HYUNDAI ROBOTICS

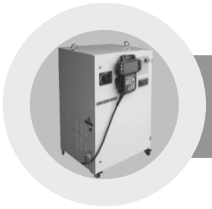
WARNING

INSTALLATION SHOULD ONLY BE PERFORMED BY QUALIFIED INSTALLATION PERSONNEL AND MUST CONFORM TO ALL NATIONAL AND LOCAL CODES

HD
HYUNDAI
ROBOTICS

Hyundai Robot

Hi5aEP230426FMEN10



# Hi5a Controller Function Manual

## Embedded PLC

HD HYUNDAI ROBOTICS

HD HYUNDAI ROBOTICS



Printed in Korea - April. 2023. 10th Edition

# Contents

# Figure Contents



# Table Contents

HD
HYUNDAI
ROBOTICS

HYUNDAI ROBOTICS

# 1 Introduction

Hi5a Controller's embedded PLC is a function that implements the common PLC's function on a Controller. As the below figure shows, by executing HRLadder on a PC or laptop that is connected to a Controller, it is enabled to compose/edit a Ladder program, to download it to a Controller, to upload the Ladder program that are being executed on a Controller, or to monitor the execution status of a Controller. Please refer to the additional function manual for HRLadder.

The Hi5a controller I/O can be connected to the upper process control panel through the filed bus or real-time Ethernet. An embedded DeviceNet master can be used to connect DeviceNet slave modules. BD580 or BD58A can be used to connect various devices using physical signals directly. The embedded PLC function controls the output signals that are connected as above with Ladder Logic.

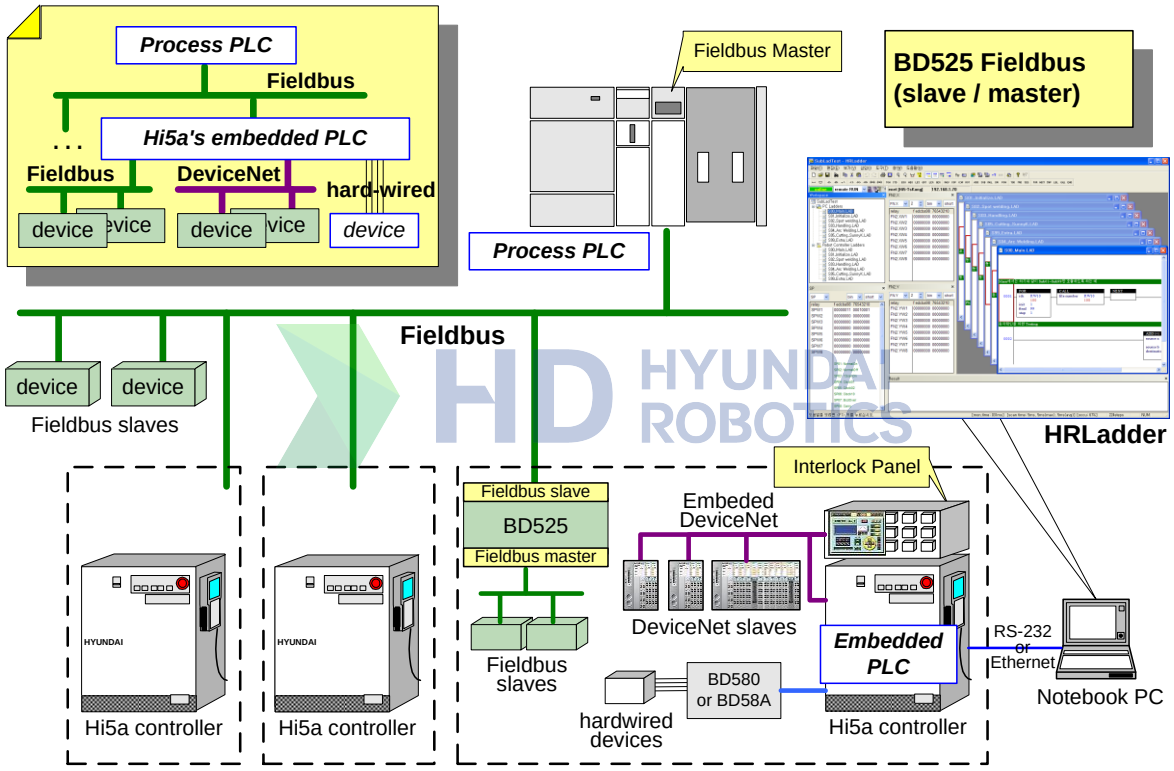



Figure 1.1 Hi5a I/O Connection

HYUNDAI ROBOTICS

# 2 Controller Set-up

## 2.1. Embedded PLC mode Set-up

Embedded PLC's operating mode can be selected as one of 〈Off, Stop, R-Stop, R-Run, Run〉 from 『[F7]: Condition setting』 → 『[F1]: App. cnd』 → 『5: Embedded PLC mode』.

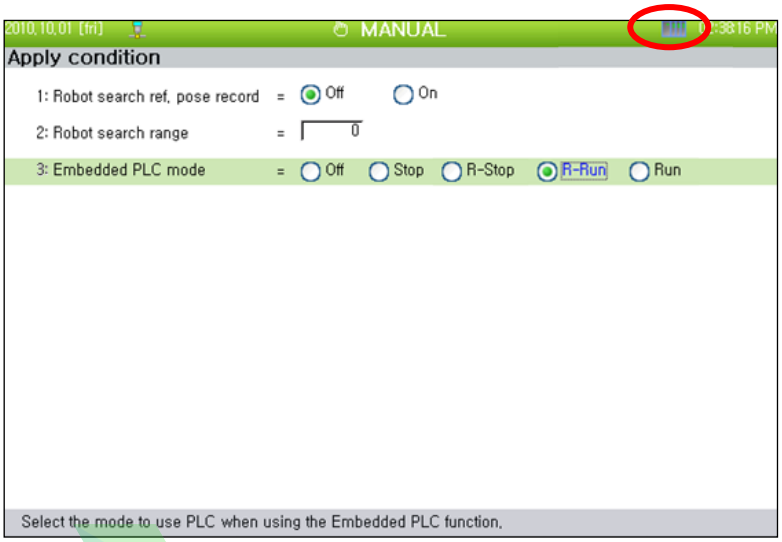



Figure 2.1 Embedded PLC Mode Set-up

Icon on the top right side of TP screen will indicates the selected mode. For example, if it is on [PLC=〈R-Run〉] or [PLC=〈Run〉] status, PLC icon as above will be displayed. If it is on [PLC=〈Off〉], the icon will be disappeared as below the figure. Also, if it is on [PLC=〈STOP〉] status, red color stop mark will be shown on the window.

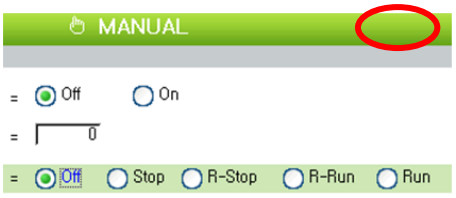



Figure 2.2 Embedded PLC Off Status

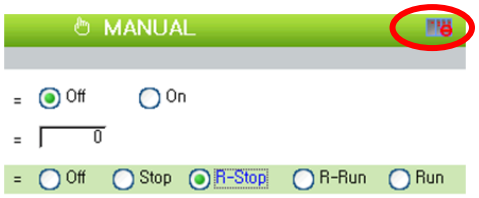



Figure 2.3 Embedded PLC Stop Status

- **Off**
  Function of embedded PLC will be disabled. In this case, DO1~DO4096 that is Digital Output of Robot's Controller will be automatically outputted as Physical Output Y1~Y4096, and the Physical Output X1~X4096 will be automatically inputted as DI1~DI4096. Yet still the properties for each D1, d0, X and Y will be applied.
  For example, if both of DO1 and Y1 is negative logic, when DO1=1 output is attempted the actual output value will be Y1=1, because 0 will be inputted as Y1 as DO1 is negative logic as well as the property of Y1.

- **R-Stop / Stop**
  This function stops the operation of embedded PLC. R-Stop shows the Remote status that can be changed on HRLadder but the operation mode cannot be changed on HRLadder while in STOP setting. When embedded PLC is Stopped, DI, Y, R relay that are PLC's output signal will automatically set to 0. However if a condition that can output the value to Y output in HRBasic or other assigned (Expect PLC) is met, such as "SP11=On", Y output will retain the previous value.

- **R-Run / Run**
  This function executes embedded PLC. R-Run indicates the Remote status that can be changed on HRLadder. Operating mode cannot be changed on HRLadder while it is on Run setting

## 2.2. Relay Condition Monitoring on Controller TP

Relay status monitoring is enabled from 『[F1]: Service』 → 『1: Monitoring』 → 『7: PLC relay data』 → 『1: PLC X Relay (External Input)』 → 『11: PLC SW Relay (System Memory)』.

## 2.3. Embedded Scan Time

"Scan Time" will be indicated on lower status bar of HRLadder. Scan Time will be automatically increase (by 5msec), if the quantity of steps increase and it cannot be processed within 5msec.

## 2.4. Embedded PLC Allocated Execution Time Control

If there is a need to accelerate the embedded PLC Scan Time for special purpose, please request it to our engineer in order to control the allocated time for PLC on Hi5a Robot. There is a menu provided on a manual mode to control the allocated time 『[F2]: System』 → 『2: Control Parameter』 → 『1: Control environment setting』 → 『7: Embedded PLC execution time』.
Embedded PLC execution time Set-up, refers to how much time for PLC process is allocated among the processes that needs to be handled by Robot Controller. The default value is 2.0/20[ms], but if too much value is assigned, the motion calculation for Robot's movement process may be disturbed. Please make sure to request our engineer to check the surplus resource of the system, and decide the time to be allocated.


Figure 2.4 Embedded PLC Execution Time Set-up

# 3

# Input / Output Diagram

Input and Output Diagram of Hi5a Robot Controller is as below.

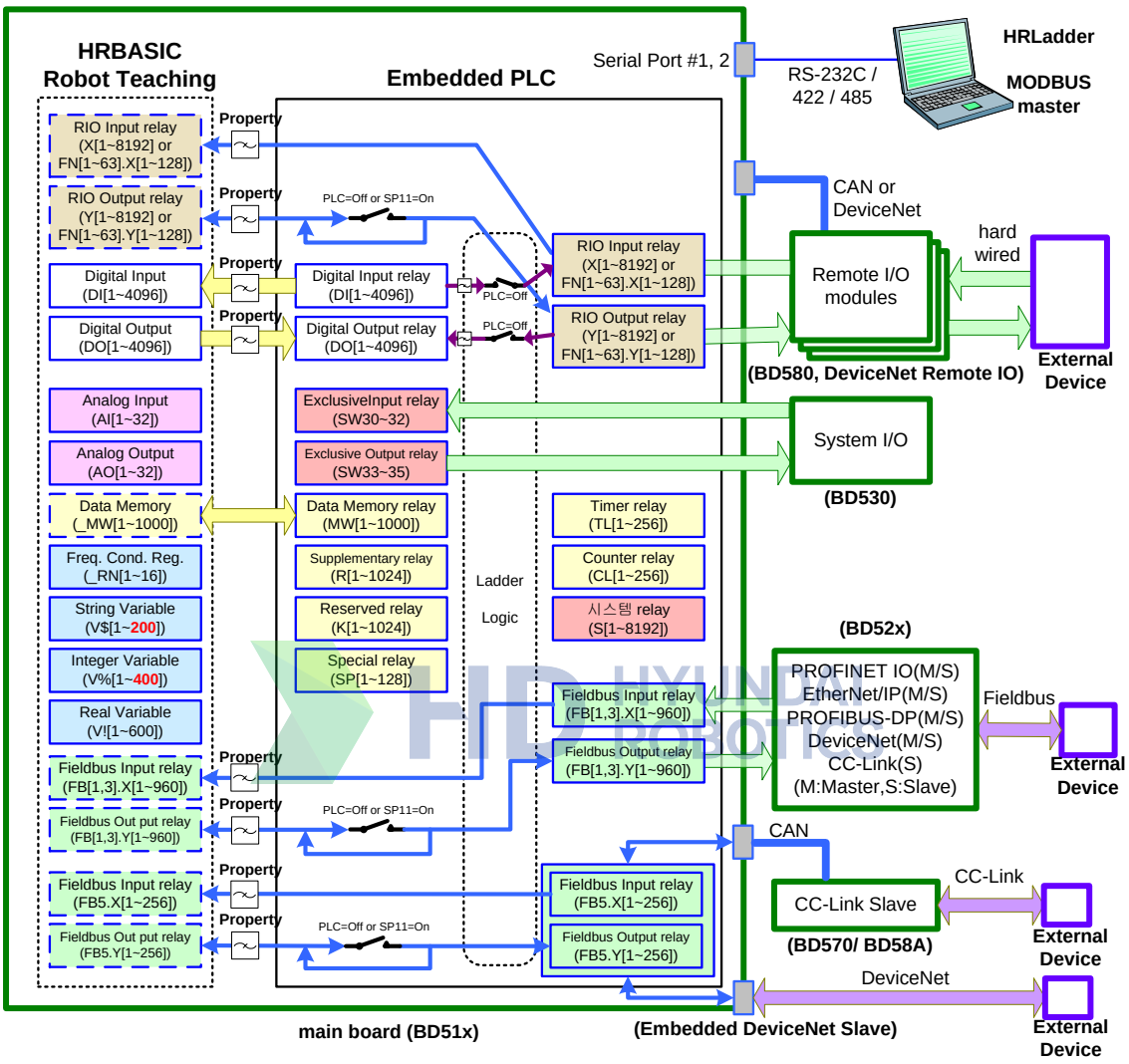



Figure 3.1 Input / Output Diagram

- Even when PLC is used, turning on SP11 will make it possible to control the relays for RIO output (Y[1~8192], FnN[1~63].Y[1~128]) or for the field bus output (FB[1,3,5].Y[1~960]) directly from the robot job program (HRBASIC).

  However, in case of input relays (X[1~8192], FN[1~63].X[1~128], FB[1,3,5].X[1~960]), it is possible to read them from the robot job program always regardless of the use of PLC.

- System Memory from the above figure is reserved for special purpose. The purpose may change in future, depending on the Controller's version.

HYUNDAI ROBOTICS

# 4 Relay Specification

## 4.1. Relay Points

Table 4-1 Relay Points

| Relay Title | Points | Relay(1bit) | Relay (Byte, Word) | (Long, FLoat) |
|---|---|---|---|---|
| Digital Input / Output Relay | Input 4096 points | **DI**1~4096 | **DIB**1~512 | **DIL**1~128 |
| | | | **DIW**1~256 | **DIF**1~128 |
| | Output 4096 points | **DO**1~4096 | **DOB**1~512 | **DOL**1~128 |
| | | | **DOW**1~256 | **DOF**1~128 |
| RIO Input / Output Relay (BD580/DeviceNet) | Input 8192 points 128 points per node (Max 64 node) | FN1~64.**X**1~128 (or **X**1~8192) | FN1~64.**XB**1~16 (or **XB**1~1024) | FN1~64.**XL**1~4 (or **XL**1~256) |
| | | | FN1~64.**XW**1~8 (or **XW**1~512) | FN1~64.**XF**1~4 (or **X**F1~256) |
| | Output 8192 points 128 points per node (Max 64 node) | FN1~64.**Y**1~128 (or **Y**1~8192) | FN1~64.**YB**1~16 (or **YB**1~1024) | FN1~64.**YL**1~4 (or **YL**1~256) |
| | | | FN1~64.**YW**1~8 (or **YW**1~512) | FN1~64.**YF**1~4 (or **YF**1~256) |
| Fieldbus Relay (PRFINET IO, EtherNet/IP, CC-Link IE, PROFIBUS-DP, DeviceNet) | Input 960 points | FB1.**X**1~960 | FB1.**XB**1~120 | FB1.**XL**1~30 |
| | | | FB1.**XW**1~60 | FB1.**XF**1~30 |
| | Output 960 points | FB1.**Y**1~960 | FB1.**YB**1~120 | FB1.**YL**1~30 |
| | | | FB1.**YW**1~60 | FB1.**YF**1~30 |
| | Input 960 points | FB3.**X**1~960 | FB3.**XB**1~120 | FB3.**XL**1~30 |
| | | | FB3.**XW**1~60 | FB3.**XF**1~30 |
| | Output 960 points | FB3.**Y**1~960 | FB3.**Y**B1~120 | FB3.**YL**1~30 |
| | | | FB3.**YW**1~60 | FB3.**YF**1~30 |
| Fieldbus Relay (CC-Link, CC-Link IE) | Input 960points | FB5.**X**1~960 | FB5.**XB**1~120 | FB5.**XL**1~30 |
| | | | FB5.**XW**1~60 | FB5.**XF**1~30 |
| | Output 960points | FB5.**Y**1~960 | FB5.**YB**1~120 | FB5.**YL**1~30 |

| Relay Title | Points | Relay(1bit) | Relay (Byte, Word) | (Long, FLoat) |
|---|---|---|---|---|
| | | | FB5.**YW**1~60 | FB5.**YF**1~30 |
| Auxiliary Relay | 1024 points | **R**1~1024 | **RB**1~128 | **RL**1~32 |
| | | | **RW**1~64 | **RF**1~32 |
| Nonvolatile Relay | 1024 points | **K**1~1024 | **KB**1~128 | **KL**1~32 |
| | | | **KW**1~64 | **KF**1~32 |
| Special Relay | 128 points | **SP**1~128 | **SPB**1~16 | **SPL**1~4 |
| | | | **SPW**1~8 | **SPF**1~4 |
| Timer Relay | 256 points | **T**1~256 | **TB**1~1024 | **TL**1~256 |
| | | | **TW**1~512 | **TF**1~256 |
| Counter Relay | 256 points | **C**1~256 | **CB**1~1024 | **CL**1~256 |
| | | | **CW**1~512 | **CF**1~256 |
| Data Memory | 16000 points | **M**1~16000 | **MB**1~2000 | **ML**1~500 |
| | | | **MW**1~1000 | **MF**1~500 |
| System Memory | 8192 points (Reserved) | **S**1~8192 (Reserved) | **SB**1~1024 | **SL**1~256 |
| | | | **SW**1~512 | **SF**1~256 |

Reference) Relay Type: Byte=8bit, Word=16bit, Long=32bit, Float=(32bit)

## 4.2. Relay Description

For the each relay bit, byte, word, long and float type can be assigned by using prefix (B/W/L/F). For bit (T/C) of Timer and Counter, it is in activation when the value is 0, or else it is in deactivation.

(1) Digital Input / Output Relay: It is a signal that can be used for HR Basic or various Input / Output allocations.

(2) RIO Input / Output Relay:
The signals are the I/O signals of the BD580 board or of the DeviceNet slave nodes connected to the embedded DeviceNet master.

(3) Fieldbus Relay:
The signals are the I/O signals of the field bus or real-time Ethernet connected through the BD521, BD522, BD525, BD570 and BD58A boards

(4) Auxiliary Relay: Auxiliary relay that is used by PLC program.

(5) Nonvolatile Relay: This relay retains the On/Off status even if the power goes Off.

(6) Special Relay: It is a relay that defined for special purposes. Refer to section 4.2.1.



(7) Timer Relay:
It is a relay for timer operation and contact point will be On when the value is 0.
(It will be reserved when power goes out).

(8) Counter Relay:
It is a relay for counter operation and contact point will be On when the value is 0.
(It will be reserved when power goes out).

(9) Data Memory:
It is used when applied commands save or read certain Data, and it also can be used when huge amount of data is being exchanged with HRBasic because HRBasic can access to it. (It will be reserved when power goes out).

(10) System Memory (Reserved):
It is used when reading or writing system values within a controller. Refer to section 4.2.2.

### 4.2.1. SP Relay

| Relay No. | Description | Others |
|---|---|---|
| SP01 | Always On relay | Controller states |
| SP02 | Always Off relay | |
| SP03 | Relay that only goes On for scan of operation start | Internal timer |
| SP04 | 0.1 second clock ( 0.05 second On → 0.05 second Off) | |
| SP05 | 0.2 second clock ( 0.1 second On → 0.1 second Off ) | |
| SP06 | 1 second clock ( 0.5 second On → 0.5 second Off ) | |
| SP07 | On when conversion of BCD operation is not valid | When TOD or FRD is executed. |
| SP08 | On when “carry” exist among the operation result | When arithmetic command is executed |
| SP09 | 2 second clock ( 1 second On → 1 second Off ) | Internal timer |
| SP10 | 4 second clock ( 2 second On → 2 second Off ) | |
| SP11 | If On, direct output to Y relay from application (HRBasic, assignment) except PLC is allowed. | |
| SP12 | On when Label is not constant | |
| SP13 | On when quantity of Label is more than 100. | |
| SP14 | On when Label is duplicated | |
| SP15 | On when Label is less than 0 or there is no Label no. to Jump | |
| SP16 | For Modbus Test purpose in Simulation status | |
| SP17 | On when Scan Time exceeds 5 second<br>(In this case, commands that listed after 5 seconds will not be executed). | |
| SP18 | On when a Sub-ladder that is summoned by Call does not exist. | |
| SP19 | 0 = Embedded DeviceNet master error/warning activated (default)<br>1 = Embedded DeviceNet master error/warning skipped | Support for the version V40.06-00 and up |

| Relay No. | Description | Others |
|---|---|---|
| SP20 | 0 = FBI communication channel error/warning activated (default)<br>1 = FBI communication channel error/warning skipped | |
| SP21 | 0 = FB3 communication channel error/warning activated (default)<br>1 = FB3 communication channel error/warning skipped | |
| SP22 | 0 = FB5 communication channel error/warning activated (default)<br>1 = FB5 communication channel error/warning skipped | |
| SP23~SP128 | Reserved | |

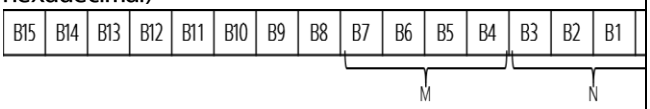
### 4.2.2. SW Relay

| No. | Description | Others |
|---|---|---|
| | | |
| SW3 | PLC Execution Mode (4: PLC OFF, 5: Program does not exist, 0:STOP, 1:R.STOP, 2:R.RUN, 3:RUN) | |
| SW4 | Quantity of extended IO board | |
| SW5 | Main SW Version 2nd + 1st | 40.05-04v→ &H05 28 |
| SW6 | I/O version +3rd of Main SW Version | 40.05-04 → &H43 04 |
| | | |
| SW10 | Scan time | |
| SW11 | Assignment time | |
| SW12 | Maximum Occupation Time | |
| SW13 | Average Occupation Time | |
| SW14 | Quantity of total steps of Ladder | |
| SW15 | CHECKSUM of Ladder Program | |
| SW17 | Occupancy rate (%) | |
| | | |
| SW20 | Current program No. of Controller | |
| SW21 | Current step No. of Controller | |
| SW22 | Current function No. of Controller | |
| SW23 | Current main program No. of Controller | |
| SW24 | FB1~FB4 active | 0th ~3rd bit |
| SW25 | Reading Hilscher status Information (1~4)<br>(1:FB1 Information, 2: FB2 Information) | |
| SW26 | LSB:Hilscher COM GlobalBits<br>MSB: reserved | (Valid only for master) |

| No. | Description | Others |
|---|---|---|
| SW27 | LSB:Error node No.<br>MSB:Error Code | (Valid only for master) |
| SW28 | Bus error counter | (Valid only for master) |
| SW29 | DeviceNet: Bus Off counter<br>Profibus-DP: Time out counter | (Valid only for master) |
| SW30 | Private Input 1 | |
| SW31 | Private Input 2 | |
| SW32 | Private Input 3 | |
| | | |
| SW33 | Private Output 1 | |
| SW34 | Private Output 2 | |
| SW35 | Private Output 3 | |
| | | |
| SW37 | LSB: For designated node No, slave_diag request for every 1 second. | |
| SW38 | Return value for salve_diag request | |
| SW39 | Station Status for the result of slave-diag request.<br>LSB: StationStatus1<br>MSB: StationStatus2 | |
| SW40~59 | Read the robot's current pose.<br>Refer to section 4.2.3 and 4.2.4. | |
| | | |
| SW60~69 | Space for designating an indirect address(Refer to 4.3 Indirect Address Designation) | -1:SW61 ~ -9:SW69 |
| | | |
| SW70 | Type of operation time<br>(0:null,1:operation time (accumulative),2: total operation time)<br>Caution) SW(n):MSW(higher), SW(n+1):LSW | |
| SW71~SW72 | Run time (10msec unit) | |
| SW73~SW74 | Cycle time (10msec unit) | |
| SW75 | Quantity of cycle | |

| No. | Description | Others |
|---|---|---|
| SW76~SW77 | Welder 1 welding time (10msec unit) | |
| SW78 | Welder 1 number of welding | |
| SW79~SW80 | Welder 2 welding time (10msec unit) | |
| SW81 | Welder 2 number of welding | |
| SW82~SW83 | Welder 3 welding time (10msec unit) | |
| SW84 | Welder 3 number of welding | |
| SW85~SW86 | Welder 4 welding time (10msec unit) | |
| SW87 | Welder 4 number of welding | |
| SW88~SW89 | Wait, DI wait time (10msec unit) | |
| SW90~SW91 | Timer wait time (10msec unit) | |
| | | |
| SW70 | Type of operation time<br>(0:null, 3: recent operation time, 4: total operation time)<br>As adding recent operating time, Swap SW to use SL.<br>SW(Odd No.):LSW(lower), SW(Even No.):MSW | |
| SL36 | Run time (10msec unit) | |
| SL37 | Cycle time (10msec unit) | |
| SL38 | Quantity of cycle | |
| SL39 | Welder 1 welding time (10msec unit) | |
| SL40 | Welder 1 number of welding | |
| SL41 | Welder 2 welding time (10msec unit) | |
| SL42 | Welder 2 number of welding | |
| SL43 | Welder 3 welding time (10msec unit) | |

| No. | Description | Others |
|---|---|---|
| SL44 | Welder 3 number of welding | |
| SL45 | Welder 4 welding time (10msec unit) | |
| SL46 | Welder 4 number of welding | |
| SL47 | Wait, DI wait Time (10msec unit) | |
| SL48 | Timer wait time (10msec unit) | Unit: 10msec |
| | | |
| SW70 | Running time type<br>(5: Running information (Arc))<br>SW swapping to allow SL to be used by adding the recent running time.<br>In other words, SW(Odd number):LSW(Lower range), SW(Even number):MSW | |
| SL36 | Measuring time(Unit: 10msec) | |
| SL37 | Cycle time (Unit: 10msec) | |
| SL38 | Cycle count | |
| SL39 | Arc welding execution time (Unit: 10msec). To be reset daily | |
| SL40 | Arc welding execution date (Daily) | |
| SW81 | Arc retry execution count | |
| SW82 | Arc overlap execution count. 1:Arc Off | |
| SW83 | Arc overlap execution count. 2:Wire Off | |
| SW84 | Arc overlap execution count. 3:Limit Over | |
| SW85 | Arc overlap execution count. 4:Gas Off | |
| SW86 | Arc Overlap execution count. 5:Coolant Off | |
| SW87 | Arc auto stick removal execution count | |
| SL45 | Arc running time in 1 cycle | |

| No. | Description | Others |
|---|---|---|
| SL46 | Tip using time | |
| SL47 | Tip using time limit | |
| SW99 | Sets whether to process it as Motor Off or not when the mode switch is switched over<br>(0: Not to process, 1: Motor Off when switchover between manual and auto occurs, 2 : Motor Off when mode switchover occurs) | |
| SW100~SW109 | Reserved in relation to Program Count | |
| SW110 | Enable the speed of each axis (SW110 = 0 × KLMN, in hexadecimal)<br>  <br><br>N (B3–B0): Resolution (0 × 1 = 10-fold,0 × 2 = 100-fold,0 × 3 = 1,000-fold)<br>M: Other options (B4:1 = absolute value, B5:0 = axial speed, 1 = motor speed) | After the setting values for bits have been determined, they should be input as converted into hexadecimals. e.g., For 1,000-fold resolution, no absolute value, and motor speed setting, input N = 0 3, M = 0 × 2 =〉 0 × 23 = “35.” |
| SW111~SW126 | Speed (16 axis) of each axis (or motor) | |
| SW130~SW133 | TP KEYPAD information | |
| SW135 | Get Applet | |
| SW136 | Set Applet | |
| | | |
| SW140 | List of the embedded DeviceNet master error slaves (0~15) | Bit0=1: Node 0 Error<br>Bit15=1: Node 15 Error |
| SW141 | List of the embedded DeviceNet master error slaves (16~31) | Bit0=1: Node 16 Error<br>Bit15=1: Node 31 Error |
| SW142 | List of the embedded DeviceNet master error slaves (32~47) | Bit0=1: Node 32 Error<br>Bit15=1: Node 47 Error |
| SW143 | List of the embedded DeviceNet master error slaves (48~63) | Bit0=1: Node 48 Error<br>Bit15=1: Node 63 Error |
| SW144 | State of the embedded DeviceNet master *1) | |

*1) State of the embedded DeviceNet master
Bit 15~9 : Reserved
Bit 8 : Initialization state (1: Initialization in progress. 0= Initialization completed)
Bit 4 : State of the master operation (0=RUN, 1=IDLE(PLC STOP))

| No. | Description | Others |
|---|---|---|
| SW145 | Node number for requesting the state information of the slaves connected to the embedded DeviceNet master | |
| SW146 | State of the slaves of the node number of SW145 *2) | |
| | | |
| SW150 | Arc Welding Info | 1=Enable(Digital Only) |
| SW151 | Arc Welding Real Current | |
| SW152 | Arc Welding Real Voltage | |
| SW153 | Arc Welder Error1 | |
| SW154 | Arc Welder Error2 | |
| SW155 | Wire Feeding Speed | |
| SW156 | Arc Welding SI 1~8 (lower byte) and SO 1~8(higher byte) | Support for the version V40.04-00 and up |
| | | |
| SW160 | DeviceNet explicit message request flag<br>– 0x0001= Request<br>DeviceNet explicit message reply flag<br>– 0x001X= Explicit message request in progress<br>– 0x0000= Explicit message handling completed<br>– 0x002X=Explicit message timeout<br>Others= Error reply | Results =〉SW166 ~ SW179 |
| SW161 | Explicit message Request: Node ID | |

Bit 3~0 : State of the DeviceNet (**0=Running**, 1=Reset(Out-of-box), 2=Init(Out-of-box), 3=Reset(Normal), 4=Init(Normal), 5=Dupplicated MAC ID Check, 6=Fault)

*2) State of the slaves
Bit 15 : **1=Node in use**, 0=Node not in use
Bit 14 : Maker (1=CREVIS, 0=Others)
Bit 13~12 : Node status (1=Standby, 2=Connection Fault, 3=Configuration Fault)
Bit 11 : **0=IO mode (State of IO data being exchanged with the master),** Not 1=IO mode
Bit 9~0 : Stage of the slaves
0x000=Online, 0x001=Offline, 0x002=IO Connection Closed,
0x004=Input Size Error, 0x008=Output Size Error, 0x010=Vendor ID mismatch,
0x020=Device type mismatch, 0x040=Production code mismatch,
0x080=CCV mismatch, 0x100=IO size is too big, 0x200=No connection

| No. | Description | Others |
|---|---|---|
| SW162 | Explicit message Request: Service | |
| SW163 | Explicit message Request: Class | |
| SW164 | Explicit message Request: Instance | |
| SW165 | Explicit message Request: Attribute | |
| SW166 | Explicit message Request/Response: Size of Data | Max. 26 |
| SW167~SW179 | Explicit message Request/Response: Service Data (SB333~SB358) | Max. 26 bytes |
| | | |
| SW180 | GUN No. to acquire the consumption rate of SPOT GUN<br>(0: invalid, 1~; GUN No.) | |
| SW181 | Moving tip consumption rate | |
| SW182 | Fixed tip consumption rate | |
| SW183 | Gun search state | |
| ~SW189 | Reserved for SPOT GUN | |
| SW190 | MON_AXIS_CTRL_OFF | 1 axis per Bit |
| SW195 | Roller hemming welding force [N] | |
| | | |
| SW200 | Accumulated distance monitoring mode for each axis<br>(For dword support, V31.11-00~, V32.01-00~) | 0:Off,<br>1:read, 2:write(word)<br>3:read, 4:write(dword) |
| SW201~SW216 | Accumulated distance monitoring for each axis (axis1~axis16)<br>If dword, SL101~SL108 (1~8 axis) | Unit: km, rad<br>Set to 0 when overflow occurs |
| | | |
| SW220 | Slot 1 code (0: Function off) | [_IDX]<br>0-base: Read value<br>30000-base: Set value<br><br>Ex1)<br>Setting 352 in SW223 |
| SW221 | Slot 1 data: Low word | |
| SW222 | Slot 1 data: High word | |

| No. | Description | Others |
|---|---|---|
| SW223~<br>SW224, 225 | Slot 2 code<br>Slot 2 data: Low and high words | reflects AO2 in SW224. |
| SW226~<br>SW227, 228 | Slot 3 code<br>Slot 3 data: Low and high words | |
| SW229~<br>SW230, 231 | Slot 4 code<br>Slot 4 data: Low and high words | |
| SW232~<br>SW233, 234 | Slot 5 code<br>Slot 5 data: Low and high words | |
| SW235~<br>SW236, 237 | Slot 6 code<br>Slot 6 data: Low and high words | |
| SW238~<br>SW239, 240 | Slot 7 code<br>Slot 7 data: Low and high words | |
| SW241~<br>SW242, 243 | Slot 8 code<br>Slot 8 data: Low and high words | |
| SW244~<br>SW245, 246 | Slot 9 code<br>Slot 9 data: Low and high words | |
| SW247~<br>SW248, 249 | Slot 10 code<br>Slot 10 data: Low and high words | |
| SW250 | (Subtask 1) generation state | |
| SW251 | (Subtask 1) current program number | |
| SW252 | (Subtask 1) current step number | |
| SW253 | (Subtask 1) current function number | |
| ~SW259 | Reserved for (Subtask 1) | |
| SW260 | (Subtask 2) generation state | |
| SW261 | (Subtask 2) current program number | |
| SW262 | (Subtask 2) current step number | |
| SW263 | (Subtask 2) current function number | |
| ~SW269 | Reserved for (Subtask 2) | |
| SW270 | (Subtask 3) generation state | |
| SW271 | (Subtask 3) current program number | |

| No. | Description | Others |
|---|---|---|
| SW272 | (Subtask 3) current step number | |
| SW273 | (Subtask 3) current function number | |
| ~SW279 | Reserved for (Subtask 3) | |
| SW280 | (Subtask 4) generation state | |
| SW281 | (Subtask 4) current program number | |
| SW282 | (Subtask 4) current step number | |
| SW283 | (Subtask 4) current function number | |
| ~SW289 | Reserved for (Subtask 4) | |
| SW290 | (Subtask 5) generation state | |
| SW291 | (Subtask 5) current program number | |
| SW292 | (Subtask 5) current step number | |
| SW293 | (Subtask 5) current function number | |
| ~SW299 | Reserved for (Subtask 5) | |
| SW300 | (Subtask 6) generation state | |
| SW301 | (Subtask 6) current program number | |
| SW302 | (Subtask 6) current step number | |
| SW303 | (Subtask 6) current function number | |
| ~SW309 | Reserved for (Subtask 6) | |
| SW310 | (Subtask 7) generation state | |
| SW311 | (Subtask 7) current program number | |
| SW312 | (Subtask 7) current step number | |

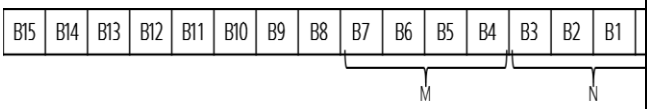
| No. | Description | Others |
|---|---|---|
| SW313 | (Subtask 7) current function number | |
| ~SW319 | Reserved for (Subtask 7) | |
| SW320 | Tool number of current robot | |
| SW321 | PRM Regenerative power: Current value (lower 16Bit) | SL161 |
| SW322 | PRM Regenerative power: Current value (higher 16Bit) | |
| SW323 | PRM Regenerative power: Accumulated value (lower 16Bit) | SL162 |
| SW324 | PRM Regenerative power: Accumulated value (higher 16Bit) | |
| SW325 | PRM Regenerative power: 5-minute average value (lower 16Bit) | SL163 |
| SW326 | PRM Regenerative power: 5-minute average value (higher 16Bit) | |
| ~SW329 | Reserved | |
| SW330 | BD525 Master: Device state *3) | |
| SW331 | BD525 Master: Count of configured slaves | 0~65535 |
| SW332 ~ SW339 | BD525 Master: List of the configured slaves | List of bits |
| SW340 | BD525 Master: Communication state (lower 8bit)*4) | |

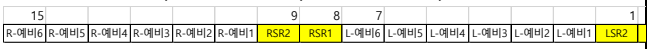
*3) BD525 Master: Device state
Bit 0: 1= Ready
Bit 1: 1=Running
Bit 2: 1=Bus On
Bit 3: 1=Configuration locked
Bit 4: 1=New configuration
Bit 5: 1=Restart required)
Bit 6: 1=Restart required Enable)
Bit 7~31: Reserved

*4) BD525 Master: Communication state
0=Unknown, 1=NOT Configured, 2=STOP, 3=IDLE, 4=OPERATE

| No. | Description | Others |
|---|---|---|
| | BD525 Master: State of the slaves (higher 8Bit)*5) | |
| SW341 | BD525 Master: Count of the running slaves | 0~65535 |
| SW342 ~ SW349 | BD525 Master: List of the running slaves | List of bits |
| SW350 | BD525 Master: Accumulated count of the running slaves | 0~65535 |
| SW351 | BD525 Master: Count of the error slaves | 0~65535 |
| SW352 ~ SW359 | BD525 Master: List of the error slaves | List of bits |
| SW360 | CC-Link IE Link status | Refer to CC-Link IE manual |
| SW361 | CC-Link IE cyclic communication status | Refer to CC-Link IE manual |
| SW366 | Low byte: iot_converter_client<br>High byte: iot_converter_server | Supported since Hi5a V40.25-00<br>ON: 1, OFF: 0 |
| SW367 | Low byte : iot_connector<br>High byte : iot_ftp | |
| SW370 ~ SW385 | Encoder temperature of each axis (J1~J16)<br>Service / Monitoring / System characteristic data / Motor, encoder data / current temperature value of each encoder<br>Cf.) Monitor motor temperature indirectly. | Supported since Hi5a V40.15-00<br>unit: ° C |
| SW388 | Temperature of the mainboard (Unit: 0.1° C) | Supported since Hi5a V40.17-00 |
| SW389 | CPU temperature of the mainboard (Unit: 0.1° C) | Supported since Hi5a V40.17-00 |
| SW390 | Controller's serial number<br>Monitor serial number in Service / System diagnosis / System version. | Supported since Hi5a V40.15-00 |
| ~SW394 | Reserved | |
| SW395 | Robot state monitoring (0: stop, 1: run, 2: wait) | Supported since Hi5a V40.15-00 |
| SW396 | Existence of path deviation<br>(1: path deviation, 0: normal path, 〈0: cannot | Supported since Hi5a V40.19-00 |

*5) BD525 Master: State of the slaves
0=UNDEFINED, 1=OK(No Fault), 2=FAILED(More than on slave error)

| No. | Description | Others |
|---|---|---|
| | determine) | |
| SW397 | Step forward/Backward Restricted<br>(Input of the number 1 is restricted, but the other numbers can be normally inputted.) | |
| ~SW399 | Reserved | |
| SW400 | Enable the speed of each axis (SW110 = 0 × KLMN, in hexadecimal)<br>  <br><br>N (B3–B0): Resolution (0 × 1 = 10-fold,0 × 2 = 100-fold,0 × 3 = 1,000-fold)<br>M: Other options (B4:1 = absolute value, B5:0 = axial speed, 1 = motor speed)<br><br>Note: This option is identical to that of SW110. If you want to view the two options concurrently, use this with SW110. | This is available in Hi5a V40.15-00 or a higher version.<br><br>Unit of speed<br>- Straight-motion axis: mm/sec<br>- Rotary axis: degree/sec<br>- Motor speed: RPM |
| SW401~ | Monitoring speed of each axis (or motor) (16 axes) | Supported since Hi5a V40.15-00 |
| SW420 | Service / Monitoring / System characteristic data / Loading rate data (1: I/Ir , 2: i/Ip, 3: continuous load rate %) | |
| SW421~ SW436 | Monitor load rate of each axis (or motor) (16 axes). | Unit: % |
| SW440 | Sets the elements to which the automatic grease injector will be applied. | Supported since Hi5a V40.18-00 |
| SW441~ SW456 | Reference distance for automatic grease injection by individual element | Supported since Hi5a V40.18-00<br>Unit of distance<br>- Linear axis: km<br>- Rotation axis: rad |
| SW461~ SW476 | Axis number assigned to each element | Supported since Hi5a V40.19-00 |
| SW480 | “Low grease” warning<br>(0 : Normal, 1~16 : Abnormal in the relevant element’s grease injector) | Supported since Hi5a V40.19-00 |
| SW481 | “Empty” warning<br>(0 : Normal, 1~16 : Abnormal in the relevant element’s grease injector) | Supported since Hi5a V40.19-00 |
| SW482 | “Overload” warning<br>(0 : normal, 1~16 : Abnormal in the relevant element’s | Supported since Hi5a V40.19-00 |

| No. | Description | Others |
|---|---|---|
| | grease injector abnormal) | |
| SW483 | FAN abnormal information monitoring<br>- Bit 0: R-FAN Board information (0:normal, 1:abnomal)<br>- Bit 8: R-FAN Board information (0:normal, 1:abnomal)<br>  | Supported<br>since Hi5a V40.22-00 |
| SW484 | LCD hand collision detection sensor data monitoring<br>- Information by bit (0: no detection, 1: detection)<br>( Bit0 : LSR1, Bit1 : LSR2, Bit8 : RSR1, Bit9 : RSR2)<br>  <br>- Monitoring data will be loaded only when the collision detection sensor is used<br>(Path : F2: System 〉 4: Application parameter 〉 29:Hand collision detection sensor) | Supported<br>since Hi5a V40.26-00 |
| SW485,<br>SW486 | SW485: Error, Warning number<br>SW486: Most significant bit(0:Error, 1:Warn)<br>Bit 1~15: Additional error information<br>(Axis number etc.) | Supported<br>since Hi5a V40.25-00 |



### 4.2.3. SW40-59 Relay – Reading the current robot pose in 16 bits

| No. | | Description | Others |
|---|---|---|---|
| | SW40 | Reading the robot pose in 16 bits (SW41~SW56)<br>0: null<br>1: Current base coordinate value (1mm or 0.1deg)<br>2: Command base coordinate value (1mm or 0.1deg)<br>3: Current axis coordinate value (0.1mm or 0.1deg)<br>4: Command axis coordinate value (0.1mm or 0.1deg)<br>5: Current axis coordinate value (1mm or 0.1deg)<br>6: Command axis coordinate value (1mm or 0.1deg) | |
| SL21 | SW41 | X coordinate value or J1-axis value | |
| | SW42 | Y coordinate value or J2-axis value | |
| SL22 | SW43 | Z coordinate value or J3-axis value | |
| | SW44 | RX coordinate value or J4-axis value | |
| SL23 | SW45 | RY coordinate value or J5-axis value | |

| | | | |
|---|---|---|---|
| | **SW46** | RZ coordinate value or J6-axis value | |
| **SL24** | **SW47 ~52** | T1~T6-axis value | |

### 4.2.4. SW40-59 Relay – Reading the current robot pose in 32 bits

| No. | | Description | Others |
|---|---|---|---|
| | SW40 | Reading the robot pose in 32 bits (SL21~SL29)<br>0:null<br>…<br>10: Current base coordinate value + additional axis 3<br>20: Command base coordinate value + additional axis 3<br>1m: Current axis coordinate value (m axis ~ m+8 axis)<br>2m: Command axis coordinate value (m axis ~ m+8axis) | (0.001mm or 0.001deg)<br>Supported since Hi5a V40.25-00 |
| SL21 | SW41 | X coordinate value<br>Jm+0 axis value | |
| | SW42 | | |
| SL22 | SW43 | Y coordinate value<br>Jm+1 axis value | |
| | SW44 | | |
| SL23 | SW45 | Z coordinate value<br>Jm+2 axis value | |
| | SW46 | | |
| SL24 | SW47 | RX coordinate value<br>Jm+3 axis value | |
| | SW48 | | |
| SL25 | SW49 | RY coordinate value<br>Jm+4 axis value | |
| | SW50 | | |
| SL26 | SW51 | RZ coordinate value<br>Jm+5 axis value | |
| | SW52 | | |
| SL27 | SW53 | T1 coordinate value<br>Jm+6 axis value | |
| | SW54 | | |
| SL28 | SW55 | T2 coordinate value<br>Jm+7 axis value | |
| | SW56 | | |
| SL29 | SW57 | T3 coordinate value | |

| | | | |
|---|---|---|---|
| | SW58 | Jm+8 axis value | |

### 4.2.5. SW220~249 Relay – System Variables

- Code

| | |
|---|---|
| 10 | SPDRATE: Speed rate (%) |
| 20 | JOGSPD: Max. speed during forward/backward step (mm/sec) |

- Reading/Writing

| | |
|---|---|
| Reading | Continuously updated. |
| Writing | Continuously updated. |

- Example (when using slot 1)

| | | |
|---|---|---|
| SW220 | **10(dec)** | Reading the SPDRATE value: 85% |
| SW221 | 85(dec) | |
| SW222 | - | |
| SW220 | **30020(dec)** | |
| SW221 | 100 | JOGSPD=100 mm/sec |
| SW222 | | |

### 4.2.6. SW220~249 Relay –Mainboard storage space

- Code

| | |
|---|---|
| 210 | The no. of low word free blocks and high word total blocks<br>Reading only |

### 4.2.7. SW220~249 Relay – Analog I/O

- Code

| | |
|---|---|
| 301~316 | AI1–AI16 (unit: 0.01 V) values: Low word<br>Reading only |
| 351~366 | AO1–AO16 (unit: 0.01 V) values: Low word |

- Example (when using slot 1)

| | | |
|---|---|---|
| SW220 | **302(dec)** | |
| SW221 | -925(dec) | Reading the AI2 value: -9.25 V |
| SW222 | - | |
| SW220 | **30353(dec)** | |
| SW221 | 380(dec) | AO3=3.80V |
| SW222 | - | |

### 4.2.8. SW220~249 Relay – Date / Time

- Code

| | |
|---|---|
| 401 | Date : High wordYYYY(dec), Low wordMMDD(dec) |
| 402 | Time (24 hr system) : High word 00HH(dec), Low word MMSS(dec) |

- Reading / Writing

| | |
|---|---|
| Reading | The current date/time values in the Hi5a controller are continuously updated as slot data (at the cycle of once per second). |
| Writing | The moment the code value changes from one value to “30401” or “30402,” the date/time values inputted as slot data will be set in the Hi5a controller. |

- Example (when using slot 1)

| | | |
|---|---|---|
| SW220 | **401(dec)** | Index: Reading the date (0-based) |
| SW221 | 0523(dec) | May 23, 2012 |
| SW222 | 2012(dec) | |
| SW223 | **402(dec)** | Index: Reading the time (0-based) |
| SW224 | 3650(dec) | 15:36:50 |
| SW225 | 15(dec) | |
| SW220 | **30401(dec)** | Index: Setting the date (30000-based) |
| SW221 | 0523(dec) | May 23, 2012 |
| SW222 | 2012(dec) | |
| SW223 | **30402(dec)** | Index: Setting the time (30000-based) |
| SW224 | 3650(dec) | 15:36:50 |
| SW225 | 15(dec) | |

## 4.2.9. SW220~249 Relay – GE Variables

- Code

| | |
|---|---|
| 1101~1128 | GE1 ~ GE128<br>The GE variable index is code-1000. |

- Reading / Writing

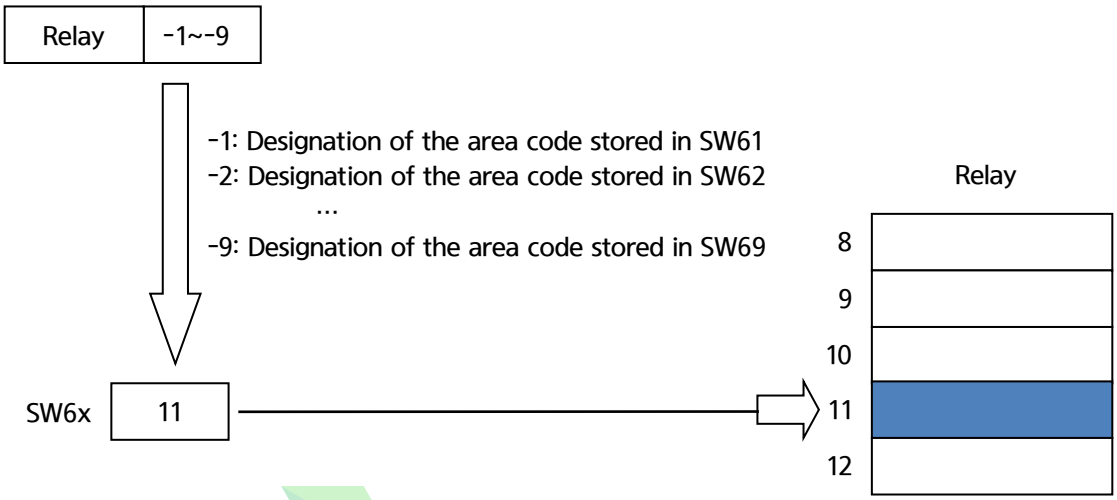
| | |
|---|---|
| Reading | Continuously updated.<br>Reading 4 bytes of GE[index] - GE[index+3] as low word and high words |
| Writing | When the system variable index value changes (value should be 31001–31128), the value inputted in the data low word will go through type conversion and then set in the GE variable. |

- Example (when using slot 1 and slot 2)

| | | |
|---|---|---|
| SW220 | **31001(dec)** | Index: Setting GE (30000-based) |
| SW221 | 2(hex) | Setting 2 to GE1 |
| SW222 | - | |
| SW223 | **31004(dec)** | |
| SW224 | 37(hex) | Setting 55 to GE4 |
| SW225 | - | |
| SW220 | **1001(dec)** | Index: Reading GE1 – (0-based) |
| SW221 | 0002(hex) | |
| SW222 | 3700(hex) | |
| SW223 | **1002(dec)** | Index: Reading GE2 – (0-based) |
| SW224 | 0000(hex) | |
| SW225 | 0037(hex) | |

## 4.3. Relay Indirect Address Designation

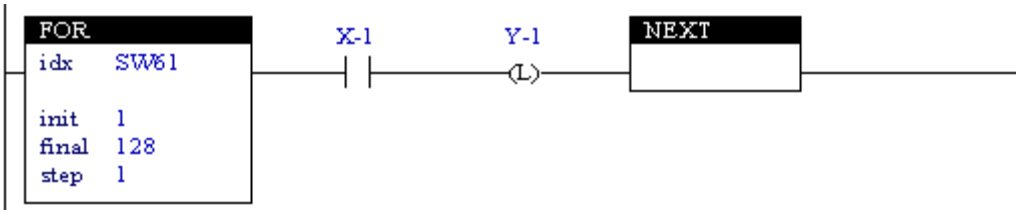
SW61 ~ SW69 are the system memories for the indirect address specification. Regardless of the type of relays, when the relay address is specified as "-1" ~ "-9", the values stored in SW61 ~ SW69 will be specified as the address.




For example, in case of SW61=12, SW65=34, SW69=56, the relays will be designated as below.

- MW-1➔ MW12, MW-5➔ MW34, MW-9➔MW56
- FN-1.X3 ➔ FN12.X3, FN-5.X3 ➔ FN34.X3, FN-9.X3 ➔ FN56.X3
- X-1 ➔ X12, X-5 ➔ X34, X-9 ➔ X56
- Y-1 ➔ Y12, Y-5 ➔ Y34, Y-9 ➔ Y56

The PLC example shown below is the case in which the operation of inputting the Y1 ~ 128 signals that match the X1 ~ X128 input signals is prepared by using the FOR/NEXT statement and the indirect address designation method.



## 4.4. Timer & Counter Relay

(1) Both timer and counter relay only support down-counting.

- Timer base can be set with 10msec unit by a user.
- Because the value of Timer is internally processed as 32bit value, so it can be count up to 2,147,483,647 [msec] (approximately 597hours).

(2) The values of Timer / Counter have meanings as below.

| Value of Timer & Counter | Description |
|---|---|
| 0 | Contact Point On (=counting completed) |
| -1 | Contact Point Off |
| Others | Contact Point Off ; timing & counting (processing) |

(3) If Rung, which is connected to Timer / Counter relay, is deactivated.

- The value of TON: TL(Timer) becomes -1.
- The value of CTD: CL(Counter) is maintained.

(4) While Rung, which is connected to Timer / Counter relay, is activated.

- **TON**
  If a value of TL is less than 0, initial value of TL is saved as “Timer base x preset x 10”. If the value is greater than 0, it will be decreased by 5 for every 5msec.

- **CTD**
  If a value of CL is less than 0, initial value of CL will be preset value.
  If the value of CL is greater than 0, it will be decreased by 1 for each time when deactivation turns to activation.

HD
HYUNDAI
ROBOTICS

HYUNDAI ROBOTICS

# 5 Command Specification

## 5.1. Command list

Table 5-1 Command table

| Command | | | | Description |
|---|---|---|---|---|
| No. | Mnemonic | Name | Symbol | |
| | RUNG | Rung | ├──┤ | Represents rung. |
| | BST | Branch Start | ┬── | Start of branch. |
| | BND | Branch End | ──┬ | End of branch. |
| | NXB | Nested Branch | └, ├ | Nested branch. |
| Logic examine command | | | | Check result,<br>if true Rung activation/(false: deactivation) |
| 1 | XIC | Examine if Closed | -| |- | Examine if contact point is closed (A contact point) |
| 2 | XIO | Examine if Open | -|/|- | Examine if contact point is open (B contact point) |
| 3 | INV | Inverting | -//- | Inverting of Rung result. |
| 4 | EQU | Equal | -▤- | Examine if equal(=). |
| 5 | NEQ | Not Equal | -▤- | Examine if not equal. |
| 6 | LES | Less Than | -▤- | Examine if less than (〈). |
| 7 | GRT | Greater Than | -▤- | Examine if greater than (〉). |
| 8 | LEQ | Less Than or Equal | -▤- | Examine if less than or equal (〈=). |
| 9 | GEQ | Greater Than or Equal | -▤- | Examine if greater than or equal (〉=). |
| Output command | | | | |
| 10 | OTE | Output Energize | -( )- | Output Rung status<br>(Activation: ON/Deactivation: OFF). |
| 11 | OTL | Output Latch | -(L)- | If Rung is activated, output is ON (high). |
| 12 | OTU | Output Unlatch | -(U)- | If Rung is activated, output is OFF (low). |

| Command | | | | Description |
|---|---|---|---|---|
| No. | Mnemonic | Name | Symbol | |
| 13 | OSR | One Shot Rising | -(OSR)- | If Rung is activated, output is ON only for one scan. |
| 14 | RES | Reset | -(RES)- | If Rung is activated, reset the timer or counter. |
| Timer and counter command | | | | |
| 15 | TON | Time On Delay | -▤- | Operate timer while Rung is in activation. |
| 16 | CTD | Count Down | -▤- | Down-count Rung's activation (deactivation->activation). |
| Arithmetic command | | | | |
| 17 | ADD | Add | -▤- | If Rung is activated, (+) operation. |
| 18 | SUB | Subtract | -▤- | If Rung is activated, (-) operation. |
| 19 | MUL | Multiply | -▤- | If Rung is activated, (x) operation. |
| 20 | DIV | Divide | -▤- | If Rung is activated, (/) operation. |
| 21 | POW | Power | -▤- | If Rung is activated, (^:power) operation. |
| Data conversion command | | | | |
| 22 | TOD | Convert int. to BCD | -▤- | If Rung is activated, convert into BCD. |
| 23 | FRD | Convert form BCD to int. | -▤- | If Rung is activated, convert into Integer. |
| 24 | SEG | 7'Segment | -▤- | If Rung is activated, convert into 7' Segment value. |
| Move and copy command | | | | |
| 25 | MOV | Move | -▤- | If Rung is activated, copy one data. |
| 26 | COP | Copy data | -▤- | If Rung is activated, copy multiple data. |
| 27 | CCOP | Conditional Copy data | -▤- | Copy multiple data according to the Rung status. |
| Applied command | | | | |

| Command | | | | Description |
|---|---|---|---|---|
| No. | Mnemonic | Name | Symbol | |
| 28 | ROT | Rotating Output | -▤- | If Rung is activated, output in rotation. |
| Block control command | | | | |
| 29 | FOR | For | -▤- | If Rung is activated, repeat execution until NEXT. |
| 30 | NEXT | Next | -▤- | If within repeat numbers, JUMP to FOR clause. |
| 31 | LBL | Label | -▤- | Designate location to Jump for Jump command. |
| 32 | JMP | Jump | -▤- | If Rung is activated, Jump to Label location.<br>(If Label〈0, Jump to –n number of Next). |
| 33 | CALL | Call | -▤- | If Rung is activated, call Sub-ladder. |
| 34 | END | End | -▤- | If Rung is activated, Sub-ladder end. |

## 5.2. Available Operands

Table 5-2 Available Operands

| inst | Relay Type arg | Input X,DO n | Input X,DO (B,W,L,F)m | Output Y,DI,R,K,SP n | Output Y,DI,R,K,SP (B,W,L,F)m | Timer T n | Timer T (B,W,L,F)m | Counter C n | Counter C (B,W,L,F)m | Memory M,S n | Memory M,S (B,W,L,F)m | Const (32bit) (L/F) |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| XIC | | | x | | x | | x | | x | | x | x |
| XIO | | | x | | x | | x | | x | | x | x |
| EQU | sA | x | | x | | x | | x | | x | | |
| | sB | x | | x | | x | | x | | x | | |
| LES | sA | x | | x | | x | | x | | x | | |
| | sB | x | | x | | x | | x | | x | | |
| GRT | sA | x | | x | | x | | x | | x | | |
| | sB | x | | x | | x | | x | | x | | |
| LEQ | sA | x | | x | | x | | x | | x | | |
| | sB | x | | x | | x | | x | | x | | |
| GEQ | sA | x | | x | | x | | x | | x | | |
| | sB | x | | x | | x | | x | | x | | |
| OTF | | x | x | | x | x | x | x | x | x | x | x |
| OTL | | x | x | | x | x | x | x | x | x | x | x |
| OUT | | x | x | | x | x | x | x | x | x | x | x |
| OSR | | x | x | | x | x | x | x | x | x | x | x |
| RES | | x | x | x | x | | x | | x | x | x | x |
| TON | tm | x | x | x | x | | x | x | x | x | x | x |
| | bs | x | | x | | x | | x | | x | | |
| | pst | x | | x | | x | | x | | x | | |
| CTD | cnt | x | x | x | x | x | x | x | x | x | x | x |
| | pst | x | | x | | x | | x | | x | | |
| ADD | sA | x | | x | | x | | x | | x | | |
| | sB | x | | x | | x | | x | | x | | |
| | dst | x | x | x | | x | | x | | x | | x |
| SUB | sA | x | | x | | x | | x | | x | | |
| | sB | x | | x | | x | | x | | x | | |
| | dst | x | x | x | | x | | x | | x | | x |
| MUL | sA | x | | x | | x | | x | | x | | |
| | sB | x | | x | | x | | x | | x | | |
| | dst | x | x | x | | x | | x | | x | | x |

| inst | Relay Type / arg | Input X,DO n | Input X,DO (B,W,L,F)m | Output Y,DI,R,K,SP n | Output Y,DI,R,K,SP (B,W,L,F)m | Timer T n | Timer T (B,W,L,F)m | Counter C n | Counter C (B,W,L,F)m | Memory M,S n | Memory M,S (B,W,L,F)m | Const (32bit) (L/F) |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| DIV | sA | x | | x | | x | | x | | x | | |
| | sB | x | | x | | x | | x | | x | | |
| | dst | x | x | x | | x | | x | | x | | x |
| POW | sA | x | | x | | x | | x | | x | | |
| | sB | x | | x | | x | | x | | x | | |
| | dst | x | x | x | | x | | x | | x | | x |
| TOD | src | x | u | x | u | x | u | x | u | x | u | u |
| | dst | x | x | x | u | x | x | x | x | x | u | x |
| FRD | src | x | u | x | u | x | u | x | u | x | u | u |
| | dst | x | x | x | u | x | x | x | x | x | u | x |
| SEG | src | x | u | x | u | x | u | x | u | x | u | u |
| | dst | x | x | x | u | x | x | x | x | x | u | x |
| MOV | src | x | | x | | x | | x | | x | | |
| | dst | x | x | x | | x | | x | | x | | x |
| COP | src | | | | | x | | x | | | | |
| | dst | x | x | | | x | | x | | | | x |
| | len | x | | x | | x | | x | | x | | |
| CCOP | sA | | | | | x | | x | | | | |
| | sB | | | | | x | | x | | | | |
| | dst | x | x | | | x | | x | | | | x |
| | len | x | | x | | x | | x | | x | | |
| ROT | st | x | | x | | x | | x | | x | | x |
| | cnt | x | | x | x | x | | x | | x | | |
| | tm | x | x | x | x | | x | x | x | x | x | x |
| | rep | x | | x | | x | | x | | x | | |
| | out | x | x | x | | x | x | x | x | x | | x |
| | rst | | x | | x | | x | | x | x | x | x |
| | tmp | x | x | x | | x | x | x | x | x | | x |
| FOR | idx | x | x | x | | x | x | x | x | x | | x |
| | init | x | | x | | x | | x | | x | | |
| | final | x | | x | | x | | x | | x | | |
| | step | x | | x | | x | | x | | x | | |
| LRI | label | x | x | x | x | x | x | x | x | x | x | |

| inst | Relay / Type / arg | Input X,DO n | Input X,DO (B,W,L,F)m | Output Y,DI,R,K,SP n | Output Y,DI,R,K,SP (B,W,L,F)m | Timer T n | Timer T (B,W,L,F)m | Counter C n | Counter C (B,W,L,F)m | Memory M,S n | Memory M,S (B,W,L,F)m | Const (32bit) (L/F) |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| JMP | label | x | | x | | x | | x | | x | | |
| CALL | S file | x | | x | | x | | x | | x | | |

(1) 'X' from the table represents unavailable.

(2) Relay's n represents the number of bit. (B,W,L,F) are prefix that distinguish the types of Byte(8bit), Word(16bit), Long(32bit), Float, and m means the data number for each types. (Example, X1, XB2, XW2, XL2, XF2).

(3) 'u' index from the table represents unsigned values.

(4) So, the data size can be selected by selecting relay type (B, W, L, F) when the command's factor is being entered.

(5) For COP (copy) and CCOP (conditional copy) commands, relay types of "src" and "dst" has to match unless the "src" is constant.

(6) Opposing value of Byte (B), Word (W), or Long word (L) can be entered for bit value. For example, identical bit address such as X34=XB5/2=XW3/2=XL2/2 can be entered in various types.

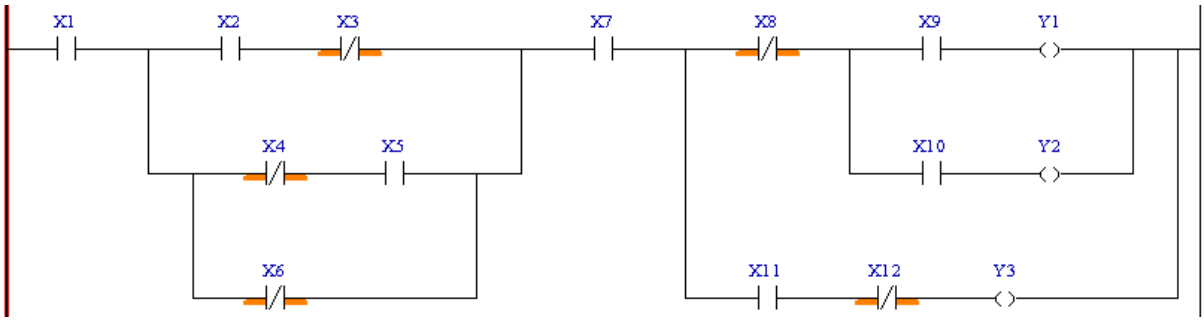
(7) Constant
Data size of constant is 32bit, and data types are binary, decimal and hexadecimal. Starts with &B refers to binary, &H refers to hexadecimal, and numbers without prefix refers to decimal. For example, 125 can be entered such as &H7D or &B01111101.

(8) All relay's (B, W, L, F) without 'u' in the table will be considered as signed value.
For example, be cautious when moving B (8bit) value to W (16bit) in MOV command as it will expand the signed bit. If move RB (-1=&HFF) to RW, it will become 0xFFFF (-1), not &H00FF (255).
However TOD and FRD command that are related to BCD data and SEG command that convert to 7' Segment data will be considered as unsigned value.

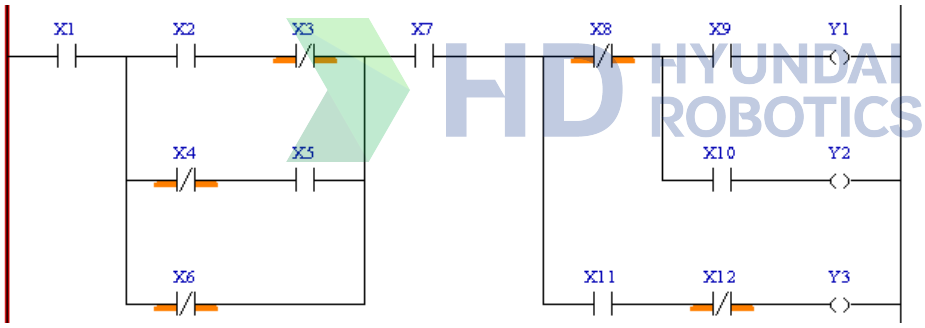
## 5.3. Relationship between Ladder Diagram and mnemonic

File that created with Ladder will be automatically converted to mnemonic and be transferred to the Controller.

(1) Ladder diagram




Or, by clicking the branch gap of HRLadder [ ], display as below.




(2) Mnemonic

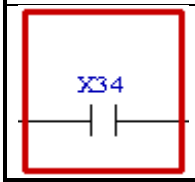
SOR XIC X1 BST XIC X2 XIO X3 NXB BST XIO X4 XIC X5 NXB XIO X6 BND BND
XIC X7 BST XIO X8 BST XIC X9 OTE Y1 NXB XIC X10 OTE Y2 BND
NXB XIC X11 XIO X12 OTE Y3 BND

HYUNDAI ROBOTICS

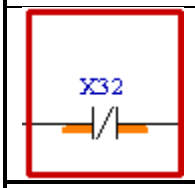
# 6

# Command Description

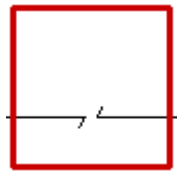
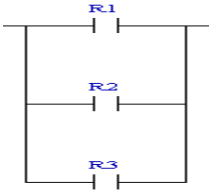
## 6.1. XIC(Examine if Closed): Examine Close

| Symbol | Combination of Relay that can be used |
|---|---|
|  | relay / type / inst / arg.: Input (X, DO; n (B,W,L,F) m) · Output (Y,DI,R,K,SP; n (B,W,L,F) m) · Timer (T; n (B,W,L,F) m) · Counter (C; n (B,W,L,F) m) · Memory (M, S; n (B,W,L,F) m) · const (32bit) (L/F)<br>XIC: Input x · Output x · Timer x · Counter x · Memory x · const x |
| Description | If a value of factor is 1, Rung is activated and if the value is 0, Rung is deactivated. |
| Usage example | XIC(X2); XIC(DO2); XIC(Y2); XIC(DI2); XIC(R2); XIC(K2); XIC(SP1); XIC(T2); XIC(C2); |
| Input method | X18 can be entered as XB3/2, XW2/2, or XL1/2. |

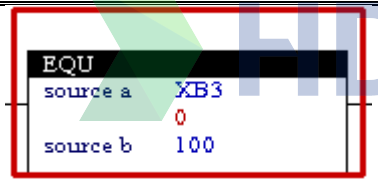
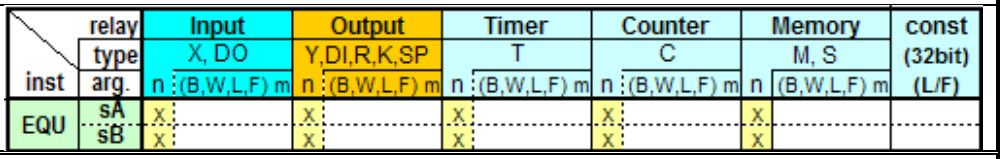
## 6.2. XIO(Examine if Open): Examine Open

| Symbol | Combination of Relay that can be used |
|---|---|
|  | relay / type / inst / arg.: Input (X, DO; n (B,W,L,F) m) · Output (Y,DI,R,K,SP; n (B,W,L,F) m) · Timer (T; n (B,W,L,F) m) · Counter (C; n (B,W,L,F) m) · Memory (M, S; n (B,W,L,F) m) · const (32bit) (L/F)<br>XIO: Input x · Output x · Timer x · Counter x · Memory x · const x |
| Description | If a value of factor is 0, Rung is activated and if the value is 1, Rung is deactivated. |
| Usage example | XIO(X2); XIO(DO2); XIO(Y2); XIO(DI2); XIO(R2); XIO(K2); XIO(SP1); XIO(T2); XIO(C2); |
| Input method | X18 can be entered as XB3/2, XW1/2 or XL1/2. |

## 6.3. INV(Inverting): Inverting

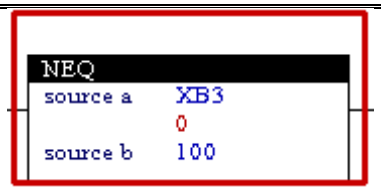
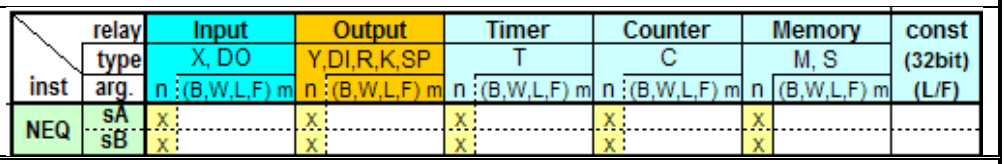
| Symbol | Description |
|---|---|
|  | It reverses the result until Rung (activate ←→ deactivate).<br>This process implements /(AxB)=/A+/B or /(A+B)=/Ax/B according to the law of De Morgan, so it can simply form AND logic without branches instead of using OR logic with many branches.<br>Therefore, as (R1+R2+R3) = /(/R1x/R2x/R3), both of two below logics will have same results.<br>  <br>  |

## 6.4. EQU(Equal): Examine Equal

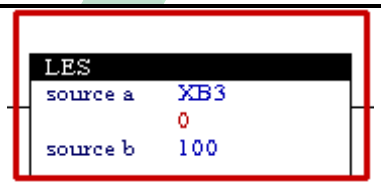
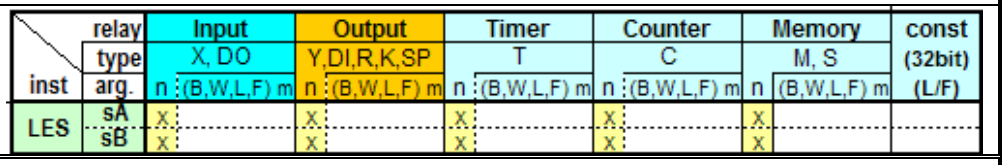
| | |
|---|---|
| Symbol |  |
| Relay combination |  |
| Factor | sA: 1st relay (or Number) to be compared<br>sB: 2nd relay (or Number) to be compared |
| Description | Compare two values and if they are equal, Rung will be activated.(Contact point activated) |
| Usage example | EQU(XB3,100): If the value on XB3 is 100, Rung will be activated.<br>EQU(XB3,RW4): If the value on both XB3 and RW4 are same, Rung will be activated. |

| inst | relay type / arg. | Input X, DO n (B,W,L,F) m | Output Y,DI,R,K,SP n (B,W,L,F) m | Timer T n (B,W,L,F) m | Counter C n (B,W,L,F) m | Memory M, S n (B,W,L,F) m | const (32bit) (L/F) |
|---|---|---|---|---|---|---|---|
| EQU | sA | x | x | x | x | x | |
| | sB | x | x | x | x | x | |

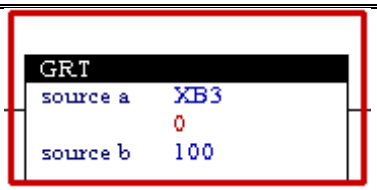
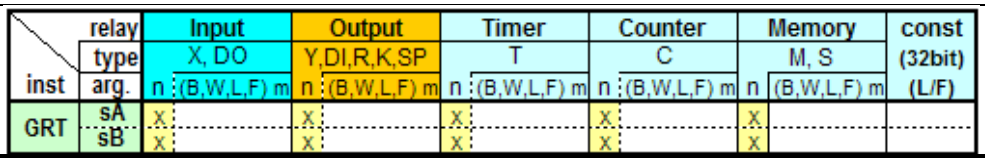
## 6.5. NEQ(Not Equal): Examine Not Equal

| | |
|---|---|
| Symbol |  |
| Relay combination |  |
| Factor | sA: 1st relay (or Number) to be compared<br>sB: 2nd relay (or Number) to be compared |
| Description | Compare two values and if they are not equal, Rung will be activated.<br>(Contact point activated) |
| Usage example | NEQ(XB3,100): if the value of XB is not 100, Rung will be activated.<br>NEQ(XB3,RW4): if the value of XB3 is not equal to the value of RW4, Rung will be activated. |

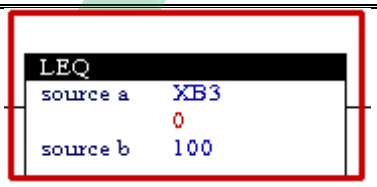
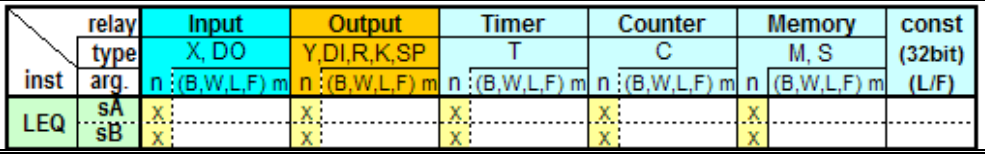
## 6.6. LES(Less Than): Examine Less Than

| | |
|---|---|
| Symbol |  |
| Relay combination |  |
| Factor | sA: 1st relay (or Number) to be compared<br>sB: 2nd relay (or Number) to be compared |
| Description | If sA is less than sB, Rung will be activated. (Contact point activated) |
| Usage example | LES(XB3,100): If the value of XB3 is less than 100, Rung will be activated.<br>LES(XB3,RW4): If the value of XB3 is less than the value of RW4, Rung will be activated. |

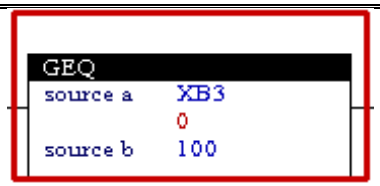
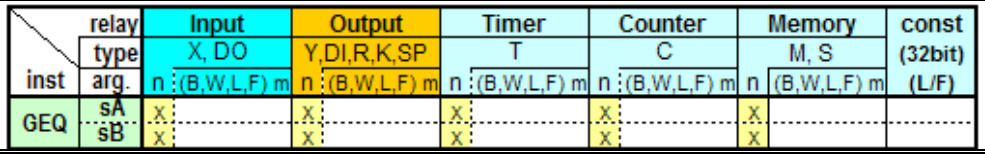
## 6.7. GRT(Greater Than): Examine Less Than

| | |
|---|---|
| Symbol |  |
| Relay combination |  |
| Factor | sA: 1st relay (or Number) to be compared<br>sB: 2nd relay (or Number) to be compared |
| Description | If sA is greater than sB, Rung will be activated. (Contact point activated) |
| Usage example | GRT(XB3,100): If the value of XB3 is greater than 100, Rung will be activated.<br>GRT(XB3,RW4): If the value of XB3 is greater than the value of RW4, Rung will be activated |

## 6.8. LEQ(Less Than or Equal): Examine Less Than or Equal

| | |
|---|---|
| Symbol |  |
| Relay combination |  |
| Factor | sA: 1st relay (or Number) to be compared<br>sB: 2nd relay (or Number) to be compared |
| Description | If sA is less than sB or equal, Rung will be activated. (Contact point activated) |
| Usage example | LEQ(XB3,100): If the value of XB3 is less than 100 or equal, Rung will be activated.<br>LEQ(XB3,RW4): If the value of XB3 is less than the value of RW4 or equal, Rung will be activated. |

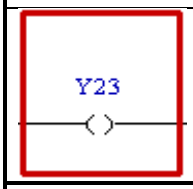
## 6.9. GEQ(Greater Than or Equal): Examine Greater Than or Equal

| | |
|---|---|
| Symbol |  |
| Relay combination |  |
| Factor | sA: 1st relay (or Number) to be compared<br>sB: 2nd relay (or Number) to be compared |
| Description | If sA is greater than sB or equal, Rung will be activated. (Contact point activated) |
| Usage example | GEQ(XB3,100): If the value of XB3 is greater than 100 or equal, Rung will be activated.<br>GEQ(XB3,RW4): If the value of XB3 is greater than the value of RW4 or equal, Rung will be activated. |

| relay | Input | Output | Timer | Counter | Memory | const |
|---|---|---|---|---|---|---|
| type | X, DO | Y,DI,R,K,SP | T | C | M, S | (32bit) |
| inst / arg. | n (B,W,L,F) m | n (B,W,L,F) m | n (B,W,L,F) m | n (B,W,L,F) m | n (B,W,L,F) m | (L/F) |
| GEQ sA | x | x | x | x | x | |
| GEQ sB | x | x | x | x | x | |

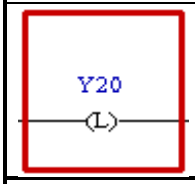


## 6.10. OTE(Output Energize): General Output

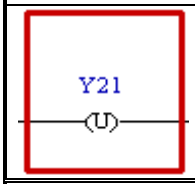
| | |
|---|---|
| Symbol | Combination of Relay that can be used |
| |  |
| Description | Output signal will be depending on the Rung status.<br>If Rung is activated, output signal will be On(high), if deactivated, output signal will be Off(low). |
| Factor Example | Y2; DI2; R2; K2; SP1; |
| Input Method | Y18 can be entered as YB3/2, YW2/2, or YL1/2. |

| relay | Input | Output | Timer | Counter | Memory | const |
|---|---|---|---|---|---|---|
| type | X, DO | Y,DI,R,K,SP | T | C | M, S | (32bit) |
| inst / arg. | n (B,W,L,F) m | n (B,W,L,F) m | n (B,W,L,F) m | n (B,W,L,F) m | n (B,W,L,F) m | (L/F) |
| OTE | x | x | x | x | x | x |

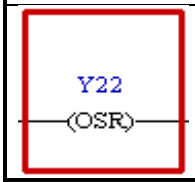
## 6.11. OTL(Output Latch): Latch Output

| Symbol | Combination of Relay that can be used. |
|---|---|
|  | relay / type / inst / arg. — Input: X, DO; Output: Y,DI,R,K,SP; Timer: T; Counter: C; Memory: M, S; const (32bit) (L/F)<br>n (B,W,L,F) m — n (B,W,L,F) m — n (B,W,L,F) m — n (B,W,L,F) m — n (B,W,L,F) m<br>OTL: x x — x — x x — x x — x x — x x — x — x |
| Description | If Rung is activated, output signal will be On(high).<br>However, if Rung is deactivated, output will not change. |
| Factor Example | Y2; DI2; R2; K2; SP1; |
| Input Method | Y18 can be entered as YB3/2, YW2/2, or YL1/2. |

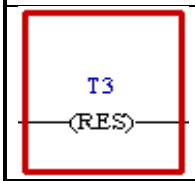
## 6.12. OTU(Output Unlatch): Unlatch Output

| Symbol | Combination of Relay that can be used |
|---|---|
|  | relay / type / inst / arg. — Input: X, DO; Output: Y,DI,R,K,SP; Timer: T; Counter: C; Memory: M, S; const (32bit) (L/F)<br>n (B,W,L,F) m — n (B,W,L,F) m — n (B,W,L,F) m — n (B,W,L,F) m — n (B,W,L,F) m<br>OTU: x x — x — x x — x x — x x — x x — x — x |
| Description | If Rung is activated, output signal will be Off(low).<br>However, if Rung is deactivated, output will not change. |
| Usage Example | Y2; DI2; R2; K2; SP1; |
| Input Method | Y18 can be entered as YB3/2, YW2/2, or YL1/2. |

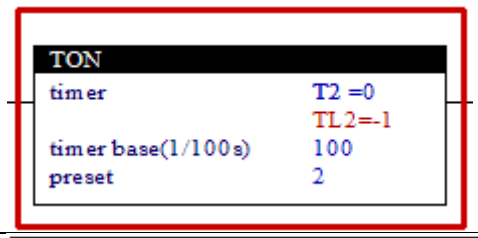
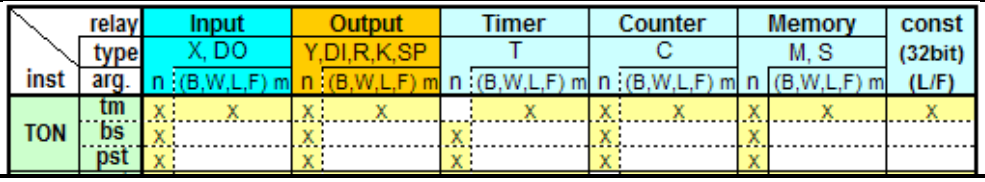
## 6.13. OSR(One Shot Rising): One Shot Output

| Symbol | Combination of Relay that can be used |
|---|---|
|  | relay / type / inst / arg. — Input: X, DO; Output: Y,DI,R,K,SP; Timer: T; Counter: C; Memory: M, S; const (32bit) (L/F)<br>n (B,W,L,F) m — n (B,W,L,F) m — n (B,W,L,F) m — n (B,W,L,F) m — n (B,W,L,F) m<br>OSR: x x — x x — x x — x x — x x — x |
| Description | If Rung is activated, output will be On(high) only for one scan.<br>Therefore, when Rung becomes from deactivated to activated, relevant relay will be On, only for one scan. |
| Factor Example | Y2; DI2; R2; K2; SP1; |
| Input Method | Y18 can be entered as YB3/2, YW2/2, or YL1/2. |

## 6.14. RES(Reset): Reset

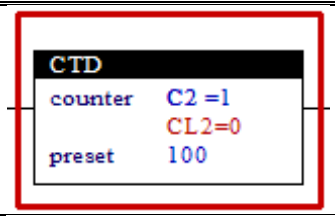
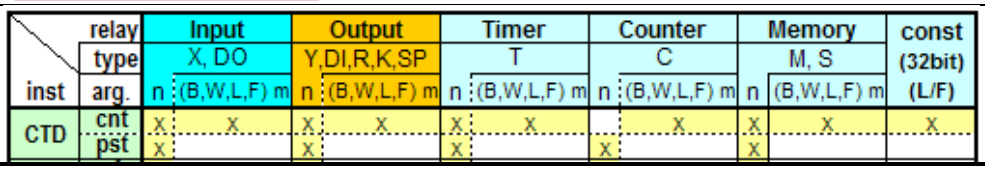
| Symbol | Combination of Relay that can be used |
|---|---|
|  | relay / type / inst / arg. — Input: X, DO; Output: Y,DI,R,K,SP; Timer: T; Counter: C; Memory: M, S; const (32bit) (L/F)<br>n (B,W,L,F) m — n (B,W,L,F) m — n (B,W,L,F) m — n (B,W,L,F) m — n (B,W,L,F) m<br>RES: x x — x x — x x — x x — x x — x |
| Description | If Rung is activated, output will be cleared(-1) |
| Factor Example | T2; C2; |

## 6.15. TON(Time On Delay): Timer

| | |
|---|---|
| Symbol |  |
| Relay combination |  |
| Factor | tm: Designate the Timer Relay No.<br>bs: Set Timer unit as relay or number (e.g. 100=1second unit, 10=0.1 second unit)<br>pst: Set Preset value (Time [msec]=bs*pst*10) as relay or number |
| Description | It will calculate the time that Rung is in active to set the time(bs x pst x 10) [msec]. After the time, corresponding timer relay will be On(high).<br>However, if Rung is deactivated, it will set to Clear(-1) immediately<br>Reference) The value of TL is 1msec unit. |
| Usage example | TON(T2,100,5): Set T2 as (100x5x10[msec]=5[sec]) timer<br>TON(T3,RW3,RW4): Set T3 as the value of (RW3)x(RW4)x10 with timer |

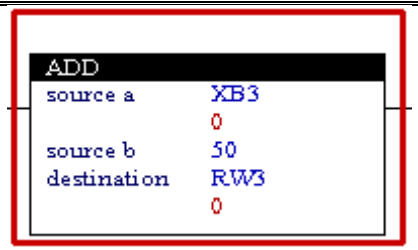
| relay | Input | | Output | | Timer | | Counter | | Memory | | const |
|---|---|---|---|---|---|---|---|---|---|---|---|
| type | X, DO | | Y,DI,R,K,SP | | T | | C | | M, S | | (32bit) |
| inst / arg. | n | (B,W,L,F) m | n | (B,W,L,F) m | n | (B,W,L,F) m | n | (B,W,L,F) m | n | (B,W,L,F) m | (L/F) |
| TON tm | x | x | x | x | | x | x | x | x | x | x |
| TON bs | x | | x | | x | | x | | x | | |
| TON pst | x | | x | | x | | x | | x | | |

## 6.16. CTD(Count Down): Counter

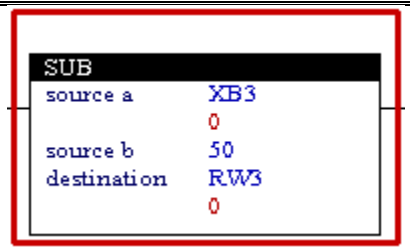
| | |
|---|---|
| Symbol |  |
| Relay combination |  |
| Factor | cnt: Designate the Counter Relay No.<br>pst: Set Preset value as relay or number |
| Description | Down-count the rise of Rung (deactivate to activated)<br>If the corresponding value of CL is 0, counter will be On(high) and not count anymore.<br>When Rung is activated, if the corresponding value of CL is negative number, the value of pst will be saved in CL.<br>Reference) Even if Rung is deactivated, CL will not be Cleared(-1). |
| Usage example | CTD(C2,100): Set C2 to Down-counter (100)<br>CTD(C3,RW3): Set C3 to Down-counter as the value of (RW3) |

| relay | Input | | Output | | Timer | | Counter | | Memory | | const |
|---|---|---|---|---|---|---|---|---|---|---|---|
| type | X, DO | | Y,DI,R,K,SP | | T | | C | | M, S | | (32bit) |
| inst / arg. | n | (B,W,L,F) m | n | (B,W,L,F) m | n | (B,W,L,F) m | n | (B,W,L,F) m | n | (B,W,L,F) m | (L/F) |
| CTD cnt | x | x | x | x | x | x | | x | x | x | x |
| CTD pst | x | | x | | x | | x | | x | | |

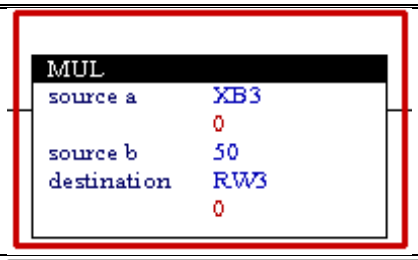
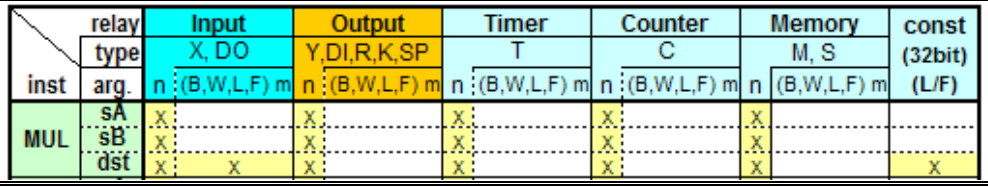
## 6.17. ADD(Add): Add

| | |
|---|---|
| Symbol |  |
| Relay combination | relay / type / inst / arg. — Input: X, DO; Output: Y,DI,R,K,SP; Timer: T; Counter: C; Memory: M, S; const (32bit) (L/F)<br>n (B,W,L,F) m<br>ADD sA: X (Input), X (Output), X (Timer), X (Counter), X (Memory)<br>ADD sB: X (Input), X (Output), X (Timer), X (Counter), X (Memory)<br>ADD dst: X (Input), X (B,W,L,F), X (Output), X (Timer), X (Counter), X (Memory), X (const) |
| Factor | sA: 1st relay (or Number) for the operation<br>sB: 2nd relay (or Number) for the operation<br>dst: Relay that will save the result |
| Description | If Rung is activated, add the value sA and sB, then save it to dst.<br>If overflow occurs on operation result, set SP8=1. |
| Usage example | ADD(XB3,RB2,RW3): (RW3)=(XB3)+(RB2)<br>ADD(XB3,RB2,RB2): (RB2)=(XB3)+(RB2)<br>ADD(XB3,50,RW3): (RW3)=(XB3)+50 |

## 6.18. SUB(Subtract): Subtract

| | |
|---|---|
| Symbol |  |
| Relay combination | relay / type / inst / arg. — Input: X, DO; Output: Y,DI,R,K,SP; Timer: T; Counter: C; Memory: M, S; const (32bit) (L/F)<br>n (B,W,L,F) m<br>SUB sA: X (Input), X (Output), X (Timer), X (Counter), X (Memory)<br>SUB sB: X (Input), X (Output), X (Timer), X (Counter), X (Memory)<br>SUB dst: X (Input), X (B,W,L,F), X (Output), X (Timer), X (Counter), X (Memory), X (const) |
| Factor | sA: 1st relay (or Number) for the operation<br>sB: 2nd relay (or Number) for the operation<br>dst: Relay that will save the result |
| Description | If Rung is activated, subtract the value sB from the value of sA, then save it to dst. |
| Usage example | SUB(XB3,RB2,RW3): (RW3)=(XB3)-(RB2)<br>SUB(XB3,RB2,RB2): (RB2)=(XB3)-(RB2)<br>SUB(XB3,50,RW3): (RW3)=(XB3)-50 |

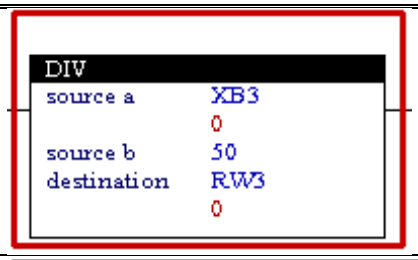
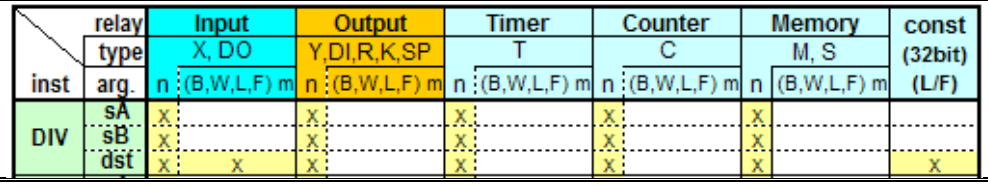
## 6.19. MUL(Multiply): Multiply

| | |
|---|---|
| Symbol |  |
| Relay combination |  |
| Factor | sA: 1st relay (or Number) for the operation<br>sB: 2nd relay (or Number) for the operation<br>dst: Relay that will save the result |
| Description | If Rung is activated, multiply the value of sA with the value of sB, then save it to dst.<br>If overflow occurs on operation result, set SP8=1. |
| Usage example | MUL(XB3,RB2,RW3): (RW3)=(XB3)x(RB2)<br>MUL(XB3,RB2,RB2): (RB2)=(XB3)x(RB2)<br>MUL(XB3,50,RW3): (RW3)=(XB3)x50 |

| relay | Input | | Output | | Timer | | Counter | | Memory | | const |
|---|---|---|---|---|---|---|---|---|---|---|---|
| type | X, DO | | Y,DI,R,K,SP | | T | | C | | M, S | | (32bit) |
| inst / arg. | n | (B,W,L,F) m | n | (B,W,L,F) m | n | (B,W,L,F) m | n | (B,W,L,F) m | n | (B,W,L,F) m | (L/F) |
| MUL sA | X | | X | | X | | X | | X | | |
| MUL sB | X | | X | | X | | X | | X | | |
| MUL dst | X | X | X | | X | | X | | X | | X |

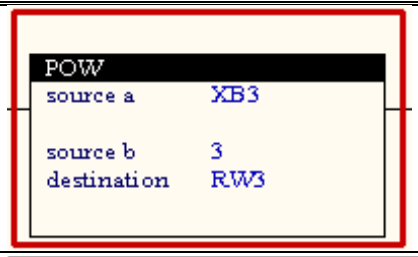
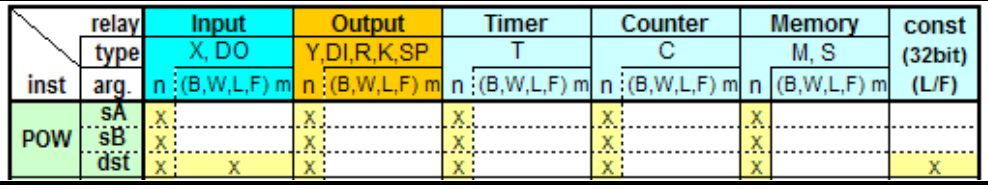


## 6.20. DIV(Divide): Divide

| | |
|---|---|
| Symbol |  |
| Relay combination |  |
| Factor | sA: 1st relay (or Number) for the operation.<br>sB: 2nd relay (or Number) for the operation.<br>dst: Relay that will save the result |
| Description | If Rung is activated, divide the value of sA with the value of sB, then save it to dst.<br>If sB is 0, or overflow occurs on operation result, set SP8=1. |
| Usage example | DIV(XB3,RB2,RW3): (RW3)=(XB3)/(RB2)<br>DIV(XB3,RB2,RB2): (RB2)=(XB3)/(RB2)<br>DIV(XB3,50,RW3): (RW3)=(XB3)/50 |

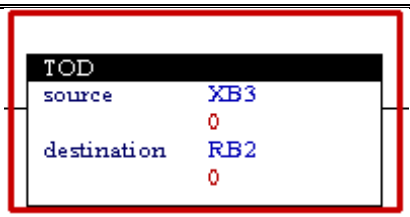
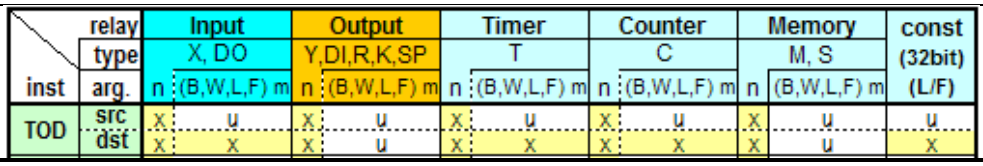
| relay | Input | | Output | | Timer | | Counter | | Memory | | const |
|---|---|---|---|---|---|---|---|---|---|---|---|
| type | X, DO | | Y,DI,R,K,SP | | T | | C | | M, S | | (32bit) |
| inst / arg. | n | (B,W,L,F) m | n | (B,W,L,F) m | n | (B,W,L,F) m | n | (B,W,L,F) m | n | (B,W,L,F) m | (L/F) |
| DIV sA | X | | X | | X | | X | | X | | |
| DIV sB | X | | X | | X | | X | | X | | |
| DIV dst | X | X | X | | X | | X | | X | | X |

## 6.21. POW(Power): Power

| | |
|---|---|
| Symbol |  |
| Relay combination |  |
| Factor | sA: 1st relay (or Number) for the operation.<br>sB: 2nd relay (or Number) for the operation.<br>dst: Relay that will save the result |
| Description | If Rung is activated, power the value of sA with the value of sB, then save it to dst.<br>If overflow occurs on operation result, set SP8=1. |
| Usage example | POW(2,3,RW3): (RW3)=2^3=2*2*2=8<br>POW(XB3,RB2,RW3): (RW3)=(XB3)^(RB2)<br>POW(XB3,RB2,RB2): (RB2)=(XB3)^(RB2)<br>POW(XB3,3,RW3): (RW3)=(XB3)^3 |

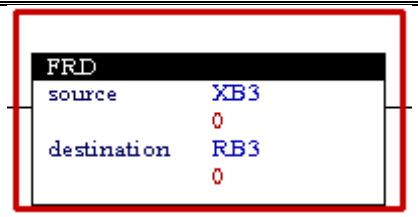
| inst | relay / type / arg. | Input X, DO n | Input X, DO (B,W,L,F) m | Output Y,DI,R,K,SP n | Output Y,DI,R,K,SP (B,W,L,F) m | Timer T n | Timer T (B,W,L,F) m | Counter C n | Counter C (B,W,L,F) m | Memory M, S n | Memory M, S (B,W,L,F) m | const (32bit) (L/F) |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| POW | sA | X | | X | | X | | X | | X | | |
| | sB | X | | X | | X | | X | | X | | |
| | dst | X | X | X | | X | | X | | X | | X |

## 6.22. TOD(Convert to BCD): Convert to BCD

| | |
|---|---|
| Symbol |  |
| Relay combination |  |
| Factor | src: Source relay (or Number)<br>dst: Destination relay |
| Description | If Rung is activated, convert the value of src to BCD and save it to dst.<br>This command is convenient when using a device that indicates value on 7' Segment with BCD type.<br>If dst is byte(B) type, it will convert into 2 digits and if dst is word(W) type, it will convert into 4 digits.<br>If the value of Src is greater than the converting digits, set SP7=1. |
| Usage example | TOD(XB3,RB2): Convert the value of XB3 to BCD and save it to RB2.<br>(Reference, BCD(Binary Coded Decimal) refers to the number that can have a value of 0~9 range with 4bit code value. For BCD, A~F that can be represented with 4 bit (among 0~F) is not used.<br>If (XB3)=&H7B(123), save &H23(34) into RB2, and set SP7=1 as &H7B(123)〉&H63(99). |

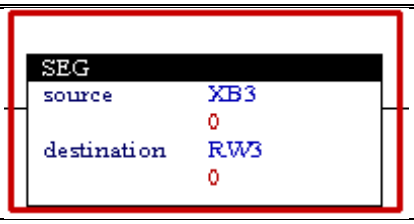
| inst | relay type / arg. | Input X, DO n | Input X, DO (B,W,L,F) m | Output Y,DI,R,K,SP n | Output Y,DI,R,K,SP (B,W,L,F) m | Timer T n | Timer T (B,W,L,F) m | Counter C n | Counter C (B,W,L,F) m | Memory M, S n | Memory M, S (B,W,L,F) m | const (32bit) (L/F) |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| TOD | src | X | u | X | u | X | u | X | u | X | u | u |
| | dst | X | X | X | u | X | X | X | X | X | u | X |

## 6.23. FRD(Convert form BCD to Integer): Convert to Integer

<table>
<tr><td>Symbol</td><td> 
</td></tr>
<tr><td>Relay combination</td><td>
<table>
<tr><td rowspan="3">inst</td><td>relay</td><td colspan="2">Input</td><td colspan="2">Output</td><td colspan="2">Timer</td><td colspan="2">Counter</td><td colspan="2">Memory</td><td rowspan="2">const<br>(32bit)<br>(L/F)</td></tr>
<tr><td>type</td><td colspan="2">X, DO</td><td colspan="2">Y,DI,R,K,SP</td><td colspan="2">T</td><td colspan="2">C</td><td colspan="2">M, S</td></tr>
<tr><td>arg.</td><td>n</td><td>(B,W,L,F) m</td><td>n</td><td>(B,W,L,F) m</td><td>n</td><td>(B,W,L,F) m</td><td>n</td><td>(B,W,L,F) m</td><td>n</td><td>(B,W,L,F) m</td></tr>
<tr><td rowspan="2">FRD</td><td>src</td><td>x</td><td>u</td><td>x</td><td>u</td><td>x</td><td>u</td><td>x</td><td>u</td><td>x</td><td>u</td><td>u</td></tr>
<tr><td>dst</td><td>x</td><td>x</td><td>x</td><td>u</td><td>x</td><td>x</td><td>x</td><td>x</td><td>x</td><td>u</td><td>x</td></tr>
</table>
</td></tr>
<tr><td>Factor</td><td>src: Source relay (or Number)<br>dst: Destination relay</td></tr>
<tr><td>Description</td><td>If Rung is activated, convert the value of src's BCD to integer and save it to dst.<br>This command is convenient when accepting a value of cam switch that is BCD type output as input.<br>If the value of src is not BCD, set SP7=1.<br>Also, if src is word(W) type and dst is byte(B) type, the maximum value of converted src will be &H9999. Conversion result of integer will be 9999(&H270F) which will exceed the range of byte &Hff and overflow will occur. In this case, set SP7=1.</td></tr>
<tr><td>Usage example</td><td>FRD(XB3,RB2): Convert the value(BCD) of XB3 into integer and save it to RB2.<br>If (XB3)=&H23(35), save 23=&H17 to RB2.</td></tr>
</table>

## 6.24. SEG(7'Segment): Convert to 7' Segment value

**Symbol**




**Relay combination**

| inst | relay type | arg. | Input X, DO n | Input X, DO (B,W,L,F) m | Output Y,DI,R,K,SP n | Output Y,DI,R,K,SP (B,W,L,F) m | Timer T n | Timer T (B,W,L,F) m | Counter C n | Counter C (B,W,L,F) m | Memory M, S n | Memory M, S (B,W,L,F) m | const (32bit) (L/F) |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| SEG | src | | X | u | X | u | X | u | X | u | X | u | u |
| | dst | | X | X | X | u | X | X | X | X | X | u | X |

**Factor**

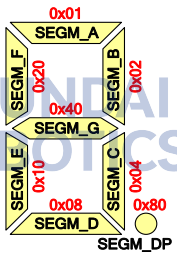
src: Source relay (or Number)
dst: Destination relay

**Description**

If Rung is activated, convert the value of src into 7' Segment value (8bit) and save it to dst.
If dst is word(W) type, save 2 of 7' Segment value (8bit) to dst.

**7' Segment Data**

```
#define SEGM_A   0x01
#define SEGM_B   0x02
#define SEGM_C   0x04
#define SEGM_D   0x08
#define SEGM_E   0x10
#define SEGM_F   0x20
#define SEGM_G   0x40
#define SEGM_DP 0x80
```

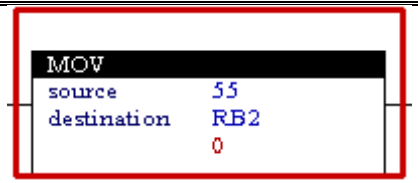
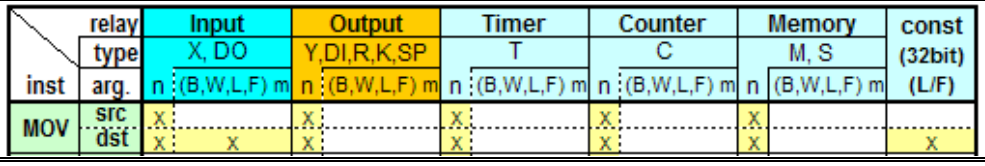



```
#define SEGD_0   (SEGM_A|SEGM_B|SEGM_C|SEGM_D|SEGM_E|SEGM_F)
#define SEGD_1   (SEGM_B|SEGM_C)
#define SEGD_2   (SEGM_A|SEGM_B|SEGM_G|SEGM_E|SEGM_D)
#define SEGD_3   (SEGM_A|SEGM_B|SEGM_C|SEGM_D|SEGM_G)
#define SEGD_4   (SEGM_B|SEGM_C|SEGM_F|SEGM_G)
#define SEGD_5   (SEGM_A|SEGM_C|SEGM_D|SEGM_F|SEGM_G)
#define SEGD_6   (SEGM_A|SEGM_C|SEGM_D|SEGM_E|SEGM_F|SEGM_G)
#define SEGD_7   (SEGM_A|SEGM_B|SEGM_C)
#define SEGD_8   (SEGM_A|SEGM_B|SEGM_C|SEGM_D|SEGM_E|SEGM_F|SEGM_G)
#define SEGD_9   (SEGM_A|SEGM_B|SEGM_C|SEGM_F|SEGM_G)
#define SEGD_A   (SEGM_A|SEGM_B|SEGM_C|SEGM_E|SEGM_F|SEGM_G)
#define SEGD_B   (SEGM_C|SEGM_D|SEGM_E|SEGM_F|SEGM_G)
#define SEGD_C   (SEGM_A|SEGM_D|SEGM_E|SEGM_F)
#define SEGD_D   (SEGM_B|SEGM_C|SEGM_D|SEGM_E|SEGM_G)
#define SEGD_E   (SEGM_A|SEGM_D|SEGM_E|SEGM_F|SEGM_G)
#define SEGD_F   (SEGM_A|SEGM_E|SEGM_F|SEGM_G)
```

**Usage example**

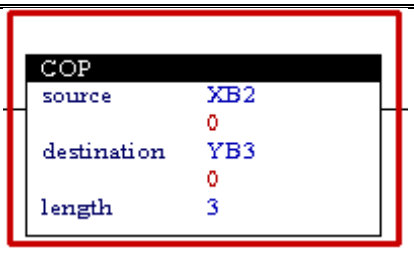
SEG(XB3,RW3): Save the 7' Segment value that correspond to the value of XB3 into RW3.
If (XB3)=(&H17), save &H0607 that combined the above
SEGD_1(SEGM_B|SEGM_C=0x02|0x04=0x06)=&H06 and
SEGD_7(SEGM_A|SEGM_B|SEGM_C=0x01|0x02|0x04=0x07)=&H07 into RW3.

## 6.25. MOV(Move): Move

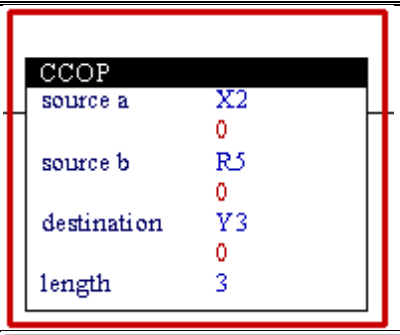
| | |
|---|---|
| Symbol |  |
| Relay combination |  |
| Factor | src: Source relay (or Number)<br>dst: Destination relay |
| Description | If Rung is activated, copy the value of scr into dst.<br>If src is word(W) type, and dst is byte(B) type, only the lower byte of src value will be copied to dst.<br>Also, all embedded PLC data is processed as a data with symbols. If src is byte(B) type, and the value is -1(&Hff), it will be copied to -1(&HFFFF) of dst. (&H00ff will be the value of 255). |
| Usage example | MOV(BX3,RB2): Copy the value of XB3 into RB2.<br>MOV(55,RB2): Copy 55 into RB2.<br>MOV(-30,RB2): Copy -30 into RB2. |

| inst | relay type / arg. | Input X, DO n | Input X, DO (B,W,L,F) m | Output Y,DI,R,K,SP n | Output Y,DI,R,K,SP (B,W,L,F) m | Timer T n | Timer T (B,W,L,F) m | Counter C n | Counter C (B,W,L,F) m | Memory M, S n | Memory M, S (B,W,L,F) m | const (32bit) (L/F) |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| MOV | src | X | | X | | X | | X | | X | | |
| | dst | X | X | X | | X | | X | | X | | X |

## 6.26. COP(Copy data): Copy

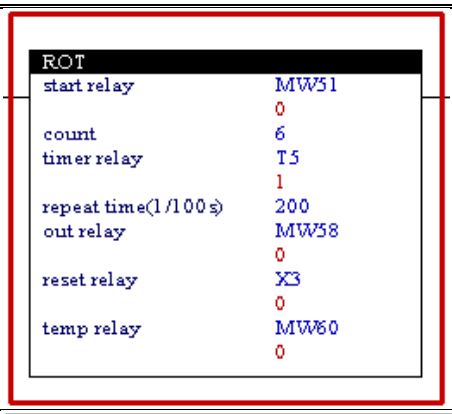
<table>
<tr><td>Symbol</td><td> 
</td></tr>
<tr><td>Relay combination</td><td>
<table>
<tr><th rowspan="3">inst</th><th>relay</th><th colspan="2">Input</th><th colspan="2">Output</th><th colspan="2">Timer</th><th colspan="2">Counter</th><th colspan="2">Memory</th><th rowspan="2">const<br>(32bit)<br>(L/F)</th></tr>
<tr><th>type</th><th colspan="2">X, DO</th><th colspan="2">Y,DI,R,K,SP</th><th colspan="2">T</th><th colspan="2">C</th><th colspan="2">M, S</th></tr>
<tr><th>arg.</th><th>n</th><th>(B,W,L,F) m</th><th>n</th><th>(B,W,L,F) m</th><th>n</th><th>(B,W,L,F) m</th><th>n</th><th>(B,W,L,F) m</th><th>n</th><th>(B,W,L,F) m</th></tr>
<tr><td rowspan="3">COP</td><td>src</td><td></td><td></td><td></td><td></td><td>X</td><td></td><td>X</td><td></td><td></td><td></td><td></td></tr>
<tr><td>dst</td><td>X</td><td>X</td><td></td><td></td><td>X</td><td></td><td>X</td><td></td><td></td><td></td><td>X</td></tr>
<tr><td>len</td><td>X</td><td></td><td>X</td><td></td><td>X</td><td></td><td>X</td><td></td><td>X</td><td></td><td></td></tr>
</table>
</td></tr>
<tr><td>Factor</td><td>src: Source relay or value to be filled with (const).<br>dst: Destination relay<br>len: Length relay (or Number).</td></tr>
<tr><td>Description</td><td>If Rung is activated, copy the values from src to dst as many as the number of len.<br>If src is a number, src value will be filled as many as the number of len from dst. In this case, when the dst is bit type, if src number is 0, OFF or else ON will be filled. If src is relay, the data type of src and dst should match.<br>For example, if src is bit, dst should be bit, if src is byte(B), dst should be byte(B), if src is word(W) then dst should be word(w) as well.<br>If src+len is greater than the maximum number of src relay, or if dst+len is greater than the maximum number of dst relay, copy will only progress until the maximum number of relay.</td></tr>
<tr><td>Usage example</td><td>Example for designated number of copy)<br>COP(X2,Y3,4): Y3=X2, Y4=X3, Y5=X4, Y6=X5<br>COP(XB2,YB3,3): YB3=XB2, YB4=XB3, YB5=XB4<br>COP(XW2,YW3,2): YW3=XW2, YW4=XW3<br><br>Example for filling up with the designated value(const))<br>COP(0,Y3,4): Y3=OFF, Y4=OFF, Y5=OFF, Y6=OFF<br>COP(1,Y3,4): Y3=ON, Y4=ON, Y5=ON, Y6=ON<br>COP(25,Y3,4): Y3=ON, Y4=ON, Y5=ON, Y6=ON<br>COP(25,YB3,3): YB3=25, YB4=25, YB5=25<br>COP(&H55AA,YW3,2): YW3=&H55AA, YW4=&H55AA<br>COP(0,MW3,50): MW3=0, MW4=0, MW5=0, MW6=0, ~, MW52=0</td></tr>
</table>

## 6.27. CCOP(Conditional Copy data): Conditional Copy

<table>
<tr><td>Symbol</td><td> 
</td></tr>
<tr><td>Relay combination</td><td>
<table>
<tr><th rowspan="3">inst</th><th>relay</th><th colspan="2">Input</th><th colspan="2">Output</th><th colspan="2">Timer</th><th colspan="2">Counter</th><th colspan="2">Memory</th><th rowspan="2">const<br>(32bit)<br>(L/F)</th></tr>
<tr><th>type</th><th colspan="2">X, DO</th><th colspan="2">Y,DI,R,K,SP</th><th colspan="2">T</th><th colspan="2">C</th><th colspan="2">M, S</th></tr>
<tr><th>arg.</th><th>n</th><th>(B,W,L,F) m</th><th>n</th><th>(B,W,L,F) m</th><th>n</th><th>(B,W,L,F) m</th><th>n</th><th>(B,W,L,F) m</th><th>n</th><th>(B,W,L,F) m</th></tr>
<tr><td rowspan="4">CCOP</td><td>sA</td><td></td><td></td><td></td><td></td><td>X</td><td></td><td>X</td><td></td><td></td><td></td><td></td></tr>
<tr><td>sB</td><td></td><td></td><td></td><td></td><td>X</td><td></td><td>X</td><td></td><td></td><td></td><td></td></tr>
<tr><td>dst</td><td>X</td><td>X</td><td></td><td></td><td>X</td><td></td><td>X</td><td></td><td></td><td></td><td>X</td></tr>
<tr><td>len</td><td>X</td><td></td><td>X</td><td></td><td>X</td><td></td><td>X</td><td></td><td>X</td><td></td><td></td></tr>
</table>
</td></tr>
<tr><td>Factor</td><td>sA: When Rung is activated, source relay to copy, or value to be filled with (const)<br>sB: When Rung is activated, source relay to copy, or value to be filled with (const)<br>dst: Destination relay<br>len: Length relay (or Number).</td></tr>
<tr><td>Description</td><td>Depends on Rung status, copy the values from sA or sB to dst as many as the number of len.<br>If src is a number, sA/sB value will be filled as many as the number of len from dst. In this case, when the dst is bit type, if sA/sB number is 0, OFF or else ON will be filled.<br>If sA/sB is relay, the data type of sA/sB and dst should match.<br>For example, if sA/sB is bit, dst should be bit, if sA/sB is byte(B), dst should be byte(B), if sA/sB is word(W) then dst should be word(w) as well.<br>If (sA/sB)+len is greater than the maximum number of sA/sB relay, or if dst+len is greater than the maximum number of dst relay, copy will only progress until the maximum number of relay.</td></tr>
<tr><td>Usage example</td><td>Example for designated number of copy)<br>CCOP(X2,R5,Y3,3): If Rung is activated, Y3=X2, Y4=X3, Y5=X4<br>If Rung is deactivated, Y3=R5, Y4=R6, Y5=R7<br>CCOP(1,0,Y3,3): If Rung is activated, Y3=ON, Y4=ON, Y5=ON<br>If Rung is deactivated, Y3=OFF, Y4=OFF, Y5=OFF<br>Reference)<br>Please refer to COP command as the examples of factor usage is similar to COP command.</td></tr>
</table>

## 6.28. ROT(Rotating Output): Rotating Output

**Symbol**




**Relay combination**

| inst | relay type / arg. | Input X, DO n | Input X, DO (B,W,L,F) m | Output Y,DI,R,K,SP n | Output Y,DI,R,K,SP (B,W,L,F) m | Timer T n | Timer T (B,W,L,F) m | Counter C n | Counter C (B,W,L,F) m | Memory M, S n | Memory M, S (B,W,L,F) m | const (32bit) (L/F) |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ROT | st | X | | X | | X | | X | | X | | X |
| | cnt | X | | X | X | X | | X | | X | | |
| | tm | X | X | X | X | | X | X | X | X | X | X |
| | rep | X | | X | | X | | X | | X | | |
| | out | X | X | X | | X | X | X | X | X | | X |
| | rst | | X | | X | | X | | X | X | X | X |
| | tmp | X | X | X | | X | X | X | X | X | | X |

**Factor**

st: Start relay
cnt: Count relay (or Number).
tm: Timer relay(1/100sec unit).
rep: Relay that designates the repeat time (or Number).
out: Relay that outputs the result.
rst: Relay that resets the output status.
tmp:Relay that indicates which value to output for the operation.

**Description**

If Rung is activated, the relay value which is not 0, between the range of st and cnt will repeatedly output to out relay for rep time.
If a signal input to rst relay, fill 0 from st relay as many number as cnt and initialize the timer value with rep value. Also 0 will be the output for out relay.
This command can be used conveniently in order to output the occurred error numbers within the designated time in case when there are many types of errors for only one device to output the error number.

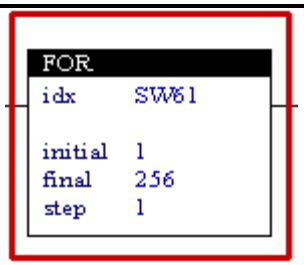
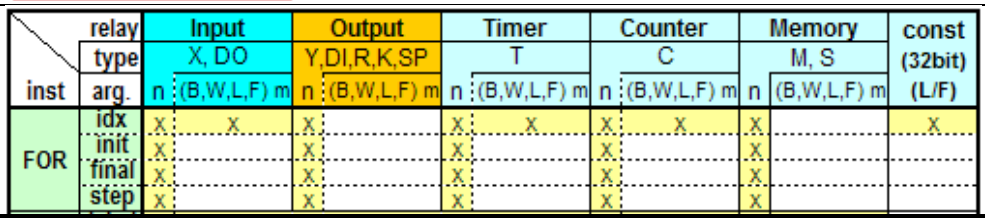
**Usage example**

```
|-------( Error condition 1)----------[MOV(21,MW51)]---|
|-------( Error condition 2)----------[MOV(22,MW52)]---|
|-------( Error condition 3)----------[MOV(23,MW53)]---|
|-------( Error condition 4)----------[MOV(24,MW54)]---|
|-------( Error condition 5)----------[MOV(25,MW55)]---|
|-------( Error condition 6)----------[MOV(26,MW56)]---|
|----------[ROT(MW51,6,T5,200,MW58,X3,MW60)]---|
|----------------------------------------[TOD(MW58,YB3)]---|
```

Above logic saves the error number into MW51~MW56, if more than one error among the 1~6 error conditions occurs. With the ROT command, the occurred error number will be saved in MW58 for 2 seconds and it will be converted to BCD value by TOD command to be displayed rotationally on a output device that is connected to YB3.

| | |
|---|---|
| | If a signal is entered to X3 that is connected to the external error reset, the contents of MW51~MW56 which stored the error number will be cleared to 0, as well as the MW58 and MW60. 0 will be displayed on output device. |

## 6.29. FOR(FOR): Block Repeat

| | |
|---|---|
| Symbol |  |
| Relay combination |  |
| Factor | idx: Index relay.<br>Init: Initial value relay or number.<br>final: Final value relay or number<br>step: Relay or number that will be used as step. |
| Description | If Rung is activated, repeat the block from init to final by increasing the idx relay value with the value of step, until it reaches to NEXT.<br>When FOR clause is executed, enter the init value to the idx.<br>FOR/NEXT clause can be nested up to 10 times.<br>→ FOR() FOR() FOR() ….NEXT NEXT NEXT.<br>If step〉0, and init〉final, then do not execute and jump to NEXT.<br>If step〈0, and init〈final, then do not execute and jump to NEXT.<br>Final and step only uses the initial value from the beginning of for clause, even if they are assigned as variable.<br>In order to exit from the middle of FOR clause exceptionally, you may use JMP(negative number) command. (Refer to JMP command description).<br>Caution) FOR command do not provide additional process for branches |
| Usage example | FOR(RB10,1,4,1): Repeatedly execute until it reaches to NEXT command from RB10=1 to RB10=4 by increasing 1 each time.<br>{\|--[**FOR(SW61,1,256,1)**]-—[XIC(X-1)]—-[OTL(Y-1)]—-[**NEXT**]--\|}: Repeatedly execute {XIC(X-1), OTL(Y-1)} command from SW61=1 to SW61=256 by increasing 1 each time.<br>Among X1~X256, only the Y relay number that correspond to High signal number will be output and Y output of not entered number will retain the previous status. It is because of that idx is using a relay for indirect addressing (SW61~SW69) and the X relay of XIC and the Y relay of OTL is “-1” which will apply the number of SW61 value.<br>Reference) Indirect addressing refers to as below. For any types of relay, if the address is set to -1 ~ -9, the value that saved in SW61~SW69 will be designated for the relay address. |

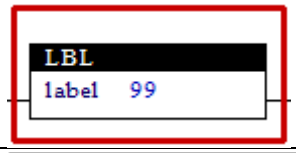
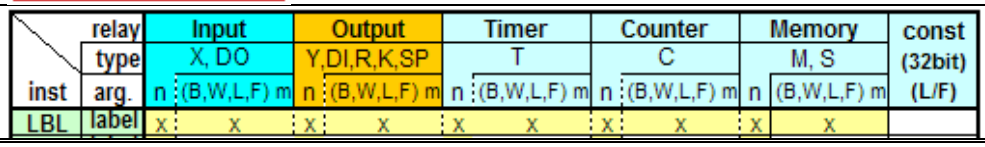
| relay | Input | | Output | | Timer | | Counter | | Memory | | const |
|---|---|---|---|---|---|---|---|---|---|---|---|
| type | X, DO | | Y,DI,R,K,SP | | T | | C | | M, S | | (32bit) |
| inst / arg. | n | (B,W,L,F) m | n | (B,W,L,F) m | n | (B,W,L,F) m | n | (B,W,L,F) m | n | (B,W,L,F) m | (L/F) |
| FOR idx | X | X | X | | X | X | X | X | X | | X |
| FOR init | X | | X | | X | | X | | X | | |
| FOR final | X | | X | | X | | X | | X | | |
| FOR step | X | | X | | X | | X | | X | | |

## 6.30. NEXT(NEXT): Block Next

| | |
|---|---|
| Symbol |  |
| Factor | None |
| Description | It will operate as below according to the step of FOR command.<br>If step>0, it will be repeatedly executed until the value of inx relay is less than, or equal to the value of final.<br>If step<0, it will be repeatedly executed until the value of inx relay is greater than, or equal to the value of final.<br>If NEXT clause is executed without FOR clause, NEXT command will be ignored.<br>Caution)<br>Since the FOR/NEXT commands do not provide additional process for branches, if FOR command is written in one branch and if the NEXT is written in outside of the branch, or in another branch, the FOR command will not operate properly. |
| Usage Example | Please refer to the usage example of FOR command. |



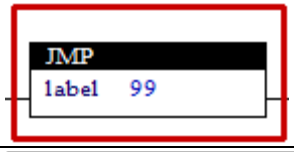
## 6.31. LBL(Label): Label Designation

| | |
|---|---|
| Symbol |  |
| Relay Combination |  |

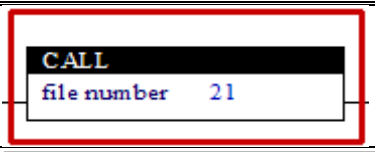
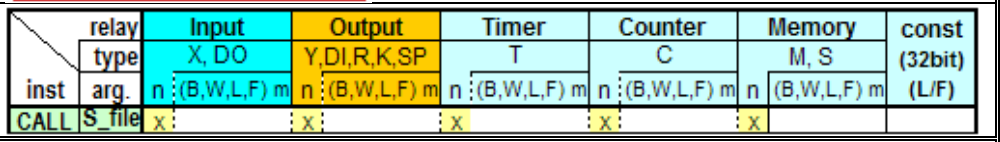
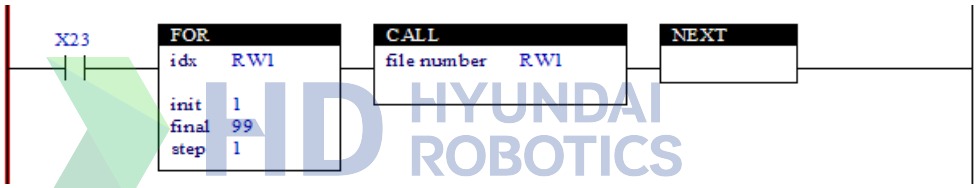
| inst \ relay | relay | Input | Output | Timer | Counter | Memory | const (32bit) (L/F) |
|---|---|---|---|---|---|---|---|
| | type | X, DO | Y,DI,R,K,SP | T | C | M, S | |
| inst | arg. | n (B,W,L,F) m | n (B,W,L,F) m | n (B,W,L,F) m | n (B,W,L,F) m | n (B,W,L,F) m | |
| LBL | label | x x | x x | x x | x x | x x | |

| | |
|---|---|
| Factor | label: Designate label No. or a value (const) greater than 0. |
| Description | Designate a location of label to jump with JMP command<br>LBL command will be designated the location regardless of activated/deactivated status.<br>In the case when there is an identical number of LBL command in one file. |
| Usage Example | Please refer to the JMP command description, because it will be used with JMP command. |

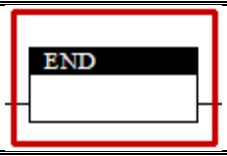
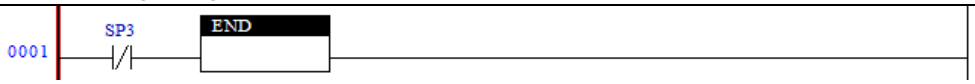
## 6.32. JMP(Jump): Jump

| | |
|---|---|
| Symbol |  |
| Relay Combination | <table><tr><td rowspan="2">inst</td><td>relay</td><td colspan="2">Input</td><td colspan="2">Output</td><td colspan="2">Timer</td><td colspan="2">Counter</td><td colspan="2">Memory</td><td rowspan="2">const (32bit) (L/F)</td></tr><tr><td>type</td><td colspan="2">X, DO</td><td colspan="2">Y,DI,R,K,SP</td><td colspan="2">T</td><td colspan="2">C</td><td colspan="2">M, S</td></tr><tr><td>JMP</td><td>arg.<br>label</td><td>n (B,W,L,F) m<br>x</td><td></td><td>n (B,W,L,F) m<br>x</td><td></td><td>n (B,W,L,F) m<br>x</td><td></td><td>n (B,W,L,F) m<br>x</td><td></td><td>n (B,W,L,F) m</td><td></td></tr></table> |
| Factor | label: Numeral(const) that is Label No. or relay |
| Description | If Rung is activated, it will jump to the location that has LBL command, which is identical with the value that is designated by label.<br>It can be used as a function to exit from the middle of FOR clause exceptionally when label<0. (skip as the number of times designated as negative number)<br>Caution1)<br>In case when a location of label is upper than the JMP command and there is no condition before the JMP command, it can be fall into an infinite loop.<br>In this case, scan time will exceed 5 seconds so it will set SP17=On.<br>Caution2)<br>In the case when exit from the FOR/NEXT block by using JMP(positive number), block control may cause an error, as there is no additional consideration. In this case, proceed to skip until the NEXT command by using JMP(negative number). |
| Usage Example | {\|--[XIC(X21)]---[**JMP(25)**]---[·····]---[**LBL(25)**]--\|}: If X21 relay is 1, it will jump to the location where LBL(25) is by JMP(25)command. Even if LBL command is located in other Rung, it will operate normally. |

## 6.33. CALL(Call): Call Sub-ladder

| | |
|---|---|
| Symbol |  |
| Relay Combination |  |
| Factor | file number: Relay or number that designates Sub-ladder's file No.(1~99) |
| Description | If Rung is activated, it will call Sub-ladder with the number that is designated by file number.<br>File names of Sub-Ladder can exist maximum of 99 between S01xxxx.LAD ~ S99xxxx.LAD, 'xxxx' in the file name can be named by a user, and it can be up to 15 characters.. |
| Usage Example |  <br>If X23 signal is being received, the files from S01xxxx.LAD to S99xxxx.LAD will be called rotationally.<br>From the result of CALL command's execution, if the Sub-Ladder of corresponding number does not exist, or if it exceeds the range of 1 ~ 99, it will set SP18=On. If it is executed normally, it will set SP18=Off. Therefore, it is able to detect error by using SP18 after calling in case when Sub-Ladder needs to exist..<br>Arrange to use CALL command to call the Sub-ladder of 1 ~ 99 from the Main-ladder, and assign the Sub-ladder number for each applications. By doing that, and load only necessary Sub-ladder according to the application with the Controller, you can expect an effect that the corresponding ladder will be executed automatically. |

## 6.34. END(End): Ladder End

| | |
|---|---|
| Symbol |  |
| Factor | None |
| Description | If Rung is activated, terminate the currently processing ladder.<br>If the currently processing ladder is a Sub-ladder, it will return to the Main-ladder. If the currently processing ladder is Main-ladder, terminate the process and restart it from the beginning. |
| Usage Example |  <br>If initialization process is required, compose the initialization process as S01 Initialize.LAD file and arrange it to call No.1 from the Main-Ladder. Then add it to the first Rung in S01. Since SP3 is a relay that only goes On during the one scan of the beginning of operation, S01 will only be executed once. From the next scan, it will be returned from the first Rung. |

● **Daegu Office (Head Office)**

50, Techno sunhwan-ro 3-gil, yuga, Dalseong-gun, Daegu, 43022, Korea

● **GRC**

477, Bundangsuseo-ro, Bundang-gu, Seongnam-si, Gyeonggi-do, 13553, Korea

● **대구 사무소**

(43022) 대구광역시 달성군 유가읍 테크노순환로 3 길 50

● **GRC**

(13553) 경기도 성남시 분당구 분당수서로 477

● ARS : +82-1588-9997 (A/S center)

● E-mail : robotics@hyundai-robotics.com