the starting point of the search is Pr + 1 (i.e. it starts from R3). After X0 has transition from 0 to 1 3 times, the results of each search may be obtained as shown in the diagram below.



FUN106 D P T-T_C	TABLE TO TABLE COMPARE	FUN106 D P T-T_C

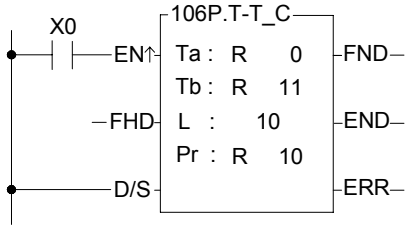
Ladder symbol



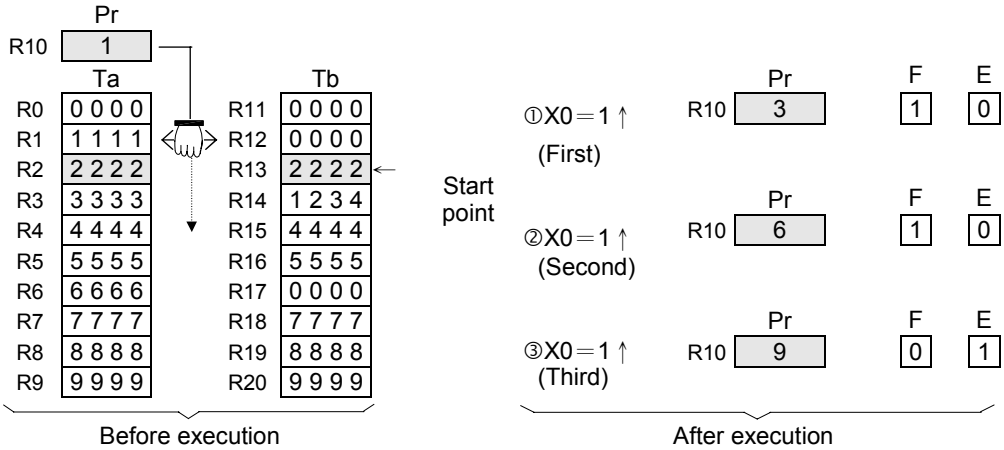
Ta : Starting register of Table a
Tb : Starting register of Table b
L : Lengths of Table
Pr : Pointer
Ta, Tb may combine with V, Z, P0~P9 to serve indirect address application.

Operand \ Range	WX	WY	WM	WS	TMR	CTR	HR	IR	OR	SR	ROR	DR	K	XR
	WX0 | WX240	WY0 | WY240	WM0 | WM1896	WS0 | WS984	T0 | T255	C0 | C255	R0 | R3839	R3840 | R3903	R3904 | R3967	R3968 | R4167	R5000 | R8071	D0 | D4095	2 | 256	V, Z | P0~P9
Ta	○	○	○	○	○	○	○	○	○	○	○	○		○
Tb	○	○	○	○	○	○	○	○	○	○	○	○		○
L							○				○*	○	○	
Pr		○	○	○	○	○	○		○	○*	○*	○		

- When compare control "EN" =1 or "EN↑" (P instruction) changes from 0 to 1, then starting from the first register in the tables Ta and Tb (when "FHD" =1 or Pr value has reached L-1) or starting from the next pair of registers (Tapr+1 and Tbpr+1) pointed by Pr ("FHD" =0, while Pr is less than L-1), this instruction will search for pairs of registers with different values (when "D/S" =1) or the same value (when "D/S" =0). When search found (either different or the same), it will immediately stop the search and the pointer Pr will point to the register pairs met the search criteria. The found flag "FND" will be set to 1. When it has searched to the last register of the table, the instruction will stop executing. Whether it found or not. The compare-to-end flag "END" will be set to 1, and the pointer value will stop at L-1. When this instruction is executed next time, Pr will automatically return to the head of the table to begin the search.
- The effective range of Pr is 0 to L-1. As this will affect the result of the search, the Pr value should not changed by other programs during the operation. If the Pr value not in the effective range, the pointer error flag "ERR" will be set to 1, and this instruction will not be carried out.

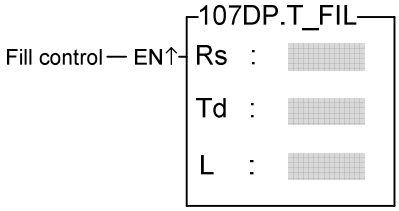


- The instruction at left starts from the register next to the register pointed by the pointer (because "FHD" is 0) to search for register pairs with different data (because "D/S" is 1) within the 2 tables. At the very beginning, Pr points to Ta1 and Tb1. There are 3 different pairs of data at the position 1, 3, 6 of the table. However, it does not compare from the beginning, and this instruction will start searching from position 3 downwards. After X0 has changed 3 times from 0 to 1, the results are shown in the diagram below.



FUN107 D P T_FIL	TABLE FILL	FUN107 D P T_FIL

Ladder symbol



Rs : Source data to fill, can be a constant or a register

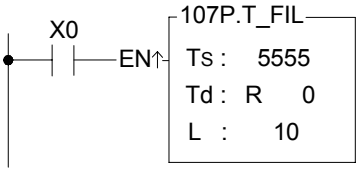
Td : Starting register of destination table

L :Table length

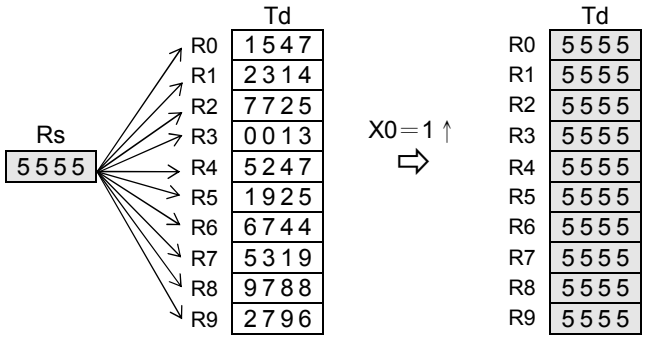
Rs, Td may combine with V, Z, P0~P9 to serve indirect address application.

Range / Operand	WX	WY	WM	WS	TMR	CTR	HR	IR	OR	SR	ROR	DR	K	XR
	WX0 | WX240	WY0 | WY240	WM0 | WM1896	WS0 | WS984	T0 | T255	C0 | C255	R0 | R3839	R3840 | R3903	R3904 | R3967	R3968 | R4167	R5000 | R8071	D0 | D4095	16/32-bit +/- number	V, Z P0~P9
Ts	○	○	○	○	○	○	○	○	○	○	○	○	○	○
Td		○	○	○	○	○	○		○	○*	○*	○		○
L							○				○*	○	2~256	

- When fill control "EN" =1 or "EN↑" (P instruction) changes from 0 to 1, the Rs data will be filled into all the registers of the table Td.
- This instruction is mainly used for clearing the table (fill 0) or unifying the table (filling in the same values). It should be used with the P instruction.



- The instruction at left will fill 5555 into the whole table Td. The results are as shown in the diagram below.

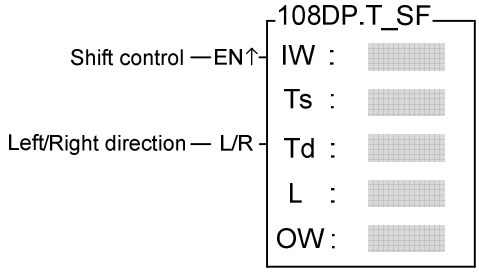


Before execution

After execution

FUN108 D P T_SHF	TABLE SHIFT	FUN108 D P T_SHF

Ladder symbol



IW : Data to fill the room after shift operation, can be a constant or a register

Ts : Source table

Td : Destination table storing shift results

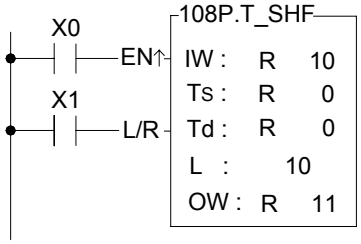
L : Lengths of tables Ts and Td

OW : Register to accept the shifted out data

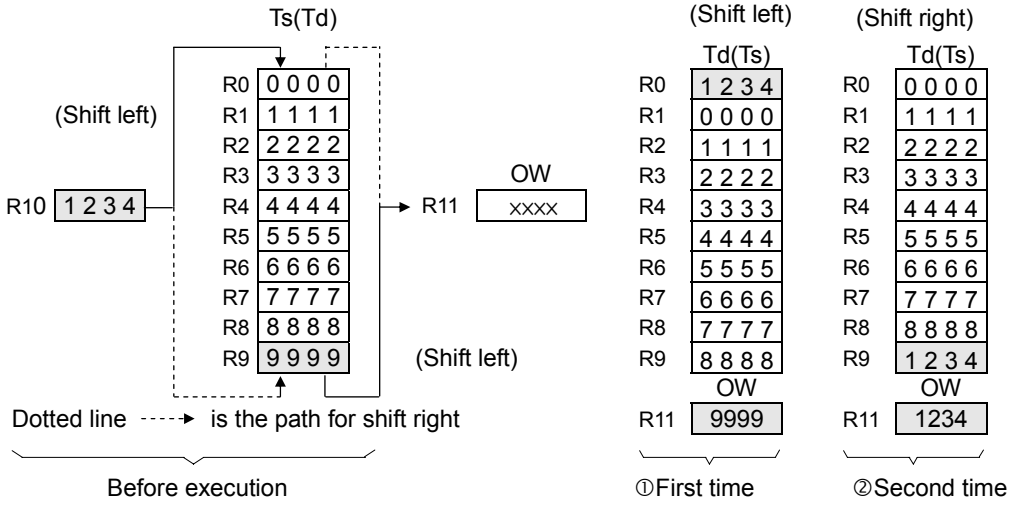
Ts, Td may combine with V, Z, P0~P9 to serve indirect address application.

Range / Operand	WX	WY	WM	WS	TMR	CTR	HR	IR	OR	SR	ROR	DR	K	XR
	WX0 | WX240	WY0 | WY240	WM0 | WM1896	WS0 | WS984	T0 | T255	C0 | C255	R0 | R3839	R3840 | R3903	R3904 | R3967	R3968 | R4167	R5000 | R8071	D0 | D4095	16/32-bit +/- number	V, Z P0~P0
IW	○	○	○	○	○	○	○	○	○	○	○	○	○	
Ts	○	○	○	○	○	○	○	○	○	○	○	○		○
Td		○	○	○	○	○	○		○	○*	○*	○		○
L							○				○*	○	2~256	
OW		○	○	○	○	○	○		○	○*	○*	○		

- When shift control "EN" =1 or "EN↑" (P instruction) changes from 0 to 1, all the data from table Ts will be taken out and shifted one position to the left (when "L/R" =1) or to the right (when "L/R" =0). The room created by the shift operation will be filled by IW and the results will be written into table Td. The data shifted out will be written into OW.

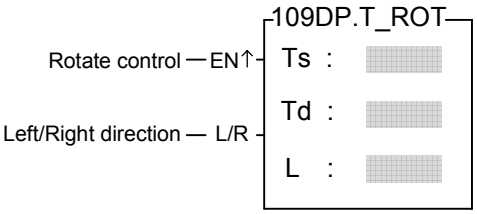


- In the program at left, Ts and Td is the same table. Therefore, the table shifts itself and then writes back to itself (the table must be writ able). It first perform a shift left operation (let X1 =1, and X0 go from 0 to 1) then perform a shift to right operation (let X1 =0, and makes X0 go from 0 to 1). The results are shown at right in the diagram below.



FUN109 D P T_ROT	TABLE ROTATE	FUN109 D P T_ROT

Ladder symbol



Ts : Source table for rotate

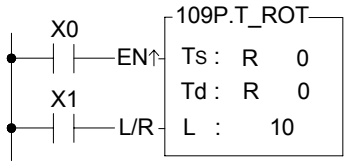
Td : Destination table storing results of rotation

L : Lengths of table

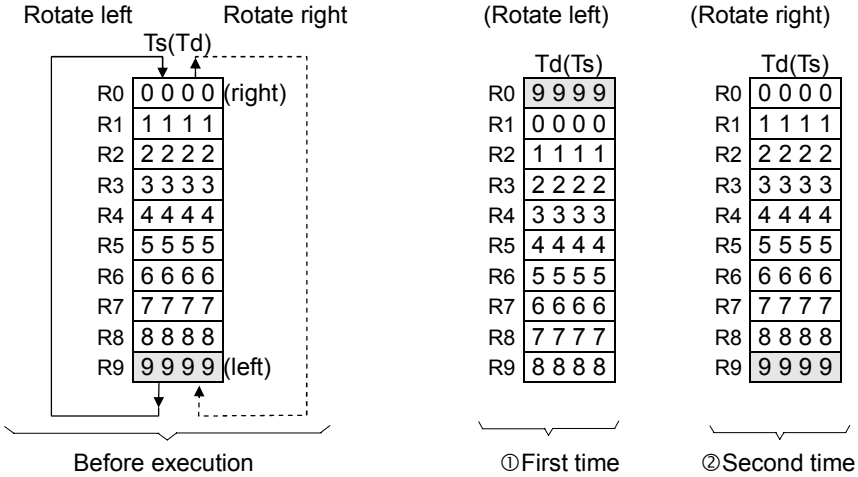
Ts, Td may combine with V, Z, P0~P9 to serve indirect address application

Operand \ Range	WX	WY	WM	WS	TMR	CTR	HR	IR	OR	SR	ROR	DR	K	XR
	WX0 | WX240	WY0 | WY240	WM0 | WM1896	WS0 | WS984	T0 | T255	C0 | C255	R0 | R3839	R3840 | R3903	R3904 | R3967	R3968 | R4167	R5000 | R8071	D0 | D4095	2 | 256	V、Z | P0~P9
Ts	○	○	○	○	○	○	○	○	○	○	○	○		○
Td		○	○	○	○	○	○		○	○*	○*	○		○
L							○				○*	○	○	

- When rotate control "EN" =1 or "EN↑" (P instruction) changes from 0 to 1, the data from the table of Ts will be rotated 1 position to the left (when "L/R" =1)or 1 position to the right (when "L/R" =0). The results of the rotation will then be written onto table Td.

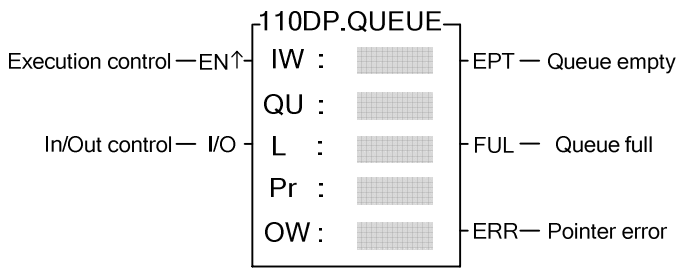


- In the program at left, Ts and Td is the same table. The table after rotation will write back to itself. It first perform one left rotation (let X1 =1, and X0 go from 0 to 1), and then performs one right rotation (let X1 =0, and X0 go from 0 to 1). The results are shown at right in the diagram below.



FUN110 D P QUEUE	QUEUE	FUN110 D P QUEUE

Ladder symbol



IW : Data pushed into queue, can be a constant or a register

QU : Starting register of queue

L : Size of queue

Pr : Pointer register

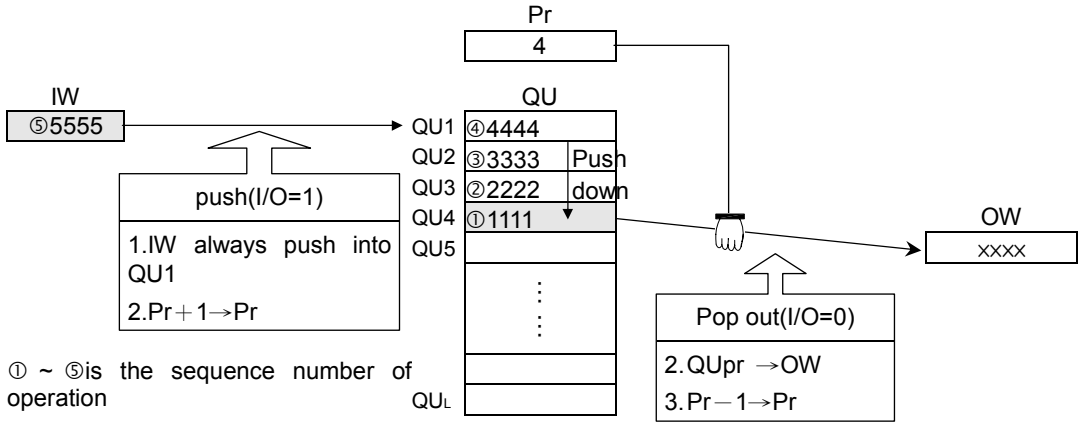
OW : Register accepting data popped out from queue

QU may combine with V, Z, P0~P9 to serve indirect address application.

Range / Operand	WX	WY	WM	WS	TMR	CTR	HR	IR	OR	SR	ROR	DR	K	XR
	WX0 | WX240	WY0 | WY240	WM0 | WM1896	WS0 | WS984	T0 | T255	C0 | C255	R0 | R3839	R3840 | R3903	R3904 | R3967	R3968 | R4167	R5000 | R8071	D0 | D4095	16/32-bit +/- number	V, Z P0~P9
IW	○	○	○	○	○	○	○	○	○	○	○	○	○	
QU		○	○	○	○	○	○		○	○	○*	○		○
L							○				○*	○	2~256	
Pr		○	○	○	○	○	○		○	○*	○*	○		
OW		○	○	○	○	○	○		○	○*	○*	○		

- Queue is also a kind of table. It is different from ordinary table in that its queue register numbers go from 1 to L and not from 0 to L-1. In other words QU_1~QU_L respectively correspond to pointers Pr =1 to L, and Pr =0 is used to show that the queue is empty.

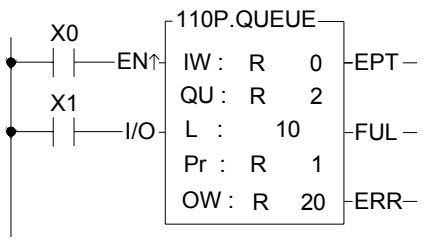
- Queue is a first in first out (FIFO) device, i.e. - the data that first pushed into the queue will be the first to pop out from the queue. A queue is comprised of L consecutive 16 or 32 bits registers (D instruction) starting from the QU register, as in the diagram below:



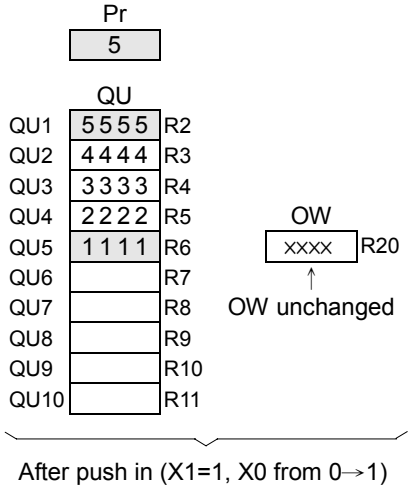
- When execution control "EN" =1 or "EN↑" (P instruction) changes from 0 to 1, the status of in/out control "I/O" determines whether the IW data will be pushed into the queue (when "I/O" =1) or be popped out and transferred to OW (when "I/O" =0). As shown in the diagram above, the IW data will always be pushed into the first (QU1) register of the queue. After it has been pushed in, Pr will immediately be increased by 1, so that the pointer can always point to the first data that was pushed into the queue. When it is popped out, the data pointed by Pr will be transferred directly to OW. Pr will be reduced by 1, so that it still point to the first data remained in the queue.

FUN110 D P QUEUE	QUEUE	FUN110 D P QUEUE

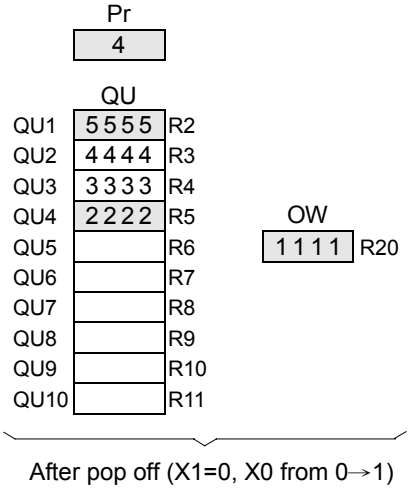
- If no data has yet been pushed into the queue or the pushed in data has already been popped out (Pr =0), then the queue empty flag will be set to 1. In this case, even if there is further popping out action, this instruction will not be executed. If data is only pushed in and not popped out, or pushed in is more than that popped out, then the queue finally becomes full (pointer Pr indicates the QU_L position), and the queue full flag is changed to 1. In this case, if there is more pushing in action, this instruction will not execute. The pointer for this instruction is used during access of the queue, to indicate the data that was pushed in the earliest. Other programs should not be allowed to change it, or else an operation error will be created. If there is a specific application, which requires the setting of a Pr value, then its permissible range is 0 to L (0 means empty, and 1 to L respectively correspond to QU1 to QUL). Beyond this range, the pointer error flag "ERR" will be set as 1, and this instruction will not be carried out.



- The program at left assumes the queue content is the same with the queue at preceding page. It will first perform queue push operation, and then perform pop out action. The results are shown below. Under any circumstance, Pr always point to the first (oldest) data that was remained in queue.



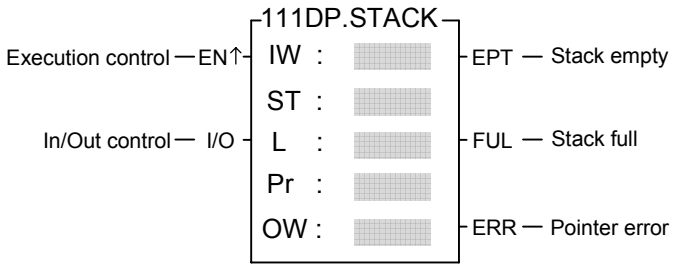
After push in (X1=1, X0 from 0→1)



After pop off (X1=0, X0 from 0→1)

FUN111 D P STACK	STACK	FUN111 D P STACK

Ladder symbol



IW : Data pushed into stack, can be a constant or a register.

ST : Starting register of stack

L : Size of stack

Pr : Pointer register

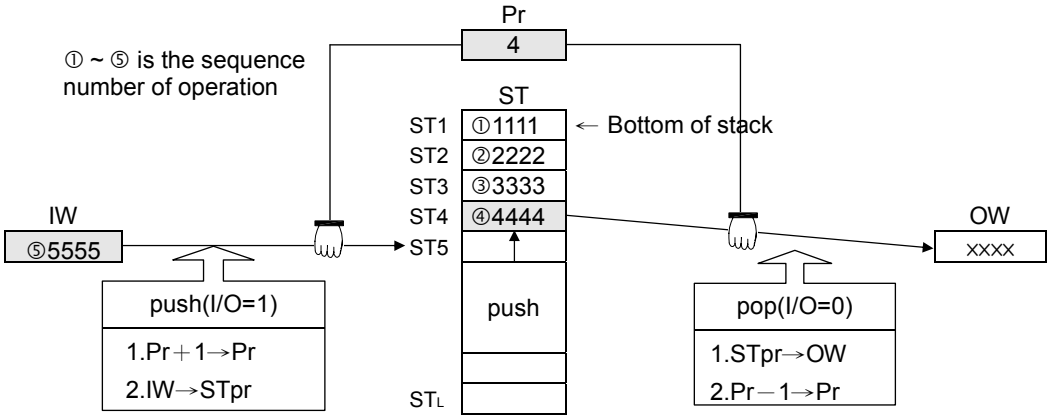
OW : Register accepting data popped out from stack.

ST may combine with V, Z, P0~P9 to serve indirect address application.

Range / Operand	WX	WY	WM	WS	TMR	CTR	HR	IR	OR	SR	ROR	DR	K	XR
	WX0 | WX240	WY0 | WY240	WM0 | WM1896	WS0 | WS984	T0 | T255	C0 | C255	R0 | R3839	R3840 | R3903	R3904 | R3967	R3968 | R4167	R5000 | R8071	D0 | D4095	16/32-bit +/- number	V, Z P0~P9
IW	○	○	○	○	○	○	○	○	○	○	○	○	○	
ST		○	○	○	○	○	○		○	○*	○*	○		○
L							○				○*	○	2~256	
Pr		○	○	○	○	○	○		○	○*	○*	○		
OW		○	○	○	○	○	○		○	○*	○*	○		

- Like queue, stack is also a kind of table. The nature of its pointer is exactly the same as with queue, i.e. Pr =1 to L, which corresponds to ST_1 to ST_L, and when Pr =0 the stack is empty.

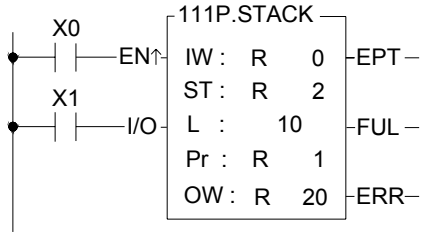
- Stack is the opposite of queue, being a last in first out (LIFO) device. This means that the data that was most recently pushed into the stack will be the first to be popped out of the stack. The stack is comprised of L consecutive 16 or 32-bit (D instruction) registers starting from ST, as shown in the following diagram:



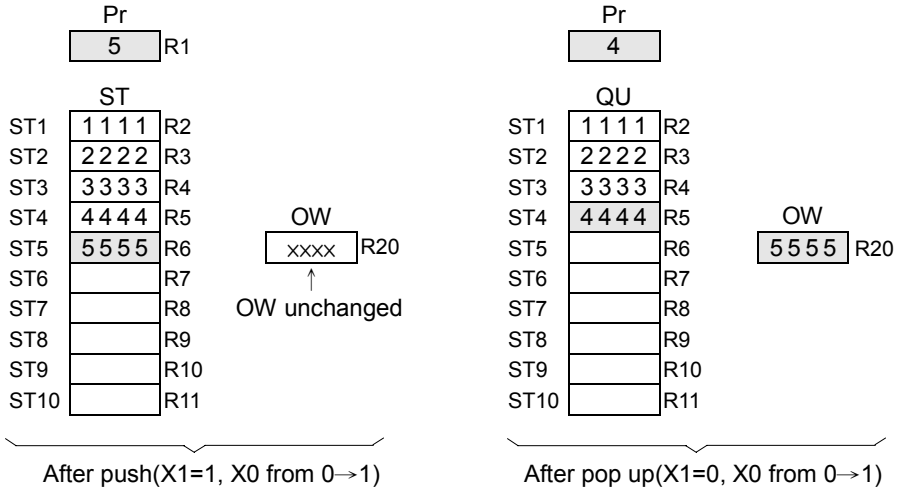
- When execution control "EN" =1 or "EN↑" (P instruction) changes from 0 to 1, the status of in/out control "I/O" determines whether the IW data will be pushed into the stack (when "I/O" =1), or the data pointed by Pr within the stack (the data most recently pushed into the stack) will be moved out and transferred to OW (when "I/O" =0). Note that the data pushed in is stacking, so before pushed in, Pr will increased by 1 to point to the top of the stack then the data will be pushed in. When it is popped out, the data pointed by pointer Pr (the most recently pushed in data) will be transferred to OW. After then Pr will be decreased by 1. Under any circumstances, the pointer Pr will always point to the data that was pushed into the stack most recently.

FUN111 D P STACK	STACK	FUN111 D P STACK

- When no data has yet been pushed into the stack or the pushed in data has already been popped out (Pr =0), the stack empty flag "EPT" will set to 1. In this case any further pop up actions, will be ignored. If more data is pushed than popped out, sooner or latter the stack will be full (pointer Pr points to ST_L position), and the stack full flag "FUL" will set to 1. In this case any further push actions, will be ignored. As with queue, the stack pointer in normal case should not be changed by other instructions. If there is a special application which requires to set the Pr value, then its effective range is 0 to L (0 means empty, 1 to L respectively correspond to ST_1 to ST_L). Beyond this range, the pointer error flag "ERR" will set to 1, and the instruction will not be carried out.

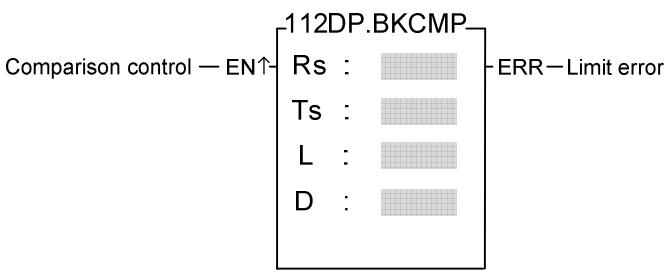


- The program at left assumes that the initial content of the stack is just as in the diagram of a stack on the preceding page. The operation illustrated in this example is to push a data and than pop it from stack. The results are shown below. Under any circumstances, Pr always point to the data that was most recently pushed into the stack.



FUN112 D P BKCMP	BLOCK COMPARE（DRUM）	FUN112 D P BKCMP

Ladder symbol



Rs : Data for compare, can be a constant or a register.

Ts : Starting register block storing upper and lower limit

L : Number of pairs of upper and lower limits

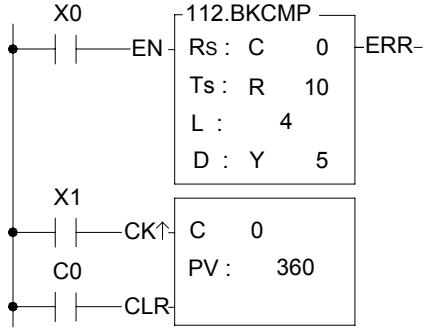
D : Starting relay storing results of comparison

Range / Operand	Y	M	S	WX	WY	WM	WS	TMR	CTR	HR	IR	OR	SR	ROR	DR	K
	Y0 | Y255	M0 | M999	S0 | S999	WX0 | WX240	WY0 | WY240	WM0 | WM1896	WS0 | WS984	T0 | T255	C0 | C255	R0 | R3839	R3840 | R3903	R3904 | R3967	R3968 | R4167	R5000 | R8071	D0 | D4095	16/32-bit +/- number
Rs				○	○	○	○	○	○	○	○	○	○	○	○	○
Ts				○	○	○	○	○	○	○	○	○	○	○	○	
L										○				○*	○	1~256
D	○	○	○													

- When comparison control "EN" =1 or "EN↑" (P instruction) changes from 0 to 1, comparisons will be perform one by one between the contents of Rs and the upper and lower limits form by L pairs of 16 or 32-bit (D instruction) registers starting from the Ts register (starting from T0 each adjoining 2 register units form a pair of upper and lower limits). If the value of Rs falls within the range of the pair, then the bit within the comparison results relay D which corresponds to that pair will be set to 1. Otherwise it will be set as 0 until comparison of all the L pairs of upper and lower limits is completed.
- When M1975=0, if there is any pair where the upper limit value is less than the lower limit value, then the limit error flag "ERR" will be set to 1, and the comparison output for that pair will be 0.
- When M1975=1, there is no restriction on the relation of upper limit and lower limit, this can apply for 360°rotary electronic drum switch application.

	Upper limit	Lower limit	Compare	Compared value	Result
0	T_{S1}	T_{S0}	⟷	Rs	D_0
1	T_{S3}	T_{S2}	⟷		D_1
.	.	.	.		.
L−1	T_{S2L-1}	T_{S2L-2}	⟷		D_{L-1}

- Actually this instruction is a drum switch, which can be used in interrupt program and when incorporate with immediate I/O instruction (IMDIO) can achieve an accurate electronic drum.

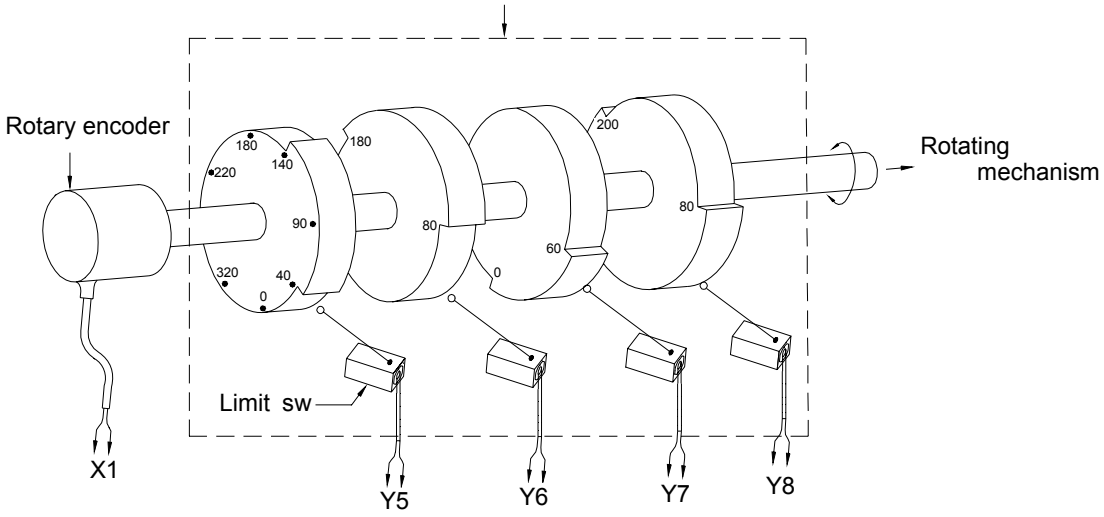


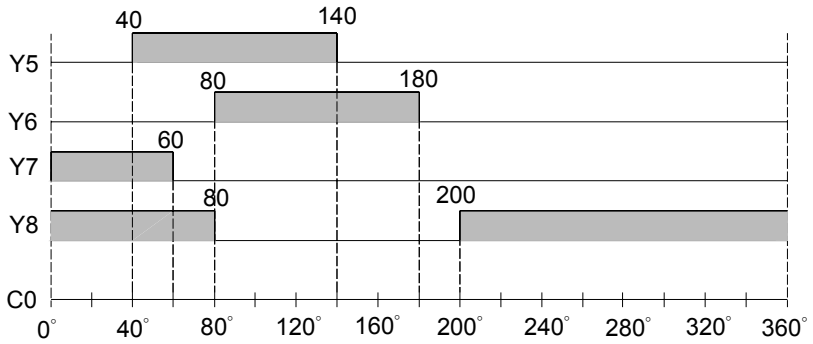
- In this program, C0 represents the rotation angle (Rs) of a drum shaft. The block compare instruction performs a comparison between Rs and the 4 pairs (L =4) of upper and lower limits, R10,R11, R12,R13, R14,R15 and R16,R17. The comparison results can be obtained from the four drum output points Y5 to Y8.
- The input point X1 is a rotation angle detector mounted on the drum shaft. With each one degree rotation of the drum shaft angle, X1 produces a pulse. When the drum shaft rotates a full cycle, X1 produces 360 pulses.

FUN112 D P BKCMP	BLOCK COMPARE（DRUM）	FUN112 D P BKCMP

- The program in the diagram above coordinates a rotary encoder or other rotating angle detection device (directly connect to a rotating mechanism), which can form a mechanical device equivalent to the mechanical structure of an actual drum (see mechanism shown within dotted line in diagram below). While the upper and lower limits are being adjusted, you can change at will the range of the activated angle of the drum. This cannot be done with the traditional drum mechanism.

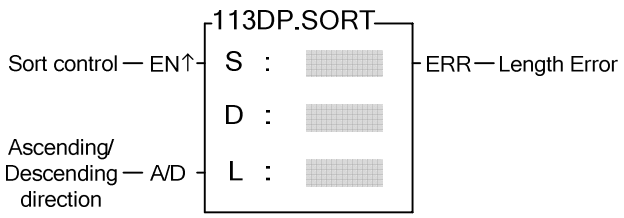
Equivalent mechanical drum emulated by above program





FUN113 D P SORT	DATA SORTING	FUN113 D P SORT

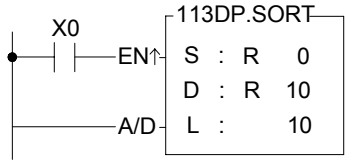
Ladder symbol



S : Starting register of source registers to sort
D : Starting register of destination registers to store the data after sorted
L : Total register for sorting
S, D operand can combine with V, Z, P0~P9 for index addressing while word operation.

Range	WX	WY	WM	WS	TMR	CTR	HR	IR	OR	SR	ROR	DR	K	XR
	WX0 | WX240	WY0 | WX240	WM0 | WM1896	WS0 | WS984	T0 | T255	C0 | C255	R0 | R3839	R3840 | R3903	R3904 | R3967	R3968 | R4167	R5000 | R8071	D0 | D4095	2 | 256	V, Z P0~P9
S	○	○	○	○	○	○	○	○	○	○	○	○		○
D		○	○	○	○	○	○		○	○*	○*	○		○
L							○				○	○	○	

- When sort control "EN" =1 or "EN↑" (P instruction) changes from 0 to 1, will sort the registers with ascending order (if A/D =1) or descending order (if A/D =0) and put the sorted result to the registers starting by D register.
- The valid data length of sort operation is between 2 and 256, other length will set the “ERR” to 1 and the sort operation will not perform.



• The example at left sorts the table comprised of R0~R9 and stores the sorted data to the table locate at R10~R19.

	S
R0	1547
R1	2314
R2	7725
R3	0013
R4	5247
R5	1925
R6	6744
R7	5319
R8	9788
R9	2796

Before

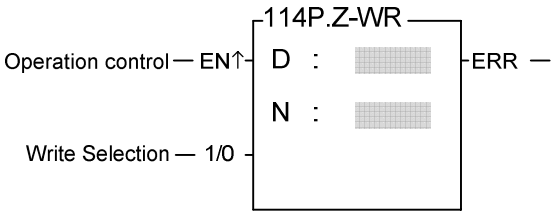
X0＝1 ↑ ⇨

	D
R10	0013
R11	1547
R12	1925
R13	2314
R14	2796
R15	5247
R16	5319
R17	6744
R18	7725
R19	9788

After

FUN114 P Z-WR	ZONE WRITE	FUN114 P Z-WR

Ladder symbol



D : Starting address of being set or reset

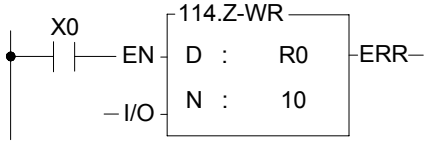
N : Quantity of being set or reset, 1~511

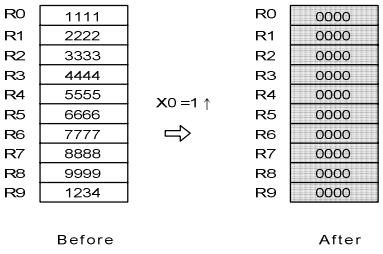
D, N operand can combine V, Z, P0~P9 for index addressing while word operation.

Operand \ Range	Y	M	S	WY	WM	WS	TMR	CTR	HR	OR	SR	ROR	DR	K	XR
	Y0 | Y255	M0 | M1911	S0 | S999	WY0 | WY240	WM0 | WM1896	WS0 | WS984	T0 | T255	C0 | C199	R0 | R3839	R3904 | R3967	R3968 | R4167	R5000 | R8071	D0 | D4095	1 | 511	V, Z | P0～P9
D	○	○	○	○	○	○	○	○	○	○	○*	○*	○		○
N									○			○	○	1-511	○

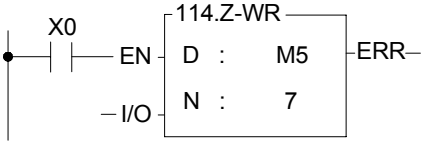
- When operation control "EN"=1 or "EN↑" (P instruction）changes from 0 to 1, it will perform the write operation according to the input status of write selection, the specified area of registers or bits will all be reset to 0 ("1/0"=0) or set to 1("1/0"=1).

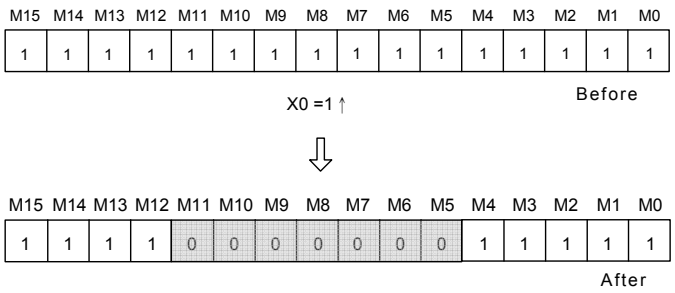
- If the N value not in the effective range (N=0, N>511), the error flag "ERR" will be set to 1,





• Above example, registers R0~R9 will be reset to 0 while X0=1.



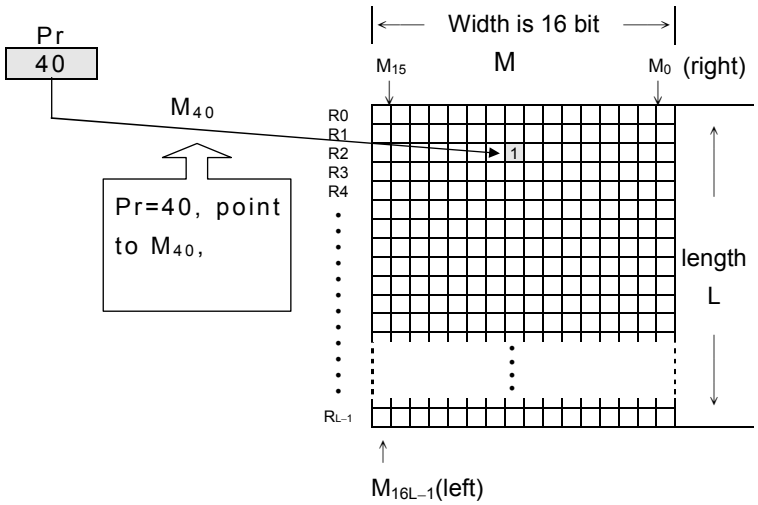


• Above example, bits M5~M11 will be reset to 0 while X0=1.

Matrix Instructions

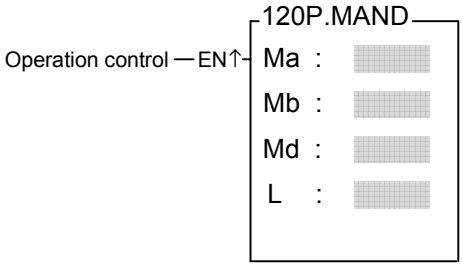
Fun No.	Functionality	Fun No.	Functionality
120	Matrix AND	126	Matrix Bit Read
121	Matrix OR	127	Matrix Bit Write
122	Matrix XOR	128	Matrix Bit Shift
123	Matrix XNOR	129	Matrix Bit Rotate
124	Matrix Inverse	130	Matrix Bit Count
125	Matrix Compare		

- A matrix is comprised of 2 or more consecutive 16-bit registers. The number of registers comprising the matrix is called the matrix length (L). One matrix altogether has L×16 bits (points), and the basic unit of the object for each operation is bit.
- The matrix instructions treats the 16×L matrix bits as a set of series points (denoted by M_0 to M_{16L-1}). Whether the matrix is formed by register or not, the operation object is the bit not numerical value.
- Matrix instructions are used mostly for discrete status processing such as moving, copying, comparing, searching, etc, of single point to multipoint (matrix), or multipoint-to-multipoint. These instructions are convenient, important for application.
- Among the matrix instructions, most instruction need to use a 16-bit register as a pointer to points a specific point within the matrix. This register is known as the matrix pointer (Pr). Its effective range is 0 to 16L-1, which corresponds respectively to the bits M_0 to M_{16L-1} within the matrix.
- Among the matrix operations, there are shift left/right, rotate left/right operations. We define the movement toward higher bit is left direction, while the movement toward lower bit is right direction, as shown in the diagram below.



FUN120 P MAND	MATRIX AND	FUN120 P MAND

Ladder symbol



Ma: Starting register of source matrix a

Mb: Starting register of source matrix b

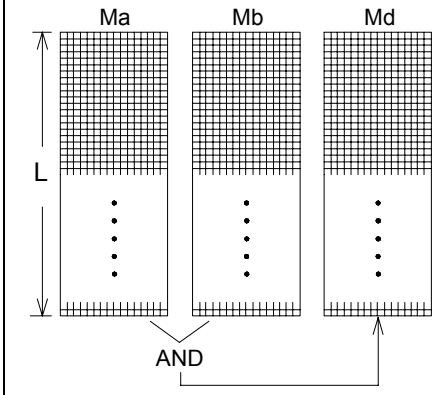
Md : Starting register of destination matrix

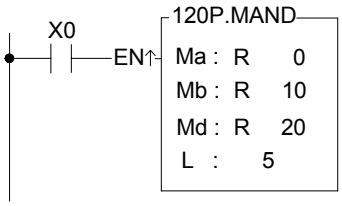
L : Length of matrix (Ma, Mb and Md)

Ma, Mb, Md may combine with V, Z, P0~P9 to serve indirect address application.

Operand \ Range	WX	WY	WM	WS	TMR	CTR	HR	IR	OR	SR	ROR	DR	K	XR
	WX0 | WX240	WY0 | WY240	WM0 | WM1896	WS0 | WS984	T0 | T255	C0 | C255	R0 | R3839	R3840 | R3903	R3904 | R3967	R3968 | R4167	R5000 | R8071	D0 | D4095	2 | 256	V, Z | P0~P9
Ma	○	○	○	○	○	○	○	○	○	○	○	○		○
Mb	○	○	○	○	○	○	○	○	○	○	○	○		○
Md		○	○	○	○	○	○		○	○*	○*	○		○
L							○				○*	○	○	

- When operation control "EN" =1 or "EN↑" (P instruction) changes from 0 to 1, this instruction will perform a logic AND (only if 2 bits are 1 will the result be 1, otherwise it will be 0) operation between two source matrixes with a length of L, Ma and Mb. The result will then be stored in the destination matrix Md, which is also the same length (the AND operation is done by bits with the same bit numbers). For example, if Ma_0 =0, Mb_0 =1, then Md_0 =0; if Ma_1 =1, Mb_1 =1, then Md_1 =1; etc, right up until AND reaches Ma_{16L-1} and Mb_{16L-1}.

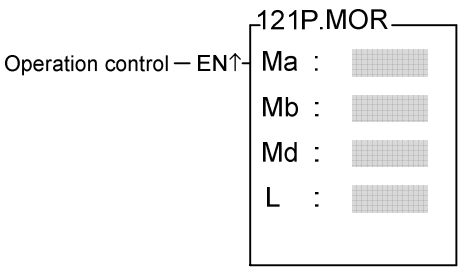




- In the program at left, when X0 goes from 0 to 1, then matrix Ma, comprised by R0 to R4, and matrix Mb, comprised by R10 to R14, will do an AND operation. The results will be stored back in matrix Md, comprised by R20 to R24. The result is shown at right in the diagram below.

FUN121 P MOR	MATRIX OR	FUN121 P MOR

Ladder symbol



Ma : Starting register of source matrix a

Mb : Starting register of source matrix b

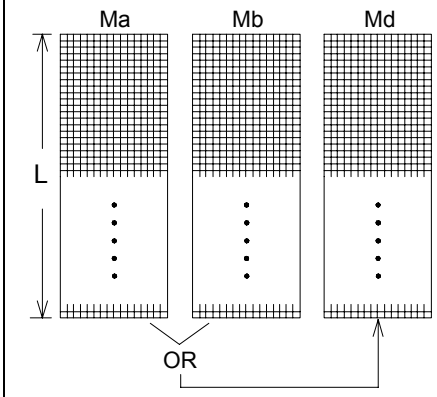
Md : Starting register of destination matrix

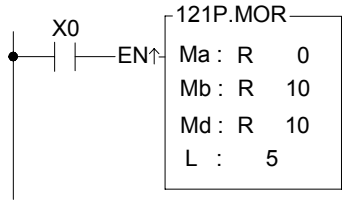
L : Length of matrix (Ma, Mb and Md)

Ma, Mb, Md may combine with V, Z, P0~P9 to serve indirect address application.

Operand \ Range	WX	WY	WM	WS	TMR	CTR	HR	IR	OR	SR	ROR	DR	K	XR
	WX0 | WX240	WY0 | WY240	WM0 | WM1896	WS0 | WS984	T0 | T255	C0 | C255	R0 | R3839	R3840 | R3903	R3904 | R3967	R3968 | R4167	R5000 | R8071	D0 | D4095	2 | 256	V, Z | P0~P9
Ma	○	○	○	○	○	○	○	○	○	○	○	○		○
Mb	○	○	○	○	○	○	○	○	○	○	○	○		○
Md		○	○	○	○	○	○		○	○*	○*	○		○
L							○				○*	○	○	

- When operation control "EN" =1 or "EN↑" (P instruction) changes from 0 to 1, this instruction will perform a logic OR(If any 2 of the bits are 1, then the result will be 1, and only if both are 0 will the result be 0) operation between 2 source matrixes with a length of L, Ma and Mb. The result will then be stored in the destination matrix Md, which is also the same length (the OR operation is done by bits with the same bit numbers). For example, if Ma_0 =0, Mb_0 =1, then Md_0 =1; if Ma_1 =0, Mb_1 =0, then Md_1 =0; etc, right up until OR reaches Ma_{16L-1} and Mb_{16L-1}.

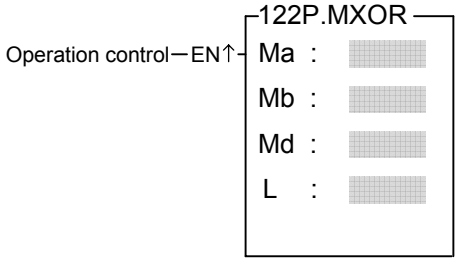




- In the program at left, when X0 goes from 0 to 1, then matrix Ma, comprised by R0 to R4, and matrix Mb, comprised by R10 to R14, will do an OR operation. The results will then be stored into the destination matrix Md, comprised by R10 to R14. In this example, Mb and Md is the same matrix, so after operation the source matrix Mb will replaced by the new value. The result is shown at right in the diagram below.

FUN122 P MXOR	MATRIX EXCLUSIVE OR（XOR）	FUN122 P MXOR

Ladder symbol



Ma: Starting register of source matrix a

Mb: Starting register of source matrix b

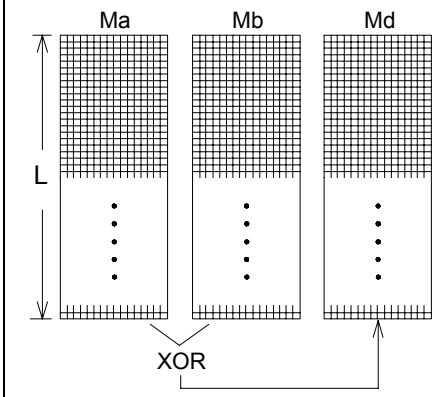
Md: Starting register of destination matrix

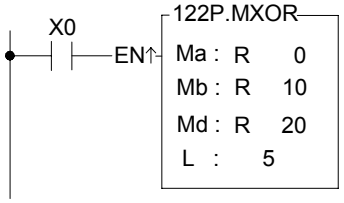
L : Length of matrix (Ma, Mb and Md)

Ma, Mb, Md may combine with V, Z, P0~P9 to serve indirect address application.

Operand \ Range	WX	WY	WM	WS	TMR	CTR	HR	IR	OR	SR	ROR	DR	K	XR
	WX0 | WX240	WY0 | WY240	WM0 | WM1896	WS0 | WS984	T0 | T255	C0 | C255	R0 | R3839	R3840 | R3903	R3904 | R3967	R3968 | R4167	R5000 | R8071	D0 | D4095	2 | 256	V, Z | P0~P9
Ma	○	○	○	○	○	○	○	○	○	○	○	○		○
Mb	○	○	○	○	○	○	○	○	○	○	○	○		○
Md		○	○	○	○	○	○		○	○*	○*	○		○
L							○				○*	○	○	

- When operation control "EN" =1 or "EN↑" (P instruction) changes from 0 to 1, this instruction will performs a logic XOR (if the 2 bits are different, then the result will be 1, otherwise it will be 0) between 2 source matrixes with a length of L, Ma and Mb. The result will then be stored back into the destination matrix Md, which also has a length of L. For example the XOR operation is done by bits with the same bit numbers - for example, if Ma_0 =0, Mb_0 =1, then Md_0 =1; if Ma_1 =1, Mb_1 =1, then Md_1 =0; etc, right up until XOR reaches Ma_{16L-1} and Mb_{16L-1}.

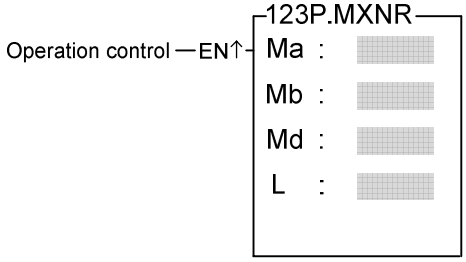




- In the program at left, when X0 goes from 0 to 1, will perform a XOR operation between matrix Ma, comprised by R0 to R4, and matrix Mb, comprised by R10 to R14. The results will then be stored in destination matrix Md, comprised by R20 to R24. The results are shown at right in the diagram below.

FUN123 P MXNR	MATRIX EXCLUSIVE NOR（XNR）	FUN123 P MXNR

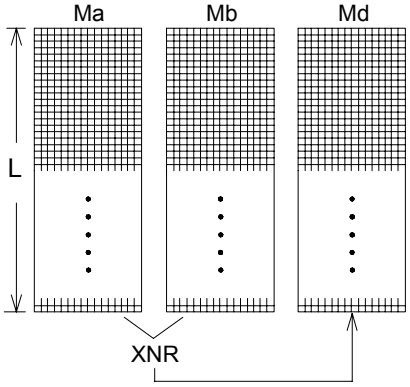
Ladder symbol



Ma : Starting register of source matrix a
Mb : Starting register of source matrix b
Md : Starting register of destination matrix
L : Length of matrix (Ma, Mb and Md)
Ma, Mb, Md may combine with V, Z, P0~P9 to serve indirect address application.

Operand \ Range	WX	WY	WM	WS	TMR	CTR	HR	IR	OR	SR	ROR	DR	K	XR
	WX0 | WX240	WY0 | WY240	WM0 | WM1896	WS0 | WS984	T0 | T255	C0 | C255	R0 | R3839	R3840 | R3903	R3904 | R3967	R3968 | R4167	R5000 | R8071	D0 | D4095	2 | 256	V, Z P0~P9
Ma	○	○	○	○	○	○	○	○	○	○	○	○		○
Mb	○	○	○	○	○	○	○	○	○	○	○	○		○
Md		○	○	○	○	○	○		○	○*	○*	○		○
L							○				○*	○	○	

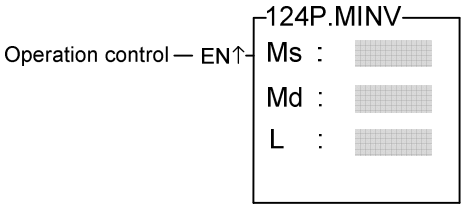
- When operation control "EN" =1 or "EN↑" (P instruction) changes from 0 to 1, will perform a logic XNR operation (if the 2 bits are the same, then the result will be 1, otherwise it will be 0)between 2 source matrixes with a length of L, Ma and Mb. The results will then be stored into the destination matrix Md, which also has the same length (the XNR operation is done by bits with the same bit numbers). For example, if Ma_0 =0, Mb_0 =1, then Md_0 =0; Ma_1 =0, Mb_1 =0, then Md_1 =1; etc, right up until XNR reaches Ma_{16L-1} and Mb_{16L-1}.



- In the program at left, when X0 goes from 0 to 1, then matrix Ma which is comprised by R0 to R4, and matrix Mb which is comprised by R10 to R14 will do an XNR operation. The results will be stored into the destination matrix Md which is comprised by R10 to R14. In this example, Mb and Md is the same matrix, so after operation the source matrix Mb will replaced by the new value. The result is shown at right in the diagram below.

FUN124 P MINV	MATRIX INVERSE	FUN124 P MINV

Ladder symbol



Ms : Starting register of source matrix

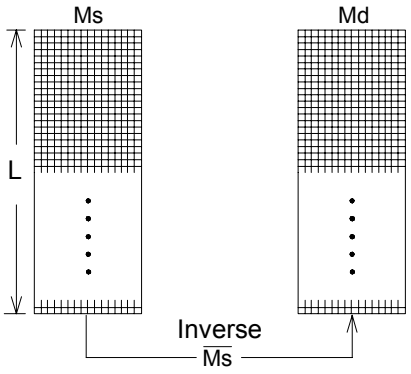
Md : Starting register of destination

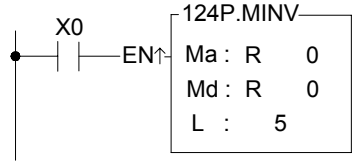
L : Length of matrix (Ms and Md)

Ma, Md may combine with V, Z, P0~P9 to serve indirect address application.

Operand \ Range	WX	WY	WM	WS	TMR	CTR	HR	IR	OR	SR	ROR	DR	K	XR
	WX0 | WX240	WY0 | WY240	WM0 | WM1896	WS0 | WS984	T0 | T255	C0 | C255	R0 | R3839	R3840 | R3903	R3904 | R3967	R3968 | R4167	R5000 | R8071	D0 | D4095	2 | 256	V, Z | P0~P9
Ms	○	○	○	○	○	○	○	○	○	○	○	○		○
Md		○	○	○	○	○	○		○	○*	○*	○		○
L							○				○*	○	○	

- When operation control "EN" =1 or "EN↑" (P instruction) changes from 0 to 1, source register Ms, which has a length of L, will be completely inverted (all the bits with a value of 1 will change to 0, and all those with a value of 0 will change to 1). The results will then be stored into destination matrix Md.

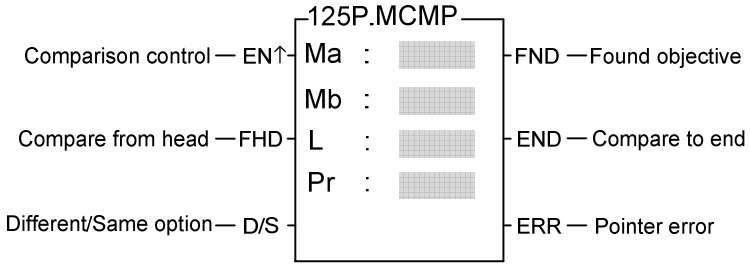




- In the program at left, when X0 goes from 0 to 1, the matrix comprised by R0 to R4 will be inverted, and then store back into itself (because in this example Ms and Md are the same matrix). The results obtained are shown at right in the diagram below.

FUN125 P MCMP	MATRIX COMPARE	FUN125 P MCMP

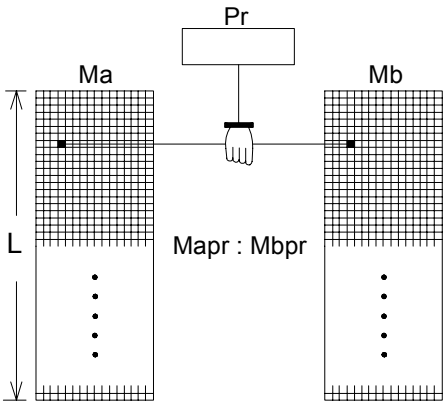
Ladder symbol



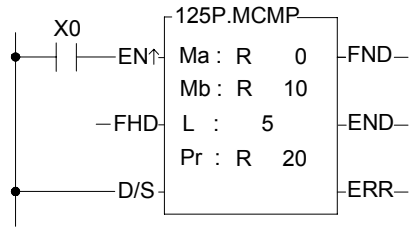
Ma: Starting register of matrix a
Mb: Starting register of matrix b
L : Length of matrix (Ma, Mb)
Pr : Pointer register
Ma, Mb may combine with V, Z, P0~P9 to serve indirect address application.

Operand \ Range	WX	WY	WM	WS	TMR	CTR	HR	IR	OR	SR	ROR	DR	K	XR
	WX0 | WX240	WY0 | WY240	WM0 | WM1896	WS0 | WS984	T0 | T255	C0 | C255	R0 | R3839	R3840 | R3903	R3904 | R3967	R3968 | R4167	R5000 | R8071	D0 | D4095	2 | 256	V, Z | P0~P9
Ma	○	○	○	○	○	○	○	○	○	○	○	○		○
Mb	○	○	○	○	○	○	○	○	○	○	○	○		○
L							○				○*	○	○	
Pr		○	○	○	○	○	○		○	○*	○*	○		

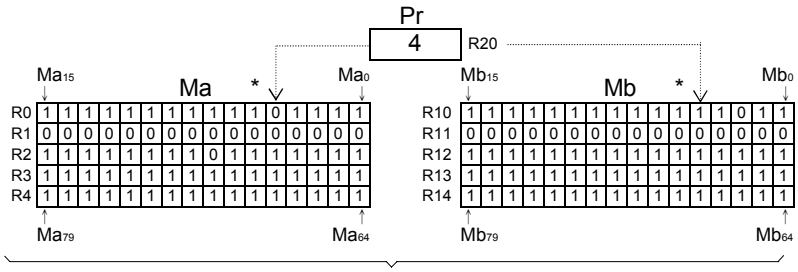
- When comparison control "EN" =1 or "EN↑" (P instruction) changes from 0 to 1, then beginning from the top pair of bits (Ma_0 and Mb_0) within the 2 matrixes Ma and Mb (when "FHD" =1 or Pr value is equal to 16L-1), or beginning from the next pair of bits (Mapr + 1 and Mbpr + 1) pointed by pointer Pr (when "FHD" =0 and Pr value is less than L-1), this instruction will compare and search for pairs of bits with different value (when D/S =1) or the same value (when D/S =0). Once match found, pointer Pr will point to the bit number in the matrix met the search condition. The found objective flag "FND" will be set to 1. When it has searched to the final pair of bits in the matrix (Ma_{16L-1}, Mb_{16L-1}), this execution of the instruction will finish, no matter it has found or not. If this happen then The compare-to-end flag "END" will be set as 1, and the Pr value will set to 16L-1 and the next time that this instruction is executed, Pr will automatically return to the starting point of the matrix (Pr =0) to begin the comparison search.

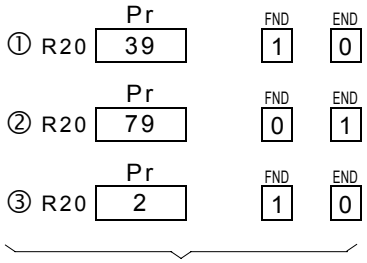


- The range for the pointer value is 0 to 16L-1. The Pr value should not be changed by other instructions, as this will affect the result of search. If the Pr value exceeds its range, then the pointer error flag "ERR" will be set to 1, and this instruction will not be carried out.



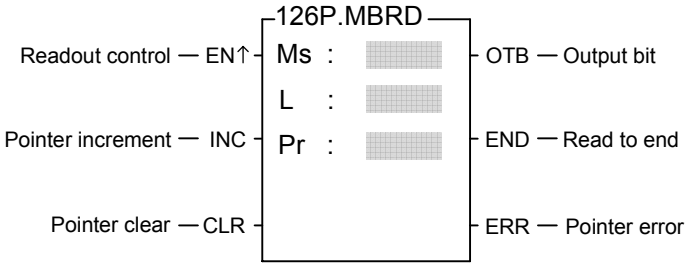
- In the program at left, the "FHD" input is 0, so starting from a position 1 greater than the pointer value at that time (marked by *), the instruction will do a search for bits with different status (because D/S =1). When X0 has a transition from 0 to 1 three times, the results are shown at right in the diagram below.





FUN126 P MBRD	MATRIX BIT READ	FUN126 P MBRD

Ladder symbol



Ms : Starting register of matrix

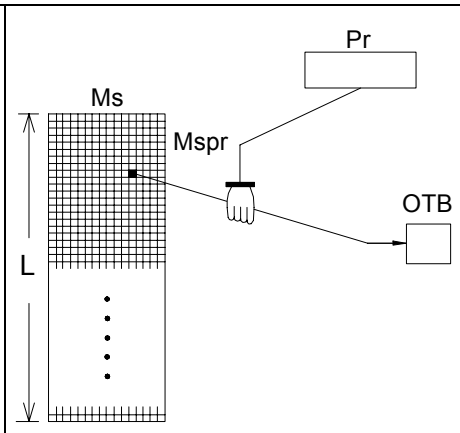
L : Matrix length

Pr : Pointer register

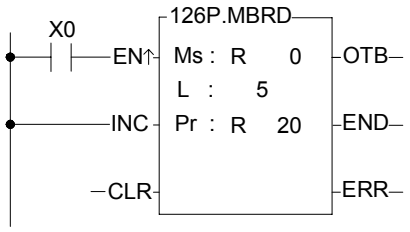
Ms may combine with V, Z, P0~P9 to serve indirect address application.

Operand \ Range	WX	WY	WM	WS	TMR	CTR	HR	IR	OR	SR	ROR	DR	K	XR
	WX0 | WX240	WY0 | WY240	WM0 | WM1896	WS0 | WS984	T0 | T255	C0 | C255	R0 | R3839	R3840 | R3903	R3904 | R3967	R3968 | R4167	R5000 | R8071	D0 | D4095	2 | 256	V, Z | P0~P9
Ms	○	○	○	○	○	○	○	○	○	○	○	○		○
L							○				○*	○	○	
Pr		○	○	○	○	○	○		○	○*	○*	○		

- When readout control "EN" =1 or "EN↑" (P instruction) changes from 0 to 1, the status of the bit Mspr pointed by pointer Pr within matrix Ms will be read out and appear at the output bit "OTB". Before the readout, this instruction will first check the input -pointer clear "CLR". If "CLR" is 1, then the Pr value will be cleared to 0 first before the readout action is carried out. After the readout is completed, If the Pr value has already reached 16L-1 (the final bit), then the read-to-end flag "END" will be set to 1. If Pr is less than 16L-1, then the status of pointer increment "INC" will be checked. If "INC" is 1, then Pr will be increased by 1. Besides this, pointer clear "CLR" can execute independently, and is not affected by other input.

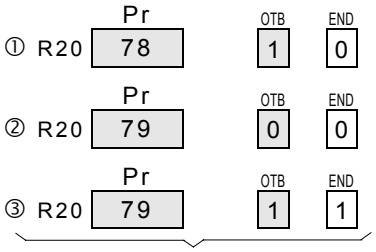


- The effective range of the pointer is 0 to 16L-1. Beyond this range the pointer error flag "ERR" will be set to 1, and this instruction will not be carried out.



- In the program at left, INC =1, so every time there is one readout the pointer will be increased by 1. With this way each bit in Ms may be read out successively, as shown at left in the diagram below. When X0 goes 3 times from 0 to 1, the results are shown at right in the diagram below .

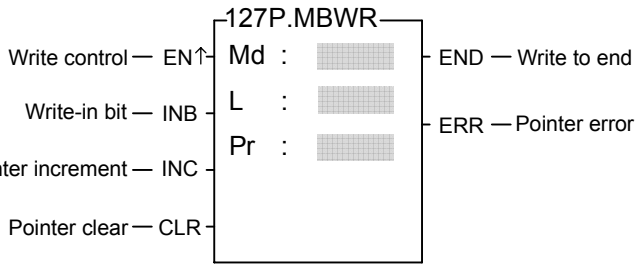
Before execution



Execution result

FUN127 P MBWR	MATRIX BIT WRITE	FUN127 P MBWR

Ladder symbol



Md : Starting register of matrix

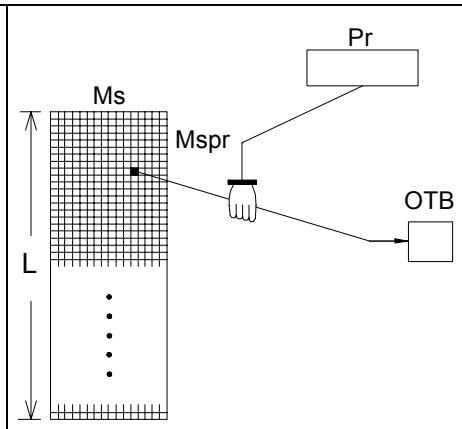
L : Matrix length

Pr : Pointer register

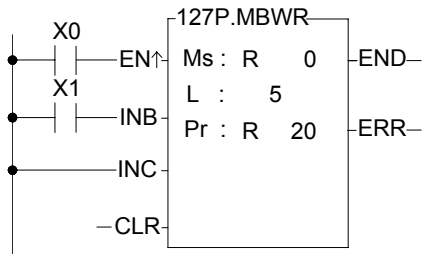
Md may combine with V, Z, P0~P9 to serve indirect address application.

Operand \ Range	WY	WM	WS	TMR	CTR	HR	OR	SR	ROR	DR	K	XR
	WY0 | WY240	WM0 | WM1896	WS0 | WS984	T0 | T255	C0 | C255	R0 | R3839	R3904 | R3967	R3968 | R4167	R5000 | R8071	D0 | D4095	2 | 256	V, Z P0~P9
Md	○	○	○	○	○	○	○	○	○	○		○
L						○			○*	○	○	
Pr	○	○	○	○	○	○	○	○*	○*	○		

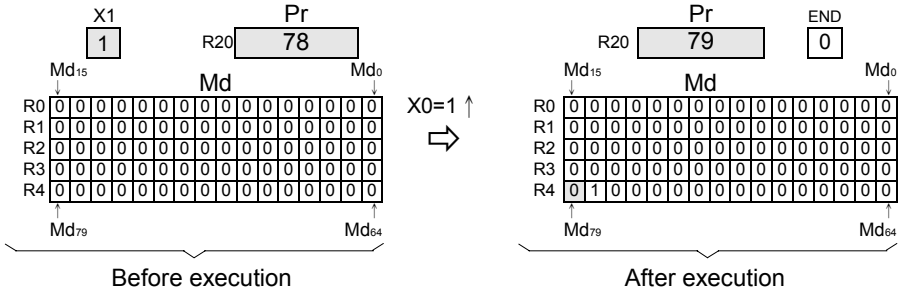
- When write control "EN" =1 or "EN↑" (P instruction) changes from 0 to 1, the status of the write-in bit "INB" will be written into the bit Mdpr pointed by pointer Pr within matrix Md. Before the write-in takes place, the status of pointer clear "CLR" will be checked. If "CLR" is 1, then Pr will be cleared to 0 before the write-in action. After the write-in action has been completed, the Pr value will be checked again. If the Pr value has already reached 16L-1 (last bit), then the write-to-end flag will be set to 1. If the Pr value is less than 16L-1 and "INC" is 1, then the pointer will increased by 1. Besides this, pointer clear "CLR" can execute independently, and is not affected by other input.



- The effective range of Pr is 0 to 16L-1. Beyond this range, the pointer error flag "ERR" will be set to 1 and this instruction will not be carried out.

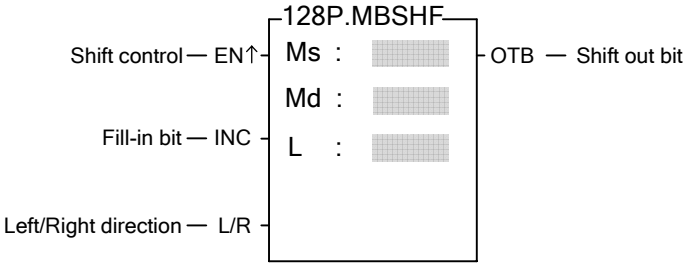


- In the program at left, pointer will be increased each time execution (because "INC" is 1). As shown in the diagram below, when X0 has a transition from 0 to 1, the status of INB (X1) will be written into the Mdpr (Md_{78}) position, and pointer Pr will increased by 1 (changing to 79). In this case, although Pr is pointing to the end, it has not yet been written into Md_{79}, so "END" flag is still 0. Only the next attempt to write to Md_{79} will set “END” to 1.



FUN128 P MBSHF	MATRIX BIT SHIFT	FUN128 P MBSHF

Ladder symbol



Ms : Starting register of source matrix

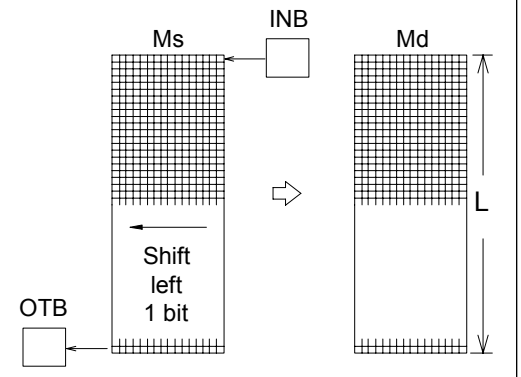
Md : Starting register of destination matrix

L : Length of matrix (Ms and Md)

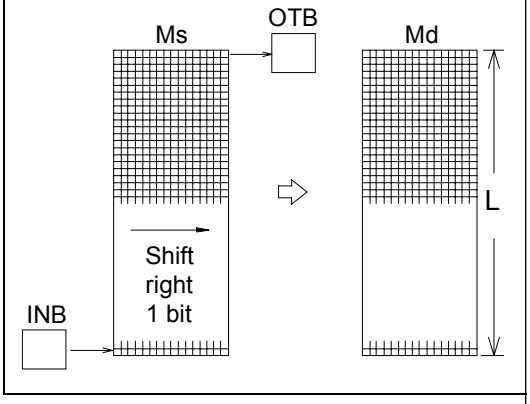
Ms, Md may combine with V, Z, P0~P9 to serve indirect address application.

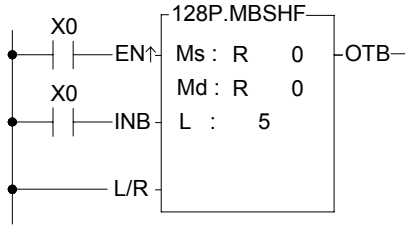
Operand \ Range	WX	WY	WM	WS	TMR	CTR	HR	IR	OR	SR	ROR	DR	K	XR
	WX0 | WX240	WY0 | WY240	WM0 | WM1896	WS0 | WS984	T0 | T255	C0 | C255	R0 | R3839	R3840 | R3903	R3904 | R3967	R3968 | R4167	R5000 | R8071	D0 | D4095	2 | 256	V, Z | P0~P9
Ms	○	○	○	○	○	○	○	○	○	○	○	○		○
Md		○	○	○	○	○	○		○	○*	○*	○		○
L							○				○*	○	○	

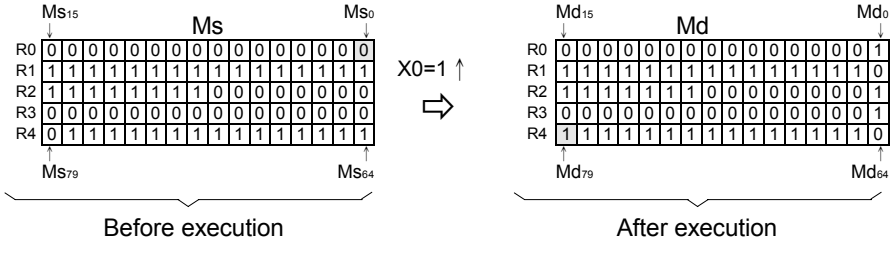
- When shift control "EN" =1 or "EN↑" (P instruction) changes from 0 to 1, source matrix Ms will be retrieved and completely shifted one position to the left (when L/R =1) or one position to the right (when L/R =0). The space caused by the shift (with a left shift it will be M_0, and with a right shift it will be M_{16L-1}), is replaced by the status of fill-in bit "INB". The status of the bits popped out (with a left shift it will be M_{16L-1}, and with a right shift it will be M_0) will appear at the output bit "OTB". Then the results of this shifted matrix will be filled into the destination matrix Md.



- The program as below is an example where Ms and Md are the same matrix. When X0 goes from 0 to 1, Ms will be completely retrieved and moved to the left (because L/R = 1) by 1 bit. It will then be stored back to Md, and the results are shown at right in the diagram below.

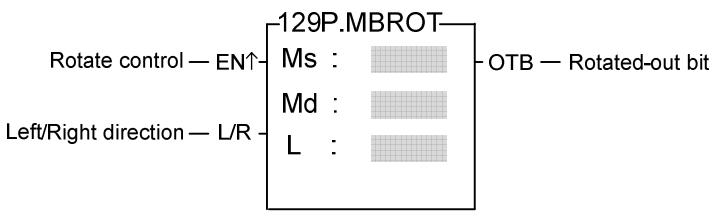






FUN129 P MBROT	MATRIX BIT ROTATE	FUN129 P MBROT

Ladder symbol



Ms : Starting register of source matrix

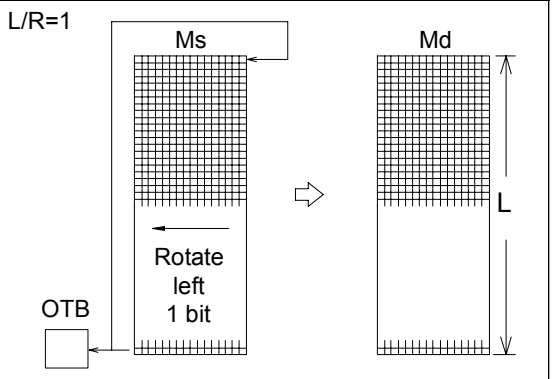
Md : Starting register of destination matrix

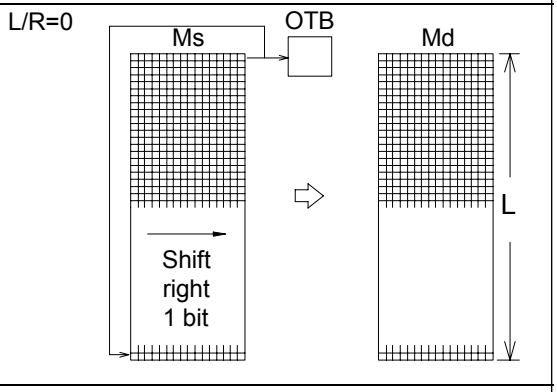
L : Length of matrix (Ms and Md)

Ms, Md may combine with V, Z, P0~P9 to serve indirect address application.

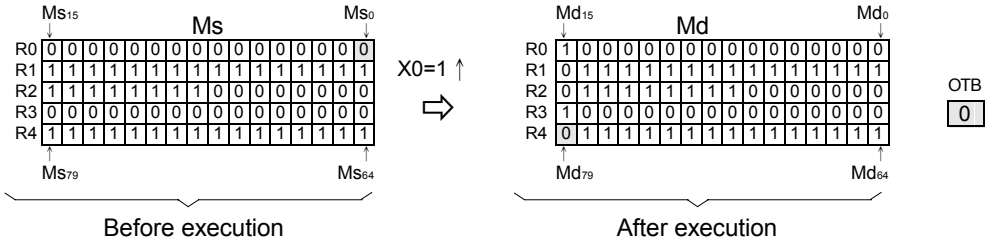
Operand \ Range	WX	WY	WM	WS	TMR	CTR	HR	IR	OR	SR	ROR	DR	K	XR
	WX0 | WX240	WY0 | WY240	WM0 | WM1896	WS0 | WS984	T0 | T255	C0 | C255	R0 | R3839	R3840 | R3903	R3904 | R3967	R3968 | R4167	R5000 | R8071	D0 | D4095	2 | 256	V, Z | P0~P9
Ms	○	○	○	○	○	○	○	○	○	○	○	○		○
Md		○	○	○	○	○	○		○	○*	○*	○		○
L							○				○*	○	○	

- When rotate control "EN" =1 or "EN↑" (P instruction) changes from 0 to 1, matrix Ms will be completely retrieved and rotated by one bit towards the left (when L/R =1) or to the right (when L/R =0). The space created by the rotation (with a left rotation it will be M0, and with a right rotation it will be M_{16L-1}) will be replaced by the status of the rotated-out bit (with a left rotation it will be M_{16L-1}, and with a right rotation it will be M0). The rotated-out bit will not only be used to fill the above-mentioned space, it will also be transferred to rotated-out bit "OTB".



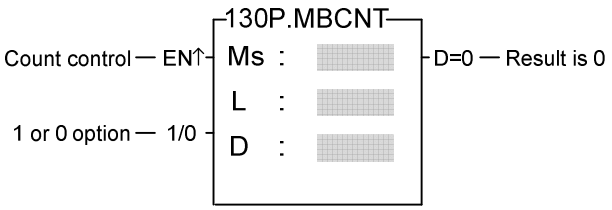


- In the program at below, Ms and Md are the same matrix. When X0 goes from 0 to 1, then the whole of Ms is retrieved and rotated right (because L/R =0) by 1 bit. It is then stored back into Ms itself (because in this example Ms and Md are the same matrix). The results are shown at right in the diagram below.



FUN130 P MBCNT	MATRIX BIT STATUS COUNT	FUN130 P MBCNT

Ladder symbol



Ms : Starting register of matrix

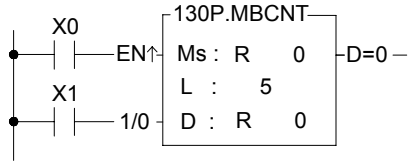
L : Matrix length

D : Register storing count results

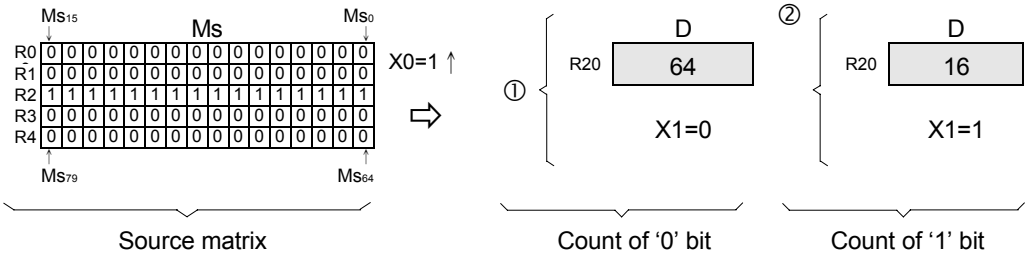
Ms may combine with V, Z, P0~P9 to serve indirect address application.

Range / Operand	WX	WY	WM	WS	TMR	CTR	HR	IR	OR	SR	ROR	DR	K	XR
	WX0 | WX240	WY0 | WY240	WM0 | WM1896	WS0 | WS984	T0 | T255	C0 | C255	R0 | R3839	R3840 | R3903	R3904 | R3967	R3968 | R4167	R5000 | R8071	D0 | D4095	2 | 256	V, Z | P0~P9
Ms	○	○	○	○	○	○	○	○	○	○	○	○		○
L							○				○*	○	○	
D		○	○	○	○	○	○		○	○*	○*	○		

- When count control "EN" =1 or "EN↑" (P instruction) changes from 0 to 1, then among the 16L bits of the Ms matrix, this instruction will count the total amount of bits with a status of 1 (when input "1/0" =1) or the total amount of bits with a status of 0 (when input "1/0" =0). The results of the counting will be stored into the register specified by D. If the value of these amounts is 0, then the Result-is-0 flag "D =0" will be set to 1.

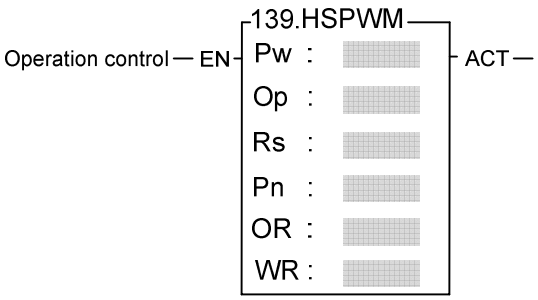


- The program at left sets X1 first as 0 (to count bits with status of 0) and then as 1 (to count bits with status of 1) and let the signal X0 has a transition from 0 to 1 for both case, the execution results are shown at right in the diagram below .



FUN 139 HSPWM	HIGH SPEED PULSE WIDTH MODULATION OUTPUT	FUN 139 HSPWM

Ladder symbol



PW : PWM output (0 = Y0, 1 = Y2, 2 = Y4, 3 = Y6)

OP : Output polarity ; 0 = Normal
1 = Inverse of output
Rs : Resolution ; 0 = 1/100 (1%)
1 = 1/1000 (0.1%)
Pn : Setting of output frequency(0~255)
OR : Setting register of output pulse width (0~100 or 0~1000)
WR : Working register

Operand \ Range	Y	WX	WY	WM	WS	TMR	CTR	HR	IR	OR	SR	ROR	DR	K
	Yn of main unit	WX0 | WX240	WY0 | WY240	WM0 | WM1896	WS0 | WS984	T0 | T255	C0 | C255	R0 | R3839	R3840 | R3903	R3904 | R3967	R3968 | R4167	R5000 | R8071	D0 | D4095	
Pw	○													0～3
Op														0～1
Rs														0～1
Pn		○	○	○	○	○	○	○	○	○	○	○	○	0～255
OR								○				○	○	0～1000
WR			○	○	○	○	○	○		○	○	○	○	

Description

- The setting of resolution (Rs) must be same between output0 (Y0) and output1 (Y2) also the setting of output frequency (Pn). It means both output0 and output1 have the same output frequency and the same output resolution, only the pulse width can be different. Same principle for output2 (Y4) and output3 (Y6).
- When operation control “EN” =1, the specified digital output will perform the PWM output, the expression for output frequency as shown bellow:

1. $f_{pwm} = \frac{184320}{(P_n + 1)}$ while Rs(Resolution)=1/100

2. $f_{pwm} = \frac{18432}{(P_n + 1)}$ while Rs(Resolution)=1/1000

Example 1 : If Pn (Setting of output frequency) = 50, Rs = 0(1/100), then

$$f_{pwm} = \frac{184320}{(50 + 1)} = 3614.117 \cdots \fallingdotseq 3.6KHz$$

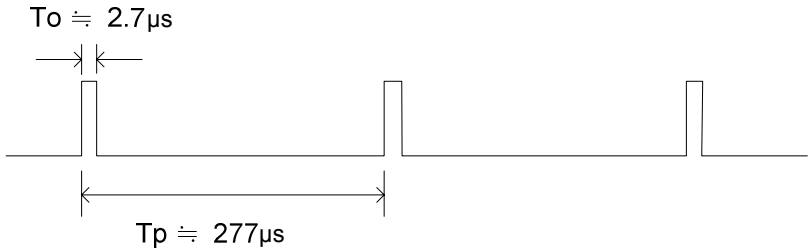
$$T(Period) = \frac{1}{f_{pwm}} \fallingdotseq 277\mu s$$

For Rs =1/100, if OR(Setting of output pulse width) =1, then T0≒2.7μs ; if OR(Setting of output pulse width) = 50, then To≒140μs .

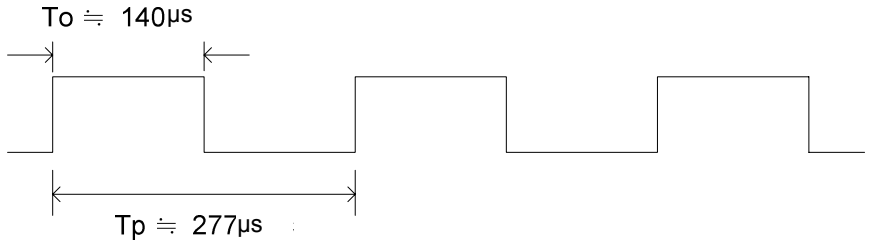
.Output waveform :

(1). Pn (Output frequency) =50, Rs =0 (1/100), OR (Output pulse width) =1 :

FUN 139 HSPWM	HIGH SPEED PULSE WIDTH MODULATION OUTPUT	FUN 139 HSPWM



(2) .Pn (Output frequency) =50, Rs = 0 (1/100), OR (Output pulse width) =50 :



Example 2 : If Pn (Setting of output frequency) =200, Rs = 1(1/1000), then

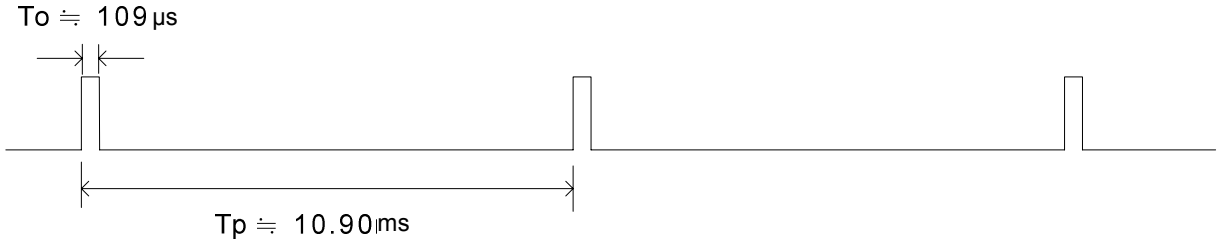
$$f_{pwm} = \frac{18432}{(200+1)} \fallingdotseq 91.7Hz$$

$$T(Period) = \frac{1}{f_{pwm}} \fallingdotseq 10.9ms$$

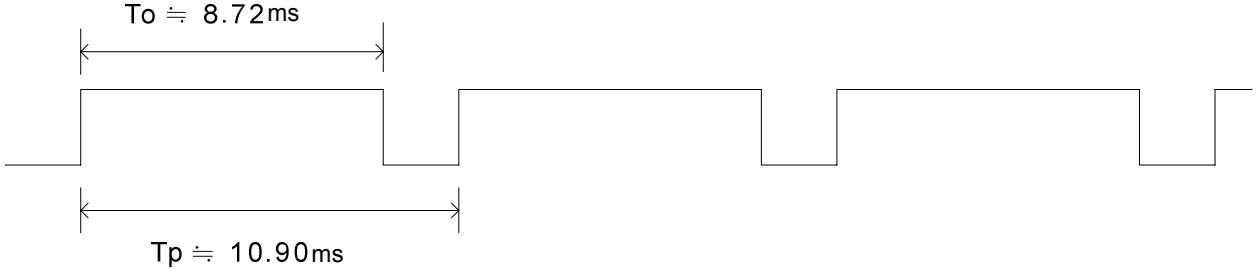
For Rs =1/1000, if OR(Setting of output pulse width) =10, then T0≒109μs; if OR(Setting of output pulse width) =800, then To≒8.72ms

.Output waveform :

(1). Pn (Output frequency) =200, Rs =1 (1/1000), OR (Output pulse width) =10 :

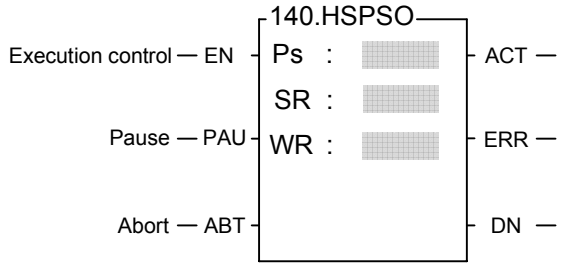


(2) .Pn (Output frequency) =200, Rs =1 (1/1000), OR (Output pulse width) =800 :



FUN140 HSPSO	HIGH SPEED PULSE OUTPUT INSTRUCTION (Brief description on function)	FUN140 HSPSO

Ladder symbol



Ps : The Pulse Output (0～3) selection
0:Y0 & Y1
1:Y2 & Y3
2:Y4 & Y5
3:Y6 & Y7
SR : Positioning program starting register.
WR : Starting working register of instruction operation, total 7 registers, can not used in any other part of program.

Operand \ Range	HR	DR	ROR	K
	R0 | R3839	D0 | D4095	R5000 | R8071	2 | 256
Ps				0～3
SR	○	○	○	
WR	○	○	○*	

Command descriptions

- The NC positioning program of HSPSO (FUN140) instruction is a program written and edited with text. The executing unit of program is divided by step (which includes output frequency, traveling distance, and transferring conditions). For one FUN140 instruction, can program 250 steps of positioning points at the most. Each step of positioning program requires 9 registers. For detailed application, please refer to chapter 13 “the NC positioning control of WSZ-controller”.
- The benefits of storing the positioning program in the register is that, while in application which use the HMI (human machine interface) as the operation console can save the positioning programs to HMI. Whenever the change of the positioning programs is requested, the download of positioning program can be simply done by a series of write register commands.
- The NC positioning of this instruction doesn’t provide the linear interpolation function.
- When execution control “EN”=1, if Ps0～3 is not controlled by other FUN140 instruction (the status of Ps0=M1992, Ps1=M1993, Ps2=M1994, and Ps3=M1995 is ON respectively), it will start to execute from the next step of positioning point (when goes to the last step, it will be restarted from the first step); if Ps0～3 are controlled by other FUN140 instruction (the status of Ps0=M1992, Ps1=M1993, Ps2=M1994, and Ps3=M1995 are OFF), this instruction will wait and acquires the control right of output point immediately right after other FUN140 release the output.
- When execution control “EN” =0, it stops the pulse output immediately.
- When output pause “PAU” =1 and execution control was 1, it will pause the pulse output. When output pause “PAU” =0 and execution control is still 1, it will continue the unfinished pulse output.
- When output abort “ABT”=1, it will halt and stop pulse output immediately. (When the execution control input “EN” becomes 1 next time, it will restart from the first step of positioning point to execute.)
- While send the output pulse, the output indication “ACT” is ON.
- When there is an execution error, the output indication “ERR” will be ON. (The error code is stored in the error code register.)
- When the execution of each step of positioning program is completed, the output indication “DN” will be ON.

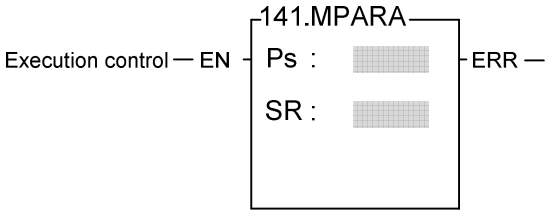
*** The working mode of Pulse Output must be configured (without setting, Y0～Y7 will be treated as normal output) to any one of following modes, before the HSPSO instruction can be worked.

U/D Mode: Y0 (Y2, Y4, Y6), as up pulse.
Y1 (Y3, Y5, Y7), as down pulse.
P/R Mode: Y0 (Y2, Y4, Y6), as the pulse out.
Y1 (Y3, Y5, Y7), as the direction.
A/B Mode: Y0 (Y2, Y4, Y6), as A phase pulse.
Y1 (Y3, Y5, Y7), as B phase pulse.

- The output polarity for Pulse Output can select to be Normally ON or Normally OFF.
- The working mode of Pulse Output can be configured by WinProladder in “Output Setup” setting page.

FUN141 MPARA	NC POSITIONING PARAMETER VALUE SETTING (Brief description on function)	FUN141 MPARA

Ladder symbol



Ps : The pulse output (0～3) selection

SR : Starting register for parameter table; it has 18 parameters totally, and occupy 24 registers.

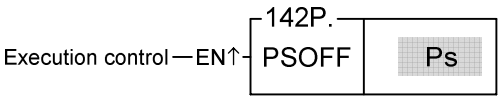
Range / Operand	HR	DR	ROR	K
	R0 | R3839	D0 | D4095	R5000 | R8071	2 | 256
Ps				0～3
SR	○	○	○	

Command descriptions

- It is not necessary to use this instruction. If the system default for parameter values is matching what user demanded, then this instruction is not needed. However, if it needs to change the parameter value dynamically, this instruction is required.
- This instruction incorporates with FUN140 or FUN147 for positioning control purpose.
- Whether the execution control “EN” =0 or 1, this instruction will be performed.
- When there are any errors in parameter value, the output indication “ERR” will be ON. (The error code is stored in the error code register.)
- For detailed functional description and usage, please refer to chapter 13 “The NC positioning control of WSZ-controller” for explanation.

FUN142 P PSOFF	STOP THE HSPSO PULSE OUTPUT (Brief description on function)	FUN142 P PSOFF

Ladder symbol



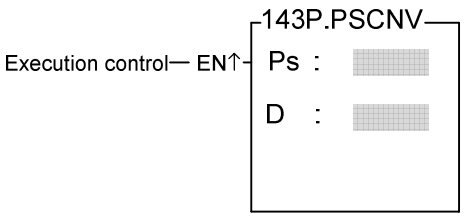
Ps : 0～3
Enforce the Pulse Output PSOn (n= Ps) to stop.

Command descriptions

- When execution control “EN” =1 or “EN↑” (P instruction) changes from 0 to 1, this instruction will enforce the assigned number set of HSPSO (High Speed Pulse Output) to stop pulse output.
- While in the application for mechanical original point reset, as soon as reach the original point can use this instruction to stop the pulse output immediately, so as to make the original point stop at the same position every time when performing mechanical original point resetting.
- For detailed functional description and usage, please refer to chapter 13 “The NC positioning control of WSZ-controller” for explanation.

FUN143 P PSCNV	CONVERT THE CURRENT PULSE VALUE TO DISPLAY VALUE (mm, Deg, Inch, PS) (Brief description on function)	FUN143 P PSCNV

Ladder symbol



Ps : 0～3; it converts the number of the pulse position to be the mm (Deg, Inch, PS) that has same unit as the set value, so as to make current position displayed.

D : Register that stores the current position after conversion. It uses 2 registers, e.g. if D = D10, which means D10 is Low Word and D11 is High Word.

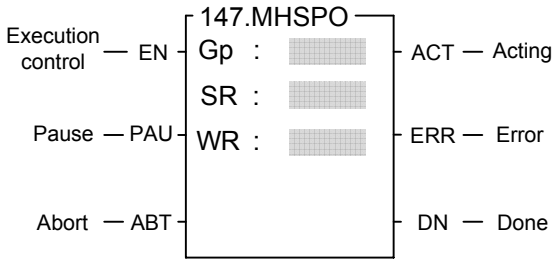
Operand \ Range	HR	DR	ROR	K
	R0 | R3839	D0 | D4095	R5000 | R8071	2 | 256
Ps				0 ～3
D	○	○	○	

Command descriptions

- When execution control “EN” =1 or “EN↑” (P instruction) changes from 0 to 1, this instruction will convert the assigned current pulse position (PS) to be the mm (or Deg, Inch, or PS) that has same unit as the set value, so as to make current position displaying.
- Only when the FUN140 instruction is executed, then it can get the correct conversion value by executing this instruction.
- For detailed functional description and usage, please refer to chapter 13 “The NC positioning control of WSZ-controller” for explanation.

FUN 147 MHSPO	MULTI-AXIS HIGH SPEED PULSE OUTPUT (Brief description on function)	FUN 147 MHSPO

Ladder symbol



Gp: Group number (0～1)

SR: Starting register for positioning program (example explanation)

WR: Starting register for instruction operation (example explanation). It controls 9 registers, which the other program cannot repeat in using.

Operand \ Range	HR	DR	ROR	K
	R0 | R3839	D0 | D4095	R5000 | R8071	
Gp				0～1
SR	○	○	○	
WR	○	○	○*	

Instruction Explanation

1. The FUN147 (MHSPO) instruction is used to support the linear interpolation for multi-axis motion control, it consists of the motion program written and edited with text programming. We named every position point as a step (which includes output frequency, traveling distance, and transfer conditions). Every step of positioning point owns 15 registers for coding.
2. The FUN147 (MHSPO) instruction can support up to 4 axes for simultaneous linear interpolation; or 2 sets of 2-axis linear interpolation (i.e. Gp0 = Axes Ps0 & Ps1; Gp1 = Axes Ps2 & Ps3)
3. The best benefit to store the positioning program into the registers is that in the case of association with MMI (Man Machine Interface) to operate settings, it may save and reload the positioning program via MMI when replacing the molds.
4. When execution control “EN” =1, if the other FUN147/FUN140 instructions to control Ps0～3 are not active (corresponding status of Ps0=M1992, Ps1=M1993, Ps2=M1994, and Ps3=M1995 will be ON), it will start to execute from the next step of positioning point (when goes to the last step, it will be restarted from the first step to perform); if Ps0～3 is controlled by other FUN147/FUN140 instruction (corresponding status of Ps0=M1992, Ps1=M1993, Ps2=M1994, and Ps3=M1995 would be OFF), this instruction will acquire the pulse output right of positioning control once the controlling FUN147/FUN140 has released the control right.
5. When execution control “EN” =0, it stops the pulse output immediately.
6. When output pause “PAU” =1 and execution control “EN” was 1 beforehand, it will pause the pulse output. When output pause “PAU” =0 and execution control is still 1, it will continue the unfinished pulse output.
7. When output abort “ABT”=1, it stops pulse output immediately. (When the execution control input “EN” becomes 1 next time, it will restart from the first step of positioning point to execute.)
8. While the pulse is in output transmitting, the output indication “ACT” is ON.
9. When there is execution error, the output indication “ERR” will be ON.
   (The error code is stored in the error code register.)
10. When each step of positioning point is complete, the output indication “DN” will be ON.

FUN 147 MHSPO	MULTI-AXIS HIGH SPEED PULSE OUTPUT (Brief description on function)	FUN 147 MHSPO

*** The working mode of Pulse Output must be set (without setting, Y0～Y7 will be treated as general output) to be one of U/D, or A/B mode, thus the Pulse Output may have a regular output.

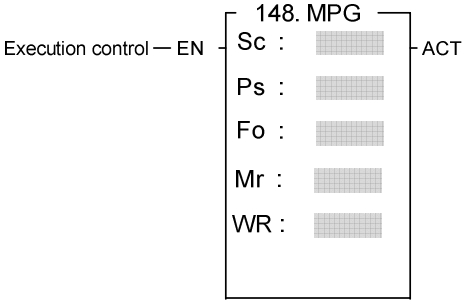
U/D mode: Y0 (Y2, Y4, Y6), it sends out upward counting pulse.
Y1 (Y3, Y5, Y7), it sends out downward counting pulse.

A/B mode: Y0 (Y2, Y4, Y6), it sends out the phase A pulse.
Y1 (Y3, Y5, Y7), it sends out the phase B pulse.

- The output polarity for Pulse Output can select to be Normal ON or Normal OFF.

FUN148 MPG	MANUAL PULSE GENERATOR FOR POSITIONING (Brief description on function)	FUN148 MPG

Ladder symbol



Sc: Source of high speed counter; 0~7

Ps: Axis of pulse output; 0~3

Fo: Setting of output speed (2 registers)

Mr: Setting of multiplier (2 registers)

Mr+0: Multiplicand (Fa)

Mr+1: Dividend (Fb)

WR: Starting address of working registers, it needs 4 registers, the other programs can not repeat in using.

Range / Operand	HR	ROR	DR	K
	R0 | R3839	R5000 | R8071	D0 | D4095	16 bit
Sc	○	○	○	0~7
Ps	○	○	○	0~3
Fo	○	○	○	
Mr	○	○	○	
WR	○	○*	○	

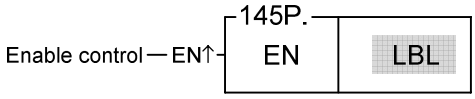
- Let this instruction be executed in 50ms fixed time interrupt service routine (50MSI)、or by using the 0.1ms high speed timer to generate 50ms fixed time interrupt service to have accurate repeat time to sample the pulse input from manual pulse generator. If it comes the input pulses, it will calculate the number of pulses needing to output according to the setting of multiplier (Mr+0 and Mr+1), and then outputs the pulse stream in the speed of setting (Fo) during this time interval.

  The setting of output speed (Fo) must be fast enough, and the acceleration / deceleration rate (Parameter 4 and parameter 8 of FUN141 instruction) must be sharp to guarantee it can complete the sending of pulse stream during the time interval if it is under high multiplier (100 or 200 times) situation.
- When execution control “EN” =1, this instruction will sample the pulse input from manual pulse generator by reading the current value of assigned high speed counter every time interval; it doesn’t have any output if it doesn’t have any input pulse; but If it senses the input pulses, it will calculate the number of pulses needing to output according to the setting of multiplier (Mr+0 and Mr+1), and then outputs the pulse stream in the speed of setting (Fo) during this time interval.

  Number of output pulses = (Number of input pulses × Fa) / Fb
- This instruction also under the control of hardware resource management; it wouldn’t be executed if the hardware is occupied.
- The output indicator “ACT“ =1 if it outputs the pulses; otherwise “ACT“ =0.
- For detailed functional description and usage, please refer to chapter 13 “The NC positioning control of WSZ-controller” for explanation.

FUN145 P EN	ENABLE CONTROL OF THE INTERRUPT AND PERIPHERAL	FUN145 P EN

Ladder symbol



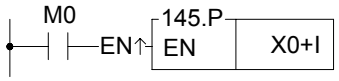
LBL : External input or peripheral label name that to be enabled.

- When enable control "EN" =1 or "EN↑" (P instruction) changes from 0 to 1, it allows the external input or peripheral interrupt action which is assigned by LBL.
- The enabled interrupt label name is as follows.

LBL name	Description	LBL name	Description	LBL name	Description
HSTAI	HSTA High speed counter interrupt	X4+I	X4 positive edge interrupt	X10+I	X10 positive edge interrupt
HSC0I	HSC0 High speed counter interrupt	X4−I	X4 negative edge interrupt	X10−I	X10 negative edge interrupt
HSC1I	HSC1 High speed counter interrupt	X5+I	X5 positive edge interrupt	X11+I	X11 positive edge interrupt
HSC2I	HSC2 High speed counter interrupt	X5−I	X5 negative edge interrupt	X11−I	X11 negative edge interrupt
HSC3I	HSC3 High speed counter interrupt	X6+I	X6 positive edge interrupt	X12+I	X12 positive edge interrupt
X0+I	X0 positive edge interrupt	X6−I	X6 negative edge interrupt	X12−I	X12 negative edge interrupt
X0−I	X0 negative edge interrupt	X7+I	X7 positive edge interrupt	X13+I	X13 positive edge interrupt
X1+I	X1 positive edge interrupt	X7−I	X7 negative edge interrupt	X13−I	X13 negative edge interrupt
X1−I	X1 negative edge interrupt	X8+I	X8 positive edge interrupt	X14+I	X14 positive edge interrupt
X2+I	X2 positive edge interrupt	X8−I	X8 negative edge interrupt	X14−I	X14 negative edge interrupt
X2−I	X2 negative edge interrupt	X9+I	X9 positive edge interrupt	X15+I	X15 positive edge interrupt
X3+I	X3 positive edge interrupt	X9−I	X9 negative edge interrupt	X15−I	X15 negative edge interrupt
X3−I	X3 negative edge interrupt				

- In practical application, some interrupt signals should not be allowed to work at sometimes, however, it should be allowed to work at some other times. Employing FUN146 (DIS) and FUN145 (EN) instructions could attain the above mentioned demand.

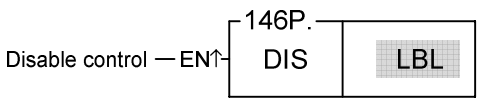
Program example



- When M0 changes from 0 to 1, it allows X0 to send interrupt when X0 changes from 0 to 1. CPU can rapidly process the interrupt service program of X0+I.

FUN146 P DIS	DISABLE CONTROL OF THE INTERRUPT AND PERIPHERAL	FUN146 P DIS

Ladder symbol



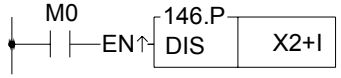
LBL : Interrupt label intended to disable or peripheral name to be disabled.

- When disable control "EN" =1 or "EN↑" (P instruction) changes from 0 to 1, it disable the interrupt or peripheral operation designated by LBL.
- The interrupt label name is as follows:

LBL name	Description	LBL name	Description	LBL name	Description
HSTAI	HSTA High speed counter interrupt	X4+I	X4 positive edge interrupt	X10+I	X10 positive edge interrupt
HSC0I	HSC0 High speed counter interrupt	X4−I	X4 negative edge interrupt	X10−I	X10 negative edge interrupt
HSC1I	HSC1 High speed counter interrupt	X5+I	X5 positive edge interrupt	X11+I	X11 positive edge interrupt
HSC2I	HSC2 High speed counter interrupt	X5−I	X5 negative edge interrupt	X11−I	X11 negative edge interrupt
HSC3I	HSC3 High speed counter interrupt	X6+I	X6 positive edge interrupt	X12+I	X12 positive edge interrupt
X0+I	X0 positive edge interrupt	X6−I	X6 negative edge interrupt	X12−I	X12 negative edge interrupt
X0−I	X0 negative edge interrupt	X7+I	X7 positive edge interrupt	X13+I	X13 positive edge interrupt
X1+I	X1 positive edge interrupt	X7−I	X7 negative edge interrupt	X13−I	X13 negative edge interrupt
X1−I	X1 negative edge interrupt	X8+I	X8 positive edge interrupt	X14+I	X14 positive edge interrupt
X2+I	X2 positive edge interrupt	X8−I	X8 negative edge interrupt	X14−I	X14 negative edge interrupt
X2−I	X2 negative edge interrupt	X9+I	X9 positive edge interrupt	X15+I	X15 positive edge interrupt
X3+I	X3 positive edge interrupt	X9−I	X9 negative edge interrupt	X15−I	X15 negative edge interrupt
X3−I	X3 negative edge interrupt				

- In practical application, some interrupt signals should not be allowed to work at certain situation. To achieve this, this instruction may be used to disable the interrupt signal.

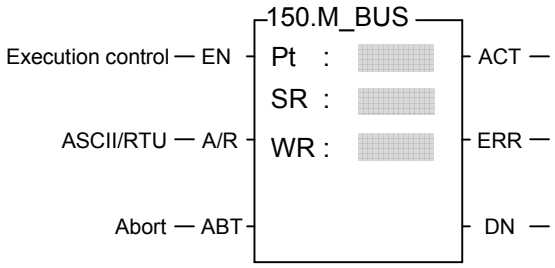
Program example



- When M0 changes from 0 to 1, it prohibits X2 from sending interrupt when X2 changes from 0 to 1.

FUN150 M-BUS	MODBUS MASTER INSTRUCTION （WHICH MAKES CONTROLLER AS THE MODBUS MASTER THROUGH PORT 1~2）	FUN150 M-BUS

Ladder symbol



Pt ：1~2, specify the communication port being acted as the Modbus master

SR：Starting register of communication program

WR：Starting register for instruction operation. It controls 8 registers, the other programs can not repeat in using.

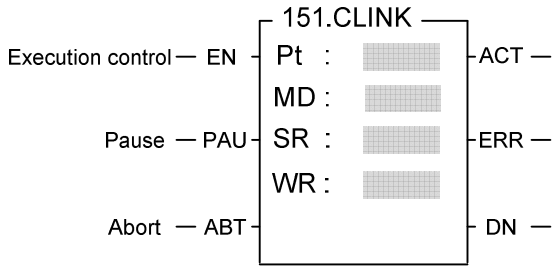
Range / Ope-rand	HR	ROR	DR	K
	R0 | R3839	R5000 | R8071	D0 | D4095	
Pt				1～2
SR	○	○	○	
WR	○	○*	○	

Description

1. FUN150 (M-BUS) instruction makes controller act as Modbus master through Port 1～2, thus it is very easy to communicate with the intelligent peripheral with Modbus protocol.
2. The master controller may connect with 247 slave stations through the RS-485 interface.
3. Only the master controller needs to use M-BUS instruction.
4. It employs the program coding method or table filling method to plan for the data flow controls; i.e. from which one of the slave station to get which type of data and save them to the master controller, or from the master controller to write which type of data to the assigned slave station. It needs only seven registries to make definition; every seven registers define one packet of data transaction.
5. When execution control “EN” changes from 0 to 1 and input Abort “ABT” is 0, and if Port 1/2 hasn’t been controlled by other communication instructions [i.e. M1960 (Port1) / M1962 (Port2) / = 1], this instruction will control the Port 1/2 immediately and set the M1960/M1962 to be 0 (which means it is being occupied), then going on a packet of data transaction immediately. If Port 1/2 has been controlled (M1960/M1962 = 0), then this instruction will enter into the standby status until the controlling communication instruction completes its transaction or abort its operation to release the control right (M1960/M1962 =1), and then this instruction will become enactive, set M1960/M1962 to be 0, and going on the data transaction immediately.
6. While in transaction processing, if operation control “ABT” becomes 1, this instruction will abort this transaction immediately and release the control right (M1960/M1962 = 1). Next time, when this instruction takes over the transmission right again, it will restart from the first packet of data transaction.
7. While “A/R” =0，Modbus RTU protocol； “A/R” =1，Modbus ASCII protocol。
8. While it is in the data transaction, the output indication “ACT” will be ON.
9. If there is error occurred when it finishes a packet of data transaction, the output indication “DN” & “ERR” will be ON.
10. If there is no error occurred when it finishes a packet of data transaction, the output indication “DN” will be ON.
11. For detailed functional description and usage, please refer to chapter 12 “The Applications for WSZ Communication Link” for explanation.

FUN 151 CLINK	COMMUNICATION LINK INSTRUCTION (WHICH MAKES CONTROLLER ACT AS THE MASTER STATION IN CPU LINK NETWORK THROUGH PORT 1~2)	FUN 151 CLINK

Ladder symbol



Pt : Assign the port, 1～2

MD : Communication mode, MD0~MD3

SR : Starting register of communication table

WR : Starting register for instruction operation. It controls 8 registers, the other programs can not repeat in using.

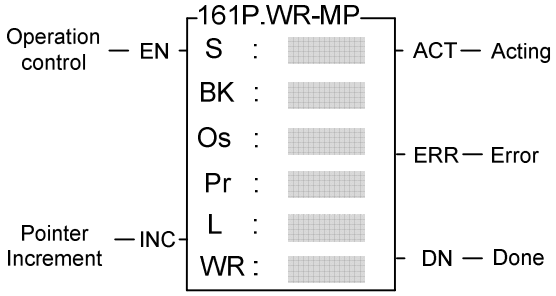
Range / Operand	HR	ROR	DR	K
	R0 | R3839	R5000 | R8071	D0 | D4095	
Pt				1～2
MD				0～3
SR	○	○	○	
WR	○	○*	○	

Description

1. This instruction provides MD0～MD3. The following are the function description of respective modes.
2. FUN151 (CLINK): MD 0, it makes controller act as the master of WSZ CPU Link Network through Port 1~ 2.
3. The master controller may connect with 254 slave stations through the RS485 interface.
4. Only the master controller needs to use FUN151 instruction, the slave doesn't need.
5. It employs the program coding method or table filling method to plan for the data flow controls; i.e. from which one of the slave station to get which type of data and save them to the master controller, or from the master controller to write which type of data to the assigned slave station. It needs only seven registries to make definition; every seven registers define one packet of data transaction.
6. When execution control “EN” changes from 0 to 1 and both inputs “PAU” and “ABT” are 0, and if Port 1/2 hasn’t been controlled by other communication instructions [i.e. M1960 (Port1) / M1962 (Port2) = 1], this instruction will control the Port 1/2 immediately and set the M1960/M1962 to be 0 (which means it is being occupied), then going on a packet of data transaction immediately. If Port 1/2 has been controlled (M1960/M1962 = 0), then this instruction will enter into the standby status until the controlling communication instruction completes its transaction or pause/abort its operation to release the control right (M1960/M1962 =1), and then this instruction will become enactive, set M1960/M1962 to be 0, and going on the data transaction immediately.
7. While in transaction processing, if operation control “PAU” becomes 1, this instruction will release the control right (M1960/M1962 = 1) after this transaction. Next time, when this instruction takes over the transmission right again, it will restart from the next packet of data transaction.
8. While in transaction processing, if operation control “ABT” becomes 1, this instruction will abort this transaction immediately and release the control right (M1960/M1962 =1). Next time, when this instruction takes over the transmission right again, it will restart from the first packet of data transaction.
9. While it is in the data transaction, the output indication “ACT” will be ON.
10. If there is error occurred when it finishes a packet of data transaction, the output indication “DN” & “ERR” will be ON.
11. If there is no error occurred when it finishes a packet of data transaction, the output indication “DN” will be ON.
12. For detailed functional description and usage, please refer to chapter 12 “The Applications for WSZ Communication Link” for explanation.

FUN161 WR-MP	Write Data Record into the MEMORY_PACK （Write memory pack）	FUN161 WR-MP

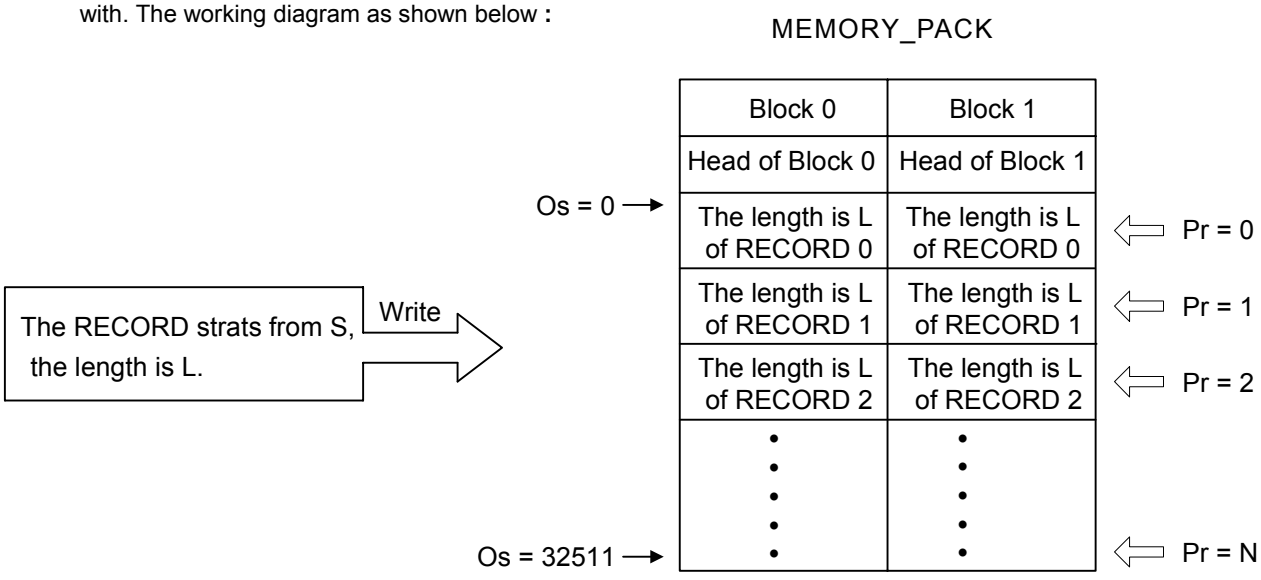
Ladder symbol



S: Starting address of the source data
BK: Block number of the MEMORY_PACK, 0～1
Os: Offset of the block
Pr: Address of the pointer
L: Quantity of writing, 1～128
WR: Starting address of working registers, it takes 2 registers
S may combine with V, Z, P0～P9 for indirect addressing application

Range / Operand	WY	WM	WS	TMR	CTR	HR	IR	OR	SR	ROR	DR	K	XR
	WY0 | WY240	WM0 | WM1896	WS0 | WS984	T0 | T255	C0 | C255	R0 | R3839	R3840 | R3903	R3904 | R3967	R3968 | R4167	R5000 | R8071	D0 | D4095		V、Z P0~P9
S						○				○	○		○
BK												0～1	
Os						○				○	○	0～32511	
Pr	○	○	○	○	○	○		○	○	○*	○		
L						○	○	○	○	○*	○	1～128	
WR						○				○*	○		

- The main purpose of the MEMORY_PACK of WSZ series' is used for long term storing of the user's ladder program, except this, through the FUN161/FUN162 instructions, the MEMORY_PACK can be worked as the portable MEMORY_PACK for machine working parameters's saving and loading.
- When execution control "EN" changes from 0 to 1, it will perform the data writing, where S is the starting address of the source data, BK is the block number of the MEMORY_PACK to store this writing, Os is the offset of specified block, Pr is the pointer to point to corresponding data area, L is the quantity of this writing. The access of MEMORY_PACK manipulation adopts the concept of RECORD data structure to implement with. The working diagram as shown below：

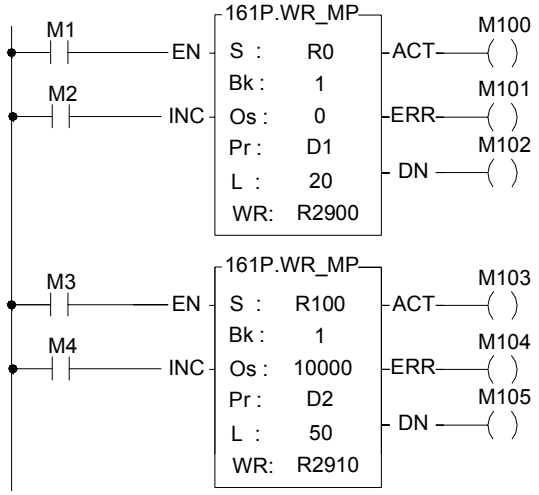


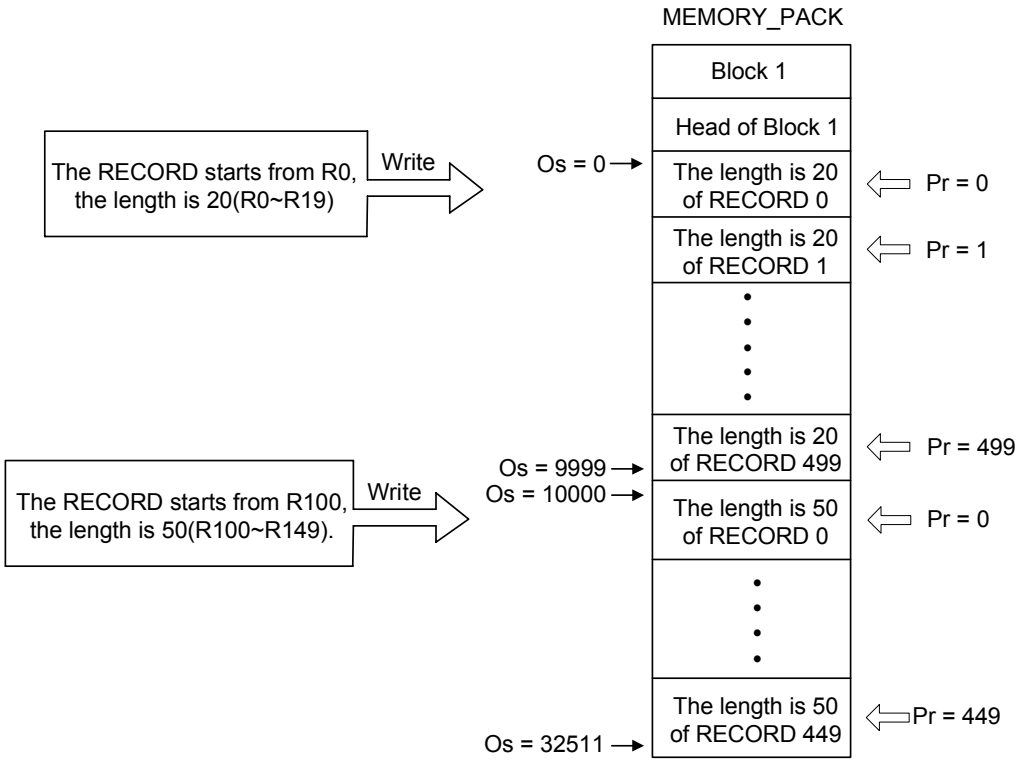
- When input "INC" = 1, the content of the pointer will be increased by one after the execution of writing, it points to next record.

FUN161 WR-MP	Write Data Record into the MEMORY_PACK （Write memory pack）	FUN161 WR-MP

- If the value of L is equal to 0 or greater than 128, or the pointed data area over the range, the output "ERR" will be 1, it will not perform the writing operation.
- It needs couple of Controller solving scans for data writing and verification; during the execution, the output "ACT" will be 1; when completing the execution and verification without the error, the output "DN" will be 1; when completing the execution and verification with the error, the output "ERR" will be 1.
- The MEMORY_PACK can be configured to store the user's ladder program or machine's working parameters, or both. The ladder program can be stored into the block 0 only, but the machine's working parameters can be stored into block 0 or 1; the memory capacity of each block has 32K Word in total.

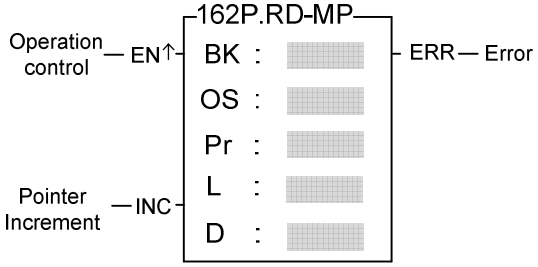
Example program: Writing the record into block 1 of MEMORY_PACK with the different length





FUN162 P RD-MP	Read Data Record from the MEMORY_PACK （Read memory pack）	FUN162 P RD-MP

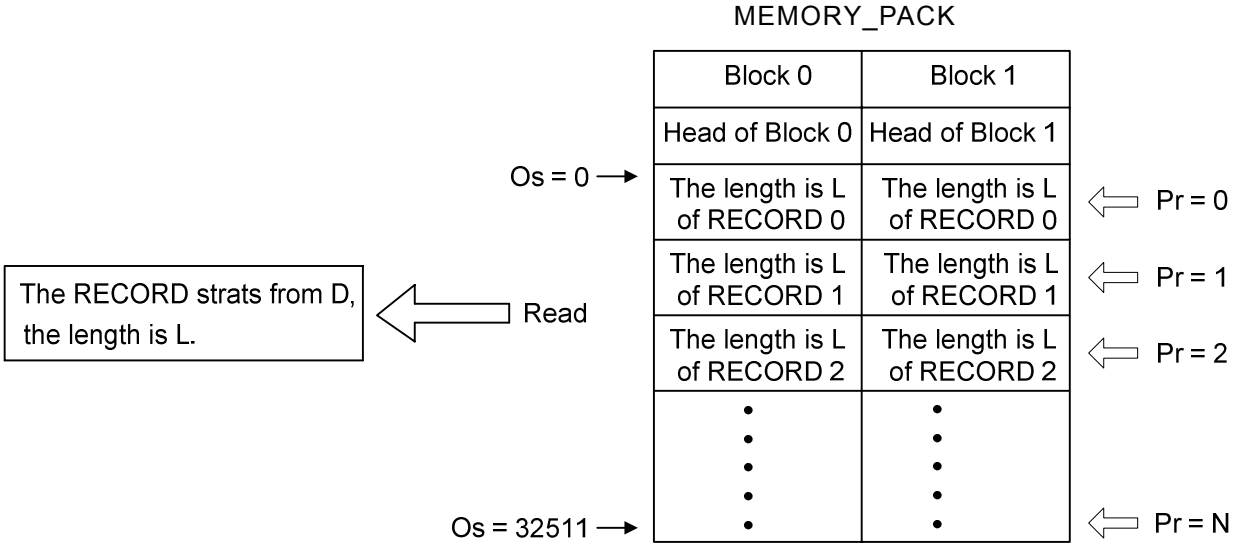
Ladder symbol



BK: Block number of the MEMORY_PACK, 0～1
Os: Offset of the block
Pr: Address of the pointer
L: Quantity of reading, 1～128
D: Starting address to store the reading record
D may combine with V, Z, P0~P9 to serve indirect address application.

Range / Operand	WY	WM	WS	TMR	CTR	HR	IR	OR	SR	ROR	DR	K	XR
	WY0 | WY240	WM0 | WM1896	WS0 | WS984	T0 | T255	C0 | C255	R0 | R3839	R3840 | R3903	R3904 | R3967	R3968 | R4167	R5000 | R8071	D0 | D4095		V、Z P0~P9
BK												0～1	
Os						○				○	○	0～32511	
Pr	○	○	○	○	○	○		○	○	○*	○		
L						○	○	○	○	○*	○	1～128	
D						○				○*	○		○

- If the MEMORY_PACK of the WSZ series' has stored the data record written by the FUN161 instruction, they can be read out for machine's working through this instruction; it will reduce the tuning time for machine operation.
- When operation control "EN" =1 or "EN↑" (P instruction) changes from 0 to 1, it will perform the data reading, where BK is the block number of the MEMORY_PACK storing the record, Os is the offset of specified block, Pr is the pointer to point to corresponding data area, L is the quantity of this record, and D is the starting address to store this reading of record. The access of MEMORY_PACK manipulation adopts the concept of RECORD data structure to implement with. The working diagram as shown below:



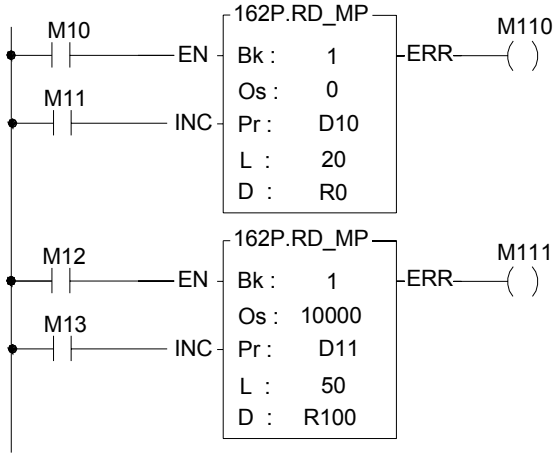
- When input "INC"=1, the content of the pointer will be increased by one after the execution of reading, it points to next record.

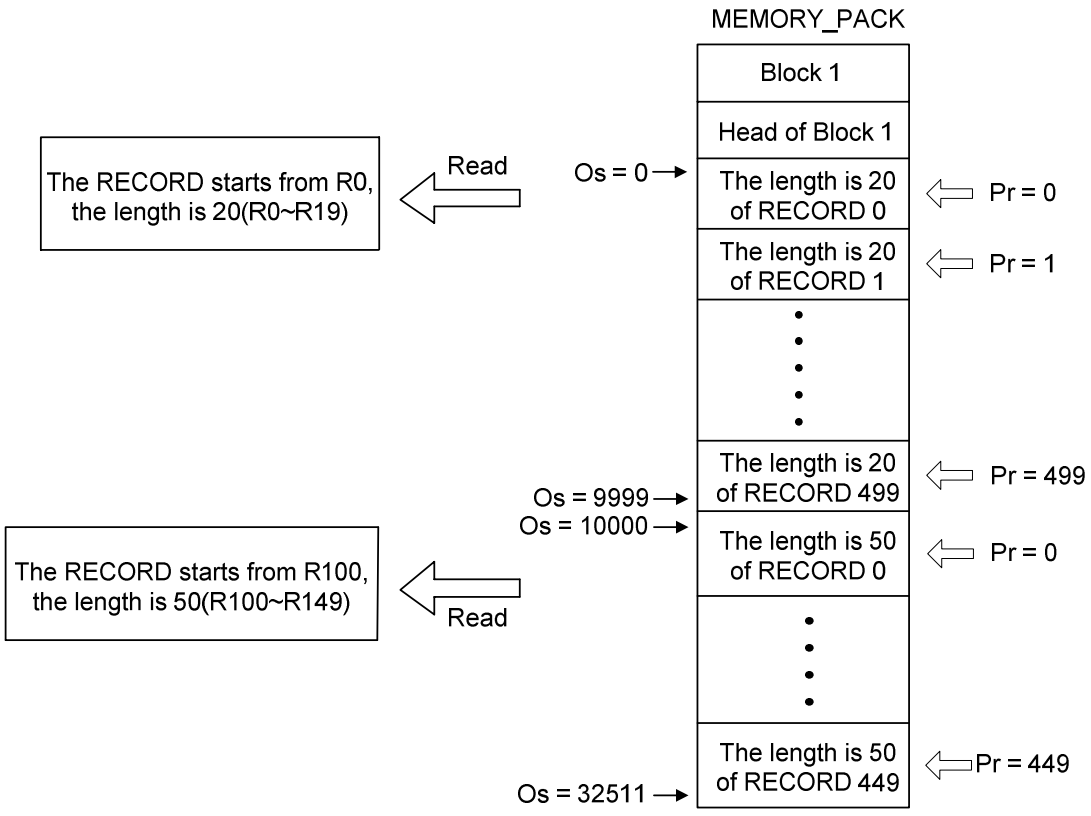
FUN162 P RD-MP	Read Data Record from the MEMORY_PACK （Read memory pack）	FUN162 P RD-MP

- If the value of L is equal to 0 or greater than 128, or the pointed data area over the range, the output "ERR" will be 1, it will not perform the reading operation.
- Output will be “ERR” if MEMORY_PACK is empty or data format not correct and user used FUN162 to read data from MEMORY_PACK.

Example program: Reading the record from block 1 of MEMORY_PACK with the different length

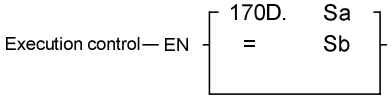
※ It is necessary that correct data in MEMORY_PACK or this example can’t execute correctly.





FUN170 D =	EQUAL TO COMPARE （Compare whether Sa is equal to Sb）	FUN170 D =

Ladder symbol

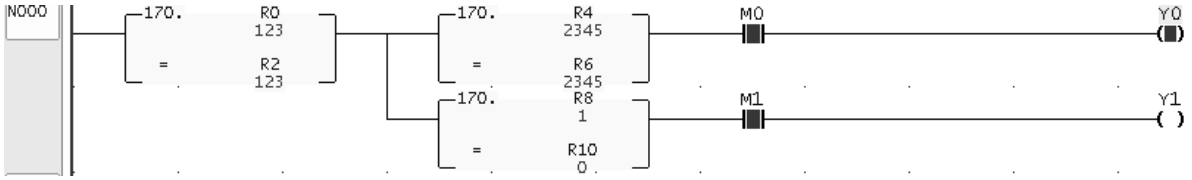


Sa: Operand A or the starting address of Sa
Sb: Operand B or the starting address of Sb
Sa, Sb may combine with V,Z,P0～P9 for indirect addressing application

Range / Ope-rand	WX	WY	WM	WS	TMR	CTR	HR	IR	OR	SR	ROR	DR	K	XR
	WX0 | WX240	WY0 | WY240	WM0 | WM1896	WS0 | WS984	T0 | T255	C0 | C199	R0 | R3839	R3840 | R3093	R3904 | R3967	R3968 | R4167	R5000 | R8071	D0 | D4095	16 bit +/- number	V、Z P0~P9
Sa	○	○	○	○	○	○	○	○	○	○	○	○	○	○
Sb	○	○	○	○	○	○	○	○	○	○	○	○	○	○

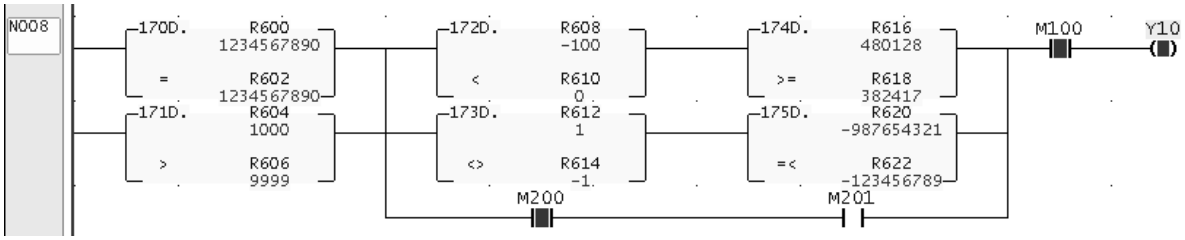
- When execution control "EN"=1, this instruction will be executed in signed number to compare Sa with Sb. If Sa=Sb, the output is 1; otherwise the output is 0.

Example 1:



Description: When R0=R2, R4=R6 and M0=1, the output status of Y0 is 1; otherwise it is 0
R0=R2, R8=R10 and M1=1, the output status of Y1 is 1; otherwise it is 0

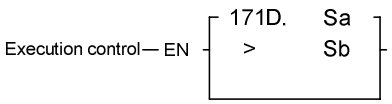
Example 2:



Description: When DR600=DR602 or DR604>DR606, after them DR608<DR610 and DR616≧DR618, or DR612≠DR614 and DR620≦DR622, or M200=1and M201=1, and then M100=1, the output status of Y10 is 1; otherwise it is 0.

FUN171 D >	GREATER THAN COMPARE （Compare whether Sa is greater than Sb）	FUN171 D >

Ladder symbol

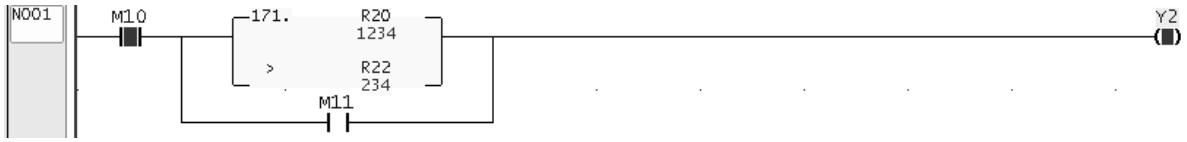


Sa: Operand A or the starting address of Sa
Sb: Operand B or the starting address of Sb
Sa, Sb may combine with V,Z,P0～P9 for indirect addressing application

Range / Operand	WX	WY	WM	WS	TMR	CTR	HR	IR	OR	SR	ROR	DR	K	XR
	WX0 | WX240	WY0 | WY240	WM0 | WM1896	WS0 | WS984	T0 | T255	C0 | C199	R0 | R3839	R3840 | R3093	R3904 | R3967	R3968 | R4167	R5000 | R8071	D0 | D4095	16 bit +/- number	V、Z P0~P9
Sa	○	○	○	○	○	○	○	○	○	○	○	○	○	○
Sb	○	○	○	○	○	○	○	○	○	○	○	○	○	○

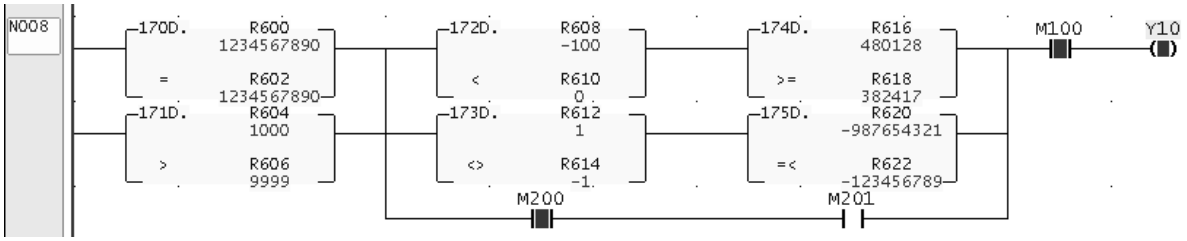
- When execution control "EN"=1, this instruction will be executed in signed number to compare Sa with Sb. If Sa>Sb, the output is 1; otherwise the output is 0.

Example 1:



Description: When M10=1, R20 > R22 or M11=1, the output status of Y2 is 1; otherwise it is 0.

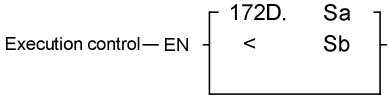
Example 2:



Description: When DR600=DR602 or DR604>DR606, after them DR608<DR610 and DR616≧DR618, or DR612≠DR614 and DR620≦DR622, or M200=1and M201=1, and then M100=1, the output status of Y10 is 1; otherwise it is 0.

FUN172 D <	LESS THAN COMPARE （Compare whether Sa is less than Sb）	FUN172 D <

Ladder symbol



Sa：Operand A or the starting address of Sa
Sb：Operand B or the starting address of Sb
Sa, Sb may combine with V,Z,P0～P9 for indirect addressing application

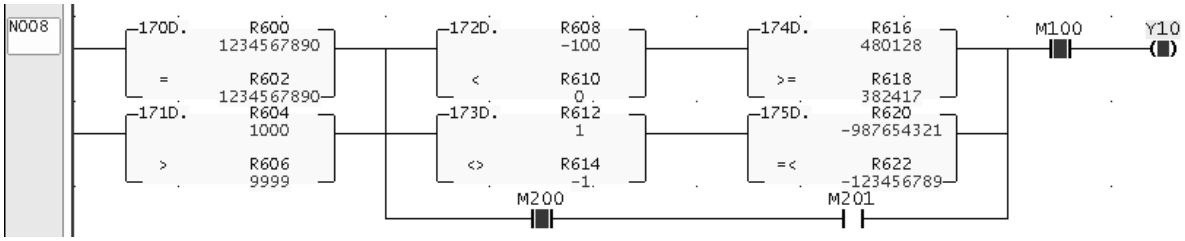
Range / Operand	WX	WY	WM	WS	TMR	CTR	HR	IR	OR	SR	ROR	DR	K	XR
	WX0 | WX240	WY0 | WY240	WM0 | WM1896	WS0 | WS984	T0 | T255	C0 | C199	R0 | R3839	R3840 | R3093	R3904 | R3967	R3968 | R4167	R5000 | R8071	D0 | D4095	16 bit +/- number	V、Z P0~P9
Sa	○	○	○	○	○	○	○	○	○	○	○	○	○	○
Sb	○	○	○	○	○	○	○	○	○	○	○	○	○	○

- When execution control "EN"=1, this instruction will be executed in signed number to compare Sa with Sb. If Sa<Sb, the output is 1; otherwise the output is 0.

Example 1:

Description: When M10=1, R20 < R22 or M11=1, the output status of Y2 is 1; otherwise it is 0.

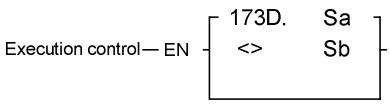
Example 2:



Description: When DR600=DR602 or DR604>DR606, after them DR608<DR610 and DR616≧DR618, or DR612≠DR614 and DR620≦DR622, or M200=1and M201=1, and then M100=1, the output status of Y10 is 1; otherwise it is 0.

FUN173 D <>	NOT EQUAL TO COMPARE (Compare whether Sa is not equal to Sb)	FUN173 D <>

Ladder symbol

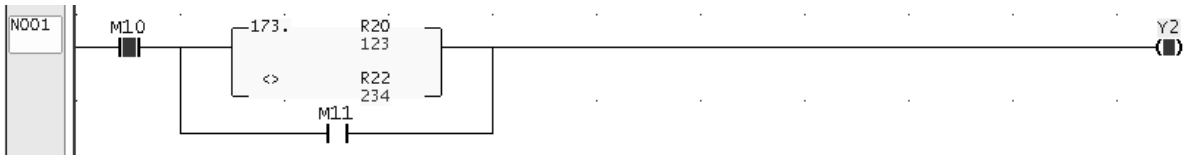


Sa: Operand A or the starting address of Sa
Sb: Operand B or the starting address of Sb
Sa, Sb may combine with V,Z,P0～P9 for indirect addressing application

Operand \ Range	WX	WY	WM	WS	TMR	CTR	HR	IR	OR	SR	ROR	DR	K	XR
	WX0 | WX240	WY0 | WY240	WM0 | WM1896	WS0 | WS984	T0 | T255	C0 | C199	R0 | R3839	R3840 | R3093	R3904 | R3967	R3968 | R4167	R5000 | R8071	D0 | D4095	16 bit +/- number	V、Z P0~P9
Sa	○	○	○	○	○	○	○	○	○	○	○	○	○	○
Sb	○	○	○	○	○	○	○	○	○	○	○	○	○	○

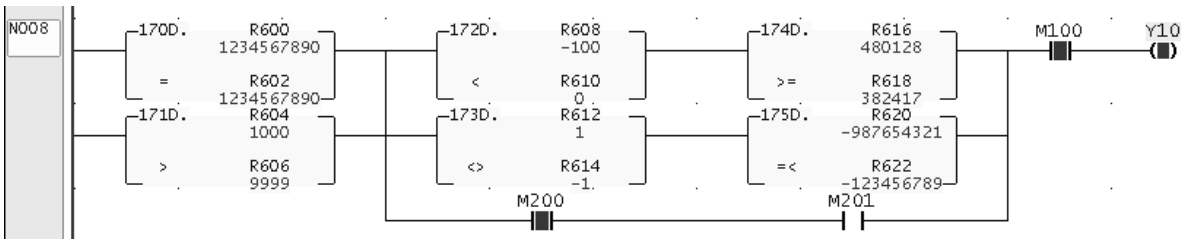
- When execution control "EN"=1, this instruction will be executed in signed number to compare Sa with Sb. If Sa≠Sb, the output is 1; otherwise the output is 0.

Example 1:



Description: When M10=1, R20≠R22 or M11=1, the output status of Y2 is 1; otherwise it is 0.

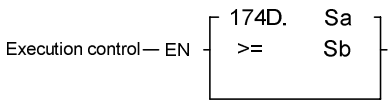
Example 2:



Description: When DR600=DR602 or DR604>DR606, after them DR608<DR610 and DR616≧DR618, or DR612≠DR614 and DR620≦DR622, or M200=1and M201=1, and then M100=1, the output status of Y10 is 1; otherwise it is 0.

FUN174 D >=	GREATER THAN OR EQUAL TO COMPARE （Compare whether Sa is greater than or equal to Sb）	FUN174 D >=

Ladder symbol

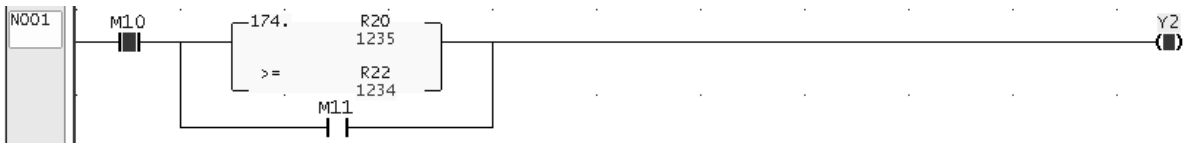


Sa: Operand A or the starting address of Sa
Sb: Operand B or the starting address of Sb
Sa, Sb may combine with V,Z,P0～P9 for indirect addressing application

Operand \ Range	WX	WY	WM	WS	TMR	CTR	HR	IR	OR	SR	ROR	DR	K	XR
	WX0 | WX240	WY0 | WY240	WM0 | WM1896	WS0 | WS984	T0 | T255	C0 | C199	R0 | R3839	R3840 | R3093	R3904 | R3967	R3968 | R4167	R5000 | R8071	D0 | D4095	16 bit +/- number	V、Z P0~P9
Sa	○	○	○	○	○	○	○	○	○	○	○	○	○	○
Sb	○	○	○	○	○	○	○	○	○	○	○	○	○	○

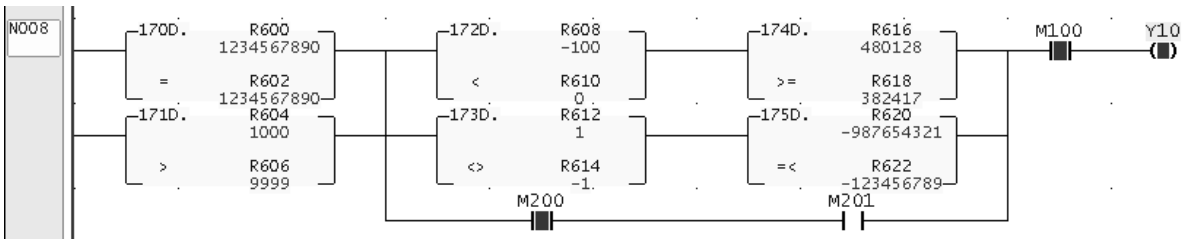
- When execution control "EN"=1, this instruction will be executed in signed number to compare Sa with Sb. If Sa≧Sb, the output is 1; otherwise the output is 0.

Example 1:



Description: When M10=1, R20≧R22 or M11=1, the output status of Y2 is 1; otherwise it is 0.

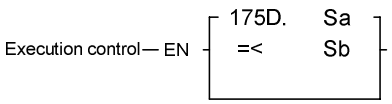
Example 2:



Description: When DR600=DR602 or DR604>DR606, after them DR608<DR610 and DR616≧DR618, or DR612≠DR614 and DR620≦DR622, or M200=1and M201=1, and then M100=1, the output status of Y10 is 1; otherwise it is 0.

FUN175 D =<	LESS THAN OR EQUAL TO COMPARE (Compare whether Sa is less than or equal to Sb)	FUN175 D =<

Ladder symbol

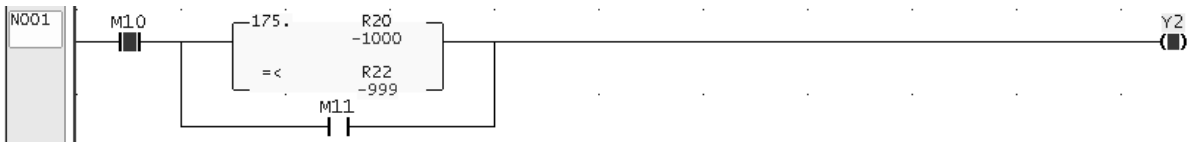


Sa: Operand A or the starting address of Sa
Sb: Operand B or the starting address of Sb
Sa, Sb may combine with V,Z,P0～P9 for indirect addressing application

Operand \ Range	WX	WY	WM	WS	TMR	CTR	HR	IR	OR	SR	ROR	DR	K	XR
	WX0 | WX240	WY0 | WY240	WM0 | WM1896	WS0 | WS984	T0 | T255	C0 | C199	R0 | R3839	R3840 | R3093	R3904 | R3967	R3968 | R4167	R5000 | R8071	D0 | D4095	16 bit +/- number	V、Z P0~P9
Sa	○	○	○	○	○	○	○	○	○	○	○	○	○	○
Sb	○	○	○	○	○	○	○	○	○	○	○	○	○	○

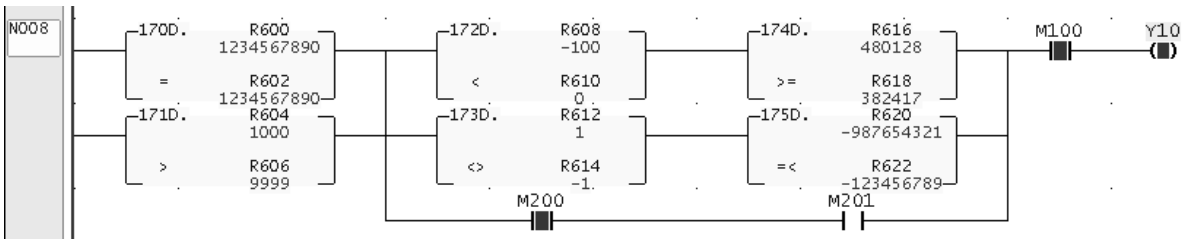
- When execution control "EN"=1, this instruction will be executed in signed number to compare Sa with Sb. If Sa≦Sb, the output is 1; otherwise the output is 0.

Example 1:



Description: When M10=1, R20≦R22 or M11=1, the output status of Y2 is 1; otherwise it is 0.

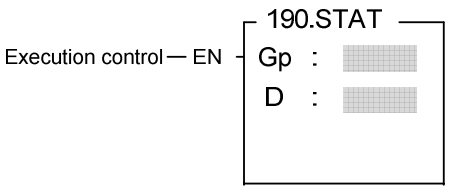
Example 2:



Description: When DR600=DR602 or DR604>DR606, after them DR608<DR610 and DR616≧DR618, or DR612≠DR614 and DR620≦DR622, or M200=1and M201=1, and then M100=1, the output status of Y10 is 1; otherwise it is 0.

FUN190 STAT	READ SYSTEM STATUS	FUN190 STAT

Ladder symbol



Gp : Specified status group
0 : Get information of I/O expansion
1~3 : Reserved

D : Starting address of register to store the system status
D+0 : Quantity of status
D+1 : Status 1
…
D+N: Status N

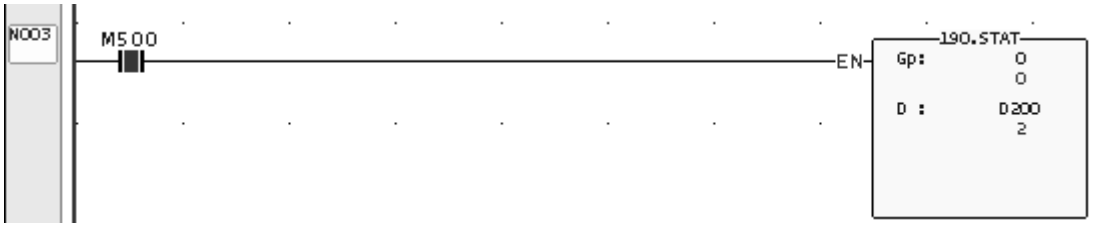
Range / Operand	HR	ROR	DR	K
	R0 | R3839	R5000 | R8071	D0 | D3999	
Gp				0~3
D	○	○*	○	

- When execution control "EN"=1, this instruction being executed, and if Gp=0, it means to get the information of I/O expansion modules; total quantity of I/O expansion modules will be stored in D register, code of I/O expansion module will be stored in D+1~D+N registers in order. Gp=1~3, reserved for future.

Code of I/O Expansion Module	Name of I/O Expansion Module	Code of I/O Expansion Module	Name of I/O Expansion Module
1	FBs-8XY◇ (Non-support)	18	FBs-4DA (Non-support)
2	FBs-8X (Non-support)	19	FBs-4PT (Non-support)
3	FBs-8Y◇ (Non-support)	20	WSZ-4A2D
4	FBs-16XY◇ (Non-support)	21	FBs-6TC (Non-support)
5	FBs-20X (Non-support)	22	FBs-6RTD (Non-support)
6	WSZ-16YR FBs-16Y◇ (Non-support)	23	WSZ-16TC
7	FBs-24X (Non-support)	24	FBs-16RTD (Non-support)
8	FBs-24Y (Non-support)	25	FBs-2TC (Non-support)
9	WSZ-24XYT FBs-24XY◇ (Non-support)	26	FBs-2A4TC (Non-support)
10	FBs-40XY◇ (Non-support)	27	FBs-2A4RTD (Non-support)
11	FBs-60XY◇ (Non-support)	28	FBs-6NTC (Non-support)
12	FBs-7SG1S (Decode,Non-support)	29	FBs-16NTC (Reserved)
13	FBs-7SG1H (Non-decode, Non-support)	30	FBs-32DGI (Non-support)
14	FBs-7SG2S (Decode,Non-support)	31	FBs-VOM (Non-support)
15	FBs-7SG2H (Non-decode, Non-support)	32	FBs-1LC (Non-support)
16	WSZ-6AD		
17	WSZ-2DA		

FUN190 STAT	READ SYSTEM STATUS	FUN190 STAT

Example : There are two I/O expansion modules WSZ-2DA + WSZ-6AD installed in one system



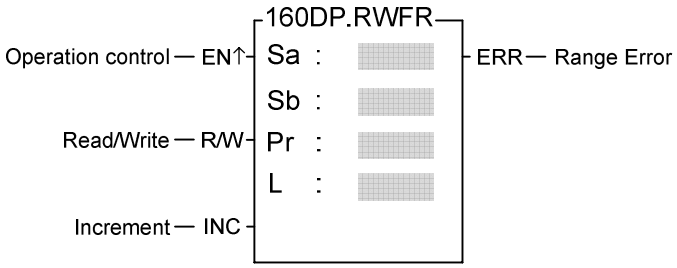
Status Monitoring

Ref. No.	Status	Data	Ref. No.	Status	Data
M500	Enable	ON			
D200	Decimal	2			
D201	Decimal	17			
D202	Decimal	16			

Description: When M500=1, this instruction being executed, register D200 is used to store the total quantity of I/O expansion modules, register D201 is used to store the code (17=WSZ-2DA) of first I/O expansion module, register D202 is used to store the code (16=WSZ-6AD) of second I/O expansion module.

FUN160 D P RWFR	READ/WRITE FILE REGISTER	FUN160 D P RWFR

Ladder symbol



Sa: Starting address of data register

Sb: Starting address of file register

Pr : Record pointer register

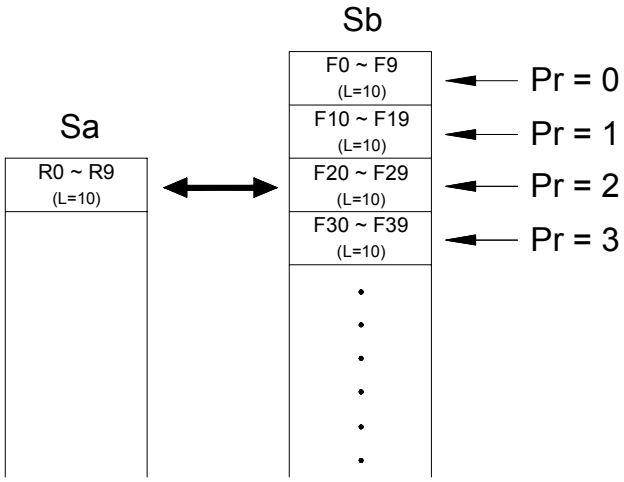
L : Quantity of register to form a record, 1~511

Sa operand can combine V, Z, P0~P9 for index addressing.

Operand \ Range	WX	WY	WM	WS	TMR	CTR	HR	IR	OR	SR	ROR	DR	K	XR	FR
	WX0 | WX240	WY0 | WY240	WM0 | WM1896	WS0 | WS984	T0 | T255	C0 | C255	R0 | R3839	R3840 | R3903	R3904 | R3967	R3968 | R4167	R5000 | R8071	D0 | D4095		V, Z P0～P9	F0 | F8191
Sa	○	○	○	○	○	○	○	○	○	○	○	○		○	
Sb															○
Pr		○	○	○	○	○	○		○	○*	○*	○			
L							○				○*	○	1~511		

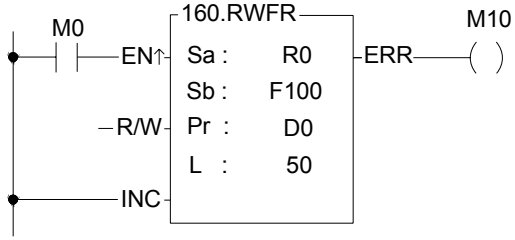
Description

- When operation control "EN"=1 or "EN↑" (P instruction) changes from 0 to 1,it will perform the read ("R/W"=1) or write ("R/W"=0) file register operation. While reading, the content of data registers starting from Sa will be overwritten by the content of file registers addressed by the base file register Sb and record pointer Pr; while writing, the content of file registers addressed by the base file register Sb and record pointer Pr will be overwritten by the content of data registers starting from Sa; L is the operation quantity or record size. The access of file register adopts the concept of RECORD data structure to implement. For example, Sa=R0, Sb=F0, L=10, the read/write details shown as below.

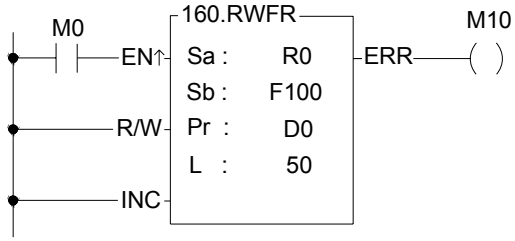


FUN160 D P RWFR	READ/WRITE FILE REGISTER	FUN160 D P RWFR

- For ladder program application, only this instruction can access the file registers.
- The record pointer will be increased by 1 after execution while pointer control input "INC"=1.
- This instruction will not be executed and error indicator ”ERR" will be 1 while incorrect record size (L=0 or > 511) or the operation out of the file register's range (F0～F8191).



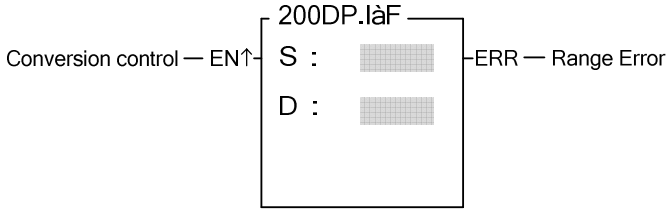
When M0 changes from 0 to 1, if D0 =2, the contents of file registers F200~F249 will be overwritten by the content of data registers R0~R49. the record length is 50.
Pointer will be increased by 1 after operation.



When M0 changes from 0 to 1, if D0 =1, the content of data registers R0~R49 will be overwritten by the file registers F150~F199.
The record pointer will be increased by 1 after operation.

FUN200 D P I→F	CONVERSION OF INTEGER TO FLOATING POINT NUMBER	FUN200 D P I→F

Ladder symbol



S : Starting register of Integer to be converted

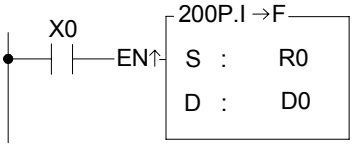
D : Starting register to store the result of conversion

S, D may combine with V, Z, P0~P9 to serve indirect addressing.

Range / Operand	HR	ROR	DR	K	XR
	R0 | R3839	R5000 | R8071	D0 | D4095	16 bit +/- number	V, Z P0～P9
S	○	○	○	○	○
D	○	○*	○		○

Description

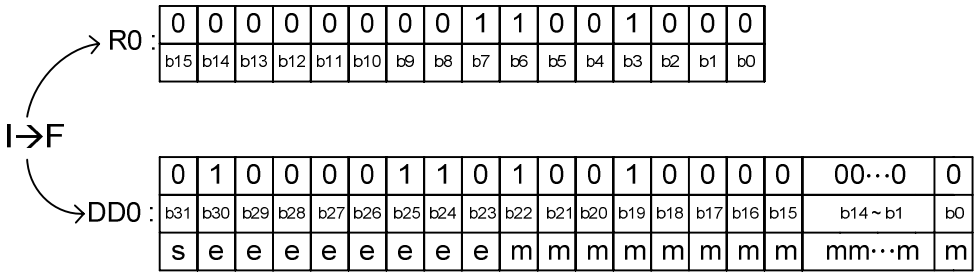
- The format of floating point number of WSZ-controller follows the IEEE-754 standard.
- When conversion control "EN" =1 or "EN↑" (P instruction) changes from 0 to 1, will convert the integer data from S register into D~D+1 32-bit register(floating point number data).
- If the value exceeds the valid range of destination, this instruction is not carried out, and the range-error flag “ERR” is set as 1 and the D register will be intact.



※ R0 = 200 (0000000011001000)

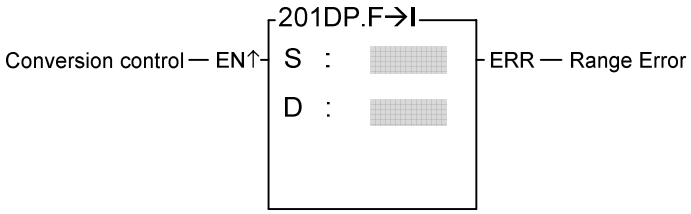
Integer To Floating

DD0 = 43480000H



FUN201 D P F→I	CONVERSION OF FLOATING POINT NUMBER TO INTEGER	FUN201 D P F→I

Ladder symbol



S : Starting register of floating point number to be converted

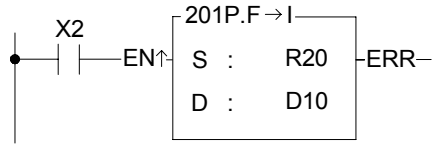
D : Starting register to store the result of conversion

S, D may combine with V, Z, P0~P9 to serve indirect addressing.

Operand \ Range	HR	ROR	DR	XR
	R0 | R3839	R5000 | R8071	D0 | D4095	V, Z P0～P9
S	○	○	○	○
D	○	○*	○	○

Description

- The format of floating point number of WSZ-controller follows the IEEE-754 standard.
- When conversion control "EN" =1 or "EN↑" (P instruction) changes from 0 to 1, will convert the floating point data from S~S+1 32 bits register into D register(integer data).
- If the value exceeds the valid range of destination, then do not carry out this instruction, and set the range-error flag “ERR” as 1 and the D register will be intact.



※ DR20 = 123.45 →Normalize→ 42F6E666H

Floating To Integer

→ D10 = 007BH

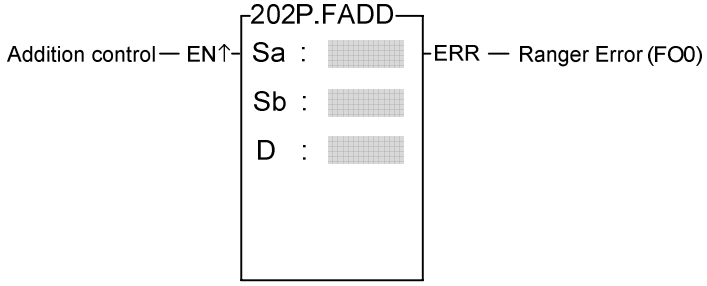
DR20:	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	0	1	0	0	0	0	1	0	1	1	1	1	0	1	1	0	1	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0
	s	e	e	e	e	e	e	e	e	m	m	m	m	m	m	m	m	m	m	m	m	m	m	m	m	m	m	m	m	m	m	m

F → I

D10:	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	0	0	0	0	0	0	0	0	0	1	1	1	1	0	1	1

FUN202 P FADD	FLOATING POINT NUMBER ADDITION	FUN202 P FADD

Ladder symbol



Sa: Augend

Sb: Addend

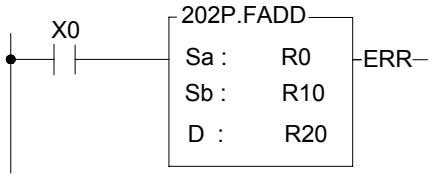
D : Destination register to store the results of the addition

Sa, Sb, D may combine with V, Z, P0~P9 to serve indirect addressing.

Range / Operand	HR	ROR	DR	K	XR
	R0 | R3839	R5000 | R8071	D0 | D4095	Floating point number	V, Z P0～P9
Sa	○	○	○	○	○
Sb	○	○	○	○	○
D	○	○*	○		○

Description

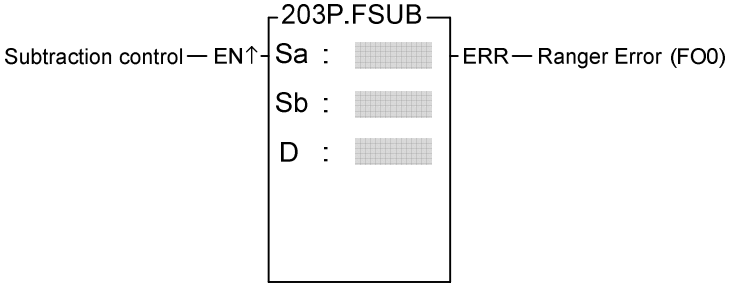
- The format of floating point number of WSZ-controller follows the IEEE-754 standard.
- Performs the addition of the data specified at Sa and Sb and writes the results to a specified register D when the addition control "EN" =1 or "EN↑" (P instruction) changes from 0 to 1. If the result exceed the range that the floating point number can be expressed($\pm 3.4 * 10^{38}$) then the error flag FO0 will be set to 1 and the D register will be intact.



DR0	200	⇨ Floating Point Number :	DR0	43480000H
DR10	150	⇨ Floating Point Number :	DR10	43160000H
		+		
			DR20	43AF0000H

FUN 203 P FSUB	FLOATING POINT NUMBER SUBTRACTION	FUN 203 P FSUB

Ladder symbol



Sa: Minuend

Sb: Subtrahend

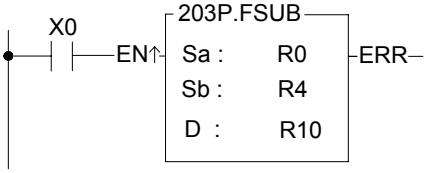
D : Destination register to store the results of the subtraction

Sa, Sb, D may combine with V, Z, P0~P9 to serve indirect addressing.

Range / Operand	HR R0 | R3839	ROR R5000 | R8071	DR D0 | D4095	K Floating point number	XR V, Z P0～P9
Sa	○	○	○	○	○
Sb	○	○	○	○	○
D	○	○*	○		○

Description

- The format of floating point number of WSZ-controller follows the IEEE-754 standard.
- Performs the subtraction of the data specified at Sa and Sb and writes the results to a specified register D when the subtract control "EN" =1 or "EN↑" (P instruction) changes from 0 to 1. If the result exceed the range that the floating point number can be expressed($\pm 3.4*10^{38}$) then the error flag FO0 will be set to 1 and the D register will be intact.



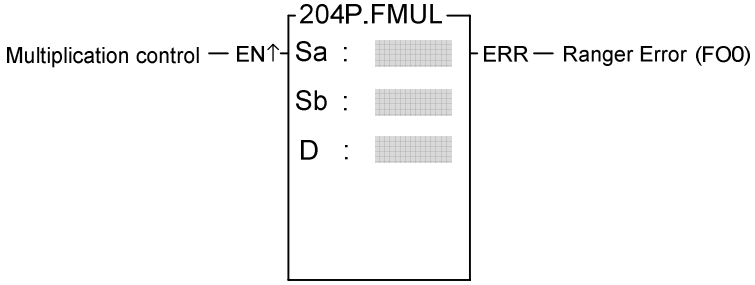
DR0	2 0 0	⇨ Floating Point Number :	DR0	4 3 4 8 0 0 0 0 H
DR4	5 0 0	⇨ Floating Point Number :	DR4	4 3 F A 0 0 0 0 H

—

DR10	C 3 9 6 0 0 0 0 H

FUN 204 P FMUL	FLOATING POINT NUMBER MULTIPLICATION	FUN 204 P FMUL

Ladder symbol



Sa: Multiplicand

Sb: Multiplier

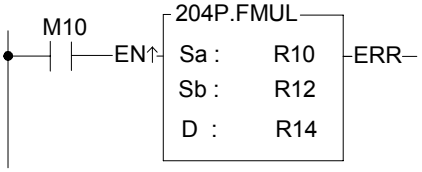
D : Destination register to store the results of the multiplication

Sa, Sb, D may combine with V, Z, P0~P9 to serve indirect addressing.

Range / Operand	HR	ROR	DR	K	XR
	R0 | R3839	R5000 | R8071	D0 | D4095	Floating point number	V, Z P0～P9
Sa	○	○	○	○	○
Sb	○	○	○	○	○
D	○	○*	○		○

Description

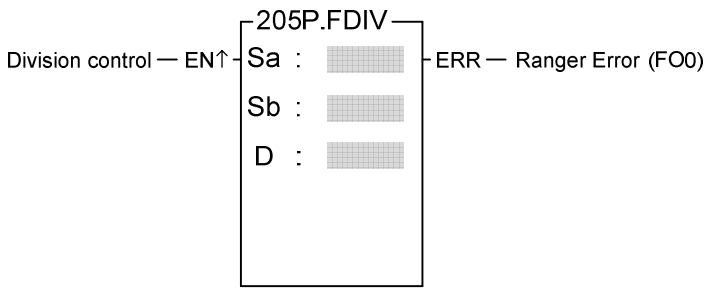
- The format of floating point number of WSZ-controller follows the IEEE-754 standard.
- Performs the multiplication of the data specified at Sa and Sb and writes the results to a specified register D when the multiplication control "EN" =1 or "EN↑" (P instruction) changes from 0 to 1. If the result exceed the range that the floating point number can be expressed($\pm 3.4*10^{38}$) then the error flag FO0 will be set to 1 and the D register will be intact.



DR10	1 2 3 . 4 5	⇨ Floating Point Number :	DR10	4 2 F 6 E 6 6 6 H
DR12	6 7 8 . 5 4	⇨ Floating Point Number :	DR12	4 4 2 9 A 2 8 F H
			×	
			DR14	4 7 A 3 9 A E 2 H

FUN 205 P FDIV	FLOATING POINT NUMBER DIVISION	FUN 205 P FDIV

Ladder symbol



Sa: Dividend

Sb: Divisor

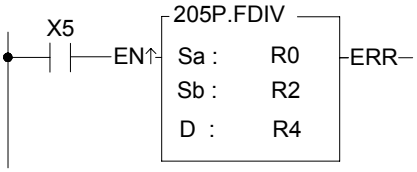
D : Destination register to store the results of the division

Sa, Sb, D may combine with V, Z, P0~P9 to serve indirect addressing.

Range / Operand	HR	ROR	DR	K	XR
	R0 | R3839	R5000 | R8071	D0 | D4095	Floating point number	V, Z P0～P9
Sa	○	○	○	○	○
Sb	○	○	○	○	○
D	○	○*	○		○

Description

- The format of floating point number of WSZ-controller follows the IEEE-754 standard.
- Performs the division of the data specified at Sa and Sb and writes the result to the registers specified by register D when the division control "EN" =1 or "EN↑" (P instruction) changes from 0 to 1. If the result exceed the range that the floating point number can be expressed($\pm 3.4 * 10^{38}$) then the error flag FO0 will be set to 1 and the D register will be intact.



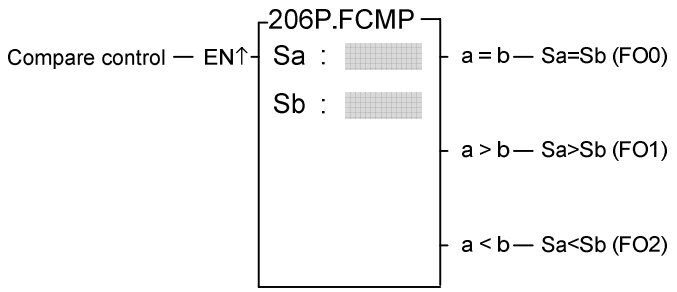
DR0	125.25	⇨ Floating Point Number :	DR0	42FA8000H
DR2	5	⇨ Floating Point Number :	DR2	40A00000H

÷

DR4	41C86666H

FUN 206 P FCMP	FLOATING POINT NUMBER COMPARE	FUN 206 P FCMP

Ladder symbol



Sa: The register to be compared

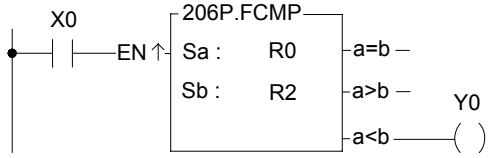
Sb: The register to be compared

Sa, Sb may combine with V, Z, P0~P9 to serve indirect addressing.

Range / Operand	HR	ROR	DR	K	XR
	R0 | R3839	R5000 | R8071	D0 | D4095	Floating point number	V, Z P0～P9
Sa	○	○	○	○	○
Sb	○	○	○	○	○

Description

- The format of floating point number of WSZ-controller follows the IEEE-754 standard.
- Compares the data of Sa and Sb when the compare control "EN" =1 or "EN↑" (P instruction) changes from 0 to 1. If the data of Sa is equal to Sb, then set FO0 to 1. If the data of Sa>Sb, then set FO1 to 1. If the data of Sa<Sb, then set FO2 to 1. If the data of Sa < Sb, then set the FO2 to 1.



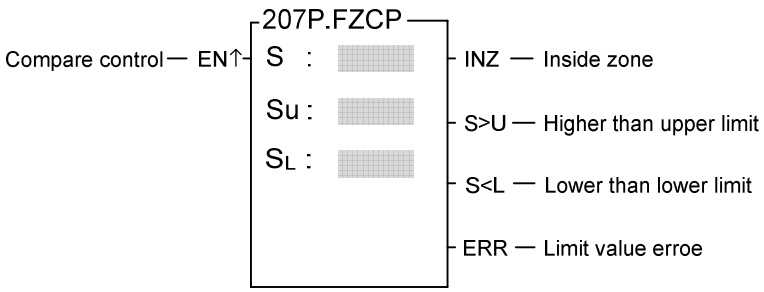
DR0	200.1	⇨ Floating Point Number :	DR0	43481999AH

DR2	200.2	⇨ Floating Point Number :	DR2	43483333H

- From the above example, we first assume the data of DR0 is 200.1 and DR2 is 200.2, and then compare the data by executing the CMP instruction. The FO0 and FO1 are set to 0 and FO2 (a<b) is set to 1 since a<b.
- If you want to have the compound results, such as≧, ≦, < > etc., please send =, < and > results to relay first and then combine the result from the relays.

FUN 207 P FZCP	FLOATING POINT NUMBER ZONE COMPARE	FUN 207 P FZCP

Ladder Symbol



S : Register for zone comparison

SU: The upper limit value

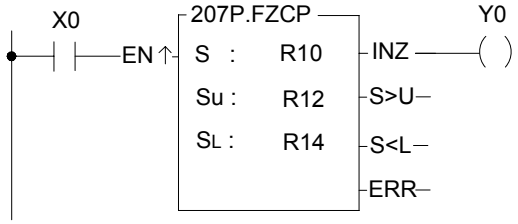
SL: The lower limit value

S, SU, SL may combine with V, Z, P0~P9 to serve indirect address application.

Range / Operand	HR	ROR	DR	K	XR
	R0 | R3839	R5000 | R8071	D0 | D4095	Floating point number	V, Z P0～P9
S	○	○	○		○
Su	○	○	○	○	○
SL	○	○	○	○	○

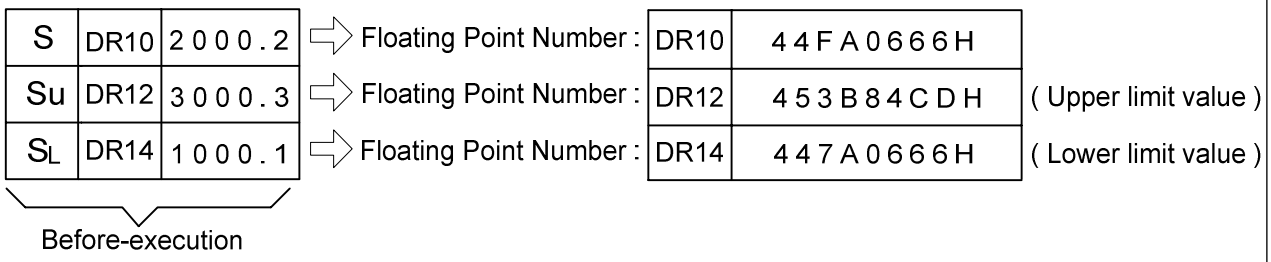
Description

- The format of floating point number of WSZ-controller follows the IEEE-754 standard.
- When compare control "EN" =1 or "EN↑" (P instruction) changes from 0 to 1, compares S with upper limit SU and lower limit SL. If S is between the upper limit and the lower limit ($S_L \leqq S \leqq S_U$), then set the inside zone flag "INZ" to 1. If the value of S is greater than the upper limit S_U, then set the higher than upper limit flag "S>U" to 1. If the value of S is smaller then the lower limit S_L, then set the lower than lower limit flag "S<L" as 1.
- The upper limit S_U should be greater than the lower limit S_L. If $S_U < S_L$, then the limit value error flag "ERR" will set to 1, and this instruction will not carry out.



- The instruction at left compares the value of DR10 with the upper and lower limit zones formed by DR12 and DR14. If the values of DR10~DR14 are as shown in the diagram at bottom left, then the result can then be obtained as at the right of this diagram.
- If want to get the status of out side the zone, then OUT NOT Y0 may be used, or an OR operation between the two outputs S>U and S<L may be carried out, and move the result to Y0.

FUN 207 P FZCP	FLOATING POINT NUMBER ZONE COMPARE	FUN 207 P FZCP

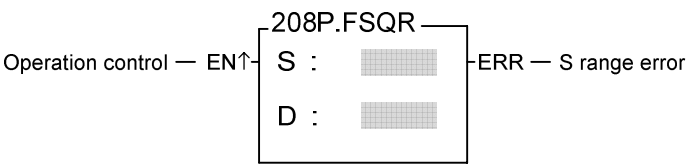


X0＝1 ↑ → FLOATING ZONE COMPARE → Y0 = 1

Results of execution

FUN 208 P FSQR	FLOATING POINT NUMBER SQUARE ROOT	FUN 208 P FSQR

Ladder symbol



S : Source register to be taken square root

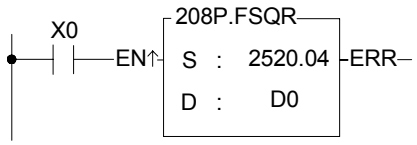
D : Register for storing result (square root value)

S, D may combine with V, Z, P0~P9 to serve indirect address application

Operand \ Range	HR	ROR	DR	K	XR
	R0 | R3839	R5000 | R8071	D0 | D4095	Floating point number	V, Z P0～P9
S	○	○	○	○	○
D	○	○*	○		○

Description

- The format of floating point number of WSZ-controller follows the IEEE-754 standard.
- When operation control "EN" =1 or "EN↑" (P instruction) changes from 0 to 1, take the square root of the data specified by the S value or S~S+1 register, and store the result into the register specified by D~D+1.
- If the value of S is negative, then the error flag "ERR" will be set to 1, and do not execute the operation.



S : K | 2520.04

⇩ X0 = 1 ↑

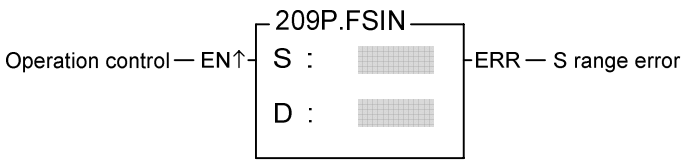
D : D1 | D0 | 50.2 ⇨ Floating Point Number : 4248 | CCCD H

D1 D0

$\sqrt{2520.04} = 50.2$

FUN 209 P FSIN	SIN TRIGONOMETRIC INSTRUCTION	FUN 209 P FSIN

Ladder symbol



S : Source register to be taken SIN

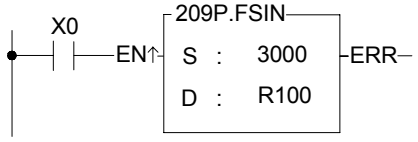
D : Register for storing result (SIN value)

S, D may combine with V, Z, P0~P9 to serve indirect address application.

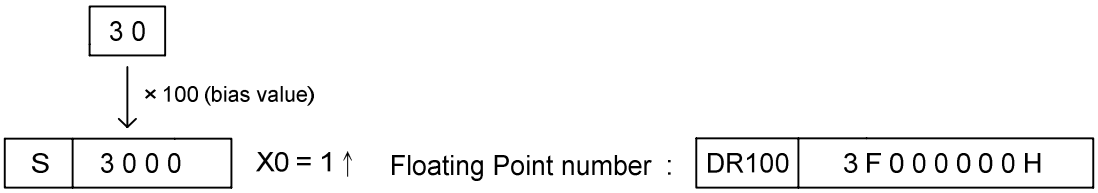
Range / Operand	HR	ROR	DR	K	XR
	R0 | R3839	R5000 | R8071	D0 | D4095	-18000 | +18000	V, Z P0～P9
S	○	○	○	○	○
D	○	○*	○		○

Description

- The format of floating point number of WSZ-controller follows the IEEE-754 standard.
- When operation control "EN" =1 or "EN↑" (P instruction) changes from 0 to 1, take the SIN value of the angle data specified by the S register and store the result into the register D~D+1 in floating point number format. The valid range of the angle is from –18000 to +18000, unit in 0.01 degree.
- If the S value is not within the valid range, then the S value error flag "ERR" will be set to 1, and do not execute the operation.



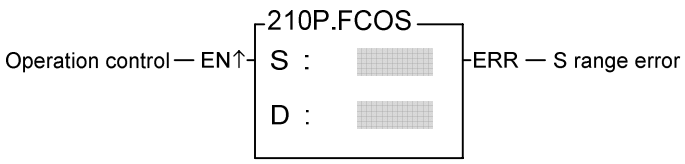
- At left, the example program gets the SIN value of 30, and stores the results the register DR100.



SIN(30) = 0.5

FUN 210 P FCOS	COS TRIGONOMETRIC INSTRUCTION	FUN 210 P FCOS

Ladder symbol



S : Source register to be taken COS

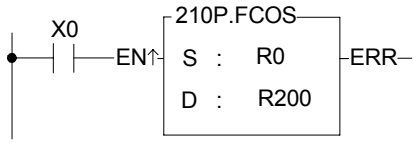
D : Register for storing result (COS value)

S, D may combine with V, Z, P0~P9 to serve indirect address application.

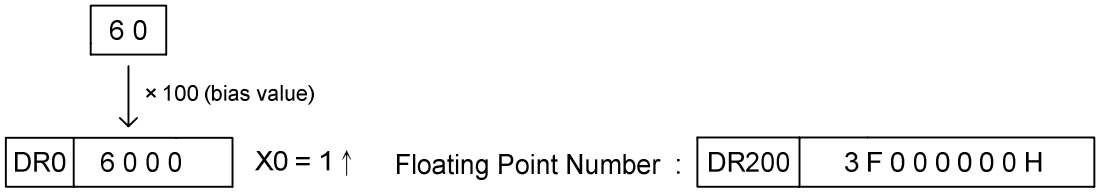
Operand \ Range	HR	ROR	DR	K	XR
	R0 | R3839	R5000 | R8071	D0 | D4095	-18000 | +18000	V, Z P0～P9
S	○	○	○	○	○
D	○	○*	○		○

Description

- The format of floating point number of WSZ-controller follows the IEEE-754 standard.
- When operation control "EN" =1 or "EN↑" (P instruction) changes from 0 to 1, take the COS value of the angle data specified by the S register and store the result into the register D~D+1 in floating point number format. The valid range of the angle is from –18000 to +18000, unit in 0.01 degree.
- If the S value is not within the valid range, then the S value error flag "ERR" will be set to 1, and do not execute the operation.



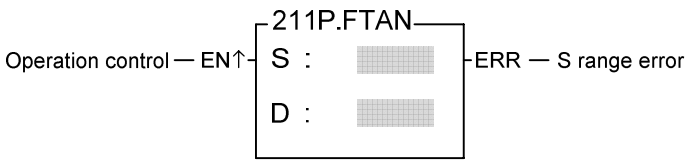
- At left, the example program gets the COS value of 60, and stores the results the register DR200.



COS(60) = 0.5

FUN 211 P FTAN	TAN TRIGONOMETRIC INSTRUCTION	FUN 211 P FTAN

Ladder symbol



S : Source register to be taken TAN

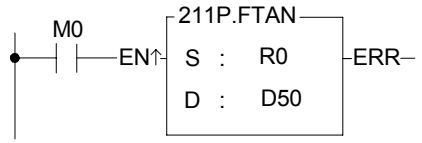
D : Register for storing result (TAN value)

S, D may combine with V, Z, P0~P9 to serve indirect address application.

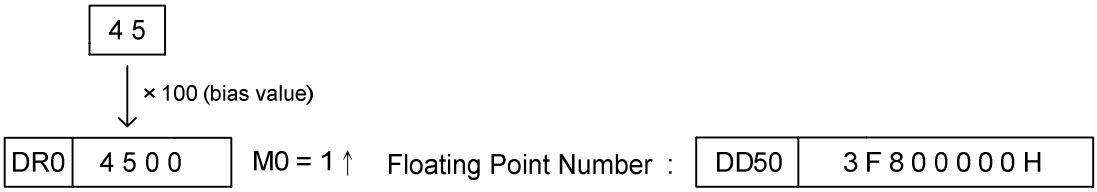
Operand \ Range	HR	ROR	DR	K	XR
	R0 | R3839	R5000 | R8071	D0 | D4095	-18000 | +18000	V, Z P0～P9
S	○	○	○	○	○
D	○	○*	○		○

Description

- The format of floating point number of WSZ-controller follows the IEEE-754 standard.
- When operation control "EN" =1 or "EN↑" (P instruction) changes from 0 to 1, take the TAN value of the angle data specified by the S register and store the result into the register D~D+1 in floating point number format. The valid range of the angle is from –18000 to +18000, unit in 0.01 degree.
- If the S value is not within the valid range, then the S value error flag "ERR" will be set to 1, and do not execute the operation.



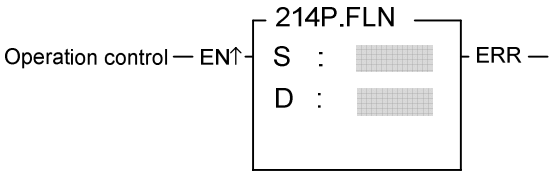
- At left, the example program gets the TAN value of 45, and stores the results the register DD50.



TAN(45) = 1

FUN 214 P FLN	FLOATING POINT NAPIERIAN LOGARITHM, $\log_e x$ or ln(x)	FUN 214 P FLN

Ladder symbol



S: Source data or register to be calculated Napierian logarithm value

D: Register for storing the result

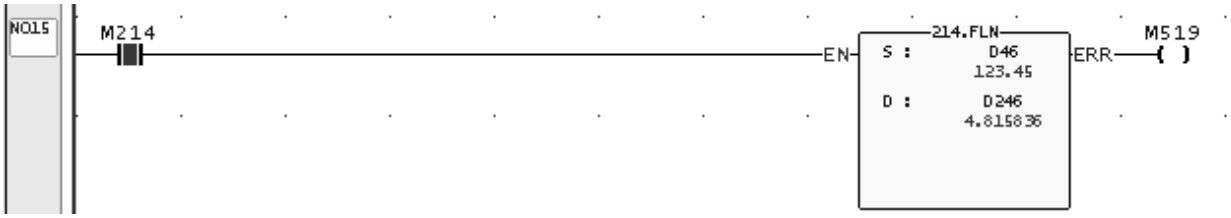
S, D may combine with V, Z, P0~P9 to serve indirect address application

Range / Ope-rand	HR	ROR	DR	K	XR
	R0 | R3838	R5000 | R8070	D0 | D4094	Floating number	V、Z P0～P9
S	○	○	○	○	○
D	○	○*	○		○

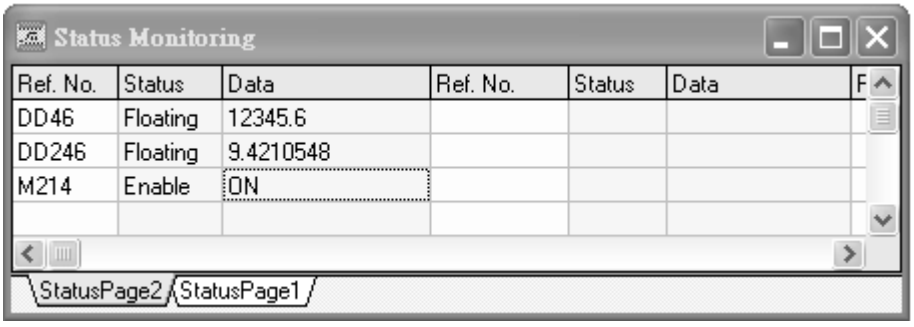
Description

- The format of floating point number of WSZ-Controller follows the IEEE-754 standard of 32-bit.
- When operation control "EN" =1 or "EN↑" (P instruction) changes from 0 to 1, take the Napierian logarithm of the data specified by the S value or S~S+1 register, and store the result into the register specified by D~D+1.
- If the value of S is negative or equal to 0, invalid indirect addressing, or over range of the result , the error flag "ERR" will be set to 1, and not update the value of D~D+1.
- All floating point instructions can't be executed in interrupt service routine.

Example

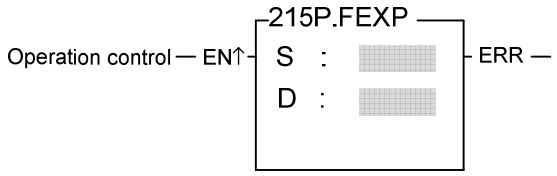


• When M214=1, calculate the Napierian logarithm value, it is DD246 = ln (DD46)



FUN 215 P FEXP	FLOATING POINT NATURE POWER FUNCTION, e^x	FUN 215 P FEXP

Ladder symbol



S: Source data or register to be calculated power function of nature number

D: Register for storing the result

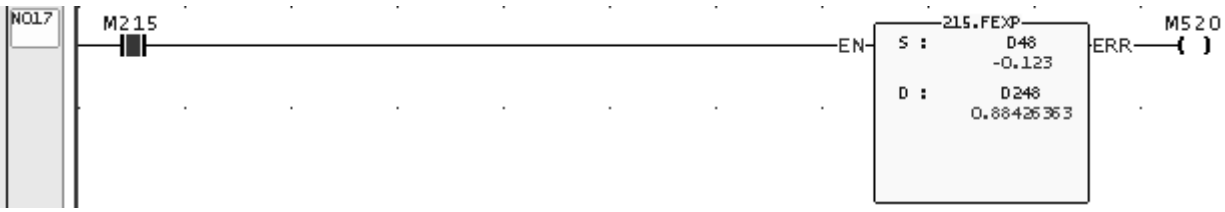
S, D may combine with V, Z, P0~P9 to serve indirect address application

Range / Operand	HR	ROR	DR	K	XR
	R0 ~ R3838	R5000 ~ R8070	D0 ~ D4094	Floating number	V、Z P0～P9
S	○	○	○	○	○
D	○	○*	○		○

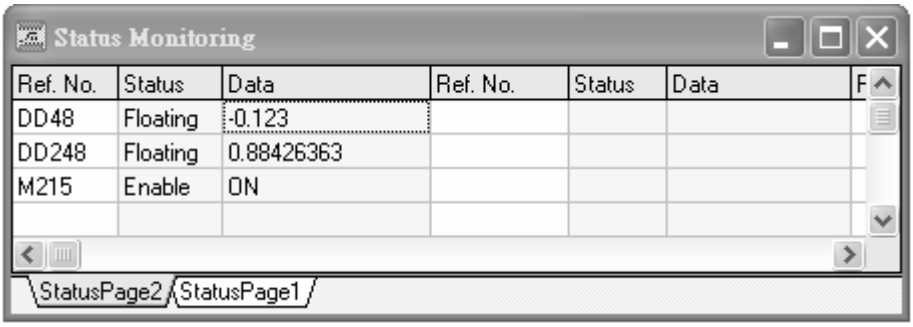
Description

- The format of floating point number of WSZ-Controller follows the IEEE-754 standard of 32-bit.
- When operation control "EN" =1 or "EN↑" (P instruction) changes from 0 to 1, calaulate the nature power function of the data specified by the S value or S~S+1 register, and store the result into the register specified by D~D+1.
- If the value of S is out of range, invalid indirect addressing, or over range of the result , the error flag "ERR" will be set to 1, and not update the value of D~D+1.
- All floating point instructions can't be executed in interrupt service routine.

Example

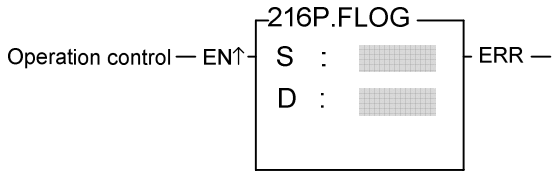


• When M215=1, calculate the nature power function, it is DD248 = e^{DD48}



FUN 216 P FLOG	FLOATING POINT LOGARITHM, $log_{10}x$ or log(x)	FUN 216 P FLOG

Ladder symbol



S: Source data or register to be calculated logarithm value

D: Register for storing the result

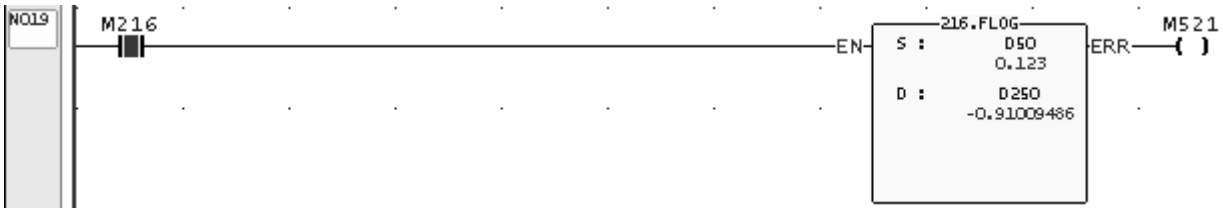
S, D may combine with V, Z, P0~P9 to serve indirect address application

Operand \ Range	HR	ROR	DR	K	XR
	R0 | R3838	R5000 | R8070	D0 | D4094	Floating number	V、Z P0～P9
S	○	○	○	○	○
D	○	○*	○		○

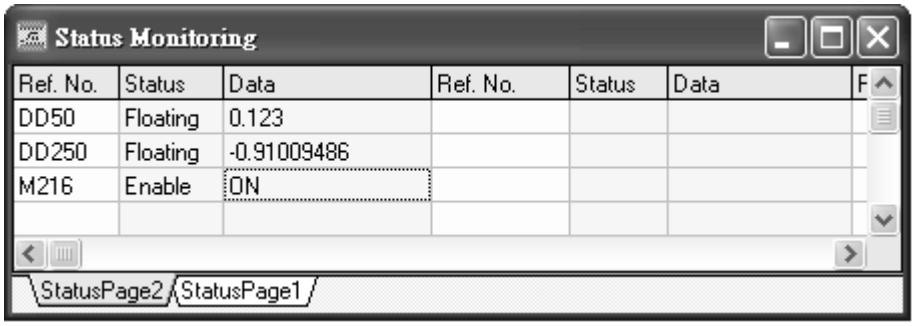
Description

- The format of floating point number of WSZ-Controller follows the IEEE-754 standard of 32-bit.
- When operation control "EN" =1 or "EN↑" (P instruction) changes from 0 to 1, calculate the logarithm value of the data specified by the S value or S~S+1 register, and store the result into the register specified by D~D+1.
- If the value of S is negative or equal to 0, invalid indirect addressing, or over range of the result , the error flag "ERR" will be set to 1, and not update the value of D~D+1.
- All floating point instructions can't be executed in interrupt service routine.

Example

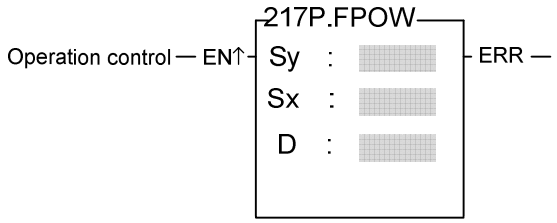


• When M216=1, calculate the logarithm value, it is DD250 = log (DD50)



FUN 217 P FPOW	FLOATING POINT POWER FUNCTION, x^y	FUN 217 P FPOW

Ladder symbol



Sy: Source data or register of exponential

Sx: Source data or register of base

D: Register for storing the result

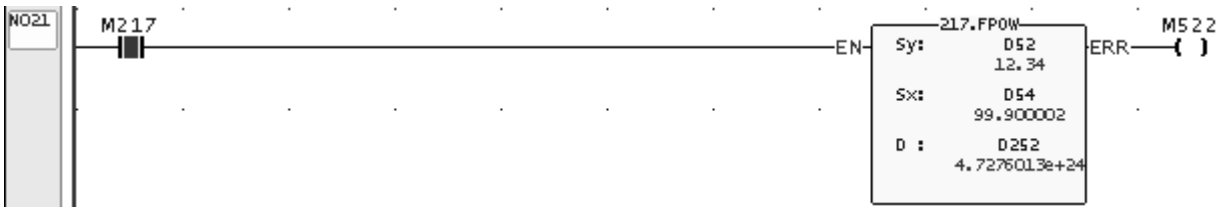
Sy, Sx, D may combine with V, Z, P0~P9 to serve indirect address application

Operand \ Range	HR	ROR	DR	K	XR
	R0 | R3838	R5000 | R8070	D0 | D4094	Floating number	V、Z P0～P9
Sy	○	○	○	○	○
Sx	○	○*	○	○	○
D	○	○	○		○

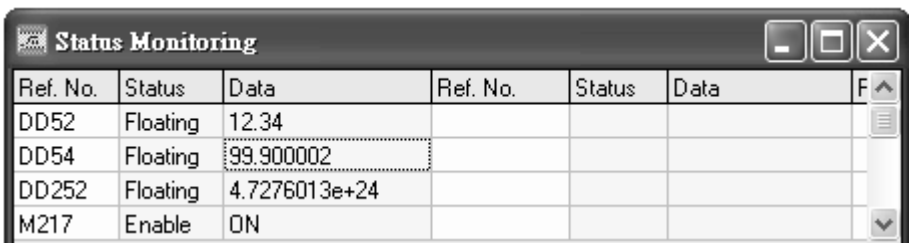
Description

- The format of floating point number of WSZ-Controller follows the IEEE-754 standard of 32-bit.
- When operation control "EN" =1 or "EN↑" (P instruction) changes from 0 to 1, calculate the power function of the exponential data specified by the Sy、base data specified by the Sx, and store the result into the register specified by D~D+1.
- If it exists invalid indirect addressing, or over range of the result , the error flag "ERR" will be set to 1, and not update the value of D~D+1.
- All floating point instructions can't be executed in interrupt service routine.

Example



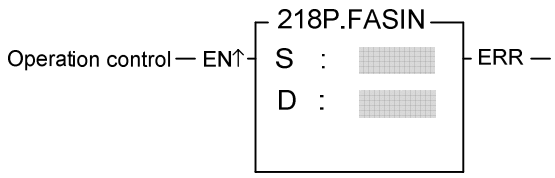
• When M217=1, calculate the power function, it is DD252 = $DD54^{DD52}$



Ref. No.	Status	Data	Ref. No.	Status	Data
DD52	Floating	12.34			
DD54	Floating	99.900002			
DD252	Floating	4.7276013e+24			
M217	Enable	ON			

FUN 218 P FASIN	FLOATING POINT ARC SINE FUNCTION, $\sin^{-1}$	FUN 218 P FASIN

Ladder symbol



S: Source data or register to be calculated the arc sine value

D: Register for storing the result

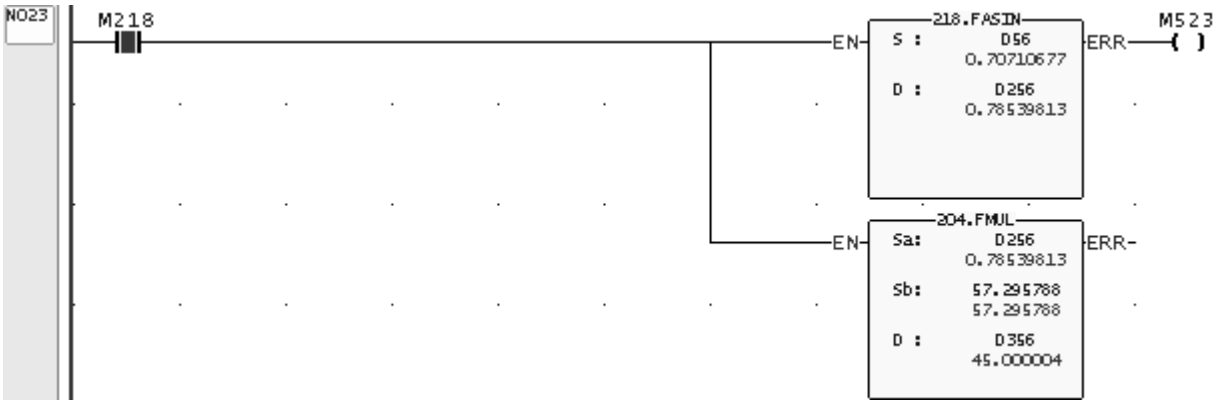
S, D may combine with V, Z, P0~P9 to serve indirect address application

Range / Operand	HR	ROR	DR	K	XR
	R0 | R3838	R5000 | R8070	D0 | D4094	Floating number	V、Z P0～P9
S	○	○	○	○	○
D	○	○*	○		○

Description

- The format of floating point number of WSZ-Controller follows the IEEE-754 standard of 32-bit.
- When operation control "EN" =1 or "EN↑" (P instruction) changes from 0 to 1, calculate the arc sine value of the data specified by the S value or S~S+1 register, and store the result into the register specified by D~D+1.
- Range of S data : -1~ +1 ; range of D value : $-\pi/2$ ~ $\pi/2$ (Unit in radian)
- If the value of S is out of range, or invalid indirect addressing, the error flag "ERR" will be set to 1, and not update the value of D~D+1.
- All floating point instructions can't be executed in interrupt service routine.

Example



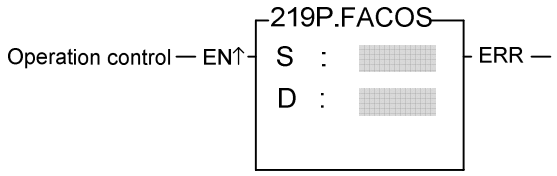
- When M218=1, calculate the arc sine value, it is DD256 = $\sin^{-1}$ DD56; DD256(Unit in radian) × 57.295788(180/π) to acquire the degree value

Status Monitoring

Ref. No.	Status	Data	Ref. No.	Status	Data
DD56	Floating	0.70710677			
DD256	Floating	0.78539813			
M218	Enable	ON			
DD356	Floating	45.000004			

FUN 219 P FACOS	FLOATING POINT ARC COSINE FUNCTION, $\cos^{-1}$	FUN 219 P FACOS

Ladder symbol



S: Source data or register to be calculated the arc cosine value

D: Register for storing the result

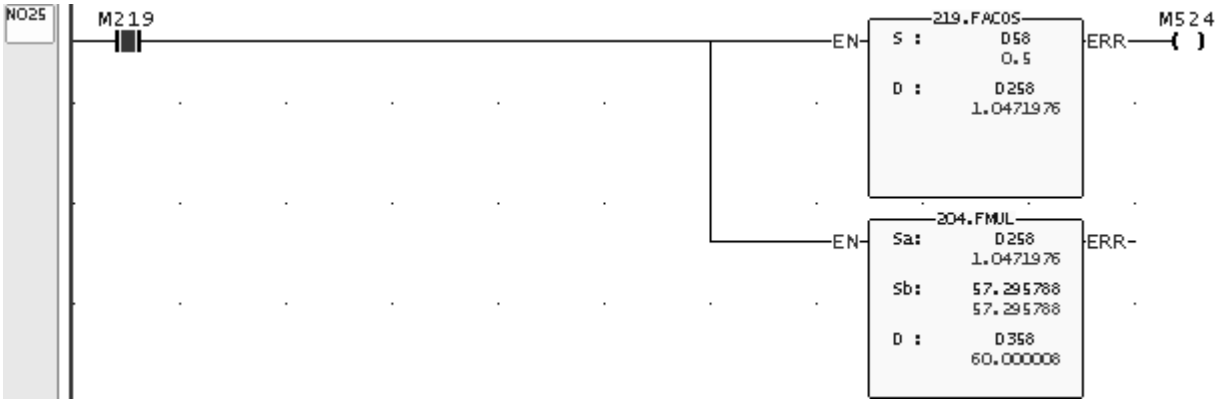
S, D may combine with V, Z, P0~P9 to serve indirect address application

Operand \ Range	HR	ROR	DR	K	XR
	R0 | R3838	R5000 | R8070	D0 | D4094	Floating number	V、Z P0～P9
S	○	○	○	○	○
D	○	○*	○		○

Description

- The format of floating point number of WSZ-Controller follows the IEEE-754 standard of 32-bit.
- When operation control "EN" =1 or "EN↑" (P instruction) changes from 0 to 1, calculate the arc cosine value of the data specified by the S value or S~S+1 register, and store the result into the register specified by D~D+1.
- Range of S data : -1~ +1 ; range of D value : 0 ~ π (Unit in radian)
- If the value of S is out of range, or invalid indirect addressing, the error flag "ERR" will be set to 1, and not update the value of D~D+1.
- All floating point instructions can't be executed in interrupt service routine.

Example



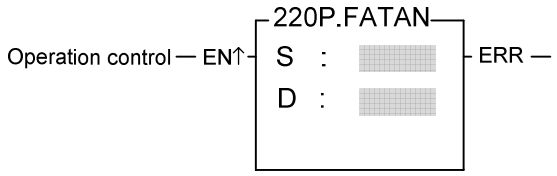
- When M219=1, calculate the arc cosine value, it is DD258 = $\cos^{-1}$ DD58; DD258(Unit in radian) × 57.295788(180/π) to acquire the degree value

Status Monitoring

Ref. No.	Status	Data	Ref. No.	Status	Data
DD58	Floating	0.5			
DD258	Floating	1.0471976			
M219	Enable	ON			
DD358	Floating	60.000008			

FUN 220 P FATAN	FLOATING POINT ARC TANGENT FUNCTION, tan^{-1}	FUN 220 P FATAN

Ladder symbol



S: Source data or register to be calculated the arc tangent value

D: Register for storing the result

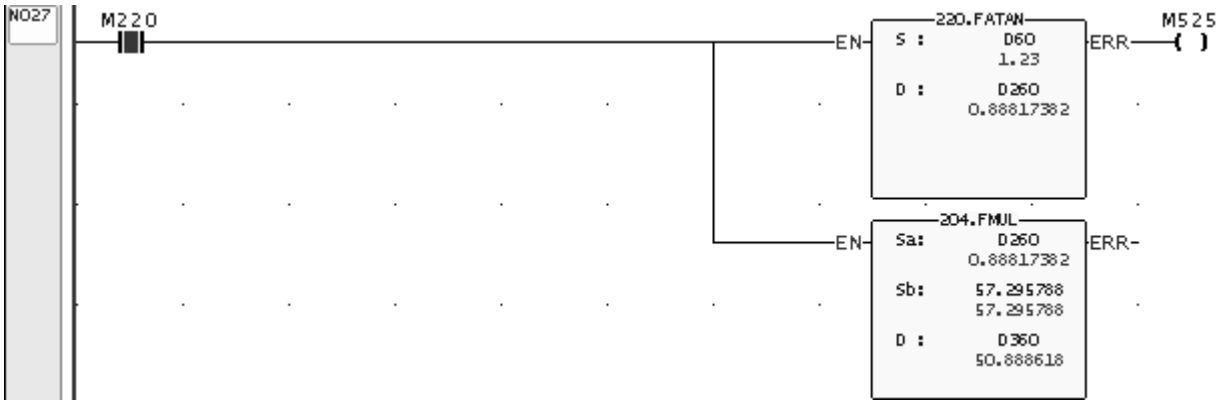
S, D may combine with V, Z, P0~P9 to serve indirect address application

Range / Operand	HR	ROR	DR	K	XR
	R0 | R3838	R5000 | R8070	D0 | D4094	Floating number	V、Z P0～P9
S	○	○	○	○	○
D	○	○*	○		○

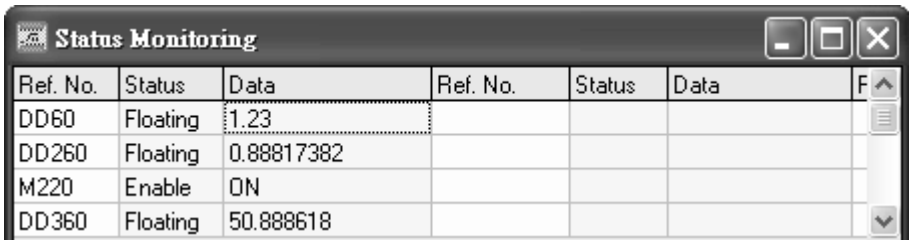
Description

- The format of floating point number of WSZ-Controller follows the IEEE-754 standard of 32-bit.
- When operation control "EN" =1 or "EN↑" (P instruction) changes from 0 to 1, calculate the arc tangent value of the data specified by the S value or S~S+1 register, and store the result into the register specified by D~D+1.
- S data is any number ; range of D value : $-\pi/2 \sim \pi/2$ (Unit in radian)
- If it exists invalid indirect addressing, the error flag "ERR" will be set to 1, and not update the value of D~D+1.
- All floating point instructions can't be executed in interrupt service routine.

Example

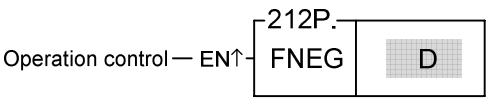


- When M220=1, calculate the arc tangent value, it is DD260 = tan^{-1} DD60; DD260(Unit in radian) × 57.295788($180/\pi$) to acquire the degree value


Status Monitoring
Ref. No. Status Data Ref. No. Status Data
DD60 Floating 1.23
DD260 Floating 0.88817382
M220 Enable ON
DD360 Floating 50.888618

FUN 212 P FNEG	CHANGE SIGN OF THE FLOATING POINT NUMBER	FUN 212 P FNEG

Ladder symbol



D : Register to be changed sign

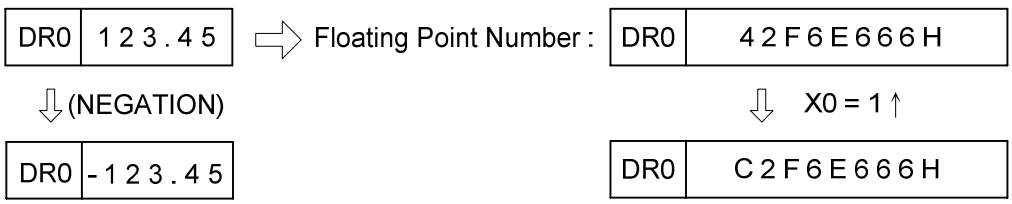
D may combine with V, Z, P0~P9 to serve indirect address application.

Range / Operand	HR	ROR	DR	XR
	R0 ~ R3839	R5000 ~ R8071	D0 ~ D4095	V, Z P0～P9
D	○	○*	○	○

Description

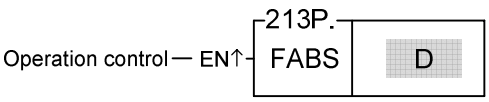
- The format of floating point number of WSZ-controller follows the IEEE-754 standard.
- When operation control "EN" =1 or "EN↑" (P instruction) changes from 0 to 1, the sign of the floating point number register specified by D will be toogled.

- The instruction at left negates the value of the DR0 register, and stores it back to DR0.



FUN 213 P FABS	FLOATING POINT NUMBER ABSOLUTE VALUE	FUN 213 P FABS

Ladder symbol



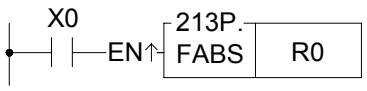
D : Register to be taken absolute value

D may combine with V, Z, P0~P9 to serve indirect address application.

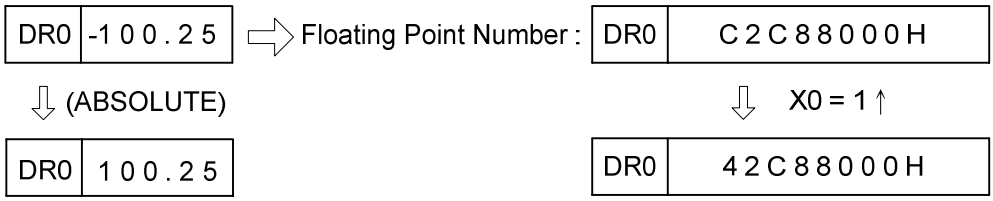
Range / Operand	HR	ROR	DR	XR
	R0 ~ R3839	R5000 ~ R8071	D0 ~ D4095	V, Z P0～P9
D	○	○*	○	○

Description

- The format of floating point number of WSZ-controller follows the IEEE-754 standard.
- When operation control "EN" =1 or "EN↑" (P instruction) changes from 0 to 1, calculate the absolute value of the floating point number register specified by D, and write it back into the original D register.



- The instruction at left calculates the absolute value of the DR0 register, and stores it back in DR0.



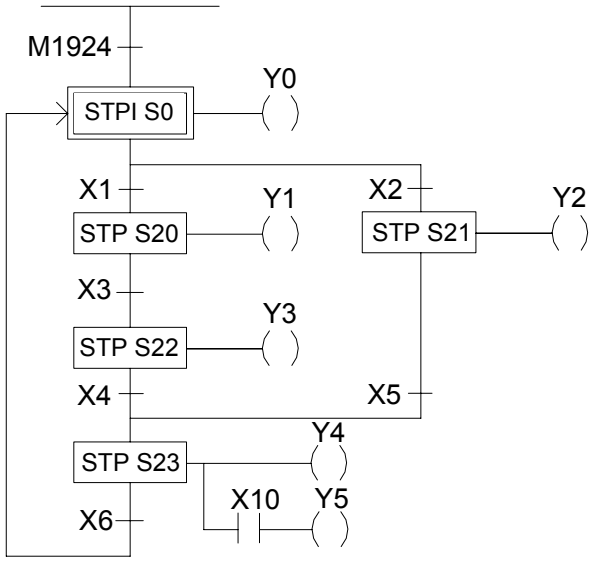
Chapter 8 Step Instruction Description

Structured programming design is a major trend in software design. The benefits are high readability, easy maintenance, convenient updating and high quality and reliability. The control application which is consisted of many sequential tasks, and designed by conventional ladder program design methodology usually makes others hard to maintain. Therefore, it is necessary to combine the current widely used ladder diagrams with the sequential controls made especially for machine working flow. With help from step instructions, the design work will become more efficient, time saving and controlled. This kind of design method that combines process control and ladder diagram together is called the step ladder language.

The basic unit of step ladder diagram is a step. A step is equivalent to a movement (stop) in the machine operation where each movement has an output. The complete machine or the overall sequential control process is the combination of steps in serial or parallel. Its step-by-step sequential execution procedure allows others to be able to understand the machine operations thoroughly, so that design, operation, and maintenance will become more effective and simpler.

8.1 The Operation Principle of Step Ladder Diagram

【Example】

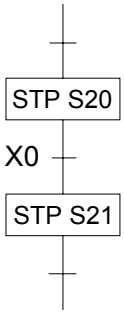


【Description】

1. STP Sxxx is the symbol representing a step Sxxx that can be one of S0 ～ S999. When executing the step (status ON), the ladder diagram on the right will be executed and the previous step and output will become OFF.
2. M1924 is on for a scan time after program start. Hence, as soon as ON, the initial step S0 is entered (S0 ON) while the other steps are kept inactive, i.e. Y1～Y5 are all OFF. This means M1924 ON→S0 ON→Y0 ON and Y0 will remain ON until one of the contacts X1 or X2 is ON.
3. Assume that X2 is ON first; the path to S21 will then be executed.
   X2 ON⇒ { S21 ON, S0 OFF } ⇒ { Y2 ON, Y0 OFF }
   Y2 will remain ON until X5 is ON.
4. Assume that X5 is ON, the process will move forward to step S23.
   i.e. X5 ON⇒ { S23 ON, S21 OFF } ⇒ { Y4 ON, Y2 OFF }
   Y4 and Y5 will remain ON until X6 is ON.
   ※If X10 is ON, then Y5 will be ON.
5. Assume that X6 is ON, the process will move forward to S0.
   i.e. X6 ON⇒ { S0 ON, S23 OFF } ⇒ { Y0 ON, Y4、Y5 OFF }
   Then, a control process cycle is completed and the next control process cycle is entered.

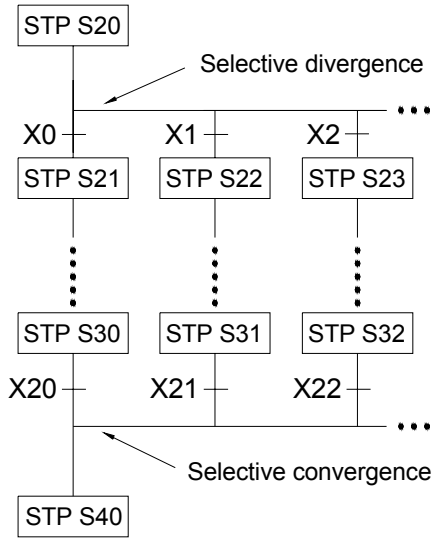
8.2 Basic Formation of Step Ladder Diagram

① Single path



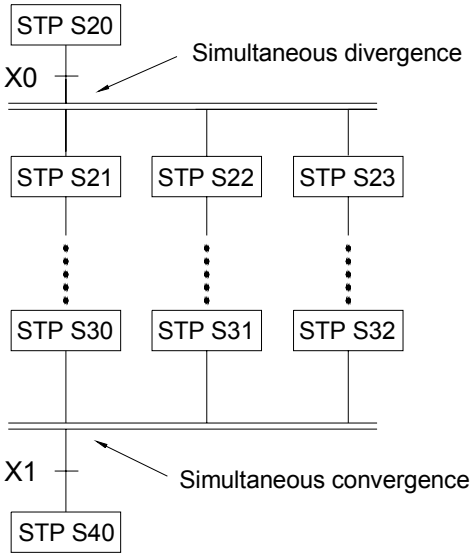
- Step S20 alone moves to step S21 through X0.
- X0 can be changed to other serial or parallel combination of contacts.

② Selective divergence/convergence



- Step S20 selects an only one path which divergent condition first met. E.g. X2 is ON first, then only the path of step S23 will be executed.
- A divergence may have up to 8 paths maximum.
- X1, X2, ….., X22 can all be replaced by the serial or parallel combination of other contacts.

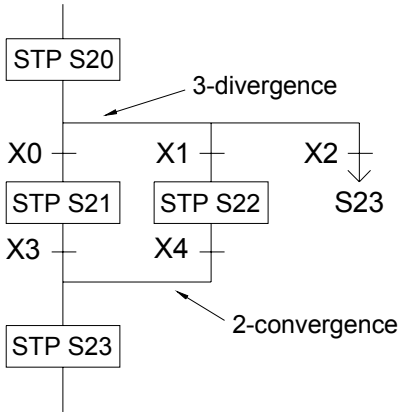
③ Simultaneous divergence/convergence



- After X0 is ON, step S20 will simultaneously execute all paths below it, i.e. all S21, S22, S23, and so on, are in action.
- All divergent paths at a convergent point will be executed to the last step (e.g. S30, S31 and S32). When X1 is ON, they can then transfer to S40 for execution.
- The number of divergent paths must be the same as the number of convergent paths. The maximum number of divergence/convergence path is 8.

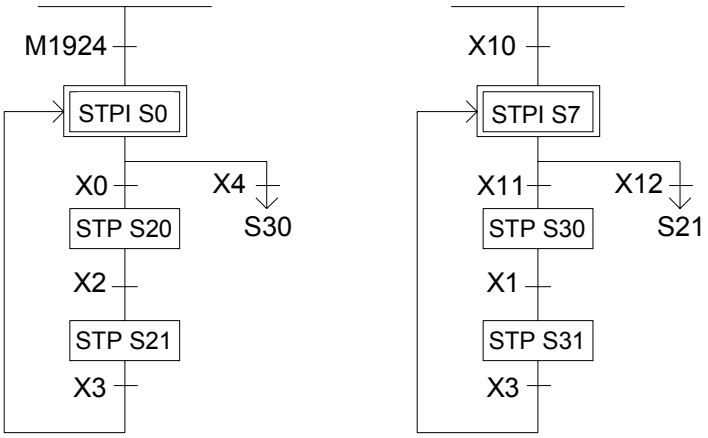
④ Jump

a. The same step loop



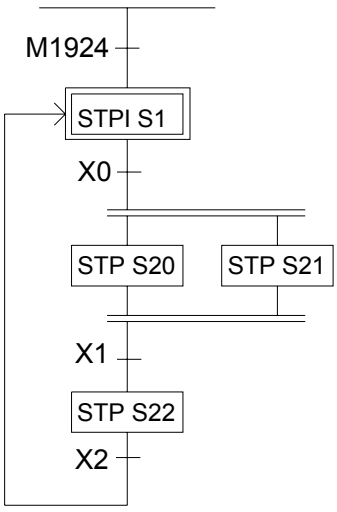
- There are 3 paths below step S20 as shown on the left. Assume that X2 is ON, then the process can jump directly to step S23 to execute without going through the process of selective convergence.
- The execution of simultaneous divergent paths can not be skipped.

b. Different step loop

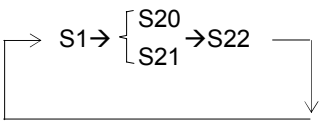


⑤Closed Loop and Single Cycle

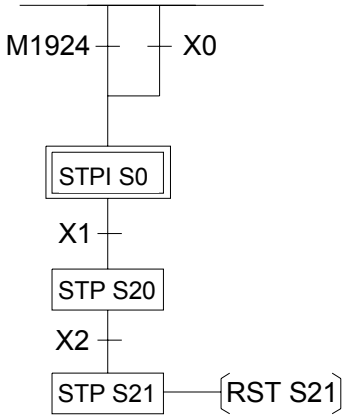
a. Closed Loop



- The initial step S1 is ON, endless cycle will be continued afterwards.

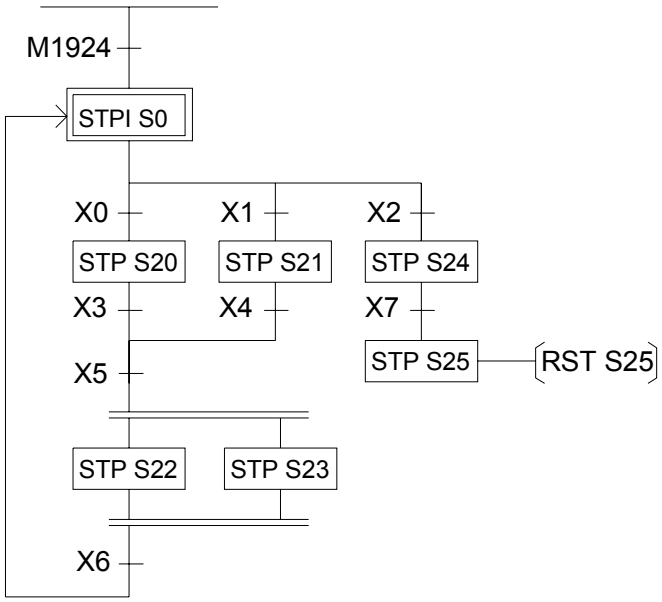


b. Single Cycle

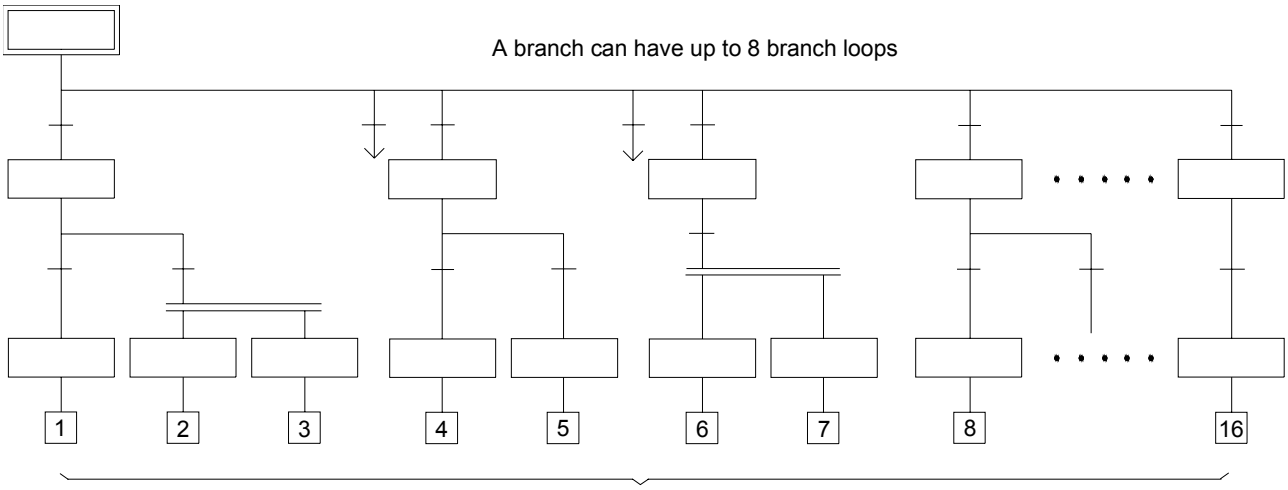


- When step S20 is ON, if X2 is also ON, then “RST S21” instruction will let S21 OFF which will stop the whole step process.

c. Mixed Process



⑥ Combined Application



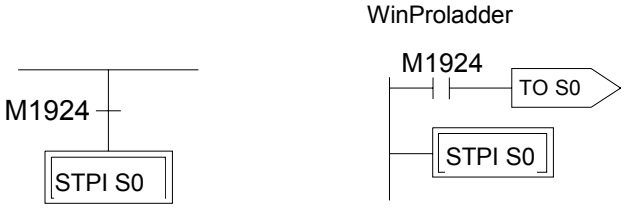
The maximum number of downward horizontal branch loops of an initial step is 16

8.3 Introduction of Step Instructions: STP, FROM, TO and STPEND

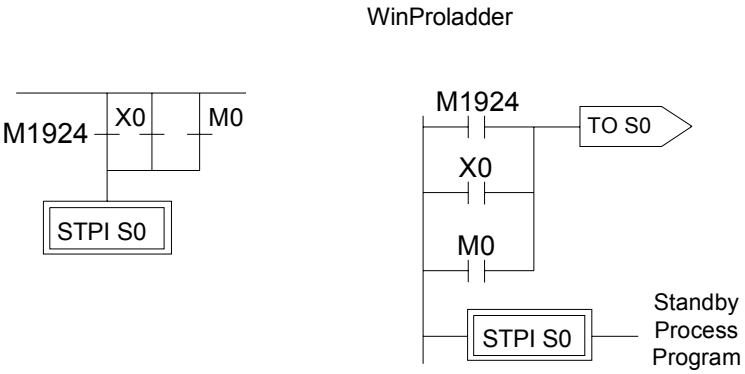
● STPI Sx : S0≦Sx≦S7 (Displayed in WinProladder)

This instruction is the initial step instruction from where the step control of each machine process can be derived. Up to 8 initial steps can be used in the WSZ series, i.e. a controller can make up to 8 process controls simultaneously. Each step process can operate independently or generate results for the reference of other processes.

【Example 1】 Go to the initial step S0 after each start (ON)



【Example 2】 Each time the device is start to run or the manual button is pressed or the device is malfunction, then the device automatically enters the initial step S0 to standby.

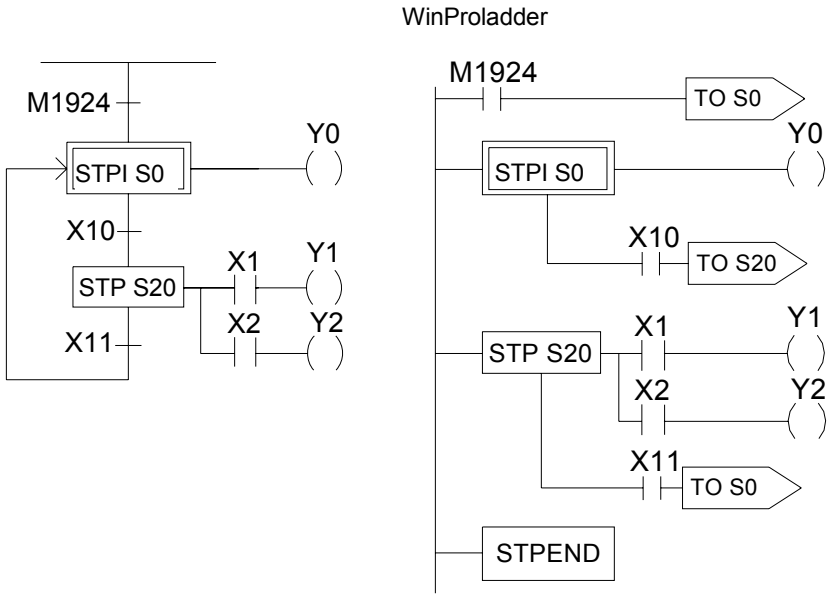


【Description】X0: Manual Button, M0: Abnormal Contact.

● STP Sxxx : S20≦Sxxx≦S999（Displayed in WinProladder）

This instruction is a step instruction, each step in a process represents a step of sequence. If the status of step is ON then the step is active and will execute the ladder program associate to the step.

【Example】



【Description】1. When ON, the initial step S0 is ON and Y0 is ON.

2. When transfer condition X10 is ON (in actual application, the transferring condition may be formed by the serial or parallel combination of the contacts X, Y, M, T and C), the step S20 is activated. The system will automatically turn S0 OFF in the current scan cycle and Y0 will be reset automatically to OFF.

i.e. X10 ON⇒ { S20 ON, S0 OFF } ⇒ { X1 ON →Y1 ON, X2 ON →Y2 ON, Y0 OFF }

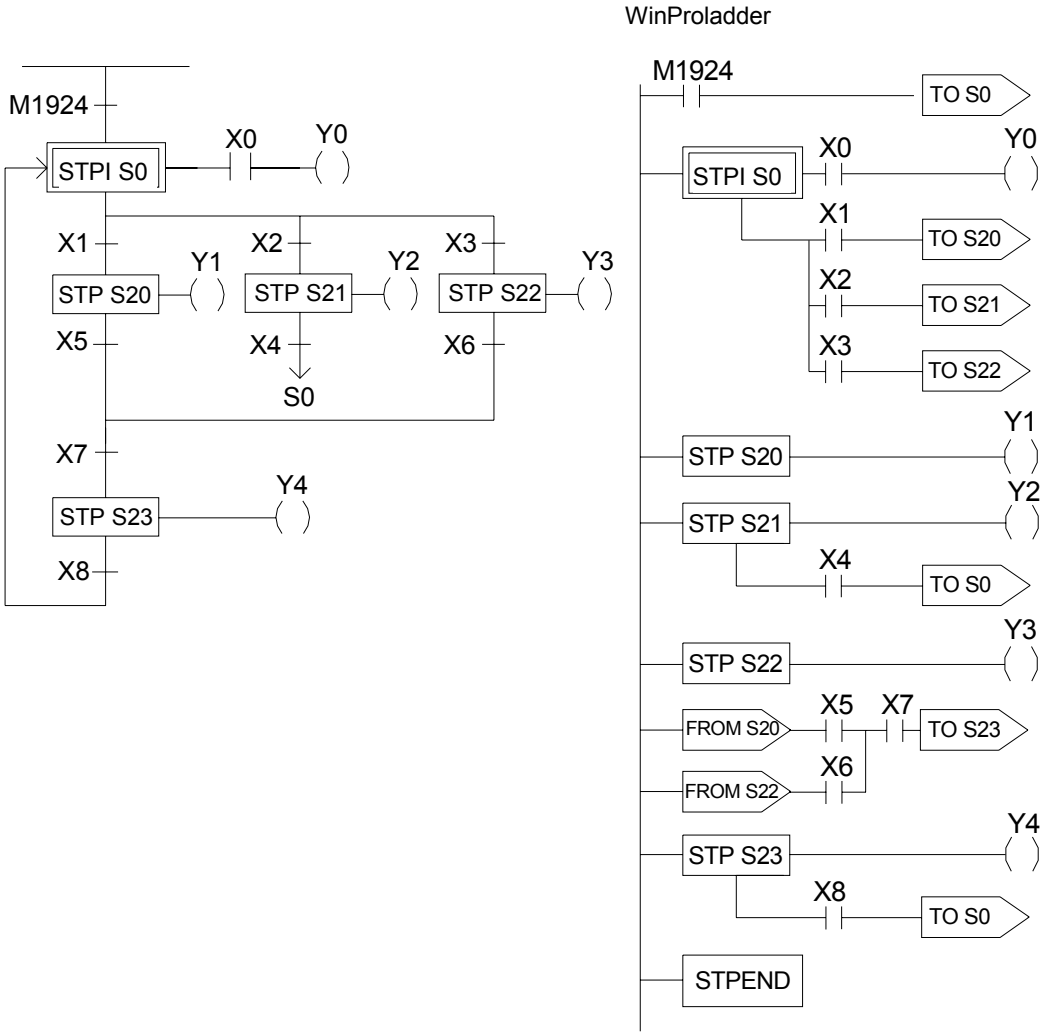
3. When the transfer condition X11 is ON, the step S0 is ON, Y0 is ON and S20, Y1 and Y2 will turn OFF at the same time.

i.e. X11 ON⇒ { S0 ON, S20 OFF } ⇒ { Y0 ON, Y1 OFF, Y2 OFF }

● FROM Sxxx : S0≦Sxxx≦S999（Displayed in WinProladder）

The instruction describes the source step of the transfer, i.e. moving from step Sxxx to the next step in coordination with transfer condition.

【Example】



【Description】: 1. When ON, the initial step S0 is ON. If X0 is ON, then Y0 will be ON.

2. When S0 is ON: a. if X1 is ON, then step S20 will be ON and Y1 will be ON.

   b. if X2 is ON, then step S21 will be ON and Y2 will be ON.

   c. if X3 is ON, then step S22 will be ON and Y3 will be ON.

   d. if X1, X2 and X3 are all ON simultaneous, then step S20 will have the priority to be ON first and either S21 or S22 will not be ON.

   e. if X2 and X3 are ON at the same time, then step S21 will have the priority to be ON first and S22 will not be ON.

3. When S20 is ON, if X5 and X7 are ON at the same time, then step S23 will be ON, Y4 will be ON and S20 and Y1 will be OFF.

4. When S21 is ON, if X4 is ON, then step S0 will be ON and S21 and Y2 will be OFF.

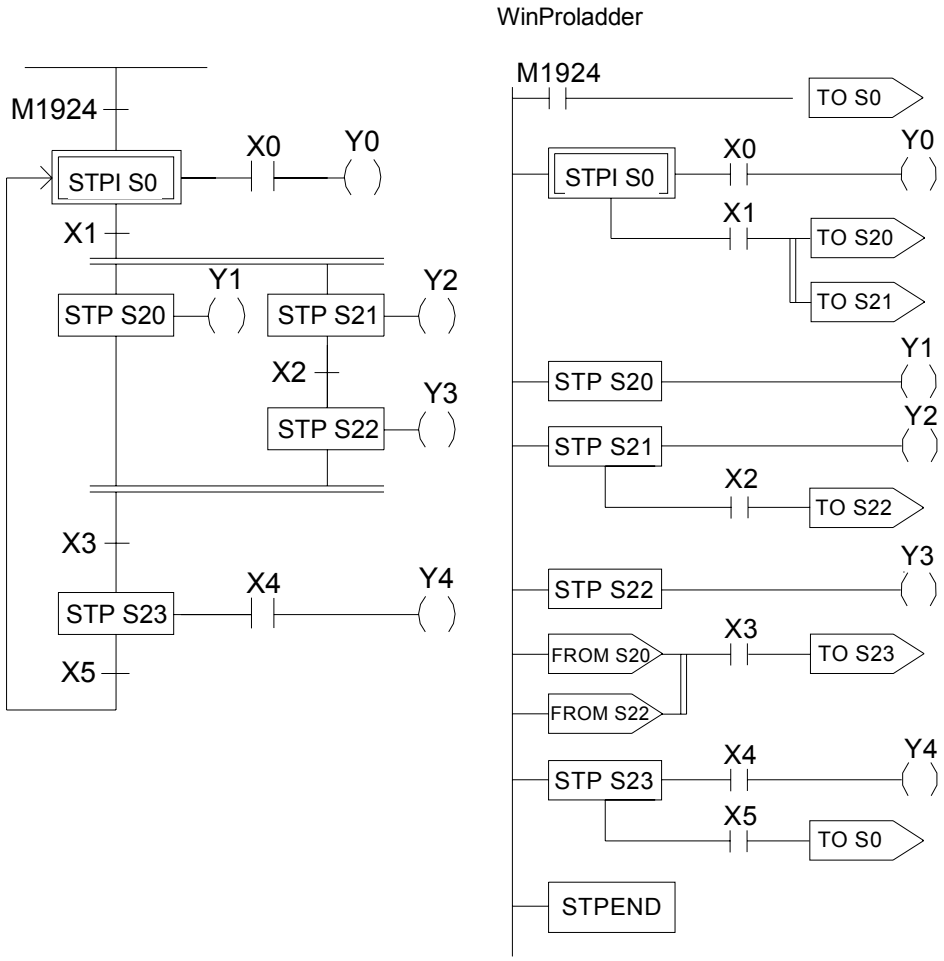
5. When S22 is ON, if X6 and X7 are ON at the same time, then step S23 will be ON, Y4 will be ON and S22 and Y3 will be OFF.

6. When S23 is ON, if X8 is ON, then step S0 will be ON and S23 and Y4 will be OFF.

● TO Sxxx : S0≦Sxxx≦S999（Displayed in WinProladder）

This instruction describes the step to be transferred to.

【Example】



【Description】: 1. When ON, the initial step S0 is ON. If X0 is ON, then Y0 will be ON.

2. When S0 is ON: if X1 is ON, then steps S20 and S21 will be ON simultaneously and Y1 and Y2 will also be ON.

3. When S21 is ON: if X2 is ON, then step S22 will be ON, Y3 will be ON and S21 and Y2 will be OFF.

4. When S20 and S22 are ON at the same time and the transferring condition X3 is ON, then step S23 will be ON (if X4 is ON, then Y4 will be ON) and S20 and S22 will automatically turn OFF and Y1 and Y3 will also turn OFF.

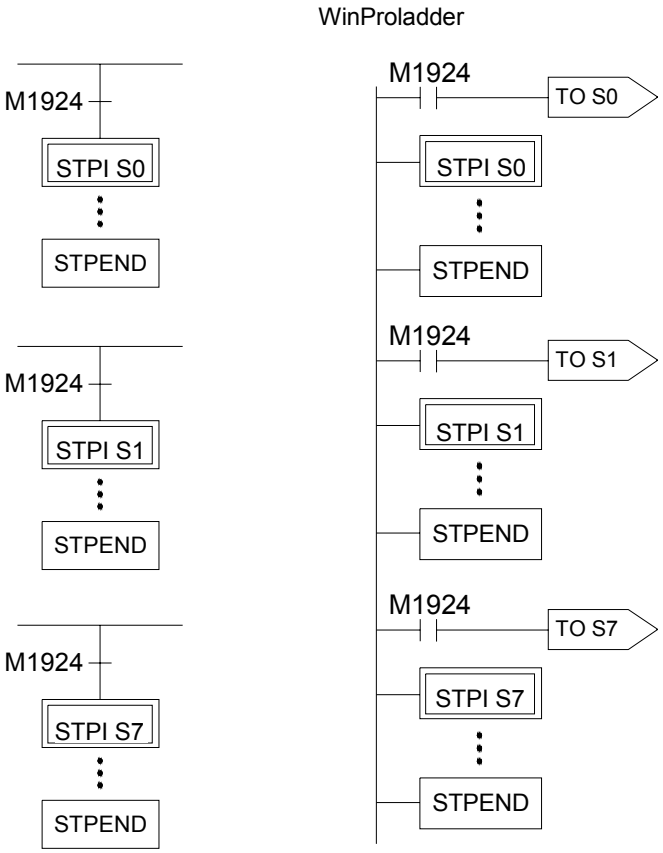
5. When S23 is ON: if X5 is ON, then the process will transfer back to the initial step, i.e. So will be ON and S23 and Y4 will be OFF.

● STPEND ：（Displayed in WinProladder）

This instruction represents the end of a process. It is necessary to include this instruction so all processes can be operated correctly.

A controller can have up to 8 step processes (S0～S7) and is able to control them simultaneously. Therefore, up to 8 STPEND instructions can be obtained.

【Example】

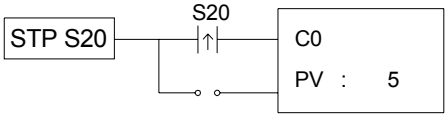


【Description】 When ON, the 8 step processes will be active simultaneously.

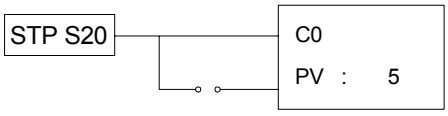
8.4 Notes for Writing a Step Ladder Diagram

【Notes】

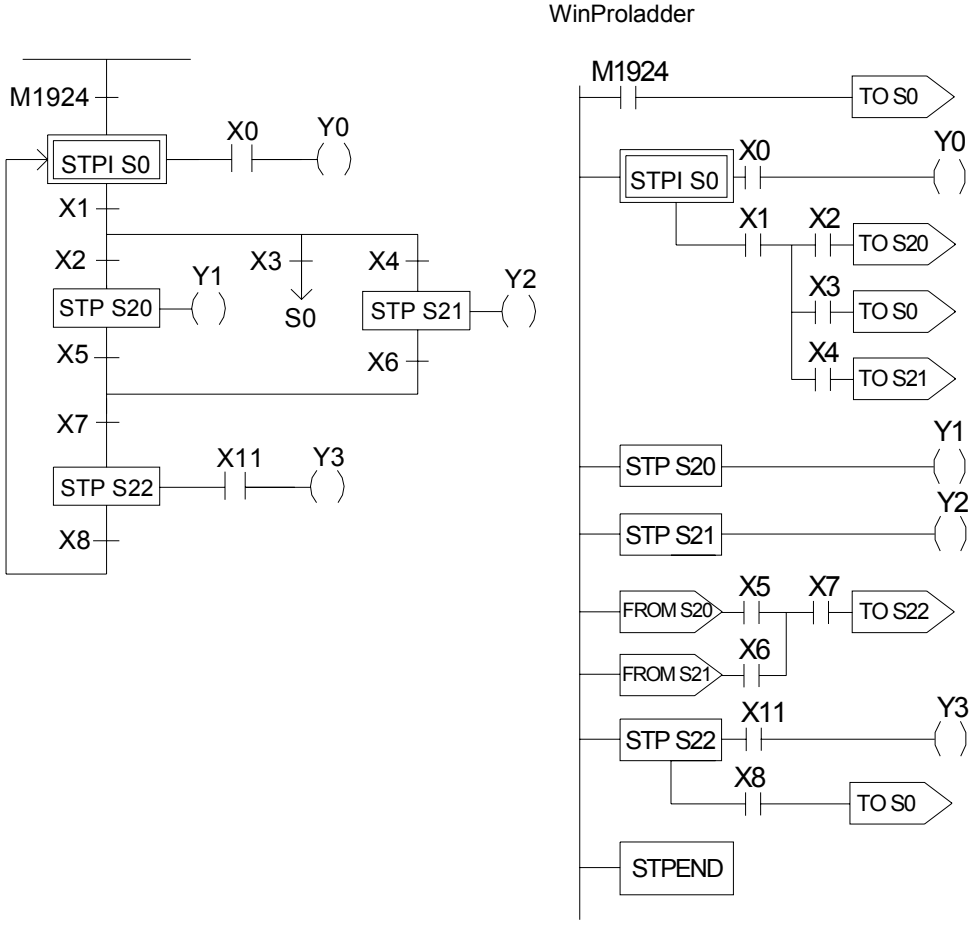
- In actual applications, the ladder diagram can be used together with the step ladder.
- There are 8 steps, S0～S7, that can be used as the starting point and are called the “initial steps”.
- When controller starts operating, it is necessary to activate the initial step. The M1924 (the first scan ON signal) provided by the system may be used to activate the initial step.
- Except the initial step, the start of any other steps must be driven by other step.
- It is necessary to have an initial step and the final STPEND instruction in a step ladder diagram to complete a step process program.
- There are 980 steps, S20～S999, available that can be used freely. However, used numbers cannot be repeated. S500～S999 are retentive(The range can be modified by users), can be used if it is required to continue the machine process after power is off.
- Basically a step must consists of three parts which are control output, transition conditions and transition targets.
- MC and SKP instructions cannot be used in a step program and the sub-programs. It’s recommended that JMP instruction should be avoided as much as possible.
- If the output point is required to stay ON after the step is divergent to other step, it is necessary to use the SET instruction to control the output point and use RST instruction to clear the output point to OFF.
- Looking down from an initial step, the maximum number of horizontal paths is 16. However, a step is only allowed to have up to 8 branch paths.
- When M1918=0（default）, if a PULSE type function instruction is used in master control loop (FUN 0) or a step program, it is necessary to connect a TU instruction before the function instruction. For example,



When M1918=1, the TU instruction is not required, e.g.:



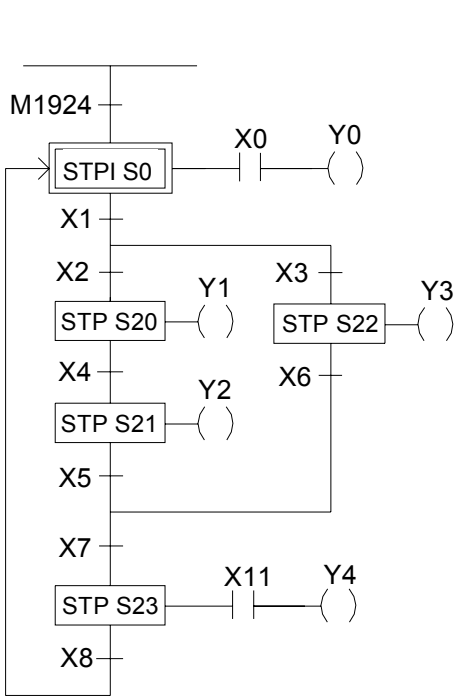
Example 1



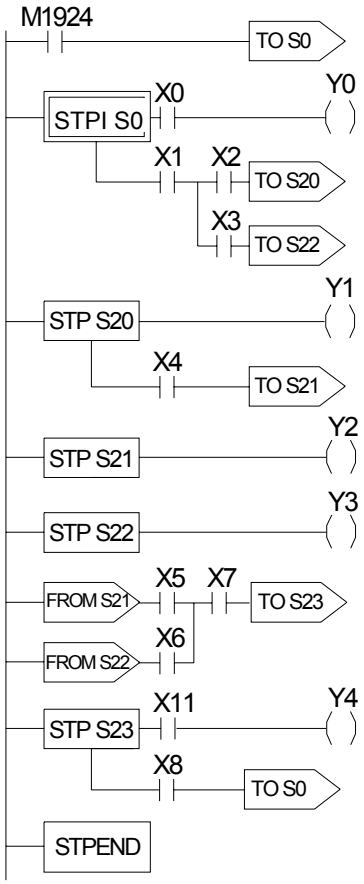
Description

1. Input the condition to initial step S0
2. Input the S0 and the divergent conditions of S20, S0 and S21
3. Input the S20
4. Input the S21
5. Input the convergence of S20 and S21
6. Input the S22

Example 2



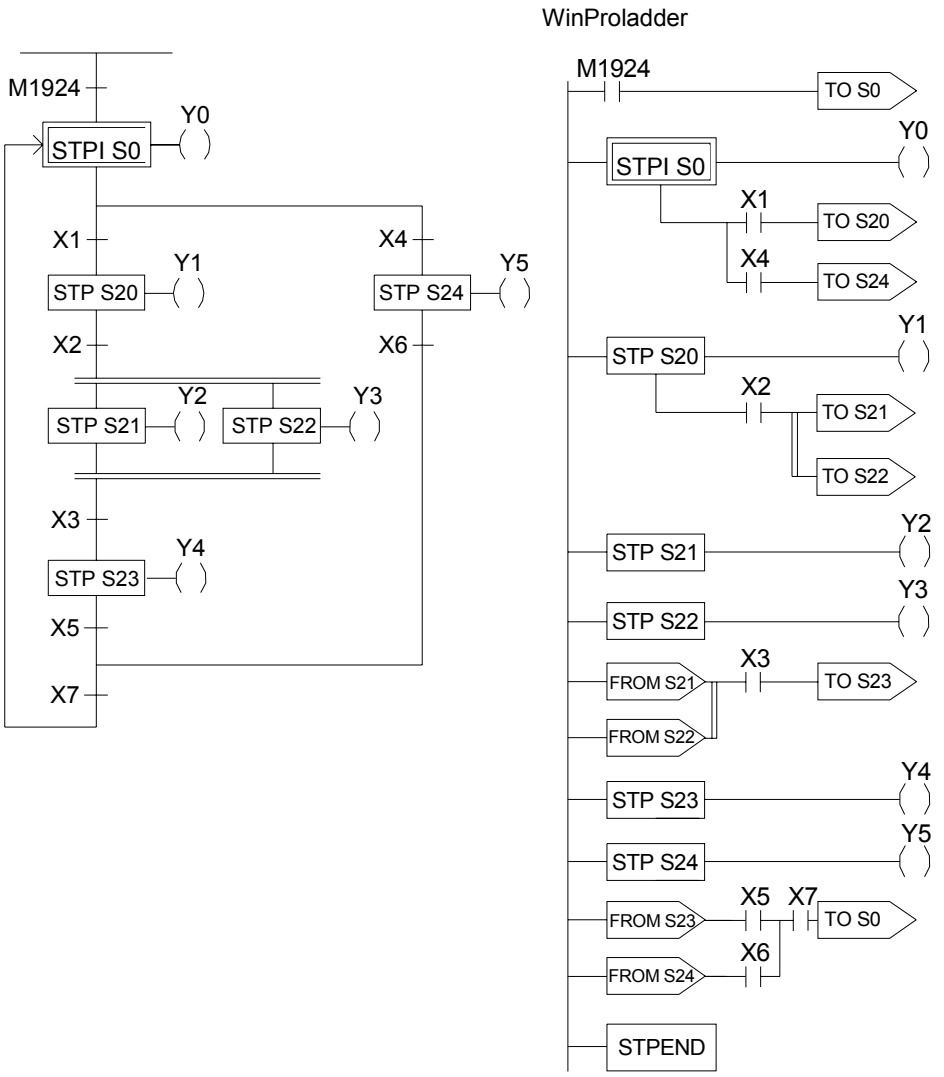
WinProladder



Description

1. Input the condition to initial step S0
2. Input the S0 and the divergent condition of S20 and S22
3. Input the S20
4. Input the S21
5. Input the S22
6. Input the convergence of S21 and S22
7. Input the S23

Example 3

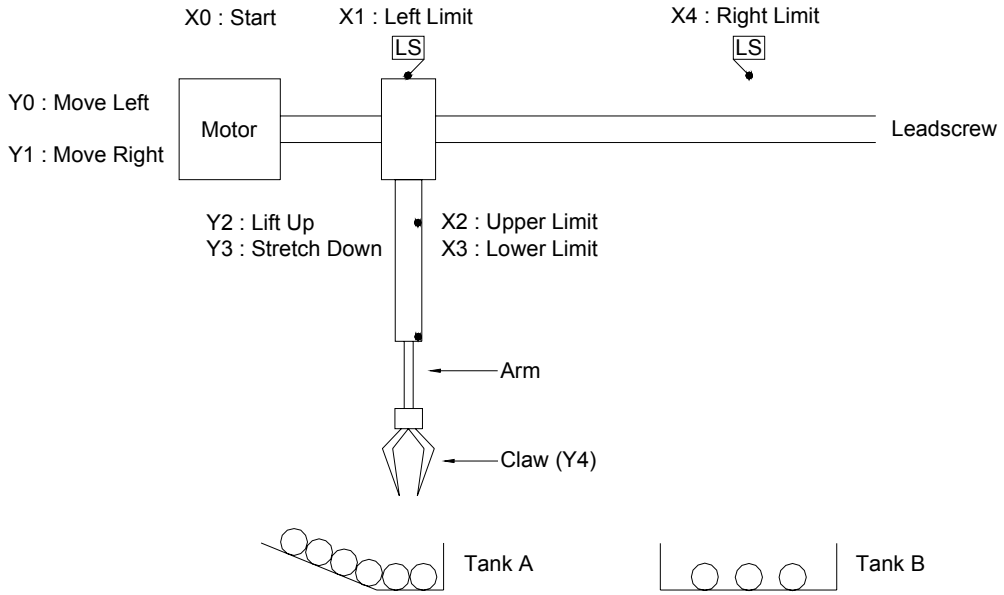


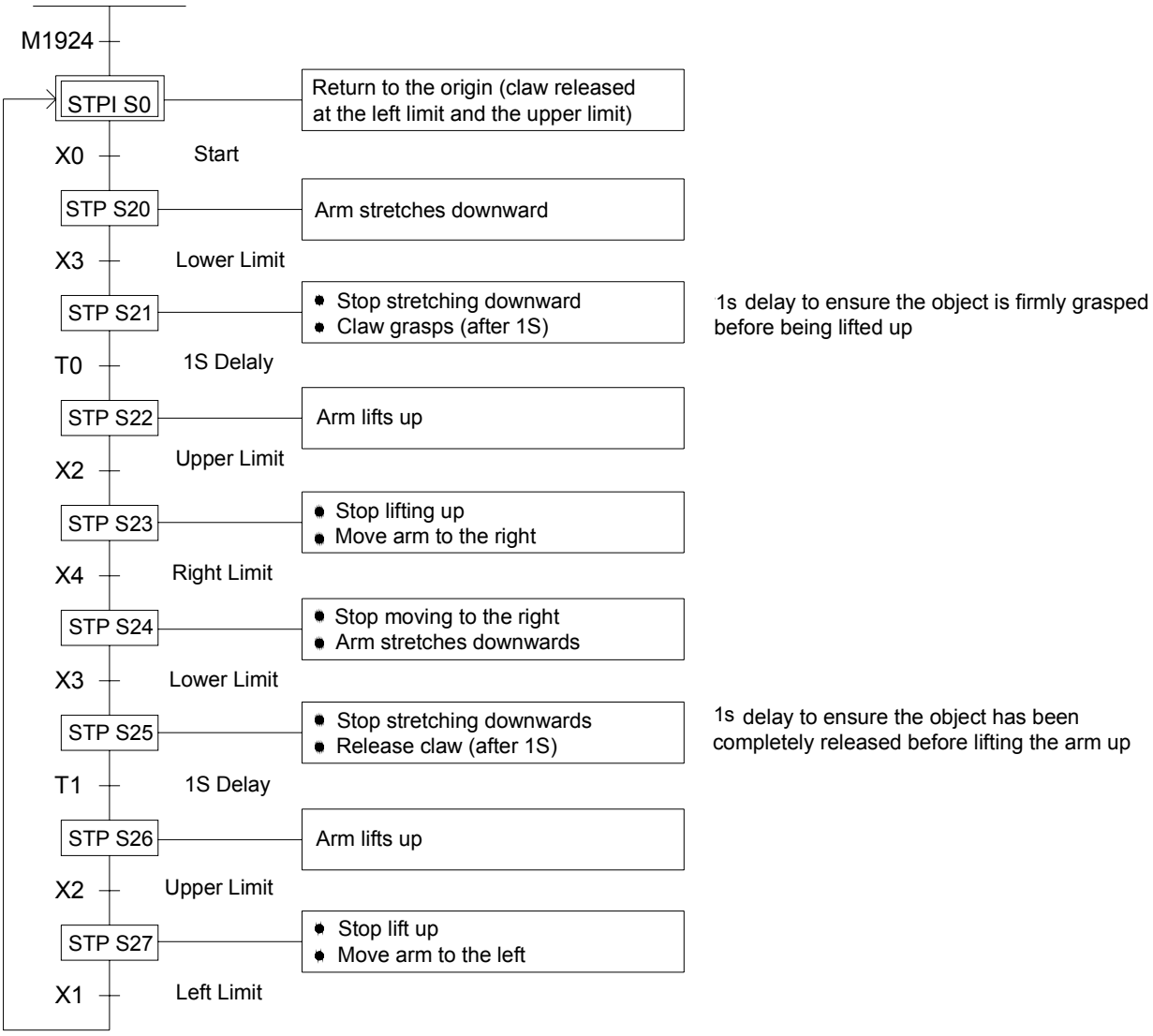
Description

1. Input the condition to initial step S0
2. Input the S0 and the divergences of S20 and S24
3. Input the S20
4. Input the S20 and the divergences of S21 and S22
5. Input the S21
6. Input the S22
7. Input the convergences of S21 and S22
8. Input the S23
9. Input the S24
10. Input the convergences of S23 and S24

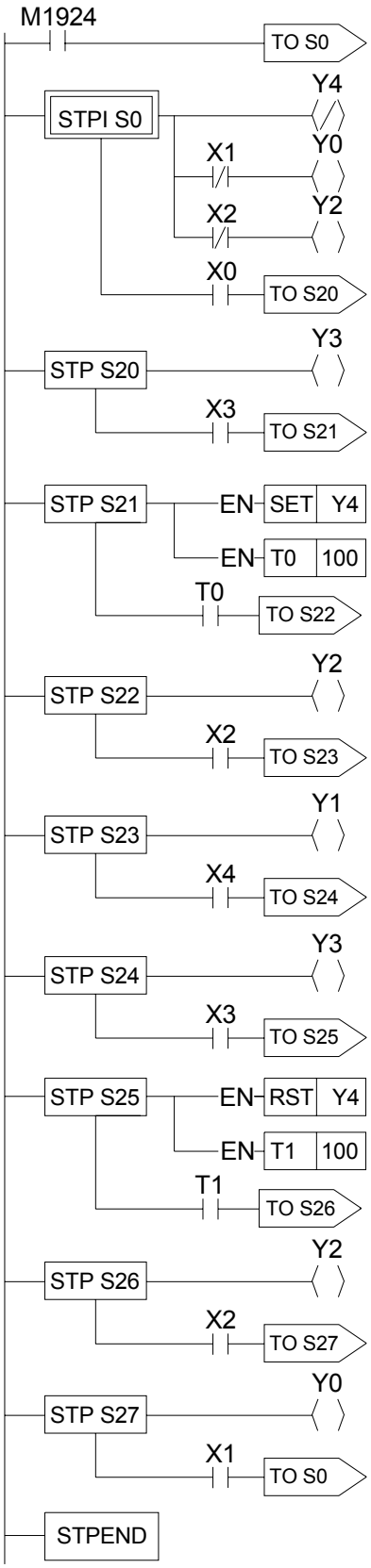
8.5 Application Examples

Example 1 Grasp an object from tank A and put it in Tank B





WinProladder



Release claw

Return to the left limit

Return to the upper limit

Turn the switch ON before moving to S20

Stretch arm downward

Move to S21 after stretching to the lower limit

Claw grasps (since the SET instruction is used, Y4 should remain ON after departing from STP S21)

Divergent into S22 after 1S

Lift the arm up

Divergent into S23 after reaching the upper limit

Move arm to the right

Divergent into S24 after moving to the right limit

Stretch the arm downward

Divergent into S25 after stretching to the lower limit

Release claw

Delay for 1S

Transfer into S26 after 1S

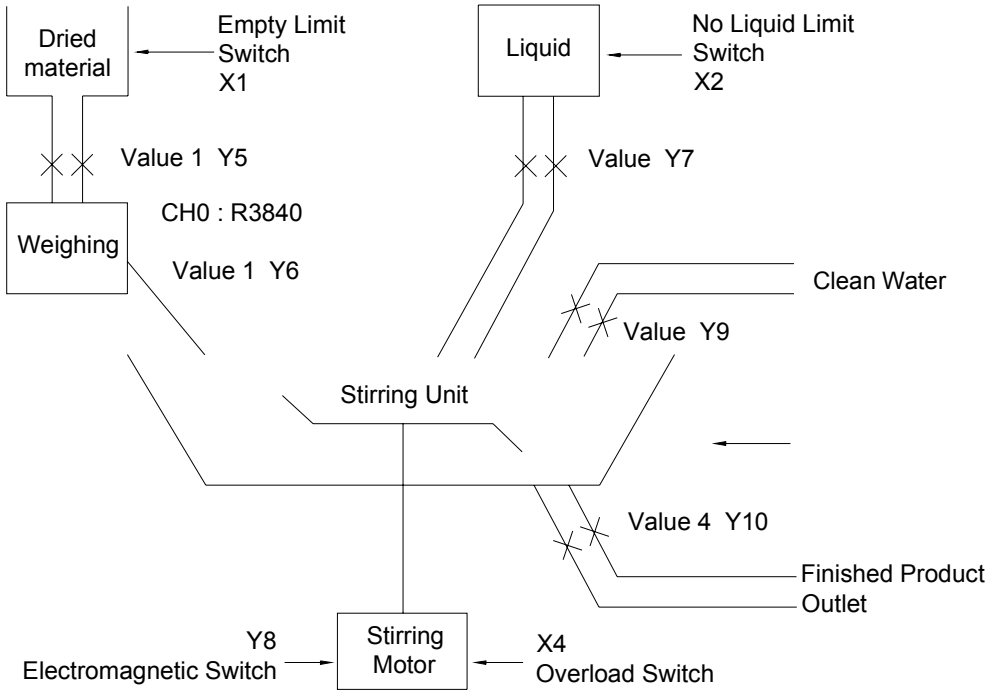
Lift the arm up

Divergent into S27 after reaching the upper limit

Move the arm to the left

Divergent into S0 after moving to the left limit (a complete cycle)

Example 2 Liquid Stirring Process



- Input Points: Empty limit switch X1
  No liquid limit switch X2
  Empty limit switch X3
  Over-load switch X4
  Warning clear button X5
  Start button X6
  Water washing button X7

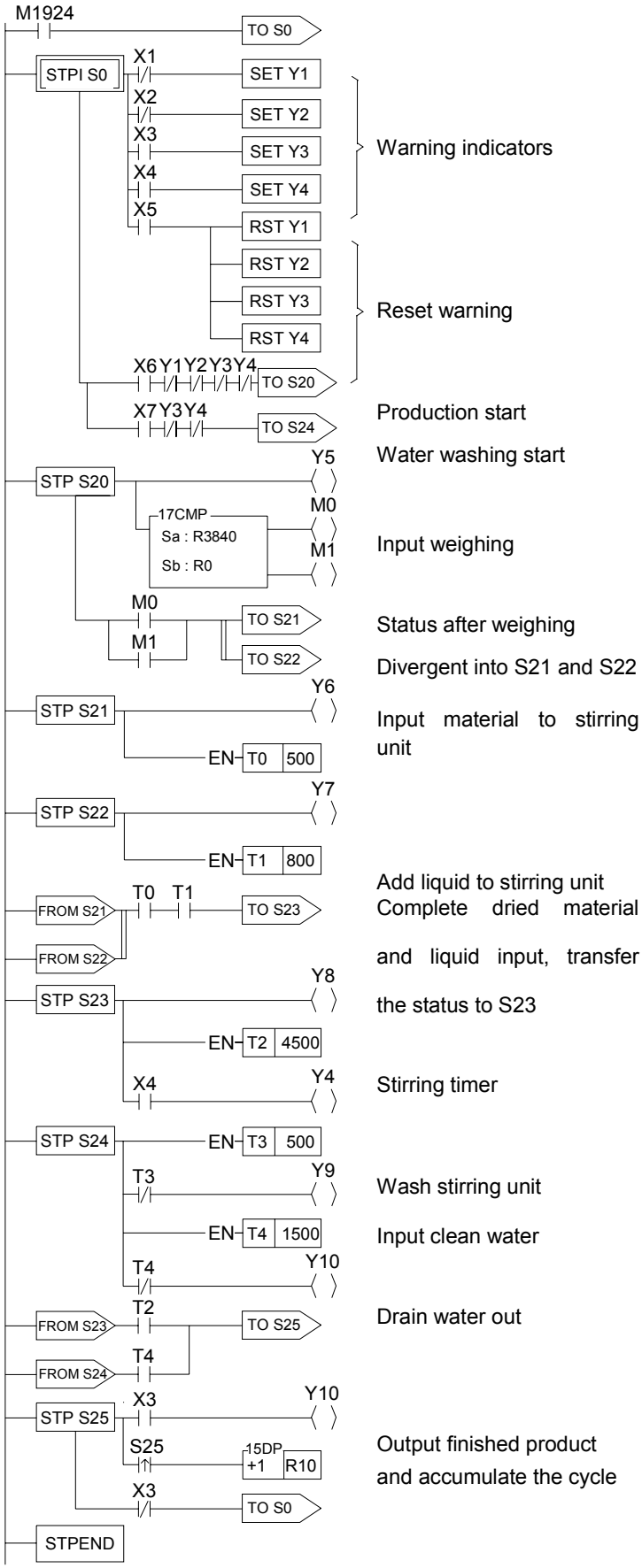
- Warning Indicators: Empty dried material Y1
  Insufficient liquid Y2
  Empty stirring unit Y3
  Motor over-load Y4

- Output Points: Dried material inlet valve Y5
  Dried material inlet valve Y6
  Liquid inlet valve Y7
  Motor start electromagnetic valve Y8
  Clean water inlet valve Y9
  Finished product outlet valve Y10

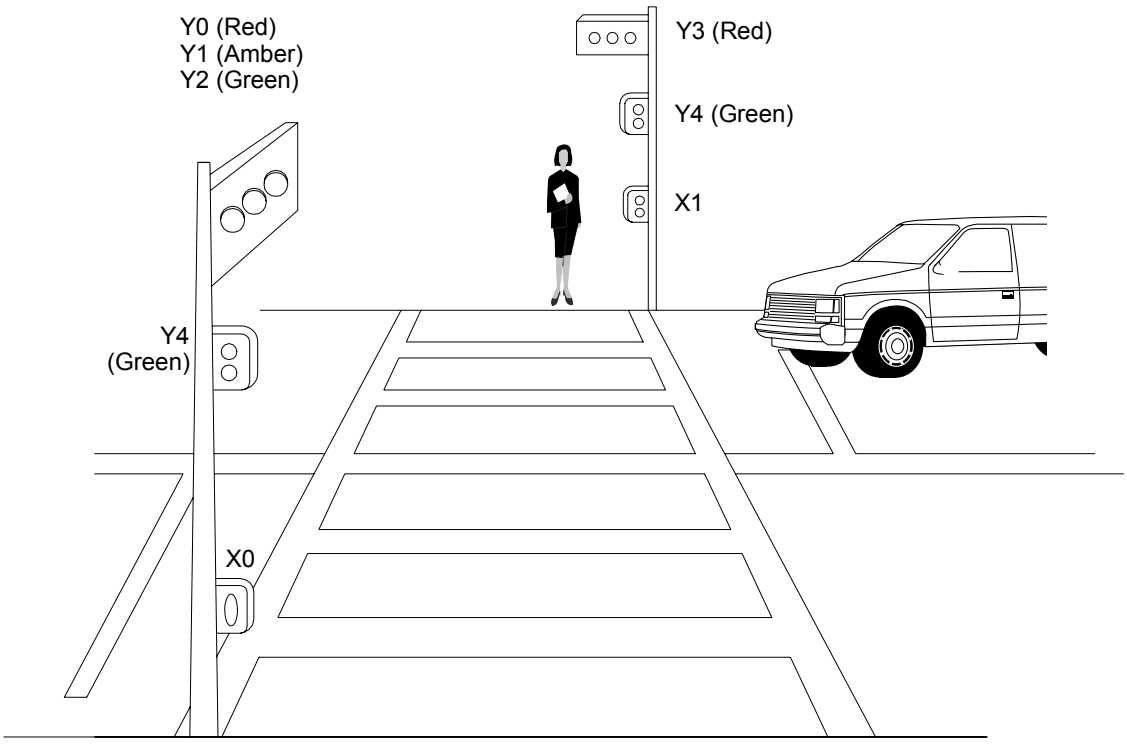
- Weighing Output: CH0（R3840）

- M1918=0

WinProladder

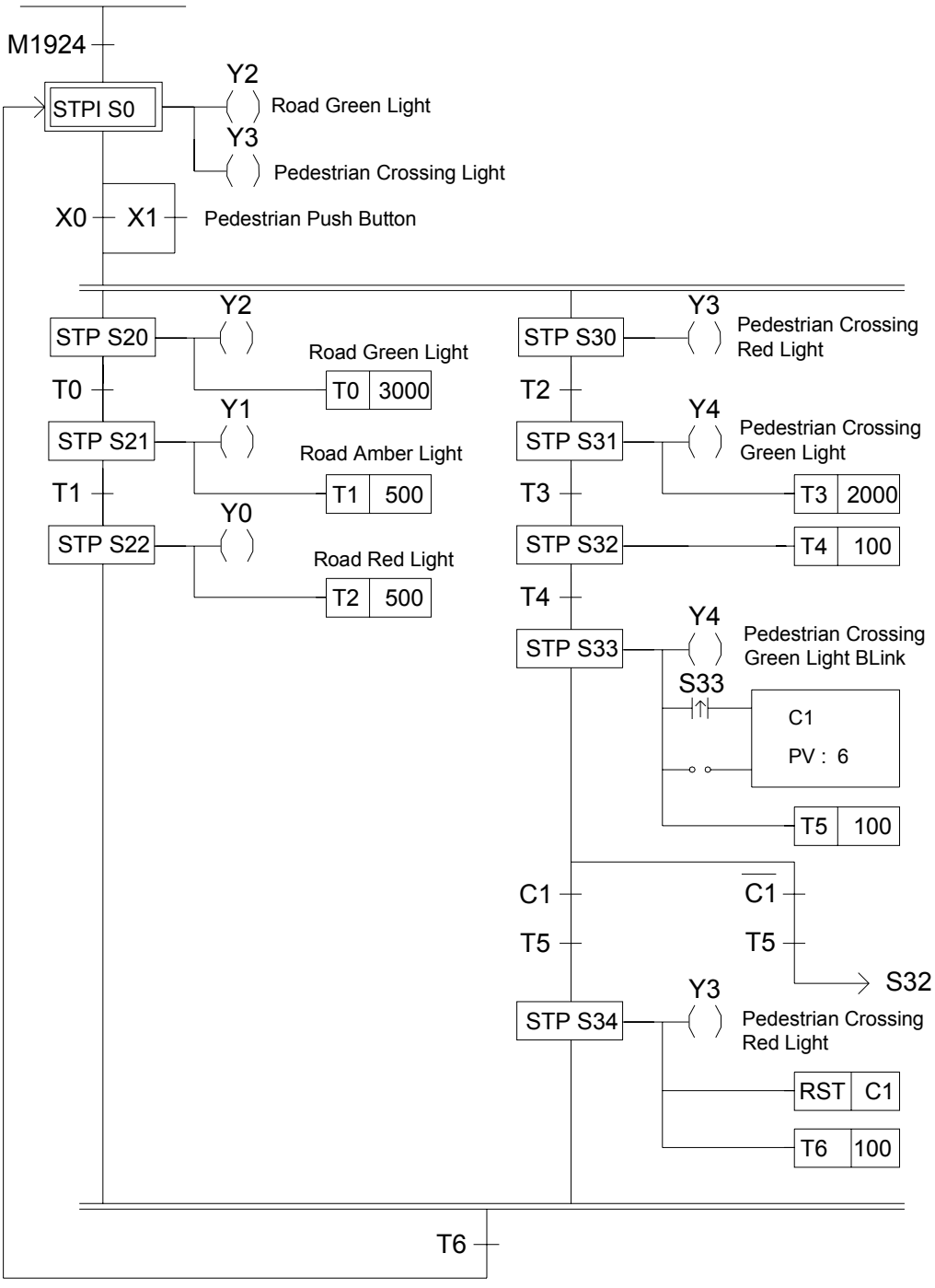


Example 3 Pedestrian Crossing Lights



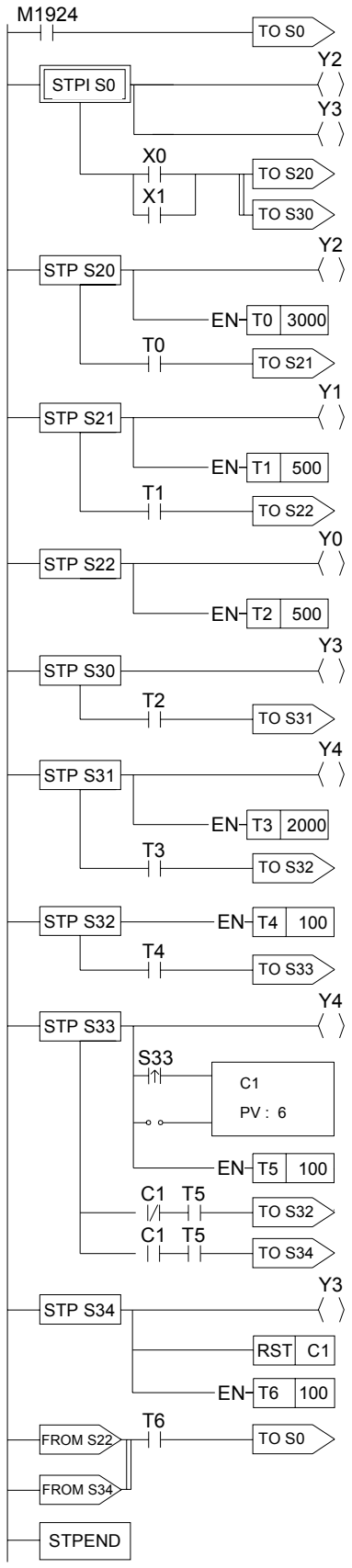
- Input Points: Pedestrian Push Button X0
  Pedestrian Push Button X1
- Output Points: Road Red Light Y0
  Road Amber light Y1
  Road Green Light Y2
  Pedestrian Crossing Red Light Y3
  Pedestrian Crossing Green Light Y4
- M1918=0

● Pedestrian Crossing Lights Control Process Diagram



● Pedestrian Crossing Lights Control Program

WinProladder



8.6 Syntax Check Error Codes for Step Instruction

The error codes for the usage of step instruction are as follows:

E51 : TO (S0-S7) must begin with ORG instruction.

E52 : TO (S20-S999) can't begin with ORG instruction.

E53 : TO instruction without matched FROM instruction.

E54 : To instruction must comes after TO, AND, OR, ANDLD or ORLD instruction.

E56 : The instructions before FROM must be AND, OR, ANDLD or ORLD

E57 : The instruction after FROM can't be a coil or a function

E58 : Coil or function must before FROM while in STEP network.

E59 : More than 8 TO# at same network.

E60 : More than 8 FROM# at same network.

E61 : TO (S0-S7) must locate at first row of the network.

E62 : A contact occupies the location for TO instruction.

E71 : Linked incomplete. (Should not occur)

E72 : Duplicated TO Sxx instruction.

E73 : Duplicated STP sxx instruction.

E74 : Duplicated FROM sxx instruction.

E76 : STP (S0~S7) without a matched STPEND or STPEND without a matched STP (S0~S7).

E77 : STP (S0 - S19) before a network is not based on ORG start only the TO (S0 - S19).

E78 : TO (S20~S999), STP (S20~S999) or FROM instructions comes before or without STP (S0~S19).

E79 : STP Sxx or FROM Sxx instructions comes before or without TO Sxx.

E80 : FROM Sxx instruction comes before or without STP Sxx.

E81 : The max. level of branches must <=16.

E82 : The max. no. of branches with same level must <=16.

E83 : Not place the step instruction with TO->STP->FROM sequence.

E84 : The definition of STP# sequence not follow the TO# sequence.

E85 : Convergence does not match the corresponding divergence.

E86 : Illegal usage of STP or FROM before convergent with TO instruction.

E87 : STP# or FROM# comes before corresponding TO#.

E88 : During this branch, STP# or FROM# comes before the corresponding TO#.

E89 : FROM# comes before corresponding TO# or STP#.

E90 : Invalid To# usage in the simultaneous branch.

E91 : Flow control function can not be used in the step ladder region.